2017

Reynolds-averaged Navier-Stokes simulation of turbulent flow in a circular pipe using OpenFOAM®

BOSTON UNIVERSITY

COLLEGE OF ENGINEERING

Thesis

REYNOLDS-AVERAGED NAVIER-STOKES SIMULATION OF TURBULENT FLOW IN A CIRCULAR PIPE USING OPENFOAM®

by

TIMOTHY DAVID STUER

B.S., Northeastern University, 2013

Submitted in partial fulfillment of the

requirements for the degree of

Master of Science

2017

Approved by

First Reader

Sheryl M. Grace, Ph.D.
Associate Professor of Mechanical Engineering

Second Reader

Emily Ryan, Ph.D.
Assistant Professor of Mechanical Engineering
Assistant Professor of Materials Science and Engineering

Third Reader

Douglas Sondak, Ph.D.
Associate Professor of Mechanical Engineering
Wentworth Institute of Technology

ACKNOWLEDGEMENTS

There are many people I would like to express gratitude for in support of this undertaking. First I would like to sincerely thank my advisor Sheryl Grace. Without her, none of this would have been possible. She has been a source of strong guidance and encouragement throughout this process.

I would also like to express my appreciation to Eric Patterson for his support in learning the OpenFOAM® tool. Your tips and guidance were vital to the work I have done. In addition I would like to thank Emily Ryan and Doug Sondak for being a part of my defense committee.

I would like to thank GE Aviation for the sponsorship and providing me with the opportunity to purse this degree. I would also like to thank Boston University for the support it has provided me in this pursuit.

Next, I would like to thank all of my friends for their support over the past few years. They have provided me a wall to bounce ideas off of, company to commiserate with, and a source of great fun during this effort. I'd like to extend a special thanks to David "the Wag" Wagner for undertaking this adventure with me when few others would. Without your support this would not have been possible.

Finally, I would like to thank my family. They have been a constant positive influence in my life and they have been overwhelmingly supportive and loving as I have worked my way through the past couple of years.

REYNOLDS-AVERAGED NAVIER-STOKES SIMULATION OF TURBULENT FLOW IN A CIRCULAR PIPE USING OPENFOAM®

ABSTRACT

A RANS simulation of flow through a pipe is performed and validated against experimental data and previous DNS results. A mesh refinement study is performed to illustrate the near wall mesh size needed to correctly predict mean flow characteristics. In addition, aspects of the model are changed to study their impact on the results as well as the computational requirements. Comparisons are made between a two-dimensional analysis with axisymmetric boundary conditions, a one-eighth axisymmetric model, a one-fourth axisymmetric model, and a full three-dimensional pipe. The two-dimensional model provides the best match to past data; however, it is noted that the model may not be well tuned for a three-dimensional mesh. The simulation is also performed using three different turbulence models and the results of each model are compared. The purpose of the model is to create a tool that can be used for design iterations. While the model does not fully capture the complexities of turbulent flow, it is able to predict the mean flow accurately enough to be useful in a design setting. The goal of this work is to create a foundation upon which further studies of pipe flow with internal obstructions can build. The overall results show the model is able to predict the mean flow well for the validation case. However, the model does not perform well when certain aspects are changed. Increasing the robustness of the model and the determination of more usable boundary conditions remains a subject for future studies.

CONTENTS

LIST OF TABLES

LIST OF FIGURES

1 INTRODUCTION

Internal flow is a common occurrence and is seen in many different applications. Heating and cooling systems, water pipes, and combustion engines are just a few examples that involve internal flow which are used in everyday life. Because it is so widely used, it is important to understand the nature of internal flows and be able to analyze them. Of the different situations for internal flow, flow through a circular duct is one of the most fundamental.

Because circular duct flow is a common scenario seen in fluids, it is of great interest to be able to easily model this flow and predict the effects of duct geometry, flow obstructions, and many other parameters. In laminar flow situations with simple geometries, like water flowing through an empty pipe at low velocity, these problems can typically be solved analytically and do not require computational fluid dynamics; however most flows of interest involve complex geometries or are turbulent in nature. Take for example air flowing into the turbine engine of a typical commercial airliner flying at a cruise Mach number of 0.80. This flow has a Reynolds number greater that $5x10^7$ making it difficult to evaluate. This is because when the flow becomes turbulent, analytical methods do not accurately predict the flow. Either experimentation or computational methods are required to study the problem. Experimentation is a good way to study the flow, however it can be expensive and is limited in the parameters that can be varied. Computational methods are the most inexpensive method for resolving turbulent flows and lend well to changing a wide range of parameters to determine the effect.

In this thesis an open source CFD tool is used to simulate flow in a pipe. Validation against both experiment and other pipe flow simulations found in the literature are presented. The goal of this work is to understand the OpenFOAM® program, test various boundary conditions, and gain mastery of grid generation. A main interest is to create a model that can determine the loss if an object is in the pipe.

The layout of this thesis begins with background information on computational fluid dynamics including the different methods, fundamental equations, and turbulence models. The thesis continues with an overview of existing literature on channel and pipe flow. Next, the general setup of the model is discussed including the geometry, mesh, and boundary conditions. The verification of the model to laminar pipe flow is performed followed by validation of the model to experimental data and results from other simulations. The work then discusses a few aspects of the model that were studied including the turbulence model used and the axisymmetric assumption. The thesis concludes by discussing the lessons gleaned from this work and suggests future work that can be undertaken.

1.1 Computational Fluid Dynamics Methods

Within the computational methods, there are multiple models that can be used. Lagrangian based methods follow a particle or collection of particles as they move about the system (Muller, 2016). Types of Lagrangian based methods include the Lattice Boltzman method (LBM) and smooth particle hydrodynamics (SPH). LBM does not use the Navier-Stokes equations, rather it utilizes collision models and the Boltzmann equation to determine how the fluid behaves. SPH is a method that does not use a mesh

to computationally solve the flow. Instead, it works by solving the interaction of individual particles (Magoules, 2011).

Eulerian based methods determine a certain space to observe and the flow of particles in and out of this space is studied (Muller, 2016). Within the Eulerian framework there exists the finite difference method (FDM), the finite element method (FEM) and the finite volume method (FVM). For FDM a set of grid points is established and the Navier-Stokes equations are satisfied at each grid point. FEM breaks up the domain into small elements and the Navier-Stokes equations are then used to define the fluid physics for each element. The FVM also subdivides the domain for the flow geometry, but it treats each domain as a separate control volume. At each time step each of the small control volumes are used to satisfy the conservation equations (Magoules, 2011).

In addition, simulations can be performed at different resolutions. Direct numerical simulation (DNS) directly discretizes the Navier Stokes equations in order to resolve the flow fully at all time and length scales. This method requires a very fine mesh in areas of high Reynolds number flow increasing the time required to solve the model. This type of simulation is useful because it allows many flow statistics to be studied in a relatively easy manner. This offers a benefit over experiment because measurement methods do not need to be developed to study a certain aspect of the flow. The limitation of DNS is in the computational expense. DNS requires a very fine grid in order to predict and study turbulence. Because of this, DNS becomes computationally expensive at moderately high Reynolds numbers. The example given earlier of a flow

into the inlet of the turbine engine of a commercial airliner cannot be solved in a reasonable amount of time using DNS with today's current computation capability. Therefore DNS is most useful for studying simple geometries at low Reynolds numbers. Assuming Moore's Law holds true, it will still be a decades before DNS becomes a useful tool for engineering design with complex geometry or highly turbulent flow (Muller, 2016).

Another method used in CFD is large eddy simulation (LES). In this method, larger eddies are fully resolved while eddies below a certain cut-off threshold are modelled. A large portion of the scales are solved, but not all length scales as is done in DNS. This method also requires a high-resolution mesh to accurately simulate near wall conditions and is computationally expensive.

A third turbulence method used is Reynolds-Average Navier-Stokes (RANS). RANS modeling takes advantage of the fact that often engineering applications are concerned only with the averaged flow quantities. To further explain this method consider the two-dimensional conservation of momentum equation in the x direction.

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y} = \frac{1}{\rho}\frac{\partial p}{\partial x} + \frac{\partial}{\partial x}\left(\nu\frac{\partial u}{\partial x}\right) + \frac{\partial}{\partial y}\left(\nu\frac{\partial u}{\partial y}\right) \quad (1)$$

For the Reynolds-Averaged Navier-Stokes approach, the velocity is decomposed into a time average portion, $\bar{u}$ and a fluctuating portion u′.

$$u = \bar{u} + u' \quad (2)$$

Substituting this into the x direction momentum equation yields

$$\frac{\partial(\bar{u}+u')}{\partial t}+(\bar{u}+u')\frac{\partial(\bar{u}+u')}{\partial x}+v\frac{\partial(\bar{u}+u')}{\partial y} = \frac{1}{\rho}\frac{\partial(\bar{p}+p')}{\partial x}+\frac{\partial}{\partial x}\left(\nu\frac{\partial(\bar{u}+u')}{\partial x}\right)+\frac{\partial}{\partial y}\left(\nu\frac{\partial(\bar{u}+u')}{\partial y}\right) \quad (3)$$

This form of the equation can be simplified by taking the long time average and utilizing the fact that the fluctuations average to zero, $\overline{u'}=\overline{v'}=\overline{w'}=0$. The x direction momentum the becomes

$$\frac{\partial\bar{u}}{\partial t}+\bar{u}\frac{\partial\bar{u}}{\partial x}+\bar{v}\frac{\partial\bar{u}}{\partial y} = \frac{1}{\rho}\frac{\partial\bar{p}}{\partial x}+\frac{\partial}{\partial x}\left(\nu\frac{\partial\bar{u}}{\partial x}\right)+\frac{\partial}{\partial x}\left(\nu\frac{\partial\bar{u}}{\partial y}\right) + \left(-\frac{\partial\overline{u'u'}}{\partial x}-\frac{\partial\overline{u'v'}}{\partial y}-\frac{\partial\overline{u'w'}}{\partial z}\right) \quad (4)$$

The average of the products of the velocity fluctuations, $\overline{u'u'}, \overline{u'v'}, and\ \overline{u'w'}$ are dissipative quantities known as Reynolds stresses. These values are typically modelled in the RANS method using turbulence models, which will be discussed in the next section (White, 2006).

Because RANS modelling is only concerned with the long time average characteristics of the flow, it does not require an extremely fine mesh to determine mean flow statistics. This allows the model to run in a shorter period of time and lends itself well to engineering design. Multiple geometries can be analyzed in a relatively short period of time allowing for optimization.

1.2 Turbulence Models

Within the Reynolds-Averaged Navier-Stokes method, there are many types of turbulence models used to predict the flow. One category of models that is popular is the two-equation model. As mentioned previously the momentum equation can be defined in the form shown in equation 4. In RANS the velocity fluctuations and dissipation of these fluctuations are modeled utilizing transport equations. The turbulent kinetic energy is defined as

$$k = \frac{1}{2}\left(\overline{u'u'} + \overline{v'v'} + \overline{w'w'}\right) \tag{5}$$

In addition, the turbulent dissipation is defined as

$$\epsilon = -\nu \overline{\frac{\partial u'_\iota}{\partial x_j}\frac{\partial u'_j}{\partial x_\iota}} \tag{6}$$

That rate of change of these quantities must be modeled in order to solve the momentum equation. This is where turbulence models are important (White, 2006).

Two widely used turbulence models are the k-ε model and k-ω model. The k-ε model uses one equation for the turbulent kinetic energy and another for the turbulent dissipation. The equation for each are as follows (White, 2006).

Turbulent Kinetic Energy

$$\rho\frac{Dk}{Dt} = \frac{\partial}{\partial y}\left(\frac{\mu_\tau \partial k}{\sigma_k \partial y}\right) + \mu_\tau\left(\frac{\partial u}{\partial y}\right)^2 - \rho\epsilon \tag{7}$$

Energy Dissipation

$$\rho\frac{D\epsilon}{Dt} = \frac{\partial}{\partial y}\left(\frac{\mu_\tau \partial \epsilon}{\sigma_\epsilon \partial y}\right) + c_1\frac{\epsilon}{k}\mu_\tau\left(\frac{\partial \mu}{\partial y}\right)^2 - c_2\frac{\rho\epsilon^2}{k} \tag{8}$$

In these two equations, μ_τ is the turbulent viscosity and is defined as

$$\mu_\tau = \frac{C_\mu \rho k^2}{\epsilon} \tag{9}$$

For these equations, C_μ, C_1, C_2, σ_k, and σ_ε are all empirical constants. There are many different k-ε models that all take the form of the two equations shown. The difference between the models is the values of the constants. The standard k-ε model as described by Jones and Launder (1972) has the constants shown in Table 1.

Table 1 - Empirical Constants for k-ε Model

C_μ	C_1	C_2	σ_k	σ_ε
0.09	1.55	2.0	1.0	1.3

The k-ω model as developed by Wilcox (1988) is defined by the following two equations.

Turbulent Mixing Energy

$$\frac{\partial}{\partial t}(\rho k) + \frac{\partial}{\partial x_j}(\rho u_j k) = \tau_{ij}\frac{\partial u_i}{\partial x_j} - \beta^* \rho \omega k + \frac{\partial}{\partial x_j}\left[(\mu + \sigma^* \mu_\tau)\frac{\partial k}{\partial x_j}\right] \tag{10}$$

Specific Dissipation Rate

$$\frac{\partial}{\partial t}(\rho \omega) + \frac{\partial}{\partial x_j}(\rho u_j \omega) = \left(\frac{\gamma \omega}{k}\right)\tau_{ij}\frac{\partial u_i}{\partial x_j} - \beta \rho \omega^2 + \frac{\partial}{\partial x_j}\left[(\mu + \sigma \mu_\tau)\frac{\partial \omega}{\partial x_j}\right] \tag{11}$$

For the k-ω model, μ_τ is defined as

$$\mu_\tau = \gamma^* \frac{\rho k}{\omega} \tag{12}$$

In addition, the following relationship is defined:

$$\omega = \frac{\epsilon}{\beta^* k} \tag{13}$$

The constants as defined by Wilcox are shown in Table 2.

Table 2 - Wilcox k-ω model constants

γ	β	β^*	σ	σ^*
5/9	3/40	9/100	1/2	1/2

These models have different strengths and weaknesses. The k-ε model does well in the outer region of high Reynolds number flow; however, it is not valid for the viscous layer near the wall. The k-ω model, on the other hand performs well near the wall but is not as good in the outer region. The k-ω SST model combines the k-ε and k-ω models switching between the two depending on the flow region. This should be the best performing model; however, the need to switch between two different models causes stability issues, (Muller, 2016). Many other turbulence models have been developed over the years and several have been implemented in OpenFOAM®, but this work focused on the k-ε, k-ω, and k-ω SST models. These three models are employed in the work described in this thesis.

2 LITERATURE REVIEW

A large amount of literature exists involving turbulent flow in channels and ducts. Many of these works involved predicting the near wall behavior of the flow. This flow regime is of great interest as it has a significant impact on the mean flow statistics. Table 3 gives a summary of the past research that is discussed in this section.

Table 3- Summary of Previous Research

Researcher	Method	Description
Prandtl (1925)	Experiment	Developing scaling equations
Taylor (1916)	Experiment	Developing scaling equations
Von Karman (1939)	Experiment	Developing scaling equations
Spaulding (1961)	Experiment	Developing scaling equations
Coles and Hirst (1968)	Experiment	Coefficients for Spaulding approximation
Kim et al. (1987)	DNS	Flow between two flat plates, Re = 3300
Eggels et al. (1993)	DNS and experiment	Flow in a circular pipe, Re = 5300
Zagarola (1996)	Experiment	Flow in a circular pipe, Re between 10,000 and 35,000,000
Wu and Moin (2008)	DNS	Flow in a circular pipe, Re = 44,000
Boersma (2011)	DNS	Flow in a circular pipe, Re = 61,000
Konig (2015)	Experiment	Flow in a circular pipe Re between 10,000 and 70,000

Earlier researchers determined that turbulent flow over a wall can be divided into three regions. The inner layer, closest to the wall, is dominated by viscous shear. The flow slows near the wall and the impact of turbulence is small compared to the viscous forces of the fluid moving over the wall. The outer layer, furthest from the wall, is dominated by turbulent shear. This is the region far enough away from the wall that the viscous forces are overcome by the turbulence in the fluid. The area of most difficulty when predicting the flow parameters, is the overlap layer. This region is impacted by both viscous shear and turbulent shear, White (2006).

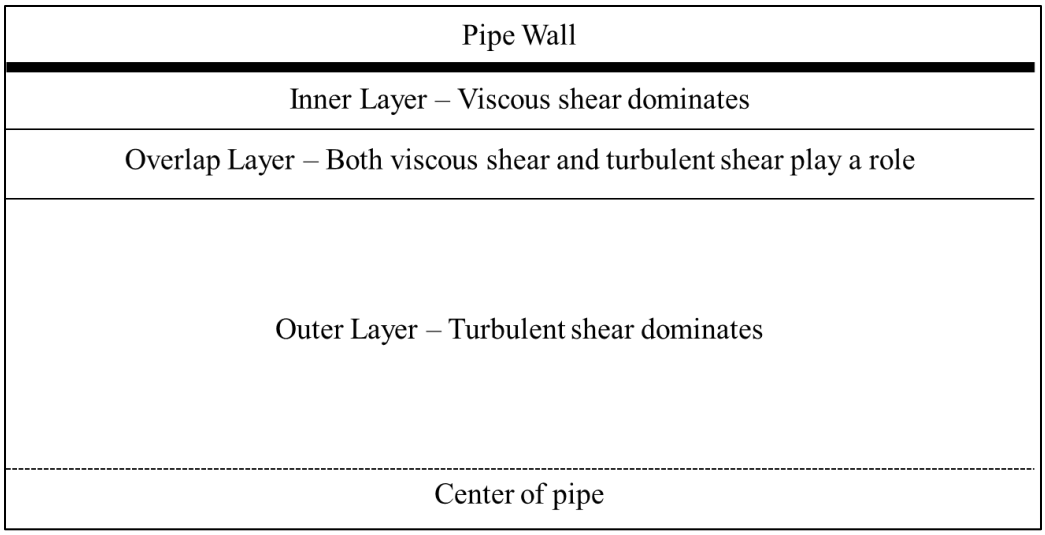


Figure 1 - Flow regions in turbulent pipe flow

Originally, many scaling methods were developed for the velocity profile in these regions that were derived from experiment. These methods related a non-dimensional velocity, u^+, to a non-dimensional distance from the wall, y^+. These characteristics are defined as follows.

$$u^+ = \frac{u}{u^*} \tag{14}$$

$$y^+ = \frac{yu^*}{\nu} \tag{15}$$

Where u^* is the friction velocity defined as

$$u^* = \sqrt{\frac{\tau_w}{\rho}} \tag{16}$$

τ_w is the wall shear stress and is given as

$$\tau_w = \mu\left(\frac{\partial u}{\partial y}\right) \tag{17}$$

Prandtl (1925) suggested that for $y^+ \leq 11.5$ the velocity profile is defined by

$$u^+ = y^+ \tag{18}$$

Taylor found that for $y^+ \geq 11.5$ the velocity profile was defined by

$$u^+ = 2.5\, ln\, y^+ + 5.5 \tag{19}$$

Von Karman (1939) define the velocity profile by three relationships

$$\begin{cases} y^+ \leq 5 & u^+ = y^+ \\ 5 \leq y^+ \leq 30 & u^+ = 5 \ln y^+ - 3.05 \\ y^+ \geq 30 & u^+ = 2.5 \ln y^+ + 5.5 \end{cases} \tag{20}$$

As can be seen, there was overlap among these relationships as they built off each other.

- Spalding (1961) was the first to determine a single formula for the entire wall region.

$$y^+ = u^+ + e^{-\kappa B}\left[e^{\kappa U^+} - 1 - \kappa u^+ - \frac{(\kappa u^+)^2}{2} - \frac{(\kappa u^+)^3}{6}\right] \tag{21}$$

Spalding originally used coefficients of $\kappa = 0.40$ and $B = 5.5$. These were later revised by Coles and Hirst (1968) to $\kappa = 0.41$ and $B = 5.5$. This approximation of the turbulent velocity profile is still widely used today.

Kim et al. (1987) studied channel flow, flow between two flat plates, at Re = 3300 using direct numerical simulation (DNS). The general results from their simulation agreed well with experimental results; however, they did find differences in the near wall region.

Eggels et al. (1993) performed DNS and experimental studies at various Reynolds numbers for flow in a circular pipe and compared the results to plane channel flow. The majority of the work done was at a bulk Re = 5300. They collected data including Reynolds number at the centerline, turbulent kinetic energy, root mean squared velocity, pressure fluctuations, skewness factor, and flatness factor. The velocity profile obtained

from experiment and simulation are qualitatively compared and show good agreement. They also compared the DNS results to previous experiments of channel flow and showed that flow in a circular pipe fails to follow the typical law of the wall for plane channel flow. This is due to the increased viscous shear of the side walls of the pipe. This work provides a good set of data to validate a circular pipe model to at a moderate Reynolds number.

One of the most prominent experimental facilities for high Reynolds number pipe flow is the Princeton SuperPipe. The experimental setup is described by Zagarola (1996). He utilizes the SuperPipe to study a new method of scaling the overlap layer in high Reynolds number pipe flow, different from Spalding's approximation. The range of Reynolds numbers studied by Zagarola is 10,000 to 35,000,000. He determined that the region with y^+ between 50 and 500 can be scaled using power-law dependence and the overlap region with y^+ greater than 500 can be scaled using log-law dependence. His proposed relationships are

$$\begin{cases} 50 \leq y^+ \leq 500 & u^+ = 8.70(y^+)^{.137} \\ y^+ \geq 500 & u^+ = \dfrac{1}{4.36}\ln y + 6.13 \end{cases} \tag{22}$$

Wu and Moin (2008) discovered the need to perform a numerical simulation of pipe flow at a Reynolds number above the minimum threshold of the Princeton Super Pipe. They used a DNS model validated to the data available at Re = 5300 and performed a simulation at Re = 44,000. The statistics from this were well matched to the experimental data from the Princeton Super Pipe. Until this point, data from DNS simulations at higher Reynolds number was not widely available and the connection

between experiment and numerical simulation was typically performed at lower Reynolds numbers. Their work allows for a more direct connection between experiment at higher Reynolds number and numerical simulation of turbulent pipe flow.

Boersma (2011) continued the work of Wu and Moin with a newly developed DNS model. He validated his model with Wu and Moin at RE = 44,000 and then performed a simulation at RE = 61,000. This was yet another increase in Reynolds number where simulation could be compared to experiment.

Konig (2015) designed and constructed the Cottbus Large Pipe to perform experiments studying high Reynolds number circular pipe flow. This setup utilizes hot-wire anemometry to measure the velocity fluctuations needed to determine turbulence statistics. His work documents the complexities in the design and implementation of the experimental study of high Reynolds number pipe flow.

Most of the work performed studying turbulent pipe flow utilizes experiment or DNS modeling. Both of these methods are important when the goal is to further understand the flow characteristics in the near wall region. Unfortunately, these methods are not well suited for engineering design that requires iteration to find an optimum solution to the posed problem. RANS modeling is a better tool to use for this type of study as it is able to run in a shorter amount of time.

3 MODEL SETUP, VERIFICATION, AND VALIDATION

3.1 OpenFOAM® Software

The program used to complete the models in this thesis is OpenFOAM®. OpenFOAM® is an open source code written in C++ that is used for finite volume modeling. The program has a wide range of functionality with various pre-processing, solving, and post-processing tools built into the program. In addition to the prebuilt software, users can write new solvers and other utilities for use in the program.

The use of OpenFOAM® in this thesis is limited to the RANS capability as implemented in the SIMPLE algorithm. Further details of this are provided below.

3.2 The SIMPLE Scheme

While many schemes exist for solving RANS models, the scheme chosen for the model described in this thesis is the Semi-Implicit Method for Pressure-Linked Equations (SIMPLE) scheme. In general, an iterative solver for incompressible flow begins with the momentum equations

$$\frac{\partial \vec{u}}{\partial t} = -\vec{u}\nabla\vec{u} - \frac{1}{\rho}\nabla p + \nu\nabla^2\vec{u} \tag{23}$$

Using a given pressure field, the momentum equations are solved for a velocity field. Then the divergence of the momentum equations is taken, which is the error. This error is pressure corrected using the following equation

$$\nabla \cdot \left(\nabla(p^* + \delta p)\right) = \nabla^2(p^* + \delta p) = \rho\nabla \cdot (-\overrightarrow{u^*}\nabla\overrightarrow{u^*} + \nu\nabla^2\overrightarrow{u^*}) \tag{24}$$

When the pressure correction is applied, the momentum equations are no longer satisfied and the iterations must continue.

In the SIMPLE scheme, an initial pressure field is given to the model for iteration 1. This pressure field is used to solve the momentum equations for a velocity field. Next, the pressure correction equation is solved for Δp. If this Δp is within the specified convergence limit, the scheme is exited, otherwise the pressure is corrected. Finally, the corrected pressure field is used to correct the velocity field. Then the next iteration begins with the corrected pressure field being given to the model.

When this algorithm is used, the flow may be driven by specifying a body force on the flow field. In previous versions of OpenFOAM® the body force pressure gradient could be directly specified as a method for forcing the flow. In this case the bulk velocity achieved is the result of this pressure gradient. In the version of OpenFOAM® used for this work, the body force is now implemented by specifying a desired a bulk flow velocity. Within each time step, the solver will iterate on the pressure field until this bulk flow is achieved. Then the regular two step, pressure correction method is used to obtain the mean velocity, pressure perturbation, and turbulence parameters.

3.2 Geometry and Boundary Conditions

The baseline geometry studied in the simulations is an empty circular pipe with a diameter of 6 inches and a length of 10 feet. The pipe is initially modeled as an axisymmetric 1/72 wedge. The mesh density is variable depending on the flow Reynolds number and is covered in a later section. The fluid of interest for this work is air. Figure 2 shows an example axisymmetric mesh used for the simulation. In many cases, the mesh density near the wall is finer than the mesh density at the center of the pipe. This is to capture the sheer stress effects of the wall accurately.

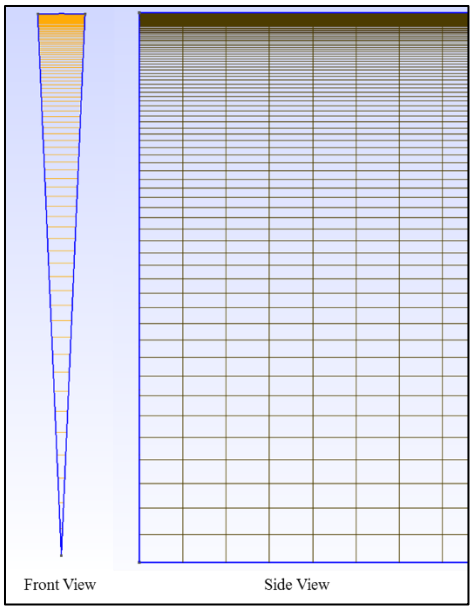


Figure 2 - Example Mesh with refinement near the wall depicted

As part of the setup of this model, multiple sets of boundary conditions were applied in order to determine the set that resulted in the best result. Multiple sets of boundary conditions were applied surrounding 3 methods for driving the flow. The first method was specifying a uniform inlet velocity to the model, essentially supplying a flow rate through the system. The second method was to supply an inlet and outlet pressure to create a differential that would drive the flow. The final type of driving force was to supply the model a pressure gradient that would drive the flow. Table 4 shows a summary of the boundary conditions applied to the model with these various methods.

Table 4 - Boundary Condition Attempts

Case	Inlet		Outlet		Wall	
	Velocity	Pressure	Velocity	Pressure	Velocity	Pressure
1	Fixed uniform axial velocity	Zero gradient	Outflow	Fixed uniform value	Zero velocity	Zero gradient
2	Inflow	Fixed uniform value	Outflow	Fixed uniform value	Zero velocity	Zero gradient
3	Zero Velocity	Fixed uniform value	Outflow	Fixed uniform value	Zero velocity	Zero gradient
4	Inflow	Fixed Gradient	Outflow	Fixed uniform value	Zero velocity	Zero gradient
5	Zero Velocity	Fixed Gradient	Outflow	Fixed uniform value	Zero velocity	Zero gradient

From the boundary conditions listed in Table 4, cases 1 and 2 were successful in driving the flow for a laminar Reynolds number. However, when the Reynolds number was increased the model did not give a sensible result. After these unsuccessful attempts, a cyclic model was created to achieve the desired flow. In this model, cyclic boundary conditions are applied to flow velocity, pressure gradient, turbulent kinetic energy, turbulence dissipation, and turbulence viscosity at the inlet and outlet of the pipe. The wall is assumed to have a no slip condition with a zero gradient condition placed on the pressure. For the 2D model, a condition of empty is given to the axisymmetric faces of the model. In OpenFOAM, this tells the solver that the geometry should be treated as 2D and no solution is required in the direction normal to the axisymmetric plane. When the model is later expanded to a 3D model, the axisymmetric faces are given a boundary condition called symmetry, which is a slip boundary in OpenFOAM. An overview of these wedge boundary conditions is shown in Figure 3.

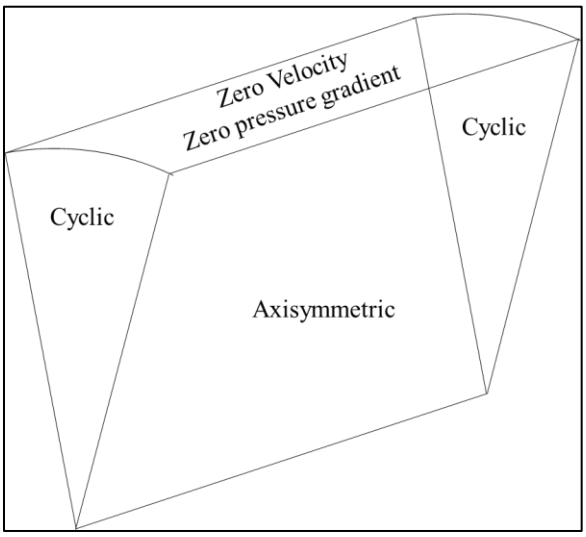


Figure 3 - Boundary Conditions

With cyclic boundary conditions applied to the inlet and outlet of the model, the flow is driven by a body force term that is added to the momentum equation. This is implemented in OpenFOAM® by specifying a desired bulk flow velocity. The model takes this bulk flow and develops a pressure gradient to match it.

When modeling the flow using RANS, it is sometimes common to utilize wall functions in the near wall region. These functions utilize scaling methods such as the ones shown in equations 18 through 22. This method assumes certain properties near the wall and is meant to eliminate the need for a fine mesh resolution in this area. An attempt was made to implement wall functions in this model, however the resulting flow statistics were not accurate. The decision was made to use a refined mesh near the wall to determine the flow characteristics in this region.

The model also requires the initialization of values for k, ε, and ω. These values are all given cyclic boundary conditions at the inlet and outlet of the pipe. Near the wall the k and epsilon are given very small but non-zero quantities. Giving these quantities a value of zero causes the model to become unstable. The internal field of the pipe is given initial estimated quantities for k and ε based on equations 25 and 26 (Magoules, 2011).

$$\mathrm{k} = \frac{3}{2}(uI)^2 \tag{25}$$

$$\epsilon = \mathrm{C}_{\mu}\frac{k^{\frac{3}{2}}}{l} \tag{26}$$

I is the turbulence intensity and is estimated by (Magoules, 2011)

$$I = 0.16\,Re^{-\frac{1}{8}} \tag{27}$$

While l is the turbulent length scale and is estimated using the equation

$$l = 0.038D \tag{28}$$

In addition, for the model runs utilizes the k-ω model, the value of ω is initialized using

$$\omega = \frac{\sqrt{k}}{l} \tag{29}$$

Since these values are estimates, a sensitivity study was performed changing the initial values of k, ε, and ω in the model. It is shown later in this thesis that no model is able to accurately match the values for turbulent kinetic energy from the available data; however, the sensitivity study showed that the simulation gave the best results when the initial values for k, ε, and ω were within approximately 10% of the calculated estimates. In addition, when the initial values were kept within 10% of the calculated estimates, the model did not give significantly different results for velocity profile or turbulence characteristics when compared to each other. When the initial given values were much different that the estimates obtained from equations 25, 16 and 29, the model does not give a physically correct result for both velocity profile and turbulence statistics. The velocity profile approaches the laminar velocity profile when initial values are too low and the turbulence statistics are nearly zero. When initial values are too high, the velocity profile transitions almost immediately from zero velocity at the wall to maximum velocity and turbulence statistics are much too high for the given Reynolds number.

3.3 Mesh

One important aspect to CFD modeling, is the mesh used for the model.

The mesh directly impacts the quality of the results, amount of time to run a simulation, and the stability of the simulation. Care has been taken in this work to iterate on the mesh properties to achieve a balance of these things.

The baseline 1/72 axisymmetric model is essentially a two-dimensional analysis of the flow field. Since OpenFOAM® cannot truly solve a model in only two dimensions, a third must be added. The mesh for this model is set up using Gmsh software. This program requires the user to input points and lines that define the flow field. In addition, line loops are used to create the surfaces on which boundary conditions are applied. The 1/72 wedge model is completely axisymmetric which allows for a structured mesh.

The basic mesh for this model is a uniform mesh in the axial direction with a graded mesh in the radial direction. The grading of the mesh allows the model to have smaller gird spacing near the wall where it is needed and larger spacing at the center of the pipe where the flow field is not as complex. This allows the overall grid to have fewer cells while maintaining the quality of results. In Gmsh, the grading factor is the ratio of the size of one cell to the size of the previous cell.

$$Progression\ Factor\ (PF) = \frac{\Delta X_n}{\Delta X_{n-1}} \tag{30}$$

Using this definition, cell size can also be related back to the size of the first cell.

$$X_n = X_1 \cdot\ PF^{n-1} \tag{31}$$

Care must be taken in Gmsh when setting the progression factor along a line. Since each line is defined from one point to another, a direction is given to the line. The progression defined uses the direction of the line. For example, if a line is defined from point 1 to point 2, a progression factor less than 1 results in smaller cell size near point 2 while a

progression factor greater than 1 results in smaller cell size near point 1. In the circumferential direction, only a single cell is used. As previously mentioned, this is needed in order to run the mesh with the OpenFOAM® software.

3.4 Model Verification

To ensure the model is correctly setup, verification is performed using the analytical solution of laminar flow in a pipe. For fully developed laminar flow in a pipe also known as Hagen-Poiseuille flow, the Navier Stokes equations can be solved analytically. The flow has a maximum velocity at the centerline of the pipe and a zero velocity at the pipe wall. The velocity profile in between the wall and centerline is parabolic and is represented by the equation

$$U(r) = U_{max}\left[1 - \left(\frac{r}{R}\right)^2\right] \tag{32}$$

Where R is the radius of the pipe, U_{max} is the maximum flow velocity at the centerline of the pipe, and r is the distance from the centerline of the pipe. As a first step in model verification, the model is run at Re = 100 and the velocity profile is compared to the analytical prediction. The mesh density and number of time steps the model is run for are iterated on to optimize the model.

An initial mesh of 100 axial cells and 25 radial cells is used with a grading factor of 0.95. Near wall resolution of the mesh is shown in Figure 4. As can be seen, this initial mesh does not have much refinement at the wall.

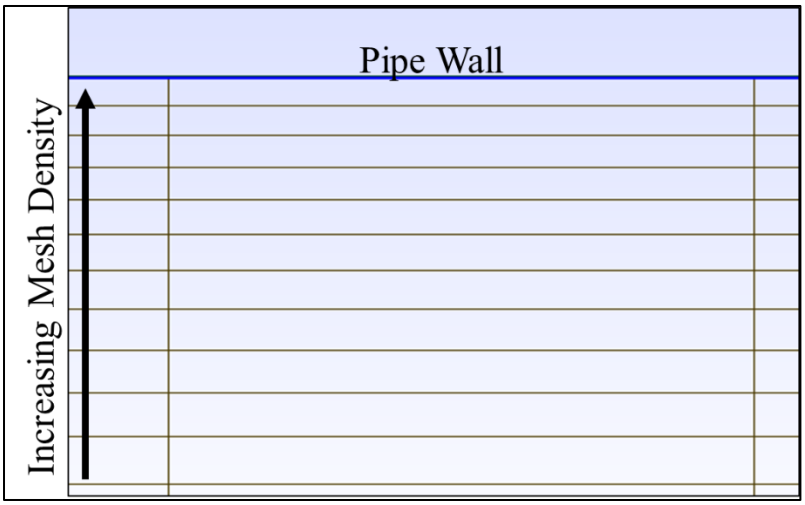


Figure 4 - Near Wall Mesh with 25 radial elements and progression factor of 0.95

The model is run for 1500 time steps with a Δt of 1 second. The model is post processed and the velocity profile is compared to the analytical result, which is shown in Figure 5. The velocity is normalized by the analytical expected centerline value for flow velocity at the center of the pipe.

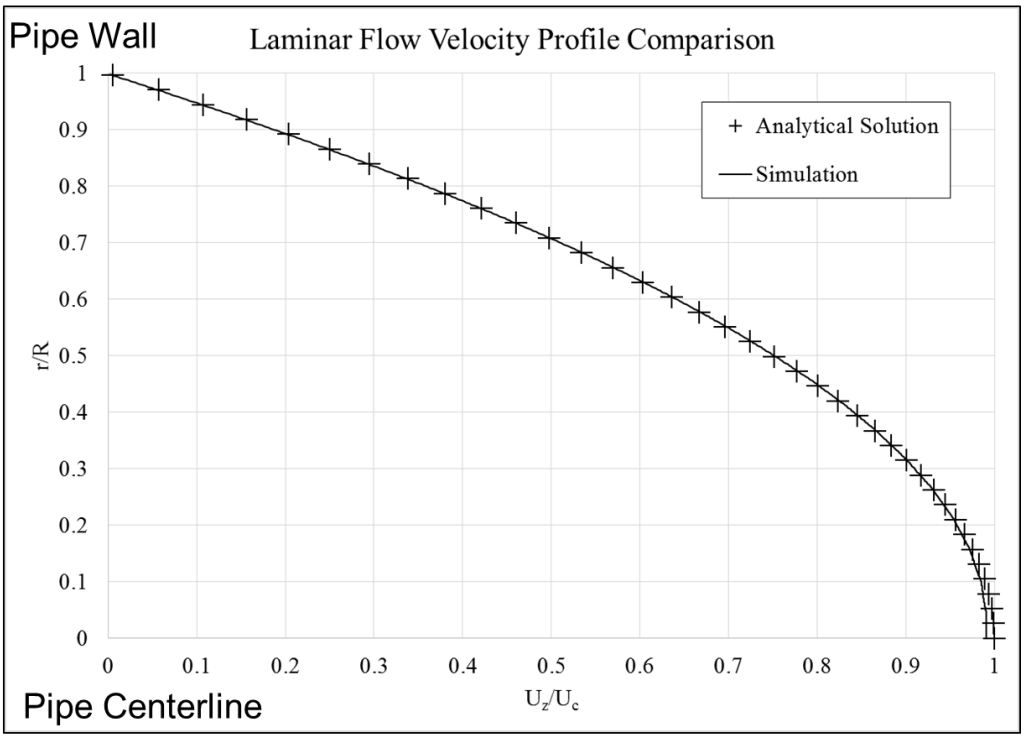


Figure 5 - Velocity Profile Comparison Mesh 100 X 25 X 1

In the figure, U_c is taken as the analytical centerline value for normalizing both the analytical and the simulated results. Figure 5 shows the results are close to the analytical result but the area at the wall deviates slightly. An L2 error analysis is performed on this data which calculated the total error using the following equation

$$Error = \sqrt{\sum_{1}^{n} (x_{n\ actual} - x_{n\ calculated})^2} \tag{33}$$

to obtain an error metric of 0.098 for the axial component of velocity. Much of this error is from the points near the wall which indicates more refinement is needed at the wall. The radial density of the mesh is incremented until the solution matches the analytical prediction. The final mesh is 100 elements in the axial direction and 50 elements in the

radial direction with a progression factor of 0.95. The near wall mesh is shown in Figure 6 and the velocity profile is shown in Figure 7.

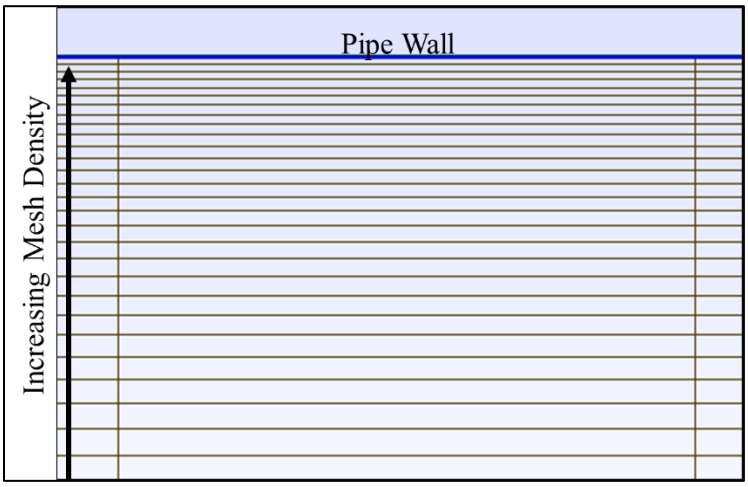
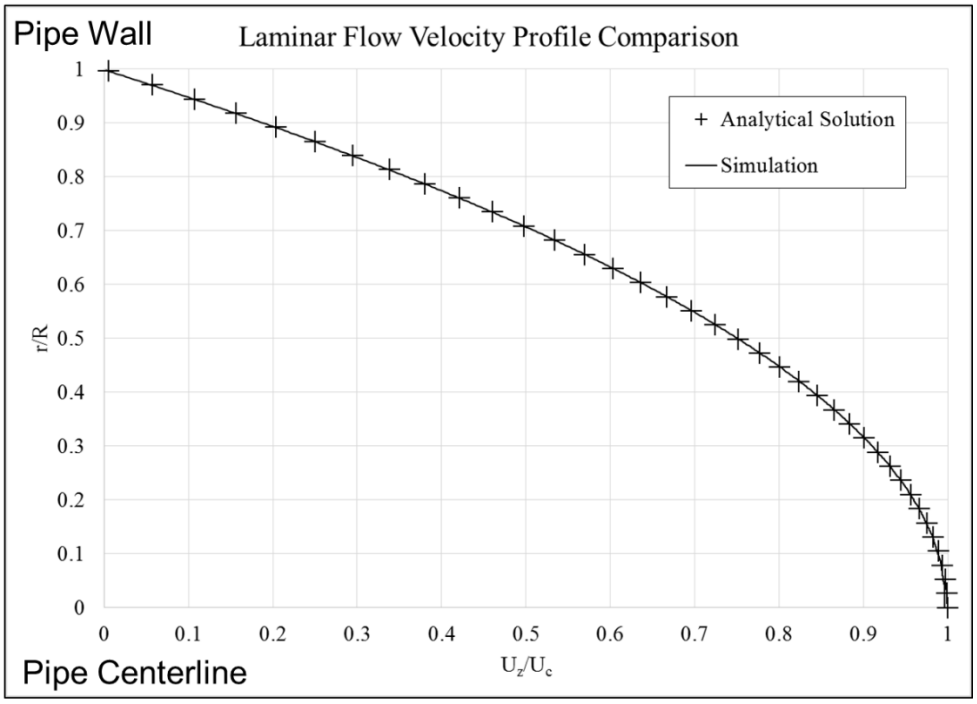


Figure 6 - Near Wall Mesh with 50 radial elements and progression factor of 0.95



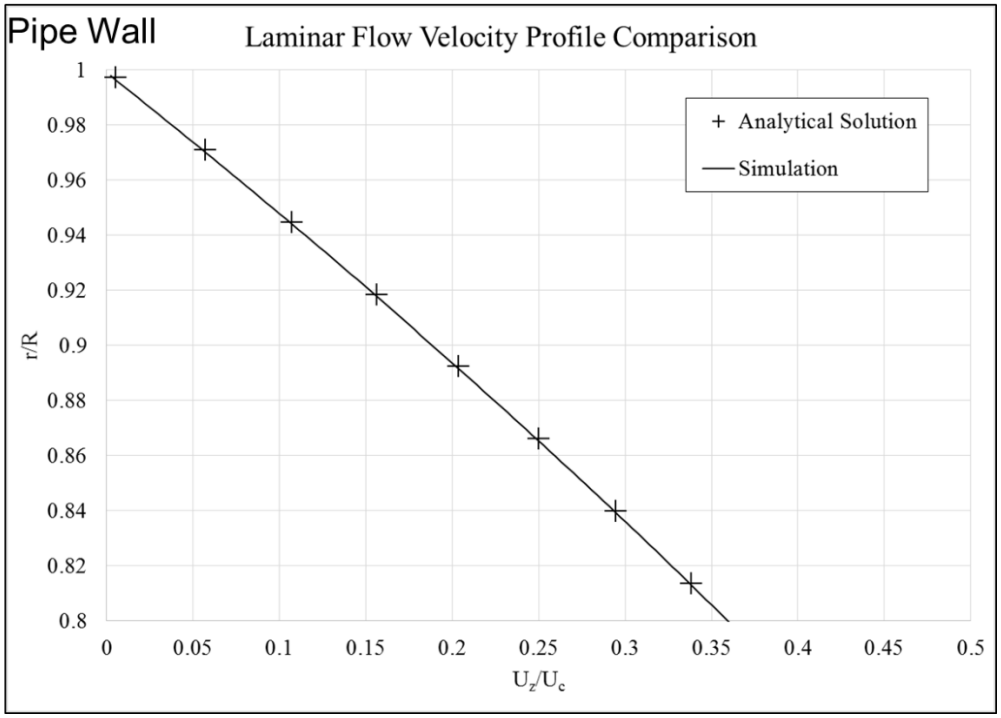


Figure 7 - Laminar Flow Velocity Comparison, Top: mean flow profile from pipe center to pipe wall. Bottom: pipe wall region.

This mesh refinement is able to reduce the L2 error from 0.098 to 0.012.

A final step is to check the length of simulation required to reach a converged solution. The data for the simulation is written out after every 100 time steps and the difference in velocity profiles is calculated. The solution appears to converge after about 300 time steps. The simulation runs in less than 1 minute of CPU time so a final check, the length of the simulation is doubled from 1500 compared with the solution after 300 times steps. Since there is no difference between the solutions after 300 times steps versus 3000 time steps, the solution is considered to be converged. The model shows good agreement with the expected analytical solution for the velocity profile. This result gives confidence in the boundary conditions and overall setup of the model as well as the optimization steps.

3.5 Validation at Re = 5300

The verification of the model using the analytical solution of laminar pipe flow give confidence that the model is correctly implement. The next step is to validate the model to available data for turbulent pipe flow. The Eggels et al. (1993) work provides a good comparison for flow at bulk velocity Re = 5300; however, a few limitations of this data should be noted. First, while plots of the radial profile are given in the work, the actual data is not available. This could result in slight error when translating the information to compare with the simulation run for this work. The data for velocity profiles obtained from experiment and DNS are plotted together and it is not possible to distinguish the two from the available plots. In addition, no details are given in the Eggels work about the size of the pipe, the fluid studied, or the velocity of the fluid. The

only information given is the Reynolds number at certain areas of the pipe. This work chose to tailor the geometry, fluid properties, and flow velocity to match the bulk velocity Reynolds number of 5300. The bulk velocity in the model is 0.2404 m/s.

The model is first run using the same mesh density as the previous laminar case, 100 axial elements by 50 radial elements with a progression factor of 0.95. The turbulence model used for this simulation is the standard k-ε model with the coefficients given in Table 1. Again, the model is run for 1500 times steps with a Δt of 1 and post processed. The radial velocity profile compared to the Eggels et al. (1993) data is shown in Figure 8. The Eggels data are given in normalized values. The velocity for the simulation data is normalized by the centerline velocity as calculated by the simulation itself.

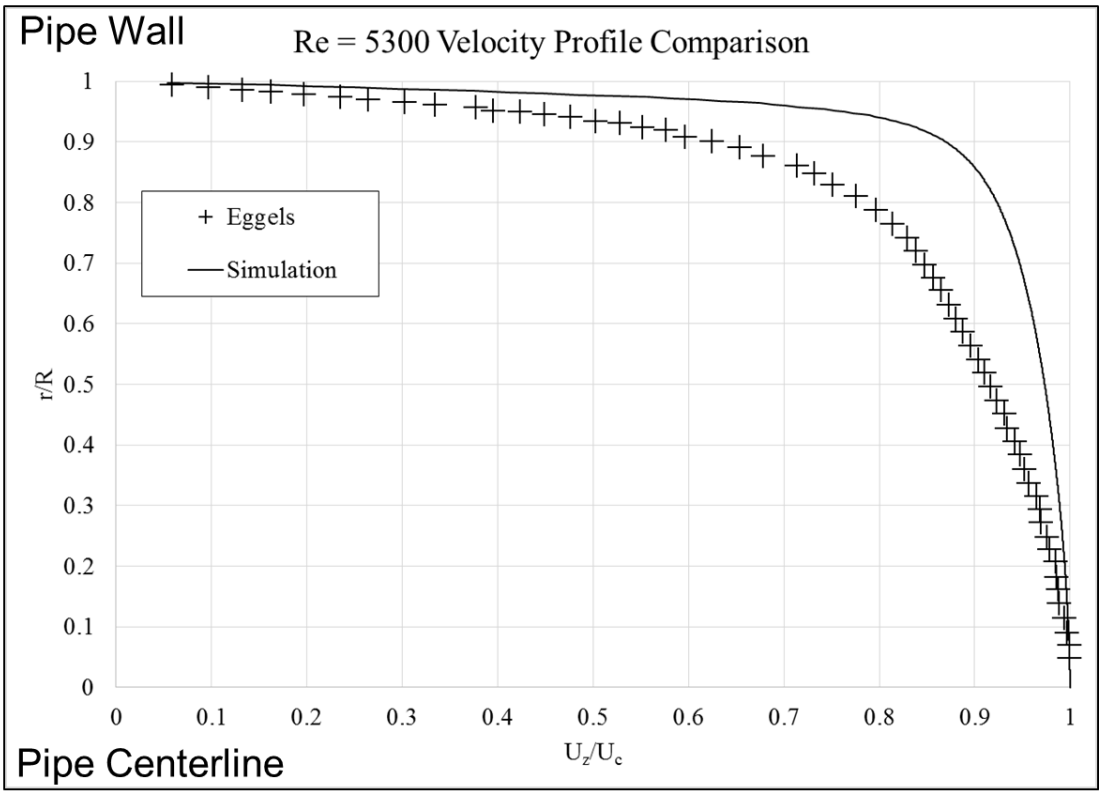


Figure 8 – Radial Velocity Profile Re=5300, 100 X 50 X 1 Mesh

The L2 error for these data is 0.752, which indicated a significant difference between the simulation and the validation data which indicates mesh refinement should be performed. The density of the mesh in the radial direction is incrementally increased and each case is post processed. In addition to increasing the number of elements in the radial direction, the axial mesh density is also increased. This is done to maintain the aspect ratio between radial height of the elements to axial width. Maintaining this aspect ratio results in a higher quality result while also ensuring model stability.

The refinement of the mesh achieves the closest match to the Eggels et al. (1993) data with 500 elements in the axial direction and 100 elements in the radial direction with a progression factor of 0.95. The first grid point is at a physical distance of y = .005 mm.

The corresponding value of y^+ one cell in from the wall as calculated by using the yplusRAS utility in OpenFOAM® is 0.02 for this mesh. This value indicates the mesh is well refined near the wall. The velocity profile result of this simulation is shown in Figure 9. This solution for this case appears to converge after 1200 time steps which takes less than 5 CPU minutes to complete. Once again the number of time steps is doubled from 1500 to 3000 time steps and the result is compared to ensure the solution is converged and there is no difference in the result after 1200 time steps and 3000 time steps.

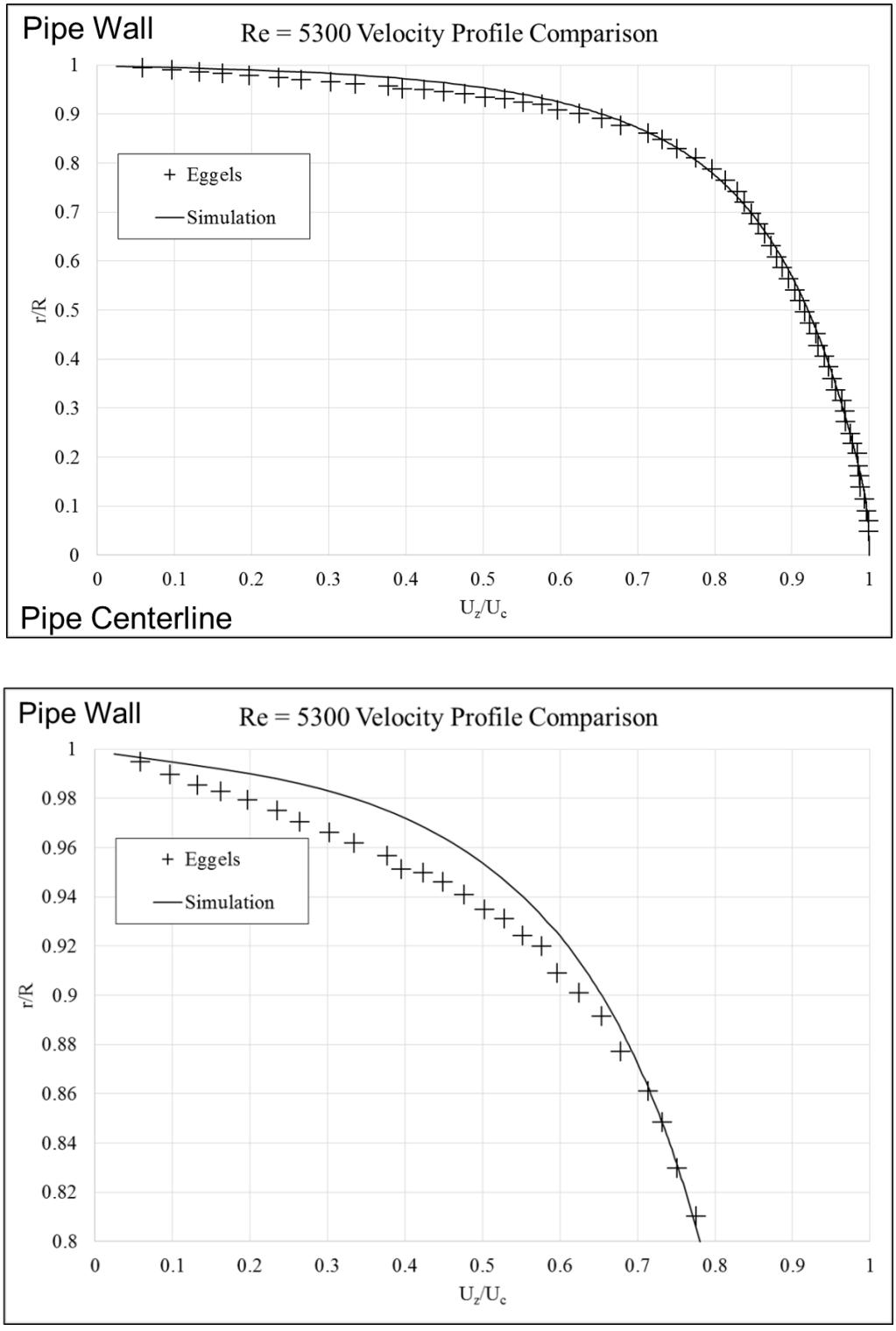


Figure 9 – Mean Velocity Profile Re=5300, 500 x 100 x 1 Mesh. Top: mean flow profile from pipe center to pipe wall. Bottom: pipe wall region.

A L2 norm error analysis was performed comparing the Eggels data and the best solution achieved in the simulation. While the maximum error value is 0.144 close to the wall, the overall mean-squared error is reduced from 0.752 to 0.300. This analysis shows that while the solution is not perfectly matched to the Eggels data near the wall, the overall match along the entire profile is good. The difference in velocity near the wall is attributed to the accuracy of RANS versus DNS. The solution is considered to match closely enough to be used for engineering design purposes.

4 ADDITIONAL STUDIES

4.1 Two-Dimensional vs Three-Dimensional Model

The validation of a two-dimensional axisymmetric model was also performed. It is of interest to look at the effect of the third dimension for two reasons. First, turbulence is inherently a three-dimensional phenomenon and studying this in 2D may not be an accurate assumption. The second reason is the future goal is to study non-axisymmetric geometries. One example of this is the fan blade at the inlet of a turbine engine. Advanced fan blades have three-dimensional geometry and cannot be studied in 2D.

To study the effect of the third dimension, models are run as a 1/8 axisymmetric, 1/4 axisymmetric, and a full 3D pipe. Each of these are run at the same Reynolds number with the same axial and radial mesh densities. The cells in the circumferential direction are divided into 7.5 degree arcs. For the 1/8 and 1/4 axisymmetric models a condition of symmetry is added to the front and back planes. The full 3D pipe requires no symmetry boundary conditions. All of the simulations were run serially. The model size and run times for these simulations are shown in Table 5. The results of all of these models including the 2D model are shown in Figure 10.

Table 5 - Model Size and Simulation Run Time

Simulation	Number of Grid Points	CPU Time (hours)
1/8 Axisymmetric	297,500	0.75
1/4 Axisymmetric	594,500	2.33
Full 3D Pipe	2,030,000	34.15

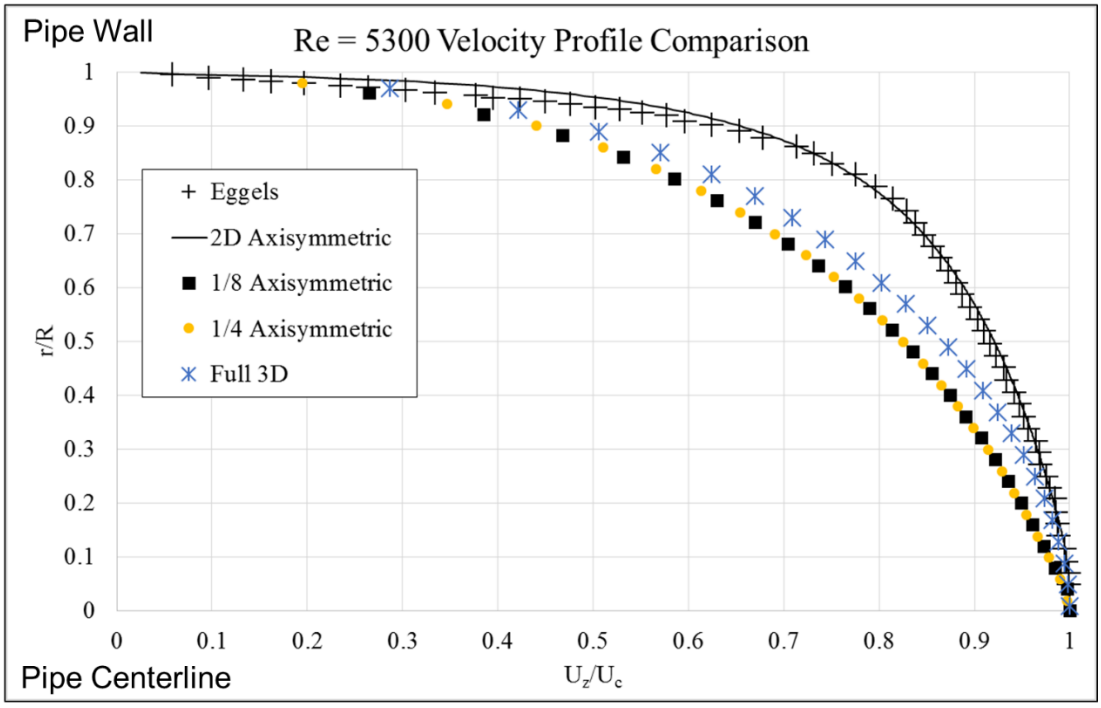


Figure 10 - Radial velocity profile comparison of multiple size models

As can been seen from the profile comparison, incorporating the third dimension has a significant impact. Another L2 norm error analysis was performed with this data and the mean-squared error is shown in

Simulation	Mean-Squared Error
1/8 Axisymmetric	1.15
1/4 Axisymmetric	1.13
Full 3D Pipe	0.79

The error values show that the model run with 1/8 and 1/4 axisymmetric geometry have results that line up well with each other; however, they do not match the existing validation data. The model with a full 3D pipe is closer to the 1/8 and 1/4

models than the 2D and validation data. One potential reason for the mismatch between the 2D and the other models is the circumferential mesh density. An investigation was performed where the circumferential mesh density is increased and the results do not show a difference until the model becomes unstable when the cell arc is increased to match the size of the 2D axisymmetric wedge. In addition, the radial mesh density was increased in the 1/4 wedge model but the result did not improve.

Another potential reason for the difference between 2D and the 3D models is the potential for the 2D to result in a different flowrate due to the size of the wedge. This concern was ruled out by performing an integration of the velocity profiles for each model and showing they all results have the same flow rate when converted into a full circular area.

The next aspect that was interrogated in order to determine why the difference in results exists between 2D and 3D was the turbulent kinetic energy profile. Figure 11 shows the profile for each case as well as from the Eggels DNS simulation. These profiles show that the 2D model does not correctly capture the turbulence. The profiles from the 3D simulations do have a similar profile compared to the validation data, however the trend deviates towards the center of the pipe and the magnitudes are off by a factor of about 150. This could explain why these models do not match the validation data for velocity profile.

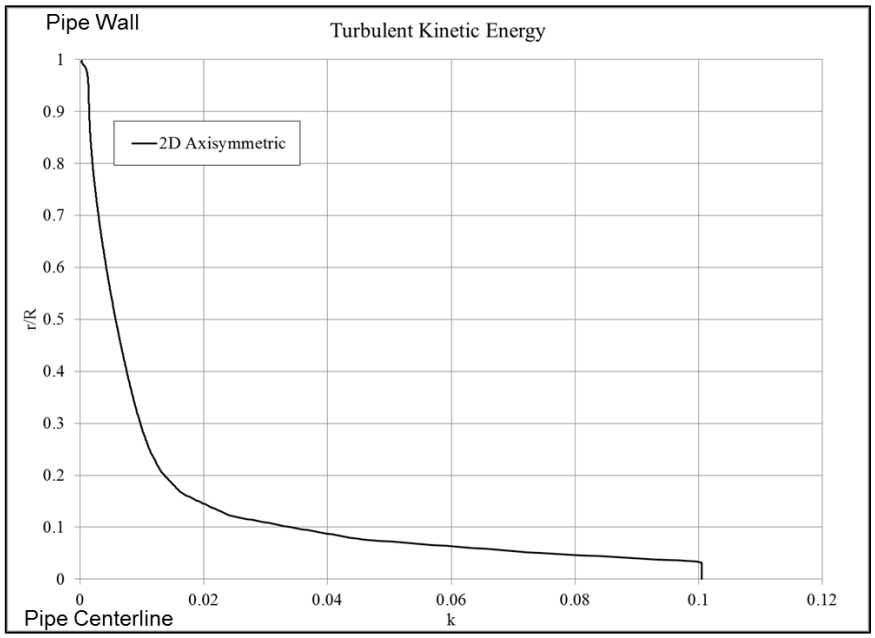


Figure 11 (a)

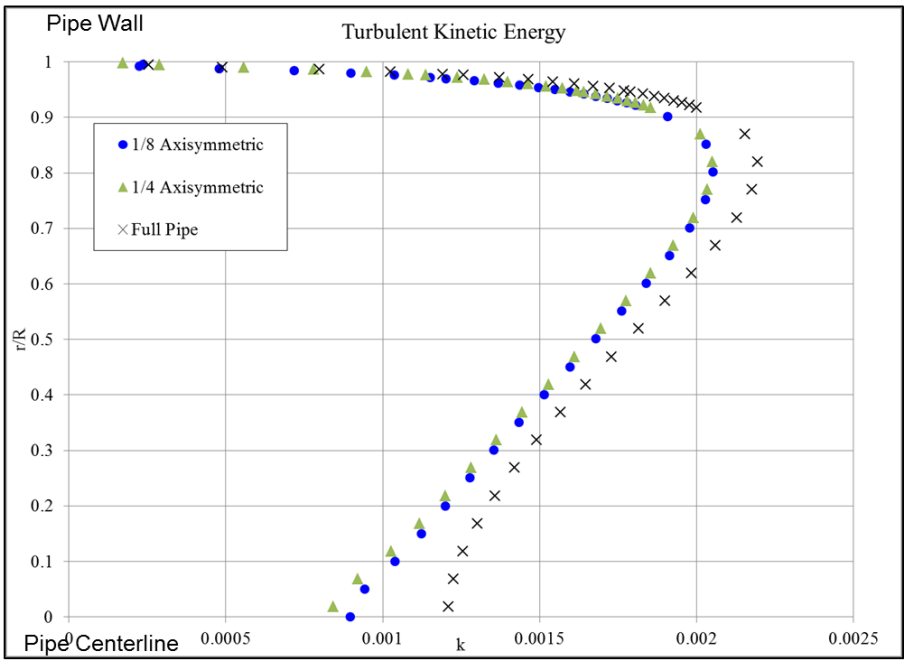


Figure 11 (b)

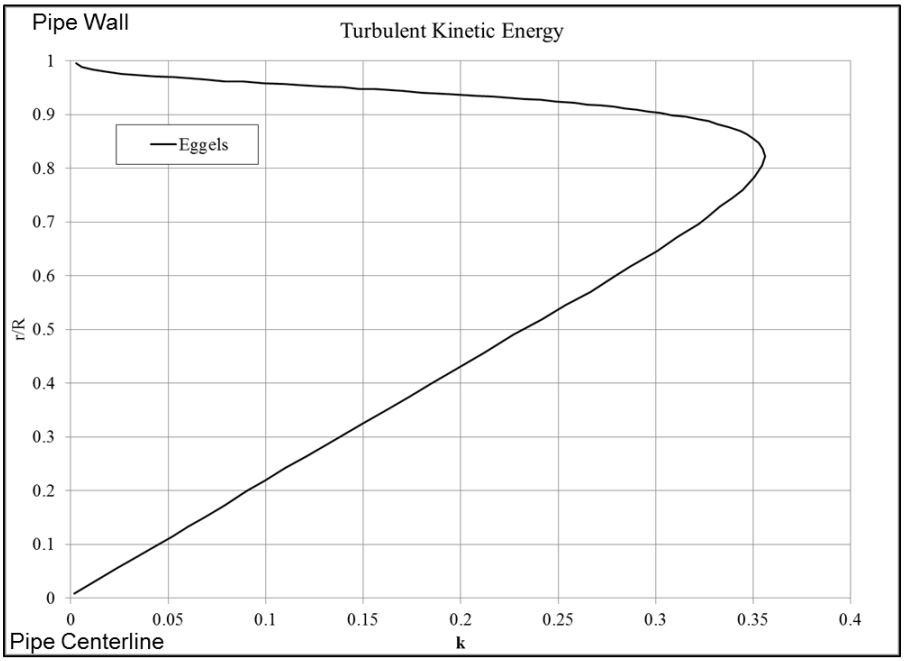


Figure 11 (c)

Figure 11 - Turbulent Kinetic Energy Profile, (a) 2D Simulation (b) 3D Simulations (c) Validation Data

This data suggests that while the velocity profile of the 2D model appears to give a correct result, this is a false indication of its performance. In reality the model does not correctly capture the turbulent properties of the flow. Figure 12 shows the turbulence dissipation profiles for each of the cases. The 2D axisymmetric case shows the opposite trend for dissipation form the 3D case, which are matched to each other. This is another piece of evidence to support the idea that the 2D axisymmetric model does not correctly capture the turbulence.

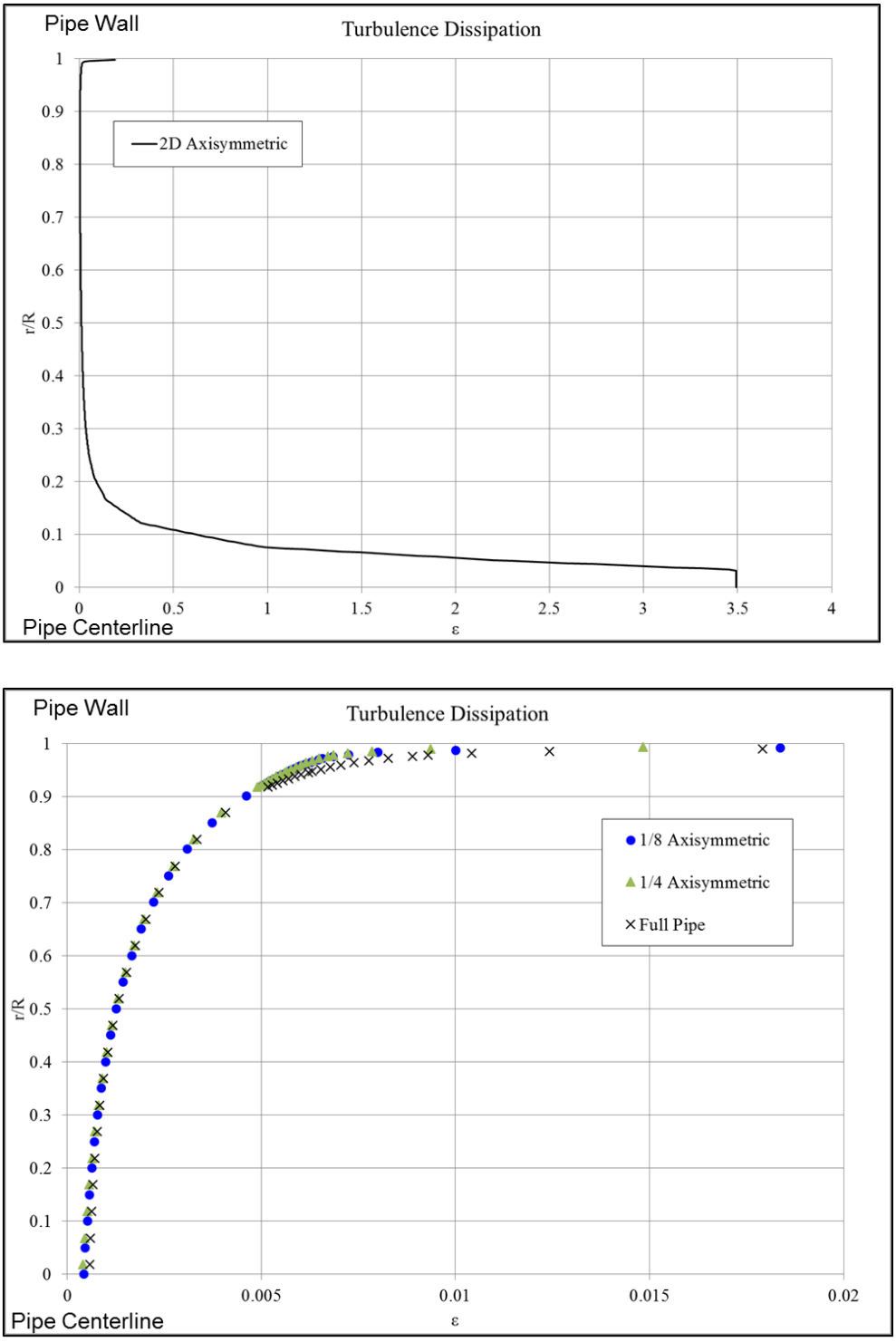


Figure 12 - Turbulence Dissipation, Top: 2D Axisymmetric Case, Bottom: 3D Case Comparison

To cement the idea that the difference between 2D and 3D is driven by turbulence, a study was then performed for the laminar case considering the addition of the third dimension. The results of this are shown in Figure 13 and demonstrate that the issue that plagues the turbulent case does not show up in the laminar case.

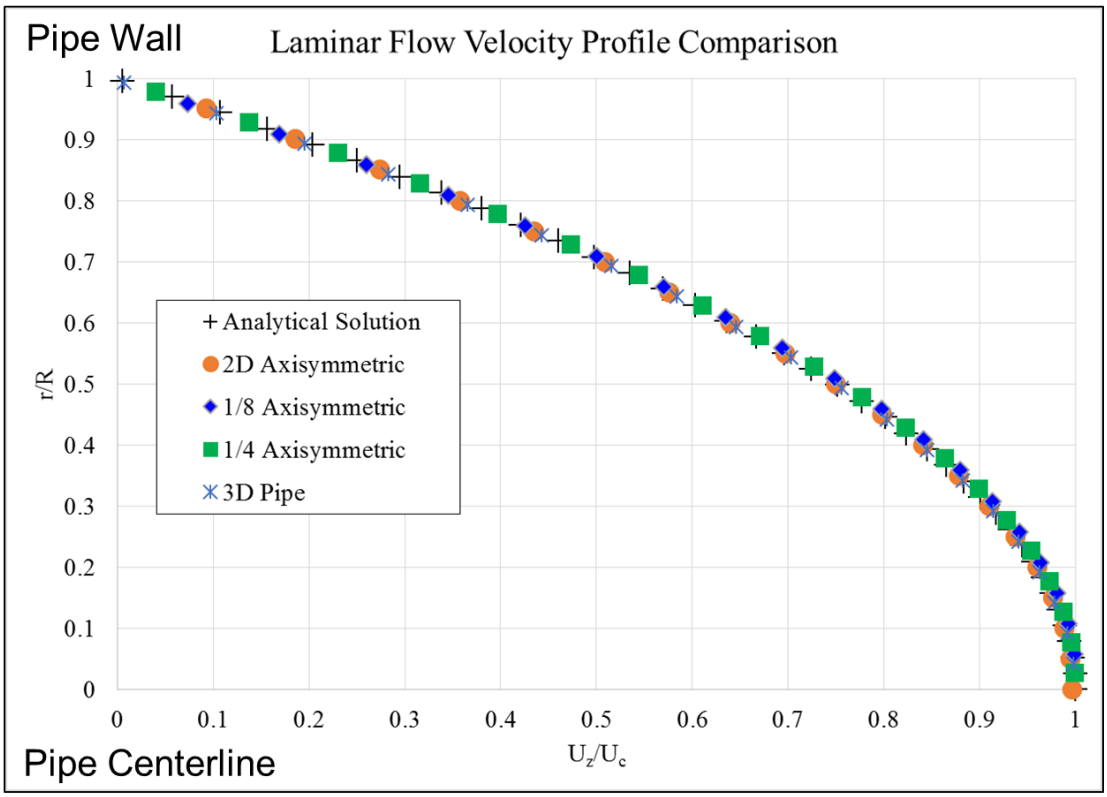


Figure 13 – Three-Dimensional Comparison for Laminar Case

The L2 error for each of these simulations is shown in Table 6.

Table 6 - L2 Error for Laminar Flow Comparison

Simulation	L2 Error
2D Axisymmetric	0.012
1/8 Axisymmetric	0.058
1/4 Axisymmetric	0.049
Full 3D Pipe	0.030

This error data indicates that the 2D simulation is the most accurate, but the other simulations also closely match the analytic results. These studies show that the 2D model developed does not correctly capture the turbulence in the flow field. They also indicate that while the 3D models perform better when predicting the turbulence, more work is needed to improve the models.

4.2 Comparison of Turbulence Models

As previously mention in Section 1, many different turbulence models exist. Each of these has benefits and drawbacks depending of the type of flow that is being studied. As part of this work, a back to back comparison was performed between the k-ε model, the k-ω model, and the k-ω SST model. All other variables such as Reynolds number and mesh density were held constant and the turbulence model used was changed. Since it was shown that the 2D model does not correctly capture turbulence statistics, the 1/8 axisymmetric was used for the comparison of the turbulence model. The results from each of the three models at Re = 5300 are shown in Figure 14.

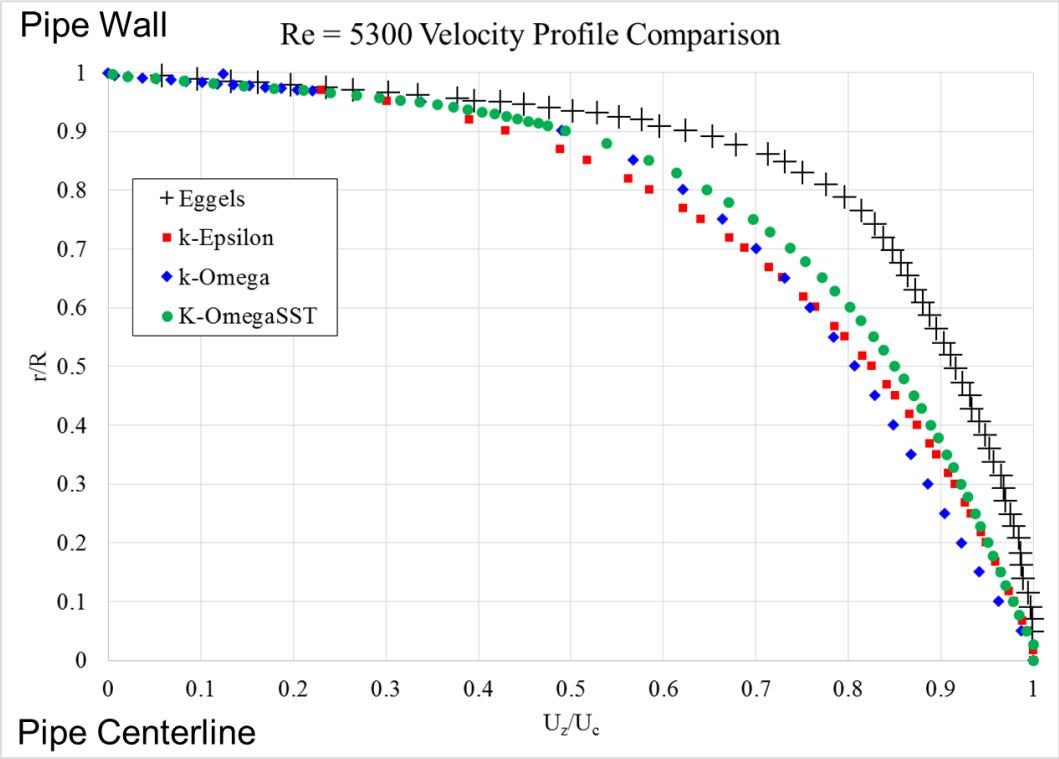


Figure 14 - Velocity Profile Comparison of Turbulence Models

From this comparison plot it is shown that the models are close to each other but there are some differences between them. An L2 norm error analysis was performed on these data sets. The mean-squared error compared to the Eggels data for each simulation is shown in Table 7.

Table 7 - Mean-Squared Error for Simulations

Simulation	Mean-Squared Error
k-ε model	1.15
k-ω model	1.07
k-ω SST model	0.81

This error analysis shows that the k-ω SST turbulent model is the best performing for predicting the velocity profile. The k-ω model seems to perform slightly better in the region 10%-20% away from the wall. The k-ε model appears to perform slightly better from halfway between the wall and center line down to the centerline. Both of these observations follow the expected outcome. The k-ω model is better at predicting the flow in lower Reynolds number flow and the k-ε model is better for higher Reynolds number. Finally, the model that is closest to the validation data is the k-ω SST model. This follows logic since the k-ω SST model combines both models. While some differences are seen, they are slight. Figure 15 shows a comparison of the turbulent kinetic energy for each of the three models.

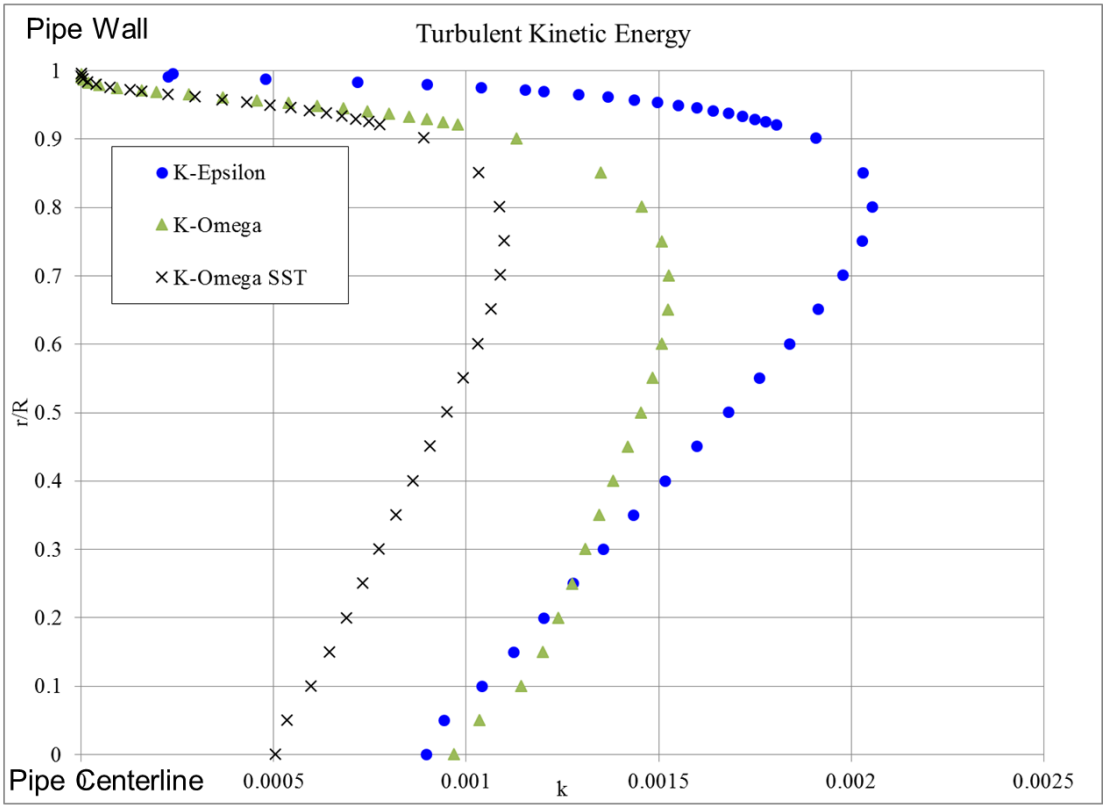


Figure 15 - Turbulent Kinetic Energy Comparison

All of the models have a similar shaped profile; however, they differ in values throughout the profile. This is attributed to the difference in the way each model handles the turbulent kinetic energy.

The k-ω SST model shows only a slight advantage over the other two models. Using the k-ω SST model may have more benefits at higher Reynolds number bulk flow because of the greater difference in centerline Reynolds number and near wall Reynolds number.

5 SUMMARY AND CONCLUSIONS

A model has been developed for predicting the mean flow for turbulent flow through a circular pipe. The model is developed for use with the open source software OpenFOAM® and utilizes the Gmsh tool for meshing the flow field. The model was verified using analytic results for laminar flow in an empty circular pipe and validated against data available for Re = 5300. The steps taken to optimize the model to achieve the best result are outlined.

The additional studies performed outside of the model validation show that the model is sensitive to many factors. Included in these are the axisymmetric flow assumption, mesh density, and turbulence model used. The studies show that changing one of these aspects detunes the model and the results given are poor. In addition, it is shown that the two-dimensional model does not capture the turbulence properties of the flow well. Based on this the conclusions is made that the model requires significant changes to improve its performance.

Despite the lack of robustness in the model, this work demonstrates the effort required to build and validate a model for predicting turbulent flow in a circular pipe. It outlines a technique for mesh development and optimization, a choice of boundary conditions that lead to reasonable solutions, and several attempts to determine a model whose results match available data. The work also explains how this model is built using open source software, which is useful to those that do not have access to commercial CFD code and meshing software.

This work can be used as a foundation for building improved models for studying flow scenarios involving turbulent flow in pipes. The first step in this work should be to make the model more robust to tolerance changes in the model. In addition, other boundary conditions should be considered beyond the cyclic inlet and outlet conditions applied to this model. This would allow more complex analysis of changes in the flow field from an object protruding into the flow.

Turbulent flow in a pipe is a foundational problem in fluids and is important for many different reasons. Although the problem appears a simple one on the surface, the complexities of modeling turbulent flow make it quite difficult to solve. The best methods for predicting this flow are computationally expensive and because of this they are limited in the problems they can solve. It is important to further the field on modeling techniques such as RANS in order to provide ways to predict these flows. Doing so allows for better tools that will improve the design of components where internal flow is a factor.

6 APPENDIX

Appendix A – Mesh Script

```
//Inlet Center Point
Point(1) = {0, 0, 0, 1.0};

//Outlet Center Point
Point(2) = {3, 0, 0, 1.0};

//Inlet Outer Wall Arc - Change Y and Z Values when changing wedge size
Point(3) = {0, 0.00332, 0.07613};
Point(4) = {0, -0.000332, 0};

//Outlet Outer Wall Arc - Change Y and Z Values when changing wedge size
Point(5) = {3, 0.00332, 0.07613};
Point(6) = {3, -0.00332, 0};

//Centerline
Line(1) = {1, 2};

//Inlet Lines
Line(2) = {1, 3};
Line(3) = {1, 4};
Circle(4) = {3, 1, 4};

//Outlet Lines
Line(5) = {2, 5};
Line(6) = {2, 6};
Circle(7) = {5, 2, 6};

//Axial Lines Outer Wall
Line(8) = {3, 5};
Line(9) = {4, 6};

//Inlet Surfaces
Line Loop(1) = {2, 4, -3};
Ruled Surface(1) = {1};
Physical Surface("inlet") = {1};

//Outlet Surfaces
Line Loop(2) = {5, 7, -6};
Ruled Surface(2) = {2};
Physical Surface("outlet") = {2};

//Outer Wall Surface
Line Loop(3) = {8, 7, -9, -4};
```

```
Ruled Surface(3) = {3};
Physical Surface("wall") = {3};

//Axisymmetric Faces
Line Loop(4) = {1, 5, -8, -2};
Plane Surface(4) = {4};
//Physical Surface("axi_symm-f") = {4}; //Uncomment this line for wedges greater than 5 degrees

Line Loop(5) = {1, 6, -9, -3};
Plane Surface(5) = {5};
//Physical Surface("axi_symm-r") = {5}; //Uncomment this line for wedges greater than 5 degrees

//Creating Volumes
Surface Loop(1) = {1, 2, 3, 4, 5};
Volume(1) = {1};
Physical Volume("Duct") = {1};

//Axial Mesh Density
Transfinite Line {1, 8, 9} = 500;

//Radial Mesh Density
Transfinite Line {2, 3, 5, 6} = 100 Using Progression 0.95;

//Circumfrential Mesh Density
Transfinite Line {4, 7} = 1;

Transfinite Surface "*";
Recombine Surface "*";
Transfinite Volume "*";
```

Appendix B – Key Dictionary Files for Laminar Flow

B1. Transport Properties File

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.4.0                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    location    "constant";
    object      transportProperties;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

transportModel  Newtonian;

nu              nu [ 0 2 -1 0 0 0 0 ] 6.91e-06;

CrossPowerLawCoeffs
{
    nu0             nu0 [ 0 2 -1 0 0 0 0 ] 1e-06;
    nuInf           nuInf [ 0 2 -1 0 0 0 0 ] 1e-06;
    m               m [ 0 0 1 0 0 0 0 ] 1;
    n               n [ 0 0 0 0 0 0 0 ] 1;
}

BirdCarreauCoeffs
{
    nu0             nu0 [ 0 2 -1 0 0 0 0 ] 1e-06;
    nuInf           nuInf [ 0 2 -1 0 0 0 0 ] 1e-06;
    k               k [ 0 0 1 0 0 0 0 ] 0;
    n               n [ 0 0 0 0 0 0 0 ] 1;
}
// ************************************************************************* //
```

B2. Boundary File

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.4.0                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       polyBoundaryMesh;
    location    "constant/polyMesh";
    object      boundary;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //
(
    inlet
    {
        type            cyclic;
        inGroups        1(cyclic);
        nFaces          49;
        startFace       24104;
            matchTolerance 0.0001;
            transform       unknown;
            neighbourPatch outlet;
    }
    outlet
    {
        type            cyclic;
        inGroups        1(cyclic);
        nFaces          49;
        startFace       24153;
            matchTolerance 0.0001;
            transform       unknown;
            neighbourPatch inlet;
    }
    wall
    {
        type            wall;
        physicalType    wall;
        nFaces          249;
        startFace       24202;
    }
    defaultFaces
    {
        type            empty;
```

```
        nFaces          24402;
        startFace       24451;
    }
)

// ************************************************************************** //
```

B3. Velocity File

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.4.0                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       volVectorField;
    object      U;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //
dimensions      [0 1 -1 0 0 0 0];
internalField   uniform (0.00454 0 0);
boundaryField
{
    inlet
    {
        type            cyclic;
    }
    outlet
    {
        type            cyclic;
    }
    wall
    {
        type            fixedValue;
        value           uniform (0 0 0);
    }
    axi_symm-f
    {
        type            wedge;
    }
    axi_symm-r
    {
        type            wedge;
    }
}
*********************************************************************************
```

B4. Pressure File

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.4.0                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       volScalarField;
    object      p;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //
dimensions      [0 2 -2 0 0 0 0];

internalField   uniform 0;

boundaryField
{
    inlet
    {
        type            cyclic;
    }
    outlet
    {
        type            cyclic;
    }
    wall
    {
        type            zeroGradient;
    }
    axi_symm-f
    {
        type            wedge;
    }
        axi_symm-r
    {
        type            wedge;
    }
    defaultFaces
    {
        type            empty;
    }
}
// ************************************************************************* *
```

B5. fvOptions File

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.4.0                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    location    "system";
    object      fvOptions;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

momentumSource
{
    type            pressureGradientExplicitSource;
    active          on;            //on/off switch
    selectionMode   all;       //cellSet // points //cellZone

    pressureGradientExplicitSourceCoeffs
    {
        fieldNames  (U);
        Ubar        (0.00454 0 0 );
    }
}


// ************************************************************************* //
```

Appendix C – Additional Key Turbulent Flow Files

C1. Boundary File

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.4.0                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       polyBoundaryMesh;
    location    "constant/polyMesh";
    object      boundary;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

5
(
    inlet
    {
        type            cyclic;
        inGroups        1(cyclic);
        nFaces          495;
        startFace       688624;
            matchTolerance 0.0001;
            transform       unknown;
            neighbourPatch outlet;
    }
    outlet
    {
        type            cyclic;
        inGroups        1(cyclic);
        nFaces          495;
        startFace       689119;
            matchTolerance 0.0001;
            transform       unknown;
            neighbourPatch inlet;
    }
    wall
    {
        type            wall;
        physicalType    wall;
        nFaces          2495;
        startFace       689614;
```

```
    }
    axi_symm-f
    {
        type            symmetry;
        inGroups        1(symmetry);
        nFaces          49401;
        startFace       692109;
    }
    axi_symm-r
    {
        type            symmetry;
        inGroups        1(symmetry);
        nFaces          49401;
        startFace       741510;
    }
)

// ************************************************************************* //
```

B2. Velocity File

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.4.0                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       volVectorField;
    object      U;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

dimensions      [0 1 -1 0 0 0 0];
internalField   uniform (0.2404 0 0);
boundaryField
{
    inlet
    {
        type            cyclic;
    }
    outlet
    {
        type            cyclic;
    }

    wall
    {
        type            fixedValue;
        value           uniform (0 0 0);
    }

    axi_symm-f
    {
        type            symmetry;
    }
    axi_symm-r
    {
        type            symmetry;
    }
}

// ************************************************************************* //
```

B3. Pressure File

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.4.0                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       volScalarField;
    object      p;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //
dimensions      [0 2 -2 0 0 0 0];
internalField   uniform 0;
boundaryField
{
    inlet
    {
        type            cyclic;
    }
    outlet
    {
        type            cyclic;
    }
    wall
    {
        type            zeroGradient;
    }
    axi_symm-f
    {
        type            symmetry;
    }
    axi_symm-r
    {
        type            symmetry;
    }
    defaultFaces
    {
        type            empty;
    }
}

// ************************************************************************* //
```

B4. Turbulent Kinetic Energy File

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.4.0                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       volScalarField;
    location    "0";
    object      k;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //
dimensions      [0 2 -2 0 0 0 0];
internalField   uniform 2.6E-4;
boundaryField
{
    inlet
    {
        type            cyclic;
    }
    outlet
    {
        type            cyclic;
    }
    wall
    {
        type            kqRWallFunction;
        value           uniform 1E-10;
    }
    axi_symm-f
    {
        type            symmetry;
    }
    axi_symm-r
    {
        type            symmetry;
    }
}


// ************************************************************************* //
```

B5. Turbulent Dissipation

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.4.0                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       volScalarField;
    location    "0";
    object      epsilon;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

dimensions      [0 2 -3 0 0 0 0];
internalField   uniform 4.52E-6;
boundaryField
{
    inlet
    {
        type            cyclic;
    }
    outlet
    {
        type            cyclic;
    }
    axi_symm-f
    {
        type            symmetry;
    }
    axi_symm-r
    {
        type            symmetry;
    }
    wall
    {
        type            epsilonWallFunction;
        value           uniform 1E-10;
    }
}


// ************************************************************************* //
```

B6. Specific Dissipation Rate File

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.4.0                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       volScalarField;
    object      omega;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //
dimensions      [0 0 -1 0 0 0 0];
internalField   uniform 1E-10;
boundaryField
{
    inlet
    {
        type            cyclic;
    }
    outlet
    {
        type            cyclic;
    }
    wall
    {
        type            omegaWallFunction;
        value           uniform 10;
    }
    axi_symm-f
    {
        type            wedge;
    }
    axi_symm-r
    {
        type            wedge;
    }
}


// ************************************************************************* //
```

7 REFERENCES

Bendiks Jan Boersma, "Direct numerical simulation of turbulent pipe flow up to a Reynolds number of 61,000," Journal of Physics: Conference Series, vol. 318, 2011.

D. E. Coles and E. A. Hirst (eds.), “Computation of turbulent boundary layers – 1968 AFOSR-IFP-Stanford Conference; Proceedings”, vol. 2, Thermosciences Division, Stanford University, Stanford, California, 1968.

J. G. M. Eggels, F. Unger, M. H. Weiss, J. Westerweel, R. J. Adrian, R. Friedrich, and F. T. M. Nieuwstadt, “Fully developed turbulent pipe flow: a comparison between direct numerical simulation and experiment,” Journal of Fluid Mechanics, vol. 268, pp. 175–209, 1994.

W. P. Jones and B. E. Launder, “The prediction of laminarisation with a 2-equation model of turbulence.” International Journal of Heat and Mass Transfer, vol. 15, pp. 301, 1972.

T. von Karmen, "The analogy between fluid friction and heat transfer," Transactions of the American Society of Mechanical Engineers, vol. 61, pp. 705–710, 1939.

J. Kim, P. Moin, and R. Moser, “Turbulence statistics in fully developed channel flow at low Reynolds number,” Journal of Fluid Mechanics, vol. 177, pp. 133–166, 1987.

Franziska Konig, “Investigation of High Reynolds Number Pipe Flow,” PhD dissertation, Dept. of Mechanical and Electrical Eng., Bradenburg University of Technology, Cottbus, Germany, 2015.

B. E. Launder and D. B. Spalding, Mathematical Models of Turbulence. New York, New York: Academic Press Inc., 1972.

Frederic Magoules, Computational Fluid Dynamics. London, UK: CRC Press, Taylor & Francis Group, 2011.

T. Maric, J. Hopken, and K. Mooney, The OpenFOAM® Technology Primer. SourceFlux, 2014.

Jens-Dominik Muller, Essentials of Computational Fluid Dynamics. Boca Raton, FL: CRC Press, Taylor & Francis Group, 2016.

L. Prandtl, “Über die ausgebildete turbulenz,” Zeitschrift für Angewandte Mathemathk und Mechanik, vol. 5, pp. 136, 1925.

C. Pozrikidis. Fluid Dynamics: Theory, Computation, and Numerical Simulation. Norwell, Massachusetts: Kluwer Academic Publishers, 2001.

D. B. Spalding, “Heat transfer to a turbulent stream from a surface with a step-wise discontinuity in wall temperatures,” International Developments in Heat Transfer; Proceedings of the 2nd Heat Transfer Conference. Part II, pp. 439–436. New York, New York: American Society of Mechanical Engineers, 1961.

D. B. Spalding, “A Single formula for the ‘Law of the Wall’,” Journal of Applied Mechanics, vol. 28, pp. 455–457, 1961.

G. I. Taylor, “Conditions at the surface of a hot body exposed to the wind,” Reports and Memoranda (Great Britain. Advisory Committee for Aeronautics), No. 272, pp. 423. London: H.M.S.O., 1916.

Frank M. White, Viscous Fluid Flow, 3rd Edition. New York, New York: McGraw-Hill, 2006.

Frank M. White, Fluid Mechanics, 7th Edition. New York, New York: McGraw-Hill, 2011.

David C. Wilcox, “Reassessment of the scale-determine equation for advanced turbulence models,” AIAA Journal, vol. 26, pp. 1299–1310, November, 1988.

Xiaohua Wu and Parviz Moin, "A direct numerical simulation study on the mean velocity characteristics in turbulent pipe flow," Journal of Fluid Mechanics, vol. 608, pp. 81–112, April 2008.

Mark Zagarola, "Mean-flow scaling of turbulent pipe flow," PhD dissertation, Dept. of Mechanical and Aerospace Engineering, Princeton University, Princeton, NJ, 1996.

VITA