Benchmark Testing the α-models of Turbulence

A Project
Presented to
the Graduate School of
Clemson University

In Partial Fulfillment
of the Requirements for the Degree
Master of Science
Mathematical Sciences

---

by
Ryan Gregory Hill
May 2010

Accepted by:
Dr. Leo Rebholz, Committee Chair
Dr. Vincent Ervin
Dr. Christopher Cox

Abstract

In this analytical and computational study we explore four promising models of turbulent flow, known as the α-models of turbulence. These models arise as a filtering process on the Navier-Stokes equations. Although these four models have seemingly minor differences, we show that these difference lead to major differences in conservation laws, and thus in physical relevance of solutions, efficiency in implementation, and most importantly in the accuracy of computed solutions.

Dedication

To my fianceé Kristin, who hung the moon.

Acknowledgments

Two years ago, I made the choice to pursue a graduate degree in mathematics. This choice was made partly to my strong desire to avoid entering the real world for as long as possible. Most importantly, this choice was made because of the great people who encouraged me, taught me, and guided me.

First I would like to thank my parents. You have always encouraged me to do my best. You have always been there to celebrate my successes, but also there to teach me to accept success with grace and humility. I will never forget from where I have come.

I would like to thank my fianceé Kristin. Thank you for never letting me accept anything less than my best. Your support and encouragement has sustained me these last two years, and I look forward to the countless years ahead of us.

I am grateful for the professors of the mathematics department of Wofford College, Matt Cathey, Lee Hagglund, Charlotte Knotts-Zides, and Ted Monroe for four years of cultivating in me a love of mathematics. My excited curiosity was always met with patience and wisdom. I would not be where I am today with out your guidance and friendship.

Finally I owe this project to my advisor, Leo Rebholz. Your passion for what you do has inspired me, and I am grateful for you sharing with me that excitement for discovery. Thank you for your encouragement and guidance through these last two years.

Table of Contents

List of Figures

Chapter 1

Introduction

The mathematical facts worthy of being studied are those which, by their analogy with other facts, are capable of leading us to the knowledge of a physical law. They reveal the kinship between other facts, long known, but wrongly believed to be strangers to one another. (H. Poincaré, in N. Rose, *Mathematical Maxims and Minims*, Rome Press, Raleigh, NC, 1988)

Turbulent flows are all around us. They are even in us! From airplanes in flight to blood flowing through our veins. From cars traveling down the highway to oil rushing through pipes. We are pervaded by the presence of turbulence. It is no wonder, then, why engineers, scientists, and mathematicians study this phenomenon.

The study of turbulence is very difficult because turbulence is very complex. It is "diffusive, chaotic and irregular, three dimensional and rotational, highly dissipative, is a continuum phenomena, and the vortex stretching mechanism generates small scales." [15] Such a complex physical phenomenon would certainly benefit from computational simulation, however turbulence is not very well understood, and thus simulating it is not fully developed.

The Navier-Stokes equations (NSE) completely describe the flow of incompressible, viscous fluids. These equations, built from physical principles, are given by

$$\begin{aligned} \mathbf{u}_t + \mathbf{u} \cdot \nabla \mathbf{u} + \nabla p - \nu \Delta \mathbf{u} &= \mathbf{f} \\ \nabla \cdot \mathbf{u} &= 0 \end{aligned} \tag{1.1}$$

Unfortunately, the breadth of information contained in the NSE is so complex that finding a

closed form solution in three dimensions is worth one million dollars as part of the Clay Mathematics Institute's Millennium Prize problems. In fact, just showing (1.1) is well-posed is worth one million dollars. Even computationally, it is extremely expensive, and thus highly impractical, to obtain accurate numerical results! In order to fully resolve a flow computationally using the NSE, we must utilize a very large number of mesh points, usually on the order of $\mathrm{Re}^{9/4}$. In many instances, the Reynolds number is at least in the thousands (Reynolds number for a car is about 10^6 and for a airplane about 10^7). This can be quite a challenge, even on the most powerful computers.

To overcome this problem, we turn to models. The question then becomes which model to choose out of the many in existence. If none can be deemed most appropriate, how does one formulate a new model? How do we evaluate the model? Does the model accurately describe important properties that characterize the flow? How do the solutions given by the model compare to those given by the Navier-Stokes equations? These and many other questions are often difficult to answer.

This thesis brings together and formulates new results about the α-models of turbulence, which are recently developed models that have shown promising preliminary results. We perform analytical as well as computational studies of these models. We find that allthough the models' respective equations appear similar, they admit major differences in conservation laws, and thus in physical relevance. These differences directly lead to different implementations, and thus result in a variety of computed solutions. These models will serve as stepping stones to developing faster and cheaper algorithms while maintaining accuracy and physical relevance.

We first study the conservation laws of the four α-models of turbulent flow: the Leray-α model, the Modified Leray-α model, the Navier-Stokes-alpha (NS-α) model, and the Navier-Stokes-omega (NS-ω) model. These models are called the α-models because they all employ the α-filter, denoted by $\bar{\cdot}$. This filter satisfies for $\phi \in L^2(\Omega)$ and filtering radius $\alpha > 0$, $\overline{\phi}$ satisfies

$$-\alpha^2 \Delta \overline{\phi} + \overline{\phi} = \phi. \tag{1.2}$$

We formally define these models below, and provide citations for the first fundamental works on the models where their well-posedness is proven.

- Leray-α [2]

$$\mathbf{u}_t + \bar{\mathbf{u}} \cdot \nabla \mathbf{u} + \nabla p - \nu \Delta \mathbf{u} = \mathbf{f}, \nabla \cdot \mathbf{u} = 0 \tag{1.3}$$

- Modified Leray-α [1]

$$\mathbf{u}_t + \mathbf{u} \cdot \nabla \bar{\mathbf{u}} + \nabla p - \nu \Delta \mathbf{u} = \mathbf{f}, \nabla \cdot \bar{\mathbf{u}} = 0 \tag{1.4}$$

- NS-α [3, 4]

$$\mathbf{u}_t - \bar{\mathbf{u}} \times (\nabla \times \mathbf{u}) + \nabla p - \nu \Delta \mathbf{u} = \mathbf{f}, \nabla \cdot \bar{\mathbf{u}} = 0 \tag{1.5}$$

- NS-ω [9, 16]

$$\mathbf{u}_t - \mathbf{u} \times (\nabla \times \bar{\mathbf{u}}) + \nabla p - \nu \Delta \mathbf{u} = \mathbf{f}, \nabla \cdot \mathbf{u} = 0 \tag{1.6}$$

We discover how these models may or may not exactly conserve energy $\left(E = \int_\Omega |u|\, dx\right)$, two dimensional enstrophy $\left(Ens = \frac{1}{2} \int_\Omega |\nabla \times u|^2\, dx\right)$, and helicity $\left(H = \int_\Omega u \cdot (\nabla \times u) dx\right)$. Enstrophy and helicity are rotational quantities. Enstrophy is a measure of the strength of the vortices formed in turblent flow. It is only conserved in two dimensional flows due to vortex stretching, which is a three dimensional quantity. Helicity has a physical interpretation of the degree to which the vortex lines coil around each other. Each of these quantities is conserved in the NSE, and their accurate treatment in a model critical is considered critical. As we will see, choosing an appropriate multiplier (to make the non-linear terms in our models vanish) is key to proving these laws.

A model that does not exactly conserve enstrophy (in two dimensions) or helicity (in three dimensions) will not accurately reflect the physics of rotational forms in the fluid. This is because the models create or dissipate helicity, instead of cascading it from large scales (where it is input) to small scales (where viscosity takes over) [13]. Thus solutions to such models can lack physical meaning, especially over long time intervals.

The Leray-α model was proposed in [2] as a more computable alternative to Leray's original 1934 model [10]. As we will show, this model conserves energy and (2d) enstrophy, but no law for helicity could be found (and we believe one does not exist). Thus three dimensional rotational quantities may not be properly treated by this model. Conservation laws for two dimensional enstrophy and helicity for the modified Leray-α model could not be found either, so we would expect the rotational forms in this model will not be represented accurately. In fact, our simulations show that this appears to be an inadequate model. The NS-α model conserves at least model quantities of energy, enstrophy, and helicity. NS-ω also conserves at least model quantities of energy, enstrophy, and helicity. The NS-α and NS-ω models thus appear to be the most physically relevant of the models we consider, and thus we believe should be the focus of future research.

Our computational study of these models is a comparison of their behavior on the benchmark test of two dimensional flow over a step. To be successful, a model's computed solution must be comparable to the NSE solution, but in a more efficient way than the NSE could give when computed directly. For numerical schemes, we discretize each model with a trapezoidal time discretization and the Galerkin finite element method in space. Here we find of the four models, the rotational form models, with carefully chosen discretization, perform the best. The Leray-α model performs well, and Modified Leray-α gives poor results.

This thesis is arranged as follows. Chapter 2 states the assumptions, definitions, and lemmas needed throughout the paper. In chapter 3 we present the conservation laws of the models. Chapter 4 considers the numerical schemes used to compute the solutions to the models. We then implement these models to solve the problem of flow over a step and present these results. Finally, in Chapter 5 we compare the results and state conclusions.

Chapter 2

Preliminaries

2.1 Assumptions, Definitions, and Lemmas

We assume throughout this paper that the domain Ω is the periodic box $(0, L)^d$, $d = 2$ or 3, however with minor modifications our results are extendable to polygonal or polyhedral domains with homogeneous Dirichlet boundary conditions. We shall also assume smooth data, $\mathbf{f}$ and $\mathbf{u}_0$, and thus in the proofs of the conservation laws, the solutions are smooth enough to justify each manipulation [4, 2, 1, 16].

The usual L^2 inner product and norm, denoted $(\cdot, \cdot)$ and $\|\cdot\|$ respectively, will used extensively throughout:

$$(\mathbf{u}, \mathbf{v}) := \int_\Omega \mathbf{u} \cdot \mathbf{v} dx, \qquad \|\mathbf{u}\| := (\mathbf{u}, \mathbf{u})^{1/2}.$$

Definition 1 *The Helmholtz filter (α-filter)*
Given $\phi \in L^2(\Omega)$ and filtering radius $\alpha > 0$, we define

$$\bar{\phi} := (-\alpha^2 \Delta + I)^{-1} \phi \tag{2.1}$$

We use the following vector identities throughout the paper.

Lemma 2 *Vector Identities*

1. *For sufficiently smooth and periodic* $\mathbf{u}, \mathbf{v}$*,*

$$(\mathbf{u}, \Delta \mathbf{v}) = -(\nabla \times \mathbf{u}, \nabla \times \mathbf{v}) \tag{2.2}$$

and if $\nabla \cdot \mathbf{v} = 0$*,*

$$(\mathbf{u}, \Delta \mathbf{v}) = -(\nabla \mathbf{u}, \nabla \mathbf{v}) \tag{2.3}$$

2. *If* $d = 2$*, then for sufficiently smooth, periodic, and divergence free* $\mathbf{u}$*,*

$$(\mathbf{u} \cdot \nabla \mathbf{u}, \Delta \mathbf{u}) = 0 \tag{2.4}$$

Proof: The proof of (2.2)-(2.3) can be found, e.g., in [8]. The proof of (2.4) can be found, e.g., in [5]. ■

Lemma 3 *If* $\mathbf{u}, \mathbf{v}$ *are periodic in* $H^1(\Omega)$*, then* $(\mathbf{u} \cdot \nabla \mathbf{v}, \mathbf{v}) = 0$ *provided* $\nabla \cdot \mathbf{u} = 0$*.*

Proof: This follows from integrating by parts, using $\nabla \cdot \mathbf{u} = 0$, and that $\mathbf{u}$ and $\mathbf{v}$ are periodic in Ω [6]. ■

Lemma 4 *Under periodic boundary conditions,* $\nabla \cdot \mathbf{u} = 0$ *if and only if* $\nabla \cdot \bar{\mathbf{u}} = 0$*.*

Proof: Using the differential filter and the symmetry of mixed partials, we see that

$$\begin{aligned}
\nabla \cdot \mathbf{u} &= \nabla \cdot (-\alpha^2 \Delta + I)(-\alpha^2 \Delta + I)^{-1} \mathbf{u} \\
&= \nabla \cdot (-\alpha^2 \Delta + I) \bar{\mathbf{u}} \\
&= (-\alpha^2 \Delta + I)(\nabla \cdot \bar{\mathbf{u}}) \\
&= -\alpha^2 \Delta (\nabla \cdot \bar{\mathbf{u}}) + (\nabla \cdot \bar{\mathbf{u}})
\end{aligned} \tag{2.5}$$

Hence we see $\nabla \cdot \bar{\mathbf{u}} = 0$ implies $\nabla \cdot \mathbf{u} = 0$. Also, multiplying (2.5) by $(\nabla \cdot \bar{\mathbf{u}})$ and integrating gives

$$\alpha^2 \left\| \nabla (\nabla \cdot \bar{\mathbf{u}}) \right\|^2 + \left\| \nabla \cdot \bar{\mathbf{u}} \right\|^2 = (\nabla \cdot \mathbf{u}, \nabla \cdot \bar{\mathbf{u}}) \leq \frac{1}{2} \left\| \nabla \cdot \mathbf{u} \right\|^2 + \frac{1}{2} \left\| \nabla \cdot \bar{\mathbf{u}} \right\|^2$$

Thus it follows that $\|\nabla \cdot \bar{\mathbf{u}}\| \leq \|\nabla \cdot \mathbf{u}\|$. Hence $\nabla \cdot \mathbf{u} = 0$ implies $\nabla \cdot \bar{\mathbf{u}} = 0$, and the proof is complete.

■

Lemma 5 *Under periodic boundary conditions, the curl is self-adjoint. That is*

$$(\nabla \times \nabla \times \mathbf{u}, \mathbf{v}) = (\nabla \times \mathbf{u}, \nabla \times \mathbf{v})$$

Chapter 3

Conservation Laws

3.1 Energy Conservation

We begin by proving energy conservation for the models. As we will show, an exact energy balance is not always attained for usual energy; for NS-α and Modified Leray-α, a model energy is conserved.

Theorem 6 *Let $\mathbf{u}_L, \mathbf{u}_{ML}, \mathbf{u}_\alpha, \mathbf{u}_\omega$ be the velocity solutions of equations (1.3), (1.4), (1.5), and (1.6), respectively. Then*

- *Leray-α conserves energy [10]:*

$$\frac{1}{2}\left\|\mathbf{u}_L(T)\right\|^2 + \nu\int_0^T \left\|\nabla\mathbf{u}_L(t)\right\|^2 dt = \frac{1}{2}\left\|\mathbf{u}_L(0)\right\|^2 + \int_0^T (\mathbf{f}(t), \mathbf{u}_L(t))dt \tag{3.1}$$

- *Modified Leray-α conserves a model energy:*

$$\begin{aligned}&\frac{1}{2}\left(\left\|\overline{\mathbf{u}}_{ML}(T)\right\|^2 + \alpha^2\left\|\nabla\overline{\mathbf{u}}_{ML}(T)\right\|^2\right) + \nu\int_0^T \left(\left\|\nabla\overline{\mathbf{u}}_{ML}(t)\right\|^2 + \alpha^2\left\|\Delta\overline{\mathbf{u}}_{ML}(t)\right\|^2\right) dt \\ &\quad = \frac{1}{2}\left(\left\|\overline{\mathbf{u}}_{ML}(0)\right\|^2 + \alpha^2\left\|\nabla\overline{\mathbf{u}}_{ML}(0)\right\|^2\right) + \int_0^T (\mathbf{f}(t), \overline{\mathbf{u}}_{ML}(t))\, dt\end{aligned} \tag{3.2}$$

- *NS-α conserves a model energy [4]:*

$$\begin{aligned}&\frac{1}{2}\left(\|\overline{\mathbf{u}}_\alpha(T)\|^2+\alpha^2\|\nabla\overline{\mathbf{u}}_\alpha(T)\|^2\right)+\nu\int_0^T\left(\|\nabla\overline{\mathbf{u}}_\alpha(t)\|^2+\alpha^2\|\Delta\overline{\mathbf{u}}_\alpha(T)\|^2\right)dt\\&=\frac{1}{2}\left(\|\overline{\mathbf{u}}_\alpha(T)\|^2+\alpha^2\|\nabla\overline{\mathbf{u}}_\alpha(T)\|^2\right)+\int_0^T(\mathbf{f}(t),\overline{\mathbf{u}}_\alpha(t))\,dt\end{aligned}\tag{3.3}$$

- *NS-ω conserves energy [9]:*

$$\frac{1}{2}\|\mathbf{u}_\omega(T)\|^2+\nu\int_0^T\|\nabla\mathbf{u}_\omega(t)\|^2\,dt=\frac{1}{2}\|\mathbf{u}_\omega(0)\|^2+\int_0^T(\mathbf{f}(t),\mathbf{u}_\omega(t))\,dt\tag{3.4}$$

Proof: For Leray-α, multiply (1.3) by $\mathbf{u}$ and integrate over the domain to obtain:

$$(\mathbf{u}_t,\mathbf{u})+(\overline{\mathbf{u}}\cdot\nabla\mathbf{u},\mathbf{u})+(\nabla p,\mathbf{u})-\nu(\Delta\mathbf{u},\mathbf{u})=(\mathbf{f},\mathbf{u}).$$

The nonlinearity vanishes by Lemma 3 and Lemma (4). The pressure term vanishes after integrating by parts, since $\mathbf{u}$ and p are periodic and $\mathbf{u}$ is divergence free. Thus we have

$$(\mathbf{u}_t,\mathbf{u})-\nu(\Delta\mathbf{u},\mathbf{u})=(\mathbf{f},\mathbf{u}).$$

The time derivative and dissipation terms do not vanish. We rewrite the time derivative and use Lemma 2 to rewrite the dissipation to obtain

$$\frac{1}{2}\frac{d}{dt}\|\mathbf{u}\|^2+\nu\|\nabla\mathbf{u}\|^2=(\mathbf{f},\mathbf{u})$$

Integrating over time gives the desired result.

Next, we prove the energy balance for the Modified Leray-α model. We multiply (1.4) by $\overline{\mathbf{u}}$ and integrate over the domain to obtain:

$$(\mathbf{u}_t,\overline{\mathbf{u}})+(\mathbf{u}\cdot\nabla\overline{\mathbf{u}},\overline{\mathbf{u}})+(\nabla p,\overline{\mathbf{u}})-\nu(\Delta\mathbf{u},\overline{\mathbf{u}})=(\mathbf{f},\overline{\mathbf{u}})$$

The nonlinearity vanishes under Lemma 3. The pressure term vanishes as well, just as before: integration by parts and that $\overline{\mathbf{u}}$ and p are periodic and $\overline{\mathbf{u}}$ is divergence free. We now have

$$(\mathbf{u}_t, \overline{\mathbf{u}}) - \nu(\Delta \mathbf{u}, \overline{\mathbf{u}}) = (\mathbf{f}, \overline{\mathbf{u}})$$

The time derivative does not simply reduce as in the Leray-α case because of the filter. We expand and simplify using the the definition of the filter and Lemma 2:

$$\begin{aligned}(\mathbf{u}_t, \overline{\mathbf{u}}) &= \frac{1}{2}\frac{d}{dt}(-\alpha^2 \Delta \overline{\mathbf{u}} + \overline{\mathbf{u}}, \overline{\mathbf{u}}) \\ &= \frac{1}{2}\frac{d}{dt}\left[\alpha^2 \|\nabla \overline{\mathbf{u}}\|^2 + \|\overline{\mathbf{u}}\|^2\right]\end{aligned}$$

Similarly, the dissipation term does not vanish. Thus we expand the filter and simplify:

$$\begin{aligned}-\nu(\Delta \mathbf{u}, \overline{\mathbf{u}}) &= -\nu(\Delta(-\alpha^2 \Delta \overline{\mathbf{u}} + \overline{\mathbf{u}}), \overline{\mathbf{u}}) \\ &= -\nu\left[-\alpha^2(\Delta(\Delta \overline{\mathbf{u}}), \overline{\mathbf{u}}) + (\Delta \overline{\mathbf{u}}, \overline{\mathbf{u}})\right] \\ &= -\nu\left[\alpha^2(\nabla(\Delta \overline{\mathbf{u}}), \nabla \overline{\mathbf{u}}) - (\nabla \overline{\mathbf{u}}, \nabla \overline{\mathbf{u}})\right] && \text{by Lemma 4.1} \\ &= -\nu\left[-\alpha^2(\Delta \overline{\mathbf{u}}, \Delta \overline{\mathbf{u}}) - \|\nabla \overline{\mathbf{u}}\|^2\right] && \text{by Lemma 4.1} \\ &= \nu\left[\alpha^2 \|\Delta \overline{\mathbf{u}}\|^2 + \|\nabla \overline{\mathbf{u}}\|^2\right]\end{aligned}$$

Thus we now have

$$\frac{1}{2}\frac{d}{dt}\left[\alpha^2 \|\nabla \overline{\mathbf{u}}\|^2 + \|\overline{\mathbf{u}}\|^2\right] + \nu\left[\alpha^2 \|\Delta \overline{\mathbf{u}}\|^2 + \|\nabla \overline{\mathbf{u}}\|^2\right] = (\mathbf{f}, \overline{\mathbf{u}})$$

Integrating over time gives the desired result.

The proof for the energy conservation of NS-α follows exactly as the proof for Modified Leray-α, except for the nonlinear term $(\overline{\mathbf{u}} \times (\nabla \times \mathbf{u}), \overline{\mathbf{u}})$. This term vanishes because the cross product of two terms is perpendicular to each term. Similarly, NS-ω follows the proof of Leray-α, with the exception of the nonlinear term, which also vanishes. ■

3.2 Enstrophy Conservation

We now show the conservation of two dimensional model enstrophy. We note that the result for NS-α is known [14]. The results for Leray-α, Modified Leray-α, and NS-ω are new. We were unable to find an exact law for Modified Leray-α. To find an exact law, we would need a multiplier that causes the nonlinearity to vanish and the time step to simplify to some enstrophy-related quantity. Given properties of the Navier-Stokes equations and the vector identities, it would appear that $\Delta\mathbf{u}$ would be the appropriate choice, which gives only an approximate conservation law.

Theorem 7 *Let $\mathbf{u}_L, \mathbf{u}_{ML}, \mathbf{u}_\alpha, \mathbf{u}_\omega$ be the velocity solutions of equations (1.3), (1.4), (1.5), and (1.6), respectively. Then*

- *Leray-α conserves a model enstrophy:*

$$\begin{aligned}&\frac{1}{2}\left[\|\nabla\overline{\mathbf{u}}_L(T)\|^2+\alpha^2\|\Delta\overline{\mathbf{u}}_L(T)\|^2\right]+\nu\int_0^T\left(\|\Delta\overline{\mathbf{u}}_L(t)\|^2+\alpha^2\|\nabla\times\Delta\overline{\mathbf{u}}_L(t)\|^2\right)dt\\&=\frac{1}{2}\left[\|\nabla\overline{\mathbf{u}}_L(0)\|^2+\alpha^2\|\Delta\overline{\mathbf{u}}_L(0)\|^2\right]+\int_0^T(\mathbf{f}(t),\Delta\overline{\mathbf{u}}_L(t))\,dt\end{aligned}\tag{3.5}$$

- *Modified Leray-α approximately conserves model enstrophy:*

$$\begin{aligned}&\frac{1}{2}\|\nabla\overline{\mathbf{u}}_{ML}(T)\|^2-\alpha^4\int_0^T(\mathbf{u}_{ML}(t)\cdot\nabla(\Delta\overline{\mathbf{u}}_{ML}(t))),\Delta\Delta\overline{\mathbf{u}}_{ML}(t))\,dt\\&=\frac{1}{2}\|\nabla\mathbf{u}_{ML}\|^2+\int_0^T(\nabla\mathbf{f}(t),\nabla\mathbf{u}_{ML}(t))\,dt\end{aligned}\tag{3.6}$$

- *NS-α conserves enstrophy:*

$$\frac{1}{2}\|\nabla\mathbf{u}_\alpha(T)\|^2+\nu\int_0^T\|\Delta\mathbf{u}_\alpha(t)\|^2\,dt=\frac{1}{2}\|\nabla\mathbf{u}_\alpha(0)\|^2+\int_0^T(\nabla\mathbf{f}(t),\nabla\mathbf{u}_\alpha(t))dt\tag{3.7}$$

- *NS-ω conserves model enstrophy:*

$$\begin{aligned}&\frac{1}{2}\left[\|\nabla\overline{\mathbf{u}}_\omega(T)\|^2+\alpha^2\|\Delta\overline{\mathbf{u}}_\omega(T)\|^2\right]+\nu\int_0^T\left(\|\Delta\overline{\mathbf{u}}_\omega(t)\|^2+\|\nabla\times\Delta\overline{\mathbf{u}}_\omega(t)\|^2\right)dt\\&=\frac{1}{2}\left[\|\nabla\overline{\mathbf{u}}_\omega(0)\|^2+\alpha^2\|\Delta\overline{\mathbf{u}}_\omega(0)\|^2\right]+\int_0^T(\mathbf{f}(t),\Delta\overline{\mathbf{u}}_\omega(t))\,dt\end{aligned}\tag{3.8}$$

Proof: We begin with Leray-α. Multiply equation (1.3) by $\Delta\overline{\mathbf{u}}$ and integrate over the domain. This yields

$$(\mathbf{u}_t, \Delta\overline{\mathbf{u}}) + (\overline{\mathbf{u}} \cdot \nabla\mathbf{u}, \Delta\overline{\mathbf{u}}) + (\nabla p, \Delta\overline{\mathbf{u}}) - \nu(\Delta\mathbf{u}, \Delta\overline{\mathbf{u}}) = (\mathbf{f}, \Delta\overline{\mathbf{u}})$$

The nonlinearity must be handled differently from the other proofs. We must utilize Lemma 2 and the definition of the filter:

$$\begin{aligned}
(\overline{\mathbf{u}} \cdot \nabla\mathbf{u}, \Delta\overline{\mathbf{u}}) &= (\overline{\mathbf{u}} \cdot \nabla(-\alpha^2\Delta\overline{\mathbf{u}} + \overline{\mathbf{u}}), \Delta\overline{\mathbf{u}}) \\
&= -\alpha^2(\overline{\mathbf{u}} \cdot \nabla(\Delta\overline{\mathbf{u}}), \Delta\overline{\mathbf{u}}) + (\overline{\mathbf{u}} \cdot \nabla\overline{\mathbf{u}}, \Delta\overline{\mathbf{u}}) \\
&= 0
\end{aligned}$$

Given that $\Delta\mathbf{v} = -\nabla \times (\nabla \times \mathbf{v})$, and that $\nabla \times (\nabla p) = 0$, we see that the pressure term vanishes:

$$\begin{aligned}
(\nabla p, \Delta\overline{\mathbf{u}}) &= -(\nabla p, \nabla \times (\nabla \times \overline{\mathbf{u}})) \\
&= -(\nabla \times (\nabla p), \nabla \times \overline{\mathbf{u}}) \\
&= 0
\end{aligned}$$

Thus we are left with

$$(\mathbf{u}_t, \Delta\overline{\mathbf{u}}) - \nu(\Delta\mathbf{u}, \Delta\overline{\mathbf{u}}) = (\mathbf{f}, \Delta\overline{\mathbf{u}}).$$

The time derivative simplifies by expanding the filter and using Lemma 2

$$\begin{aligned}
(\mathbf{u}_t, \Delta\overline{\mathbf{u}}) &= \frac{1}{2}\frac{d}{dt}(\mathbf{u}, \Delta\overline{\mathbf{u}}) \\
&= \frac{1}{2}\frac{d}{dt}(-\alpha^2\Delta\overline{\mathbf{u}} + \overline{\mathbf{u}}, \Delta\overline{\mathbf{u}}) \\
&= \frac{1}{2}\frac{d}{dt}\left[-\alpha^2(\Delta\overline{\mathbf{u}}, \Delta\overline{\mathbf{u}}) - (\nabla\overline{\mathbf{u}}, \nabla\overline{\mathbf{u}})\right] \\
&= -\frac{1}{2}\frac{d}{dt}\left[\alpha^2 \left\|\Delta\overline{\mathbf{u}}\right\|^2 + \left\|\nabla\overline{\mathbf{u}}\right\|^2\right]
\end{aligned}$$

We use the fact that $\Delta = -\nabla \times \nabla\times$ and Lemma (5) to manipulate the dissipation term:

$$\begin{aligned}
-\nu(\Delta\mathbf{u}, \Delta\overline{\mathbf{u}}) &= -\nu\left(\Delta(\overline{\mathbf{u}} - \alpha^2\Delta\overline{\mathbf{u}}), \Delta\overline{\mathbf{u}}\right) \\
&= -\nu\left[(\Delta\overline{\mathbf{u}}, \Delta\overline{\mathbf{u}}) - \alpha^2(\Delta\Delta\overline{\mathbf{u}}, \Delta\overline{\mathbf{u}})\right]
\end{aligned}$$

$$
\begin{aligned}
&= -\nu \left[\|\Delta \overline{\mathbf{u}}\|^2 + \alpha^2 (\nabla \times (\nabla \times \Delta \overline{\mathbf{u}}), \Delta \overline{\mathbf{u}}) \right] \\
&= -\nu \left[\|\Delta \overline{\mathbf{u}}\|^2 + \alpha^2 (\nabla \times \Delta \overline{\mathbf{u}}, \nabla \times \Delta \overline{\mathbf{u}}) \right] \\
&= -\nu \left[\|\Delta \overline{\mathbf{u}}\|^2 + \alpha^2 \|\nabla \times \Delta \overline{\mathbf{u}}\|^2 \right]
\end{aligned}
$$

Thus we have

$$
-\frac{1}{2}\frac{d}{dt}\left[\|\nabla \overline{\mathbf{u}}\|^2 + \alpha^2 \|\Delta \overline{\mathbf{u}}\|^2 \right] - \nu \left[\|\Delta \overline{\mathbf{u}}\|^2 + \alpha^2 \|\nabla \times \Delta \overline{\mathbf{u}}\|^2 \right] = -(\nabla \mathbf{f}, \nabla \overline{\mathbf{u}})
$$

Integrating over time proves (3.5).

Next we show the approximate conservation for Modified Leray-α. We choose to multiply by $\Delta \mathbf{u}$. We then integrate over the domain:

$$
(\mathbf{u}_t, \Delta \mathbf{u}) + (\mathbf{u} \cdot \nabla \overline{\mathbf{u}}, \Delta \mathbf{u}) + (\nabla p, \Delta \mathbf{u}) - \nu (\Delta \mathbf{u}, \Delta \mathbf{u}) = (\mathbf{f}, \Delta \mathbf{u})
$$

As above, the pressure term vanishes. The dissipation term immediately becomes $-\nu \|\Delta \mathbf{u}\|^2$. By Lemma 2, the time derivative becomes $-\frac{1}{2}\frac{d}{dt}\|\nabla \mathbf{u}\|^2$. The nonlinear term "reduces" using Lemma 2

$$
\begin{aligned}
(\mathbf{u} \cdot \nabla \overline{\mathbf{u}}, \Delta \mathbf{u}) &= (\mathbf{u} \cdot \nabla (\mathbf{u} + \alpha^2 \Delta \overline{\mathbf{u}}), \Delta \mathbf{u}) \\
&= \alpha^2 (\mathbf{u} \cdot \nabla (\Delta \overline{\mathbf{u}}), \Delta \mathbf{u}) \\
&= \alpha^2 (\mathbf{u} \cdot \nabla (\Delta \overline{\mathbf{u}}), \Delta (-\alpha^2 \Delta \overline{\mathbf{u}} + \overline{\mathbf{u}})) \\
&= \alpha^4 (\mathbf{u} \cdot \nabla (\Delta \overline{\mathbf{u}}), \Delta (\Delta \overline{\mathbf{u}}))
\end{aligned}
$$

These results produce

$$
-\frac{1}{2}\frac{d}{dt}\|\nabla \mathbf{u}\|^2 + \alpha^4 (\mathbf{u} \cdot \nabla (\Delta \overline{\mathbf{u}}), \Delta (\Delta \overline{\mathbf{u}})) - \nu \|\Delta \mathbf{u}\|^2 = -(\nabla \mathbf{f}, \nabla \mathbf{u}).
$$

Integrating over time yields the desired result.

Next we prove the enstrophy balance for the NS-α model. Multiply equation (1.5) by $\Delta \mathbf{u}$ and integrate over the domain to obtain

$$
(\mathbf{u}_t, \Delta \mathbf{u}) + (\overline{\mathbf{u}} \times (\nabla \times \mathbf{u}), \Delta \mathbf{u}) + (\nabla p, \Delta \mathbf{u}) - \nu (\Delta \mathbf{u}, \Delta \mathbf{u}) = (\mathbf{f}, \Delta \mathbf{u})
$$

From above, we immediately have

$$-\frac{1}{2}\frac{d}{dt}\left\|\nabla\mathbf{u}\right\|^2+(\overline{\mathbf{u}}\times(\nabla\times\mathbf{u}),\Delta\mathbf{u})-\nu\left\|\Delta\mathbf{u}\right\|^2=-(\nabla\mathbf{f},\nabla\mathbf{u}).$$

The nonlinearity does indeed vanish since we consider enstrophy only in two dimensions and by Lemma 3:

$$\begin{aligned}(\overline{\mathbf{u}}\times(\nabla\times\mathbf{u}),\Delta\mathbf{u})&=-(\overline{\mathbf{u}}\times(\nabla\times\mathbf{u}),\nabla\times(\nabla\times\mathbf{u}))\\&=-(\nabla\times(\overline{\mathbf{u}}\times(\nabla\times\mathbf{u})),\nabla\times\mathbf{u})\\&=-(\overline{\mathbf{u}}\cdot\nabla(\nabla\times\mathbf{u})+(\nabla\times\mathbf{u})\cdot\nabla\overline{\mathbf{u}},\nabla\times\mathbf{u})\\&=-(\overline{\mathbf{u}}\cdot\nabla(\nabla\times\mathbf{u}),\nabla\times\mathbf{u})-((\nabla\times\mathbf{u})\cdot\nabla\overline{\mathbf{u}},\nabla\times\mathbf{u})\\&=0\end{aligned}$$

Thus

$$-\frac{1}{2}\frac{d}{dt}\left\|\nabla\mathbf{u}\right\|^2-\nu\left\|\Delta\mathbf{u}\right\|^2=-(\nabla\mathbf{f},\nabla\mathbf{u})$$

and we integrate with respect to time to obtain the desired result.

Finally we show the enstrophy balance for the NS-ω model. Multiply equation (1.6) by $\Delta\overline{\mathbf{u}}$ and integrate over the domain to see that

$$(\mathbf{u}_t,\Delta\overline{\mathbf{u}})+(\mathbf{u}\times(\nabla\times\overline{\mathbf{u}}),\Delta\overline{\mathbf{u}})+(\nabla p,\Delta\overline{\mathbf{u}})-\nu\left(\Delta\mathbf{u},\Delta\overline{\mathbf{u}}\right)=(\mathbf{f},\Delta\overline{\mathbf{u}})$$

The time step, dissipation term, and pressure all vanish as in the Leray-α case. The nonlinearity term vanishes again because we consider only 2 dimensions and Lemma 3.

$$\begin{aligned}(\mathbf{u}\times(\nabla\times\overline{\mathbf{u}}),\Delta\overline{\mathbf{u}})&=-\left(\mathbf{u}\times(\nabla\times\overline{\mathbf{u}}),\nabla\times(\nabla\times\overline{\mathbf{u}})\right)\\&=-\left(\nabla\times(\mathbf{u}\times(\nabla\times\overline{\mathbf{u}})),\nabla\times\overline{\mathbf{u}}\right)\\&=-\left(\mathbf{u}\cdot\nabla(\nabla\times\overline{\mathbf{u}})+(\nabla\times\overline{\mathbf{u}})\cdot\nabla\mathbf{u},\nabla\times\overline{\mathbf{u}}\right)\\&=-\left(\mathbf{u}\cdot\nabla(\nabla\times\overline{\mathbf{u}}),\nabla\times\overline{\mathbf{u}}\right)-\left((\nabla\times\overline{\mathbf{u}})\cdot\nabla\mathbf{u},\nabla\times\overline{\mathbf{u}}\right)\\&=0\end{aligned}$$

Therefore we have the same enstrophy conservation as for Leray-α, and we have proven (3.8). ■

3.3 Helicity Conservation

We now show the conservation of model helicity. NS-α exactly conserves helicity [3] and NS-ω conserves a model helicity [9]. The Leray-α and Modified Leray-α models approximately conserve a model helicity.

Theorem 8 *Let $\mathbf{u}_L, \mathbf{u}_{ML}, \mathbf{u}_\alpha$, and $\mathbf{u}_\omega$ be velocity solutions to equations (1.3), (1.4), (1.5) and (1.6), respectively. Then*

- *NS-α conserves helicity:*

$$\begin{aligned}(\mathbf{u}(T), \nabla \times \mathbf{u}(T)) + 2\nu \int_0^T (\nabla \times (\nabla \times \mathbf{u}(t)), \nabla \times \mathbf{u}(t))\, dt \\ = (\mathbf{u}(0), \nabla \times \mathbf{u}(0)) + 2 \int_0^T (\mathbf{f}(t), \nabla \times \mathbf{u}(t)) dt\end{aligned} \tag{3.9}$$

- *NS-ω conserves a model helicity:*

$$\begin{aligned}&(\overline{\mathbf{u}}(T), \nabla \times \overline{\mathbf{u}}(T)) + \alpha^2 (\nabla \times (\nabla \times \overline{\mathbf{u}}(T)), \nabla \times \overline{\mathbf{u}}(T)) \\ &\quad + 2\nu \int_0^T \left[(\nabla \times (\nabla \times \overline{\mathbf{u}}), \nabla \times \overline{\mathbf{u}}) + \alpha^2 (\nabla \times (\nabla \times (\Delta \overline{\mathbf{u}})), \nabla \times \overline{\mathbf{u}}(T)) \right] dt \\ &\quad = (\overline{\mathbf{u}}(0), \nabla \times \overline{\mathbf{u}}(0)) + \alpha^2 (\nabla \times (\nabla \times \overline{\mathbf{u}}(0)), \nabla \times \overline{\mathbf{u}}(0)) + 2 \int_0^T (\mathbf{f}(t), \nabla \times \overline{\mathbf{u}}) dt\end{aligned} \tag{3.10}$$

- *Leray-α approximately conserves model helicity [12]:*

$$\begin{aligned}(\mathbf{u}(T), \nabla \times \mathbf{u}(T)) + \alpha^2 \int_0^T (\Delta \overline{\mathbf{u}} \cdot \nabla \mathbf{u}, \nabla \times \mathbf{u}) dt \\ + 2\nu\alpha^2 \int_0^T (\nabla \times (\nabla \times \mathbf{u}(t)), \nabla \times \mathbf{u}(t))\, dt \\ = (\mathbf{u}(0), \nabla \times \mathbf{u}(0)) + 2 \int_0^T (\mathbf{f}(t), \nabla \times \mathbf{u}(t)) dt\end{aligned} \tag{3.11}$$

- *Modified Leray-α approximately conserves model helicity:*

$$
\begin{aligned}
&(\mathbf{u}_{ML}(T), \nabla \times \mathbf{u}_{ML}(T)) + \alpha^2 \int_0^T (\mathbf{u}_{ML} \cdot \nabla(\Delta \overline{\mathbf{u}}_{ML}), \nabla \times \mathbf{u}_{ML}) dt \\
&\quad + 2\nu\alpha^2 \int_0^T (\nabla \times (\nabla \times \mathbf{u}_{ML}(t)), \nabla \times \mathbf{u}_{ML}(t)) dt \\
&\quad = (\mathbf{u}_{ML}(0), \nabla \times \mathbf{u}_{ML}(0)) + 2 \int_0^T (\mathbf{f}(t), \nabla \times \mathbf{u}_{ML}(t)) dt
\end{aligned}
\tag{3.12}
$$

Proof: We begin with NS-α. Multiply equation (1.5) by $\nabla \times \mathbf{u}$ and integrate over the domain to obtain

$$
(\mathbf{u}_t, \nabla \times \mathbf{u}) + (\overline{\mathbf{u}} \times (\nabla \times \mathbf{u}), \nabla \times \mathbf{u}) + (\nabla p, \nabla \times \mathbf{u}) - \nu(\Delta \mathbf{u}, \nabla \times \mathbf{u}) = (\mathbf{f}, \nabla \times \mathbf{u})
$$

The nonlinearity term vanishes immediately since the curl of two vectors is perpendicular to each. The pressure term vanishes as in the proof for enstrophy balance. We rewrite dissipation as $\nu(\nabla \times (\nabla \times \mathbf{u}), \nabla \times \mathbf{u})$. Integrating over time yields the desired result.

Now consider NS-ω. Multiply (1.6) by $\nabla \times \overline{\mathbf{u}}$ and integrate over the domain. Then (1.6) becomes

$$
(\mathbf{u}_t, \nabla \times \overline{\mathbf{u}}) - (\mathbf{u} \times (\nabla \times \overline{\mathbf{u}}), \nabla \times \overline{\mathbf{u}}) + (\nabla p, \nabla \times \overline{\mathbf{u}}) - \nu(\Delta \mathbf{u}, \nabla \times \overline{\mathbf{u}}) = (\mathbf{f}, \nabla \times \overline{\mathbf{u}})
$$

Again, the nonlinearity and pressure terms vanish. We write out the differential filter to manipulate the time step:

$$
\begin{aligned}
(\mathbf{u}_t, \nabla \times \overline{\mathbf{u}}) &= \left((\overline{\mathbf{u}} - \alpha^2 \Delta \overline{\mathbf{u}})_t, \nabla \times \overline{\mathbf{u}}\right) \\
&= (\overline{\mathbf{u}}_t, \nabla \times \overline{\mathbf{u}}) - \alpha^2((\Delta \overline{\mathbf{u}})_t, \nabla \times \overline{\mathbf{u}}) \\
&= \frac{1}{2}\frac{d}{dt}(\overline{\mathbf{u}}, \nabla \times \overline{\mathbf{u}}) - \frac{\alpha^2}{2}\frac{d}{dt}(\Delta \overline{\mathbf{u}}, \nabla \times \overline{\mathbf{u}}) \\
&= \frac{1}{2}\frac{d}{dt}(\overline{\mathbf{u}}, \nabla \times \overline{\mathbf{u}}) + \frac{\alpha^2}{2}\frac{d}{dt}(\nabla \times (\nabla \times \overline{\mathbf{u}}), \nabla \times \overline{\mathbf{u}}) \\
&= \frac{1}{2}\frac{d}{dt}\left[(\overline{\mathbf{u}}, \nabla \times \overline{\mathbf{u}}) + \alpha^2(\nabla \times (\nabla \times \overline{\mathbf{u}}), \nabla \times \overline{\mathbf{u}})\right]
\end{aligned}
$$

The the dissipation term, using the differential filter and rewriting the Laplacian, becomes

$$\begin{aligned}
-\nu(\Delta\mathbf{u}, \nabla\times\overline{\mathbf{u}}) &= -\nu(\Delta(\overline{\mathbf{u}} - \alpha^2\Delta\overline{\mathbf{u}}), \nabla\times\overline{\mathbf{u}}) \\
&= -\nu\left[(\Delta\overline{\mathbf{u}}, \nabla\times\overline{\mathbf{u}}) - \alpha^2(\Delta\Delta\overline{\mathbf{u}}, \nabla\times\overline{\mathbf{u}})\right] \\
&= \nu\left[(\nabla\times(\nabla\times\overline{\mathbf{u}}), \nabla\times\overline{\mathbf{u}}) + \alpha^2(\nabla\times(\nabla\times\Delta\overline{\mathbf{u}}), \nabla\times\overline{\mathbf{u}})\right]
\end{aligned}$$

Integrating over time gives (3.10).

We now consider the Leray-α model. We multiply by $\nabla\times\overline{\mathbf{u}}$ and integrate over the domain. The proof is similar to that of NS-ω. The difference is in the nonlinearity. Writing out the differential filter leads to

$$\begin{aligned}
(\overline{\mathbf{u}}\cdot\nabla\mathbf{u}, \nabla\times\overline{\mathbf{u}}) &= (\overline{\mathbf{u}}\cdot\nabla(-\alpha^2\Delta\overline{\mathbf{u}} + \overline{\mathbf{u}}), \nabla\times\overline{\mathbf{u}}) \\
&= -\alpha^2(\overline{\mathbf{u}}\cdot\nabla(\Delta\mathbf{u}), \nabla\times\overline{\mathbf{u}}) + (\overline{\mathbf{u}}\cdot\nabla\overline{\mathbf{u}}, \nabla\times\overline{\mathbf{u}}) \\
&= -\alpha^2(\overline{\mathbf{u}}\cdot\nabla(\Delta\mathbf{u}), \nabla\times\overline{\mathbf{u}}) + (-\overline{\mathbf{u}}\times(\nabla\times\overline{\mathbf{u}}) + \frac{1}{2}\nabla(\overline{\mathbf{u}}\cdot\overline{\mathbf{u}}), \nabla\times\overline{\mathbf{u}}) \\
&= -\alpha^2(\overline{\mathbf{u}}\cdot\nabla(\Delta\mathbf{u}), \nabla\times\overline{\mathbf{u}}) + \frac{1}{2}(\nabla(\overline{\mathbf{u}}\cdot\overline{\mathbf{u}}), \nabla\times\overline{\mathbf{u}}) \\
&= -\alpha^2(\overline{\mathbf{u}}\cdot\nabla(\Delta\mathbf{u}), \nabla\times\overline{\mathbf{u}}) + \frac{1}{2}(\nabla\times\nabla(\overline{\mathbf{u}}\cdot\overline{\mathbf{u}}), \overline{\mathbf{u}}) \\
&= -\alpha^2(\overline{\mathbf{u}}\cdot\nabla(\Delta\mathbf{u}), \nabla\times\overline{\mathbf{u}})
\end{aligned}$$

We can find no reason why this term should vanish, and do not believe it should. Replacing the nonlinear term in NS-ω with this and integrating with respect to time gives (3.11).

The case for Modified Leray-α is similar. We multiply by $\nabla\times\mathbf{u}$ and integrate over the domain. This proof is similar to NS-α. When we rewrite $\overline{\mathbf{u}}$ in the nonlinear term using the filter, we find

$$(\mathbf{u}\cdot\nabla\overline{\mathbf{u}}, \nabla\times\mathbf{u}) = \alpha^2(\mathbf{u}\cdot\nabla(\Delta\overline{\mathbf{u}}), \nabla\times\mathbf{u})$$

Replacing the nonlinear term in NS-α with this and integrating with respect to time gives (3.12). ■

Chapter 4

Computational Results

Once we have chosen a model, we must formulate an implementation of that model for computation. For time-stepping, we choose the Crank-Nicolson since it is A-stable and $\mathcal{O}(\Delta t^2)$ accurate. The backward Euler method is also A-stable, but it is only $\mathcal{O}(\Delta t)$ accurate. Another reason to choose Crank-Nicolson is that this method conserves energy, while backward Euler does not [8]. For a spatial discretization, we use the Galerkin finite element method. All computations were done using FreeFEM++ [7].

We will use the following function spaces to define our discrete schemes.

Definition 9 *Function spaces*
Let $d = 2$ or 3.

$$P_n := \left\{ p : \Omega \to \mathbb{R}^d \middle| p \text{ is a polynomial of degree } n \right\} \tag{4.1}$$

$$X_h := \left\{ v_h \in C_0(\Omega) \middle| v_h = 0 \text{ on } \partial\Omega,\ v_h \in P_2 \right\} \tag{4.2}$$

$$Q_h := \left\{ q_h \in C_0(\Omega) \middle| \int_\Omega q_h dx = 0,\ q_h \in P_1 \right\} \tag{4.3}$$

We develop finite element formulations, and to ensure existence and uniqueness, the inf-sup condition, also known as the Ladyzhenskaya-Babuska-Brezzi (LBB) condition, must hold. We choose to implement the most common inf-sup stable elements known as the Taylor-Hood elements. In this space, velocity is approximated by second degree polynomials, and pressure is approximated by first degree polynomials, on each element.

For each model, we ran simulations to observe the model's performance in computing solutions to the two dimensional step problem. The step problem models flow through a 40×10 channel with a 1×1 step at the bottom of the channel, 5 units into the channel. We first computed solutions using the Navier-Stokes equations. We tested on a course mesh and on a fine mesh. The code used for each algorithm is given in the appendix. We discuss these findings first, followed by the results for Leray-α, NS-α, NS-ω, and Modified Leray-α.

4.1 On the Navier-Stokes Equations

4.1.1 Numerical Scheme

For the NSE, the nonlinear scheme is as follows:

$$\begin{aligned}
&\frac{1}{\Delta t}(\mathbf{u}_h^{n+1} - \mathbf{u}_h^n, \mathbf{v}_h) + b^*(\mathbf{u}_h^{n+\frac{1}{2}}, \nabla \mathbf{u}_h^{n+\frac{1}{2}}, \mathbf{v}_h) \\
&\qquad -(\tilde{p}_h^{n+1}, \nabla \cdot \mathbf{v}_h) + \nu(\nabla \mathbf{u}_h^{n+\frac{1}{2}}, \nabla \mathbf{v}_h) = (\mathbf{f}(t^{n+\frac{1}{2}}), \mathbf{v}_h) \qquad \forall\, \mathbf{v}_h \in X_h \qquad (4.4)\\
&\qquad\qquad (\nabla \cdot \mathbf{u}_h^{n+1}, q_h) = 0 \qquad \forall\, q_h \in Q_h \qquad (4.5)
\end{aligned}$$

This scheme leads to the Newton iteration

$$\begin{aligned}
&\frac{1}{\Delta t}(\mathbf{u}_h^{n+1,k}, \mathbf{v}_h) - (p_h^{n+1,k}, \nabla \cdot \mathbf{v}_h) + \frac{\nu}{2}(\nabla \mathbf{u}_h^{n+1,k}, \nabla \mathbf{v}_h) \\
&\quad + \frac{1}{8}(\mathbf{u}_h^{n+1,k} \cdot \nabla \mathbf{u}_h^{n+1,k-1}, \mathbf{v}_h) - \frac{1}{8}(\mathbf{u}_h^{n+1,k} \cdot \nabla \mathbf{v}_h, \mathbf{u}_h^{n+1,k-1}) \\
&\quad + \frac{1}{8}(\mathbf{u}_h^{n+1,k-1} \cdot \nabla \mathbf{u}_h^{n+1,k}, \mathbf{v}_h) - \frac{1}{8}(\mathbf{u}_h^{n+1,k-1} \cdot \nabla \mathbf{v}_h, \mathbf{u}_h^{n+1,k}) \\
&\quad + \frac{1}{8}(\mathbf{u}_h^{n+1,k} \cdot \nabla \mathbf{u}_h^{n}, \mathbf{v}_h) - \frac{1}{8}(\mathbf{u}_h^{n+1,k} \cdot \nabla \mathbf{v}_h, \mathbf{u}_h^{n}) \\
&\quad + \frac{1}{8}(\mathbf{u}_h^{n} \cdot \nabla \mathbf{u}_h^{n+1,k}, \mathbf{v}_h) - \frac{1}{8}(\mathbf{u}_h^{n} \cdot \nabla \mathbf{v}_h, \mathbf{u}_h^{n+1,k}) \\
&= \frac{1}{\Delta t}(\mathbf{u}_h^{n}, \mathbf{v}_h) - \frac{1}{8}(\mathbf{u}_h^{n} \cdot \nabla \mathbf{u}_h^{n}, \mathbf{v}_h) + \frac{1}{8}(\mathbf{u}_h^{n} \cdot \nabla \mathbf{v}_h, \mathbf{u}_h^{n}) \\
&\quad + \frac{1}{8}(\mathbf{u}_h^{n+1,k-1} \cdot \nabla \mathbf{u}_h^{n+1,k-1}, \mathbf{v}_h) - \frac{1}{8}(\mathbf{u}_h^{n+1,k-1} \cdot \nabla \mathbf{v}_h, \mathbf{u}_h^{n+1,k-1}) \\
&\quad - \frac{\nu}{2}(\nabla \mathbf{u}_h^{n}, \mathbf{v}_h) + (\mathbf{f}(t^{n+\frac{1}{2}}), \mathbf{v}_h) \qquad \forall\, \mathbf{v}_h \in X_h
\end{aligned} \tag{4.6}$$

$$(\nabla \cdot \mathbf{u}_h^{n+1}, q_h) = 0 \quad \forall\, q_h \in Q_h \tag{4.7}$$

at each time step.

4.1.2 Remarks

Here, as in all of our computations, $\mathbf{u}_h^{n+1,k}$ is the unknown velocity, $\mathbf{u}_h^{n+1,k-1}$ is the velocity solution of the previous iteration, $\mathbf{u}_h^n$ is the velocity solution of the previous time step, and $\mathbf{v}_h$ is the test function. The time step, Δt, is set to be .01, ie., 1 second corresponds to 100 iterations. We saw that for the fine mesh, the solution was smooth, and the eddies form and shed as expected. The fine mesh required 24,625 degrees of freedom.

The course mesh, oscillations start early and compound as time moves forward. The eddies do not shed, but deform. This mesh required only 5,845 degrees of freedom. We look to see if the models are an improvement on the course mesh.

4.1.3 Results

The following images show the results from computing the Navier-Stokes Equations on the fine and course meshes.

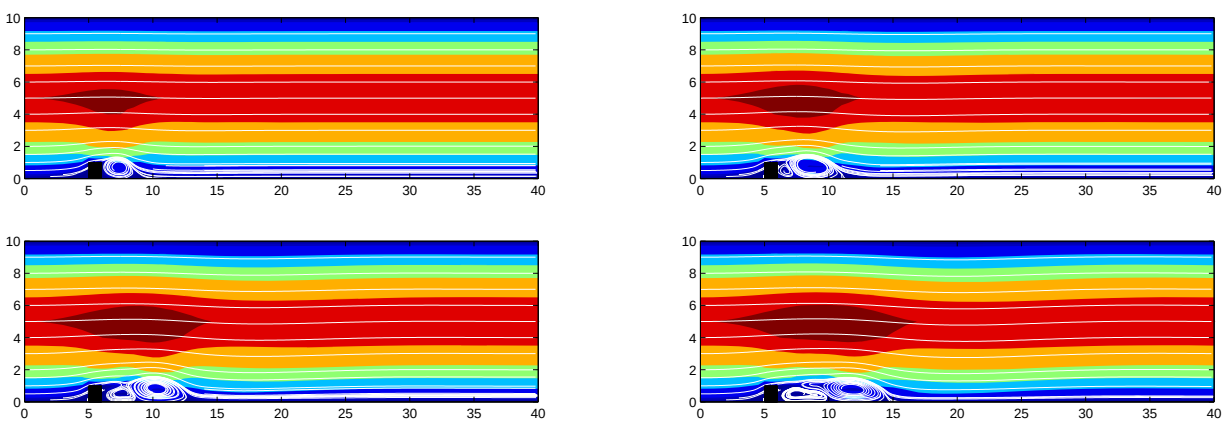

Figure 4.1: Solution to the Navier-Stokes Equations on a fine mesh

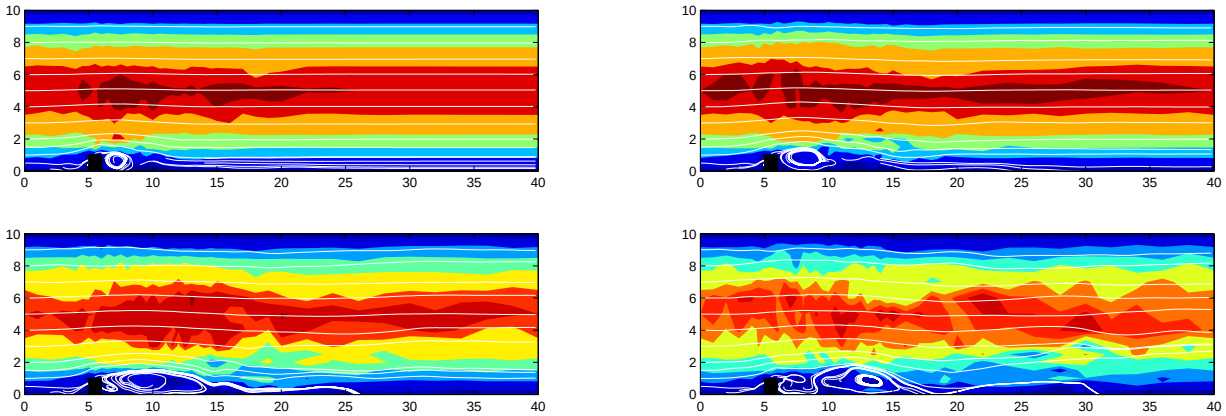

Figure 4.2: Solution to the Navier-Stokes Equations on a coarse mesh.

4.2 The Leray-α Model

4.2.1 Numerical Scheme

The nonlinear scheme for the Leray-α model is given as follows:

$$\frac{1}{\Delta t}(\mathbf{u}_h^{n+1} - \mathbf{u}_h^n, \mathbf{v}_h) + b^*(\mathbf{w}_h^{n+\frac{1}{2}}, \nabla \mathbf{u}_h^{n+\frac{1}{2}}, \mathbf{v}_h)$$

$$-(\tilde{p}_h^{n+1}, \nabla \cdot \mathbf{v}_h) + \nu(\nabla \mathbf{u}_h^{n+\frac{1}{2}}, \nabla \mathbf{v}_h) = (\mathbf{f}(t^{n+\frac{1}{2}}), \mathbf{v}_h) \qquad \forall\, \mathbf{v}_h \in X_h \tag{4.8}$$

$$(\nabla \cdot \mathbf{u}_h^{n+1}, q_h) = 0 \qquad \forall\, q_h \in Q_h \tag{4.9}$$

$$(\mathbf{w}_h^{n+1}, \chi_h) + \alpha^2(\nabla \mathbf{w}_h^{n+1}, \nabla \chi_h) - (\mathbf{u}_h^{n+1}, \chi_h) = 0 \qquad \forall\, \chi_h \in X_h \tag{4.10}$$

This leads to the Newton iteration

$$
\begin{aligned}
&\frac{1}{\Delta t}(\mathbf{u}_h^{n+1,k}, \mathbf{v}_h) - (p_h^{n+1,k}, \nabla \cdot \mathbf{v}_h) + \frac{\nu}{2}(\nabla \mathbf{u}_h^{n+1,k}, \nabla \mathbf{v}_h) \\
&\quad + \frac{1}{8}(\mathbf{w}_h^{n+1,k} \cdot \nabla \mathbf{u}_h^{n+1,k-1}, \mathbf{v}_h) - \frac{1}{8}(\mathbf{w}_h^{n+1,k} \cdot \nabla \mathbf{v}_h, \mathbf{u}_h^{n+1,k-1}) \\
&\quad + \frac{1}{8}(\mathbf{w}_h^{n+1,k-1} \cdot \nabla \mathbf{u}_h^{n+1,}, \mathbf{v}_h) - \frac{1}{8}(\mathbf{w}_h^{n+1,k-1} \cdot \nabla \mathbf{v}_h, \mathbf{u}_h^{n+1,k}) \\
&\quad + \frac{1}{8}(\mathbf{w}_h^{n+1,k} \cdot \nabla \mathbf{u}_h^{n}, \mathbf{v}_h) - \frac{1}{8}(\mathbf{w}_h^{n+1,k} \cdot \nabla \mathbf{v}_h, \mathbf{u}_h^{n}) \\
&\quad + \frac{1}{8}(\mathbf{w}_h^{n} \cdot \nabla \mathbf{u}_h^{n+1,k}, \mathbf{v}_h) - \frac{1}{8}(\mathbf{w}_h^{n} \cdot \nabla \mathbf{v}_h, \mathbf{u}_h^{n+1,k}) \\
&\quad + \frac{1}{8}(\mathbf{w}_h^{n+1,k} \cdot \nabla \mathbf{u}_h^{n+1,k-1}, \mathbf{v}_h) - \frac{1}{8}(\mathbf{w}_h^{n+1,k} \cdot \nabla \mathbf{v}_h, \mathbf{u}_h^{n+1,k-1}) \\
= &\frac{1}{\Delta t}(\mathbf{u}_h^{n}, \mathbf{v}_h) - \frac{1}{8}(\mathbf{w}_h^{n} \cdot \nabla \mathbf{u}_h^{n}, \mathbf{v}_h) - \frac{1}{8}(\mathbf{w}_h^{n} \cdot \nabla \mathbf{v}_h, \mathbf{u}_h^{n}) \\
&\quad + \frac{1}{8}(\mathbf{w}_h^{n+1,k-1} \cdot \nabla \mathbf{u}_h^{n+1,k-1}, \mathbf{v}_h) - \frac{1}{8}(\mathbf{w}_h^{n+1,k-1} \cdot \nabla \mathbf{v}_h, \mathbf{u}_h^{n+1,k-1}) \\
&\quad - \frac{\nu}{2}(\nabla \mathbf{u}_h^{n}, \nabla \mathbf{v}_h) + (\mathbf{f}(t^{n+\frac{1}{2}}), \mathbf{v}_h) \qquad \forall\, \mathbf{v}_h \in X_h
\end{aligned}
\tag{4.11}
$$

$$
(\nabla \cdot \mathbf{u}_h^{n+1}, q) = 0 \qquad \forall q_h \in Q_h \tag{4.12}
$$

$$
(\mathbf{w}_h^{n+1}, \chi_h) + \alpha^2(\nabla \mathbf{w}_h^{n+1}, \nabla \chi_h) - (\mathbf{u}_h^{n+1}, \chi_h) = 0 \qquad \forall\, \chi_h \in X_h \tag{4.13}
$$

at each timestep.

4.2.2 Remarks

For the α-models, we denote the filtered solution as $\mathbf{w}$. Using a filter introduces a new variable, thus we need to solve a filter system for $\mathbf{w}$, with

$$
\alpha^2(\nabla \mathbf{w}^{n+1}, \nabla \gamma) + (\mathbf{w}^{n+1}, \gamma) - (\mathbf{u}^{n+1}, \gamma) = 0 \tag{4.14}
$$

coupled to the momentum and mass conservation equations.

We see in Figure 4.3, the Leray-α model, that the solutions are a significant improvement on the Navier-Stokes equations on the course mesh, and essentially correctly predicts the flow. Behind the step, an eddy forms and sheds, but it is not clear that a new eddy is formin. It appears there is some rotation behind the step after the first eddy moves down the channel, but never forms a full

eddy.

4.2.3 Results

The following images show the results from computing the Leray-α model on the course mesh.

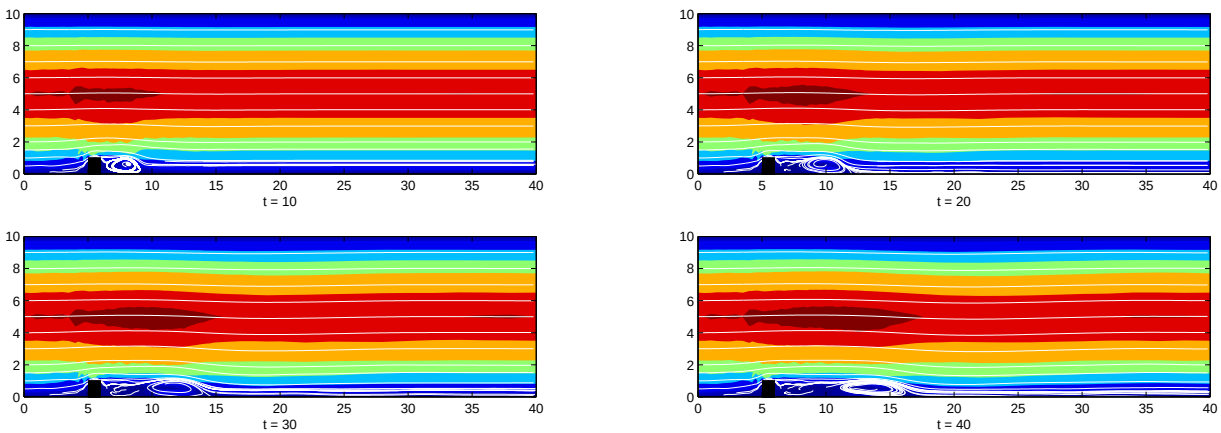

Figure 4.3: Leray-α solution on coarse mesh.

4.3 The Modified Leray-α Model

4.3.1 Numerical Scheme

The nonlinear scheme for the Modified Leray-α model is given as follows:

$$\frac{1}{\Delta t}(\mathbf{u}_h^{n+1} - \mathbf{u}_h^n, \mathbf{v}_h) + b^*(\mathbf{u}_h^{n+\frac{1}{2}}, \nabla \mathbf{w}_h^{n+\frac{1}{2}}, \mathbf{v}_h)$$
$$-(p_h^{n+\frac{1}{2}}, \nabla \cdot \mathbf{v}_h) + \nu(\nabla \mathbf{u}_h^{n+\frac{1}{2}}, \nabla \mathbf{v}_h) = (\mathbf{f}(t^{n+\frac{1}{2}}), \mathbf{v}_h) \quad \forall\, \mathbf{v}_h \in X_h \tag{4.15}$$

$$(\nabla \cdot \mathbf{w}_h^{n+1}, q_h) = 0 \quad \forall\, q_h \in Q_h \tag{4.16}$$

$$(\mathbf{w}_h^{n+1}, \chi_h) + \alpha^2(\nabla \mathbf{w}_h^{n+1}, \nabla \chi_h) - (\mathbf{u}_h^{n+1}, \chi_h) = 0 \qquad \forall\, \chi_h \in X_h \tag{4.17}$$

This leads to the Newton iteration

$$
\begin{aligned}
&\frac{1}{\Delta t}(\mathbf{u}_h^{n+1,k}, \mathbf{v}_h) - (p_h^{n+1,k}, \nabla \cdot \mathbf{v}_h) + \frac{\nu}{2}(\nabla \mathbf{u}_h^{n+1,k}, \nabla \mathbf{v}_h) \\
&\quad + \frac{1}{8}(\mathbf{u}_h^{n+1,k} \cdot \nabla \mathbf{w}_h^{n+1,k-1}, \mathbf{v}_h) - \frac{1}{8}(\mathbf{u}_h^{n+1,k} \cdot \nabla \mathbf{v}_h, \mathbf{w}_h^{n+1,k-1}) \\
&\quad + \frac{1}{8}(\mathbf{u}_h^{n+1,k-1} \cdot \nabla \mathbf{w}_h^{n+1,k}, \mathbf{v}_h) - \frac{1}{8}(\mathbf{u}_h^{n+1,k-1} \cdot \nabla \mathbf{v}_h, \mathbf{w}_h^{n+1,k}) \\
&\quad + \frac{1}{8}(\mathbf{u}_h^{n+1,k} \cdot \nabla \mathbf{w}_h^{n}, \mathbf{v}_h) - \frac{1}{8}(\mathbf{u}_h^{n+1,k} \cdot \nabla \mathbf{v}_h, \mathbf{w}_h^{n}) \\
&\quad + \frac{1}{8}(\mathbf{u}_h^{n} \cdot \nabla \mathbf{w}_h^{n+1,k}, \mathbf{v}_h) - \frac{1}{8}(\mathbf{u}_h^{n} \cdot \nabla \mathbf{v}_h, \mathbf{w}_h^{n+1,k}) \\
= &\frac{1}{\Delta t}(\mathbf{u}_h^{n}, \mathbf{v}_h) - \frac{1}{8}(\mathbf{u}_h^{n} \cdot \nabla \mathbf{w}_h^{n}, \mathbf{v}_h) + \frac{1}{8}(\mathbf{u}_h^{n} \cdot \nabla \mathbf{v}h, \mathbf{w}_h^{n}) \\
&\quad + \frac{1}{8}(\mathbf{u}_h^{n+1,k-1} \cdot \nabla \mathbf{w}_h^{n+1,k-1}, \mathbf{v}_h) - \frac{1}{8}(\mathbf{u}_h^{n+1,k-1} \cdot \nabla \mathbf{v}_h, \mathbf{w}_h^{n+1,k-1}) \\
&\quad - \frac{\nu}{2}(\nabla \mathbf{u}_h^{n}, \nabla \mathbf{v}_h) + (\mathbf{f}(t^{n+\frac{1}{2}}), \mathbf{v}_h) \qquad \forall\, \mathbf{v}_h \in X_h
\end{aligned}
\tag{4.18}
$$

$$
(\nabla \cdot \mathbf{w}_h^{n+1}, q) = 0 \qquad \forall q_h \in Q_h \tag{4.19}
$$

$$
(\mathbf{w}_h^{n+1}, \chi_h) + \alpha^2(\nabla \mathbf{w}_h^{n+1}, \nabla \chi_h) - (\mathbf{u}_h^{n+1}, \chi_h) = 0 \qquad \forall\, \chi_h \in X_h \tag{4.20}
$$

at each timestep.

4.3.2 Remarks and Results

The following images show the poor results from computing the Modified Leray-α model on the course mesh. We see that the model develops oscillations very early and becomes numerically unstable.

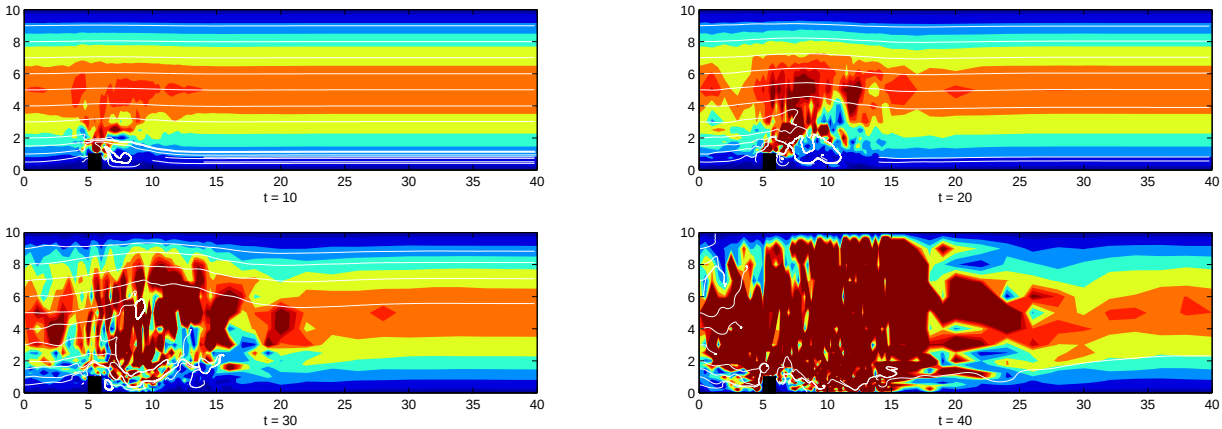

Figure 4.4: Modified Leray-α solution on coarse mesh.

4.4 The NS-α Model

4.4.1 Numerical Scheme

The NS-α model yields the following nonlinear scheme:

$$\frac{1}{\Delta t}(\mathbf{u}_h^{n+1} - \mathbf{u}_h^n, \mathbf{v}_h) - (\mathbf{w}_h^{n+\frac{1}{2}} \times (\nabla \times \mathbf{u}_h^{n+\frac{1}{2}}), \mathbf{v}_h)$$

$$+(\tilde{p}_h^{n+1}, \nabla \cdot \mathbf{v}_h) + \nu(\nabla \mathbf{u}_h^{n+\frac{1}{2}}, \nabla \mathbf{v}_h) = (\mathbf{f}(t^{t+\frac{1}{2}}), \mathbf{v}_h) \qquad \forall\, \mathbf{v}_h \in X_h \tag{4.21}$$

$$(\nabla \cdot \mathbf{w}_h^{n+\frac{1}{2}}, q_h) = 0 \qquad \forall\, q_h \in Q_h \tag{4.22}$$

$$(\mathbf{w}_h^{n+1}, \chi_h) + \alpha^2(\nabla \mathbf{w}_h^{n+1}, \nabla \chi_h) - (\mathbf{u}_h^{n+1}, \chi_h) = 0 \qquad \forall\, \chi_h \in X_h \tag{4.23}$$

We also chose to implement grad-div stabilization to the model to dampen the oscillations. This implementation is given by

$$\frac{1}{\Delta t}(\mathbf{u}_h^{n+1} - \mathbf{u}_h^n, \mathbf{v}_h) - (\mathbf{w}_h^{n+\frac{1}{2}} \times (\nabla \times \mathbf{u}_h^{n+\frac{1}{2}}), \mathbf{v}_h)$$

$$+(\tilde{p}_h^{n+1}, \nabla \cdot \mathbf{v}_h) + \nu(\nabla \mathbf{u}_h^{n+\frac{1}{2}}, \nabla \mathbf{v}_h) + (\nabla \cdot \mathbf{u}_h^{n+\frac{1}{2}}, \nabla \cdot \mathbf{v}_h) = (\mathbf{f}(t^{t+\frac{1}{2}}), \mathbf{v}_h) \qquad \forall\, \mathbf{v}_h \in X_h \tag{4.24}$$

$$(\nabla \cdot \mathbf{w}_h^{n+\frac{1}{2}}, q_h) = 0 \qquad \forall\, q_h \in Q_h \tag{4.25}$$

$$(\mathbf{w}_h^{n+1}, \chi_h) + \alpha^2(\nabla \mathbf{w}_h^{n+1}, \nabla \chi_h) - (\mathbf{u}_h^{n+1}, \chi_h) = 0 \qquad \forall\, \chi_h \in X_h \tag{4.26}$$

The original scheme leads to the following Newton iteration at each time step:

$$
\begin{aligned}
&\frac{1}{\Delta t}(\mathbf{u}_h^{n+1,k}, \mathbf{v}_h) - (p_h^{n+1,k}, \nabla\cdot\mathbf{v}_h) + \frac{\nu}{2}(\nabla\mathbf{u}_h^{n+1,k}, \nabla\mathbf{v}_h) \\
&\quad - \frac{1}{4}(\mathbf{w}_h^{n+1,k}\times(\nabla\times\mathbf{u}_h^{n+1,k-1}), \mathbf{v}_h) - \frac{1}{4}(\mathbf{w}_h^{n+1,k-1}\times(\nabla\times\mathbf{u}^{n+1,k}), \mathbf{v}_h) \\
&\quad + \frac{1}{4}(\mathbf{w}_h^{n+1,k-1}\times(\nabla\times\mathbf{u}_h^{n+1,k-1})) - \frac{1}{4}(\mathbf{w}_h^{n,k}\times(\nabla\times\mathbf{u}_h^{n+1,k}), \mathbf{v}_h) \\
&\quad - \frac{1}{4}(\mathbf{w}_h^{n+1,k}\times(\nabla\times\mathbf{u}_h^{n,k}), \mathbf{v}_h) - \frac{1}{4}(\mathbf{w}_h^{n,k}\times(\nabla\times\mathbf{u}_h^{n,k}), \mathbf{v}_h) \\
&= \frac{1}{\Delta t}(\mathbf{u}_h^{n}, \mathbf{v}_h) - \frac{\nu}{2}(\nabla\mathbf{u}_h^{u}, \nabla\mathbf{v}_h) + (\mathbf{f}(t^{n+\frac{1}{2}}), \mathbf{v}_h) \qquad \forall\,\mathbf{v}_h\in X_h
\end{aligned}
\tag{4.27}
$$

$$
(\nabla\cdot\mathbf{w}_h^{n+1}, q) = 0 \qquad \forall q_h\in Q_h \tag{4.28}
$$

$$
(\mathbf{w}_h^{n+1}, \chi_h) + \alpha^2(\nabla\mathbf{w}_h^{n+1}, \nabla\chi_h) - (\mathbf{u}_h^{n+1}, \chi_h) = 0 \qquad \forall\,\chi_h\in X_h \tag{4.29}
$$

The scheme with grad-div stabilization yields the Newton iteration

$$
\begin{aligned}
&\frac{1}{\Delta t}(\mathbf{u}_h^{n+1,k}, \mathbf{v}_h) - (p_h^{n+1,k}, \nabla\cdot\mathbf{v}_h) + \frac{\nu}{2}(\nabla\mathbf{u}_h^{n+1,k}, \nabla\mathbf{v}_h) \\
&\quad - \frac{1}{4}(\mathbf{w}_h^{n+1,k}\times(\nabla\times\mathbf{u}_h^{n+1,k-1}), \mathbf{v}_h) - \frac{1}{4}(\mathbf{w}_h^{n+1,k-1}\times(\nabla\times\mathbf{u}^{n+1,k}), \mathbf{v}_h) \\
&\quad + \frac{1}{4}(\mathbf{w}_h^{n+1,k-1}\times(\nabla\times\mathbf{u}_h^{n+1,k-1})) - \frac{1}{4}(\mathbf{w}_h^{n,k}\times(\nabla\times\mathbf{u}_h^{n+1,k}), \mathbf{v}_h) \\
&\quad - \frac{1}{4}(\mathbf{w}_h^{n+1,k}\times(\nabla\times\mathbf{u}_h^{n,k}), \mathbf{v}_h) - \frac{1}{4}(\mathbf{w}_h^{n,k}\times(\nabla\times\mathbf{u}_h^{n,k}), \mathbf{v}_h) \\
&\quad + \frac{1}{2}(\nabla\cdot\mathbf{u}_h^{n+1}, \nabla\cdot\mathbf{v}_h) \\
&= \frac{1}{\Delta t}(\mathbf{u}_h^{n}, \mathbf{v}_h) - \frac{\nu}{2}(\nabla\mathbf{u}_h^{u}, \nabla\mathbf{v}_h) \\
&\quad - \frac{1}{2}(\nabla\cdot\mathbf{u}_h^{n}, \nabla\cdot\mathbf{v}_h) + (\mathbf{f}(t^{n+\frac{1}{2}}), \mathbf{v}_h) \qquad \forall\,\mathbf{v}_h\in X_h
\end{aligned}
\tag{4.30}
$$

$$
(\nabla\cdot\mathbf{w}_h^{n+1}, q) = 0 \qquad \forall q_h\in Q_h \tag{4.31}
$$

$$
(\mathbf{w}_h^{n+1}, \chi_h) + \alpha^2(\nabla\mathbf{w}_h^{n+1}, \nabla\chi_h) - (\mathbf{u}_h^{n+1}, \chi_h) = 0 \qquad \forall\,\chi_h\in X_h \tag{4.32}
$$

4.4.2 Remarks and Results

We ran two simulations with the NS-α model, shown in Figures 4.5 and 4.4.2. One simulation was with the model with no alterations. The second simulation was the NS-α with grad-div stabilization. The purpose of the grad-div stabilization is to "have a stabilizing effect for small ν values" [11]. The addition of the grad-div term had a significant effect on the solution. Without it, the solutions begin to oscillate very quickly. The oscillations take over, and the solution becomes unstable. The stabilizations dampen the oscillations and the model correctly predicts the flow.

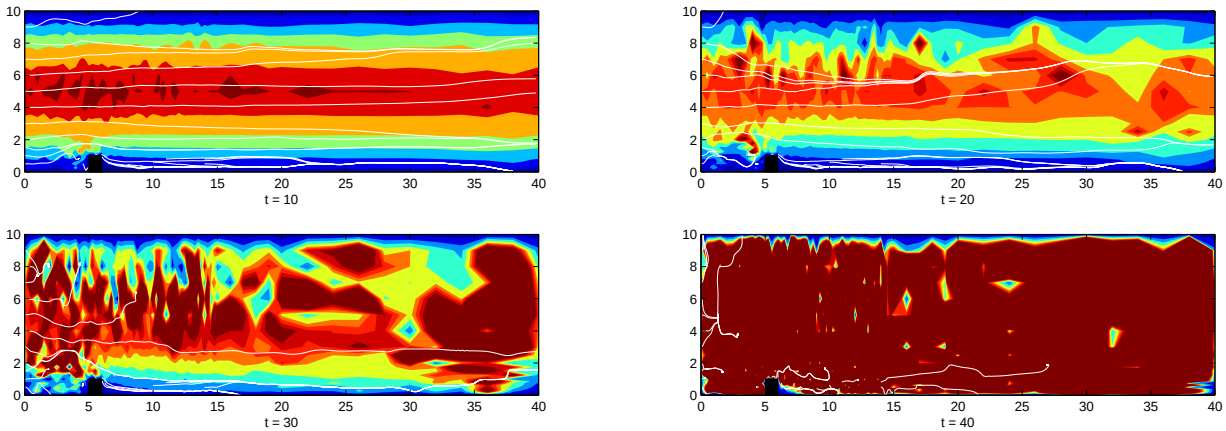

Figure 4.5: NS-α solution without grad-div stabilization.

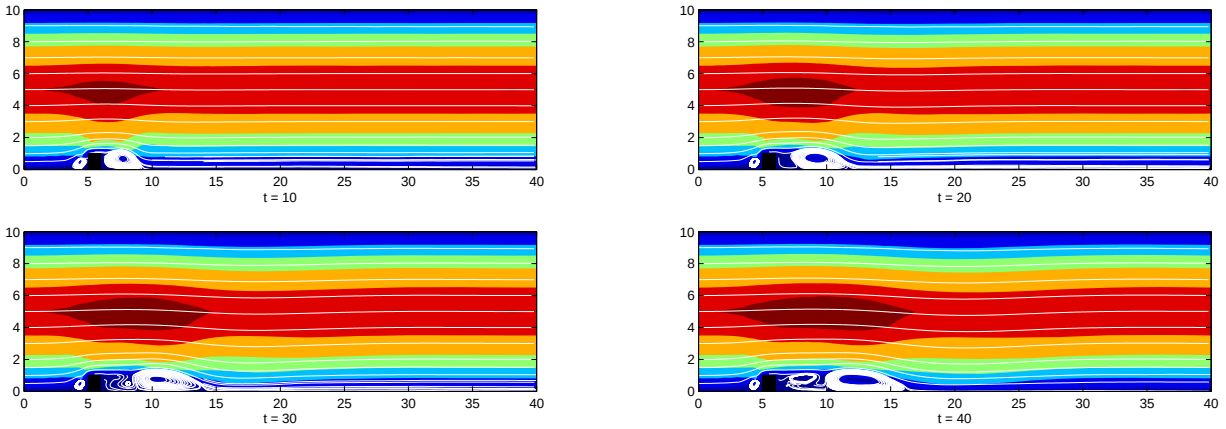

Figure 4.6: NS-α solution with grad-div stabilization.

4.5 The NS-ω Model

4.5.1 Numerical Scheme

As with the NS-α model, we used two schemes. We first give the nonlinear scheme for NS-ω.

$$\frac{1}{\Delta t}(\mathbf{u}_h^{n+1} - \mathbf{u}_h^n, \mathbf{v}_h) - (\mathbf{u}_h^{n+\frac{1}{2}} \times (\nabla \times \mathbf{w}_h^{n+\frac{1}{2}}), \mathbf{v}_h)$$
$$+(p_h^{n+\frac{1}{2}}, \nabla \cdot \mathbf{v}_h) + \nu(\nabla \mathbf{u}_h^{n+\frac{1}{2}}, \nabla \mathbf{v}_h) = (\mathbf{f}(t^{t+\frac{1}{2}}), \mathbf{v}_h) \qquad \forall\, \mathbf{v}_h \in X_h \tag{4.33}$$
$$(\nabla \cdot \mathbf{u}_h^{n+\frac{1}{2}}, q_h) = 0 \qquad \forall\, q_h \in Q_h \tag{4.34}$$
$$(\mathbf{w}_h^{n+1}, \chi_h) + \alpha^2(\nabla \mathbf{w}_h^{n+1}, \nabla \chi_h) - (\mathbf{u}_h^{n+1}, \chi_h) = 0 \qquad \forall\, \chi_h \in X_h \tag{4.35}$$

We also chose to implement grad-div stabilization to the model. This implementation is given by

$$\frac{1}{\Delta t}(\mathbf{u}_h^{n+1} - \mathbf{u}_h^n, \mathbf{v}_h) - (\mathbf{u}_h^{n+\frac{1}{2}} \times (\nabla \times \mathbf{w}_h^{n+\frac{1}{2}}), \mathbf{v}_h)$$
$$+(\tilde{p}_h^{n+1}, \nabla \cdot \mathbf{v}_h) + \nu(\nabla \mathbf{u}_h^{n+\frac{1}{2}}, \nabla \mathbf{v}_h) + (\nabla \cdot \mathbf{u}_h^{n+\frac{1}{2}}, \nabla \cdot \mathbf{v}_h) = (\mathbf{f}(t^{t+\frac{1}{2}}), \mathbf{v}_h) \tag{4.36}$$
$$(\nabla \cdot \mathbf{w}^{n+\frac{1}{2}}, q) = 0 \tag{4.37}$$

The original nonlinear scheme yields the following Newton iteration per timestep:

$$\begin{aligned}
&\frac{1}{\Delta t}(\mathbf{u}_h^{n+1,k}, \mathbf{v}_h) - (p_h^{n+1,k}, \nabla \cdot \mathbf{v}_h) + \frac{\nu}{2}(\nabla \mathbf{u}_h^{n+1,k}, \nabla \mathbf{v}_h) \\
&\quad - \frac{1}{4}(\mathbf{u}_h^{n+1,k} \times (\nabla \times \mathbf{w}_h^{n+1,k-1}), \mathbf{v}_h) - \frac{1}{4}(\mathbf{u}_h^{n+1,k-1} \times (\nabla \times \mathbf{w}^{n+1,k}), \mathbf{v}_h) \\
&\quad + \frac{1}{4}(\mathbf{u}_h^{n+1,k-1} \times (\nabla \times \mathbf{w}_h^{n+1,k-1})) - \frac{1}{4}(\mathbf{u}_h^{n,k} \times (\nabla \times \mathbf{w}_h^{n+1,k}), \mathbf{v}_h) \\
&\quad - \frac{1}{4}(\mathbf{u}_h^{n+1,k} \times (\nabla \times \mathbf{w}_h^{n,k}), \mathbf{v}_h) - \frac{1}{4}(\mathbf{u}_h^{n,k} \times (\nabla \times \mathbf{w}_h^{n,k}), \mathbf{v}_h) \\
&= \frac{1}{\Delta t}(\mathbf{u}_h^n, \mathbf{v}_h) - \frac{\nu}{2}(\nabla \mathbf{u}_h^u, \nabla \mathbf{v}_h) + (\mathbf{f}(t^{n+\frac{1}{2}}), \mathbf{v}_h) \qquad \forall\, \mathbf{v}_h \in X_h
\end{aligned} \tag{4.38}$$

$$(\nabla \cdot \mathbf{u}_h^{n+1}, q) = 0 \qquad \forall q_h \in Q_h \tag{4.39}$$

$$(\mathbf{w}_h^{n+1}, \chi_h) + \alpha^2(\nabla \mathbf{w}_h^{n+1}, \nabla \chi_h) - (\mathbf{u}_h^{n+1}, \chi_h) = 0 \qquad \forall\, \chi_h \in X_h \tag{4.40}$$

The scheme with grad-div stabilization yields the Newton iteration

$$
\begin{aligned}
&\frac{1}{\Delta t}(\mathbf{u}_h^{n+1,k}, \mathbf{v}_h) - (p_h^{n+1,k}, \nabla \cdot \mathbf{v}_h) + \frac{\nu}{2}(\nabla \mathbf{u}_h^{n+1,k}, \nabla \mathbf{v}_h) \\
&\quad - \frac{1}{4}(\mathbf{u}_h^{n+1,k} \times (\nabla \times \mathbf{w}_h^{n+1,k-1}), \mathbf{v}_h) - \frac{1}{4}(\mathbf{u}_h^{n+1,k-1} \times (\nabla \times \mathbf{w}^{n+1,k}), \mathbf{v}_h) \\
&\quad + \frac{1}{4}(\mathbf{u}_h^{n+1,k-1} \times (\nabla \times \mathbf{w}_h^{n+1,k-1})) - \frac{1}{4}(\mathbf{u}_h^{n,k} \times (\nabla \times \mathbf{w}_h^{n+1,k}), \mathbf{v}_h) \\
&\quad - \frac{1}{4}(\mathbf{u}_h^{n+1,k} \times (\nabla \times \mathbf{w}_h^{n,k}), \mathbf{v}_h) - \frac{1}{4}(\mathbf{u}_h^{n,k} \times (\nabla \times \mathbf{w}_h^{n,k}), \mathbf{v}_h) \\
&\quad + \frac{1}{2}(\nabla \cdot \mathbf{u}_h^{n+1}, \nabla \cdot \mathbf{v}_h) \\
&= \frac{1}{\Delta t}(\mathbf{u}_h^{n}, \mathbf{v}_h) - \frac{\nu}{2}(\nabla \mathbf{u}_h^{u}, \nabla \mathbf{v}_h) \\
&\quad - \frac{1}{2}(\nabla \cdot \mathbf{u}_h^{n}, \nabla \cdot \mathbf{v}_h) + (\mathbf{f}(t^{n+\frac{1}{2}}), \mathbf{v}_h) \qquad \forall\, \mathbf{v}_h \in X_h
\end{aligned}
\tag{4.41}
$$

$$
(\nabla \cdot \mathbf{u}_h^{n+1}, q) = 0 \qquad \forall q_h \in Q_h \tag{4.42}
$$

$$
(\mathbf{w}_h^{n+1}, \chi_h) + \alpha^2(\nabla \mathbf{w}_h^{n+1}, \nabla \chi_h) - (\mathbf{u}_h^{n+1}, \chi_h) = 0 \qquad \forall\, \chi_h \in X_h \tag{4.43}
$$

4.5.2 Remarks and results

The NS-ω model is similar to the NS-α model in many respects, particularly that grad-div stabilization is needed to dampen oscillations. The effect of grad-div stabilization on the model is significant. It transforms the model from numerically unstable to correctly predicting the flow through the channel.

The following images show the results from computing the NS-ω model, first without grad-div stabilization, then with grad-div stabilization. We see that implementing grad-div stabilization greatly improves the stability of the scheme.

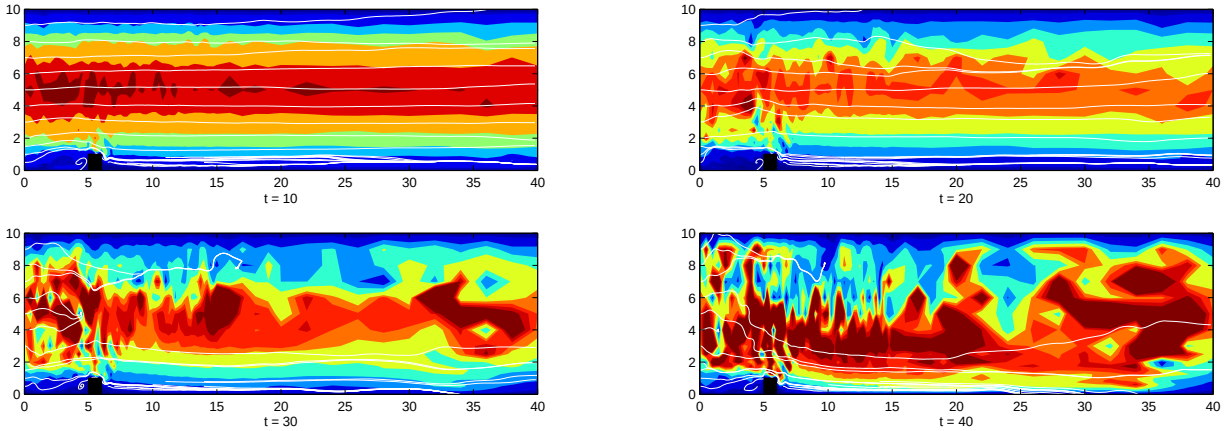

Figure 4.7: NS-ω solution without grad-div stabilization.

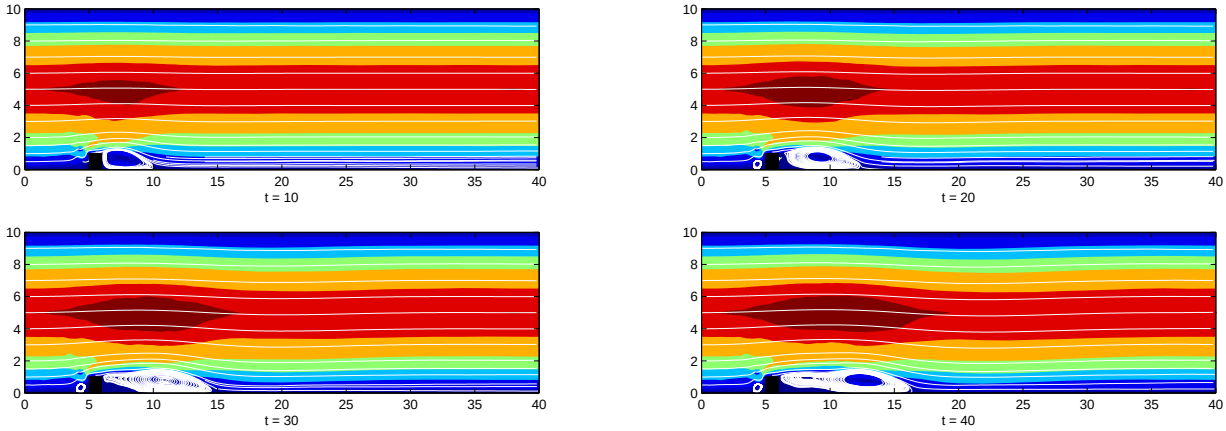

Figure 4.8: NS-ω solution with grad-div stabilization.

Chapter 5

Conclusions

We have shown that all four models conserve at least a model energy. The Leray-α, NS-α, and NS-ω models all conserve at least a model enstrophy, while we could only show that the Modified Leray-α model approximately conserves model enstrophy. The only models that we could show conserve helicity were the NS-α and NS-ω model. We could not show an exact law for Leray-α or Modified Leray-α, and we do not believe one to exist.

To obtain reasonable results, we had to implement grad-div stabilization with the NS-α and NS-ω models. This term was added to dampen oscillations and has a remarkable effect on the stability of the models. We believe these models to be the most promising, and that they should be the focus of future work.

The Leray-α model produces good results. Further study to develop reliable methods to further smooth the solution should prove promising. The Modified Leray-α scheme does not perform well in simulation. This is not surprising because the model does not conserve two of the three physical quantities. Future research on this model may be limited due its limited physical fidelity.

Appendices

Appendix A FreeFEM++ Implementation of the Navier-Stokes Equations

```
//   Crank-Nicolson FEM for 2d Navier-Stokes equations for step problem
//
real s0=clock();

// This defines the boundary in several pieces.  Note the bdry is the [0,40]x[0,10] rectangle with a
// 1x1 step on the bottom between x=5 and x=6.  The "label" is given so that boundary conditions
// can be easily enforced later.
//
border A1(t=0,5){x=t; y=0; label=1;};
border A2(t=0,3){x=6+t; y=0; label=2;};
border A3(t=0,12){x=9+t; y=0; label=9;};
border A4(t=0,19){x=21+t; y=0; label=10;};
border S1(t=0,1){x=5; y=t; label=3;};
border S2(t=0,1){x=5+t; y=1; label=4;};
border S3(t=0,1){x=6; y=1-t; label=5;};
border B(t=0,10){x=40; y=t; label=6;};
border C(t=0,40){x=40-t; y=10; label=7;};
border D(t=0,10){x=0; y=10-t; label=8;};

// There are several different meshes that Carolina has predefined for easy use.  Mesh level 1
// is the coarest (least number of dof) and level 4 is the finest.
//
//  mesh level 4
//mesh Th=buildmesh( A1(16)+A2(48)+A3(70)+A4(60)+S1(8)+S2(8)+S3(16)+B(48)+C(64)+D(48) );
//  mesh level 3 ----I used this one before-------
//mesh Th=buildmesh( A1(16)+A2(24)+A3(32)+A4(40)+S1(8)+S2(8)+S3(8)+B(48)+C(64)+D(48) );
//  mesh level 3
mesh Th=buildmesh( A1(8)+A2(24)+A3(40)+A4(40)+S1(4)+S2(4)+S3(8)+B(48)+C(64)+D(48) );
///////////mesh level 2
//mesh Th=buildmesh( A1(8)+A2(12)+A3(25)+A4(25)+S1(2)+S2(2)+S3(4)+B(8)+C(32)+D(12) );
//
///////////mesh level 1
//mesh Th=buildmesh( A1(4)+A2(6)+A3(10)+A4(10)+S1(2)+S2(2)+S3(2)+B(12)+C(16)+D(12) );

// IF YOU WANT TO PLOT THE MESH, USE THIS COMMAND
//plot(Th,ps="Level2_mesh.eps",wait=1);

// Define the finite element spaces.  Vh2 will be the velocity space with P2 elements, and Vh
// the pressure space with P1 elements.  The (P2,P1) element pair is inf-sup stable and is
// often called the "Taylor-Hood" element.
fespace Vh2(Th,P2);
fespace Vh(Th,P1);

// Declare the variables that will be used.
//
// Velocity variables: Even though velocity is 2d, freefem treats only
// 1d variables, so the naming convention will be to use u1 for x component and u2 for ycomponent
//
Vh2 u2,v2,un2,up2,e2;
Vh2 u1,v1,un1,up1,e1;
```

```
// Pressure variables
Vh pm=0, p=0, q;


// At Reynolds number 600, eddies should form behind the step, detach and shed into the flow
real reylnods=600;
real err1, err2;

// INITIALIZE VARIABLES.  G IS A FUNCTION THAT WILL BE USED TO ENFORCE PARABOLIC INFLOW.
// INTIAL CONDITION WILL BE PARABOLIC PROFILE FOR THE WHOLE CHANNEL
func g=(y)*(10-y)/25;
u1=g; u2=0;
un1=0; un2=0;
up1=0; up2=0;

real  nu=1;
int itnl=0,iter=0, MaxNlIt=30;
real dt=0;

// The following is used for outputting data.  After certain specified timesteps, we will print out velocity
// and pressure (to a file) at these points.
// LEO SAYS: I THINK WE NEED TO HAVE MORE POINTS!  ESPECIALLY BETWEEN 0<Y<2 AND 0<X<15
real[int] xvec = [0, 0.5, 1, 1.5, 2, 2.5, 3, 3.5, 4, 4.25, 4.5, 4.75, 5, 5.25, 5.5, 5.75, 6, 6.25,
 6.5, 6.75, 7, 7.25, 7.5, 7.75, 8, 8.25, 8.5, 8.75, 9, 9.25, 9.5, 9.75, 10, 10.25, 10.5,
 10.75, 11, 11.25, 11.5, 11.75, 12, 12.25, 12.5, 12.75, 13, 13.25, 13.5, 13.75, 14,
 14.25, 14.5, 14.75, 15, 16, 17, 18, 19, 20, 22, 24, 26, 28, 30, 32, 34, 36, 38, 40] ;
real[int] yvec = [0, 0.125, 0.25, 0.375, 0.5, 0.625, 0.75, 0.875, 1, 1.25, 1.5, 1.75, 2, 2.5,
           3, 4, 5, 6, 7, 8, 9, 10] ;
real uval, vval ;


// THE LINEAR SOLVE FOR THE NONLINEAR ITERATION IS DEFINED AS A ``problem'', WHICH IS A VARIABLE TYPE
// Also note that it assumes everything is on LHS, i.e. set up problem as F(u,p)=0.
// Note that eps*(p,q) term is there as a nonzero placeholder so there are no zeros on the diagonal...this
// has to do with the linear solver, and is small enough to not affect the answer.  If the "eps" was larger,
// especially relative to h, then this term is known as a pressure stabilization and has a damping effect.
problem CNNSE ([u1,u2,p],[v1,v2,q],solver=UMFPACK) =
    int2d(Th)(
              1/dt * ( u1*v1 + u2*v2 )
             - p * ( dx(v1) + dy(v2) )
             + 0.5* nu * ( dx(u1)*dx(v1) + dy(u1)*dy(v1) + dx(u2)*dx(v2) + dy(u2)*dy(v2) )
     + 0.125 * (v1 * (u1 * dx(up1) + u2 * dy(up1)) + v2 * (u1 * dx(up2) + u2 * dy(up2)))
    - 0.125 * (up1 * (u1 * dx(v1) + u2 * dy(v1)) + up2 * (u1 * dx(v2) + u2 * dy(v2)))
+ 0.125 * (v1 * (up1 * dx(u1) + up2 * dy(u1)) + v2 * (up1 * dx(u2) + up2 * dy(u2)))
- 0.125 * (u1 * (up1 * dx(v1) + up2 * dy(v1)) + u2 * (up1 * dx(v2) + up2 * dy(v2)))
   + 0.125 *  (v1 * (u1 * dx(un1) + u2 * dy(un1)) + v2 * (u1 * dx(un2) + u2 * dy(un2)))
- 0.125 *  (un1 * (u1 * dx(v1) + u2 * dy(v1)) + un2 * (u1 * dx(v2) + u2 * dy(v2)))
+ 0.125 *  (v1 * (un1 * dx(u1) + un2 * dy(u1)) + v2 * (un1 * dx(u2) + un2 * dy(u2)))
- 0.125 *  (u1 * (un1 * dx(v1) + un2 * dy(v1)) + u2 * (un1 * dx(v2) + un2 * dy(v2)))
+ q * (dx(u1) + dy(u2))
- p*q*(0.000001)
    )
    + int2d(Th)(
-1.0 / dt * (un1 * v1 + un2*v2)
```

```
+ 0.125 * (v1 * (un1 * dx(un1) + un2 * dy(un1)) + v2 * (un1 * dx(un2) + un2 * dy(un2)))
- 0.125 * (un1 * (un1 * dx(v1) + un2 * dy(v1)) + un2 * (un1 * dx(v2) + un2 * dy(v2)))
- 0.125 * (v1 * (up1 * dx(up1) + up2 * dy(up1)) + v2 * (up1 * dx(up2) + up2 * dy(up2)))
+ 0.125 * (up1 * (up1 * dx(v1) + up2 * dy(v1)) + up2 * (up1 * dx(v2) + up2 * dy(v2)))
+ 0.5 * nu * ( dx(un1)*dx(v1) + dy(un1)*dy(v1) + dx(un2)*dx(v2) + dy(un2)*dy(v2) )
    )
  + on(6,8,u1=g,u2=0)
  + on(1,2,3,4,5,7,9,10,u1=0,u2=0) ;

// "problem" definition complete.  It will be called at each iteration in every timestep.

//timestep, total number of iterations, viscosity definitions
dt = 0.01;
int nbiter = 4000;
nu=1./reylnods;

// This is the main loop over the timesteps
for (iter=1;iter<=nbiter;iter++)
{
   cout<< "iter = " << iter << endl ;

   //The "new" velocity at the last time step is the current u1,u2
   un1 = u1;
   un2 = u2;

   // keep track of iterations, LEO SAYS: CAN'T WE JUST DO THIS WITH itnl?
   err1 = 1;

// This is the nonlinear iteration for this timestep.
   for (itnl=1; itnl<=MaxNlIt; itnl++)
   {
        cout<< "itnl = " << itnl << endl ;

        // define the velocity for the previous iterate
up1 = u1;
up2 = u2;

        // Solve the linear system
CNNSE;

// determine the difference between successive iterates and check for convergence
       e1 = abs(u1 - up1);
       e2 = abs(u2 - up2);
       cout<< "e1[].max " << e1[].max << "     e2[].max " << e2[].max<< endl ;
       cout<< " " << endl ;
       if ( (e1[].max < 0.00001) && (e2[].max < 0.00001))
        {
           break ;
        }
        err1 = err1 + 1 ;
   }

// CHECK THAT NONLINEAR ITERATION CONVERGED.  IF NOT, BREAK/EXIT
   if ( err1 >= MaxNlIt )
   {
```

```
      cout << "Nonlinear iteration failed for iter = " << iter << endl ;
      break;
   }

//-----------------------------------------------------------------------------------------
//
if (iter==1 || iter==500 || iter==1000 || iter==1500 || iter==2000 || iter==2500 || iter==3000 ||
                     iter==3500 || iter==4000 ){
// after these particular data sets, print the data to a file.

        ofstream myn("NSE_DataCN_L3_"+iter) ;

// we print it out twice as a placeholder
        myn << xvec.n  << "  " << yvec.n << " " << xvec.n  << "  " << yvec.n << endl;

        for (int ix = 0; ix < xvec.n ; ix++)
        {
        for (int iy = 0; iy < yvec.n ; iy++)
        {
         if ( (yvec[iy] < 1) && ( (xvec[ix] > 5) &&
                 (xvec[ix] < 6) ) )
            {
              uval = 0.0 ;
              vval = 0.0 ;
            }
         else
            {
              uval = u1(xvec[ix],yvec[iy]) ;
              vval = u2(xvec[ix],yvec[iy]) ;
            }

         myn << xvec[ix] << "  " << yvec[iy]
           << "  " << uval << "  " << vval << endl;
        }
        }
     }
//-------------------------------------------------------------------------------------------
 }

//plot(coef=0.05,cmm=" [u1,u2] ",[u1,u2]);

cout << "CPU " << clock()-s0 << "s " << endl;
cout << "Degrees of freedom " << u1[].n + u2[].n + p[].n << endl;
cout << Pressure DOF: " << p[].n << endl;
```

Appendix B FreeFEM++ Implementation of the Leray-α Model

We use the same code as for the Navier-Stokes equations, but we change the problem definition for each model.

```
// THE LINEAR SOLVE FOR THE NONLINEAR ITERATION IS DEFINED AS A ‘‘problem’’, WHICH IS A VARIABLE TYPE
// Also note that it assumes everything is on LHS, i.e. set up problem as F(u,p)=0.
// Note that eps*(p,q) term is there as a nonzero placeholder so there are no zeros on the diagonal...this
// has to do with the linear solver, and is small enough to not affect the answer.  If the "eps" was larger,
// especially relative to h, then this term is known as a pressure stabilization and has a damping effect.



problem filter ([w1,w2],[v1,v2],solver=UMFPACK) =
   int2d(Th)(
//  a^2  (grad w,grad x) + (w,x) - (u,x)
     a^(2) * ( dx(w1)*dx(v1) + dx(w2)*dx(v2) + dy(w1)*dy(v1) + dy(w2)*dy(v2) )
// (w,v)
        + (w1*v1 + w2*v2)
    )
    + int2d(Th)(
// -(u,v)
- (u1*v1 + u2*v2)
    )
  + on(6,8,w1=g,w2=0)
  + on(1,2,3,4,5,7,9,10,w1=0,w2=0);



problem CNNSE ([u1,u2,w1,w2,p],[v1,v2,x1,x2,q],solver=UMFPACK) =
    int2d(Th)(
// 1/dt (u,v)
              1/dt * ( u1*v1 + u2*v2 )
// -(p,div v)
             - p * ( dx(v1) + dy(v2) )
// nu/2 (grad u,grad v)
             + 0.5* nu * ( dx(u1)*dx(v1) + dy(u1)*dy(v1) + dx(u2)*dx(v2) + dy(u2)*dy(v2) )
// b*(w,uprev_it,v)
     + 0.125 * (v1 * (w1 * dx(up1) + w2 * dy(up1)) + v2 * (w1 * dx(up2) + w2 * dy(up2)))
    - 0.125 * (up1 * (w1 * dx(v1) + w2 * dy(v1)) + up2 * (w1 * dx(v2) + w2 * dy(v2)))
// b*(wprev_it,u,v)
+ 0.125 * (v1 * (wp1 * dx(u1) + wp2 * dy(u1)) + v2 * (wp1 * dx(u2) + wp2 * dy(u2)))
- 0.125 * (u1 * (wp1 * dx(v1) + wp2 * dy(v1)) + u2 * (wp1 * dx(v2) + wp2 * dy(v2)))
// b*(w,u_prevtime,v)
   + 0.125 *  (v1 * (w1 * dx(un1) + w2 * dy(un1)) + v2 * (w1 * dx(un2) + w2 * dy(un2)))
- 0.125 *  (un1 * (w1 * dx(v1) + w2 * dy(v1)) + un2 * (w1 * dx(v2) + w2 * dy(v2)))
// b*(w_prevtime,u,v)
+ 0.125 *  (v1 * (wn1 * dx(u1) + wn2 * dy(u1)) + v2 * (wn1 * dx(u2) + wn2 * dy(u2)))
- 0.125 *  (u1 * (wn1 * dx(v1) + wn2 * dy(v1)) + u2 * (wn1 * dx(v2) + wn2 * dy(v2)))
// (div u,q)
+ q * (dx(u1) + dy(u2))
// a^2  (grad w,grad x) + (w,x) - (u,x)
+ a^(2) * (dx(w1)*dx(x1) + dy(w1)*dy(x1) + dx(w2)*dx(x2) + dy(w2)*dy(x2)) + (w1*x1 + w2*x2) - (u1*x1 + u2*x2)
// -eps(p,q)
- p*q*(0.000001)
    )
```

```
    + int2d(Th)(
// - 1/dt * (u_prevtime,v)
-1.0 / dt * (un1*v1 + un2*v2)
// b*(wprevtime,uprevtime,v)
+ 0.125 * (v1 * (wn1 * dx(un1) + wn2 * dy(un1)) + v2 * (wn1 * dx(un2) + wn2 * dy(un2)))
- 0.125 * (un1 * (wn1 * dx(v1) + wn2 * dy(v1)) + un2 * (wn1 * dx(v2) + wn2 * dy(v2)))
// b*(wprev_it,uprev_it,v)
- 0.125 * (v1 * (wp1 * dx(up1) + wp2 * dy(up1)) + v2 * (wp1 * dx(up2) + wp2 * dy(up2)))
+ 0.125 * (up1 * (wp1 * dx(v1) + wp2 * dy(v1)) + up2 * (wp1 * dx(v2) + wp2 * dy(v2)))
// nu/2 (grad uprevtime,grad v)
+ 0.5 * nu * ( dx(un1)*dx(v1) + dy(un1)*dy(v1) + dx(un2)*dx(v2) + dy(un2)*dy(v2) )
    )
  + on(6,8,u1=g,u2=0,w1=g,w2=0)
  + on(1,2,3,4,5,7,9,10,u1=0,u2=0,w1=0,w2=0) ;

// "problem" definition complete.  It will be called at each iteration in every timestep.
```

Appendix C FreeFEM++ Implementation of the Modified Leray-α Model

```
// THE LINEAR SOLVE FOR THE NONLINEAR ITERATION IS DEFINED AS A ``problem'', WHICH IS A VARIABLE TYPE
// Also note that it assumes everything is on LHS, i.e. set up problem as F(u,p)=0.
// Note that eps*(p,q) term is there as a nonzero placeholder so there are no zeros on the diagonal...this
// has to do with the linear solver, and is small enough to not affect the answer.  If the "eps" was larger,
// especially relative to h, then this term is known as a pressure stabilization and has a damping effect.



problem filter ([w1,w2],[v1,v2],solver=UMFPACK) =
   int2d(Th)(
//  a^2  (grad w,grad x) + (w,x) - (u,x)
     a^(2) * ( dx(w1)*dx(v1) + dx(w2)*dx(v2) + dy(w1)*dy(v1) + dy(w2)*dy(v2) )
// (w,v)
        + (w1*v1 + w2*v2)
    )
    + int2d(Th)(
// -(u,v)
- (u1*v1 + u2*v2)
    )
  + on(6,8,w1=g,w2=0)
  + on(1,2,3,4,5,7,9,10,w1=0,w2=0);



problem CNNSE ([u1,u2,w1,w2,p,L],[v1,v2,x1,x2,q,r],solver=UMFPACK) =
    int2d(Th)(
// 1/dt (u,v)
              1/dt * ( u1*v1 + u2*v2 )
// -(p,div v)
             - p * ( dx(v1) + dy(v2) )
// nu/2 (grad u,grad v)
             + 0.5* nu * ( dx(u1)*dx(v1) + dy(u1)*dy(v1) + dx(u2)*dx(v2) + dy(u2)*dy(v2) )
// b*(w,uprev_it,v)

          // Leray
         /*  + 0.125 * (v1 * (w1 * dx(up1) + w2 * dy(up1)) + v2 * (w1 * dx(up2) + w2 * dy(up2)))
    - 0.125 * (up1 * (w1 * dx(v1) + w2 * dy(v1)) + up2 * (w1 * dx(v2) + w2 * dy(v2)))
// b*(wprev_it,u,v)
+ 0.125 * (v1 * (wp1 * dx(u1) + wp2 * dy(u1)) + v2 * (wp1 * dx(u2) + wp2 * dy(u2)))
- 0.125 * (u1 * (wp1 * dx(v1) + wp2 * dy(v1)) + u2 * (wp1 * dx(v2) + wp2 * dy(v2)))
// b*(w,u_prevtime,v)
   + 0.125 *  (v1 * (w1 * dx(un1) + w2 * dy(un1)) + v2 * (w1 * dx(un2) + w2 * dy(un2)))
- 0.125 *  (un1 * (w1 * dx(v1) + w2 * dy(v1)) + un2 * (w1 * dx(v2) + w2 * dy(v2)))
// b*(w_prevtime,u,v)
+ 0.125 *  (v1 * (wn1 * dx(u1) + wn2 * dy(u1)) + v2 * (wn1 * dx(u2) + wn2 * dy(u2)))
- 0.125 *  (u1 * (wn1 * dx(v1) + wn2 * dy(v1)) + u2 * (wn1 * dx(v2) + wn2 * dy(v2)))  */



         // b*(u,wprev,v)
                +0.125 * ( v1 * (u1 * dx(wp1) + u2 * dy(wp1)) + v2 * (u1 * dx(wp2) + u2 * dy(wp2)))
                -0.125 * (wp1 * (u1 * dx(v1) + u2 * dy(v1)) + wp2 * (u1 * dx(v2) + u2 * dy(v2)))
```

```
        // b*(uprev,w,v)
                +0.125 * (v1 * (up1 * dx(w1) + up2 * dy(w1)) + v2 * (up1 * dx(w2) + up2 * dy(w2)))
                -0.125 * (w1 * (up1 * dx(v1) +up2 * dy(v1)) + w2 * (up1 * dx(v2) + up2 * dy(v2)))
        // b*(u,wknown,v)
                +0.125 * (v1 * (u1 * dx(wn1) + u2 * dy(wn1)) + v2 * (u1 * dx(wn2) + u2 * dy(wn2)))
                -0.125 * (wn1 * (u1 * dx(v1) + u2 * dy(v1)) + wn2 * (u1 * dx(v2) + u2 * dy(v2)))
        // b*(uknown,w,v)
                +0.125 * (v1 * (un1 * dx(w1) + un2 * dy(w1)) + v2 * (un1 * dx(w2) + un2 * dy(w2)))
                -0.125 * (w1 * (un1 * dx(v1) + un2 * dy(v1)) + w2 * (un1 * dx(v2) + un2 * dy(v2)))
// (div u,q)
+ q * (dx(u1) + dy(u2))
// a^2  (grad w,grad x) + (w,x) - (u,x)
+ dx(x1)*L + dy(x2)*L+ a^(2) * (dx(w1)*dx(x1) + dy(w1)*dy(x1) + dx(w2)*dx(x2) + dy(w2)*dy(x2)) + (w1*x1 + w2*x2
// (div w,r)
+dx(w1)*r + dy(w2)*r
// +eps(p,q)
         + p*q*(0.000001)
        // +eps(l,r)
         + L*r*(0.000001)
    )
    + int2d(Th)(
// - 1/dt * (u_prevtime,v)
-1.0 / dt * (un1*v1 + un2*v2)
// b*(uprevtime,wprevtime,v)
+ 0.125 * (v1 * (un1 * dx(wn1) + un2 * dy(wn1)) + v2 * (un1 * dx(wn2) + un2 * dy(wn2)))
- 0.125 * (wn1 * (un1 * dx(v1) + un2 * dy(v1)) + wn2 * (un1 * dx(v2) + un2 * dy(v2)))
// b*(uprev_it,wprev_it,v)
- 0.125 * (v1 * (up1 * dx(wp1) + up2 * dy(wp1)) + v2 * (up1 * dx(wp2) + up2 * dy(wp2)))
+ 0.125 * (wp1 * (up1 * dx(v1) + up2 * dy(v1)) + wp2 * (up1 * dx(v2) + up2 * dy(v2)))
// nu/2 (grad uprevtime,grad v)
+ 0.5 * nu * ( dx(un1)*dx(v1) + dy(un1)*dy(v1) + dx(un2)*dx(v2) + dy(un2)*dy(v2) )
    )
  + on(6,8,u1=g,u2=0,w1=g,w2=0)
  + on(1,2,3,4,5,7,9,10,u1=0,u2=0,w1=0,w2=0) ;

// "problem" definition complete.  It will be called at each iteration in every timestep.
```

Appendix D FreeFEM++ Implementation of the NS-α Model

```
// THE LINEAR SOLVE FOR THE NONLINEAR ITERATION IS DEFINED AS A ‘‘problem’’, WHICH IS A VARIABLE TYPE
// Also note that it assumes everything is on LHS, i.e. set up problem as F(u,p)=0.
// Note that eps*(p,q) term is there as a nonzero placeholder so there are no zeros on the diagonal...this
// has to do with the linear solver, and is small enough to not affect the answer.  If the "eps" was larger,
// especially relative to h, then this term is known as a pressure stabilization and has a damping effect.



problem filter ([w1,w2],[v1,v2],solver=UMFPACK) =
   int2d(Th)(
//  a^2  (grad w,grad x) + (w,x) - (u,x)
     a^(2) * ( dx(w1)*dx(v1) + dx(w2)*dx(v2) + dy(w1)*dy(v1) + dy(w2)*dy(v2) )
// (w,v)
        + (w1*v1 + w2*v2)
    )
    + int2d(Th)(
// -(u,v)
- (u1*v1 + u2*v2)
    )
  + on(6,8,w1=g,w2=0)
  + on(1,2,3,4,5,7,9,10,w1=0,w2=0);



problem CNNSE ([u1,u2,w1,w2,p],[v1,v2,x1,x2,q],solver=UMFPACK) =
    int2d(Th)(
// 1/dt (u,v)
              1/dt * ( u1*v1 + u2*v2 )
// -(p,div v)
             - p * ( dx(v1) + dy(v2) )
// nu/2 (grad u,grad v)
             + 0.5* nu * ( dx(u1)*dx(v1) + dy(u1)*dy(v1) + dx(u2)*dx(v2) + dy(u2)*dy(v2) )
// -(u x curl wprevtime ,v)
      + 0.25 * (v1 * (-w2 * dx(up2) + w2 * dy(up1)) + v2 * (w1 * dx(up2) - w1 * dy(up1)))
        // -(uprevtime x curl w ,v)
             + 0.25 * (v1 * (-wp2 * dx(u2) + wp2 * dy(u1)) + v2 * (wp1 * dx(u2) - wp1 * dy(u1)))
// -(uknown x curl w, v)
        + 0.25 * (v1 * (-wn2 * dx(u2) + wn2 * dy(u1)) + v2 * (wn1 * dx(u2) - wn1 * dy(u1)))
// -(u x curl wknown, v)
     + 0.25 * (v1 * (-w2 * dx(un2) + w2 * dy(un1)) + v2 * (w1 * dx(un2) - w1 * dy(un1)))
// (div u,q)
+ q * (dx(u1) + dy(u2))
// (div u, div v)
+ 0.5 * (dx(u1) + dy(u2)) * (dx(v1) + dy(v2))
// a^2  (grad w,grad x) + (w,x) - (u,x)
            + a^(2) * (dx(w1)*dx(x1) + dy(w1)*dy(x1) + dx(w2)*dx(x2) + dy(w2)*dy(x2)) + (w1*x1 + w2*x2) - (u1*x
// -eps(p,q)
- p*q*(0.000001)
    )
    + int2d(Th)(
// - 1/dt * (u_prevtime,v)
-1.0 / dt * (un1*v1 + un2*v2)
// (uprevtime x curl wprevtime,v)
- 0.25 * (v1 * (-wp2 * dx(up2) + wp2 * dy(up1)) + v2 * (wp1 * dx(up2) - wp1 * dy(up1)))
```

```
// b*(wprev_it,uprev_it,v)
+ 0.25 * (v1 * (wn2 * dx(un2) + wn2 * dy(un1)) + v2 * (wn1 * dx(un2) - wn1 * dy(un1)))
// nu/2 (grad uprevtime,grad v)
+ 0.5 * nu * ( dx(un1)*dx(v1) + dy(un1)*dy(v1) + dx(un2)*dx(v2) + dy(un2)*dy(v2) )
// (div u, div v)
+ 0.5 * ( dx(un1) + dy(un2) ) * (dx(v1) + dy(v2))
    )
  + on(6,8,u1=g,u2=0,w1=g,w2=0)
  + on(1,2,3,4,5,7,9,10,u1=0,u2=0,w1=0,w2=0) ;

// "problem" definition complete.  It will be called at each iteration in every timestep.
```

Appendix E FreeFEM++ Implementation of the NS-ω Model

```
// THE LINEAR SOLVE FOR THE NONLINEAR ITERATION IS DEFINED AS A ‘‘problem’’, WHICH IS A VARIABLE TYPE
// Also note that it assumes everything is on LHS, i.e. set up problem as F(u,p)=0.
// Note that eps*(p,q) term is there as a nonzero placeholder so there are no zeros on the diagonal...this
// has to do with the linear solver, and is small enough to not affect the answer.  If the "eps" was larger,
// especially relative to h, then this term is known as a pressure stabilization and has a damping effect.

problem filter ([w1,w2],[v1,v2],solver=UMFPACK) =
   int2d(Th)(
//  a^2  (grad w,grad x) + (w,x) - (u,x)
     a^(2) * ( dx(w1)*dx(v1) + dx(w2)*dx(v2) + dy(w1)*dy(v1) + dy(w2)*dy(v2) )
// (w,v)
        + (w1*v1 + w2*v2)
    )
    + int2d(Th)(
// -(u,v)
- (u1*v1 + u2*v2)
    )
  + on(6,8,w1=g,w2=0)
  + on(1,2,3,4,5,7,9,10,w1=0,w2=0);

problem CNNSE ([u1,u2,w1,w2,p],[v1,v2,x1,x2,q],solver=UMFPACK) =
    int2d(Th)(
// 1/dt (u,v)
              1/dt * ( u1*v1 + u2*v2 )
// -(p,div v)
             - p * ( dx(v1) + dy(v2) )
// nu/2 (grad u,grad v)
             + 0.5* nu * ( dx(u1)*dx(v1) + dy(u1)*dy(v1) + dx(u2)*dx(v2) + dy(u2)*dy(v2) )
// -(u x curl wprevtime ,v)
      + 0.25 * (v1 * (-u2 * dx(wp2) + u2 * dy(wp1)) + v2 * (u1 * dx(wp2) - u1 * dy(wp1)))
        // -(uprevtime x curl w ,v)
             + 0.25 * (v1 * (-up2 * dx(w2) + up2 * dy(w1)) + v2 * (up1 * dx(w2) - up1 * dy(w1)))
// -(uknown x curl w, v)
        + 0.25 * (v1 * (-un2 * dx(w2) + un2 * dy(w1)) + v2 * (un1 * dx(w2) - un1 * dy(w1)))
// -(u x curl wknown, v)
     + 0.25 * (v1 * (-u2 * dx(wn2) + u2 * dy(wn1)) + v2 * (u1 * dx(wn2) - u1 * dy(wn1)))
// (div u,q)
+ q * (dx(u1) + dy(u2))
// (div u, div v)
+ 0.5 * (dx(u1) + dy(u2)) * (dx(v1) + dy(v2))
// a^2  (grad w,grad x) + (w,x) - (u,x)
            + a^(2) * (dx(w1)*dx(x1) + dy(w1)*dy(x1) + dx(w2)*dx(x2) + dy(w2)*dy(x2)) + (w1*x1 + w2*x2) - (u1*x
// -eps(p,q)
- p*q*(0.000001)
    )
    + int2d(Th)(
// - 1/dt * (u_prevtime,v)
-1.0 / dt * (un1*v1 + un2*v2)
// (uprevtime x curl wprevtime,v)
- 0.25 * (v1 * (-up2 * dx(wp2) + up2 * dy(wp1)) + v2 * (up1 * dx(wp2) - up1 * dy(wp1)))
// b*(wprev_it,uprev_it,v)
+ 0.25 * (v1 * (un2 * dx(wn2) + un2 * dy(wn1)) + v2 * (un1 * dx(wn2) - un1 * dy(wn1)))
// nu/2 (grad uprevtime,grad v)
+ 0.5 * nu * ( dx(un1)*dx(v1) + dy(un1)*dy(v1) + dx(un2)*dx(v2) + dy(un2)*dy(v2) )
```

```
// (div u, div v)
+ 0.5 * ( dx(un1) + dy(un2) ) * (dx(v1) + dy(v2))
    )
  + on(6,8,u1=g,u2=0,w1=g,w2=0)
  + on(1,2,3,4,5,7,9,10,u1=0,u2=0,w1=0,w2=0) ;

// "problem" definition complete.  It will be called at each iteration in every timestep.
```

Bibliography

[1] E.S. Titi A.A. Ilyin, E.M. Lunasin. A modified-Leray-α sub-grid scale model of turbulence. *Nonlinearity*, 19:879–897, 2006.

[2] A. Cheskidov, D. Holm, E. Olson, and E. Titi. On a Leray-alpha model of turbulence. *Royal Society London, Proceedings, Series A, Mathematical, Physical and Engineering Sciences*, pages 629–649, 2005.

[3] C. Foias, D. Holm, and E. Titi. The Navier-Stokes-alpha model of fluid turbulence. *Physica D*, pages 505–519, May 2001.

[4] C. Foias, D. Holm, and E. Titi. The three dimensional viscous Camassa-Holm equations, and their relation to the Navier-Stokes equations and turbulence theory. *Journal of Dynamics and Differential Equations*, 14:1–35, 2002.

[5] P. Gresho and R. Sani. *Incompressible Flow and the Finite Element Method*, volume 2. Wiley, 1998.

[6] M. Gunzburger. *Finite Element Methods for Viscous Incompressible Flow: A Guide to Theory, Practice, and Algorithms*. Academic Press, Boston, 1989.

[7] F. Hecht, O. Pironneau, and K. Ohtsuka. Software freefem++. http://www.freefem.org, 2005.

[8] W. Layton. *Introduction to the numerical analysis of incompressible viscous flows*. SIAM, 2008.

[9] W. Layton, C. Manica, M. Neda, and L. Rebholz. Numerical analysis and computational comparisons of the NS-omega and NS-alpha regularizations. *Computer Methods Applied Mechanics and Engineering*, 199:916–931, 2010.

[10] J. Leray. Sur le mouvemennt d'un fluide visquex emplissant l'espace. *Acta. Math.*, 63:193–248, 1934.

[11] M. Olshanskii and A. Reusken. Grad-div stabilization for stokes equations. *Mathematics of Computation*, 73(248):1699–1718, 2003.

[12] L. Rebholz. Conservation laws of turbulence models. *Journal of Mathematical Analysis and Applications*, 326(1):33–45, 2007.

[13] L. Rebholz. A family of new high order NS-alpha models arising from helicity correction in Leray turbulence models. *Journal of Mathematical Analysis and Applications*, 342(1):246–254, 2008.

[14] L. Rebholz and M. Sussman. On the high accuracy NS-α-deconvolution model of turbulence. *Mathematical Models and Methods in Applied Sciences*, to appear, 2010.

[15] Leo G. Rebholz. *Helicity and Physical Fidelity in Turbulence Modeling*. PhD thesis, University of Pittsburgh, 2006.

[16] C. Trenchea W.J. Layton, I. Stanculescu. Theory of the NS-$\overline{\omega}$ model: A complement to the NS-α model. *Technical Report, University of Pittsburgh*, 2008.