DISSERTATION DEFENSE COMPUTATIONAL FLUID DYNAMICS UNCERTAINTY ANALYSIS FOR PAYLOAD FAIRING SPACECRAFT ENVIRONMENTAL CONTROL SYSTEMS

by

CURTIS EDWARD GROVES

B.S. West Virginia University, 2008
B.S. West Virginia University, 2008
M.S. University of Central Florida, 2012

A dissertation submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy
in the Department of Mechanical and Aerospace Engineering
in the College of Engineering and Computer Science
at the University of Central Florida
Orlando, Florida

Spring Term
2014

Major Professor: Alain Kassab

ABSTRACT

Spacecraft thermal protection systems are at risk of being damaged due to airflow produced from Environmental Control Systems. There are inherent uncertainties and errors associated with using Computational Fluid Dynamics to predict the airflow field around a spacecraft from the Environmental Control System. This paper describes an approach to quantify the uncertainty in using Computational Fluid Dynamics to predict airflow speeds around an encapsulated spacecraft without the use of test data. Quantifying the uncertainty in analytical predictions is imperative to the success of any simulation-based product. The method could provide an alternative to traditional "validation by test only" mentality. This method could be extended to other disciplines and has potential to provide uncertainty for any numerical simulation, thus lowering the cost of performing these verifications while increasing the confidence in those predictions.

Spacecraft requirements can include a maximum airflow speed to protect delicate instruments during ground processing. Computational Fluid Dynamics can be used to verify these requirements; however, the model must be validated by test data. This research includes the following three objectives and methods. Objective one is develop, model, and perform a Computational Fluid Dynamics analysis of three (3) generic, non-proprietary, environmental control systems and spacecraft configurations. Several commercially available and open source solvers have the capability to model the turbulent, highly three-dimensional, incompressible flow regime. The proposed method uses FLUENT, STARCCM+, and OPENFOAM. Objective two is to perform an uncertainty analysis of the Computational Fluid Dynamics model using the

methodology found in "Comprehensive Approach to Verification and Validation of Computational Fluid Dynamics Simulations". This method requires three separate grids and solutions, which quantify the error bars around Computational Fluid Dynamics predictions. The method accounts for all uncertainty terms from both numerical and input variables. Objective three is to compile a table of uncertainty parameters that could be used to estimate the error in a Computational Fluid Dynamics model of the Environmental Control System /spacecraft system.

Previous studies have looked at the uncertainty in a Computational Fluid Dynamics model for a single output variable at a single point, for example the re-attachment length of a backward facing step. For the flow regime being analyzed (turbulent, three-dimensional, incompressible), the error at a single point can propagate into the solution both via flow physics and numerical methods. Calculating the uncertainty in using Computational Fluid Dynamics to accurately predict airflow speeds around encapsulated spacecraft in is imperative to the success of future missions.

This work is dedicated to my wife: Emily…
Thank you for always being there and not settling...

ACKNOWLEDGMENTS

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ACRONYMS AND SYMBOLS

a	=	channel width
c	=	heat transfer coefficient correlation factor
$\boldsymbol{\delta_D}$	=	experimental error
$\boldsymbol{\delta_{input}}$	=	input error
$\boldsymbol{\delta_{model}}$	=	modeling error
$\boldsymbol{\delta_{num}}$	=	numerical error
$\boldsymbol{\delta_S}$	=	simulated error
D	=	experimental value
dp/dx	=	pressure gradient
$\boldsymbol{\varepsilon_{21}}$	=	solution changes medium to fine grid
$\boldsymbol{\varepsilon_{32}}$	=	solution changes coarse to medium grid
$e_a{}^{21}$	=	extrapolated error
E	=	comparison error
EELV	=	evolved expendable launch vehicle
$GCI_{fine}{}^{21}$	=	grid convergence index
h	=	representative grid size
h	=	heat transfer coefficient
l	=	characteristic length
p	=	observed order
$\boldsymbol{\rho}$	=	density

R_k	=	convergence parameter
r_{21}	=	ratio of grid sizes between grid 1 and 2
r_{32}	=	ratio of grid sizes between grid 3 and 2
S	=	simulated result
S_{k1}	=	solution variable for fine grid
S_{k2}	=	solution variable for medium grid
S_{k3}	=	solution variable for coarse grid
$S_{ext}{}^{21}$	=	extrapolated solution variable
S_L	=	lowest solution variable
S_U	=	highest solution variable
$U_{oscillatory}$	=	uncertainty for oscillatory portion of the solution
$U_{monotonic}$	=	uncertainty for monotonic portion of the solution
Uh	=	uncertainty in heat transfer coefficient
Uu	=	uncertainty in velocity profile between parallel plates
u	=	velocity profile between parallel plates
u_{input}	=	input uncertainty
u_D	=	experimental uncertainty
u_{num}	=	numerical uncertainty
u_{val}	=	validation uncertainty
μ	=	viscosity
V	=	velocity
$\overline{V}$	=	average velocity

y = distance from wall

CHAPTER ONE: INTRODUCTION

1.1 <u>Motivation</u>

Delicate spacecraft instruments are needed for satellite technology enhancement of agricultural yield, environment sustainability, or telecommunications. Before spacecraft are released into orbit to complete their science goals, the spacecraft must survive the ground and launch environments. Environmental Control Systems (ECS) systems supply air to keep the spacecraft cool, dry, and clean. Delicate spacecraft instruments are sensitive to high velocity flow from the ECS systems and manufactures set impingement requirements to protect these instruments. CFD is often chosen to complete verifications of the impingement requirements rather than testing. Using CFD to predict the airflow field around a spacecraft enclosed in a fairing has been documented and validated using test data [1], [2].

The problem is there are inherent uncertainties and errors associated with using CFD to predict the airflow field, and there is no standard method for evaluating uncertainty in the CFD community [3]. Some potentials errors include physical approximation error, computer round-off error, iterative convergence error, discretization errors, computer programming errors, and usage errors [4]. An uncertainty, as defined by the American Institute of Aeronautics and Astronautics (AIAA), is a potential deficiency in any phase or activity of modeling and simulation that is due to the lack of knowledge [5]. An example of an uncertainty in performing a CFD analysis is turbulence modeling [4]. There is a lot about turbulence modeling that is not understood [4]. There has been progress in estimating the uncertainty of CFD, but the approaches have not converged [3].

CFD is used primarily for analytical predictions of the velocity, heat transfer coefficient, and pressure. CFD is the current state of the art and industry standard used for spacecraft ECS flow analysis; however CFD has many challenges. The users must select the appropriate models to characterize their specific problem. The proposed research will use different turbulence models as an input uncertainty to help the community evaluate the accuracy of turbulence modeling. There are many other input variables. These include boundary conditions, wall functions, fluid properties, turbulence models, solution schemes, solvers, mesh, and numerical calculations. The current state of the art uncertainty analysis will evaluate each of the error sources and provide the corresponding uncertainty of the velocity around a spacecraft due to the ECS system. No one to date has ever calculated the uncertainty in using CFD to predict the velocity of spacecraft/ECS systems for the entire domain. The benefit to the community is a proven, documented approach, and provided table of all uncertainty variables, which can be used to estimate the error in a velocity prediction.

1.2 Research Goals

The work focuses on solving the following issues:

1. Demonstrate a CFD Uncertainty Analysis for 3-D, low speed, incompressible, highly turbulent, internal flow can be calculated for an entire simulation domain.

2. Investigate a higher order interpolation scheme to be used for grid interpolations and uncertainty quantification.

3. Investigate the applicability of using the ASME 5-Step procedure for the entire computational domain to estimate numerical uncertainties.

4. Calculate the uncertainty in using different turbulent models.

5. Demonstrate this method can contribute to the study of importance of input parameters in CFD.

6. Compile a table for uncertainty estimates by input parameter. The table will benefit the community by providing an uncertainty estimate in lieu of running hundreds of CFD simulations.

7. Demonstrate the ability to use OPENFOAM to calculate the velocity field of an Environmental Control System.

8. Compare the results of OPENFOAM verses an industry standard CFD software program (ie FLUENT and STARCCM+).

1.3 <u>Outline</u>

Chapter Two is a thorough Literature Review to find the "current state of the art" method for performing Computational Fluid Dynamics (CFD) Uncertainty Analysis. Application of the current state of the art method to a simple backward facing academic problem is discussed in Chapter Three. Modeling of the Spacecraft/ECS System is provided in Chapter Four. Chapter Five describes the proposed CFD Uncertainty Approach. Chapter Five applies the proposed CFD Uncertainty Approach to the simple problems of fully developed flow between parallel plates and heat transfer over a flat plate. The results are compared to an exact solution and experimental data.

Chapter 6 documents the approach for spacecraft/ECS systems. The conclusions are provided in Chapter Seven.

1.4 <u>Biographical Sketch</u>

The author, Mr. Curtis Groves, is a PhD candidate at the University of Central Florida. Mr. Groves has worked for NASA at the Kennedy Space Center in the Launch Services Program since 2006 where he performs independent verifications of NASA's science payload requirements. Mr. Groves has performed ECS impingement verifications for the following missions: GLORY, MSL, TDRSS-K/L, and IRIS and external aerodynamics verification on the Atlas V vehicle. Mr. Groves completed dual Bachelor's Degrees in aerospace engineering and mechanical engineering from West Virginia University and graduated Summa Cum Laude. Mr. Groves has graduated from the University of Central Florida with a master's in aerospace engineering in May 2012 and is working to complete a PhD in May 2014. A summary of Mr. Groves' background is provided in Appendix F. Mr. Groves has research interests in Computational Fluid Dynamics, Turbulence Modeling, Uncertainty Analysis, External Aerodynamics, Spacecraft Venting, Environmental Control Systems, and Heat Transfer.

CHAPTER TWO: LITERATURE REVIEW

A literature review was performed to determine the "State of the Art" method for calculating CFD uncertainties. CFD is extensively used in industry, government, and academia to design, investigate, operate, and improve understanding of fluid physics [5]. The rate of growth in using CFD as a research and engineering tool will be directly proportional to the level of credibility that the simulation can produce [5]. One needs to evaluate the uncertainty in the results of a CFD simulation to postulate a level of credibility. In 1986, The American Society of Mechanical Engineers (ASME) Journal of Fluids Engineering published a policy statement stating the need for quantification of numerical accuracy [3]. These statements lead to research on the best method to determine numerical uncertainty. In 1995, Celik and Zhang published "Calculation of Numerical Uncertainty Using Richardson Extrapolation: Application to Some Turbulent Flow Calculations" which used Richardson's Extrapolation method to estimate the uncertainty in CFD [6]. In 1997, Roache published "Quantification of Uncertainty in Computational Fluid Dynamics" [7]. Roaches research also used the Richardson Extrapolation method to quantify CFD uncertainties.

In 1998, the AIAA has published a "Guide for the Verification and Validation of Computational Fluid Dynamics Simulations" [5]. This document provides guidelines for assessing credibility via verification and validation [5]. The document does not recommend standards due to issues not yet resolved, but defines several terms [5]. "Uncertainty is defined as a potential deficiency in any phase or activity of the modeling process that is due to lack of knowledge [5]." "Error is defined as a recognizable deficiency in any phase or activity of modeling and simulation

that is not due to lack of knowledge [5]." "Prediction is defined as the use of a CFD model to

foretell the state of a physical system under conditions for which the CFD model has not been

validated [5]." Uncertainty and error are normally linked to accuracy in modeling and simulation[3].

The guide defines four predominate error sources: insufficient spatial discretization convergence,

insufficient temporal discretization convergence, lack of iterative convergence, and computer

programming, but does not make claims about the accuracy of predictions [5]. The guide

emphasizes that systematically refining the grid size and time step is the most important activity

in verification [5]. Once the grid has been refined such that the discretization error is in the

asymptotic region, Richardson's extrapolation can be used to estimate zero-grid spacing [5]. A

sensitivity analysis and uncertainty analysis are two methods for determining the uncertainty in

CFD [5]. The validation test compares a CFD solution to experimental data [5]. The guide has

outlined the terms and an overall structure to performing validation, but does not offer a

quantitative method.

   In 1999, Stern, Wilson, Coleman, and Paterson, E. G., published Iowa Institute of Hydraulic

Research (IIHR) Report No. 407 titled "Verification and Validation of CFD Simulations" [8]. In

2001, the American Society of Mechanical Engineers (ASME) Journal of Fluids Engineering

published a "Comprehensive Approach to Verification and Validation of CFD Simulations" in an

attempt to provide a comprehensive framework for overall procedures and methodology [9]. Two

papers were published on the subject in Parts I [9] and Parts II [10] and used the methodology

documented in IIHR Report 407. Numerical errors and uncertainties in CFD can be estimated

using iterative and parameter convergence studies [9]. The method uses three convergence

conditions as possible in estimating uncertainties; (1) monotonic convergence which uses

Richardson's extrapolation, (2) oscillatory convergence which uses the upper and lower bounds to estimate uncertainty, (3) divergence in which errors and uncertainties cannot be estimated [9]. The literature provides an approach for estimating errors and uncertainties in CFD simulations for each of the three cases [9], [10], [8]. The approach uses Richardson's extrapolation, which is not new, however; the method has been extended to use input parameters and correction factors to estimate errors and uncertainties [9], [10], [8]. The method examines two sources for error and uncertainty: modeling and simulation. Examples of modeling errors include geometry, mathematical equations, boundary conditions, turbulence models, etc. [9]. Examples of numerical errors include discretization, artificial dissipations, incomplete iterative and grid convergence, lack of conservation of mass, momentum, energy, internal and external boundary non-continuity, computer round-off etc. [4]. The method lacks correlations among errors and assumes these are negligible, which may be inappropriate for some circumstances [9]. Additionally, the method provides a quantitative approach for determining the iterative convergence uncertainty [9]. Iterative Convergence must be evaluated and is typically done by monitoring the residuals order of magnitude drop graphically [9]. For oscillatory convergence, the deviation of a residual from the mean provides estimates of the iterative convergence [9]. This is based on the range of the maximum S_U and minimum S_L values [9]. For convergent iterative convergence, a curve-fit is used [9]. For a mixed convergent/oscillatory, iterative convergence is estimated using the amplitude and the maximum and minimum values [9]. A method for confirming validation is presented as compared to experimental data [9].

In 2008, the International Towing Tank Conference (ITTC) has published "Recommended Procedures and Guidelines – Uncertainty Analysis in CFD Verification and Validation

Methodology and Procedures" [11]. The ITTC guide was largely based off of the methodology and procedures presented in the ASME Journal of Fluids Engineering a "Comprehensive Approach to Verification and Validation of CFD Simulations" [11]. Also in 2008, the ASME Journal of Fluids Engineering published a "Procedure for Estimating and Reporting of Uncertainty Due to Discretization in CFD Applications" [12].

In 2011, the National Energy Technology Laboratory (NETL) conference proceedings held a major section related to CFD Uncertainty Calculation [13]. Celik presented "Critical Issues with Quantification of Discretization Uncertainty in CFD" [13]. The proceedings were based off of the ASME "Comprehensive Approach to Verification and Validation of CFD Simulations" [9].

In 2009, the American Society of Mechanical Engineers published "Standard for Verification and Validation in Computational Fluid Dynamics and Heat Transfer", ASME V&V 20-2009 [14]. The standard provides a procedure for estimating the uncertainty and is based off of the literature presented above.

In 2010, Roy published "Review of Discretization Error Estimators in Scientific Computing" [15]. The paper is quite extensive and discusses the methodology of using the Grid Convergence Index (GCI) that is the basis of the ASME standard and previous research. Roy references the original paper by Roache in 1994 [16]. Roache's GCI method is based off of the original Richardson's Extrapolation in 1911 and 1927 which is a basic concept of having separate solutions for systematically refined grids, one can approximate the exact solution from two systematically refined meshes [15], [17], [18]. The assumptions to Richardson's extrapolation are as follows: 1) both discrete solutions are in the asymptotic range, 2) the meshes have a uniform spacing, 3) coarse

and fine meshes are related through systematic refinement, 4) solutions are smooth, 5) other sources of numerical error are small [15]. Roy further provides a section on open research issues [15]. These include singularities and discontinuities, oscillatory convergence with mesh refinement, multi-scale models, and coarse grid error estimators [15].

2.1 Summary of Literature Review

A thorough literature review has been performed to determine the best method to evaluate the uncertainty in CFD predictions. Both major journals in mechanical and aerospace engineering, AIAA and ASME, have published articles on this subject. The ASME Standard methodology has been adopted by many researchers and provides a detailed approach to calculate uncertainty in CFD from different levels of grid refinement and input parameter studies. The method published by the ASME Journal of Fluids Engineering (ASME V&V 20-2009 [14]) is the state of the art for determining the uncertainty in CFD predictions and was used for the completed research problem.

2.2 Summary of the State of the Art CFD Uncertainty Analysis

A summary of the ASME V&V 20-2009 "**Standard for Verification and Validation in Computational Fluid Dynamics and Heat Transfer**" is provided in this section. The methodology is as follows. The validation comparison error, E, is the difference between the simulated result, S, and the experimental value, D [14]. The goal is to characterize the interval modeling error, δ_{model}. The coverage factor, k, used to provide a given degree of confidence (ie 90% assuming a uniform distribution, k=1.65) [14]. The standard also outlines procedures to

9

calculate numerical uncertainty, u_{num}, the uncertainty in the simulated result from input parameters, u_{input}, and the experimental uncertainty, u_D [14].

$$\delta_{model} \; \varepsilon[E - u_{val}, E + u_{val}] \qquad (1)$$

$$E = S - D \qquad (2)$$

$$u_{val} = k \left(\sqrt{u_{num}^2 + u_{input}^2 + u_D^2} \right) \qquad (3)$$

U_{num} is calculated using a Richardson's Extrapolation approach and defined as a five-step procedure [14].

Step 1, calculate representative grid size, h as shown in (4), (5), (6).

$$h_1 = \left(\frac{Total\ Volume}{total\ number\ of\ cells\ in\ fine\ grid} \right)^{\frac{1}{3}} \qquad (4)$$

$$h_2 = \left(\frac{Total\ Volume}{total\ number\ of\ cells\ in\ medium\ grid} \right)^{\frac{1}{3}} \qquad (5)$$

$$h_3 = \left(\frac{Total\ Volume}{total\ number\ of\ cells\ in\ coarse\ grid} \right)^{\frac{1}{3}} \qquad (6)$$

Step 2 is to select three significantly (r>1.3) grid sizes and computer the ratio as shown in equation (7), (8) [14].

$$r_{21} = \frac{h_2}{h_1} \qquad (7)$$

$$r_{32} = \frac{h_3}{h_2} \qquad (8)$$

Step 3 is to calculate the observed order, p, as shown in equation (11) [14]. This equation must be solved iteratively.

$$\varepsilon_{21} = S_{k2} - S_{k1} \qquad (9)$$

$$\varepsilon_{32} = S_{k3} - S_{k2} \qquad (10)$$

$$p = \left[\frac{1}{\ln(r_{21})}\right] * \left[\ln\left(\frac{\varepsilon_{32}}{\varepsilon_{21}}\right) + \ln\left(\frac{r_{21}^{p} - sign\left(\frac{\varepsilon_{32}}{\varepsilon_{21}}\right)}{r_{32}^{p} - sign\left(\frac{\varepsilon_{32}}{\varepsilon_{21}}\right)}\right)\right] \qquad (11)$$

Step 4 is to calculate the extrapolated values as shown in equation (12) [14].

$$S_{ext}^{21} = \frac{(r_{21}^{p} * S_{k1} - S_{k2})}{(r_{21}^{p} - 1)} \qquad (12)$$

$$e_a^{21} = \frac{(S_{k1} - S_{k2})}{(S_{k1})} \qquad (13)$$

Step 5 is to calculate the fine grid convergence index and numerical uncertainty as shown in equation (14) [14]. This approached used a factor of safety of 1.25 and assumed that the distribution is Gaussian about the fine grid, 90 % confidence.

$$GCI_{fine}^{21} = \frac{1.25 * e_a^{21}}{(r_{21}^{p} - 1)} \qquad (14)$$

$$u_{num} = \frac{GCI_{fine}^{21}}{2} \qquad (15)$$

U_{input} is calculated using a Taylor Series expansion in parameter space [14].

$$u_{input} = \sqrt{\sum_{i=1}^{n} \left(\frac{\vartheta S}{\vartheta X_i} u_{xi} \right)^2} \qquad (16)$$

U_D is calculated using test uncertainty methodology as defied in the standard [14]. The purpose of this paper is to show an estimate of numerical uncertainty without test data. The reader is referred to the ASME standard for further information.

2.3 Proposed Methodology without Test Data

There are a few items to note from the summary of the ASME standard. The summary assumes that there are no random errors and that none of the input variables are correlated. Additionally, the standard states that the numerical error can be calculated by the 5-step procedure, which is essentially Richardson's Extrapolation Method. There are additional assumptions to Richardson's Extrapolation. To apply this method, the variable must be monotonically increasing or decreasing (ie in the extrapolated region). The input variables are assumed to be oscillatory convergence. A convergence study can be calculated to determine if the grid is monotonic, oscillatory, or divergence.

Convergence studies require a minimum of three solutions to evaluate convergence with respect to an input parameter [8]. Consider the situation for 3 solutions corresponding to fine S_{k1}, medium S_{k2}, and coarse S_{k3} values for the *kth* input parameter [8]. Solution changes ε for medium-fine and coarse-medium solutions and their ratio R_k are defined by [8]:

$$\varepsilon_{21} = S_{k2} - S_{k1} \qquad (17)$$

12

$$\varepsilon_{32} = S_{k3} - S_{k2} \qquad (18)$$

$$R_k = \varepsilon_{21} / \varepsilon_{32} \qquad (19)$$

Three convergence conditions are possible [8]:

      (i)   Monotonic convergence: $0 < R_k < 1$

      (ii)   Oscillatory convergence: $R_k < 0^i$

      (iii)   Divergence: $R_k > 1$

The methodology outlined in ASME V&V-2009 [14] assumes monotonic convergence criteria for u_{num}. Further increasing the grid does not always provide a monotonically increasing result. This is shown in AIAA-2013-0258 [19]. The proposed methodology is to treat all input parameters including the grid as an oscillatory convergence study. The uncertainty for cells with oscillatory convergence, using the following method outlined by Stern, Wilson, Coleman, and Paterson [8], can be calculated as follows in equation (20). S is the simulated result. For this case it is the upper velocity S_U and the lower velocity S_L.

$$U_{Oscillatory} = \left| \frac{1}{2}(S_U - S_L) \right| \qquad (20)$$

The proposed methodology as compared to the ASME Standard is as follows. If there is no experimental data, D=0, δ_D=0, and u_D=0.

$$E = S - D = S \qquad (21)$$

$$\delta s = S - T \qquad (22)$$

$$E = S - D = T + \delta s - (T + \delta_D) = \delta_S - \delta_D = \delta_S \qquad (23)$$

$$u_{val} = k \left(\sqrt{u_{num}^2 + u_{input}^2 + u_D^2} \right) = u_{val} = k \left(\sqrt{u_{num}^2 + u_{input}^2} \right) \qquad (24)$$

Report the simulated result, S as

$$S \pm u_{val} \tag{25}$$

Also instead of assuming a gauss-normal distribution as in the standard when including test data, the k-value will come from the Student-T distribution as shown in Table 1. The Student-T distribution is used for experimental uncertainty calculations for a limited number of samples [20].

The quantity of interest for the ECS / spacecraft system is velocity magnitude. Three grids can be compared, and the convergence conditions determined for every point in the computational domain. This is accomplished through interpolation between the medium to coarse grid and the fine to coarse grid. The velocity magnitude from the medium and fine grids are interpolated on to the coarse grid. Then the solutions changes, ε_{21}, ε_{32}, $\mathbf{R_k}$, and convergence conditions are calculated for every point in the domain.

This interpolation can induce errors in the solution, which has been seen by the author in recent publication. The method that was used in the backward facing step used a 'zeroth' order interpolation scheme in FLUENT. Section 5.1 describes an approach to find a higher order interpolation scheme and plot the three different convergence conditions. Treating the grid as a monotonically increasing parameter in the entire domain may be inappropriate. Additionally for an oscillatory convergence parameter, Stern, Wilson, Coleman, and Paterson recommend equation 12.

It will be shown in section 5.2 that treating the grid as an oscillatory input parameter provides an alternative method to estimate the uncertainty in the numeric's.

14

Table 1: Student T Distribution, k Values [20]

Number of Cases	Degrees of Freedom	Confidence 90%
2	1	6.314
3	2	2.92
4	3	2.353
5	4	2.132
6	5	2.015
7	6	1.943
8	7	1.895
9	8	1.86
10	9	1.833
11	10	1.812
12	11	1.796
13	12	1.782
14	13	1.771
15	14	1.761
16	15	1.753
17	16	1.746
18	17	1.74
19	18	1.734
20	19	1.729
21	20	1.725
22	21	1.721
23	22	1.717
24	23	1.714
25	24	1.711
26	25	1.708
27	26	1.706
28	27	1.703
29	28	1.701
30	29	1.699
31	30	1.697
41	40	1.684
51	50	1.676
61	60	1.671
81	80	1.664
101	100	1.66
121	120	1.658
infty	infty	1.645

CHAPTER THREE: APPLYING THE "STATE OF THE ART" CFD UNCERTAINTY ANALYSIS TO A BACKWARD FACING STEP

3.1 Grid Refinement Study and Velocity Prediction

The author applied the ASME standard to a backward facing step in AIAA-2013-0258 [19]. A summary of this paper is included here.

The quantity of interest for the backward facing setup is velocity magnitude. Three grids were compared, and the convergence conditions were determined for every point in the computational domain. This is accomplished through interpolation between the medium to coarse grid and the fine to coarse grid. The velocity magnitude from the medium and fine grids are interpolated on to the coarse grid. Then the solutions changes, $\varepsilon_{21}, \varepsilon_{32}, R_k$, and convergence conditions are calculated for every point in the domain. Figure 1 shows the different convergence conditions inside the computational domain for the grid refinement study.
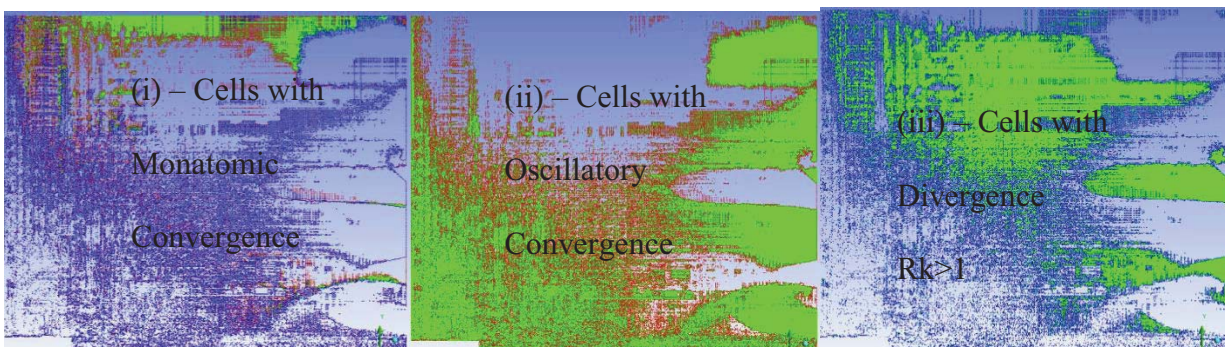
Figure 1: Convergence conditions for the backward facing step – Grid refinement
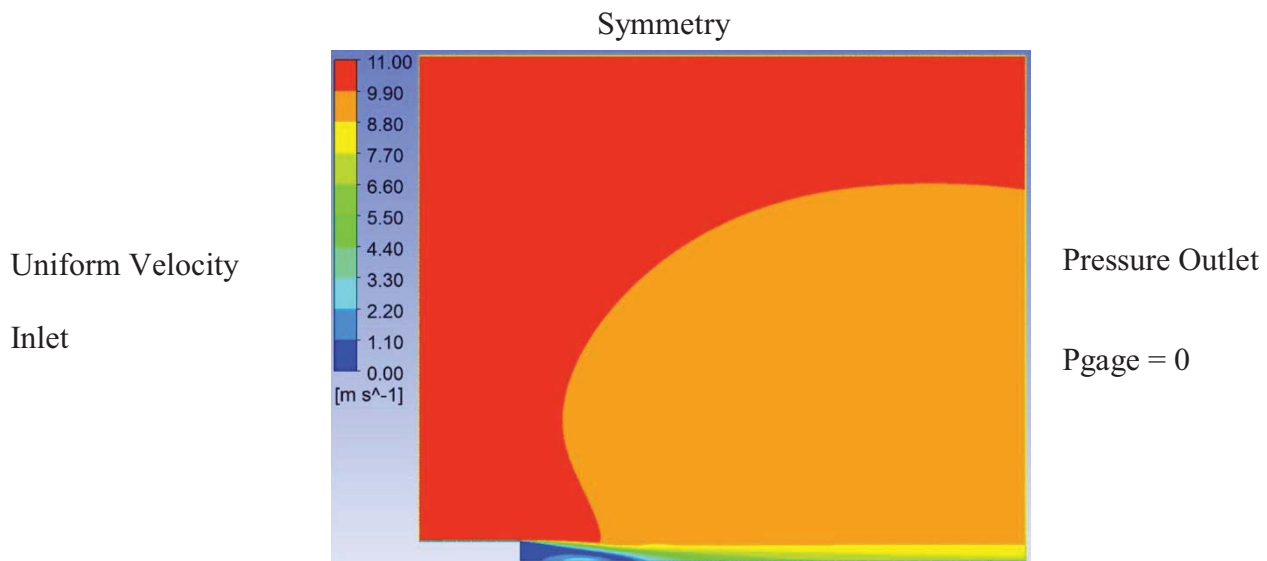
Figure 2: Velocity Magnitude for Flow over a Backward Facing Step – 1,192,000 cells

3.2 CFD Uncertainty Analysis of Backward Facing Step

The following input variables were considered for the uncertainty analysis.

Input Variables:

$$U_{CFD} = \left(\sum_{i=1}^{J} \left\{ \left(\frac{\partial r}{\partial X_i} \right)^2 B_i^2 \right\} \right)^{1/2} \qquad (\,26\,)$$

A list of variables for the k-e-realizable turbulence model analyzed is listed in Table 2. Expanding the data reduction equation for the listed variables as shown in order from top to bottom.

17

$$U_{CFD-Velocity} = \left(\left(\left(\frac{\partial V}{\partial e} \right)^2 B_e^2 \right) + \left(\left(\frac{\partial V}{\partial k} \right)^2 B_k^2 \right) + \left(\left(\frac{\partial V}{\partial p} \right)^2 B_p^2 \right) \right.$$

$$+ \left(\left(\frac{\partial V}{\partial U} \right)^2 B_u^2 \right) + \left(\left(\frac{\partial V}{\partial nu} \right)^2 B_{nu}^2 \right) + \left(\left(\frac{\partial V}{\partial g} \right)^2 B_g^2 \right)$$

$$+ \left(\left(\frac{\partial V}{\partial num} \right)^2 B_{num}^2 \right) + \left(\left(\frac{\partial V}{\partial solver} \right)^2 B_{solver}^2 \right) \qquad (27)$$

$$\left. + \left(\left(\frac{\partial V}{\partial turb} \right)^2 B_{turb}^2 \right) \right)^{1/2}$$

Table 2: Uncertainty Variables, Xi – Backward Facing Step

Type of Variable	Variables Xi	Value	Bias Error
Boundary Conditions	epsilion turbulent mixing length dissipation rate inlet (m2/s3)	0.5	0.5
	k turbulent intensity kinetic energy inlet (m2/s2)	0.05	0.05
	pressure outlet (Pa)	101325	2%
	velocity inlet (m/s)	10	0.5
Fluid Properties	kinematic viscosity nu represents air [0-50-100] deg C	1.79E-06	[13.6e-06 -> 23.06e-06]
Grid Size	Method - Uses Oscillatory Uncertainty	1,192,000 1,862,500 3,311,689	
Numerical	Method - Uses Richardson's Extrapolation (ASME 5 Step Procedure) – Calculated for Velocity at each Cell		
Solver	OpenFOAM (SimpleFoam) vs. Fluent		
Turbulence Models	ke-realiable, kwSST, and SpalartAllmaras		

3.3 <u>Results and Discussion</u>

Each of the variables was analyzed separately for their elemental error sources. The following plots show the each variables and their corresponding uncertainty plot as a function of the percent uncertainty in the CFD Velocity prediction. The percent uncertainty is calculated by dividing by the local velocity (ie the uncertainty velocity in each cell divided by the velocity in each cell). There may be a more appropriate way to non-dimensionalize , such as using the average inlet velocity.

The uncertainty for each of the following was calculated as shown below for each cell using the following method outlined by Stern, Wilson, Coleman, and Paterson [8]. S is the simulated result. For this case it is the upper velocity S_U and the lower velocity S_L.

$$U_{Oscillatory} = \frac{1}{2}(S_U - S_L) \qquad (28)$$

For the epsilon turbulent mixing dissipation rate of the inlet a value of 0.5 +/- 0.5 m²/s³ was used. The uncertainty in the velocity prediction was 0 – 1.155 percent as shown in Figure 3.

Figure 3: Epsilon Turbulent Mixing Length Dissipation Rate Inlet – Velocity Uncertainty Percentage

For the turbulent intensity kinetic energy of the inlet a value of 0.05 +/- 0.05 m²/s² was used. The uncertainty in the velocity prediction was 0 – 0.785 percent as shown in Figure 4.
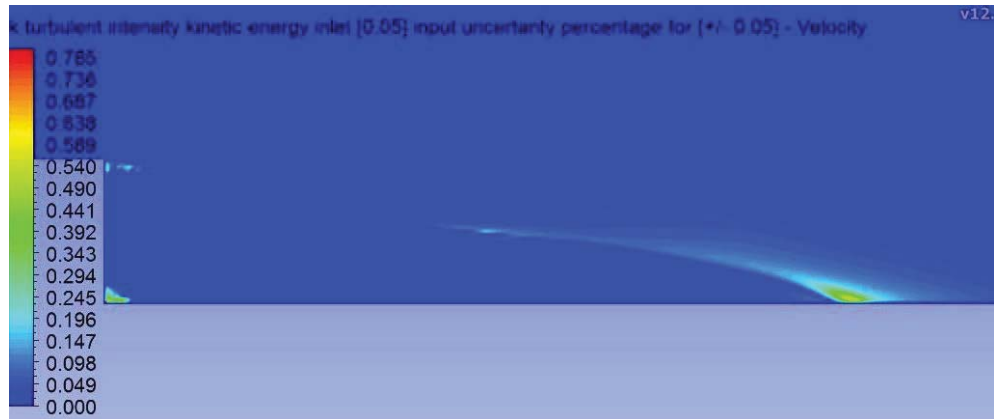
Figure 4: kTurbulent Intensity Kinetic Energy Inlet – Velocity Uncertainty Percentage

For the pressure outlet a value of 101325 +/- 2% Pa was used and the uncertainty in the velocity prediction was 0 – 20 percent as shown in Figure 5.
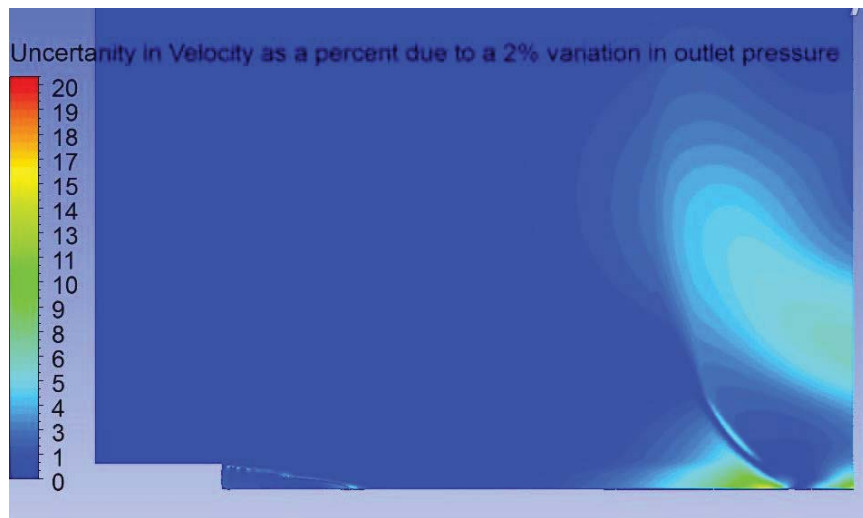
Figure 5: Pressure Outlet – Velocity Uncertainty Percentage

For the inlet velocity a value of 10 +/- 0.5 m/s was used and the uncertainty in the velocity prediction was 0 – 6.558 percent as shown in Figure 6.
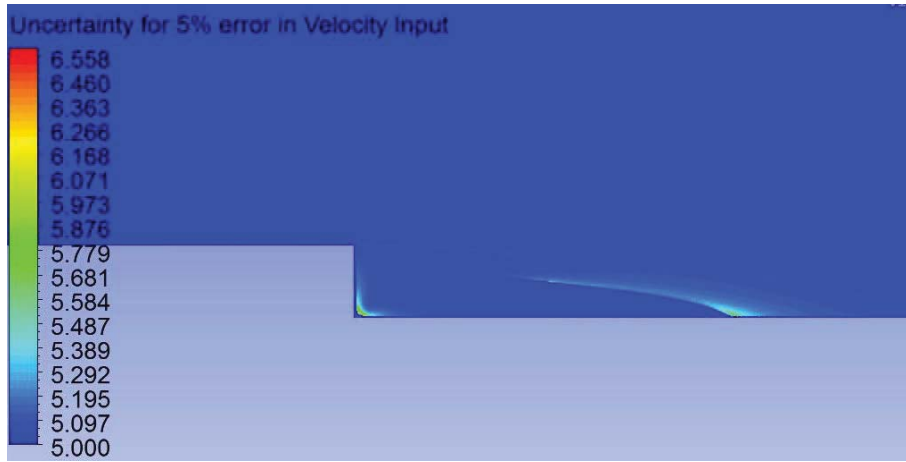
Figure 6: Velocity Inlet – Velocity Uncertainty Percentage

For the kinematic viscosity of air a value of nu=17.06e-06 [13.6e-06 -> 23.06e-06] (m^2/s) was chosen to represent air from 0 to 100 degrees Celsius. The uncertainty in the velocity prediction was 0 – 27.727 percent as shown in Figure 7.

Figure 7: Kinematic Viscosity – Velocity Uncertainty Percentage

For a grid size of 1,192,000 cells [grid 2 -1,862,500 cells], [grid3 - 3,311,689 cells], the uncertainty in the velocity prediction was 0 – 698 percent as shown in Figure 8.
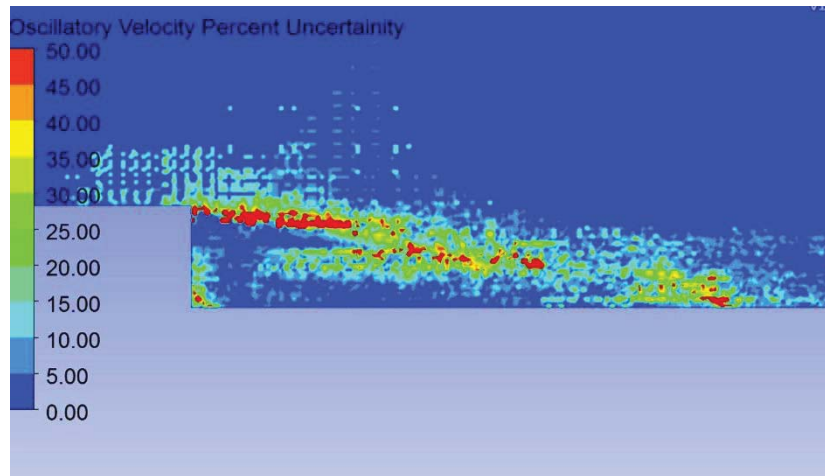
Figure 8: Grid Size – Velocity Uncertainty Percentage

The ke-realiable, kwSST, and SpalartAllmaras turbulence models converged using

OpenFoam and the uncertainty was calculated as an oscillatory input parameter as shown in

Figure 9.
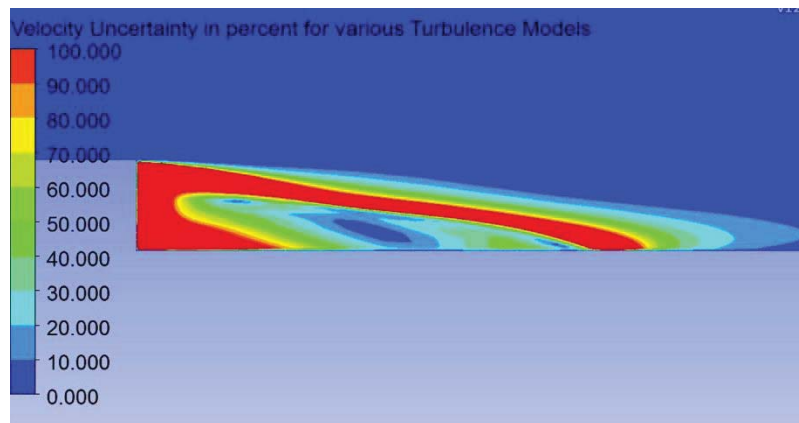
Figure 9: Turbulence Models – Velocity Uncertainty Percentage

OpenFoam and Fluent were used as the solvers to calculate the velocity distribution on the

backward facing step and the uncertainty was calculated as an oscillatory input parameter as

shown in Figure 10.
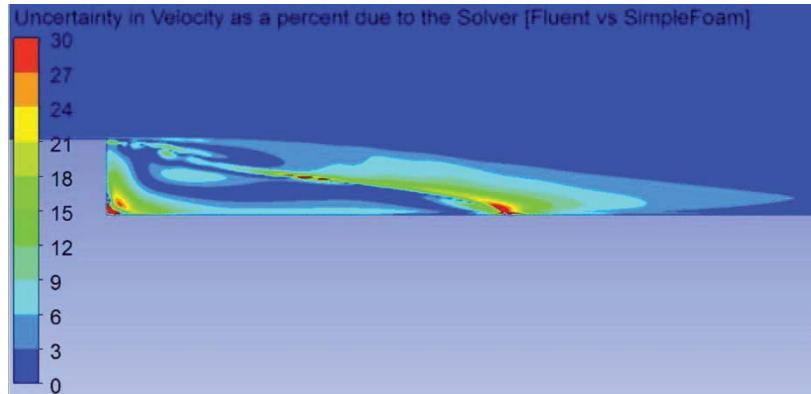
Figure 10: Solver – Velocity Uncertainty Percentage

The uncertainties of the variables with monotonic convergence (numerical) are calculated using Richardson's extrapolation as outlines by ASME V&V-2009 [14]. This is accomplished through the five-step procedure described in equations (4), (5), (6), (7), (8), (9), (10), (11), (12), (13), (14) .

For a grid size of 1,192,000 cells [grid 2 -1,862,500 cells], [grid3 - 3,311,689 cells], the uncertainty in the velocity prediction was 0 – 5300 percent as shown in Figure 11 as estimated by Richardson's extrapolation method.
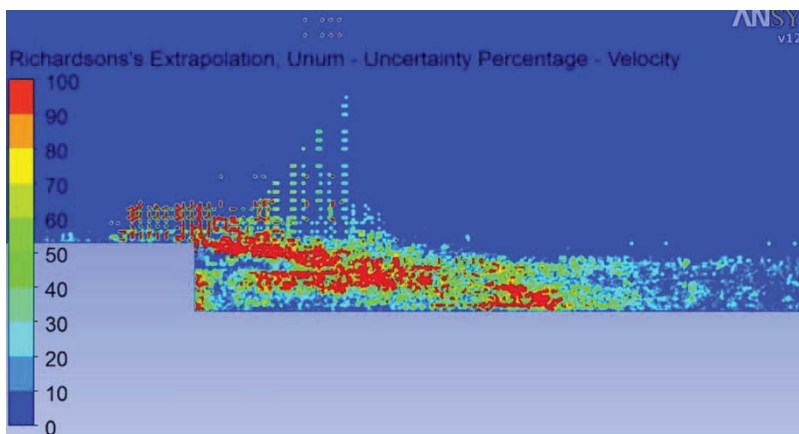
Figure 11: Numerical – Velocity Uncertainty Percentage

A root-sum-squared (rss) of the uncertainty variables was calculated (omitting Richardson's Extrapolation) and the velocity magnitude is shown in Figure 12 with the corresponding uncertainty.
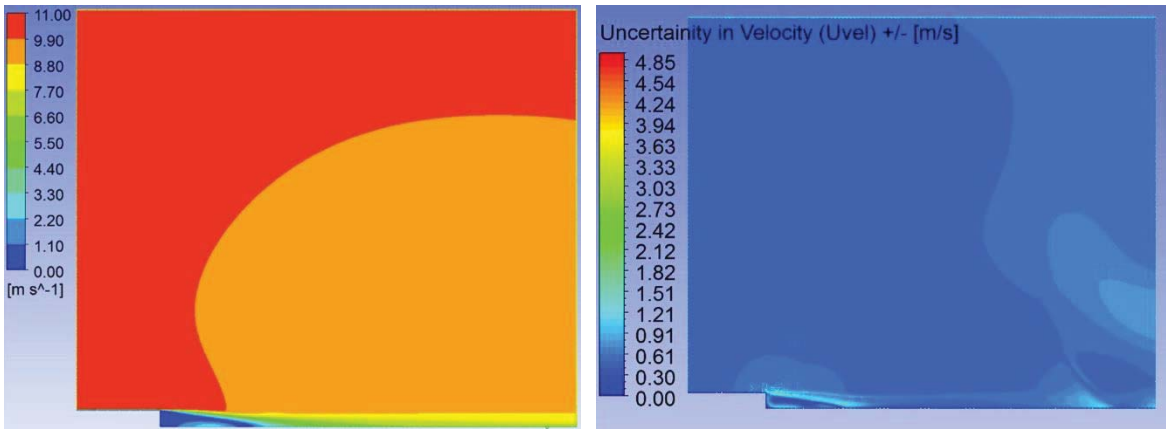
Figure 12: Velocity Prediction and Uncertainty Plot for ke-realizable Turbulence Model

The highest uncertainty is +/- 4.85 m/s. This occurs in the region shown in Figure 14 in red. Figure 14 is the same data presented on the right hand side of Figure 13, except zoomed in to the region near the backward step and a smaller scale is used.
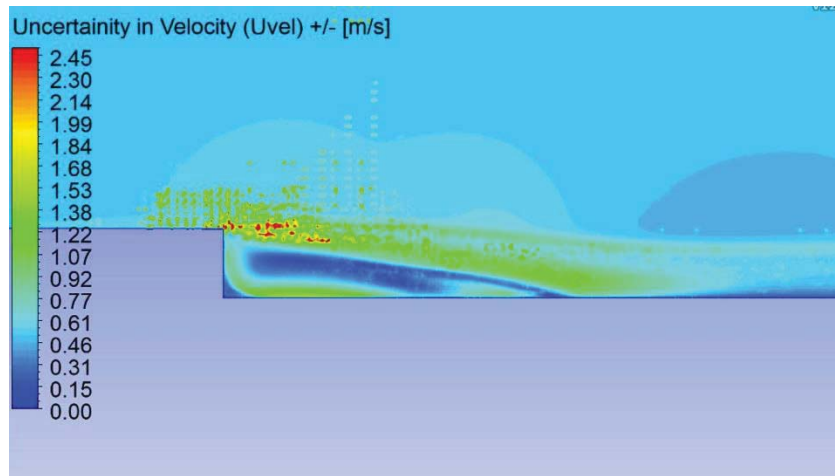
Figure 13: Velocity Uncertainty Plot for ke-realiable Turbulence Model

The monotonic convergence uncertainty calculation was omitted in the rss uncertainty plot due to the values that were produced by using this method. The method produced uncertainty values that were on the order of 5000 percent of the localized velocity in the region near the backward step. It is believed this is due to the turbulence and/or the interpolation between the 3 grids. Turbulence is calculated as a steady state value and fluctuations about that steady state. The fluctuations are inducing a non-linear result between the three grids and providing very large uncertainty bands in the localized region near the backward step. However, once you move approximately 5 lengths downstream of the backward step, the method begins producing reasonable results of 0 – 30 percent of the localized velocity. Treating the highly turbulent region behind the backward step as a monotonic case is inappropriate. It is believed that treating the grid as an input parameter with oscillatory convergence provides better results for a steady state, turbulent CFD simulation. This is evident in the R_k values shown in Figure 2. Most of the cells are exhibiting oscillatory convergence. It is believed all cells are exhibiting oscillatory convergence, however depending on when the sample takes place, one could misrule the results as monotonic or divergent. The interpolation between the three grids could also be inducing this non-linear result. The current method for interpolation is using FLUENT to write out an interpolation file, then reading that file back into FLUENT onto a different grid. This method has been evaluated in section 5.1 and other interpolations methods considered.

CHAPTER FOUR: SPACECRAFT ECS SYSTEM OVERVIEW AND MODELING

4.1 Spacecraft ECS System Overview

The author published the work in this chapter as AIAA-2014-0440 [21]. Prior to launch, cold air (air conditioning) flows downward around the spacecraft after it has been encapsulated in the Payload Fairing [2]. The cold air is delivered through an air-conditioning (AC) pipe, which intersects the fairing and flows past a diffuser located at the pipe/fairing interface [2]. After passing over the spacecraft, it is finally discharged through vents [2]. The Payload Fairing air conditioning is cut off at lift off [2]. An overview of the geometry for an Environmental Control System (ECS) along with the swirled airflow is shown in Figure 14 and Figure 15, respectively.
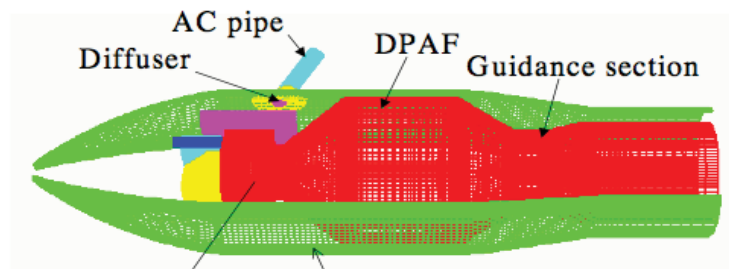
Figure 14: Environmental Control System (ECS) Overview [22]
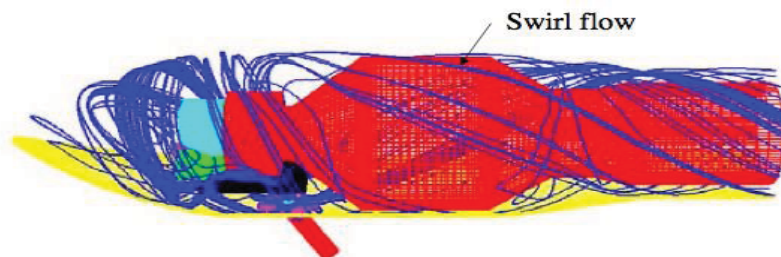
Figure 15: Environmental Control System (ECS) Airflow Swirl [22]

This problem has been previously solved using overset grids and compared to laser doppler test data as described in AIAA-2005-4910 [23]. An example of the airflow testing performed is shown in Figure 16.

Figure 16: Environmental Control System (ECS) Airflow Testing [24]

The example shown above is the only published result of the ECS airflow problem. It is difficult to publish this material due to the proprietary information needed. There are seven different rockets currently being used in the United States for Evolved Expendable Launch Vehicles (EELV). These rockets include the Delta II, Delta IV, Atlas V, Pegasus, Taurus, and Falcon 9 [25]. A summary of each of these rockets's ECS systems that are available in the public information is included below. To investigate the problem (3) generic representations were created that encompass the flow regimes seen in the EELV fleet. Each of these rockets has a publicly

available source called a payload planners guide or users guide. Each of these guides has been studied extensively and the appropriate information related to the ECS systems are presented next.

4.1.1 Publically Available Information on EELV ECS Systems

For the Delta II vehicle, air-conditioning is supplied to the spacecraft via an umbilical after the payload fairing is mated to the launch vehicle [26]. The payload air-distribution system provides air at the required temperature, relative humidity, and flow rate as measured [26]. The air-distribution system uses a diffuser on the inlet air-conditioning duct at the fairing interface [26]. If required, a deflector can be installed on the inlet to direct the airflow away from sensitive spacecraft components [26]. The air can be supplied to the payload between a rate of 1300 to 1700 scfm [26]. The diameter of Fairing is 3 meters [26].

For the Delta IV, the air is supplied to the payload at a maximum flow rate of 36.3 kg/min to 72.6 kg/min (80 to 160 lb/min) for 4-m fairing launch vehicles and 90.7 kg/min to 136.0 kg/min (200 to 300 lb/min) for 5-m fairing launch vehicles [27]. Air flows around the payload and is discharged through vents in the aft end of the fairing [27]. Fairing sizes are 4 meters and 5 meters in diameter [27].

For the Atlas V, internal ducting defectors in the PLF direct the gas upward to prevent direct impingement on the spacecraft [28]. The conditioning gas is vented to the atmosphere through one-way flapper doors below the spacecraft [28]. The PLF air distribution system will provide a maximum air flow velocity in all directions of no more than 9.75 mps (32 fps) for the Atlas V 400 and 10.67 mps (35 fps) for the Atlas V 500 [28]. There will be localized areas of higher flow

28

velocity at, near, or associated with the air conditioning outlet [28]. Maximum air flow velocities correspond to maximum inlet mass flow rates [28]. Reduced flow velocities are achievable using lower inlet mass flow rates [28].

- Flow Rates

    A) Atlas V 400: 0.38–1.21 kg/s ±0.038 kg/s (50–l60 lb/min ±5 lb/min), [28]

    B) Atlas V 500: 0.38–2.27 kg/s ±0.095 kg/s (50–300 lb/min ±12.5 lb/min) [28]

The fairing sizes are 4meters and 5 meters in diameter [28].

For the Pegasus vehicle, the fairing is continuously purged with filtered air [29]. The flowrate of air through the fairing is maintained between 50 and 200 cfm [29]. The air flow enters the fairing forward of the payload and exits aft of the payload [29]. There are baffles on the inlet that minimize the impingement velocity of the air on the payload [29]. The fairing diameter is 0.97 meters [29].

For the Taurus vehicle, upon encapsulation within the fairing and for the remainder of ground operations, the payload environment will be maintained by the Taurus Environmental Control System (ECS) [30]. The fairing inlet conditions are selected by the Customer [30]. The fairing diameters are 63 inches and 92 inches [30].

For the Falcon 9 vehicle, once fully encapsulated and horizontal, the Environmental Control System (ECS) is connected [31]. Payload environments during various processing phases are [31]:

- In hanger, encapsulated – Flow Rate: 1,000 cfm [31]
- During rollout: 1,000 cfm [31]

- On pad: Variable from 1000 to 4500 cfm [31]

The fairing diameter is 5.2 meters [31].

4.1.2 Modeling and CFD Analysis of (3) Generic Non-Proprietary Environmental Control System and Spacecraft Configurations

The following information can be concluded about the publically releasable ECS system data presented in the previous section. The fairing sizes are approximately 1m, 1.6m, 2.3m, 3m, 4m, 5m in diameter. The following (3) generic fairing diameters are selected to envelop the EELV fairing configurations as follows.

- 0.75m
- 3.5 m
- 5.5 m

The inlet conditions range from 1000 cfm to 4500 cfm.

The three proposed generic models have been created via Computer Aided Drafting (CAD) model software Pro/ENGINEER. The configurations are shown in Figure 17, Figure 18, and Figure 19, respectively.
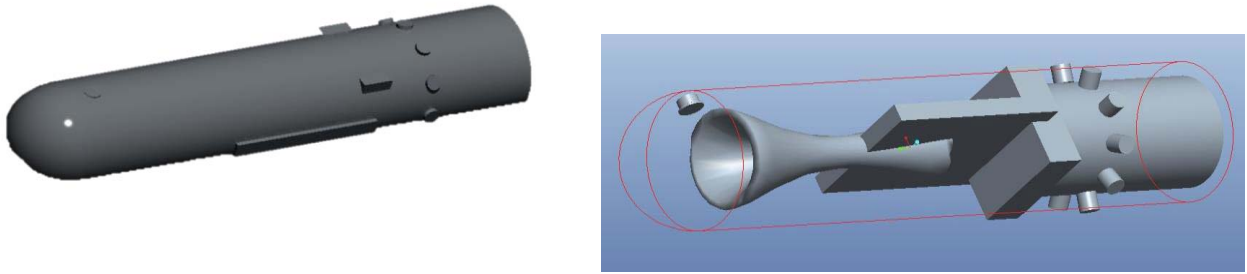
Figure 17: 0.75m Diameter Fairing CAD Model of ECS / Spacecraft
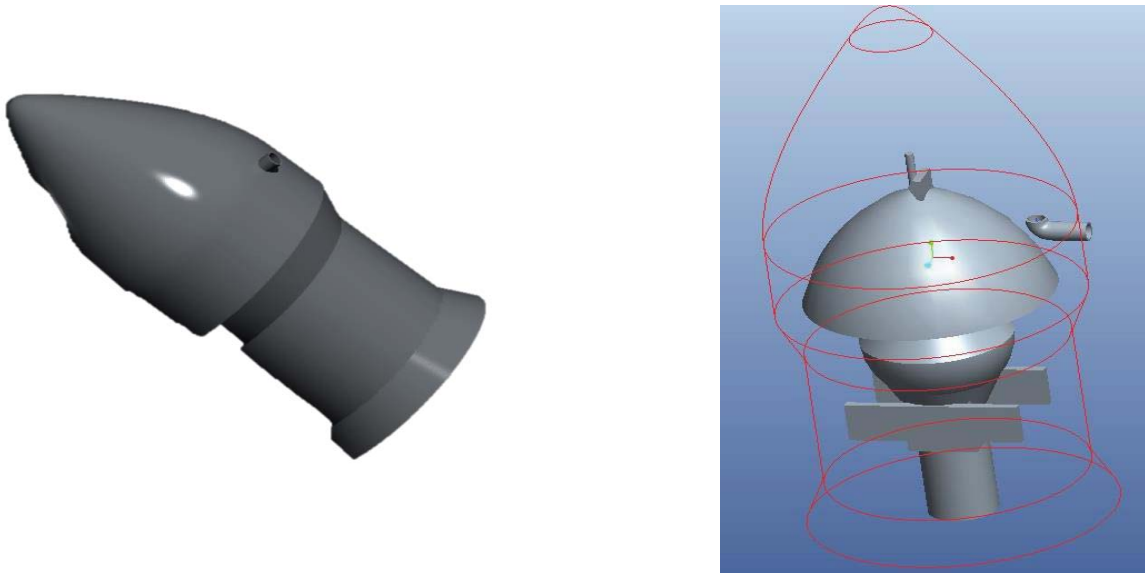
Figure 18: 3.5m Diameter Fairing CAD Model of ECS / Spacecraft
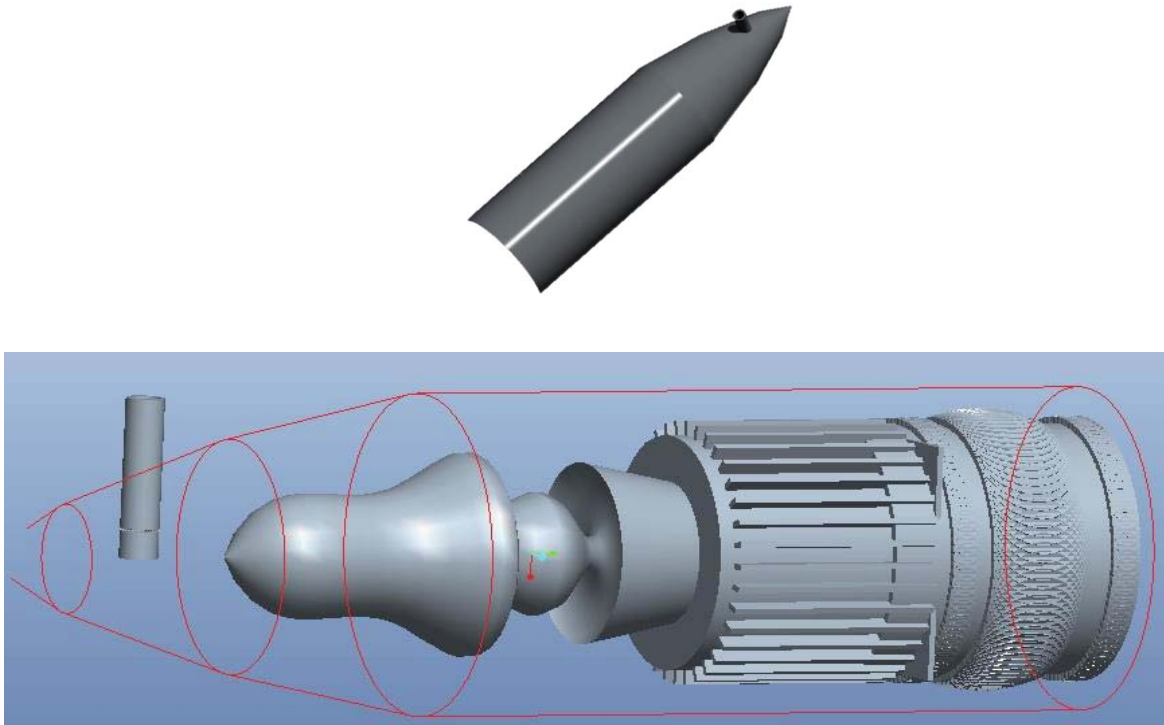
Figure 19: 5.5m Diameter Fairing CAD Model of ECS / Spacecraft

OPENFOAM was used as the primary solver. FLUENT and STARCCM+ were only used to access the uncertainty of the solver. OPENFOAM is more versatile for this research problem due to the open source code and no licensing issues. OPENFOAM additionally has the capability through snappy hex-mesh to import the CAD as an .STL files and mesh the geometry. Each of the CAD models shown above that were created in Pro/ENGINEER were translated into an .STL file and meshed using snappy hex-mesh. The 0.75 meter geometry is shown in Figure 20, 3.5 meter geometry in Figure 21, and the 5.5 m geometry shown in Figure 22.
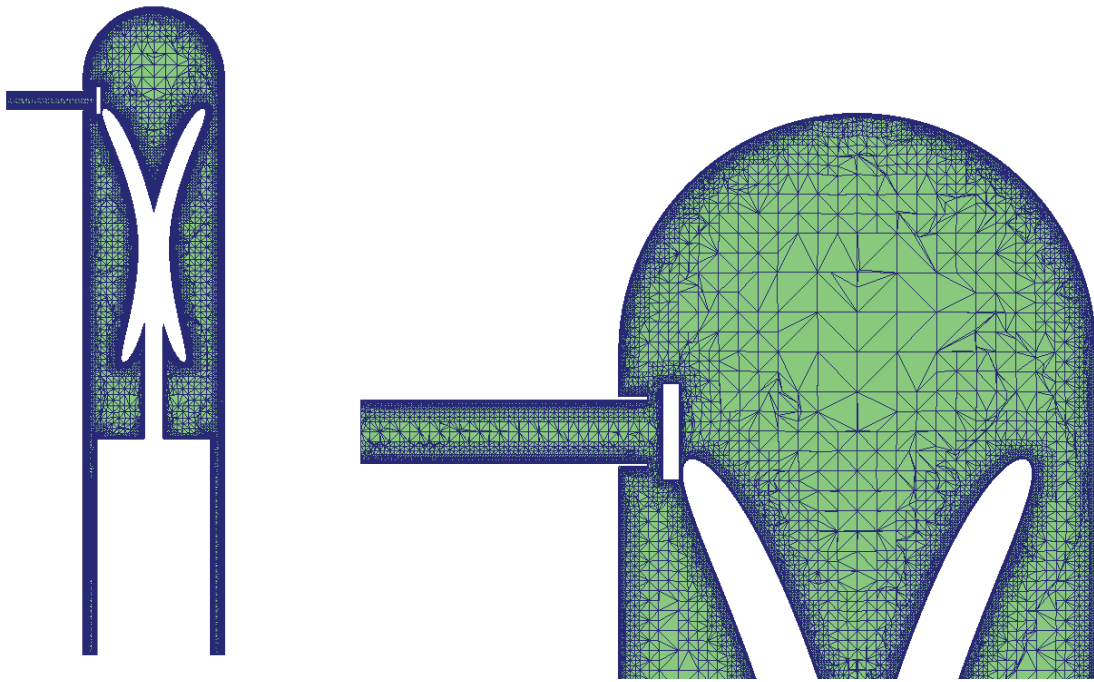
Figure 20: OpenFOAM SnappyHex Mesh Modeling CAD Model of ECS / Spacecraft – 0.75m Configuration (6762865 number of cells)

Figure 21: OpenFOAM SnappyHex Mesh Modeling CAD Model of ECS / Spacecraft – 3.5m Configuration (8594480 number of cells)
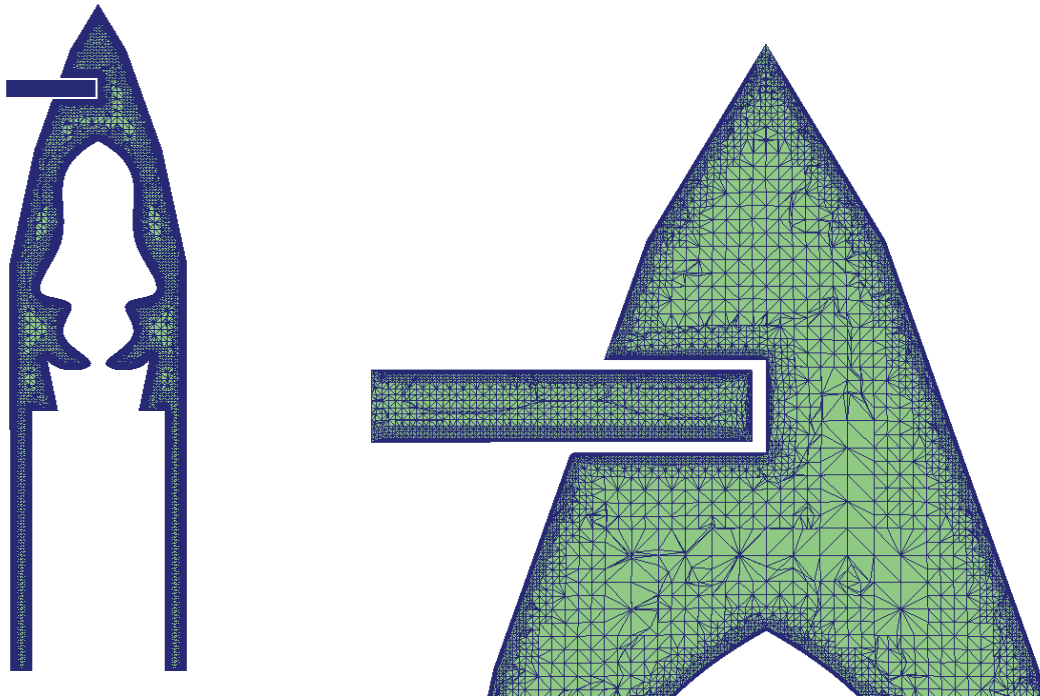
Figure 22: OpenFOAM SnappyHex Mesh Modeling CAD Model of ECS / Spacecraft – 5.5m Configuration (6980673number of cells)

The results from the CFD modeling using OPENFOAM's simpleFoam are displayed using ParaFoam. The velocity contours are shown for each configuration in Figure 23, Figure 24, and Figure 25, respectively.
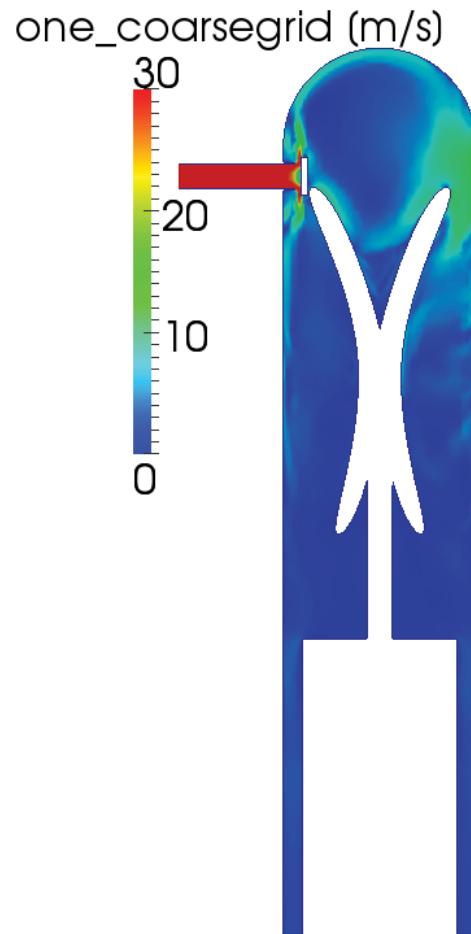
Figure 23: OpenFOAM SimpleFOAM Velocity Contours of ECS / Spacecraft – 0.75m Configuration
(6762865 number of cells)
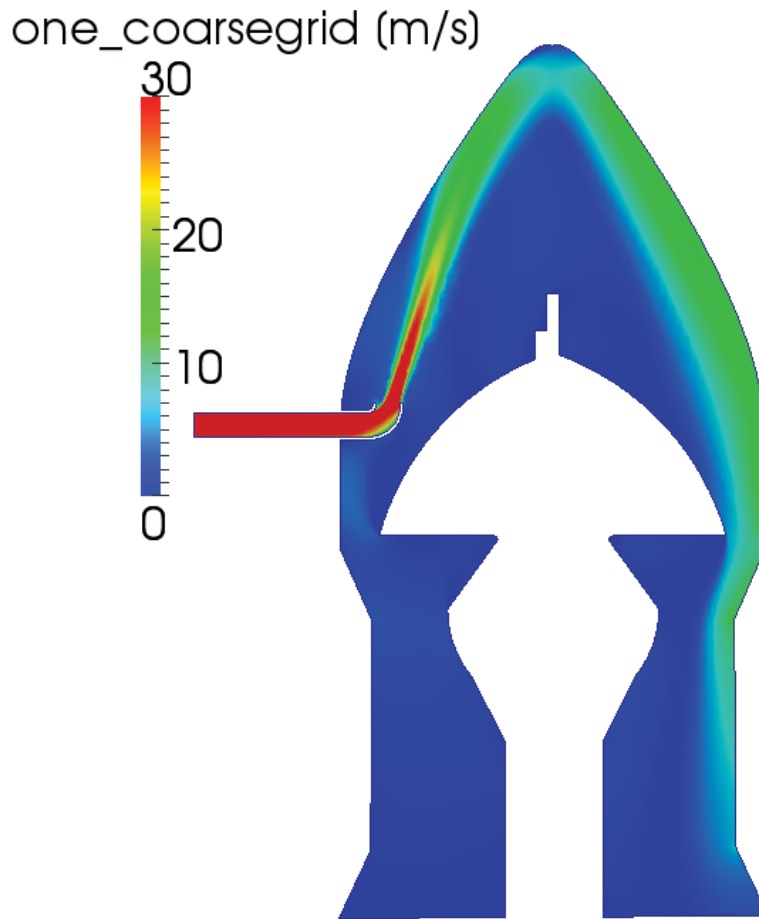
Figure 24: OpenFOAM SimpleFOAM Velocity Contours of ECS / Spacecraft – 3.5m Configuration
(6980673number of cells)
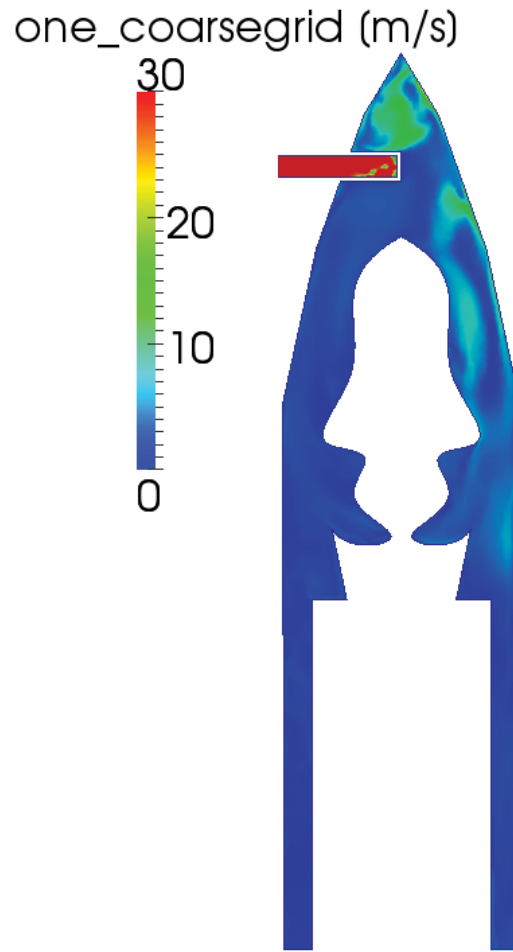
Figure 25: OpenFOAM SimpleFOAM Velocity Contours of ECS / Spacecraft – 5.5m Configuration (6980673number of cells)

CHAPTER FIVE: COMPUTATIONAL FLUID DYNAMICS UNCERTAINTY ANALYSIS

The uncertainty of using CFD to analyze the ECS system for airflow velocities around spacecraft is unknown and not documented. The objective of this section is to apply the methodology laid out in section 2.3. There are several items that will need to be addressed in the section. First, the interpolation scheme that was used for the backward step is unacceptable. The interpolation scheme is introducing errors that make using the monotonic (Richardson's extrapolation method) un-realistic. A better interpolation scheme between the three grids will be developed. Second, using the monotonic numerical results and extrapolating a solution for the entire computational domain will need to be assessed for feasibility. It may prove that using the grid as a separate oscillatory input parameter will suffice. To complete this objective a Comprehensive Approach to Verification and Validation of CFD Simulations – ASME Journal of Fluids Methodology outlined in previous section 2 will be used as a starting point and any inconsistencies or issues will be analyzed and solutions recommended. Again summarizing the method, three separate grids (rough, medium, fine) along with the uncertainty of all input parameters will be used to evaluate the uncertainty in the CFD velocity prediction. The velocity at every point in each of the three solutions will be compared to one another.

Before proceeding directly into the spacecraft/ECS system, two simpler problems are analyzed using the proposed methodology (laminar flow between parallel stationary plates and heat transfer over a flat plate).

5.1 Interpolation Scheme Needed for CFD Uncertainty Analysis

The author published the work in this section as AIAA-2014-1433 [32]. CFD is used in many forums to approximate flow solutions of the Navier-Stokes equations. The Navier-Stokes equations are second order, non-homogenous, non-linear partial differential equations. Several papers have been published on the use of progressive grid refinement to estimate the errors in a CFD Simulation [8], [5], [33], [9], [7], [6], [10], [11], [12], [14], [19]. The procedure is to compare the differences in the solutions between at least three different grids. The computational domain discretization (grid) is significantly different in terms of the number of cells. This requires an interpolation between the grids and solutions to approximate the error. This interpolation will induce errors and the extrapolated uncertainty estimates become unreasonable and inaccurate [19]. The commercially available code ANSYS FLUENT includes mesh-to-mesh interpolation functionality [34]. This method performs a zeroth-order interpolation (nearest neighbor) for interpolating the solution data from one mesh to another [34]. This functionality is used to initialize data from one mesh on to another mesh for the purpose of an initial condition only. Using this method to approximate errors is inappropriate. OPENFOAM is an open source solver, which includes a similar functionality to the FLUENT mesh-to-mesh interpolation using a "mapfields" function [35]. The mapfields function is also designed as an initial guess to be used when iterating a solution. Using the FLUENT interpolation file or OpenFOAM mapfields to estimate errors in a grid convergence study will produce unrealistic results. The purpose of this section is to compare several other interpolation schemes that could be used for post-processing different solutions on different grids for the purpose of uncertainty estimation.

Matlab is a high-level language used for numerical computations and includes several interpolation functions for one-dimensional data, uniformly spaced, gridded data in two and three dimensions, and scattered data interpolation [36]. CFD data comes in various forms, 1D, 2D, 3D, uniform, and non-uniform data. Matlab offers interp1, interp2, and interp3 for the corresponding dimensions. Interp1,2,3 includes the following schemes as shown in Table 3.

Table 3: Interpolation Schemes Available in Matlab

	Matlab Function			
	interp1	interp2	interp3	
Interpolation Method				
'nearest' - Nearest neighbor interpolation	X	X	X	
'linear' - Linear interpolation (default)	X	X	X	
'spline' - Cubic spline interpolation	X	X	X	
'pchip' - Piecewise cubic Hermite interpolation	X			
'cubic'	X	X (uniformly-spaced only)	X (uniformly-spaced only)	
'v5cubic' - cubic interpolation used in Matlab 5	X			

A generic scheme is sought that would be available for 1D, 2D, 3D, uniform, and non-uniform grids. The three schemes available are "nearest", "linear", and "spline". The nearest is the same scheme available in the current CFD codes for mesh-to-mesh interpolation and would not provide additional benefit. The linear scheme and spline however could provide a better interpolation method for estimating numerical uncertainty in grid refinement studies.

Flow between parallel plates has an exact solution and provides a good example of the interpolation errors that can be induced by using the "nearest" scheme and will provide a metric for comparing the errors in the "linear" and "spline" to an exact solution.

Fully developed laminar flow between stationary, parallel plates is an exact solution to the Navier-Stokes Equations as derived in "Introduction to Fluid Mechanics" [37]. The width of the channel is (a).

$$u = \frac{a^2}{2\mu}\left(\frac{\partial P}{\partial x}\right)\left[\left(\frac{y}{a}\right)^2 - \left(\frac{y}{a}\right)\right] \qquad (29)$$

A CFD model of this problem was created in FLUENT. The fluid is air. Table 4 outlines the parameters used.

Table 4: Parameters for Exact Solution

a (m)	0.1
rho (kg/m3)	1.225
mu (Ns/m2)	0.00001789
dp/dx (N/m3)	-0.0004

The exact solution is shown in Figure 26.
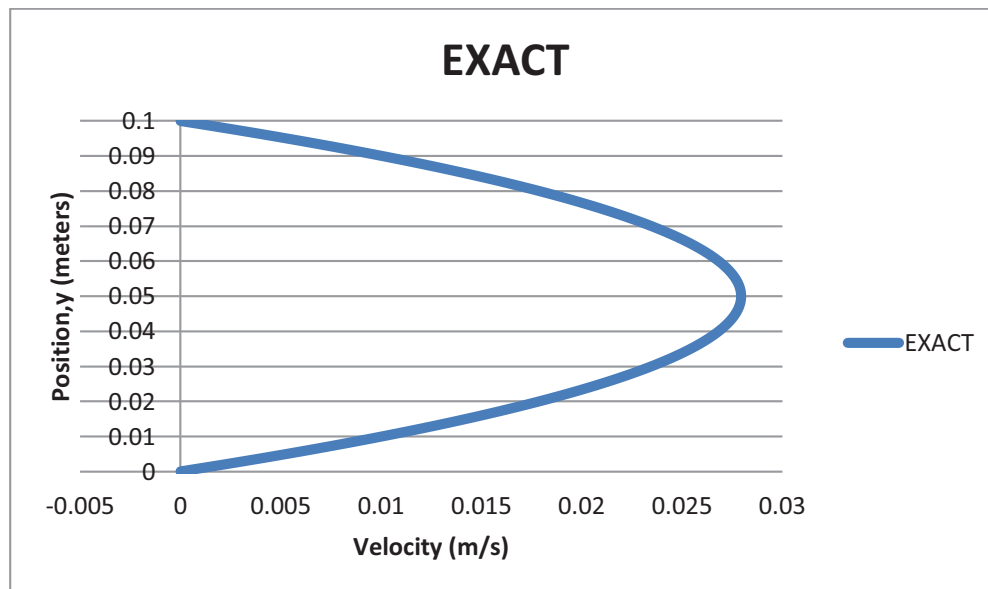
Figure 26: Exact Solution

A CFD model was created for half of the domain. Flow between parallel plates has symmetry about the centerline. The inlet boundary condition used was the average velocity as shown in equation 16 and the domain was made long enough to be considered fully developed.

$$\bar{V} = -\frac{1}{12\mu}\left(\frac{\partial P}{\partial x}\right)a^2 \qquad (\,30\,)$$

Three grids can be used to extrapolate an error. Three separate CFD models were created (coarse, medium, and fine). The coarse medium and fine grids have the following number of cells, 7140, 14186, 24780, respectively. The three solutions plotted for flow between parallel plates and compared to the exact solution are shown in Figure 27.
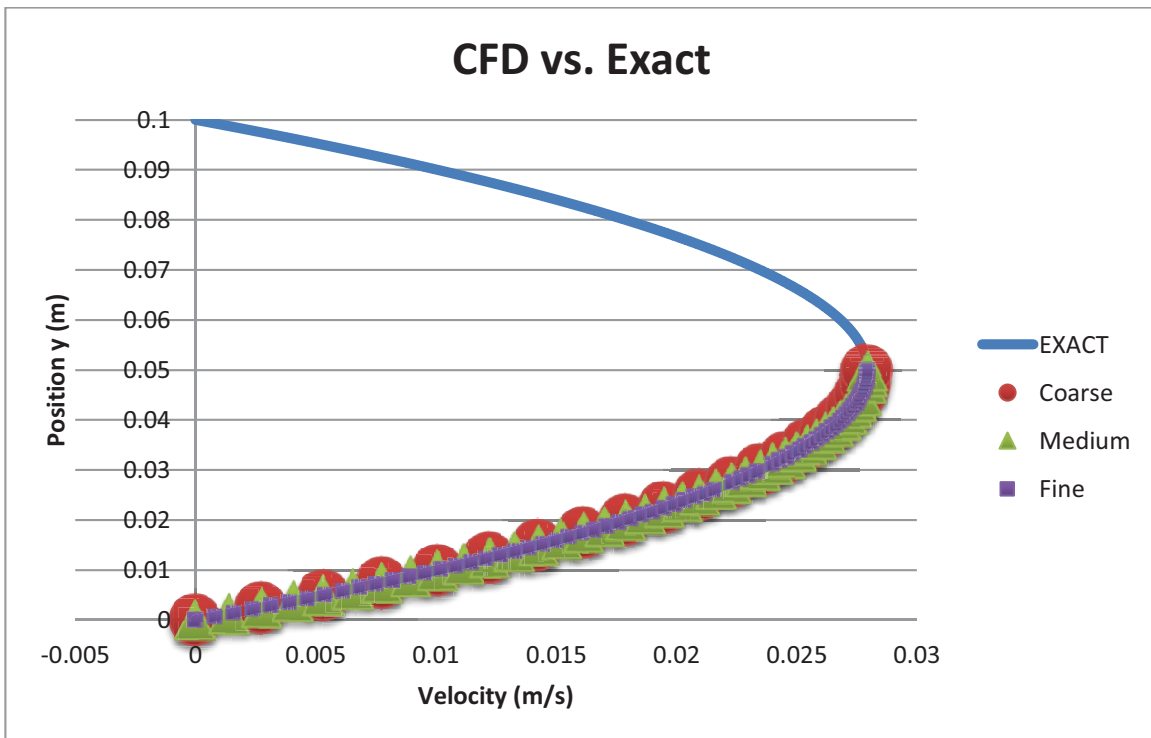
Figure 27: CFD Results (coarse, medium, fine) vs. Exact Solution

The interpolation methods outlined "nearest", "linear", and "cubic" were investigated by interpolating the results from the fine grid and medium grid onto the coarse grid. The coarse grid was chosen because ideally it should be good enough to approximate the solution and all recommendations for normal grid refinement already followed. The medium and fine grids are used only for the error approximation.

FLUENT's mesh-to-mesh interpolation functionality was used and the results are shown in Figure 28. From a plot of the entire computational domain, the reader would not be able to see the variation, so the plot was zoomed in to show the errors being induced by using the "nearest" interpolation.
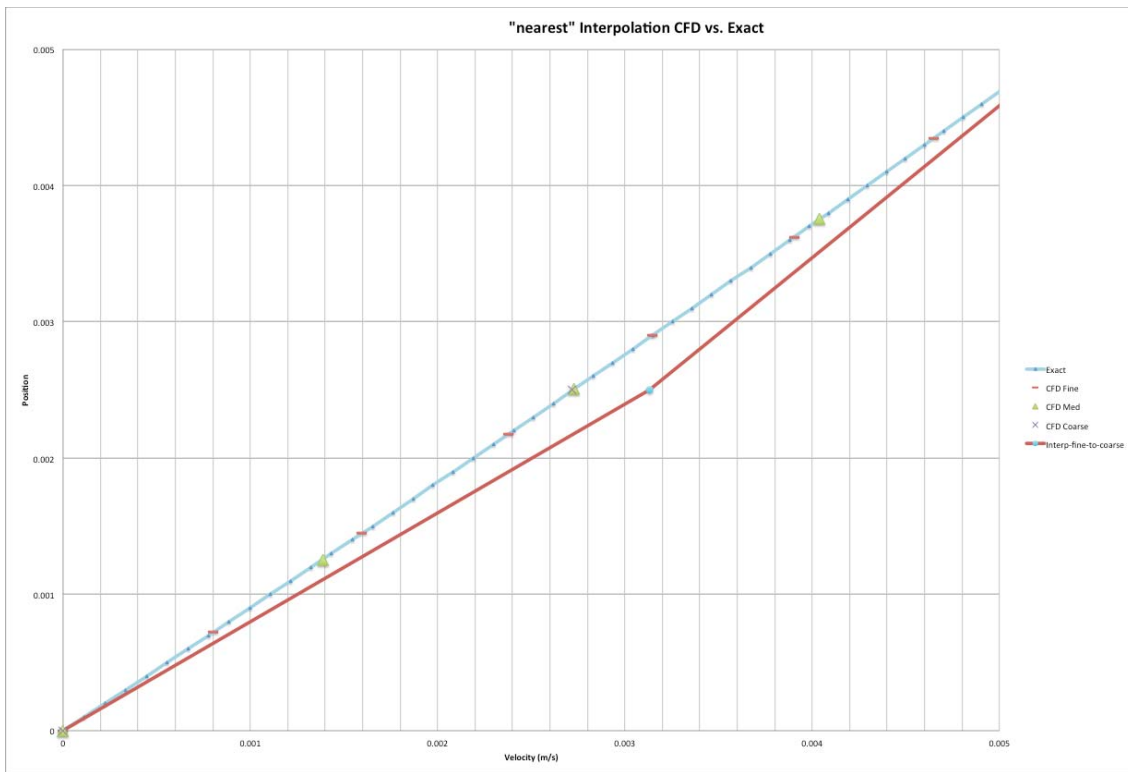
Figure 28: "nearest" interpolation of CFD Results (coarse, medium, fine) vs. Exact Solution

The error induced by using the "nearest" interpolation scheme was as high as 15 percent of the

exact value. Also, this high error was in the critical region closest to the wall.

To compare the linear interpolation scheme Matlab was used.

$$\text{yfi} = \text{interp1(fine(:,2),fine(:,1),coarse(:,2),'linear')}$$

The percent difference was greatly reduced to 0.08 percent of the exact solution and plotting

the results does not visually show a difference as shown in Figure 29.
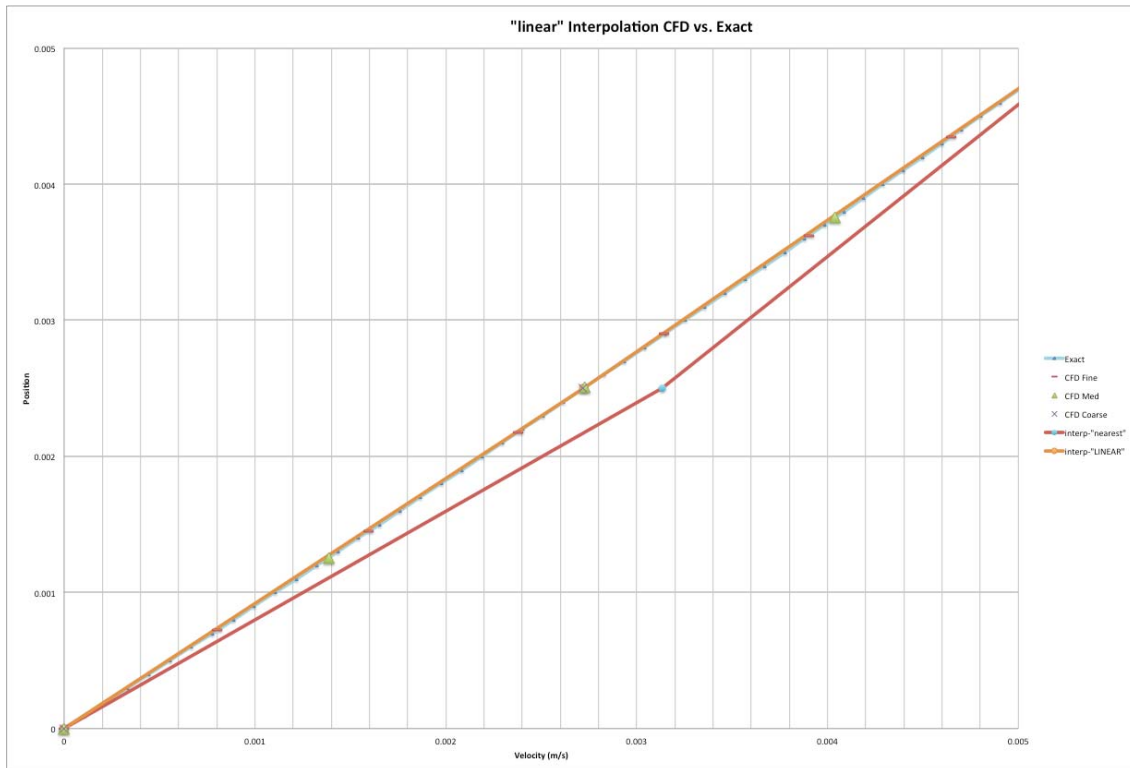
Figure 29: "linear" interpolation of CFD Results (coarse, medium, fine) vs. Exact Solution

To compare the cubic interpolation scheme, Matlab was again used as follows.

$$yfi = interp1(fine(:,2),fine(:,1),coarse(:,2),'cubic')$$

The percent difference was even further reduced to 0.07 percent of the exact solution and plotting the results does not visually show a difference as shown in Figure 30.
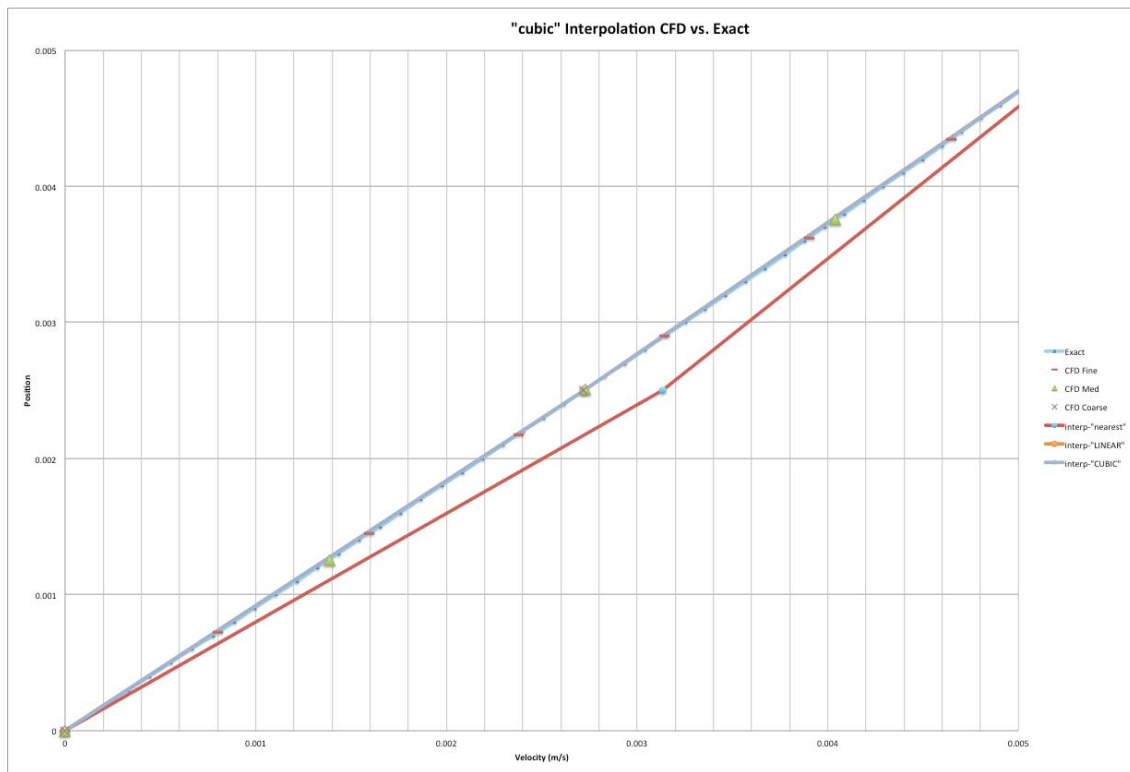
Figure 30: "cubic" interpolation of CFD Results (coarse, medium, fine) vs. Exact Solution

The method was extended to 2D and 3D. Matlab uses a meshgrid function, which is needed before using the interp2 or interp3 functions. Using the meshgrid function transforms the domain specified by vectors into arrays. Performing this option for CFD meshes in the 4 million to 8 million cells yields a "maximum variable size allowed by the program is exceeded". Using the interp2 or interp3 did not provide a feasible way to perform the interpolation.

Matlab contains "griddata" function, which includes the following methods ('nearest', 'linear', 'natural', 'cubic', and 'v4'). The only options available in both 2D and 3D are 'nearest', 'linear', and 'natural'. Again, a general method is sought that works in 1D, 2D, and 3D. The only options available in interp1 and griddata are 'nearest' and 'linear'. The linear method is considered only from this point forward.

A CFD model was created for a 3D internal flow problem and three grids considered. The 'linear' option was used to interpolate between the fine and coarse grids. It can be shown that the interpolation does induce some error between the meshes; however, the error is very similar to the variation between the grids. Figure 31 shows the velocity contour plots between the fine grid and the coarse grid. The contour plots are visually identical.

A line plot was constructed comparing the three grids and the interpolated solutions as shown in Figure 32. The line is extracted from the contour plot as shown with the pink line in Figure 33.
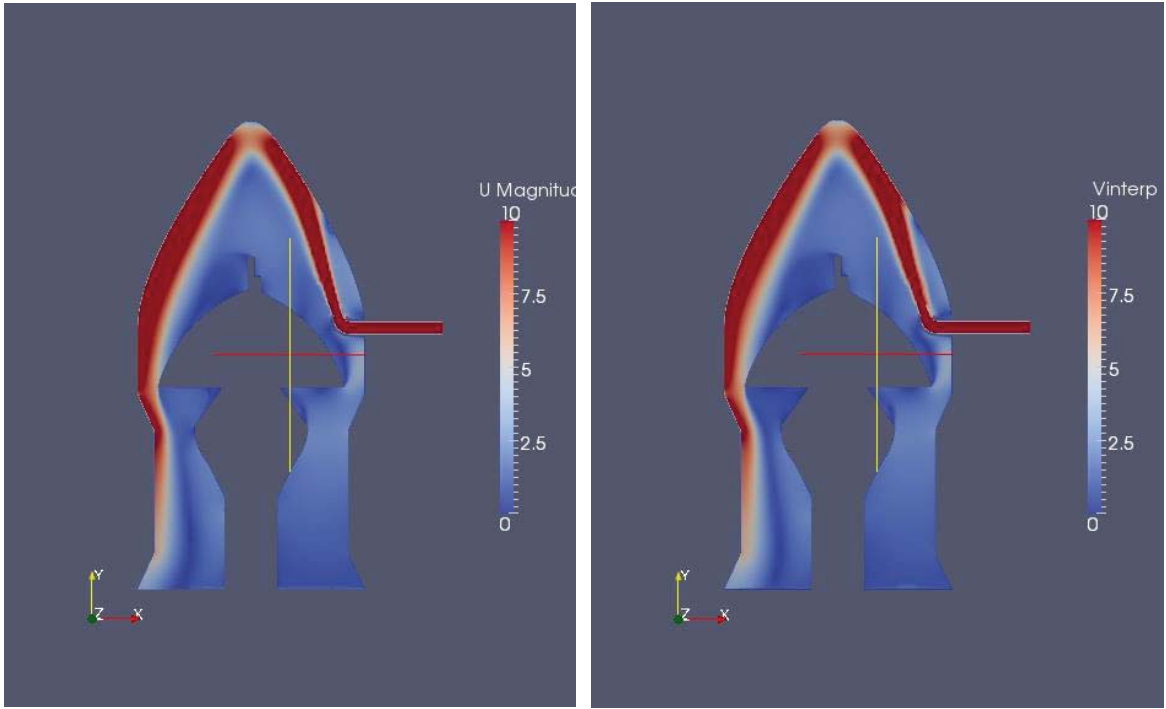
Figure 31: 3D Linear Interpolation Contour Plot vs. Fine Solution
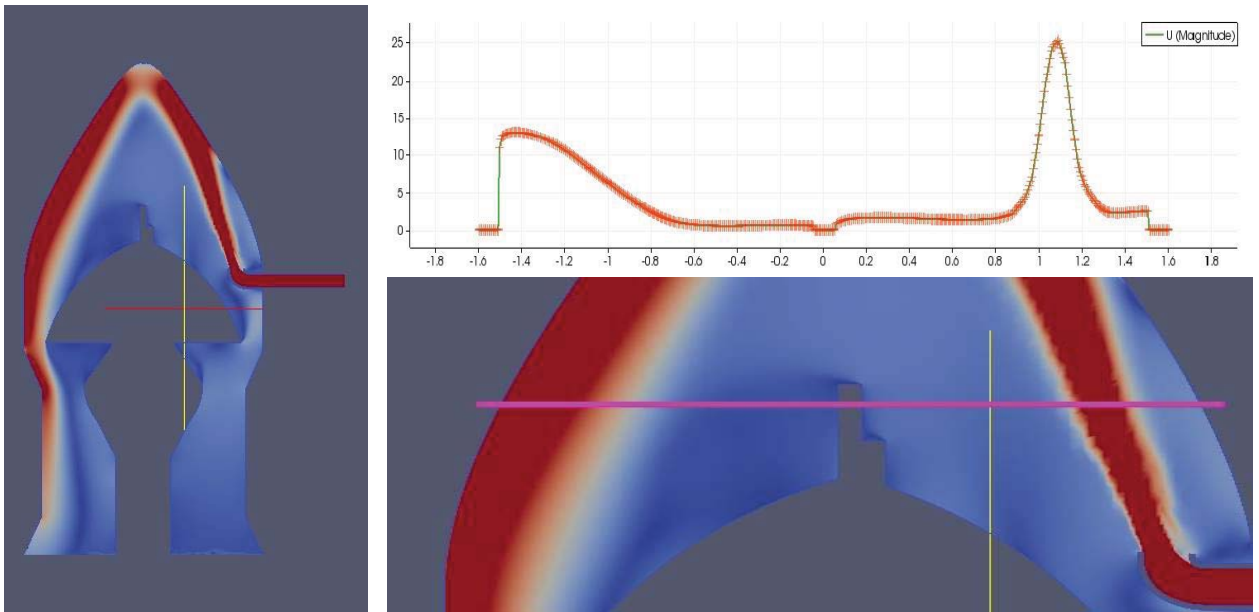
Figure 32: 3D Linear Interpolation Line Plot
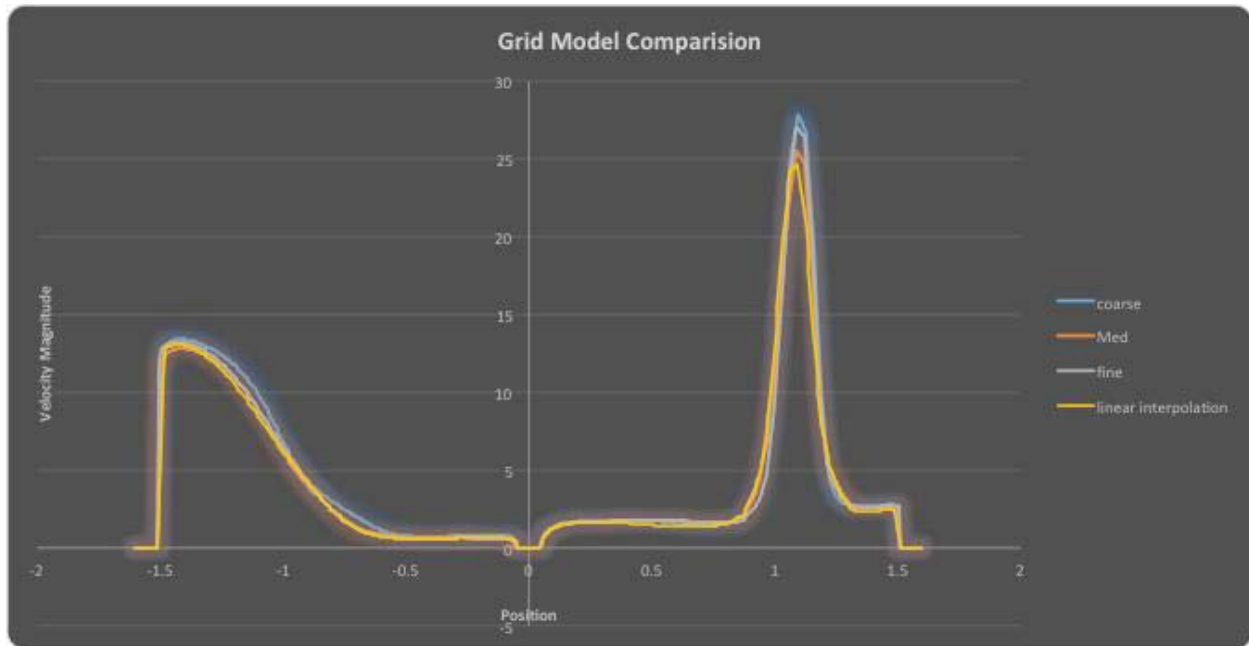
Figure 33: 3D Linear Interpolation Line Plot vs. Fine, Medium, and Coarse Solutions

By comparing the interpolation schemes in one, two, and three dimensions and investigating the options that are readily available in Matlab, it is recommended that the "linear" option be used when comparing the error or uncertainty due to the grid. The matlab code used for this interpolation is provided in Appendix A: InterpOPENFOAM.m.

There is another issue with interpolation. Which direction to use?

1. Interpolate Coarse and Medium Mesh -> Fine

φ_3
coarse

φ_2
medium

φ_1
fine
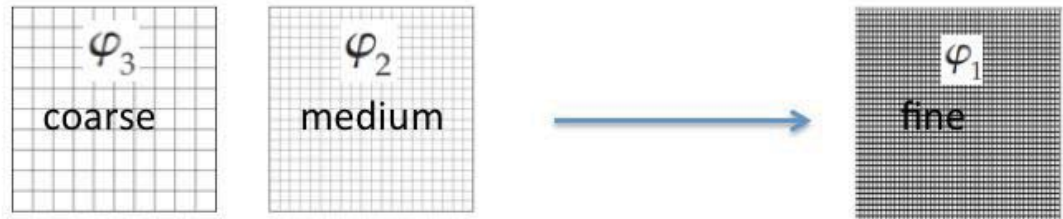
2. Interpolate Medium and Fine Mesh -> Coarse

φ_1
fine

φ_2
medium

φ_3
coarse
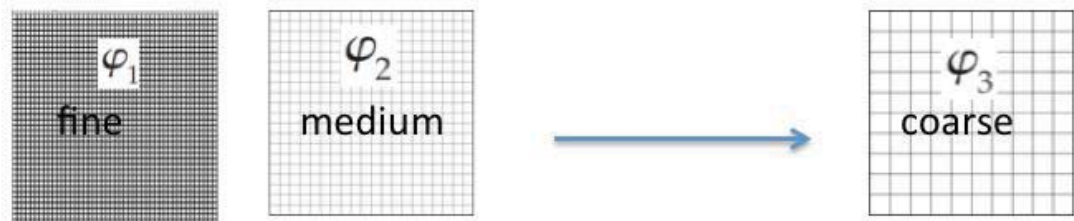
Figure 34: Interpolation Direction

The paper provided by Roach and Knupp suggests, "There is no value in producing two grids with the fine and coarse spacings. Only the fine grid need to be generated, and the course grid formed by using every other point" [38]. The paper uses the fine grid extrapolated solution. This is equivalent to interpolating via option1, all grids get interpolated onto the fine grid and the extrapolated solution is about the fine grid. The interpolation direction was investigated by further comparing the schemes and Richardson's extrapolation to the exact solution of parallel flow between parallel plates via the 5 step procedure. The results are shown in Figure 35, Figure 36 and Table 5, Table 6, Table 7.
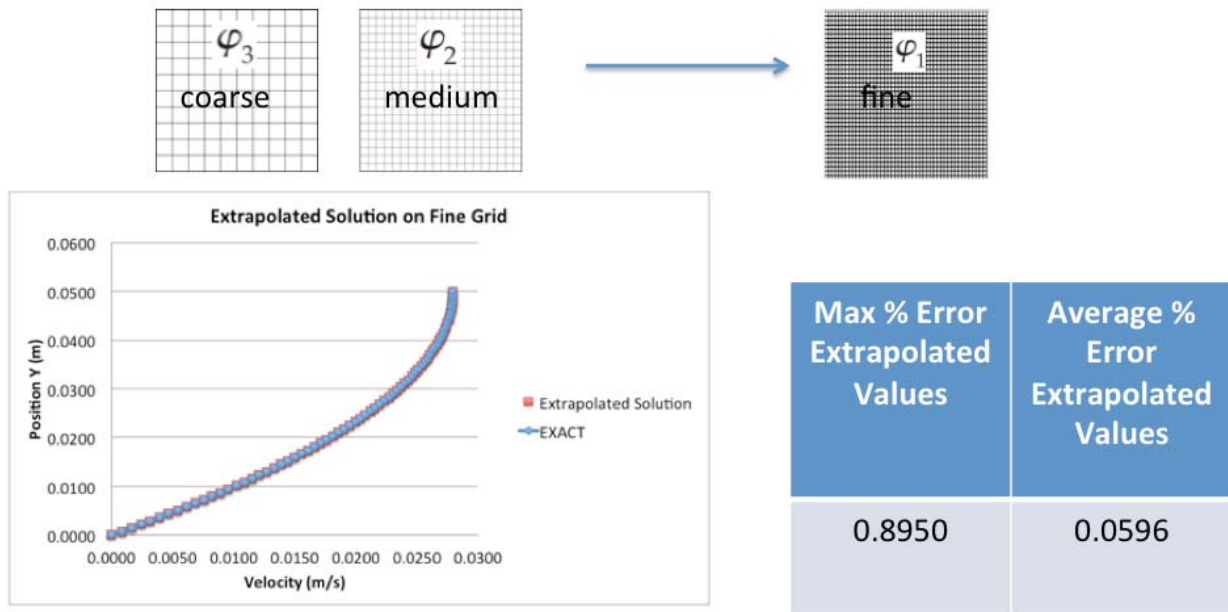
Figure 35: Interpolation Direction onto Fine Grid
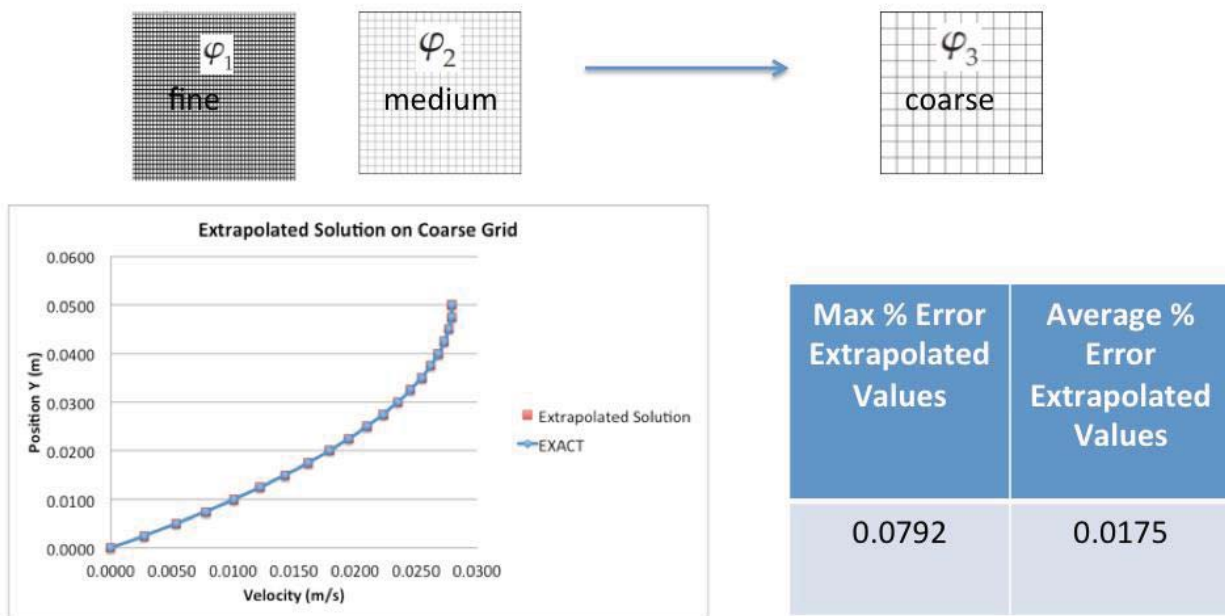
Figure 36: Interpolation Direction onto Coarse Grid

Table 5: Results Interpolating to Fine Grid for Flow Between Parallel Plates

<u>Interpolation to Fine Grid 'Linear'</u>

Average Observed Order, p	5.0441
Max % Error of Extrapolated Values	0.8950
Average % Error of Extrapolated Values	0.0596

<u>Interpolation to Fine Grid 'Cubic'</u>

Average Observed Order, p	5.4401
Max % Error of Extrapolated Values	1.1612
Average % Error of Extrapolated Values	0.0395

Table 6: Results Interpolating to Coarse Grid for Flow Between Parallel Plates

<u>Interpolation to Coarse Grid 'Linear'</u>

Average Observed Order, p	7.5200
Max % Error of Extrapolated Values	0.0792
Average % Error of Extrapolated Values	0.0175

<u>Interpolation to Coarse Grid 'Cubic'</u>

Average Observed Order, p	5.5725
Max % Error of Extrapolated Values	0.0176
Average % Error of Extrapolated Values	0.0028

Table 7: Comparison of Interpolation Results

Grids	Max % Error	Average % Error
(Coarse vs Exact)	0.1910	0.1265
(Medium vs Exact)	0.0969	0.0367
(Fine vs Exact)	0.0289	0.0121

1. Linearly Interpolated Coarse, Medium to Fine

	Max % Error	Average % Error
(Interpolated Coarse vs Exact)	1.9760	0.2528
(Interpolated Medium vs Exact)	0.6322	0.0679

2. Linearly Interpolated Medium, Fine to Coarse

	Max % Error	Average % Error
(Interpolated Medium vs Exact)	0.0728	0.0362
(Interpolated Fine vs Exact)	0.0787	0.0223

Extrapolated

	Max % Error	Average % Error
1. Linear Interpolation Coarse and Medium to Fine (Extrapolated vs Exact)	0.8950	0.0596
2. Linear Interpolation Medium and Fine to Coarse (Extrapolated vs Exact)	0.0792	0.0175

It can be shown that interpolating the fine and medium grids onto the coarse grid and performing Richardson's Extrapolation provides better results. The average error between the extrapolated values and exact solution is reduced by factor of 3.4 for the 'linear' method and factor of 14 for the 'cubic' method. However, there is an underlying advantage to interpolating to the finest grid because the analyst has more data available. Also, by interpolating from the fine and

medium grids onto the coarse grid to perform Richardson's Extrapolation, a higher observed order of convergence is accomplished and a lower error in the extrapolated solution.

5.2 Feasibility of Using Richardson's Extrapolation for Entire Computational Domain

To use Richardson's extrapolation, the solution must be monotonically increasing or decreasing with grid refinement. That is to say the solution is in the asymptotic range. By plotting some data in the computational domain for three different grids, it can be shown that increasing the grid does not provide a monotonic solution at every point. Please see Figure 37. The two_medgrid would have to be between the one_coarsegrid and three_fine grid at every location to use Richardson's extrapolation. Using this method to estimate uncertainty for the entire computational domain is inappropriate due to the grid refinement not being monotonically increasing or decreasing at every location.

Another way to present this argument is to look at the R_k values as described in equation 11. For all R_k values greater than 1, the solution is divergent. For all R_k values between 0 and 1, the solution is monotonic and Richardson's extrapolation may be used. For all R_k values less than 0, oscillatory convergence is prevalent. Figure 38, Figure 39, and Figure 40 are plots of the R_k values for each configuration being analyzed. For the plots all values in blue represent cells with oscillatory convergence. All values shown in red represent the cells with divergence. All other colors represent cells with monotonic convergence. Therefore, it is inappropriate to use Richardson's extrapolation or the ASME 5 step procedure for estimating uncertainties associated with the numeric's.
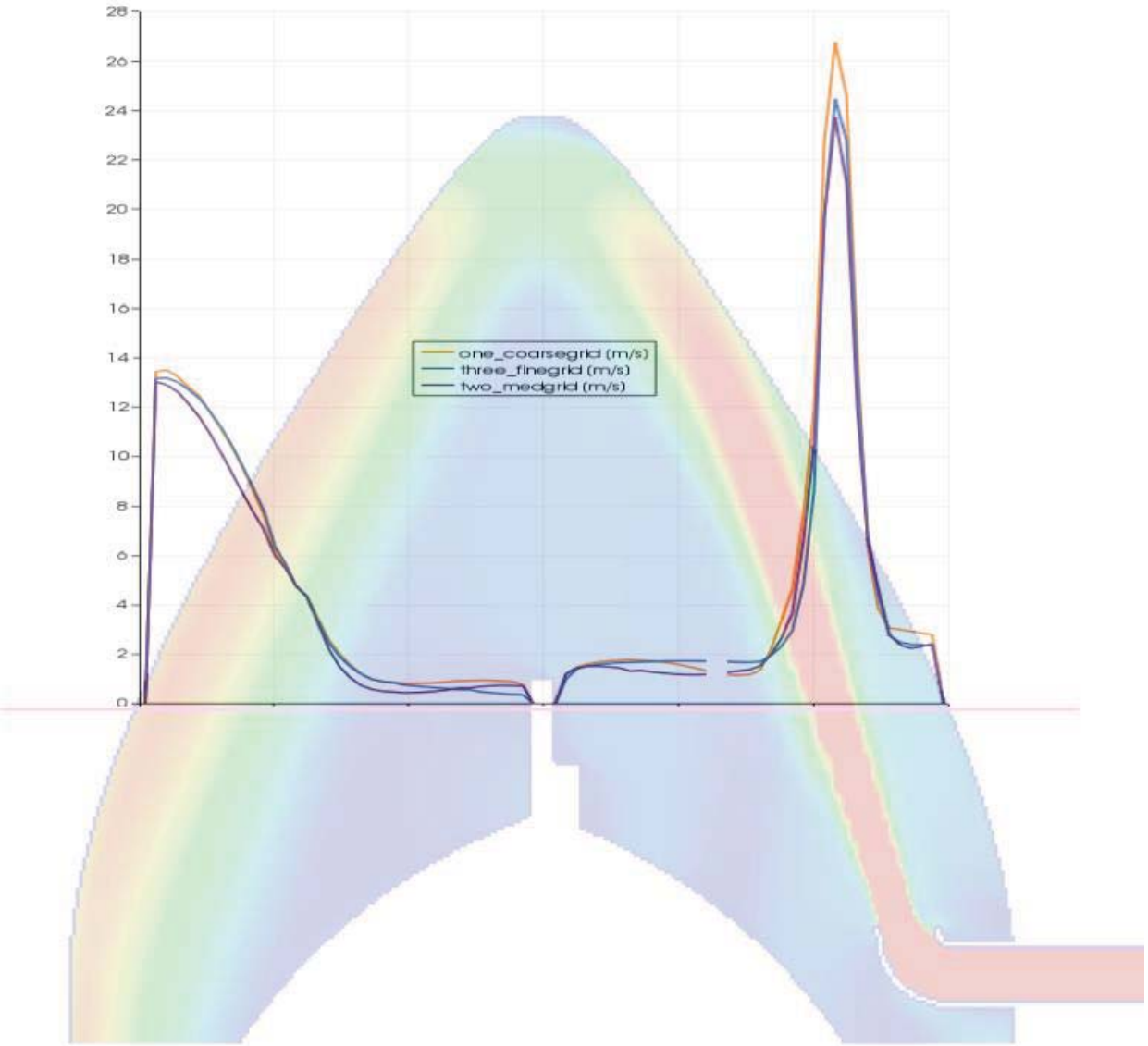
Figure 37: Fine, Medium, and Coarse Solutions
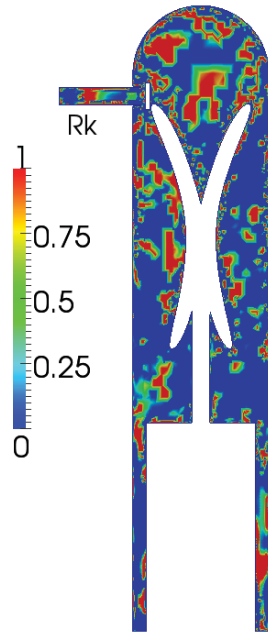
Figure 38: 075 Configuration R_k Plot
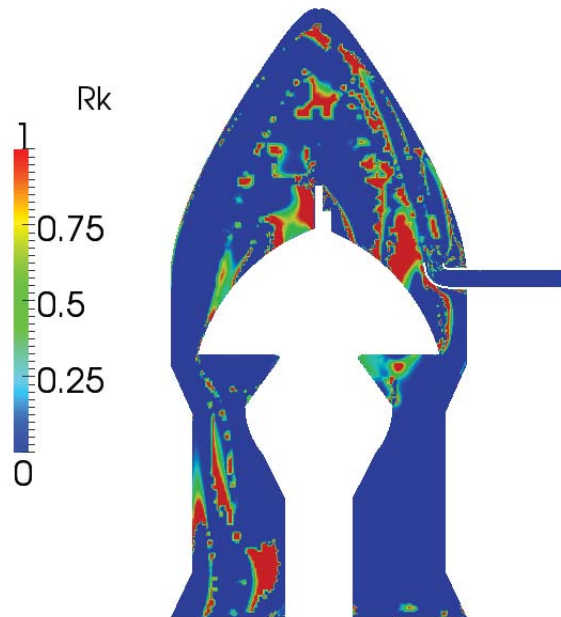
Figure 39: 35 Configuration R_k Plot
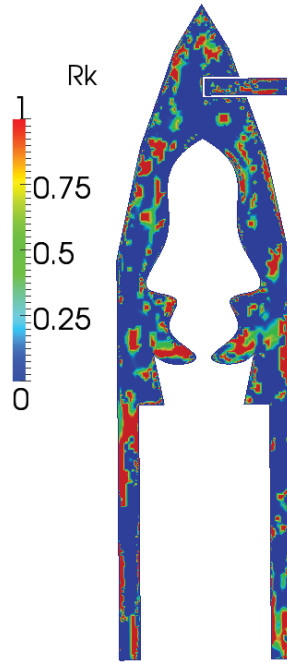
Figure 40: 55 Configuration R_k Plot

It further should be discussed that the solution is not monotonic or divergent. The solution's residuals showed fully converged and well behaved for all models. The solution is actually oscillatory at all points but is being miss represented as monotonic or divergent depending on where the points were taken on the oscillatory curve. This is still an area of discussion [15], [39], [40], [41]. It is believed that it is extremely difficult to achieve the asymptotic range for complex problem [15] or that this is due to non-uniform mesh refinement [42]. The proposed methodology is to treat the grid as a separate oscillatory parameter as shown in the next section.

5.3   Proposed CFD Uncertainty Method Compared to Exact Solution –Laminar Flow Between Parallel, Stationary Plates

The uncertainty can be calculated by expanding equation 13 for pressure, density, numerical (grid), and solver.

$$u_{val} = k * \left(\left(\left(\frac{\partial V}{\partial pressure} \right)^2 B_{pressure}^2 \right) + \left(\left(\frac{\partial V}{\partial rho} \right)^2 B_{rho}^2 \right) + \right.$$
$$\left. \left(\left(\frac{\partial V}{\partial num} \right)^2 B_{num}^2 \right) + \left(\left(\frac{\partial V}{\partial solver} \right)^2 B_{solver}^2 \right) + + \left(\left(\frac{\partial V}{\partial velocity} \right)^2 B_{velocity}^2 \right) \right)^{1/2}$$

(31)

The proposed method is to calculate the uncertainty as an oscillatory input parameter and multiply by the appropriate Student-T k-factor.

For Numerical, three grids were used and the t value of 2.92.

$$u_{val} = 2.92 * \left(\left(\left(\frac{\partial V}{\partial num} \right)^2 B_{num}^2 \right) \right)^{1/2}$$

(32)

$$u_{val} = 2.92 * \left| \frac{1}{2} (S_U - S_L) \right|$$

(33)

The centerline velocity was chosen as an example to plot, however at all points the uncertainty bands always encompass the exact solution.
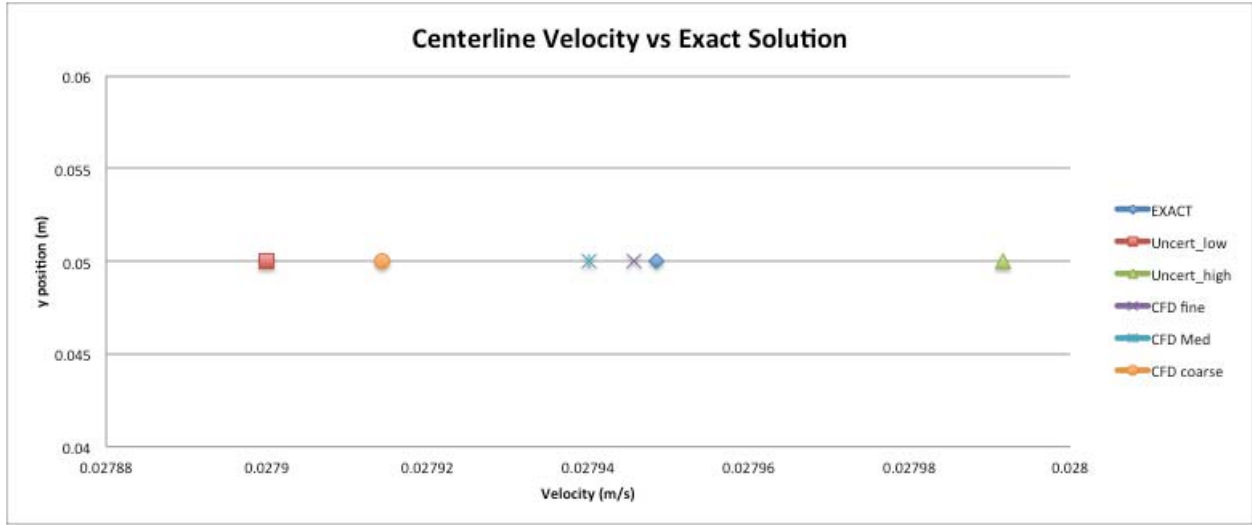
Figure 41: Exact Solution vs. CFD with Uncertainty (Centerline Velocity) - Grid

If there is also a variation in the inlet velocity due to a tolerance or known bias, run the model at the low and high limits and use a new t-value of 2.132, which corresponds to five cases. The five cases would be three for grids and two for flow rates. A five percent variation in inlet velocity was chosen for this example.

$$u_{val} = 2.132 * \left(\left(\left(\frac{\partial V}{\partial num} \right)^2 B_{num}^2 \right) + \left(\left(\frac{\partial V}{\partial velocity} \right)^2 B_{velocity}^2 \right) \right)^{1/2} \tag{34}$$

$$u_{val} = 2.132 * \left| \frac{1}{2} (S_U - S_L) \right| \tag{35}$$
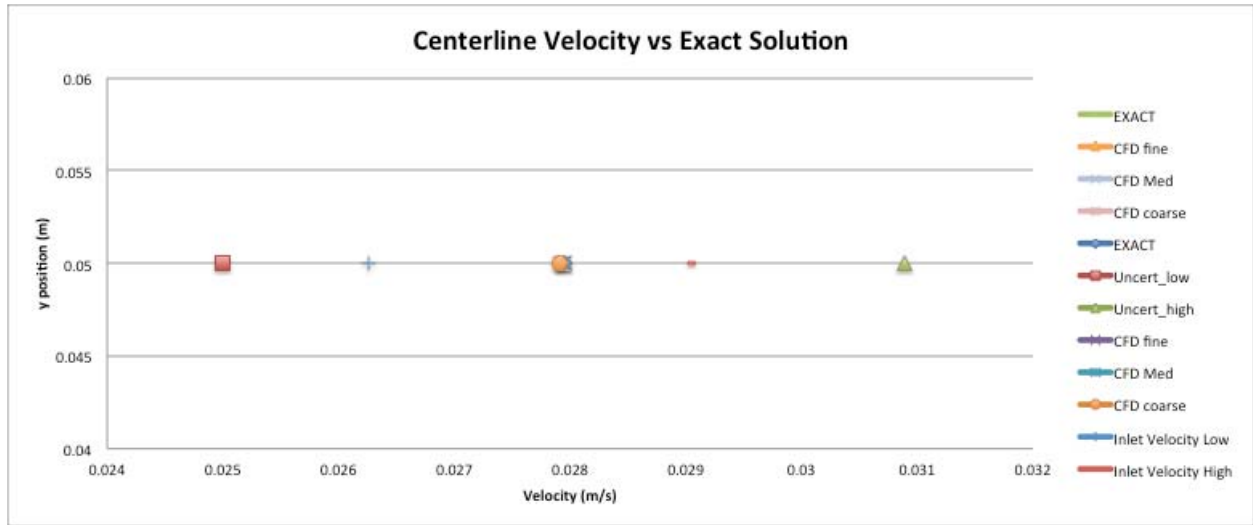
Figure 42: Exact Solution vs. CFD with Uncertainty (Centerline Velocity) – Grid and Inlet Velocity

Also to include the outlet pressure boundary condition, run the model at the low and high known bias or tolerances and use a new t-value of 1.943, which corresponds to seven cases. The seven cases would be three for grid, two for flow rate, and two for pressure outlet boundary condition.

$$
u_{val} = 1.943 * \left(\left(\left(\frac{\partial V}{\partial num} \right)^2 B_{num}^2 \right) + \left(\left(\frac{\partial V}{\partial velocity} \right)^2 B_{velocity}^2 \right) + \left(\left(\frac{\partial V}{\partial pressure} \right)^2 B_{pressure}^2 \right) \right)^{1/2}
$$

(36)

$$
u_{val} = 1.943 * \left| \frac{1}{2} (S_U - S_L) \right|
$$

(37)

60
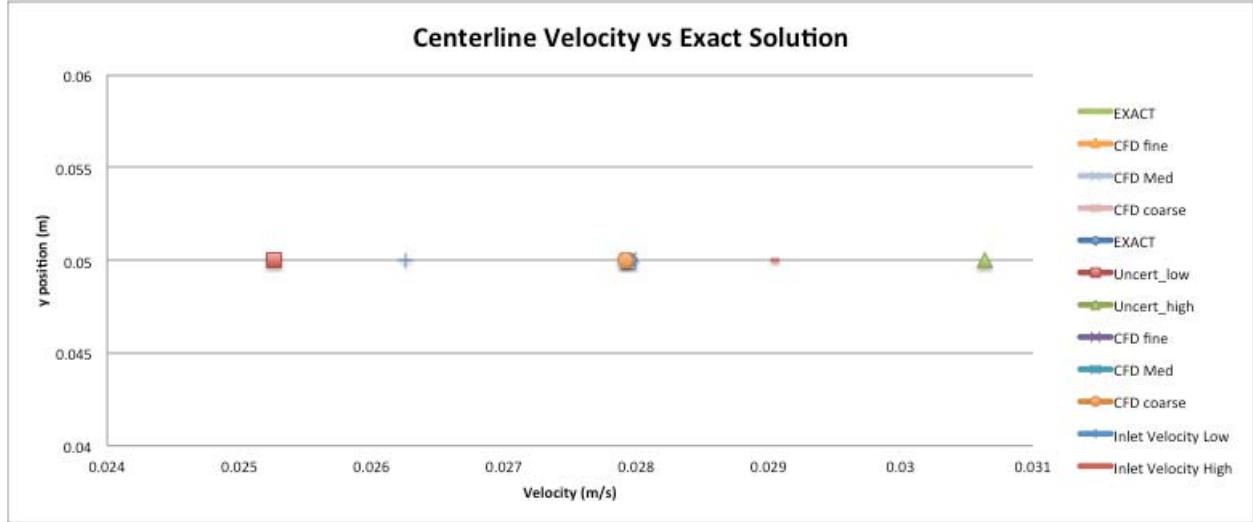
Figure 43: Exact Solution vs. CFD with Uncertainty (Centerline Velocity) – Grid, Inlet Velocity, and Outlet Pressure

To account for the variation in fluid properties, the kinematic viscosity for air between 0 and 100 degrees Celsius is 13.6×10^{-6} to 23.06×10^{-6}. The model was run at these limits to account for the possible variation in fluid properties and a new value of t= 1.86 was chosen, which corresponds to the nine cases.

$$
u_{val} = 1.86 * \left(\left(\left(\frac{\partial V}{\partial num} \right)^2 B_{num}^2 \right) + \left(\left(\frac{\partial V}{\partial velocity} \right)^2 B_{velocity}^2 \right) + \right.
$$
$$
\left. \left(\left(\frac{\partial V}{\partial pressure} \right)^2 B_{pressure}^2 \right) + + \left(\left(\frac{\partial V}{\partial rho} \right)^2 B_{rho}^2 \right) \right)^{1/2}
$$
(38)

$$
u_{val} = 1.86 * \left| \frac{1}{2} (S_U - S_L) \right|
$$
(39)
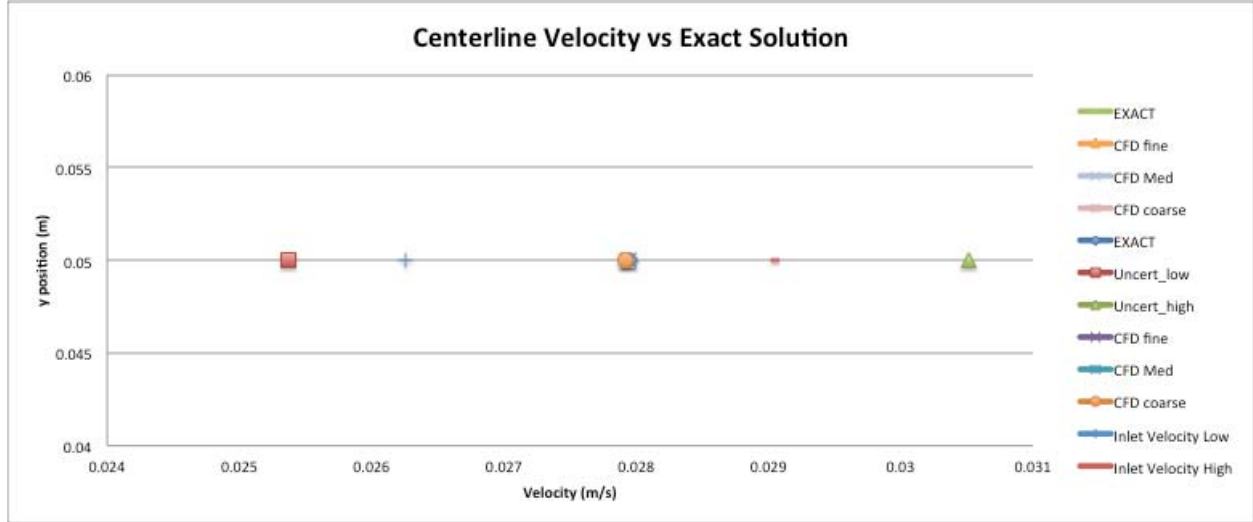
Figure 44: Exact Solution vs. CFD with Uncertainty (Centerline Velocity) – Grid, Inlet Velocity, Outlet Pressure, and Density

Fluent has been used to calculate the results above; we also consider the solver as an input to the model. To account for the variation in the solver, the model was run in OpenFOAM. The t value was updated to 1.833 because the numbers of cases are ten.

$$u_{val} = 1.833 * \left(\left(\left(\frac{\partial V}{\partial num} \right)^2 B_{num}^2 \right) + \left(\left(\frac{\partial V}{\partial velocity} \right)^2 B_{velocity}^2 \right) + \right.$$
$$\left. \left(\left(\frac{\partial V}{\partial pressure} \right)^2 B_{pressure}^2 \right) + + \left(\left(\frac{\partial V}{\partial rho} \right)^2 B_{rho}^2 \right) + \left(\frac{\partial V}{\partial solver} \right)^2 B_{solver}^2 \right)^{1/2} \quad (40)$$

$$u_{val} = 1.833 * \left| \frac{1}{2} (S_U - S_L) \right| \quad (41)$$
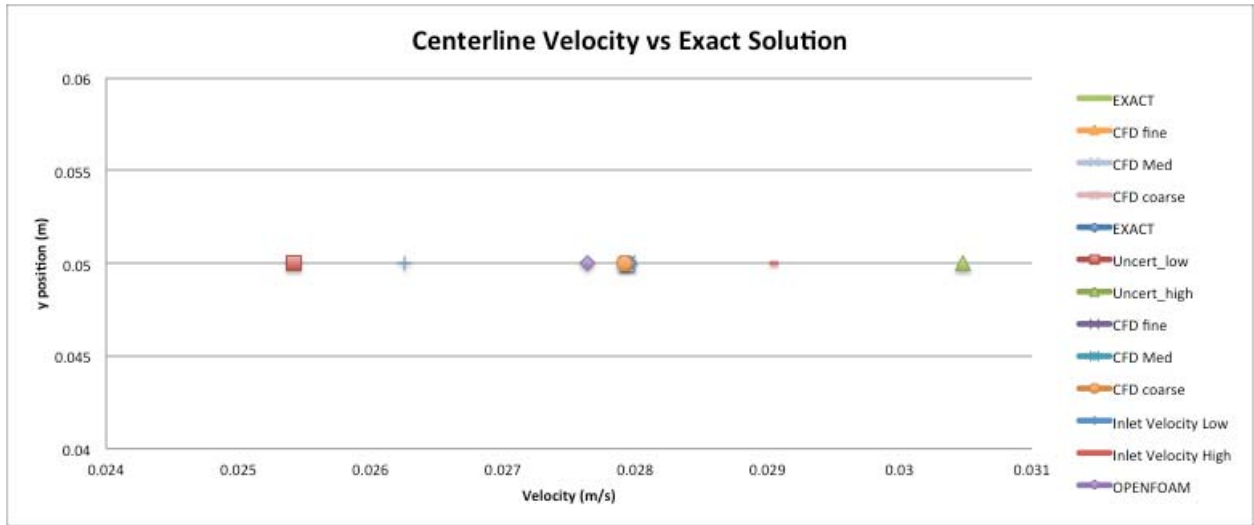
Figure 45: Exact Solution vs. CFD with Uncertainty (Centerline Velocity) – Grid, Inlet Velocity, Outlet Pressure, Density, and Solver

Figure 46 is a plot of all the CFD cases, uncertainty, and an exact comparison.
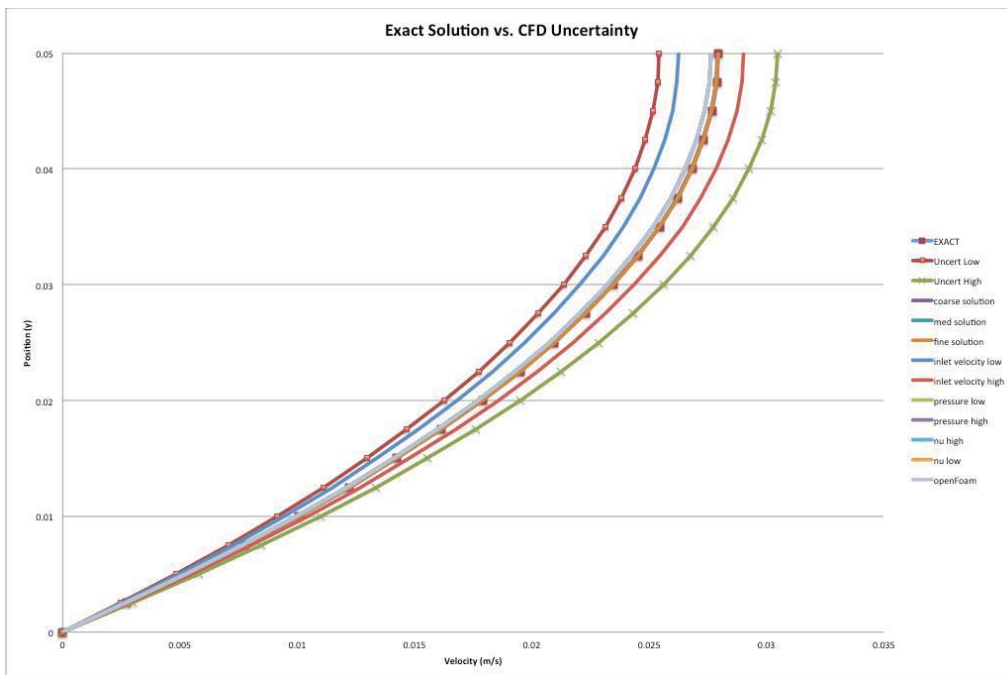
Figure 46: Exact Solution vs. CFD with Uncertainty (Parallel Plates – Half of Domain) – Grid, Inlet Velocity, Outlet Pressure, Density, and Solver

It can be concluded that treating all inputs to a CFD model as oscillatory uncertainty parameters coupled with the Student-T distribution can supply an uncertainty estimate that encompasses the exact solution for the case considered above (fully developed, laminar, flow between stationary parallel plates). To summarize the approach and general idea, there is a standard[2] for calculating verification and validation of CFD using a combined numerical and experimental data. The approach described above is a way to estimate the uncertainty of a model if test data is not available. An analyst should make use of all test data that is available or able to be funded and use the ASME standard. However, if test data is missing or not attainable, the method described makes assumptions that each CFD solution belongs to an underlying Student-T distribution and a corresponding uncertainty can be estimated for a selected confidence interval.

This method can be contrasted with traditional uncertainty analysis of the exact solution. The exact solution for the velocity profile between parallel, stationary plates is shown in equation (29).

Traditional uncertainty analysis of this equation can be calculated for the parameters considered (5 % uncertainty in $\frac{\partial P}{\partial x}$ and μ). This should provide a nearly identical result as compared to the numerical method described above.

$$u_u = \left(\left(\left(\frac{\partial u}{\partial \mu} \right)^2 B_\mu^2 \right) + \left(\left(\frac{\partial u}{\partial \frac{\partial P}{\partial x}} \right)^2 B_{\frac{\partial P}{\partial x}}^2 \right) \right)^{1/2} \tag{42}$$

Calculating the partial derivatives:

$$\frac{\partial u}{\partial \mu} = -\frac{\eth P}{\eth x} * y * (y - a)/(2\mu)$$

(43)

$$\frac{\partial u}{\partial \frac{\eth P}{\eth x}} = \frac{a^2}{2\mu}\left[\left(\frac{y}{a}\right)^2 - \left(\frac{y}{a}\right)\right]$$

(44)

Plugging the partial derivatives into equation (42). The equation becomes the exact uncertainty

for a 5% variation in $\frac{\eth P}{\eth x}$ and μ as was the numerical method described above.

$$u_u = \left(\left(\left(-\frac{\eth P}{\eth x} * y * (y - a)/(2\mu)\right)^2 B_\mu^2\right) \right.$$

$$\left. + \left(\left(\frac{a^2}{2\mu}\left[\left(\frac{y}{a}\right)^2 - \left(\frac{y}{a}\right)\right]\right)^2 B_{\frac{\eth P}{\eth x}}^2\right) \right)^{1/2}$$

(45)

Figure 47 and Figure 48 was created to show the uncertainty and the exact solution with uncertainty

bands.
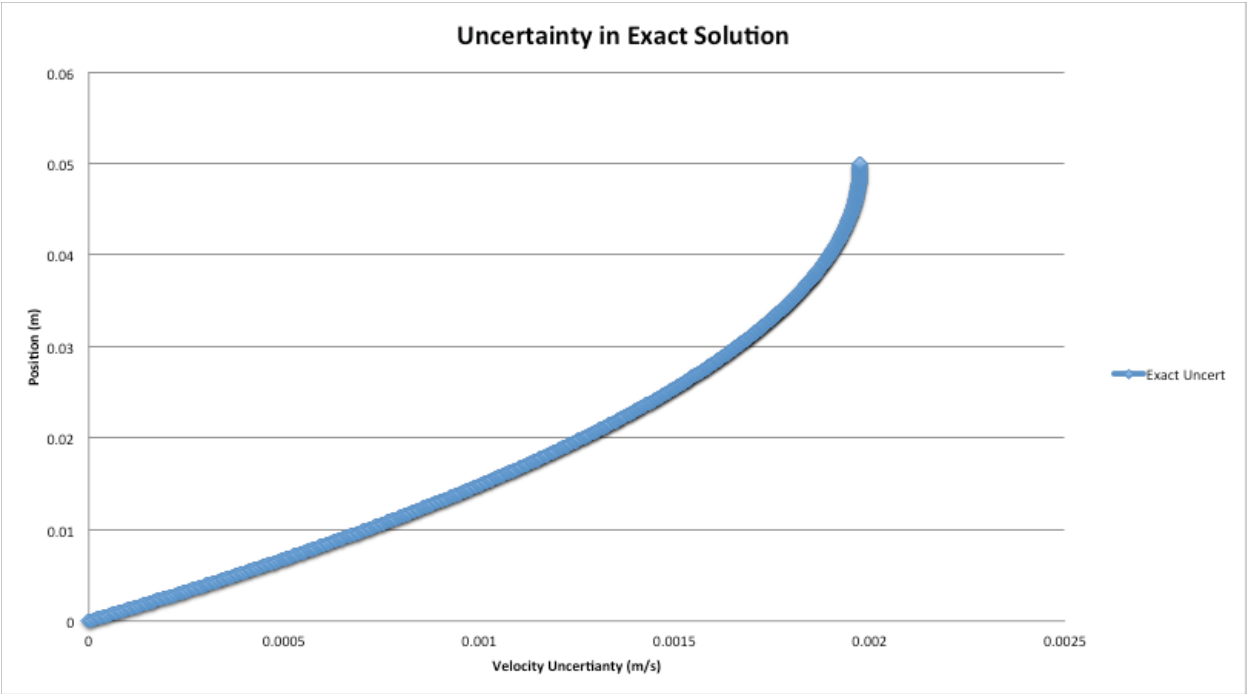
Figure 47: Exact Solution Uncertainty
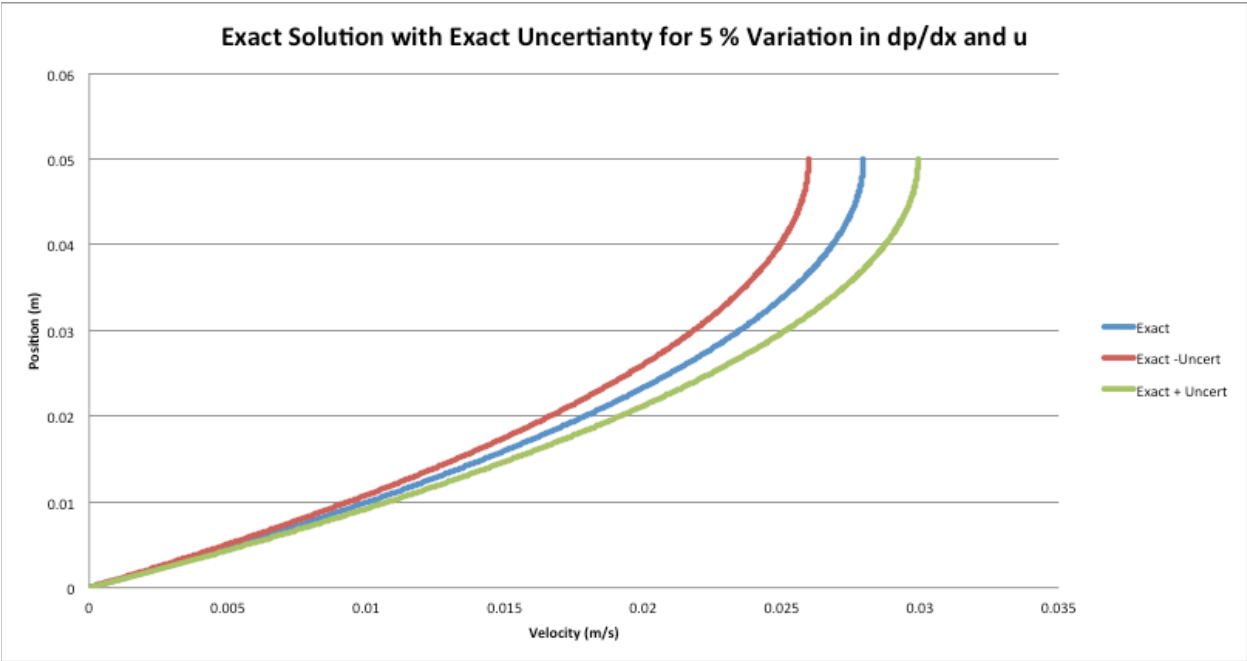
Figure 48: Exact Solution Uncertainty

To contrast the methods, the numerically derived uncertainty in equations 26, 27 and Figure 43 are compared to the traditional exact derivation of uncertainty as shown in Figure 50.
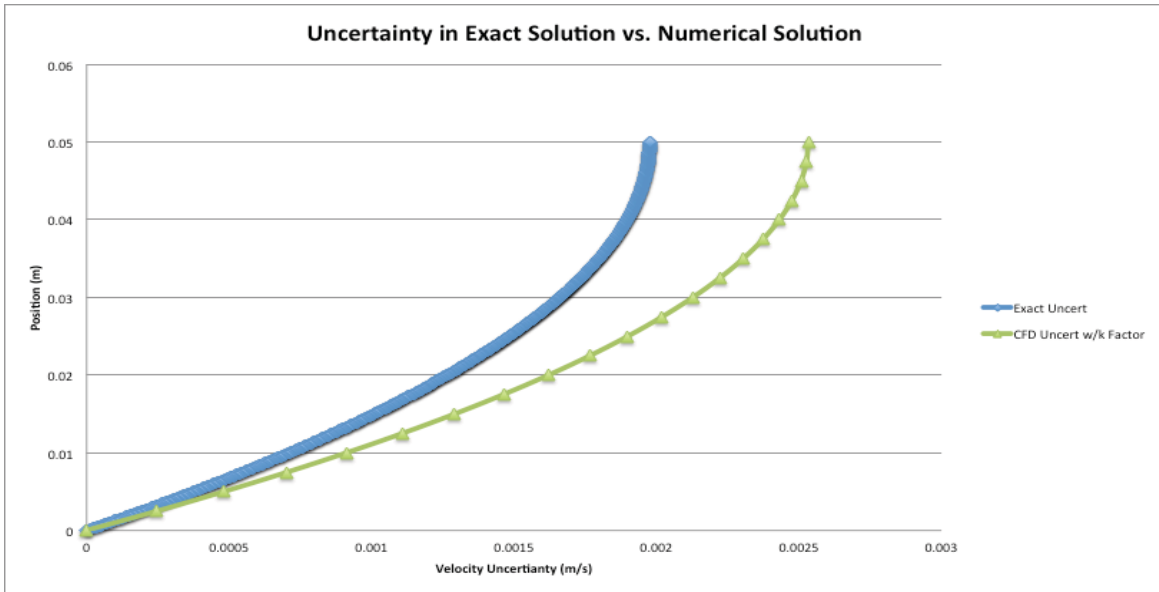
Figure 49: Exact Solution Uncertainty vs. Numerically Derived Uncertainty
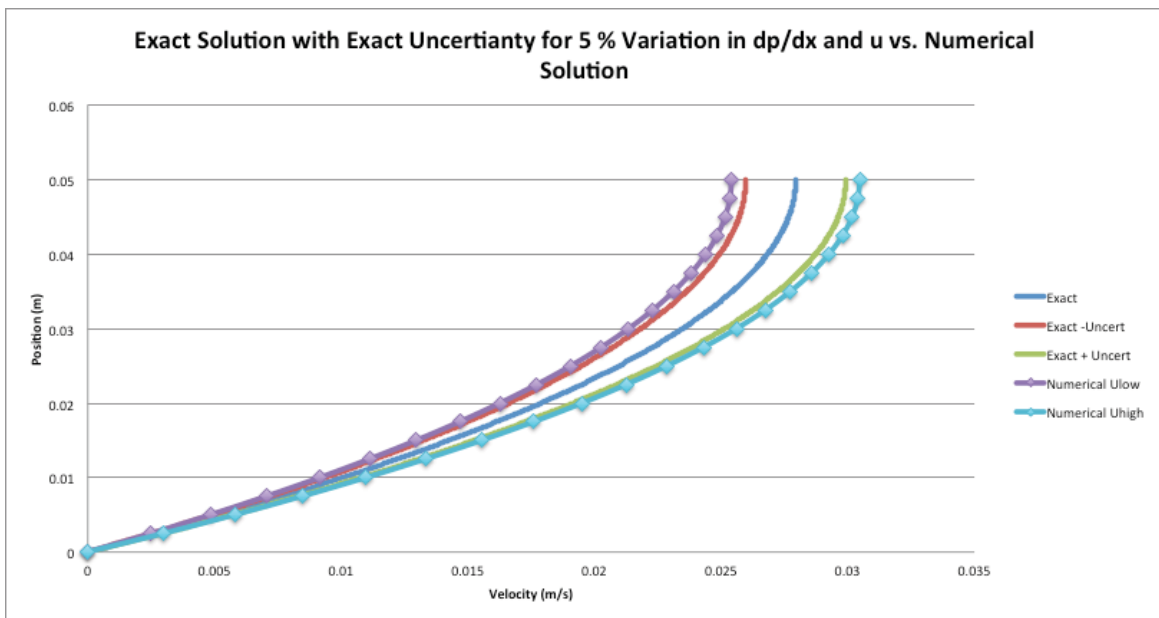
Figure 50: Exact Solution with Uncertainty vs. Numerically Derived Uncertainty

It can be concluded further that the numerically derived uncertainty method using the student t value of k=1.833 is conservative and over-predicts the uncertainty. Also important to note is that a k-value of 1.43 matches the two methods as shown in Figure 52. K=1.43 is an interesting result and there may be some correlation between this number and other distributions.
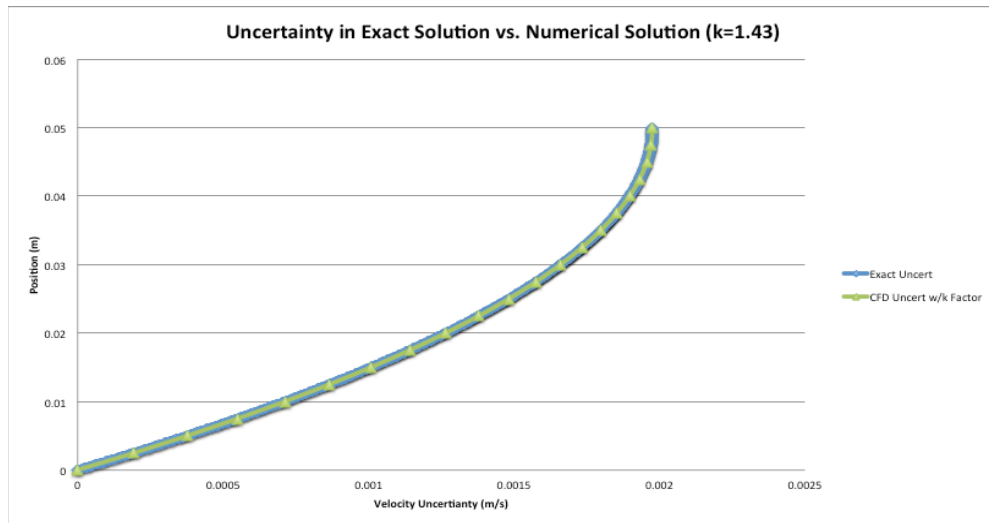
Figure 51: Exact Solution Uncertainty vs. Numerically Derived Uncertainty (with k=1.43)
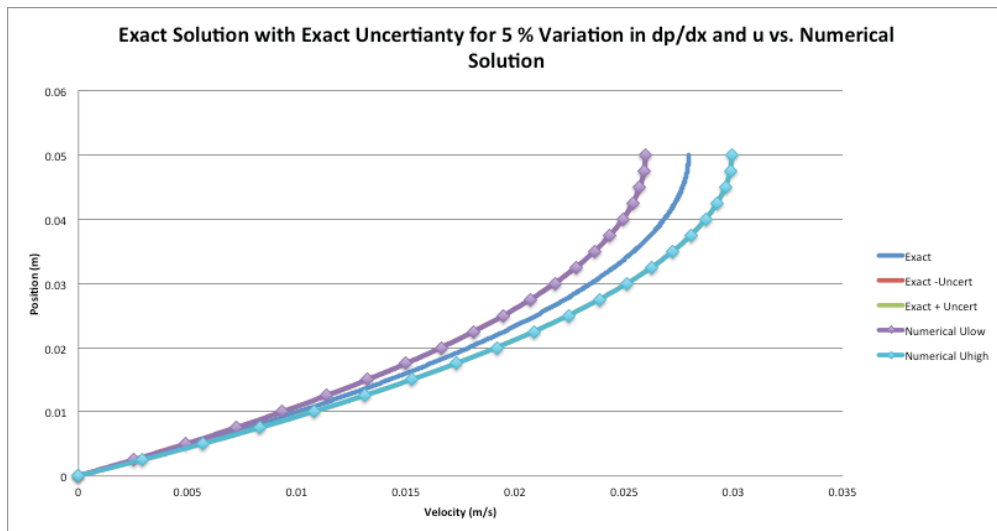
Figure 52: Exact Solution Uncertainty vs. Numerically Derived Uncertainty (with k=1.43)

5.4 Proposed CFD Uncertainty Method Applied to Heat Transfer over a Flat Plate

The data in this section was published by the author in DFD13-2013-00087 [43]. Cornell University has posted a FLUENT example problem of forced convection over a flat plate [44]. This example problem is a good demonstration of traditional uncertainty analysis with experimental data verses the proposed methodology using CFD.

5.4.1 Correlation Uncertainty Calculation

Different correlations have been derived for this simple problem based off experimental data. They are generally in the form of equation (46).

$$\frac{hL}{k} = c\left(\frac{\rho V L}{\mu}\right)^{4/5} \tag{46}$$

The uncertainty in the heat transfer coefficient can be calculated as follows.

$$h = c\left(\frac{\rho V L}{\mu}\right)^{4/5}\frac{k}{L} \tag{47}$$

Differentiating with respect to each of the variables:

$$\frac{dh}{dV} = \frac{4ck\rho}{5\mu\left(\frac{LV\rho}{\mu}\right)^{\frac{1}{5}}} \tag{48}$$

$$\frac{dh}{d\rho} = \frac{4ckV}{5\mu\left(\frac{LV\rho}{\mu}\right)^{\frac{1}{5}}} \tag{49}$$

$$\frac{dh}{dk} = \frac{c}{L}\left(\frac{LV\rho}{\mu}\right)^{4/5}$$

(50)

$$\frac{dh}{d\mu} = -\frac{4CVk\rho}{5\mu^2 \left(\frac{LV\rho}{\mu}\right)^{\frac{1}{5}}}$$

(51)

$$\frac{dh}{dL} = \frac{4CVk\rho}{5L\mu \left(\frac{LV\rho}{\mu}\right)^{\frac{1}{5}}} - \frac{Ck\left(\frac{LV\rho}{\mu}\right)^{4/5}}{L^2}$$

(52)

$$\frac{dh}{dC} = \frac{k}{L}\left(\frac{\rho VL}{\mu}\right)^{4/5}$$

(53)

Expanding the uncertainty equation

$$U_h = \left(\left(\left(\frac{\partial h}{\partial V}\right)^2 B_V^2\right) + \left(\left(\frac{\partial h}{\partial \rho}\right)^2 B_\rho^2\right) + \left(\left(\frac{\partial h}{\partial k}\right)^2 B_k^2\right) + \left(\left(\frac{\partial h}{\partial \mu}\right)^2 B_\mu^2\right)\right.$$

$$+ \left(\left(\frac{\partial h}{\partial L}\right)^2 B_L^2\right) + \left(\left(\frac{\partial h}{\partial C}\right)^2 P_C^2\right) + 2\left(\frac{\partial h}{\partial \rho}\right)\left(\frac{\partial h}{\partial k}\right) B_\rho B_k$$

(54)

$$\left. + 2\left(\frac{\partial h}{\partial \rho}\right)\left(\frac{\partial h}{\partial \mu}\right) B_\rho B_\mu + 2\left(\frac{\partial h}{\partial k}\right)\left(\frac{\partial h}{\partial \mu}\right) B_k B_\mu\right)^{1/2}$$

Plugging in the partial derivatives

$$U_h = \left(\left(\left(\frac{4ck\rho}{5\mu\left(\frac{LV\rho}{\mu}\right)^{\frac{1}{5}}}\right)^2 B_V^2\right) + \left(\left(\frac{4ckV}{5\mu\left(\frac{LV\rho}{\mu}\right)^{\frac{1}{5}}}\right)^2 B_\rho^2\right) + \left(\left(\frac{c}{L}\left(\frac{LV\rho}{\mu}\right)^{4/5}\right)^2 B_k^2\right)\right.$$

$$+ \left(\left(\frac{\partial h}{\partial \mu}\right)^2 B_\mu^2\right) + \left(\left(\frac{4CVk\rho}{5L\mu\left(\frac{LV\rho}{\mu}\right)^{\frac{1}{5}}} - \frac{Ck\left(\frac{LV\rho}{\mu}\right)^{4/5}}{L^2}\right)^2 B_L^2\right)$$

$$+ \left(\left(\frac{k}{L}\left(\frac{\rho VL}{\mu}\right)^{4/5}\right)^2 P_C^2\right) + 2\left(\frac{4ckV}{5\mu\left(\frac{LV\rho}{\mu}\right)^{\frac{1}{5}}}\right)\left(\frac{c}{L}\left(\frac{LV\rho}{\mu}\right)^{4/5}\right)B_\rho B_k \qquad (\,55\,)$$

$$+ \; 2\left(\frac{4ckV}{5\mu\left(\frac{LV\rho}{\mu}\right)^{\frac{1}{5}}}\right)\left(-\frac{4CVk\rho}{5\mu^2\left(\frac{LV\rho}{\mu}\right)^{\frac{1}{5}}}\right)B_\rho B_\mu$$

$$+ \; \left. 2\left(\frac{c}{L}\left(\frac{LV\rho}{\mu}\right)^{4/5}\right)\left(-\frac{4CVk\rho}{5\mu^2\left(\frac{LV\rho}{\mu}\right)^{\frac{1}{5}}}\right)B_k B_\mu\right)^{1/2}$$

Using the variables and bias errors in Table 8 and Table 9 to numerically evaluating the expression in equation (55) results in the following. The c values are shown below for a couple of correlations.

Table 8: c Value

c	
Seban & Doughty	0.0236
Jakob	0.024
Sugawara	0.023
Fundamentals of Heat and Mass Transfer	0.0296
c middle	0.0263
c uncert (random)	0.0033

Table 9: Variables and Bias Errors

Variable	Bias
Velocity, V	3%
Density, rho	3%
Thermal Conductivity, k	3%
Viscosity, mu	3%
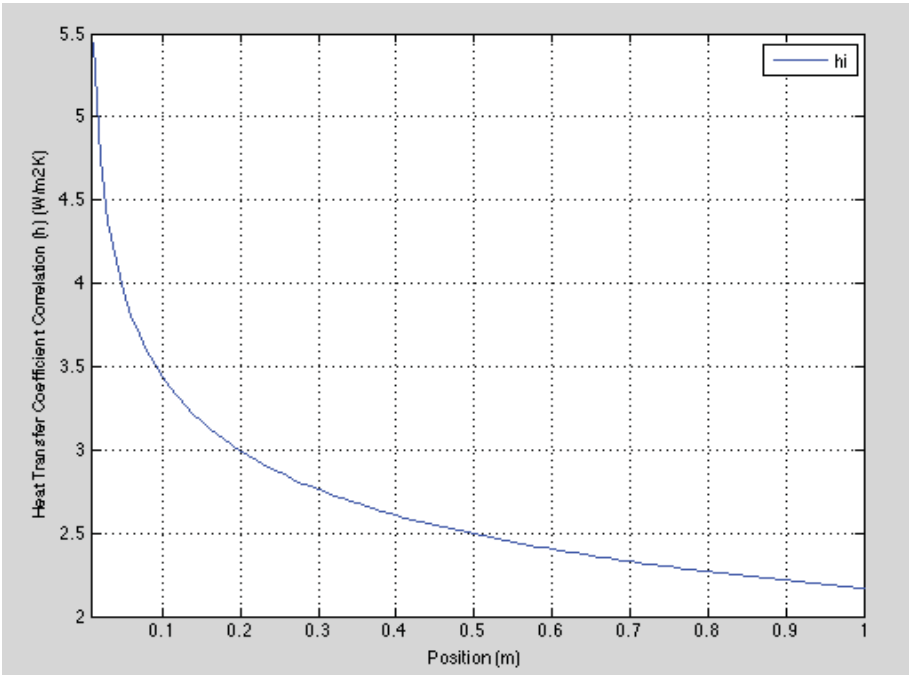
Figure 53: Heat Transfer Coefficient - Correlation
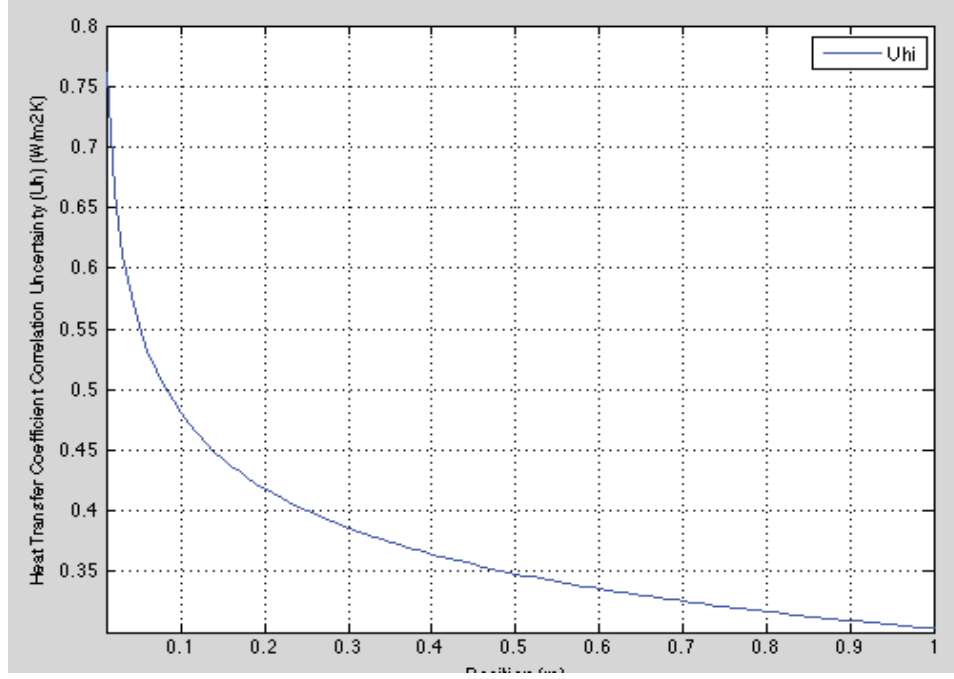
Figure 54: Heat Transfer Coefficient and Uncertainty - Correlation

5.4.2 CFD Uncertainty Calculation

To calculate the uncertainty of heat transfer over a flat plate using CFD, the proposed methodology was used as follows.

$$
\begin{aligned}
U_h = \Bigg(& \left(\left(\frac{\partial h}{\partial V}\right)^2 B_V^2 \right) + \left(\left(\frac{\partial h}{\partial \rho}\right)^2 B_\rho^2 \right) + \left(\left(\frac{\partial h}{\partial k}\right)^2 B_k^2 \right) + \left(\left(\frac{\partial h}{\partial \mu}\right)^2 B_\mu^2 \right) + \left(\left(\frac{\partial h}{\partial L}\right)^2 B_L^2 \right) \\
& + \left(\left(\frac{\partial h}{\partial C}\right)^2 P_C^2 \right) + 2 \left(\frac{\partial h}{\partial \rho}\right) \left(\frac{\partial h}{\partial k}\right) B_\rho B_k + 2 \left(\frac{\partial h}{\partial \rho}\right) \left(\frac{\partial h}{\partial \mu}\right) B_\rho B_\mu \\
& + 2 \left(\frac{\partial h}{\partial k}\right) \left(\frac{\partial h}{\partial \mu}\right) B_k B_\mu \Bigg)^{1/2}
\end{aligned}
\tag{56}
$$

73

$$u_{val} = 1.782 * \left| \frac{1}{2} (S_U - S_L) \right| \qquad\qquad (\ 57\ )$$

The following cases were run in FLUENT and shown in Table 10.

Table 10: CFD Cases – Flat Plate HTC

	CFD Uncertainty Cases
1	Coarse Grid
2	Medium Grid
3	Fine Grid
4	Velocity Low
5	Velocity High
6	Density Low
7	Density High
8	Thermal Conductivity High
9	Thermal Conductivity Low
10	Viscosity Low
11	Viscosity High
12	SA Turbulence Model
13	kwSST Turbulence Model
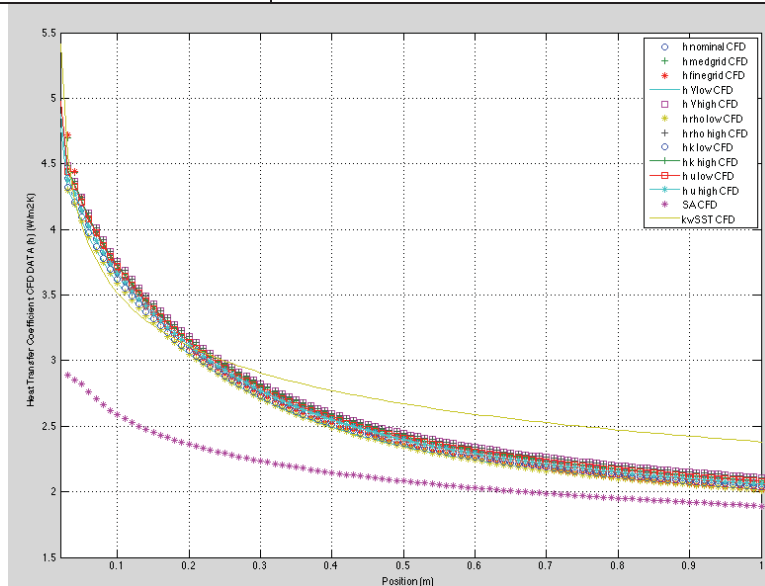
Figure 55: Heat Transfer Coefficient and Uncertainty – CFD Data

74
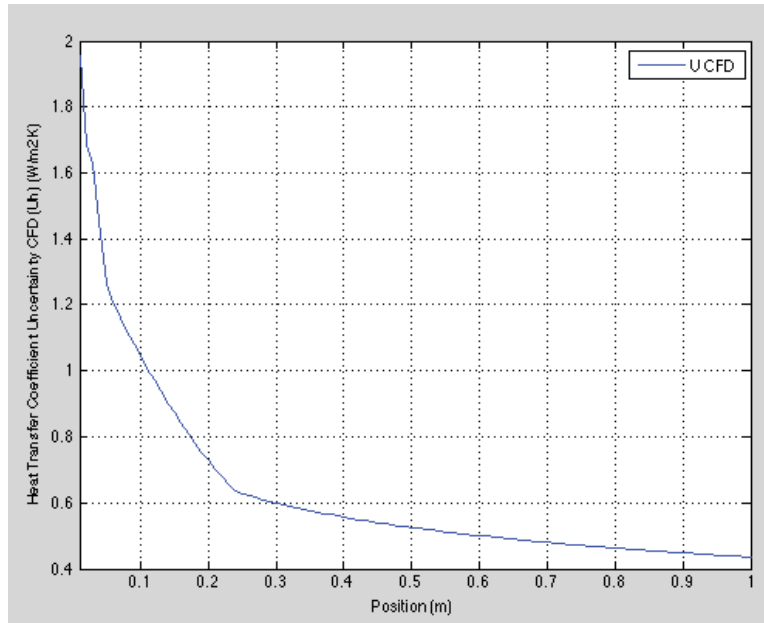
Figure 56: Heat Transfer Coefficient and Uncertainty – CFD Data

5.4.3   Comparison and Discussion

To compare the two methods, the results from both cases are plotted together.   It can be shown that the CFD uncertainty method is more conservative than the traditional uncertainty analysis using the correlation.
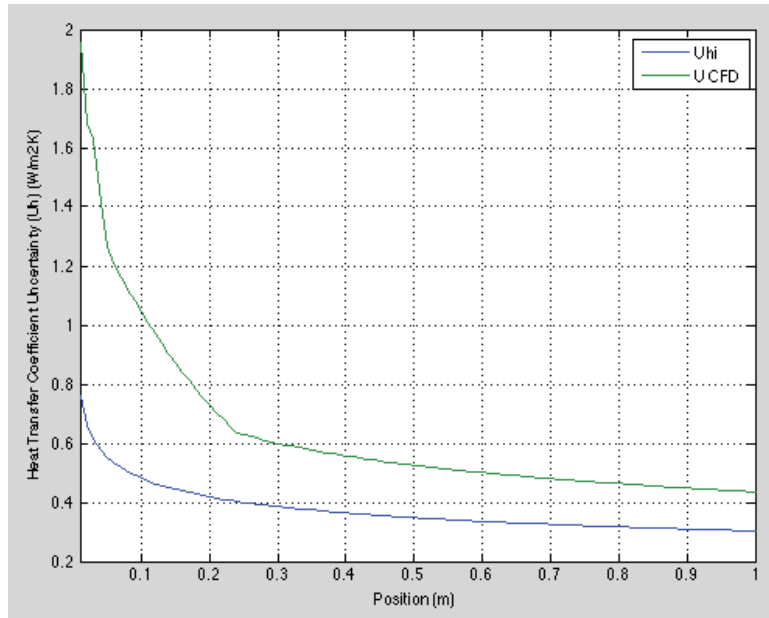
Figure 57: Heat Transfer Coefficient Uncertainty – CFD Data vs. Correlation

Figure 58 is a plot of both the CFD results with uncertainty and the correlation results with uncertainty. It can be concluded that the proposed CFD uncertainty method encompasses the correlation results.
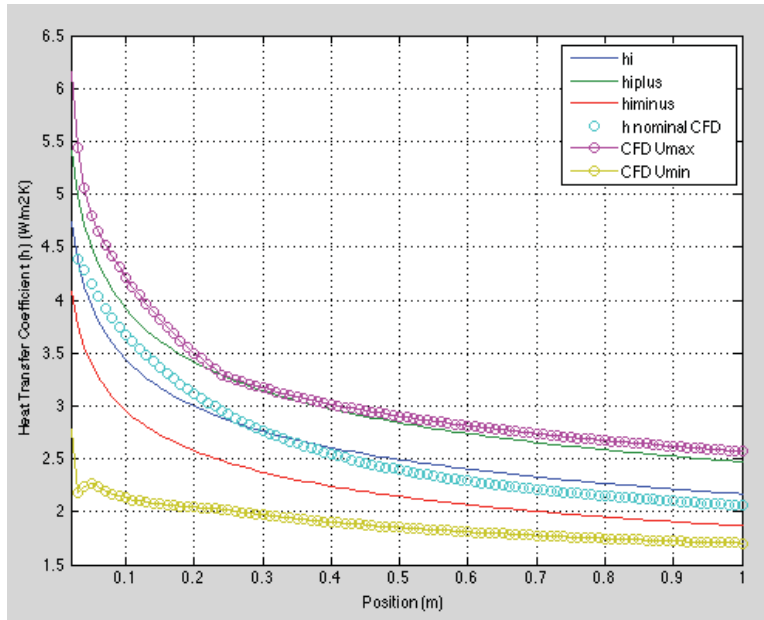
Figure 58: Heat Transfer Coefficient with Uncertainty – CFD Data vs. Correlation

Another interesting item to note is that a k-value of 1.2 provides an uncertainty estimate close to

the traditional correlation estimate as shown in Figure 59.
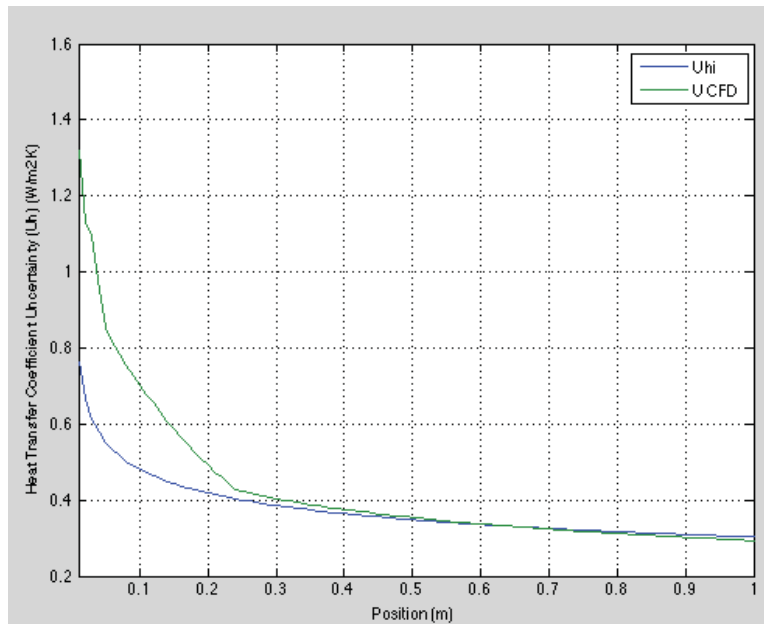
Figure 59: Heat Transfer Coefficient Uncertainty – CFD Data (k=1.2) vs. Correlation

Another item of interest is the average heat transfer coefficient over the entire flat plate and uncertainty as calculated from each method. The traditional way using the correlation and partial derivatives the average heat transfer coefficient is 2.66+/- 0.74 [W/m^{2}K] and the proposed methodology is 2.66+/-1.39 [W/m^{2}K].

The work is this chapter was needed before moving into the problem of interest, Spacecraft ECS Systems. The proposed methodology proved accurate for two simple problems, laminar flow between parallel plates and heat transfer over a flat plate. Additionally, the interpolation direction and methodology was developed using the 'linear' option to interpolate all solutions onto the course grid.

CHAPTER SIX: DEMONSTRATION AND IMPLEMENTATION OF THE PROPOSED CFD UNCERTAINITY METHOD FOR SPACECRAFT ECS SYSTEMS

The information in this chapter was published by the author in AIAA-2014-0440 [21]. To further elaborate the problem and purpose. There have been few discussions on using Computational Fluid Dynamics (CFD) without experimental validation. Pairing experimental data, uncertainty analysis, and analytical predictions provides a comprehensive approach to verification and is the current state of the art. With pressed budgets, collecting experimental data is rare or non-existent. This section investigates and proposes a method to perform CFD uncertainty analysis only from computational data. The method uses current CFD uncertainty techniques coupled with the Student-T distribution to predict the velocity magnitude and uncertainty for payload fairing ECS systems. The inputs to the CFD model are varied from a specified tolerance or bias error and the difference in the results are used to estimate the uncertainty. The variation in each input is ranked from least to greatest to determine the order of importance. The results provide a tactic to analytically estimate the uncertainty in a CFD model when experimental data is unavailable.

Chapter 6 is organized by configuration. Chapter 6.1 describes the results of the 0.75m configuration, chapter 6.2 the 3.5m configuration, and chapter 6.3 the 5.5m configuration. Chapter 6.4 demonstrates the proposed methodology as compared to experimental data previously published. The remainder of this section is a summary of the uncertainty calculation and input parameters used for the ECS system analysis. The following uncertainty parameters were considered as shown in Table 11.

Table 11: Parameters for ECS System

Input Variable	Description	Bias
Grid	3 grids considered for each configuration	
Inlet Velocity	Boundary Condition low and high	10%
Outlet Pressure	Boundary Condition low and high	2%
Turbulence Model	SA, ke-realizable, kwSST	
Wall Functions	with and without	
Rough Wall Function	smooth vs. rough	
Compressibility	incompressible vs. compressible	
Solver	OpenFoam, Fluent, STARCCM+	
Fluid Properties	kinematic viscosity nu represents air [0-50-100] deg C	1.36,1.5,2.306e-05

The proposed uncertainty calculation is shown in equation 42 and 43. The process is the same as described in section 2.3. The k-value of 1.746 was used because there are seventeen different CFD cases that ran for each configuration. The cases are listed in Table 12.

Table 12: Cases Solved

Case #	Parameter Grid		Configuration 0.75	3.5	5.5
1		coarse	1	1	1
2		med	2	2	2
3		fine	3	3	3

Case #	Boundary Conditions				
4		inlet velocity low	4	4	4
5		inlet velocity high	5	5	5
6		pressure outlet low	6	6	6
7		pressure outlet high	7	7	7

Turbulence Models					
8		SA	8	8	8
9		ke-realizable - same as1	9	9	9
10		kwsst	10	10	10

11	Wall Functions	without wall functions	11	11	11
12	Surface Roughness	rough wall function	12	12	12
13	Compressibility	different openfoam solver	13	13	13

Solver					
14		fluent	14	14	14
15		starccm	15	15	15

Fluid Properties					
16		nut high	16	16	16
17		nut low	17	17	17

$$u_{val} = 1.746 * \left(\left(\left(\frac{\partial V}{\partial grid} \right)^2 B_{grid}^2 \right) + \left(\left(\frac{\partial V}{\partial inletvelocity} \right)^2 B_{inletvelocity}^2 \right) \right.$$

$$+ \left(\left(\frac{\partial V}{\partial outletpressure} \right)^2 B_{outletpressure}^2 \right)$$

$$+ + \left(\left(\frac{\partial V}{\partial turbulence} \right)^2 B_{turbulence}^2 \right)$$

$$+ \left(\frac{\partial V}{\partial wallfunctions} \right)^2 B_{wallfunctions}^2 + \left(\frac{\partial V}{\partial roughwall} \right)^2 B_{roughwall}^2$$

$$+ \left(\frac{\partial V}{\partial compressibility} \right)^2 B_{compressibility}^2 + \left(\frac{\partial V}{\partial solver} \right)^2 B_{solver}^2$$

$$\left. + \left(\frac{\partial V}{\partial fluidproperties} \right)^2 B_{fluidproperties}^2 \right)^{1/2}$$

(58)

$$u_{val} = 1.746 * \left| \frac{1}{2} (S_U - S_L) \right|$$

(59)

6.1 0.75m Configuration

The 0.75m configuration was analyzed according to the methodology laid out above and a solution iterated on three separate grids and all input variables considered in the uncertainty analysis. The solution is shown in section 6.1.1, the uncertainty in section 6.1.2, the ranking of input parameters in section 6.1.3. Section 6.1.4 is a discussion of the data presented.

6.1.1 Solution

Contour plots of the 0.75m configuration are shown in this section along with corresponding line plots. The line plots include all of the models that were ran for the uncertainty analysis.
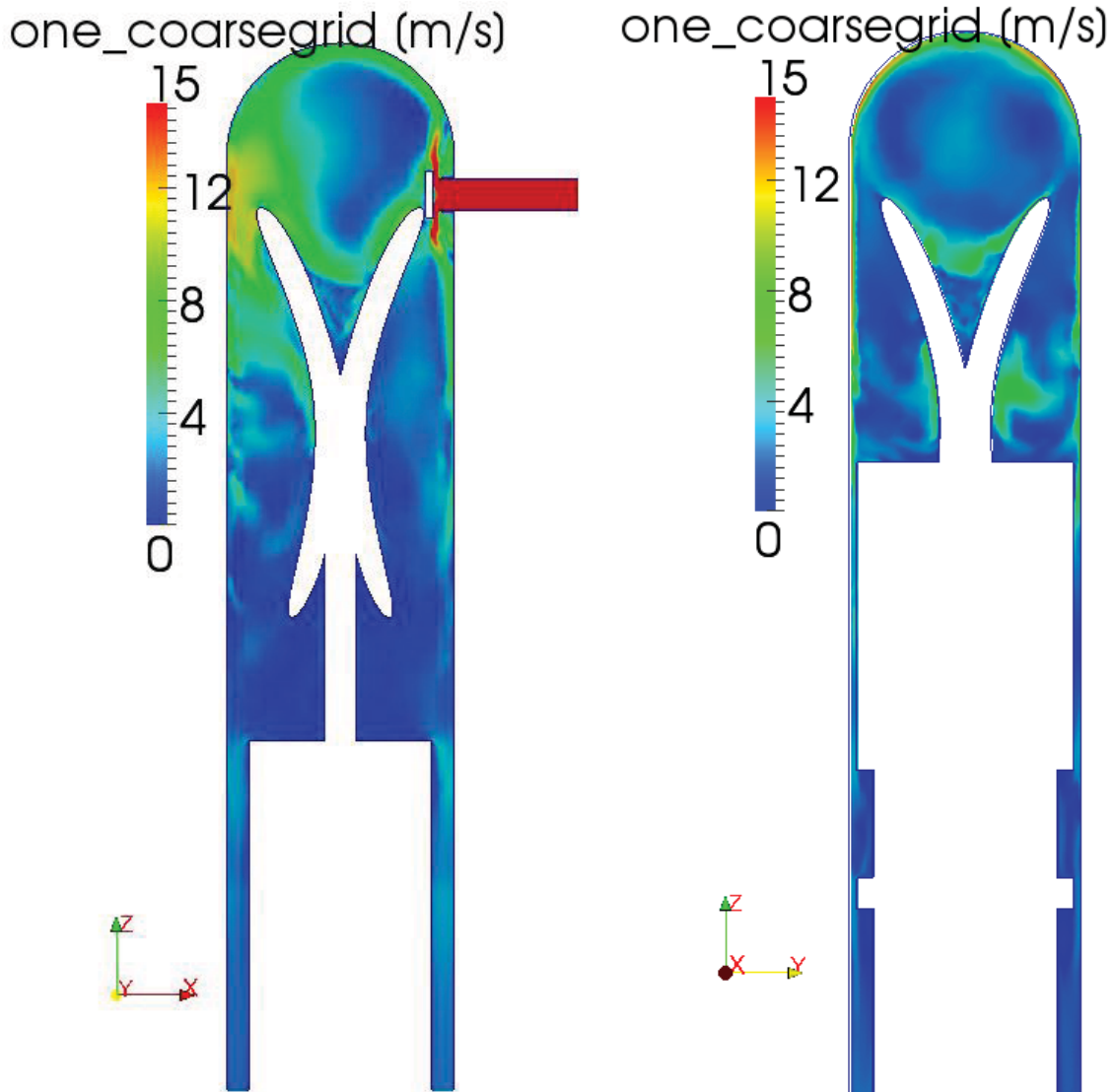
Figure 60: CFD Results for 0.75m Configuration – Contour Plot
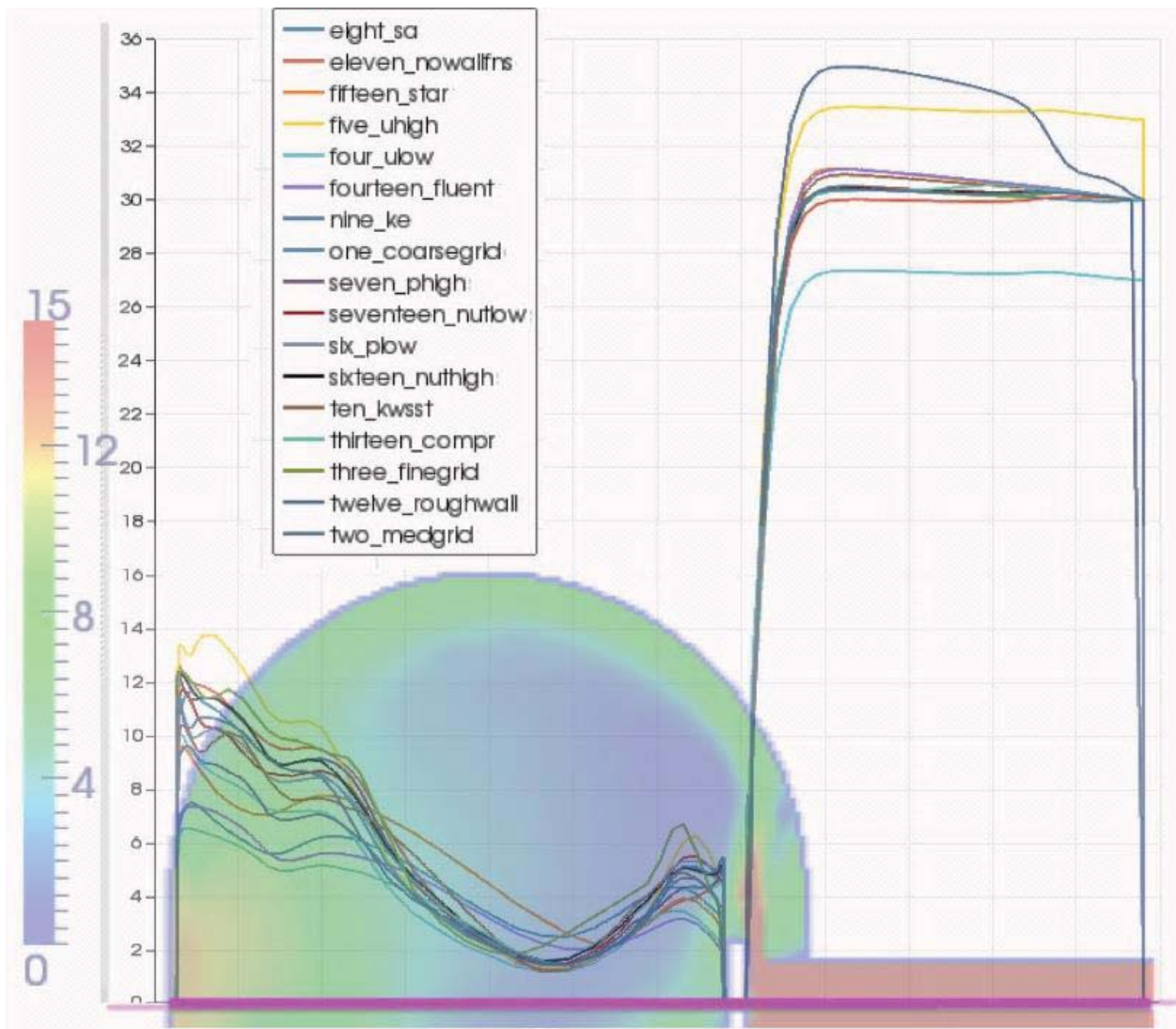
Figure 61: CFD Results for 0.75m Configuration – Line Plot (m/s)
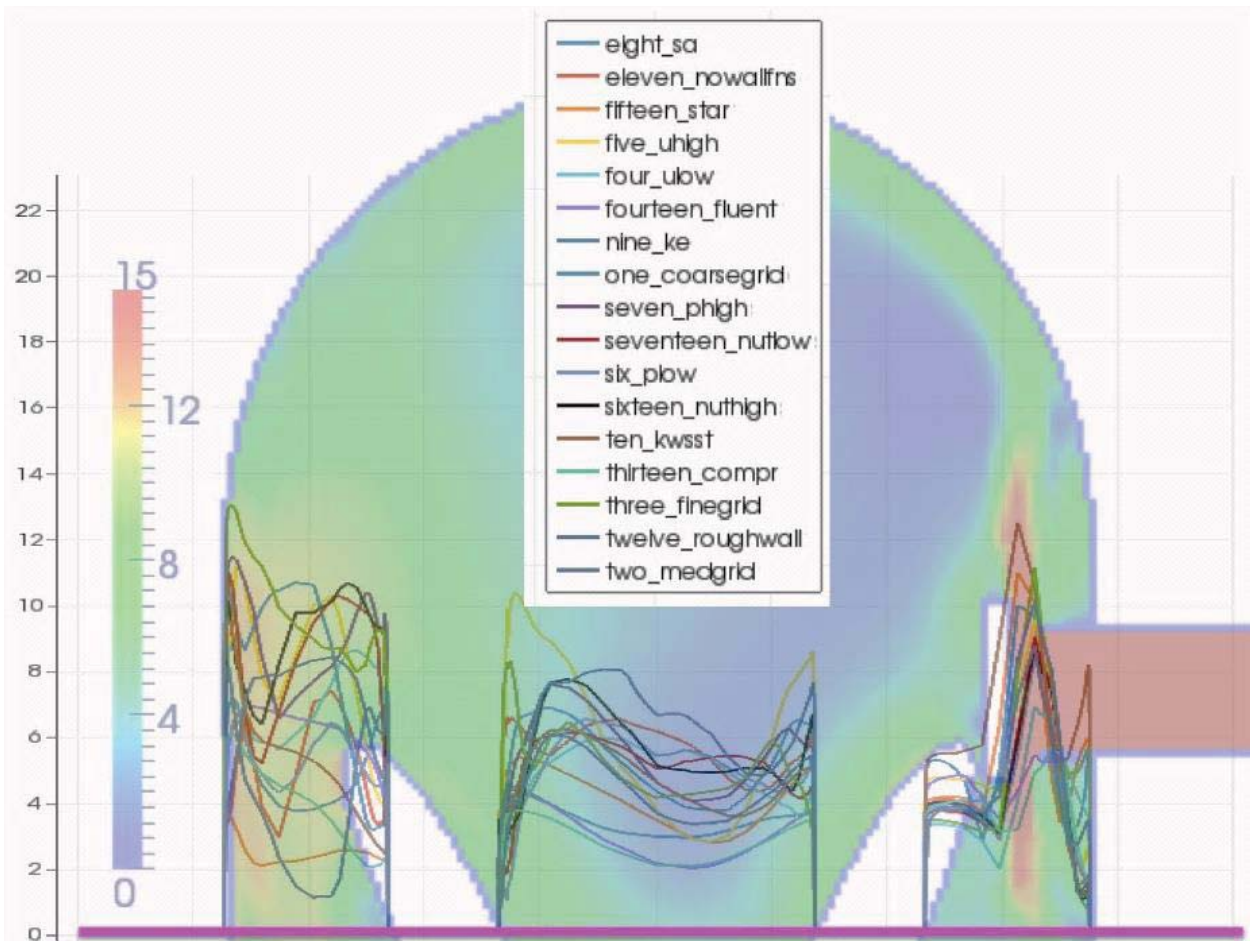
Figure 62: CFD Results for 0.75m Configuration – Line Plot (m/s)
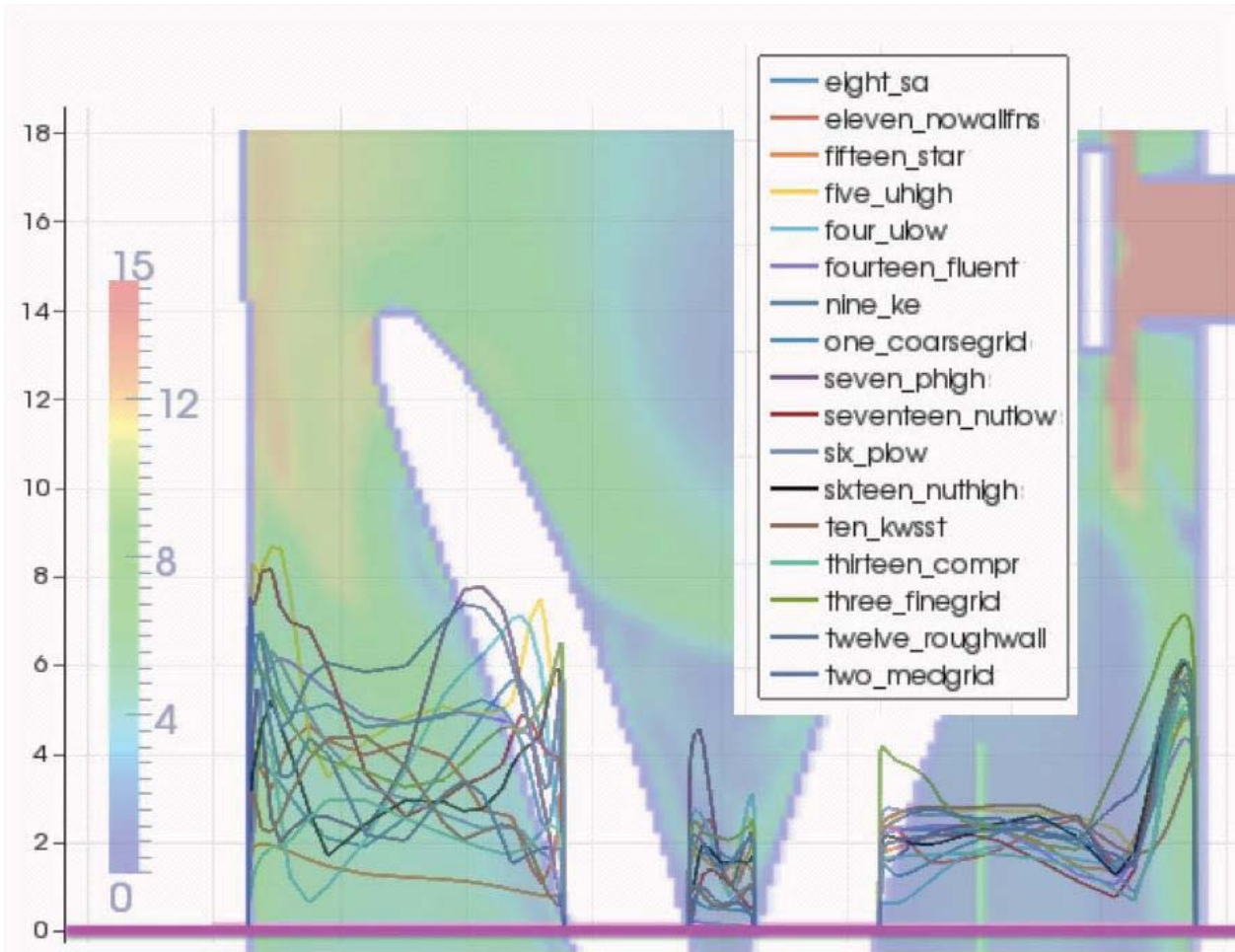
Figure 63: CFD Results for 0.75m Configuration – Line Plot (m/s)
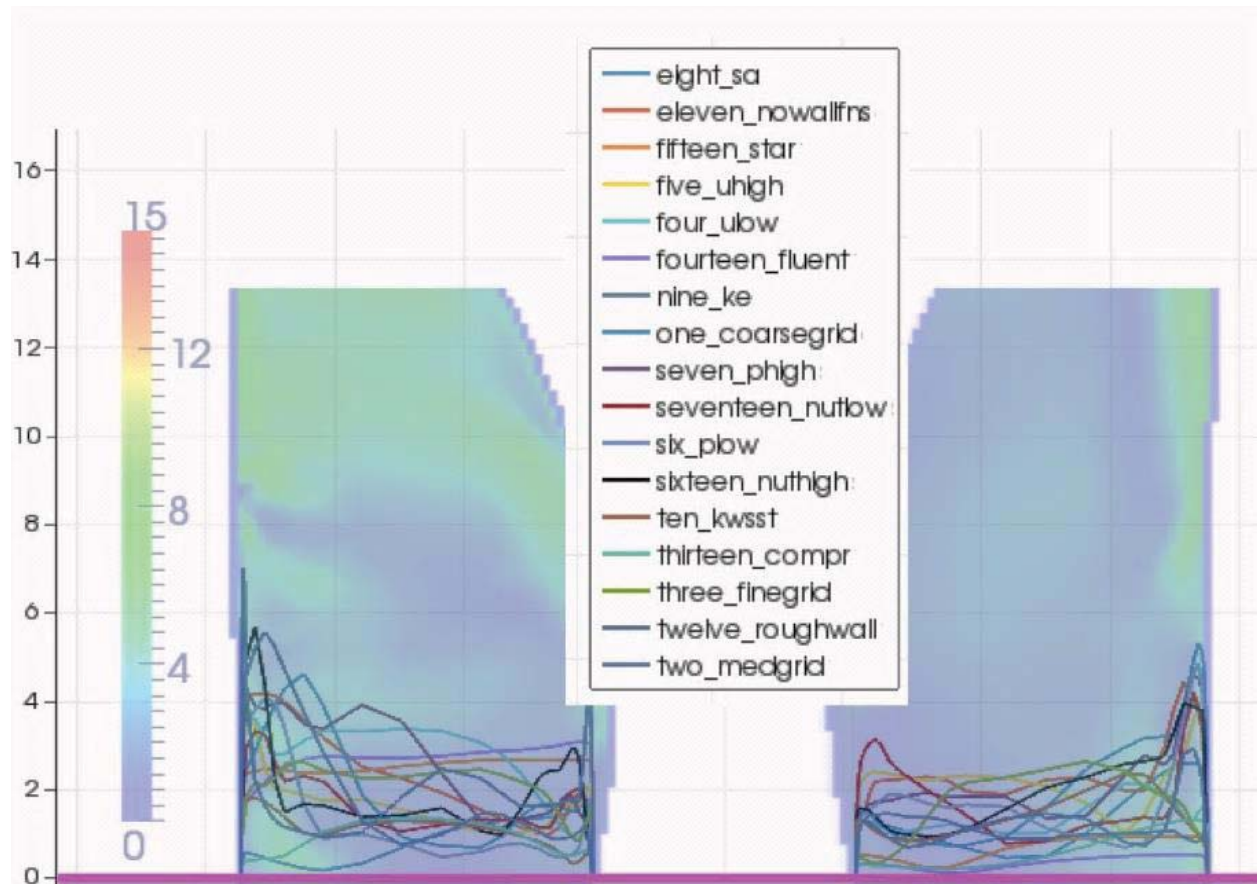
Figure 64: CFD Results for 0.75m Configuration – Line Plot (m/s)

6.1.2 Uncertainty

The uncertainty was calculated using equation 29. The process was to find the minimum, maximum, and middle for each point in the domain and add or subtract the uncertainty calculated from equation 29. Figure 65 plots the results in contour plot with the corresponding uncertainty contour plot.
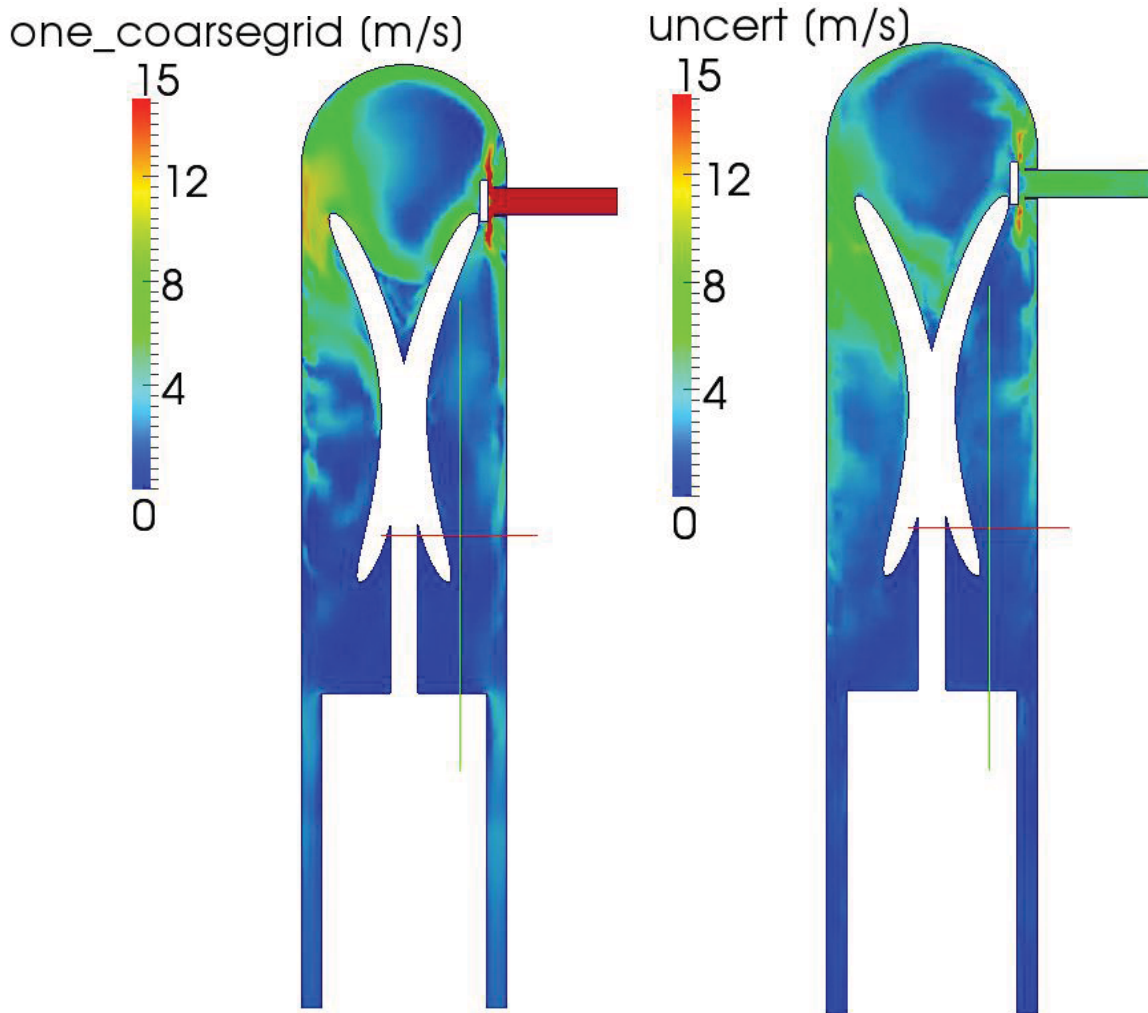
Figure 65: CFD Results and Uncertainty for 0.75m Configuration – Contour Plot (m/s)

A more qualitative way to view the results is to plot the uncertainty minimum and maximum

for the locations of interest in the domain on a line plot. This is shown in Figure 66 to Figure 69.
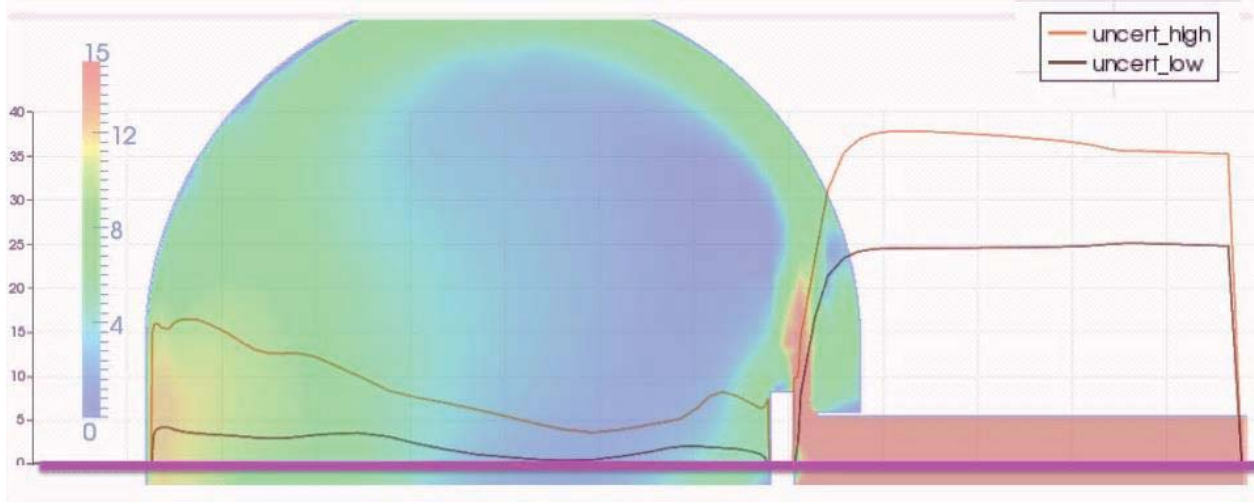
Figure 66: CFD Uncertainty Bounds for 0.75m Configuration – Line Plot (m/s)
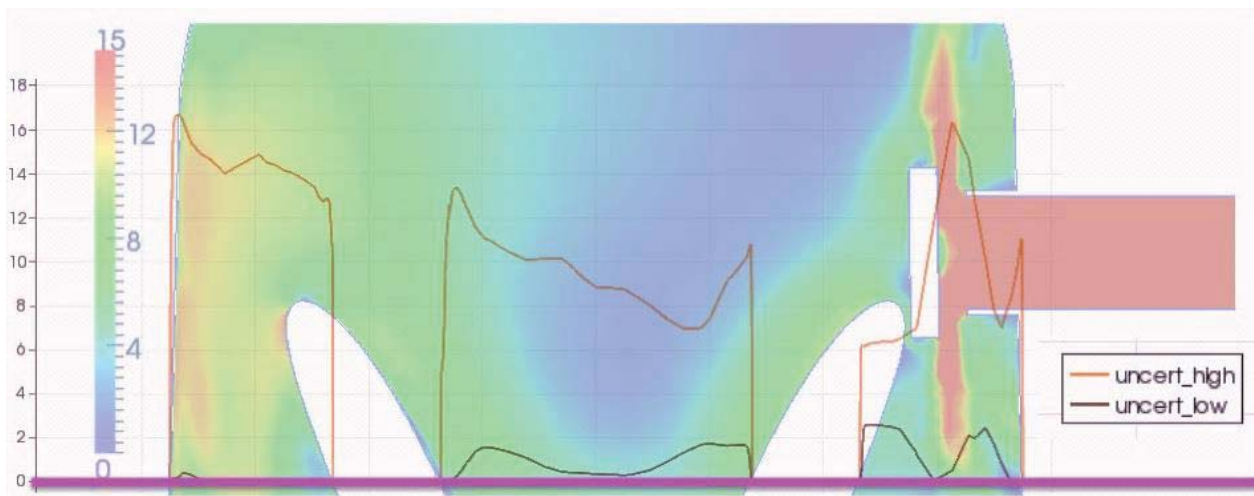
Figure 67: CFD Uncertainty Bounds for 0.75m Configuration – Line Plot (m/s)
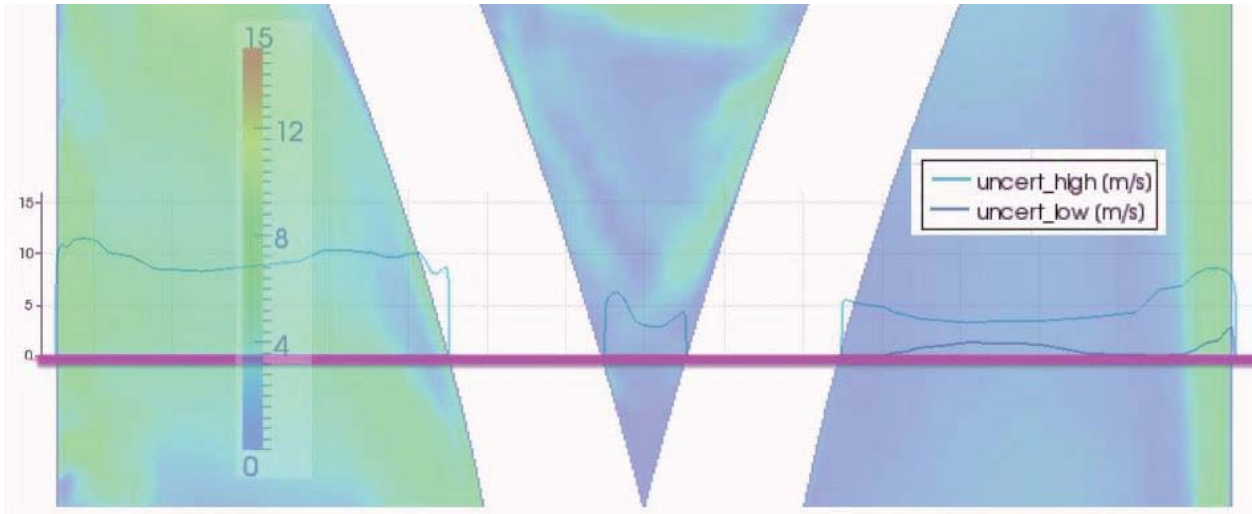
Figure 68: CFD Uncertainty Bounds for 0.75m Configuration – Line Plot (m/s)
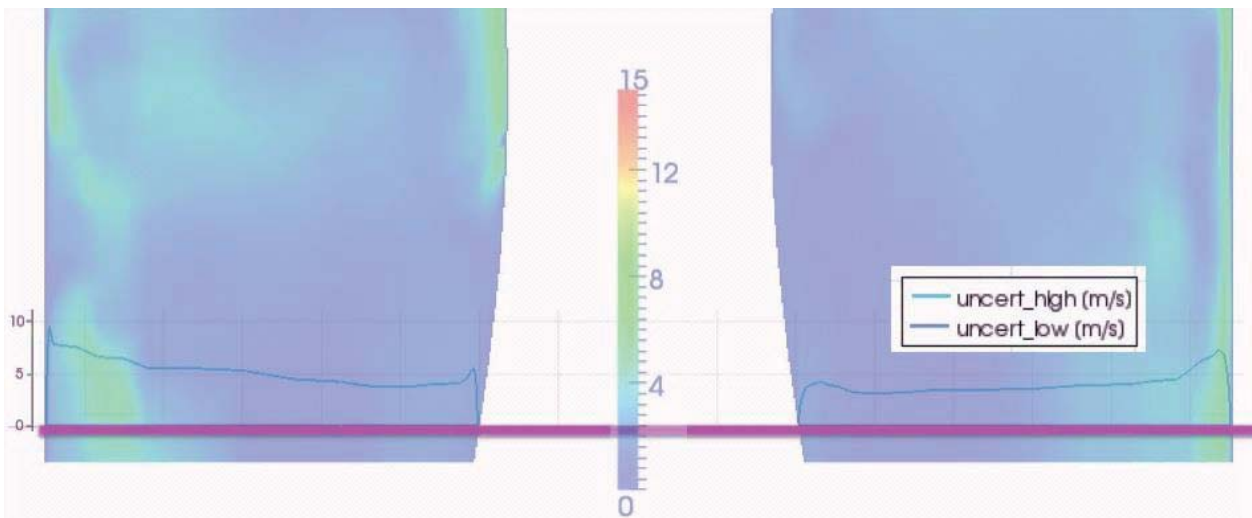
Figure 69: CFD Uncertainty Bounds for 0.75m Configuration – Line Plot (m/s)

6.1.3   Ranking

The uncertainty for each of the input variables were ranked by the non-dimensionalizing the difference in the results by the freestream value and ranking from greatest uncertainty to least uncertainty. The results are provided in Table 13. The mean velocity uncertainty is the average of all the cells uncertainty by input variable. The normalized ranking percentage is calculated by adding all the mean non-dimensionalized uncertainties and normalizing each input variable by the sum. The ranking is numbered from 1 to 9 and corresponds to the input variables that have the most uncertainty from greatest to least. This was done to rank the inputs from most uncertain to least uncertain in terms of the velocity magnitude results.

Table 13: 0.75m Configuration – Non-Dimensional Ranking of Uncertainty

Input Variable	Description	Bias	Mean Velocity Uncertainty (m/s)	Mean Non-Dimensionalized Uncertainty	Normalized Ranking %	Numbered Ranking
Grid	3 grids considered		1.6287	0.0543	13.40	2
Inlet Velocity	Boundary Condition	10%	1.3115	0.04737	11.69	5
Outlet Pressure	Boundary Condition	2%	1.1478	0.0383	9.45	8
Turbulence Model	SA, ke-realizable, kwSST		1.4628	0.0488	12.04	4
Wall Functions	with and without		0.8286	0.0276	6.81	9
Rough Wall Function	smooth vs. rough		1.5237	0.0508	12.53	3
Compressibility	incompressible vs. compressible		1.3128	0.0438	10.81	6
Solver	OpenFoam, Fluent, STARCCM+		1.673	0.0558	13.77	1
Fluid Properties	kinematic viscosity nu represents air [0-50-100] deg C	1.36,1.5,2.306e-05	1.1536	0.0385	9.50	7

6.1.4   Discussion

It is interested to note that the variable with the most uncertainty for this configuration was the

solver. The same grid and boundary conditions were used in OPENFOAM, STARCCM+, and

FLUENT. The other items to not are the uncertainty for each input variable is on the same order

of magnitude. It was hypothesized that there would be only a few input variables of significant importance. No order of magnitude analysis can be completed or conclusions drawn about the significance of each input variable. All of the input variables are significant and contribute to the overall uncertainty calculated.

6.2   3.5m Configuration

The 3.5m configuration was analyzed according to the methodology laid out above and a solution iterated on three separate grids and all input variables considered in the uncertainty analysis. The solution is shown in section 6.2.1, the uncertainty in section 6.2.2, the ranking of input parameters in section 6.2.3. Section 6.2.4 is a discussion of the data presented.

6.2.1   Solution

Contour plots of the 3.5m configuration are shown in this section along with corresponding line plots. The line plots include all of the models that were ran for the uncertainty analysis.
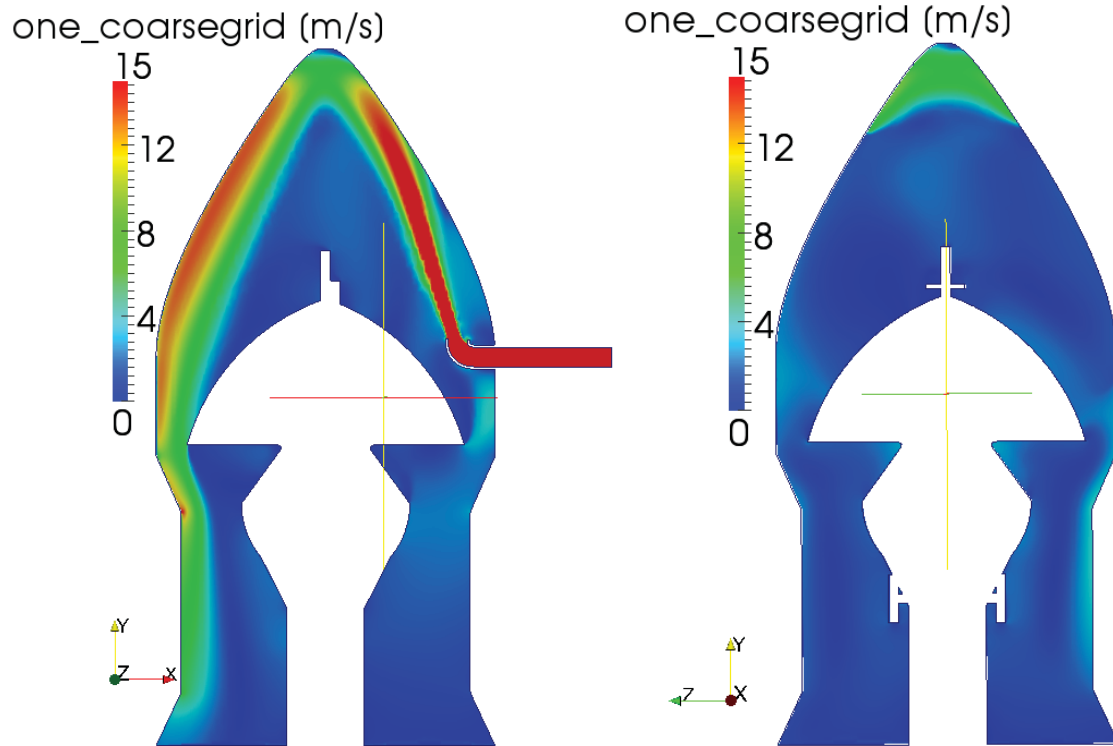
Figure 70: CFD Results for 3.5m Configuration – Contour Plot (m/s)
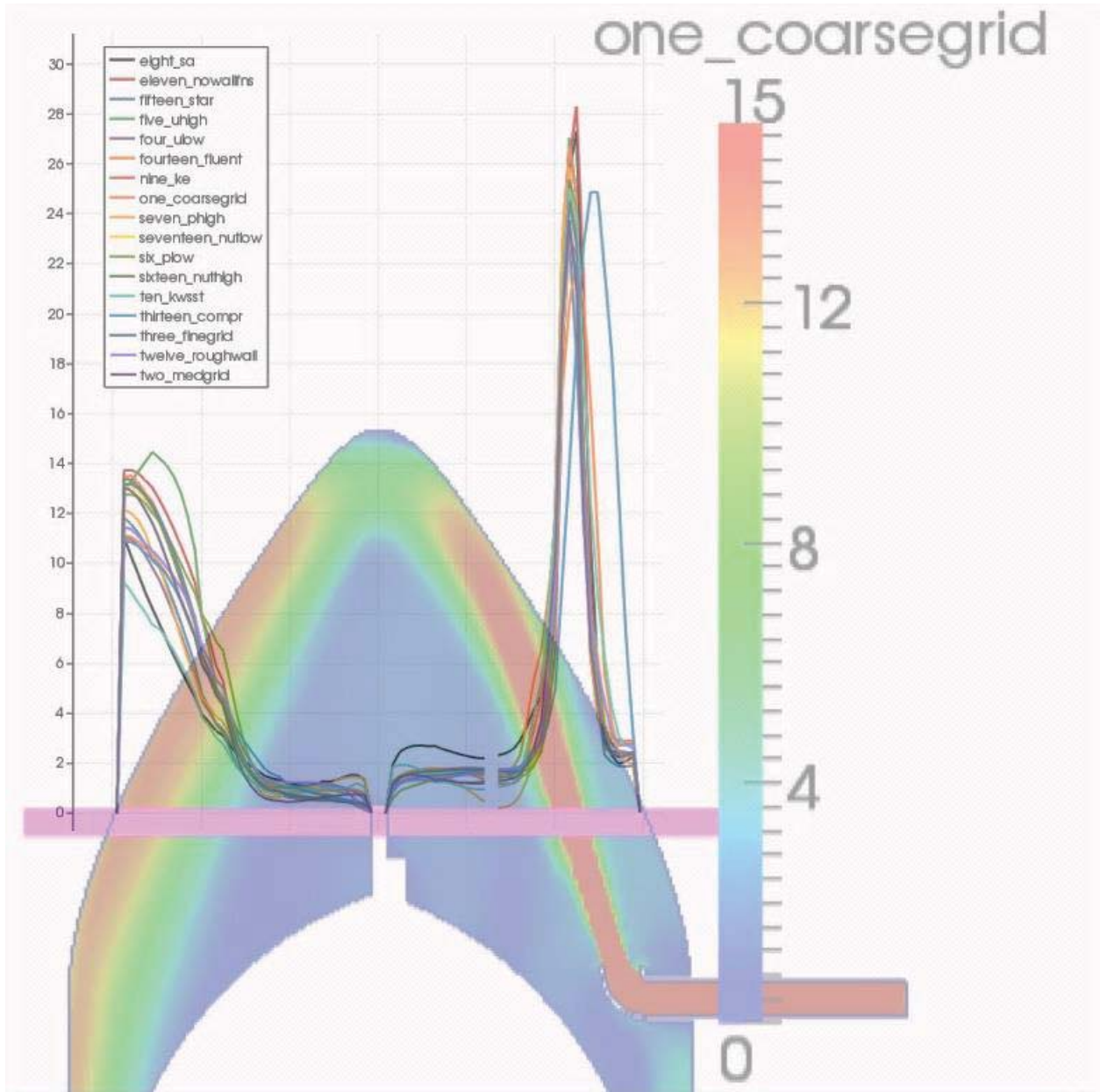
Figure 71: CFD Results for 3.5m Configuration – Line Plot (m/s)
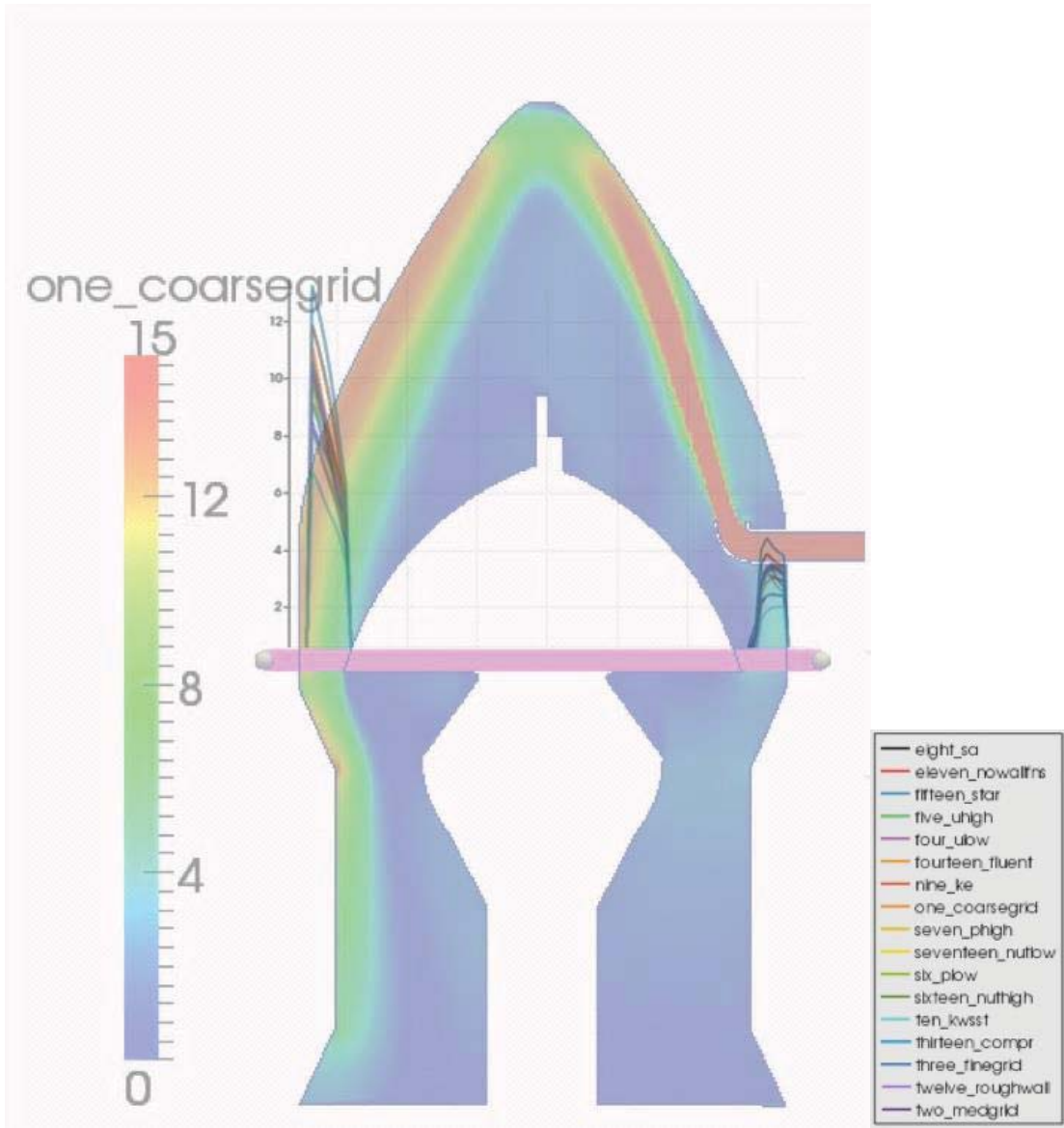
Figure 72: CFD Results for 3.5m Configuration – Line Plot (m/s)

6.2.2   Uncertainty

The uncertainty was calculated using equation 29. The process was to find the minimum, maximum, and middle for each point in the domain and add or subtract the uncertainty calculated from equation 29. Figure 73 plots the results in contour plot with the corresponding uncertainty contour plot.
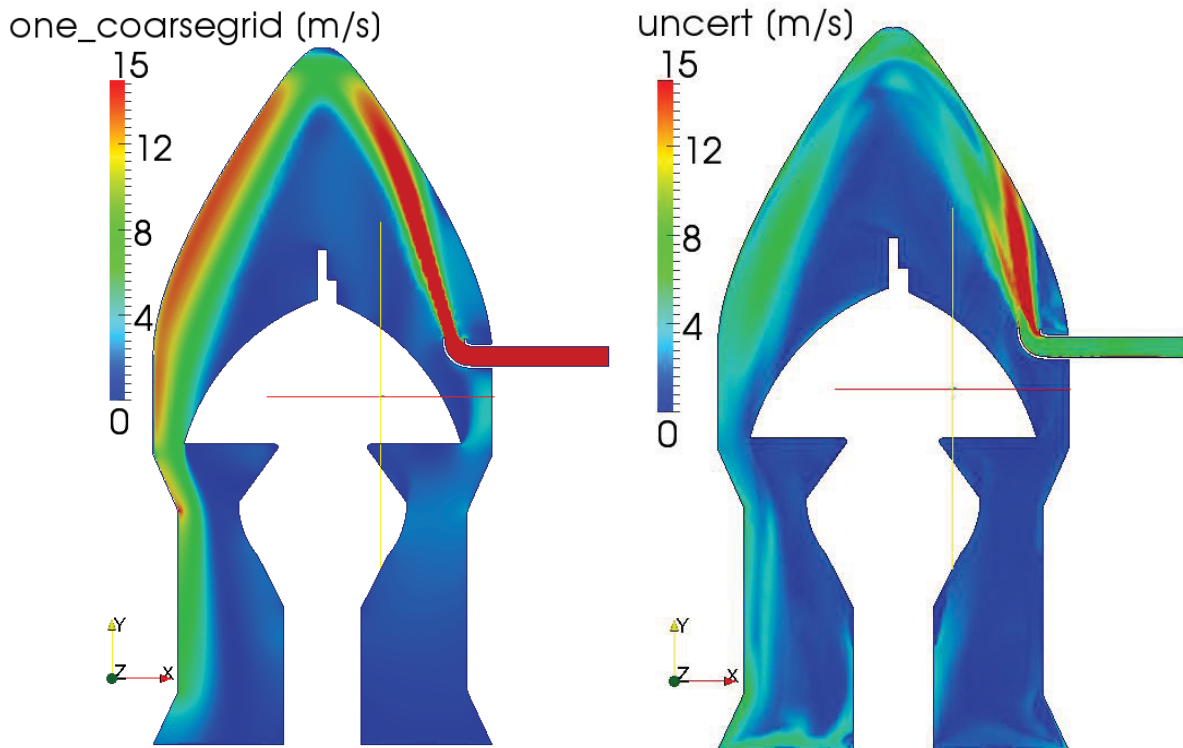
Figure 73: CFD Results and Uncertainty for 3.5m Configuration – Contour Plot (m/s)
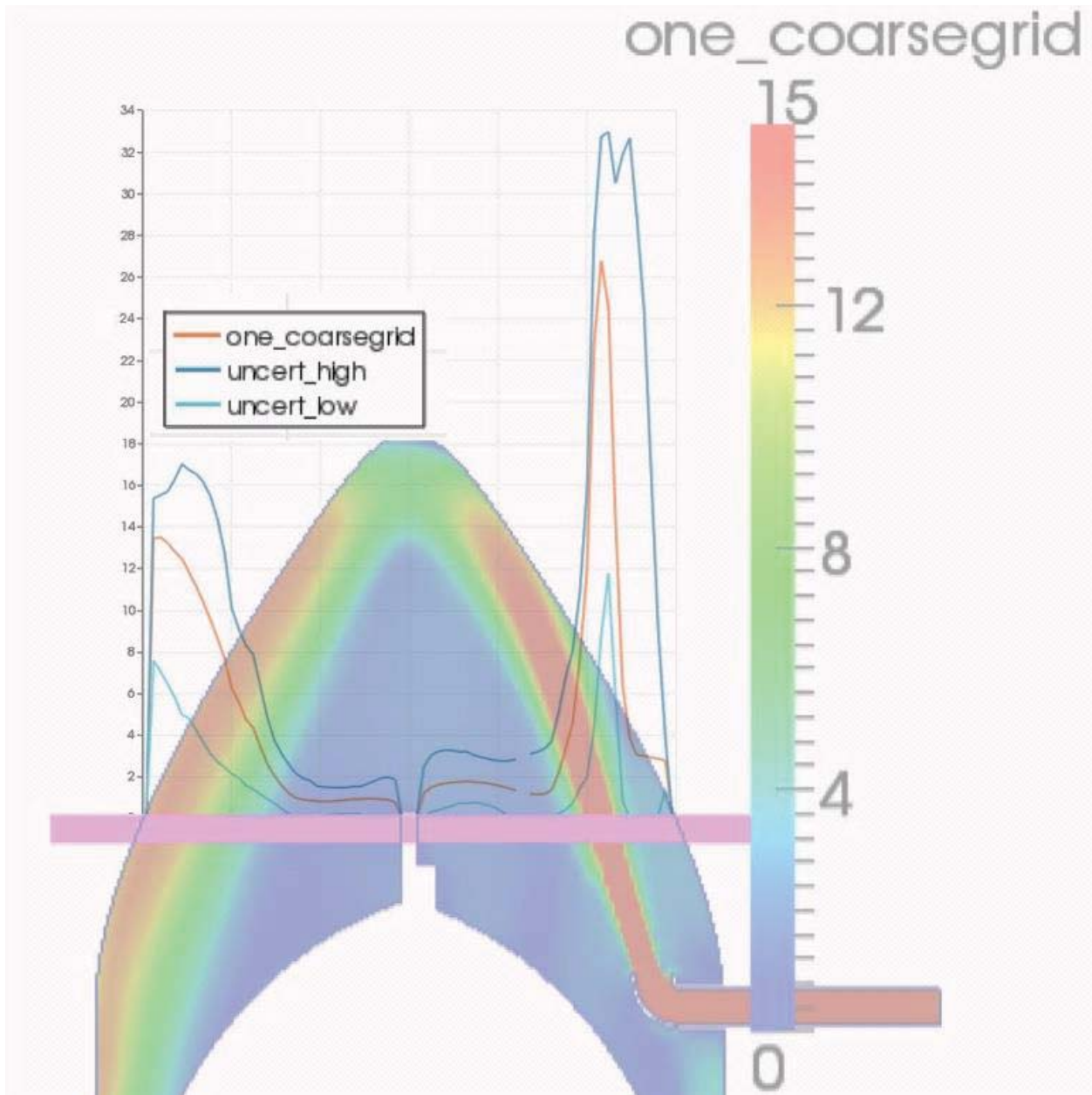
Figure 74: CFD Uncertainty Bounds for 3.5m Configuration – Line Plot (m/s)
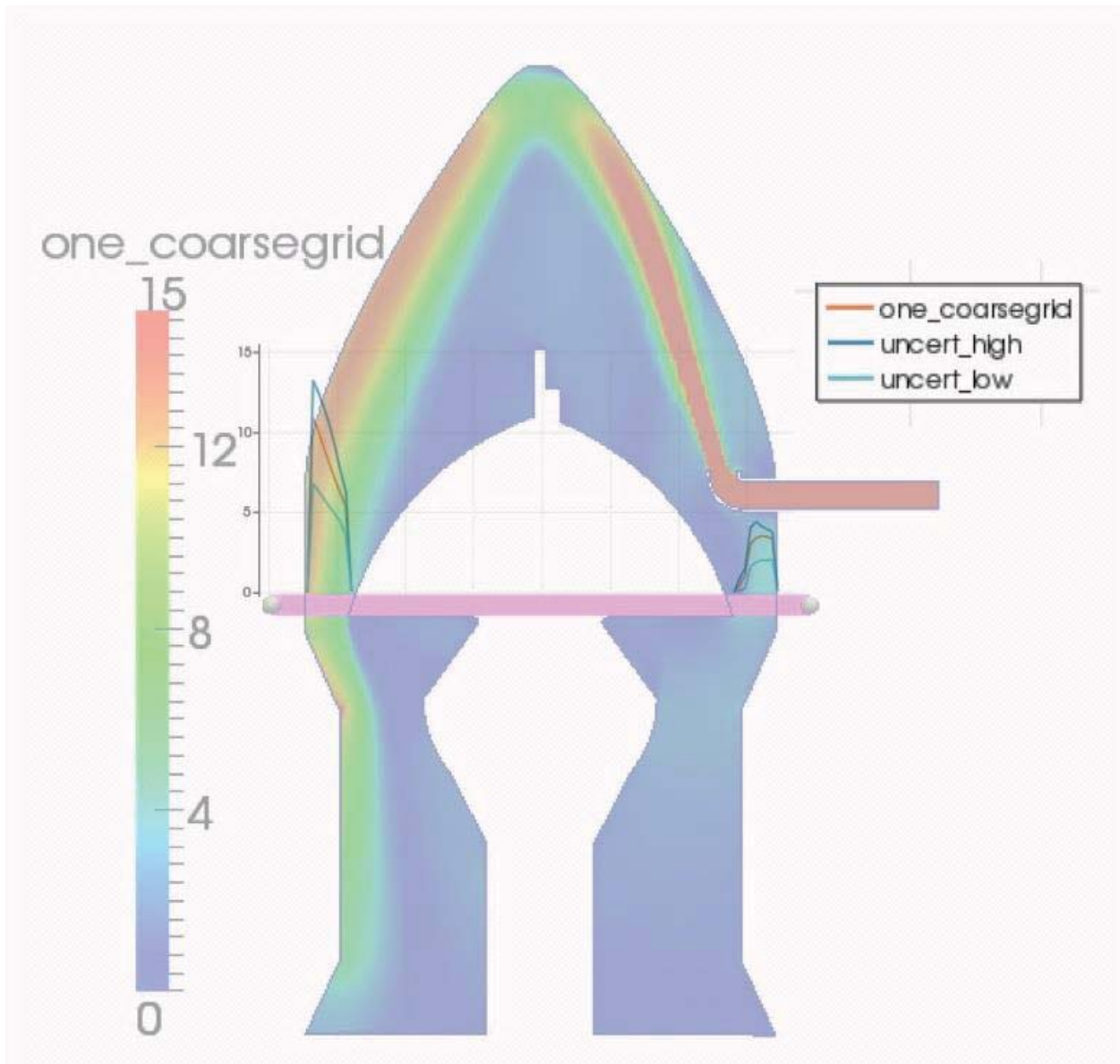
Figure 75: CFD Uncertainty Bounds for 3.5m Configuration – Line Plot (m/s)

6.2.3 Ranking

The uncertainty for each of the input variables were ranked by the non-dimensionalizing the difference in the results by the freestream value and ranking from greatest uncertainty to least uncertainty. The results are provided in Table 14.

Table 14: 3.5m Configuration – Non-Dimensional Ranking of Uncertainty

Input Variable	Description	Bias	Mean Velocity Uncertainty (m/s)	Mean Non-Dimensionalized Uncertainty	Normalized Ranking %	Numbered Ranking
Grid	3 grids considered		0.6829	0.0228	8.28	7
Inlet Velocity	Boundary Condition	10%	0.7919	0.0264	9.59	6
Outlet Pressure	Boundary Condition	2%	1.4606	0.0487	17.70	1
Turbulence Model	SA, ke-realizable, kwSST		1.3487	0.045	16.35	2
Wall Functions	with and without		0.6139	0.0205	7.45	9
Rough Wall Function	smooth vs. rough		1.0531	0.0351	12.75	3
Compressibility	incompressible vs. compressible		0.8252	0.0275	9.99	5
Solver	OpenFoam, Fluent, STARCCM+		0.841	0.028	10.17	4
Fluid Properties	kinematic viscosity nu represents air [0-50-100] deg C	1.36,1.5, 2.306e-05	0.6345	0.0212	7.70	8

6.2.4 Discussion

The ranking analysis of the 3.5m configuration shows a different trend of input variables as being most significant to least significant when compared to the 0.75m configuration. The most significant for this configuration was the outlet pressure boundary condition, followed by turbulence model, wall functions, and solver. Again noting that there is not a significant difference in the uncertainty for the different input variables. Each variable is of the same order of magnitude and cannot be regarded as insignificant to the overall uncertainty calculation.

6.3   5.5m Configuration

The 5.5m configuration was analyzed according to the methodology laid out above and a solution iterated on three separate grids and all input variables considered in the uncertainty analysis. The solution is shown in section 6.3.1, the uncertainty in section 6.3.2, the ranking of input parameters in section 6.3.3. Section 6.3.4 is a discussion of the data presented.

6.3.1 Solution

Contour plots of the 5.5m configuration are shown in this section along with corresponding line plots. The line plots include all of the models that were ran for the uncertainty analysis.
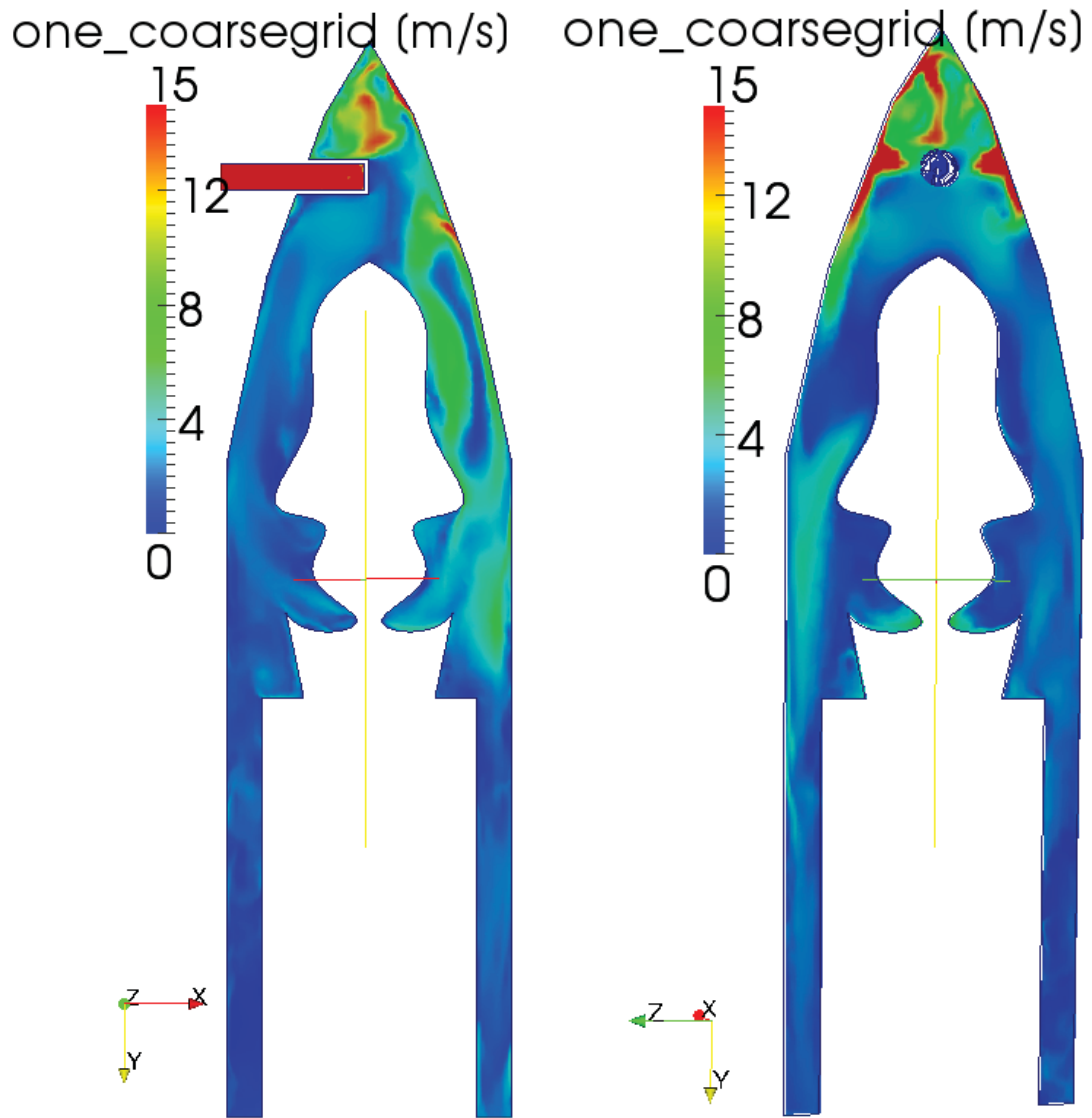
Figure 76: CFD Results for 5.5m Configuration – Contour Plot (m/s)
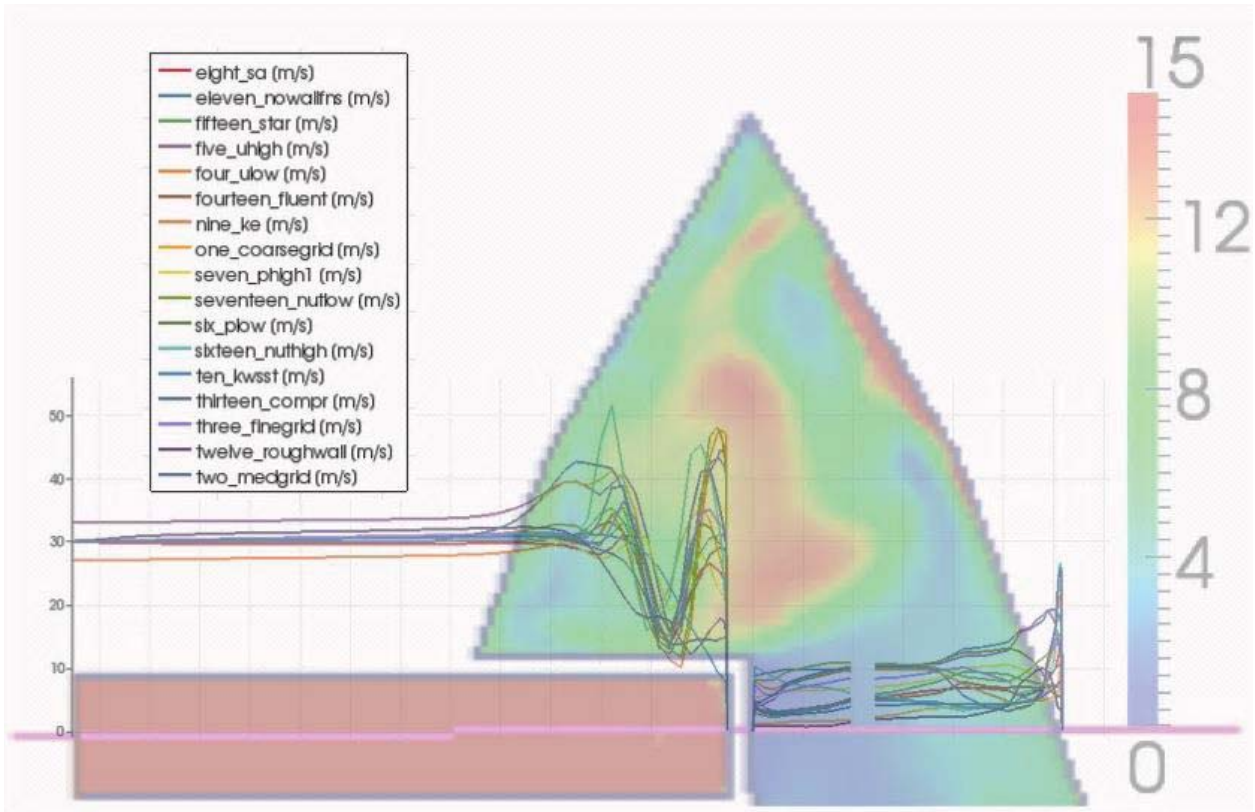
Figure 77: CFD Results for 5.5m Configuration – Line Plot (m/s)
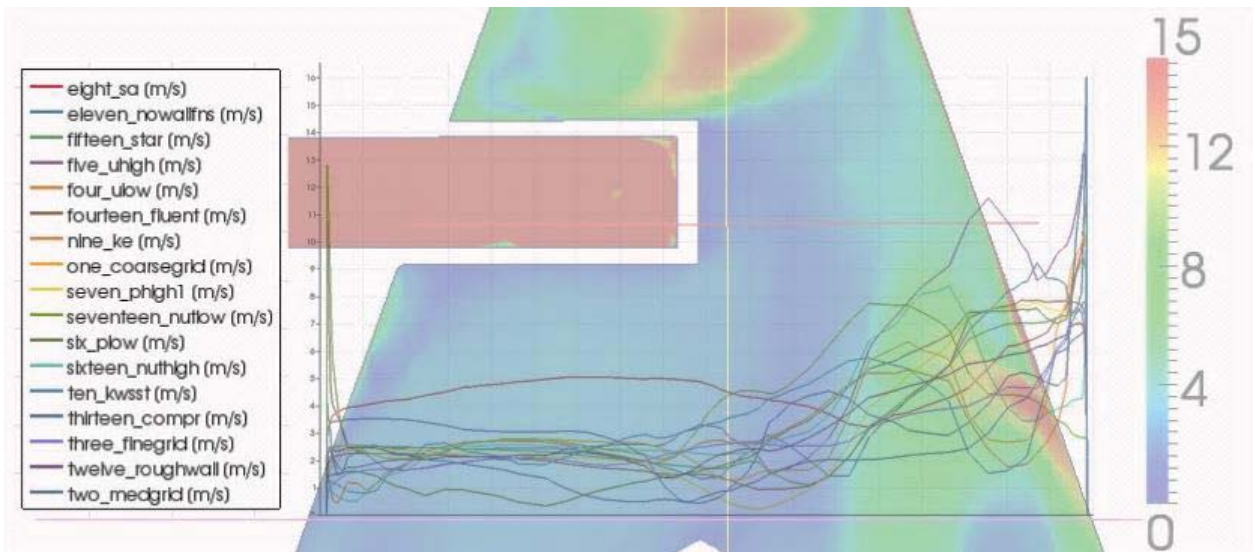

Figure 78: CFD Results for 5.5m Configuration – Line Plot (m/s)
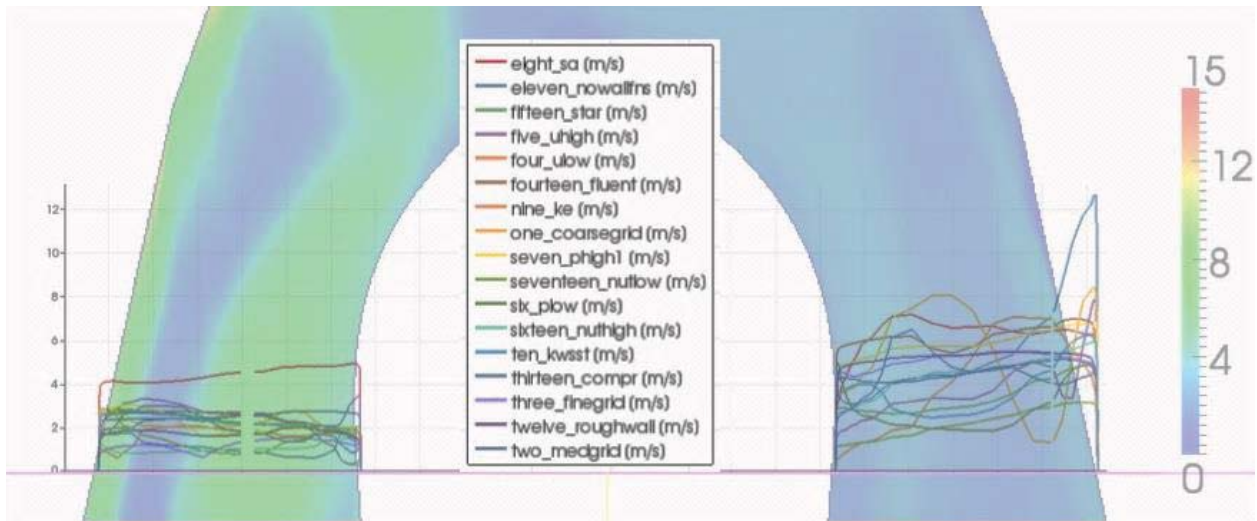
Figure 79: CFD Results for 5.5m Configuration – Line Plot (m/s)
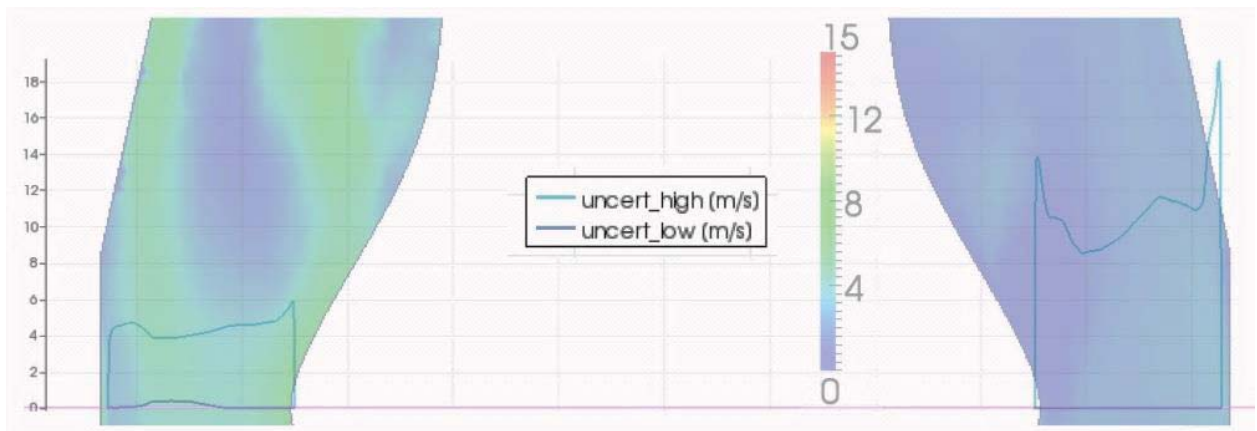
Figure 80: CFD Results for 5.5m Configuration – Line Plot (m/s)

6.3.2   Uncertainty

The uncertainty was calculated using equation 29. The process was to find the minimum,

maximum, and middle for each point in the domain and add or subtract the uncertainty calculated

from equation 29. Figure 81 plots the results in contour plot with the corresponding uncertainty contour plot.
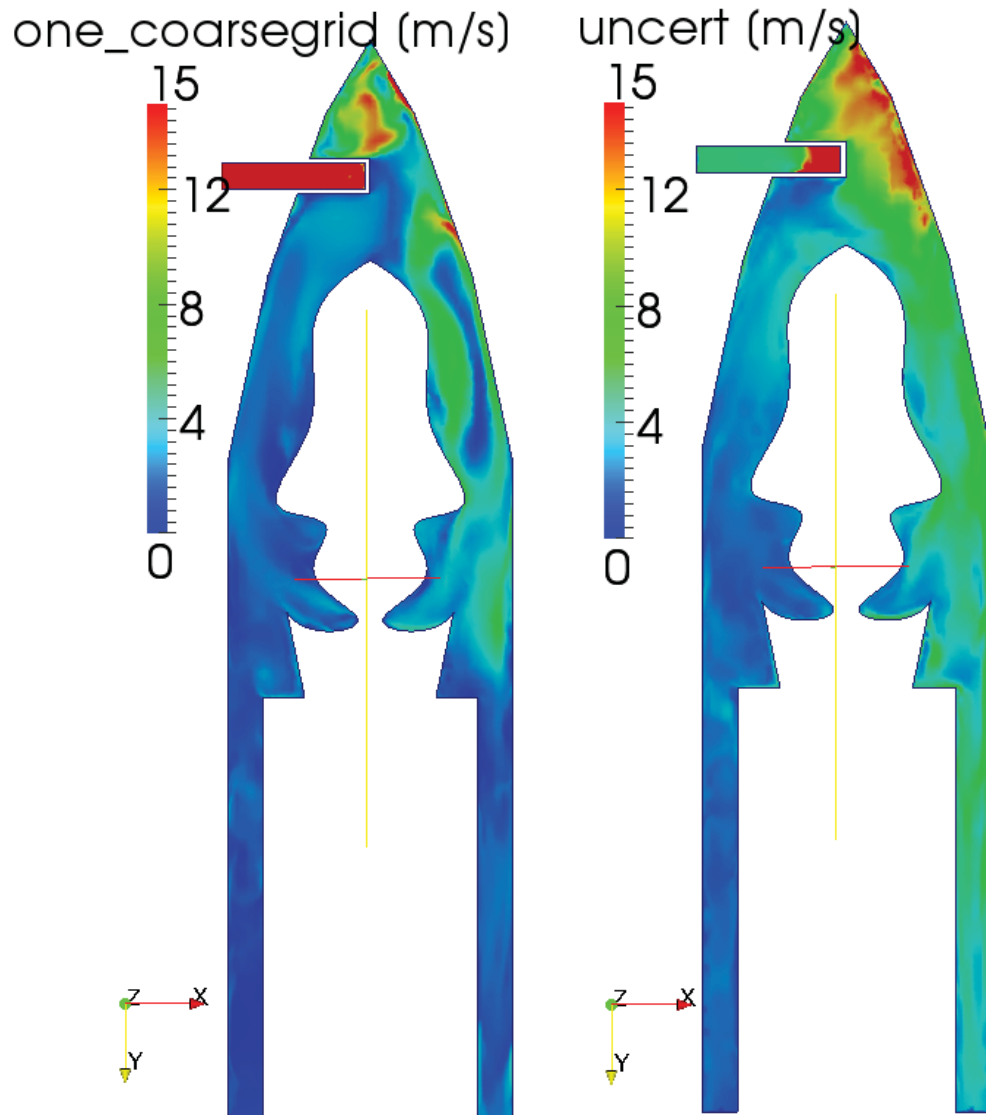

Figure 81: CFD Results and Uncertainty for 5.5m Configuration – Contour Plot (m/s)

A more qualitative way to view the results is to plot the uncertainty minimum and maximum for the locations of interest in the domain on a line plot. This is shown in Figure 82 to Figure 85.
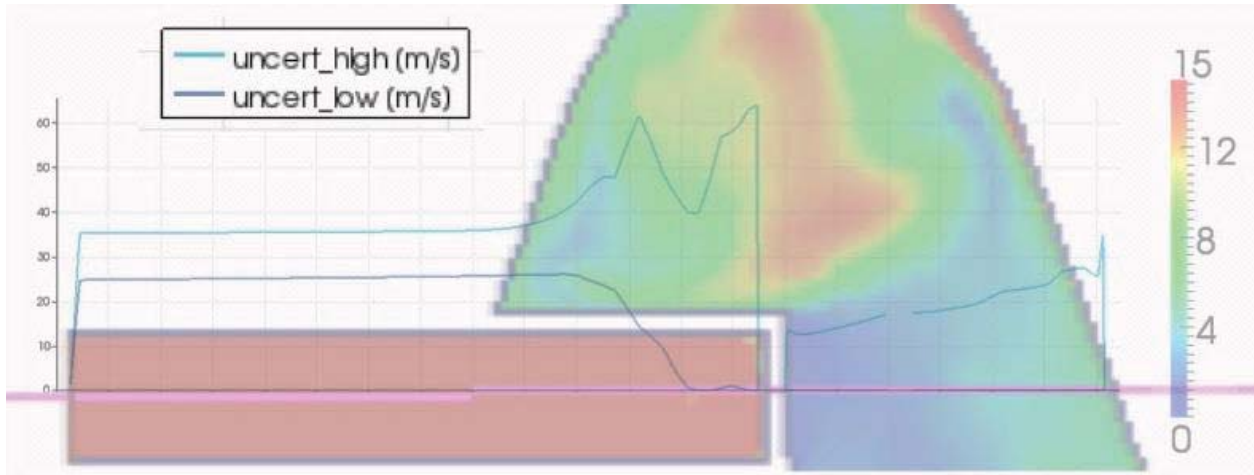
Figure 82: CFD Results and Uncertainty for 5.5m Configuration – Line Plot (m/s)
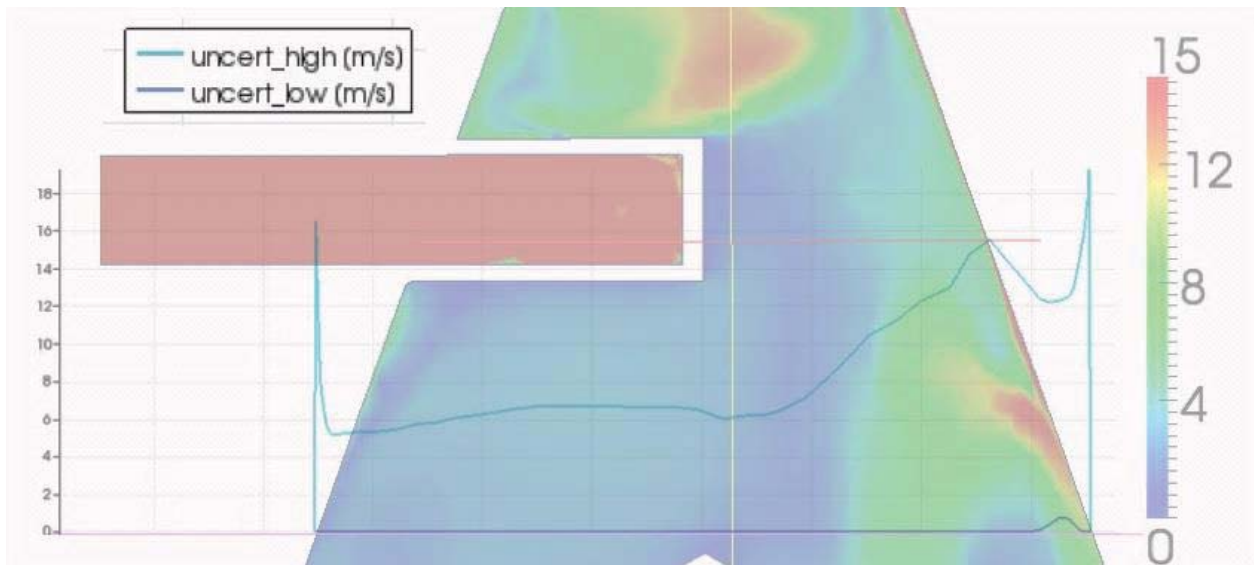
Figure 83: CFD Results and Uncertainty for 5.5m Configuration – Line Plot (m/s)
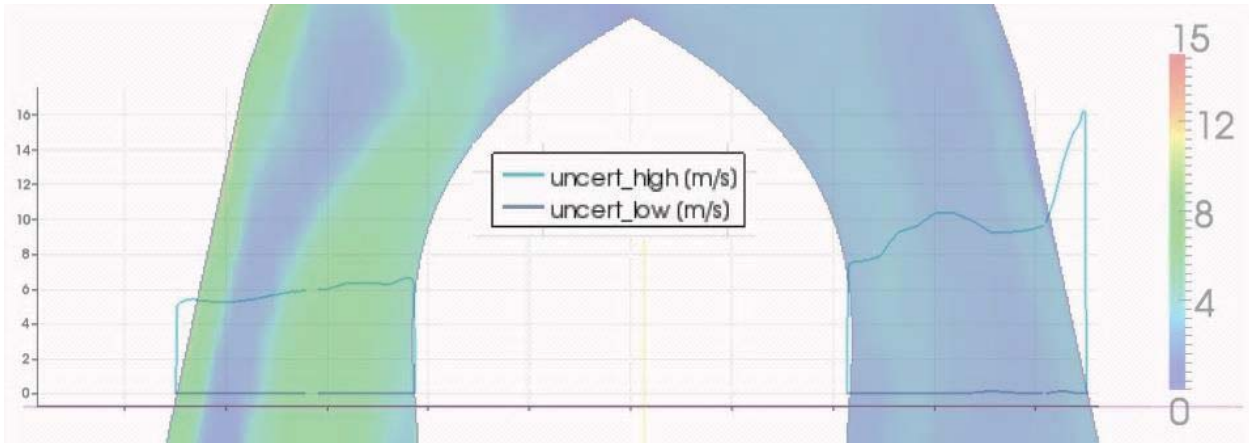
Figure 84: CFD Results and Uncertainty for 5.5m Configuration – Line Plot (m/s)
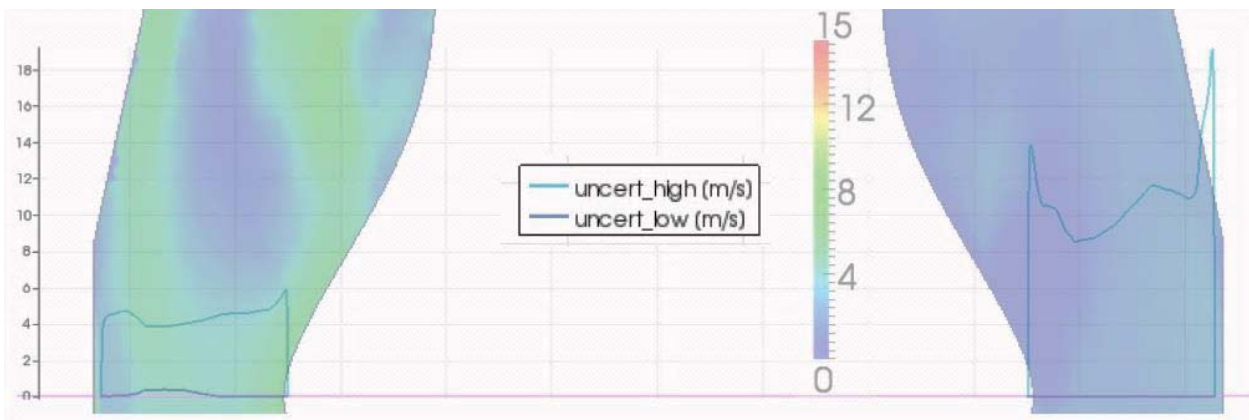
Figure 85: CFD Results and Uncertainty for 5.5m Configuration – Line Plot (m/s)

6.3.3   Ranking

The uncertainty for each of the input variables were ranked by the non-dimensionalizing the difference in the results by the freestream value and ranking from greatest uncertainty to least uncertainty. The results are provided in Table 15.

Table 15: 5.5m Configuration – Non-Dimensional Ranking of Uncertainty

Input Variable	Description	Bias	Mean Velocity Uncertainty (m/s)	Mean Non-Dimensionalized Uncertainty	Normalized Ranking %	Numbered Ranking
Grid	3 grids considered		2.0203	0.0673	12.44	3
Inlet Velocity	Boundary Condition	10%	1.6198	0.054	9.98	6
Outlet Pressure	Boundary Condition	2%	2.0173	0.0672	12.42	4
Turbulence Model	SA, ke-realizable, kwSST		2.3049	0.0768	14.19	1
Wall Functions	with and without		1.4902	0.0497	9.18	7
Rough Wall Function	smooth vs. rough		1.4901	0.0497	9.18	8
Compressibility	incompressible vs. compressible		1.4256	0.0475	8.78	9
Solver	OpenFoam, Fluent, STARCCM+		1.8172	0.0606	11.20	5
Fluid Properties	kinematic viscosity nu represents air [0-50-100] deg C	1.36,1.5,2.306e-05	2.05	0.0683	12.62	2

6.3.4   Discussion

The top five most significant input variables in regard to the uncertainty calculation were turbulence model, fluid properties, grid, outlet pressure boundary condition, and solver. This is a

different trend than the 0.75m or 3.5m configurations, however the numerical values on the same order of magnitude.

6.4   ECS System Experimental Comparison

The problem shown in references 15, 16, and 18 is the only published result prior to this research of the ECS airflow problem. The author's were able to receive the CAD models and test data from reference 15, 16, and 18. CFD models were constructed from the provided CAD models via Pro/ENGINEER, SnappyHex Mesh, and OpenFOAM. The CFD results of the proposed uncertainty methodology were compared to the test data 'blind'. The variables considered in the uncertainty analysis were inlet velocity, viscosity, outlet pressure, and turbulence. The inlet velocity and outlet pressure were assumed to have a bias error of 3 percent. The kinematic viscosity used was varied from 1.36e-5 to 2.306e-5 to account for all temperature ranges between 0 and 100 degrees Celsius. The turbulence models considered were Spalart Allmaras, ke-realizable, and kwsst. The confidence level was assumed to be 99 percent and a k factor of 3.335 was used. The results are presented for the measurement locations in reference 15 and 16. The Uy and Uz velocity components were compared at three locations of constant x and four locations of constant z (x=7in, 8.5 in, and 9in) as shown in Figure 86 and (z=-5 in, -3in, -2in, and 0 in) as shown in Figure 87.
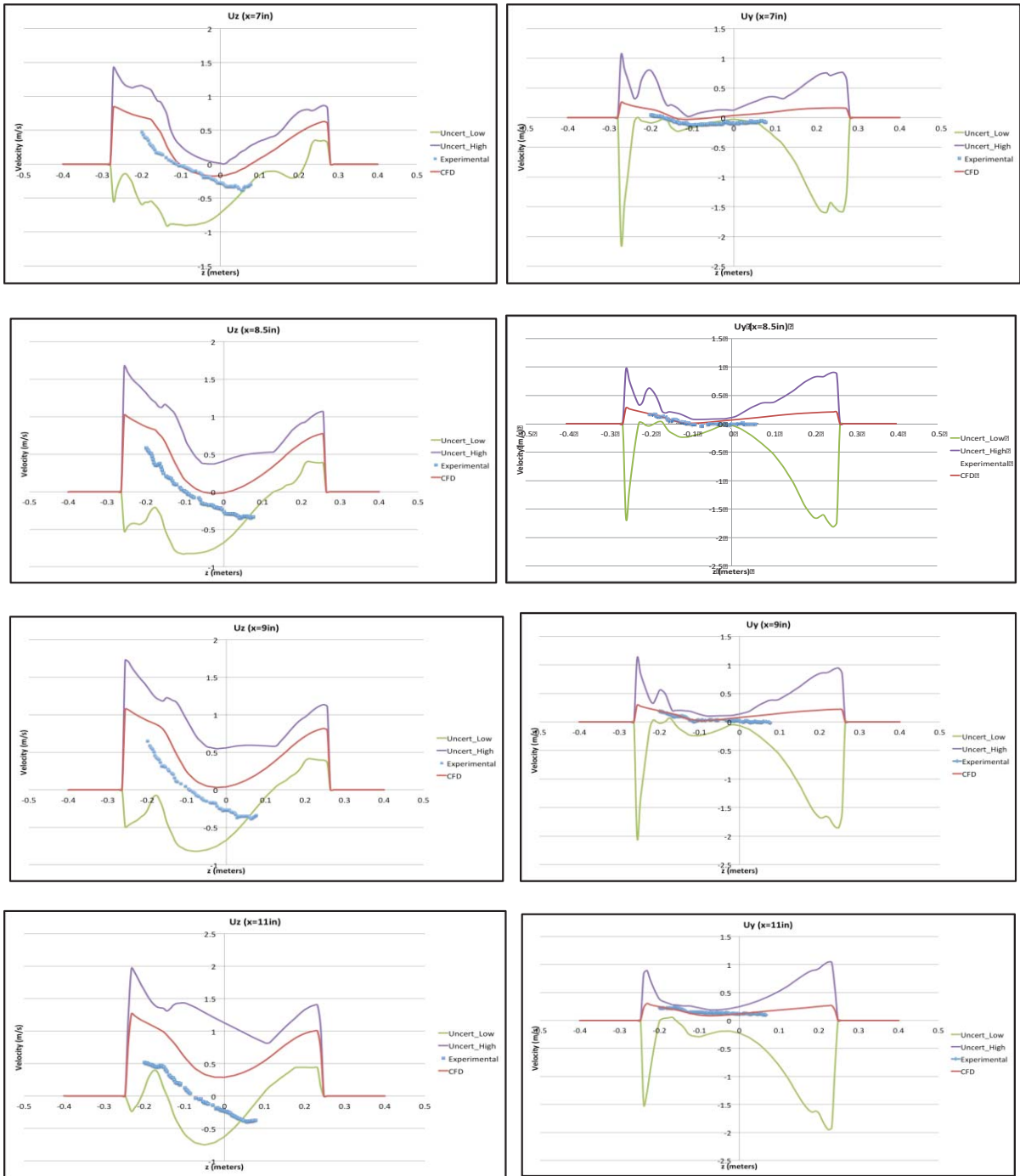
Figure 86: Proposed CFD Uncertainty Method vs. Experimental Data constant x
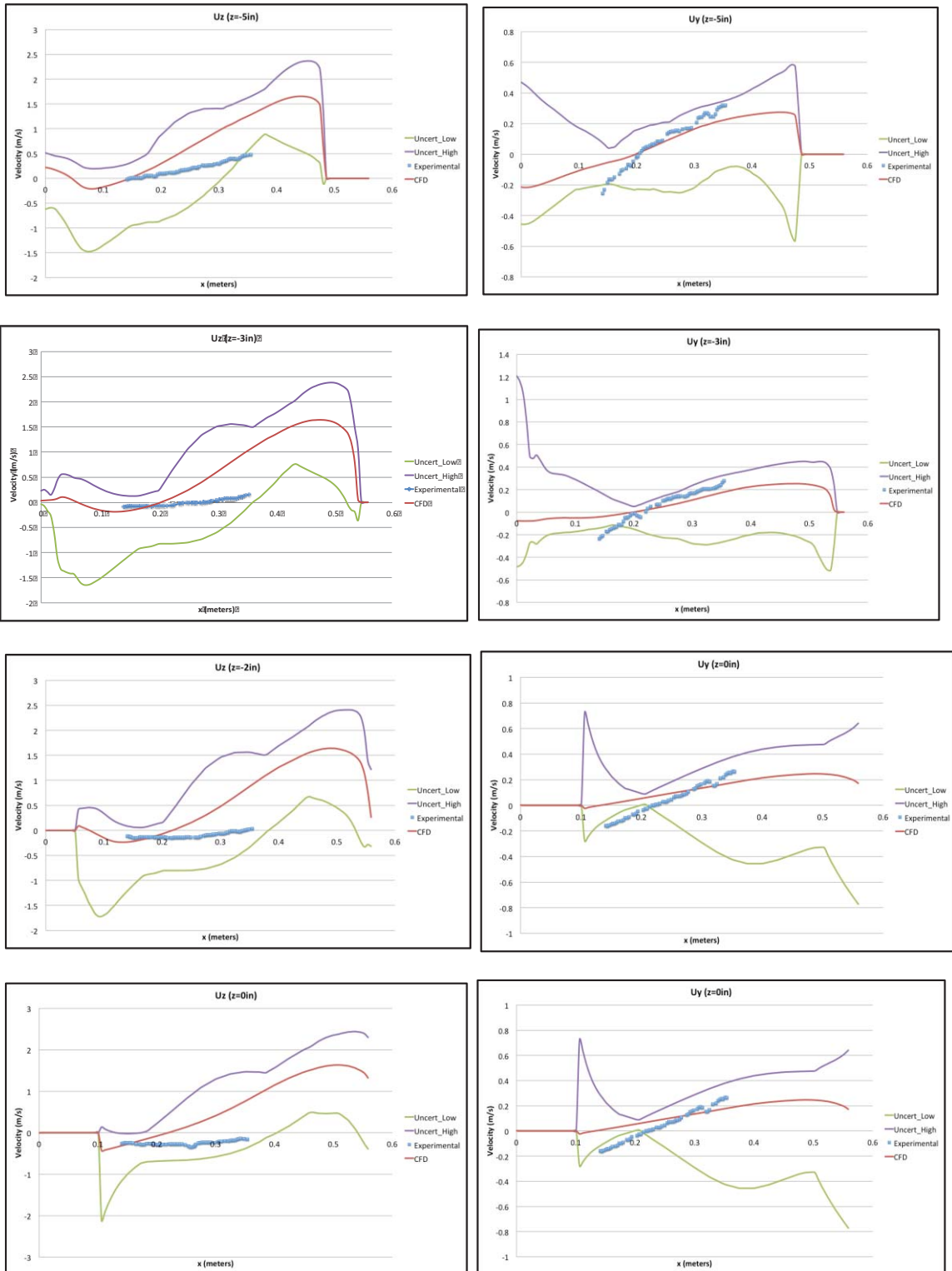
Figure 87: Proposed CFD Uncertainty Method vs. Experimental Data constant z

The proposed methodology for the limited set of test data encompassed most of the results. There were a total of 1,085 points measured and 977 of those were inside the 90% CFD Uncertainty methodology. This further validates the proposed methodology. Nighty percent of the experimental data lies within the 90% proposed CFD methodology. The CFD model and testing apparatus is shown in Figure 88.
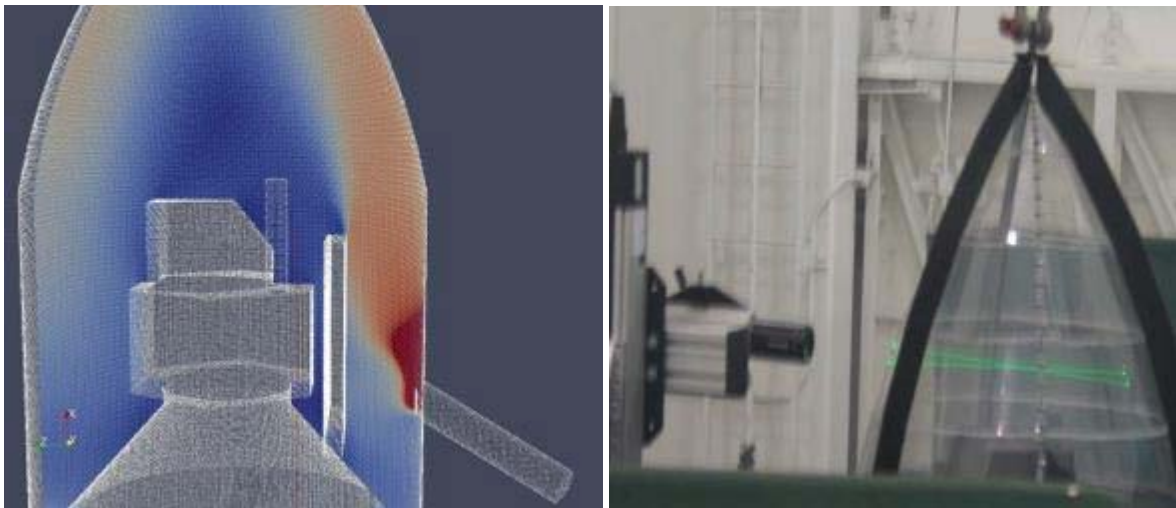
Figure 88: CFD Model and Experimental Apparatus [24]

This section provided uncertainty estimates for the three generic configurations, demonstrated the methodology, and proved the approach as compared to experimental data.

CHAPTER SEVEN: CONCLUSIONS

Industry is using Computational Fluid Dynamics to predict airflow fields without proper uncertainty calculations or validations. The purpose of this research is to propose a method to estimate the uncertainty in a Computational Fluid Dynamics (CFD) model using only numerical simulations and without any test data. This method is not designed to eliminate the need for testing or proper validation. The method strictly provides an uncertainty estimate when test data and validation are not obtainable.

The proposed method draws from industry standards in experimental and numerical model validation and couples the methods to provide a calculation that parallels traditional uncertainty predictions. There were two items learned from performing the uncertainty calculation of flow over a backward facing step. The first, Richardson's extrapolation method cannot be used for the entire computational domain since the flow velocity is not always monotonically increasing or decreasing. Two, when interpolating between grids, a 'linear' interpolation method must be used.

The following two simple examples were considered: the exact solution of laminar flow between parallel plates and experimentally derived turbulent heat transfer coefficients. The proposed methodology encompassed the uncertainty prediction from the 'traditional' way of performing these calculations using only CFD models for both cases considered. Additionally, it was shown that interpolating all solutions onto the coarse grid can provided higher order of accuracy and less error in the extrapolated solution.

The problem of interest is environmental control systems that have spacecraft impingement requirements. CFD can be used to verify these requirements, however proper validation and experimental data should be used to anchor the CFD predictions. If the data is not available or obtainable, the proposed methodology couples the Student-T distribution to the number of CFD models and input parameters to provide a realistic uncertainty prediction. The cases considered encompass the EELV fleet, however the data has not been experimentally verified. Future work should consider performing an experiment of one of the configurations considered and compare the experimental data to the proposed methodology. Additionally, the flow regime considered in this paper was internal, low speed, steady, incompressible air. The method has not been verified for other flow regimes that are dominated by compressibility or any other factor. Future work should consider extending the proposed methodology to external flows, compressible flows, unsteady flows, and other disciplines. It is believed the method could be applied to any discipline or numerical simulation.

It was hypothesized that for the ECS / spacecraft configurations that certain input variables would be negligible in the overall uncertainty calculation and through an order of magnitude analysis may be neglected. This hypothesis was proved false for the following input parameters: Grid, Inlet Velocity, Outlet Pressure, Turbulence Model, Wall Functions, Compressibility, Solver, and Fluid Properties. Each of these inputs should be considered and have the same order of magnitude importance to the overall uncertainty estimate.

Based off of all the models and data considered, it is recommended that all the input variables in Table 11 and all seventeen cases in Table 12 be considered along with the corresponding k factor

in Table 1 to provide an uncertainty estimate for CFD when proper testing and validation cannot

be performed.

APPENDIX A: INTERPOPENFOAM.M

```matlab
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%Curtis Groves
%%Interpolation For OPENFOAM
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%import OPENFOAM Data %%
%coarse
xc = importdata('ccx',',',22);
yc = importdata('ccy',',',22);
zc = importdata('ccz',',',22);
v = importdata('magU',',',22);
%fine
xf = importdata('ccxf',',',22);
yf = importdata('ccyf',',',22);
zf = importdata('cczf',',',22);


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%Other forms can also be read into Matlab using the following
%fluent = importdata('fluent.data',',',1);
%star=csvread('star.csv');
%VCI = griddata(fluent.data(:,2),fluent.data(:,3),fluent.data(:,4),fluent.data(:,5),xc.data,yc.data,zc.data,'natural')
%x=csvread('x.csv');
%y=csvread('y.csv');
%z=csvread('z.csv');
%v=csvread('V.csv');
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%Perform Interpolation using one of the following%%
%VCI = griddata3(x,y,z,v,xc.data,yc.data,zc.data,'linear');
%VCI = griddata3(xf.data,yf.data,zf.data,v.data,xc.data,yc.data,zc.data,'linear');
%VCI = griddata3(fluent(:,2),fluent(:,3),fluent(:,4),fluent(:,1),xc.data,yc.data,zc.data,'linear');
%VCI = griddata3(star(:,2),star(:,3),star(:,4),star(:,1),xc.data,yc.data,zc.data,'linear');
VCI = griddata3(xc.data,yc.data,zc.data,v.data,xf.data,yf.data,zf.data,'linear');
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%fix nans
VCI1 = inpaint_nans(VCI);
```

117

```
%write the file in OPENFOAM format
fid = fopen('one_coarsegrid', 'w');
fprintf(fid,'%s \n','/*--------------------------------*- C++ -*----------------------------------*\');
fprintf(fid,'%s \n','| =========                 |');
fprintf(fid,'%s \n','| \\      /  F ield          OpenFOAM: The Open Source CFD Toolbox  |');
fprintf(fid,'%s \n','|  \\    /   O peration       Version:  2.1.1                        |');
fprintf(fid,'%s \n','|   \\  /    A nd             Web:      www.OpenFOAM.org             |');
fprintf(fid,'%s \n','|    \\/     M anipulation    |');
fprintf(fid,'%s \n','\*--------------------------------------------------------------------------*/');
fprintf(fid,'%s \n','FoamFile');
fprintf(fid,'%s \n','{');
fprintf(fid,'%s \n','    version     2.0;');
fprintf(fid,'%s \n','    format      ascii;');
fprintf(fid,'%s \n','    class       volScalarField;');
fprintf(fid,'%s \n','    location    "6100";');
fprintf(fid,'%s \n','    object      one_coarsegrid;');
fprintf(fid,'%s \n','}');
fprintf(fid,'%s \n','// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //');
fprintf(fid,'%s \n','');
fprintf(fid,'%s \n','dimensions      [0 1 -1 0 0 0 0];');
fprintf(fid,'%s \n','');
fprintf(fid,'%s \n','internalField   nonuniform List<scalar>');
fprintf(fid,'%d \n',length(VCI1)');
fprintf(fid,'%s \n','(');

% write data
lng=length(VCI1);

for i=1:lng
fprintf(fid,'%d \n',VCI1(i,1));
i=i+1;
end

fprintf(fid,'%s \n',')');
fprintf(fid,'%s \n',';');
fprintf(fid,'%s \n','');
fprintf(fid,'%s \n','boundaryField');
fprintf(fid,'%s \n','{');
fprintf(fid,'%s \n','    outlet');
fprintf(fid,'%s \n','    { ');
fprintf(fid,'%s \n','        type            calculated;');
fprintf(fid,'%s \n','        value           uniform 0;');
fprintf(fid,'%s \n','    }');
fprintf(fid,'%s \n','    inlet');
fprintf(fid,'%s \n','    { ');
fprintf(fid,'%s \n','        type            calculated;');
fprintf(fid,'%s \n','        value           uniform 0;');
fprintf(fid,'%s \n','    }');
fprintf(fid,'%s \n','    wall');
fprintf(fid,'%s \n','    { ');
fprintf(fid,'%s \n','        type            calculated;');
fprintf(fid,'%s \n','        value           uniform 0;');
fprintf(fid,'%s \n','    }');
fprintf(fid,'%s \n','}');
fprintf(fid,'%s \n','');
fprintf(fid,'%s \n','');
fprintf(fid,'%s \n','// *************************************************************************');

fclose(fid)

disp ('wrote OpenFoam file Results');
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

APPENDIX B: UNCERTOPENFOAM.M

```matlab
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%Curtis Groves
%%Uncertainty Analysis For OPENFOAM
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%Import OPENFOAM Data %%%%%%%%%%%%%
one_coarsegrid = importdata('one_coarsegrid',',',22);
two_medgrid = importdata('two_medgrid',',',22);
three_finegrid = importdata('three_finegrid',',',22);
four_ulow = importdata('four_ulow',',',22);
five_uhigh = importdata('five_uhigh',',',22);
six_plow = importdata('six_plow',',',22);
seven_phigh = importdata('seven_phigh',',',22);
eight_sa = importdata('eight_sa',',',22);
nine_ke = importdata('nine_ke',',',22);
ten_kwsst = importdata('ten_kwsst',',',22);
eleven_nowallfns = importdata('eleven_nowallfns',',',22);
twelve_roughwall = importdata('twelve_roughwall',',',22);
thirteen_compr = importdata('thirteen_compr',',',22);
fourteen_fluent = importdata('fourteen_fluent',',',22);
fifteen_star = importdata('fifteen_star',',',22);
sixteen_nuthigh = importdata('sixteen_nuthigh',',',22);
seventeen_nutlow = importdata('seventeen_nutlow',',',22);

save

% load matlab.mat
%
%%%create a matrix with all data
data(:,1) = one_coarsegrid.data(:,1);
data(:,2) = two_medgrid.data(:,1);
data(:,3) = three_finegrid.data(:,1);
data(:,4) = four_ulow.data(:,1);
data(:,5) = five_uhigh.data(:,1);
data(:,6) = six_plow.data(:,1);
data(:,7) = seven_phigh.data(:,1);
data(:,8) = eight_sa.data(:,1);
data(:,9) = nine_ke.data(:,1);
data(:,10) = ten_kwsst.data(:,1);
data(:,11) = eleven_nowallfns.data(:,1);
data(:,12) = twelve_roughwall.data(:,1);
data(:,13) = thirteen_compr.data(:,1);
data(:,14) = fourteen_fluent.data(:,1);
data(:,15) = fifteen_star.data(:,1);
data(:,16) = sixteen_nuthigh.data(:,1);
data(:,17) = seventeen_nutlow.data(:,1);
%%%
```

Appendix B.1   Uncertopenfoam.M –Max,Min,Uncert

```matlab
%Calculate Maximium
lng=length(data);

for i=1:lng
    maximium(i)=max(data(i,:));
end

maximium=maximium';
%%%%%%%%%

%Calculate Minimium
for i=1:lng
    minimum(i)=min(data(i,:));
end

minimum=minimum';
%%%%%%%%%


%Calculate Uncertainty
%There are seventeen Cases therfore using a k factor of k=1.74
k = 1.74;
uncert = k*(maximium - minimum)/2;
middle = (maximium - minimum)/2 + minimum;
uncert_high = middle + uncert;
uncert_low = middle - uncert;
uncert_low(find(uncert_low<0))=0;
%%%
```

Appendix B.2 Uncertopenfoam.M – Write Min

```matlab
%write minimum in OPENFOAM format
fid = fopen('min', 'w');
fprintf(fid,'%s \n','/*--------------------------------*- C++ -*----------------------------------*\');
fprintf(fid,'%s \n','| =========                 |');
fprintf(fid,'%s \n','| \\      /  F ield           | OpenFOAM: The Open Source CFD Toolbox           |');
fprintf(fid,'%s \n','|  \\    /   O peration        | Version:  2.1.1                                 |');
fprintf(fid,'%s \n','|   \\  /    A nd              | Web:      www.OpenFOAM.org                       |');
fprintf(fid,'%s \n','|    \\/     M anipulation     |                                                 |');
fprintf(fid,'%s \n','\*---------------------------------------------------------------------------*/');
fprintf(fid,'%s \n','FoamFile');
fprintf(fid,'%s \n','{');
fprintf(fid,'%s \n','    version     2.0;');
fprintf(fid,'%s \n','    format      ascii;');
fprintf(fid,'%s \n','    class       volScalarField;');
fprintf(fid,'%s \n','    location    "6100";');
fprintf(fid,'%s \n','    object      min;');
fprintf(fid,'%s \n','}');
fprintf(fid,'%s \n','// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //');
fprintf(fid,'%s \n','');
fprintf(fid,'%s \n','dimensions      [0 1 -1 0 0 0 0];');
fprintf(fid,'%s \n','');
fprintf(fid,'%s \n','internalField   nonuniform List<scalar>');
fprintf(fid,'%d \n',length(minimum)');
fprintf(fid,'%s \n','(');

% write data
lng=length(minimum);

for i=1:lng
fprintf(fid,'%d \n',minimum(i,1));
i=i+1;
end

fprintf(fid,'%s \n',')');
fprintf(fid,'%s \n',';');
fprintf(fid,'%s \n','');
fprintf(fid,'%s \n','boundaryField');
fprintf(fid,'%s \n','{');
fprintf(fid,'%s \n','    outlet');
fprintf(fid,'%s \n','    { ');
fprintf(fid,'%s \n','        type            calculated;');
fprintf(fid,'%s \n','        value           uniform 0;');
fprintf(fid,'%s \n','    }');
fprintf(fid,'%s \n','    inlet');
fprintf(fid,'%s \n','    { ');
fprintf(fid,'%s \n','        type            calculated;');
fprintf(fid,'%s \n','        value           uniform 0;');
fprintf(fid,'%s \n','    }');
fprintf(fid,'%s \n','    wall');
fprintf(fid,'%s \n','    { ');
fprintf(fid,'%s \n','        type            calculated;');
fprintf(fid,'%s \n','        value           uniform 0;');
fprintf(fid,'%s \n','    }');
fprintf(fid,'%s \n','}');
fprintf(fid,'%s \n','');
fprintf(fid,'%s \n','');
fprintf(fid,'%s \n','// *************************************************************************');

fclose(fid)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Appendix B.3 Unceropenfoam.M – Write Max

```matlab
%write maximum in OPENFOAM format
fid = fopen('max', 'w');
fprintf(fid,'%s \n','/*--------------------------------*- C++ -*----------------------------------*\');
fprintf(fid,'%s \n','| =========                 |');
fprintf(fid,'%s \n','|  \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox |');
fprintf(fid,'%s \n','|   \\    /   O peration     | Version:  2.1.1                       |');
fprintf(fid,'%s \n','|    \\  /    A nd           | Web:      www.OpenFOAM.org            |');
fprintf(fid,'%s \n','|     \\/     M anipulation  |                                       |');
fprintf(fid,'%s \n','\*---------------------------------------------------------------------------*/');
fprintf(fid,'%s \n','FoamFile');
fprintf(fid,'%s \n','{');
fprintf(fid,'%s \n','    version     2.0;');
fprintf(fid,'%s \n','    format      ascii;');
fprintf(fid,'%s \n','    class       volScalarField;');
fprintf(fid,'%s \n','    location    "6100";');
fprintf(fid,'%s \n','    object      max;');
fprintf(fid,'%s \n','}');
fprintf(fid,'%s \n','// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //');
fprintf(fid,'%s \n','');
fprintf(fid,'%s \n','dimensions      [0 1 -1 0 0 0 0];');
fprintf(fid,'%s \n','');
fprintf(fid,'%s \n','internalField   nonuniform List<scalar>');
fprintf(fid,'%d \n',length(maximum)');
fprintf(fid,'%s \n','(');

% write data
lng=length(maximium);

for i=1:lng
fprintf(fid,'%d \n',maximium(i,1));
i=i+1;
end

fprintf(fid,'%s \n',')');
fprintf(fid,'%s \n',';');
fprintf(fid,'%s \n','');
fprintf(fid,'%s \n','boundaryField');
fprintf(fid,'%s \n','{');
fprintf(fid,'%s \n','    outlet');
fprintf(fid,'%s \n','    { ');
fprintf(fid,'%s \n','        type            calculated;');
fprintf(fid,'%s \n','        value           uniform 0;');
fprintf(fid,'%s \n','    }');
fprintf(fid,'%s \n','    inlet');
fprintf(fid,'%s \n','    { ');
fprintf(fid,'%s \n','        type            calculated;');
fprintf(fid,'%s \n','        value           uniform 0;');
fprintf(fid,'%s \n','    }');
fprintf(fid,'%s \n','    wall');
fprintf(fid,'%s \n','    { ');
fprintf(fid,'%s \n','        type            calculated;');
fprintf(fid,'%s \n','        value           uniform 0;');
fprintf(fid,'%s \n','    }');
fprintf(fid,'%s \n','}');
fprintf(fid,'%s \n','');
fprintf(fid,'%s \n','');
fprintf(fid,'%s \n','// ************************************************************************* ');

fclose(fid)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Appendix B.4 Uncertopenfoam.M –Write Middle

```matlab
%write middle in OPENFOAM format
fid = fopen('mid', 'w');
fprintf(fid,'%s \n','/*--------------------------------*- C++ -*----------------------------------*\');
fprintf(fid,'%s \n','| =========                 |                                                 |');
fprintf(fid,'%s \n','| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |');
fprintf(fid,'%s \n','|  \\    /   O peration     | Version:  2.1.1                                 |');
fprintf(fid,'%s \n','|   \\  /    A nd           | Web:      www.OpenFOAM.org                       |');
fprintf(fid,'%s \n','|    \\/     M anipulation  |                                                 |');
fprintf(fid,'%s \n','\*---------------------------------------------------------------------------*/');
fprintf(fid,'%s \n','FoamFile');
fprintf(fid,'%s \n','{');
fprintf(fid,'%s \n','    version     2.0;');
fprintf(fid,'%s \n','    format      ascii;');
fprintf(fid,'%s \n','    class       volScalarField;');
fprintf(fid,'%s \n','    location    "6100";');
fprintf(fid,'%s \n','    object      mid;');
fprintf(fid,'%s \n','}');
fprintf(fid,'%s \n','// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //');
fprintf(fid,'%s \n','');
fprintf(fid,'%s \n','dimensions      [0 1 -1 0 0 0 0];');
fprintf(fid,'%s \n','');
fprintf(fid,'%s \n','internalField   nonuniform List<scalar>');
fprintf(fid,'%d \n',length(middle));
fprintf(fid,'%s \n','(');

% write data
lng=length(middle);

for i=1:lng
fprintf(fid,'%d \n',middle(i,1));
i=i+1;
end

fprintf(fid,'%s \n',')');
fprintf(fid,'%s \n',';');
fprintf(fid,'%s \n','');
fprintf(fid,'%s \n','boundaryField');
fprintf(fid,'%s \n','{');
fprintf(fid,'%s \n','    outlet');
fprintf(fid,'%s \n','    { ');
fprintf(fid,'%s \n','        type            calculated;');
fprintf(fid,'%s \n','        value           uniform 0;');
fprintf(fid,'%s \n','    }');
fprintf(fid,'%s \n','    inlet');
fprintf(fid,'%s \n','    { ');
fprintf(fid,'%s \n','        type            calculated;');
fprintf(fid,'%s \n','        value           uniform 0;');
fprintf(fid,'%s \n','    }');
fprintf(fid,'%s \n','    wall');
fprintf(fid,'%s \n','    { ');
fprintf(fid,'%s \n','        type            calculated;');
fprintf(fid,'%s \n','        value           uniform 0;');
fprintf(fid,'%s \n','    }');
fprintf(fid,'%s \n','}');
fprintf(fid,'%s \n','');
fprintf(fid,'%s \n','');
fprintf(fid,'%s \n','// *************************************************************************');

fclose(fid)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Appendix B.5 Uncertopenfoam.M – Write Uncertainty

```matlab
%write uncert in OPENFOAM format
fid = fopen('uncert', 'w');
fprintf(fid,'%s \n','/*--------------------------------*- C++ -*----------------------------------*\');
fprintf(fid,'%s \n','| =========                 |');
fprintf(fid,'%s \n','| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox      |');
fprintf(fid,'%s \n','|  \\    /   O peration      | Version:  2.1.1                            |');
fprintf(fid,'%s \n','|   \\  /    A nd            | Web:      www.OpenFOAM.org                 |');
fprintf(fid,'%s \n','|    \\/     M anipulation   |                                            |');
fprintf(fid,'%s \n','\*--------------------------------------------------------------------------*/');
fprintf(fid,'%s \n','FoamFile');
fprintf(fid,'%s \n','{');
fprintf(fid,'%s \n','    version     2.0;');
fprintf(fid,'%s \n','    format      ascii;');
fprintf(fid,'%s \n','    class       volScalarField;');
fprintf(fid,'%s \n','    location    "6100";');
fprintf(fid,'%s \n','    object      uncert;');
fprintf(fid,'%s \n','}');
fprintf(fid,'%s \n','// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //');
fprintf(fid,'%s \n','');
fprintf(fid,'%s \n','dimensions      [0 1 -1 0 0 0 0];');
fprintf(fid,'%s \n','');
fprintf(fid,'%s \n','internalField   nonuniform List<scalar>');
fprintf(fid,'%d \n',length(uncert));
fprintf(fid,'%s \n','(');

% write data
lng=length(uncert);

for i=1:lng
fprintf(fid,'%d \n',uncert(i,1));
i=i+1;
end

fprintf(fid,'%s \n',')');
fprintf(fid,'%s \n',';');
fprintf(fid,'%s \n','');
fprintf(fid,'%s \n','boundaryField');
fprintf(fid,'%s \n','{');
fprintf(fid,'%s \n','    outlet');
fprintf(fid,'%s \n','    { ');
fprintf(fid,'%s \n','        type            calculated;');
fprintf(fid,'%s \n','        value           uniform 0;');
fprintf(fid,'%s \n','    }');
fprintf(fid,'%s \n','    inlet');
fprintf(fid,'%s \n','    { ');
fprintf(fid,'%s \n','        type            calculated;');
fprintf(fid,'%s \n','        value           uniform 0;');
fprintf(fid,'%s \n','    }');
fprintf(fid,'%s \n','    wall');
fprintf(fid,'%s \n','    { ');
fprintf(fid,'%s \n','        type            calculated;');
fprintf(fid,'%s \n','        value           uniform 0;');
fprintf(fid,'%s \n','    }');
fprintf(fid,'%s \n','}');
fprintf(fid,'%s \n','');
fprintf(fid,'%s \n','');
fprintf(fid,'%s \n','// ****************************************************************** ');

fclose(fid)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Appendix B.6 Uncertopenfoam.M – Write Uncertainty Low

```matlab
%write uncert low in OPENFOAM format
fid = fopen('uncert_low', 'w');
fprintf(fid,'%s \n','/*--------------------------------*- C++ -*----------------------------------*\');
fprintf(fid,'%s \n','| =========                 |');
fprintf(fid,'%s \n','| \\      / F ield         OpenFOAM: The Open Source CFD Toolbox           |');
fprintf(fid,'%s \n','|  \\    /  O peration     Version:  2.1.1                                 |');
fprintf(fid,'%s \n','|   \\  /   A nd           Web:      www.OpenFOAM.org                      |');
fprintf(fid,'%s \n','|    \\/    M anipulation  |');
fprintf(fid,'%s \n','\*---------------------------------------------------------------------------*/');
fprintf(fid,'%s \n','FoamFile');
fprintf(fid,'%s \n','{');
fprintf(fid,'%s \n','    version     2.0;');
fprintf(fid,'%s \n','    format      ascii;');
fprintf(fid,'%s \n','    class       volScalarField;');
fprintf(fid,'%s \n','    location    "6100";');
fprintf(fid,'%s \n','    object      uncert_low;');
fprintf(fid,'%s \n','}');
fprintf(fid,'%s \n','// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //');
fprintf(fid,'%s \n','');
fprintf(fid,'%s \n','dimensions      [0 1 -1 0 0 0 0];');
fprintf(fid,'%s \n','');
fprintf(fid,'%s \n','internalField   nonuniform List<scalar>');
fprintf(fid,'%d \n',length(uncert_low)');
fprintf(fid,'%s \n','(');

write data
lng=length(uncert_low);

for i=1:lng
fprintf(fid,'%d \n',uncert_low(i,1));
i=i+1;
end

fprintf(fid,'%s \n',')');
fprintf(fid,'%s \n',';');
fprintf(fid,'%s \n','');
fprintf(fid,'%s \n','boundaryField');
fprintf(fid,'%s \n','{');
fprintf(fid,'%s \n','    outlet');
fprintf(fid,'%s \n','    { ');
fprintf(fid,'%s \n','        type            calculated;');
fprintf(fid,'%s \n','        value           uniform 0;');
fprintf(fid,'%s \n','    }');
fprintf(fid,'%s \n','    inlet');
fprintf(fid,'%s \n','    { ');
fprintf(fid,'%s \n','        type            calculated;');
fprintf(fid,'%s \n','        value           uniform 0;');
fprintf(fid,'%s \n','    }');
fprintf(fid,'%s \n','    wall');
fprintf(fid,'%s \n','    { ');
fprintf(fid,'%s \n','        type            calculated;');
fprintf(fid,'%s \n','        value           uniform 0;');
fprintf(fid,'%s \n','    }');
fprintf(fid,'%s \n','}');
fprintf(fid,'%s \n','');
fprintf(fid,'%s \n','');
fprintf(fid,'%s \n','// *************************************************************************');

fclose(fid)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Appendix B.7 Uncertopenfoam.M – Write Uncertainty High

```matlab
%write uncert high in OPENFOAM format
fid = fopen('uncert_high', 'w');
fprintf(fid,'%s \n','/*--------------------------------*- C++ -*----------------------------------*\');
fprintf(fid,'%s \n','| =========                 |                                                 |');
fprintf(fid,'%s \n','| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |');
fprintf(fid,'%s \n','|  \\    /   O peration     | Version:  2.1.1                                 |');
fprintf(fid,'%s \n','|   \\  /    A nd           | Web:      www.OpenFOAM.org                       |');
fprintf(fid,'%s \n','|    \\/     M anipulation  |                                                 |');
fprintf(fid,'%s \n','\*---------------------------------------------------------------------------*/');
fprintf(fid,'%s \n','FoamFile');
fprintf(fid,'%s \n','{');
fprintf(fid,'%s \n','    version     2.0;');
fprintf(fid,'%s \n','    format      ascii;');
fprintf(fid,'%s \n','    class       volScalarField;');
fprintf(fid,'%s \n','    location    "6100";');
fprintf(fid,'%s \n','    object      uncert_high;');
fprintf(fid,'%s \n','}');
fprintf(fid,'%s \n','// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //');
fprintf(fid,'%s \n','');
fprintf(fid,'%s \n','dimensions      [0 1 -1 0 0 0 0];');
fprintf(fid,'%s \n','');
fprintf(fid,'%s \n','internalField   nonuniform List<scalar>');
fprintf(fid,'%d \n',length(uncert_high));
fprintf(fid,'%s \n','(');

% write data
lng=length(uncert_high);

for i=1:lng
fprintf(fid,'%d \n',uncert_high(i,1));
i=i+1;
end

fprintf(fid,'%s \n',')');
fprintf(fid,'%s \n',';');
fprintf(fid,'%s \n','');
fprintf(fid,'%s \n','boundaryField');
fprintf(fid,'%s \n','{');
fprintf(fid,'%s \n','    outlet');
fprintf(fid,'%s \n','    { ');
fprintf(fid,'%s \n','        type            calculated;');
fprintf(fid,'%s \n','        value           uniform 0;');
fprintf(fid,'%s \n','    }');
fprintf(fid,'%s \n','    inlet');
fprintf(fid,'%s \n','    { ');
fprintf(fid,'%s \n','        type            calculated;');
fprintf(fid,'%s \n','        value           uniform 0;');
fprintf(fid,'%s \n','    }');
fprintf(fid,'%s \n','    wall');
fprintf(fid,'%s \n','    { ');
fprintf(fid,'%s \n','        type            calculated;');
fprintf(fid,'%s \n','        value           uniform 0;');
fprintf(fid,'%s \n','    }');
fprintf(fid,'%s \n','}');
fprintf(fid,'%s \n','');
fprintf(fid,'%s \n','');
fprintf(fid,'%s \n','// *************************************************************************');

fclose(fid)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

APPENDIX C: RKOPENFOAM.M – CALCULATE RK AND WRITE OPENFOAM

```
%%Compute Rk Values are write OPENFOAM RK File
load matlab.mat
e_21 = two_medgrid.data(:,1) - three_finegrid.data(:,1);
e_32 = one_coarsegrid.data(:,1) - two_medgrid.data(:,1);
Rk = e_21./e_32;
Rk(~isfinite(Rk))=0;

%write uncert low in OPENFOAM format
fid = fopen('Rk', 'w');
fprintf(fid,'%s \n','/*--------------------------------*- C++ -*----------------------------------*\');
fprintf(fid,'%s \n','| =========                 |');
fprintf(fid,'%s \n','| \\      /  F ield           OpenFOAM: The Open Source CFD Toolbox         |');
fprintf(fid,'%s \n','|  \\    /   O peration        Version:  2.1.1                              |');
fprintf(fid,'%s \n','|   \\  /    A nd              Web:      www.OpenFOAM.org                   |');
fprintf(fid,'%s \n','|    \\/     M anipulation                                                   |');
fprintf(fid,'%s \n','\*---------------------------------------------------------------------------*/');
fprintf(fid,'%s \n','FoamFile');
fprintf(fid,'%s \n','{');
fprintf(fid,'%s \n','    version     2.0;');
fprintf(fid,'%s \n','    format      ascii;');
fprintf(fid,'%s \n','    class       volScalarField;');
fprintf(fid,'%s \n','    location    "6100";');
fprintf(fid,'%s \n','    object      Rk;');
fprintf(fid,'%s \n','}');
fprintf(fid,'%s \n','// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //');
fprintf(fid,'%s \n','');
fprintf(fid,'%s \n','dimensions      [0 1 -1 0 0 0 0];');
fprintf(fid,'%s \n','');
fprintf(fid,'%s \n','internalField   nonuniform List<scalar>');
fprintf(fid,'%d \n',length(Rk)');
fprintf(fid,'%s \n','(');
% write data
lng=length(Rk);
for i=1:lng
fprintf(fid,'%d \n',Rk(i,1));
i=i+1;
end
fprintf(fid,'%s \n',')');
fprintf(fid,'%s \n',';');
fprintf(fid,'%s \n','');
fprintf(fid,'%s \n','boundaryField');
fprintf(fid,'%s \n','{');
fprintf(fid,'%s \n','    outlet');
fprintf(fid,'%s \n','    { ');
fprintf(fid,'%s \n','        type            calculated;');
fprintf(fid,'%s \n','        value           uniform 0;');
fprintf(fid,'%s \n','    }');
fprintf(fid,'%s \n','    inlet');
fprintf(fid,'%s \n','    { ');
fprintf(fid,'%s \n','        type            calculated;');
fprintf(fid,'%s \n','        value           uniform 0;');
fprintf(fid,'%s \n','    }');
fprintf(fid,'%s \n','    wall');
fprintf(fid,'%s \n','    { ');
fprintf(fid,'%s \n','        type            calculated;');
fprintf(fid,'%s \n','        value           uniform 0;');
fprintf(fid,'%s \n','    }');
fprintf(fid,'%s \n','}');
fprintf(fid,'%s \n','');
fprintf(fid,'%s \n','');
fprintf(fid,'%s \n','// *************************************************************************');

fclose(fid)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

APPENDIX D: NONDIMRANK.M – NON-DIMENSIONALIZATION AND RANKING OF INPUT UNCERTAINTY

```matlab
%Non-Dimensionalization and Ranking

%%%%%%%%%%%%%%%%%%%%%%%%%%USED FOR RANKING ONLY NON-DIM BY MAX VELOCITY
load matlab.mat

%30 is 1x the free-stream value
uinf=30;
%max_val=max(data);
%max_val=max(max_val);

%nondim = (max_val- data ) / max_val;
nondim=data;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%grid is 1,2,3
nondim_grid = [nondim(:,1), nondim(:,2), nondim(:,3)];

for i=1:length(nondim_grid)

max_nondim_grid(i) = max(nondim_grid(i,:));

end
max_nondim_grid=max_nondim_grid';


for i=1:length(nondim_grid)

min_nondim_grid(i) = min(nondim_grid(i,:));

end

min_nondim_grid = min_nondim_grid';

diff_nondim_grid = (max_nondim_grid - min_nondim_grid);

%histfit(diff_nondim_grid,2932,'exponential')
%grid on

%max_diff_dim_grid = max(diff_nondim_grid);
%$min_diff_dim_grid = min(diff_nondim_grid);
mean_diff_dim_grid = mean(diff_nondim_grid);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```matlab
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%inlet velocity
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%is 4 and 5
nondim_inletvelocity = [nondim(:,4), nondim(:,5)];

for i=1:length(nondim_inletvelocity)

max_nondim_inletvelocity(i) = max(nondim_inletvelocity(i,:));

end
max_nondim_inletvelocity=max_nondim_inletvelocity';

for i=1:length(nondim_inletvelocity)

min_nondim_inletvelocity(i) = min(nondim_inletvelocity(i,:));

end

min_nondim_inletvelocity = min_nondim_inletvelocity';

diff_nondim_inletvelocity = (max_nondim_inletvelocity - min_nondim_inletvelocity);

%histfit(diff_nondim_inletvelocity,2932,'exponential')
%grid on
%
%max_diff_dim_inletvelocity = max(diff_nondim_inletvelocity);
%min_diff_dim_inletvelocity = min(diff_nondim_inletvelocity);
mean_diff_dim_inletvelocity = mean(diff_nondim_inletvelocity);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```matlab
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%turbulence
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%is 8 9 10
nondim_turb = [nondim(:,8), nondim(:,9), nondim(:,10)];

for i=1:length(nondim_turb)

max_nondim_turb(i) = max(nondim_turb(i,:));

end
max_nondim_turb=max_nondim_turb';


for i=1:length(nondim_turb)

min_nondim_turb(i) = min(nondim_turb(i,:));

end

min_nondim_turb = min_nondim_turb';

diff_nondim_turb = (max_nondim_turb - min_nondim_turb);

%histfit(diff_nondim_turb,2932,'exponential')
%grid on

%max_diff_dim_turb = max(diff_nondim_turb);
%min_diff_dim_turb = min(diff_nondim_turb);
mean_diff_dim_turb = mean(diff_nondim_turb);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```matlab
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%wall functions 1 vs 11
nondim_wallfns = [nondim(:,1), nondim(:,11)];

for i=1:length(nondim_wallfns)

max_nondim_wallfns(i) = max(nondim_wallfns(i,:));

end
max_nondim_wallfns=max_nondim_wallfns';


for i=1:length(nondim_wallfns)

min_nondim_wallfns(i) = min(nondim_wallfns(i,:));

end

min_nondim_wallfns = min_nondim_wallfns';

diff_nondim_wallfns = (max_nondim_wallfns - min_nondim_wallfns);

%histfit(diff_nondim_turb,2932,'exponential')
%grid on

%max_diff_dim_wallfns = max(diff_nondim_wallfns);
%min_diff_dim_wallfns = min(diff_nondim_wallfns);
mean_diff_dim_wallfns = mean(diff_nondim_wallfns);
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```matlab
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%surface roughness 1 vs 12
nondim_roughwall = [nondim(:,1), nondim(:,12)];

for i=1:length(nondim_roughwall)

max_nondim_roughwall(i) = max(nondim_roughwall(i,:));

end
max_nondim_roughwall=max_nondim_roughwall';


for i=1:length(nondim_roughwall)

min_nondim_roughwall(i) = min(nondim_roughwall(i,:));

end

min_nondim_roughwall = min_nondim_roughwall';

diff_nondim_roughwall = (max_nondim_roughwall - min_nondim_roughwall);

%histfit(diff_nondim_turb,2932,'exponential')
%grid on

%max_diff_dim_roughwall = max(diff_nondim_roughwall);
%min_diff_dim_roughwall = min(diff_nondim_roughwall);
mean_diff_dim_roughwall = mean(diff_nondim_roughwall);
% % %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%



%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%surface roughness 1 vs 13
nondim_compr = [nondim(:,1), nondim(:,13)];

for i=1:length(nondim_compr)

max_nondim_compr(i) = max(nondim_compr(i,:));

end
max_nondim_compr=max_nondim_compr';


for i=1:length(nondim_compr)

min_nondim_compr(i) = min(nondim_compr(i,:));

end

min_nondim_compr = min_nondim_compr';

diff_nondim_compr = (max_nondim_compr - min_nondim_compr);

%histfit(diff_nondim_turb,2932,'exponential')
%grid on

%max_diff_dim_compr = max(diff_nondim_compr);
%min_diff_dim_compr = min(diff_nondim_compr);
mean_diff_dim_compr = mean(diff_nondim_compr);
% % %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```matlab
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%solver 1 vs 14 vs 15
nondim_solver = [nondim(:,1), nondim(:,14) , nondim(:,15)];

for i=1:length(nondim_solver)

max_nondim_solver(i) = max(nondim_solver(i,:));

end
max_nondim_solver=max_nondim_solver';


for i=1:length(nondim_solver)

min_nondim_solver(i) = min(nondim_solver(i,:));

end

min_nondim_solver = min_nondim_solver';

diff_nondim_solver = (max_nondim_solver - min_nondim_solver);

%histfit(diff_nondim_turb,2932,'exponential')
%grid on

%max_diff_dim_solver = max(diff_nondim_solver);
%min_diff_dim_solver = min(diff_nondim_solver);
mean_diff_dim_solver = mean(diff_nondim_solver);
% % %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%nut 1 vs 16 vs 17
nondim_nut = [nondim(:,1), nondim(:,16) , nondim(:,17)];

for i=1:length(nondim_solver)

max_nondim_nut(i) = max(nondim_nut(i,:));

end
max_nondim_nut=max_nondim_nut';


for i=1:length(nondim_nut)

min_nondim_nut(i) = min(nondim_nut(i,:));

end

min_nondim_nut = min_nondim_nut';

diff_nondim_nut = (max_nondim_nut - min_nondim_nut);

%histfit(diff_nondim_turb,2932,'exponential')
%grid on

%max_diff_dim_nut = max(diff_nondim_nut);
%min_diff_dim_nut = min(diff_nondim_nut);
mean_diff_dim_nut = mean(diff_nondim_nut);
% % %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```matlab
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%press 1 vs 6 vs 7
nondim_press = [nondim(:,1), nondim(:,6) , nondim(:,7)];

for i=1:length(nondim_press)

max_nondim_press(i) = max(nondim_press(i,:));

end
max_nondim_press=max_nondim_press';

for i=1:length(nondim_press)

min_nondim_press(i) = min(nondim_press(i,:));

end

min_nondim_press = min_nondim_press';

diff_nondim_press = (max_nondim_press - min_nondim_press);

%histfit(diff_nondim_turb,2932,'exponential')
%grid on

%max_diff_dim_press = max(diff_nondim_press);
%min_diff_dim_press = min(diff_nondim_press);
mean_diff_dim_press = mean(diff_nondim_press);
% % %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


%%non-demsionalize based off of free-stream velocity
mean_diff_nondim_grid = mean_diff_dim_grid/uinf;
mean_diff_nondim_inletvelocity = mean_diff_dim_inletvelocity/uinf;
mean_diff_nondim_turb = mean_diff_dim_turb/uinf;
mean_diff_nondim_wallfns = mean_diff_dim_wallfns/uinf;
mean_diff_nondim_roughwall = mean_diff_dim_roughwall/uinf;
mean_diff_nondim_compr = mean_diff_dim_compr/uinf;
mean_diff_nondim_solver = mean_diff_dim_solver/uinf;
mean_diff_nondim_nut = mean_diff_dim_nut/uinf;
mean_diff_nondim_press = mean_diff_dim_press/uinf;
```

144

APPENDIX E: FLAT PLATE UNCERTAINTY MATLAB (FLATPLAT.M)

```matlab
%Curtis Groves Uncertainty Analysis of Flat Plate
clear all
clc

x = 0:0.01:1;
Ve=1;
lng=length(x);

hi=zeros(1,lng);
Uhi=zeros(1,lng);

%%%Flat Plate Correlation Calculation%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
for i=1:lng
    [hi(i),Uhi(i)]=Uncert_hi(Ve,x(i));
end
hi(1)=0;
Uhi(1)=0;
hiplusUh = hi + Uhi;
himinusUh = hi - Uhi;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


%Import and Interpolate CFD Data on to regular grid ----------------------
h_nominal = importdata('h_nominal');
h_nominal.data = sortrows(h_nominal.data);
[one(:,1),one(:,2)]=rem_dup_2(h_nominal.data(:,1),h_nominal.data(:,2));
one = interp1(one(:,1),one(:,2),x,'linear');

h_medgrid = importdata('h_medgrid');
h_medgrid.data = sortrows(h_medgrid.data);
[two(:,1),two(:,2)]=rem_dup_2(h_medgrid.data(:,1),h_medgrid.data(:,2));
two = interp1(two(:,1),two(:,2),x,'linear');

h_finegrid = importdata('h_finegrid');
h_finegrid.data = sortrows(h_finegrid.data);
[three(:,1),three(:,2)]=rem_dup_2(h_finegrid.data(:,1),h_finegrid.data(:,2));
three = interp1(three(:,1),three(:,2),x,'linear');

four = importdata('four');
four = sortrows(four.data);
[tmp(:,1),tmp(:,2)]=rem_dup_2(four(:,1),four(:,2));
four = interp1(tmp(:,1),tmp(:,2),x,'linear');
clear tmp

five = importdata('five');
five = sortrows(five.data);
[tmp(:,1),tmp(:,2)]=rem_dup_2(five(:,1),five(:,2));
five = interp1(tmp(:,1),tmp(:,2),x,'linear');
clear tmp

six = importdata('six');
six = sortrows(six.data);
[tmp(:,1),tmp(:,2)]=rem_dup_2(six(:,1),six(:,2));
six = interp1(tmp(:,1),tmp(:,2),x,'linear');
clear tmp
```

```matlab
seven = importdata('seven');
seven = sortrows(seven.data);
[tmp(:,1),tmp(:,2)]=rem_dup_2(seven(:,1),seven(:,2));
seven = interp1(tmp(:,1),tmp(:,2),x,'linear');
clear tmp

eight = importdata('eight');
eight = sortrows(eight.data);
[tmp(:,1),tmp(:,2)]=rem_dup_2(eight(:,1),eight(:,2));
eight = interp1(tmp(:,1),tmp(:,2),x,'linear');
clear tmp

nine = importdata('nine');
nine = sortrows(nine.data);
[tmp(:,1),tmp(:,2)]=rem_dup_2(nine(:,1),nine(:,2));
nine = interp1(tmp(:,1),tmp(:,2),x,'linear');
clear tmp

ten = importdata('ten');
ten = sortrows(ten.data);
[tmp(:,1),tmp(:,2)]=rem_dup_2(ten(:,1),ten(:,2));
ten = interp1(tmp(:,1),tmp(:,2),x,'linear');
clear tmp

eleven = importdata('eleven');
eleven = sortrows(eleven.data);
[tmp(:,1),tmp(:,2)]=rem_dup_2(eleven(:,1),eleven(:,2));
eleven = interp1(tmp(:,1),tmp(:,2),x,'linear');
clear tmp


twelve = importdata('twelve');
twelve = sortrows(twelve.data);
[tmp(:,1),tmp(:,2)]=rem_dup_2(twelve(:,1),twelve(:,2));
twelve = interp1(tmp(:,1),tmp(:,2),x,'linear');
clear tmp


thirteen = importdata('thirteen');
thirteen = sortrows(thirteen.data);
[tmp(:,1),tmp(:,2)]=rem_dup_2(thirteen(:,1),thirteen(:,2));
thirteen = interp1(tmp(:,1),tmp(:,2),x,'linear');
clear tmp
%Import and Interpolate CFD Data on to regular grid ----------------------
```

```matlab
%--------------Calculate CFD Uncertainty as Oscillatory Input Parametes
%Define k
k = 1.782;

CFD_DATA = [one',two',three',four',five',six',seven',eight',nine',ten',eleven',twelve',thirteen'];

for j=1:length(CFD_DATA)

Max(j,:) = max(CFD_DATA(j,:));
Min(j,:) = min(CFD_DATA(j,:));

end
Middle = 1/2*(Max-Min)+Min;
UCFD=k*1/2*(Max-Min);

CFD_Umax = Middle +UCFD;
CFD_Umin = Middle -UCFD;
%--------------Calculate CFD Uncertainty as Oscillatory Input Parametes


figure(1)
plot(x,hi ,x,hiplusUh, x,himinusUh ,x ,one ,'o', x, two,'+', x, three,'*', x,four,'-', x,five,'s', x,six,'*', x
grid on
xlabel('Position (m)')
ylabel('Heat Transfer Coefficient (h) (W/m2K)')
xlim([0.02 1])
legend('hi','hiplus','himinus','h nominal CFD','h medgrid CFD','h finegrid CFD', 'h Vlow CFD', 'h Vhigh CFD', 'l

figure(2)
plot(x,Uhi,x,UCFD)
legend 'Uhi' 'U CFD'
xlabel('Position (m)')
ylabel('Heat Transfer Coefficient Uncertainty (Uh) (W/m2K)')
xlim([0.01 1])
grid on

figure(3)
plot(x,hi)
legend 'hi'
xlabel('Position (m)')
ylabel('Heat Transfer Coefficient Correlation (h) (W/m2K)')
xlim([0.01 1])
grid on

figure(4)
plot(x,Uhi)
legend 'Uhi'
xlabel('Position (m)')
ylabel('Heat Transfer Coefficient Correlation Uncertainty (Uh) (W/m2K)')
xlim([0.01 1])
grid on


figure(5)
plot(x ,one ,'o', x, two,'+', x, three,'*', x,four,'-', x,five,'s', x,six,'*', x,seven,'+', x,eight,'o', x,nine
grid on
xlabel('Position (m)')
ylabel('Heat Transfer Coefficient CFD DATA (h) (W/m2K)')
xlim([0.02 1])
legend('h nominal CFD','h medgrid CFD','h finegrid CFD', 'h Vlow CFD', 'h Vhigh CFD', 'h rho low CFD', 'h rho h.

figure(6)
plot(x,UCFD)
legend 'U CFD'
xlabel('Position (m)')
ylabel('Heat Transfer Coefficient Uncertainty CFD (Uh) (W/m2K)')
xlim([0.01 1])
grid on


figure(7)
plot(x,hi ,x,hiplusUh, x,himinusUh ,x ,one ,'o',x,CFD_Umax,'-o',x,CFD_Umin,'-o')
grid on
xlabel('Position (m)')
ylabel('Heat Transfer Coefficient (h) (W/m2K)')
xlim([0.02 1])
legend('hi','hiplus','himinus','h nominal CFD', 'CFD Umax', 'CFD Umin')
```

Appendix E.1 Heat Transfer Correlation Uncertainty Calculation (Uncert_Hi.M)

```matlab
function [hi,Uhi]=Uncert_hi(Vel,x)
%diary('hi.txt')
%symbolically%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
syms V BV PV C;
syms L BL BC;

%Air Properties
syms rho Brho;
syms k Bk;
syms u Bu;
syms Pr BPr;

%calulate h
syms h Bh Uhb Uhc Uhr;


%%%%%%%%%%%%%%%%%%%%%Calculation Flat Plate Correlation


h=((k/L))*(C*(rho*V*L/u)^(0.8));

%
% disp('h=')
% pretty(h)
% disp(' ')
% disp(' ')
%
% disp('Displaying Partial Derivatives')
% disp(' ')
% disp(' ')




% %%%Uncertainity
% %%%derivation%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%
% %%%%%%%%%%%%%%Calculate Partial Derivatives

dV= diff(h,V);
% disp('dh/dV=')
% pretty(dV)
% disp(' ')
% disp(' ')

drho= diff(h,rho);
% disp('dh/drho=')
% pretty(drho)
% disp(' ')
% disp(' ')
```

```matlab
dk= diff(h,k);
% disp('dh/dk=')
% pretty(dk)
% disp(' ')
% disp(' ')

du= diff(h,u);
% disp('dh/du=')
% pretty(du)
% disp(' ')
% disp(' ')

dPr= diff(h,Pr);
% disp('dh/dPr=')
% pretty(dPr)
% disp(' ')
% disp(' ')

dL= diff(h,L);
% disp('dh/dL=')
% pretty(dL)
% disp(' ')
% disp(' ')

dC= diff(h,C);
% disp('dh/dC=')
% pretty(dC)
% disp(' ')
% disp(' ')
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```matlab
%Plug into Uncertainity Equation
Uhb=( ( (dV^2 * BV^2) + (drho^2 * Brho^2) + (dk^2 * Bk^2) + (du^2 * Bu^2) + (dPr^2 * BPr^2) + (dL^2 * BL^2) + (dC^2 * BC^2)));
% disp('Uhb=')
% pretty(Uhb)

%correlated errors
Uhc = abs((2*drho*dk*Brho*Bk + 2*drho*du*Brho*Bu + 2*drho*dPr*Brho*BPr + 2*dk*du*Bk*Bu + 2*dk*dPr*Bk*BPr + 2*du*dPr*Bu*BPr));
% disp('Uhc=')
% pretty(Uhc)

%random errors
Uhr = dV^2 * PV^2;
% disp('Uhr=')
% pretty(Uhr)

%Total
Uh = (Uhb + Uhc + Uhr)^.5;
% disp('Uh=')
% pretty(Uh)


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% Numerical Values
V=Vel;            BV=V*0.03;         PV=0;
L=x;              BL=0;
C=0.0263;         BC=0.0033;
%Air Properties
rho=1;            Brho=rho*0.03;
k=9.4505*10^-4;   Bk=k*0.03;
u=6.667*10^-7;    Bu=u*0.03;
Pr=.71;           BPr=Pr*0.03;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Calculate Values % Print to screen
%sprintf('havg= %0.2f +/- %0.2f (BTU / hr ft2 F)', eval(h),eval(Uh))

%sprintf('Percent = %0.2f', eval(Uh)/eval(h)*100)

hi=eval(h);
Uhi=eval(Uh);
```

Appendix E.2 Remove Duplicates Script (Rem_Dup_2.M)

```matlab
function [A,B]=rem_dup_2(b,c)
[m,n]=size(b);
counter=1;
for i=1:m-1
    if b(i)~=b(i+1)
        A(counter,1)=b(i);
        B(counter,1)=c(i);
        counter=counter+1;
    end
end

lng=length(A);
A(lng+1)=b(m);
B(lng+1)=c(m);
```

APPENDIX F: SUMMARY OF AUTHORS PRIOR WORK AND BACKGROUND

<div align="center">Curtis E. Groves</div>

Research Interests:
Computational Fluid Dynamics, Turbulence Modeling, Uncertainty Analysis, External Aerodynamics, Spacecraft Venting, Environmental Control Systems, Heat Transfer.

Degrees:

- Ph.D., Mechanical Engineering, University of Central Florida, Expected Completion Date 2014.
- M.S., Aerospace Engineering, University of Central Florida, May 2012.
- B.S., Aerospace Engineering, West Virginia University, May 2008.
- B.S., Mechanical Engineering, West Virginia University, May 2008.

Recent Work:

- Mars Science Laboratory Mission, NASA, Environmental Control System Impingement Analysis
- Mars Science Laboratory Mission, NASA, Nuclear Launch Approval Processes
- NASA, Source Evaluation Board
- Tracking and Data Relay Satellites (TDRS K/L), NASA, Environmental Control System Impingement Analysis
- Tracking and Data Relay Satellites (TDRS K/L) Missions, NASA, Spacecraft Venting Analysis
- Interface Region Imaging Spectrograph (IRIS), NASA, Environmental Control System Impingement Analysis
- GLORY Mission, NASA, Environmental Control System Impingement Analysis, Testing, Uncertainty Analysis

Honors and Awards:

- Group Achievement Award, ELVIS 2 Source Evaluation Board Team, NASA Kennedy Space Center, 2013.
- Group Achievement Award, Mars Science Lab, NASA Kennedy Space Center, 2012.
- Space Flight Awareness Award, NASA Kennedy Space Center, 2012.
- Certificate of Appreciation, NASA Kennedy Space Center, 2011.
- Distinguished Performance Rating, NASA, Launch Services Program, 2011.
- Completion of Accelerated Training Program, NASA Kennedy Space Center, 2008.
- NASA Cooperative Education Program, NASA Kennedy Space Center, 2006-2008.
- NASA West Virginia Space Grant Scholar, 2004-2008.
- Summa Cum Laude, West Virginia University, 2008.

- Ralph. M Barnes Senior Scholastic Achievement Award, West Virginia University (Graduate with highest GPA during the junior and senior years), 2008.
- Promise Scholar, West Virginia University, 2003-2008.
- President's List, West Virginia University (4.0 GPA), 2003-2008.

Publications:

1. Brink, J., Godfrey, G., Wittenborn, D., O'Keefe, D., **Groves. C**. (2005). Orbiter-External Tank (ET) Mate Simulation. National Aeronautics and Space Administration: John F Kennedy Space Center 2005 Technology, Development and Application Annual Report. 129-130.
2. Godfrey, G., O'Keefe, D., Whittenborn, D., **Groves, C**., & Kapr, F. (2006, December) Orbiter to External Fuel Tank Mating Simulation. Paper presented at Engineering Design Graphics Division Mid-Year Meeting in Ft. Lauderdale, Florida.
3. **Groves, C**., Ilie, M., Schallhorn, Paul. "Comprehensive Approach to Verification and Validation of CFD Simulations Applied to Backward Facing Step," AIAA-2013-0258, 2013.
4. **Groves, C**., Ilie, M., Schallhorn, Paul. "Computational Fluid Dynamics Uncertainty Analysis applied to Heat Transfer over a Flat Plate," American Physical Society Division of Fluid Dynamics, 2013. DFD13-2013-00087
5. **Groves, C**., Ilie, M., Schallhorn, Paul. "Computational Fluid Dynamics Uncertainty Analysis for Payload Fairing Spacecraft Environmental Control Systems," AIAA-2014-0440, 2014.
6. **Groves, C**., Ilie, M., Schallhorn, Paul. "Interpolation Method needed for Numerical Uncertainty Analysis of Computational Fluid Dynamics," AIAA-2014-1433, 2014.

LIST OF REFERENCES

[1] M. Kandula, K. Hammad and P. Schallhorn, "CFD Validation with LDV Test Data for Payload/Fairing Internal Flow," in *AIAA-2005-4910*, 2005.

[2] M. Kandula and L. Walls, "An application of overset grids to payload/fairing internal flow CFD analysis," in *20th AIAA Applied Aerodynamics Conference*, AIAA-2002-3253, June 2002.

[3] P. J. Roache, K. N. Ghia and F. M. White, "Editorial Policy Statement on the Control of Numerical Accuracy," *ASME Journal of Fluids Engineering,* vol. 108, p. 2, 1986.

[4] J. Slater, "Uncertainty and Error in CFD Applications," NPARC Alliance CFD Verification and Validation, July 2008. [Online]. Available: http://www.grc.nasa.gov/WWW/wind/valid/tutorial/errors.html.

[5] F. Iudicello, "Guide for Verification and Validation of Comutational Fluid Dynamics Simulations," *American Institute of Aeronautics and Astronautics (AIAA),* no. AIAA G-077-1998, 1998.

[6] I. Celik and W. Zhang, "Calculation of Numerical Uncertainty Using Richardson Extrapolation: Application to Some Simple Turbulent Flow Calculations," *American Society of Mechanical Engineers Journal of Fluid Mechanics,* 1995.

[7] P. J. Roache, Verification and Validation in Computational Science and Engineering, Hermosa Publishers, 1998.

[8] F. Stern, R. V. Wilson, H. W. Coleman and E. G. Patterson, "Verification and Validation of CFD Simulations," Iowa Institute of Hydraulic Research No.407, Septermber 1999.

[9] F. Stern, R. V. Wilson, H. W. Coleman and E. G. Patterson, "Comprehensive Approach to Verification anD Validation of CFD Simulations - Part 1: Methodology and Procedures," *ASME Journal of Fluids Engineering,* no. 123, pp. 793-802, 2001.

[10] F. Stern, R. V. Wilson, H. W. Coleman and E. G. Patterson, "Comprehensive Approach to Verification and Validation of CFD Simulations - Part 2: Application for Rans Simulation of a Cargo/Container Ship," *ASME Journal of Fluids Engineering,* vol. 123, pp. 803-810, 2001.

[11] "Uncertainty Analysis in CFD Verification and Validation Methodology and Procedures," in *International Towing Tank Conference – Recommended Procedures and Guidelines*, September 2008.

[12] I. Celik, U. Ghia, P. Roache, C. Freitas, H. Coleman and P. Raad, ""Procedure for Estimating and Reporting of Uncertainity Due to Discretization in CFD Applications," *American Society of Mechanical Enigneers Journal of Fluids Engineering,* vol. 130, 2008.

[13] I. Celik, *Critical Issues With Quantification of Discretization Uncertainty in CFD,* National Energy Technology Laboratory Workshop on Multiphase Flow Science, August 2011.

[14] "Standard for Verification and Validation in Computational Fluid Dynamics and Heat Transfer," *The American Society of Mechanical Engineers,* vol. An American National Standard, no. ASME V&V 20-2009, 2009.

[15] C. Roy, "Review of Discretization Error Estimators in Scientific Computing," *American Institute of Aeronautics and Astronautics,* Vols. AIAA 2010-126.

[16] P. J. Roache, "Perspective: A Method for Uniform Reporting of Grid Refinement Studies," *Journal of Fluids Engineering,* vol. 116, pp. 405-413, 1994.

[17] L. F. Richardson, ""The Approximate Arithmetical Solution by Finite Differences of Physical Problems involving Differential Equations, with an Application to the Stresses in a Masonry Dam," *Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character,* vol. 210, pp. 307-357, 1911.

[18] L. F. Richardson, "The Deferred Approach to the Limit, PartI. Single Lattice," *Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character,* vol. 226, pp. 299-349, 1927.

[19] C. Groves, M. Ilie and P. Schallhorn, ""Comprehensive Approach to Verification and Validation of CFD Simulations Applied to Backward Facing Step," in *AIAA-2013-0258*, 2013.

[20] H. W. Coleman and W. G. Steele, Experimentation and Uncertainty Analysis for Engineers, 2nd Ed, New York: John Wiley & Sons, Inc, 1999.

[21] C. E. Groves, P. A. Schallhorn and M. Ilie, "Computational Fluid Dynamics Uncertainty Analysis for Payload Fairing Spacecraft Enviornmental Control Systems," in *AIAA-2014-0440*, 2014.

[22] M. Kandula, R. Nallasamy, P. Schallhorn and L. Duncil, "An Application of Overset Grids to Payload / Fairing Three-Dimensional Internal Flow CFD Analysis," KSC-2007-215, 2007.

[23] M. Kandula, R. Nallasamy, P. Schallhorn and L. Duncil, ""An Application of Overset Grids to Payload/Fairing Three-Dimensional Internal Flow CFD Analysis," *Engineering Applications of Computational Fluid Mechanics,* vol. 2, no. 1, pp. 119-129, 2008.

[24] M. Kandula, K. Hammad and P. Schallhorn, "CFD Validation with LDV Test Data for Payload/Fairing Internal Flow," KSC-2005-4910, 2005.

[25] "Launching Rockets," NASA, [Online]. Available: http://www.nasa.gov/centers/kennedy/launchingrockets/. [Accessed 28 January 2013].

[26] "Delta II Payload Planners Guide," United Launch Alliance, [Online]. Available: http://www.ulalaunch.com/site/docs/product_cards/guides/DeltaIIPayloadPlannersGuide2007.pdf. [Accessed 28 January 2013].

[27] "Delta IV Payload Planners Guide," [Online]. Available: http://www.ulalaunch.com/site/docs/product_cards/guides/DeltaIVPayloadPlannersGuide2007.pdf. [Accessed 28 January 2013].

[28] "Atlas 5 Payload Planners Guide," [Online]. Available: http://www.ulalaunch.com/site/docs/product_cards/guides/AtlasVUsersGuide2010.pdf. [Accessed 28 January 2013].

[29] "Pegasus Users Guide.," Orbital Sciences Coorporation, [Online]. Available: http://www.orbital.com/NewsInfo/Publications/Pegasus_UG.pdf. [Accessed 28 January 2013].

[30] "Taurus Users Guide.," [Online]. Available: http://www.orbital.com/NewsInfo/Publications/taurus-user-guide.pdf. [Accessed 28 January 2013].

[31] "Space X Falcon 9 Users Guide," Space X, [Online]. Available: http://www.spacex.com/Falcon9UsersGuide_2009.pdf. [Accessed 28 January 2013].

[32] C. E. Groves, P. A. Schallhorn and M. Ilie, "Interpolation Method needed for Numerical Uncertainty Analysis of Computational Fluid Dynamics," in *AIAA-2014-1433*, 2014.

[33] I. Celik and O. Karatekin,, "Numerical Experiments on Application of Richardson Extrapolation With Nonuniform Grids," *ASME Journal of Fluids Engineering,* vol. 119, no. 3, p. 584, 1997.

[34] *ANSYS FLUENT 12.0 User's Guide.*

[35] "OpenFOAM User Guide," The OpenFOAM Foundation, [Online]. Available: http://www.openfoam.org/docs/user/.

[36] "Matlab," Mathworks Product Website, [Online]. Available: http://www.mathworks.com/products/matlab.

[37] R. Fox and A. McDonald, Introduction to Fluid Mechanics, 3rd Edition, John Wiley & Sons, 1985.

[38] P. L. Roache and P. M. Knupp, "Completed Richardson Extrapolation," *Communications in Numerical Methods in Engineering,* vol. 9, pp. 365-374, 1993.

[39] J. S. Brock, "Bounded Numerical Error Estimates for Oscillatory Convergence of Simulation Data," in *AIAA-2007-4091*, 2007.

[40] I. Celik, J. Li, G. Hu and C. Shaffer, "Limitations of Richardson's Extrapolation and Some Possible Remedies," in *2004 ASME Heat Transfer/Fluids Engineering Summer Conference*, Charlotte, North Carolina, 2004.

[41] H. W. Coleman, F. Stern, A. Di Mascio and E. Campana, "The Problem with Oscillatory Behavior in Grid Convergence Studies," *ASME Journal of Fluids Engineering,* vol. 123, pp. 438-439, 2001.

[42] L. Eca and M. Hoekstra, "An Evaluation of Verification Procedures for CFD Applications," in *24th Symposium on Naval Hydrodynamics*, Fukuoka, Japan, 2002.

[43] C. E. Groves, P. A. Schallhorn and M. Ilie, "Computational Fluid Dynamics Uncertainty Analysis applied to Heat Transfer over a Flat Plate," in *American Physical Society Division of Fluid Dynamics DFD13-2013-00087*, 2013.

[44] "FLUENT - Forced Convection over a Flat Plate," Cornell University, [Online]. Available: https://confluence.cornell.edu/display/SIMULATION/FLUENT+-+Forced+Convection+over+a+Flat+Plate.