QUANTIFYING THE UNCERTAINTY IN A COMPUTATIONAL FLUID DYNAMICS TURBULENT TWIN JET MODEL

Seth Sheldon Lawrence

A Thesis

Submitted in Partial Fulfillment

of the Requirements for the Degree of

Master of Science
In
Mechanical Engineering

Thomas L. Acker, Ph.D., Chair
Earl P.N. Duque, Ph.D.
Sagnik Mazumdar, Ph.D.

Northern Arizona University

May 2017

KEYWORDS: Verification, Validation, Uncertainty Quantification, Richardson Extrapolation, Grid Convergence Index, Aleatory, Epistemic, Twinjet

QUANTIFYING THE UNCERTAINTY IN A COMPUTATIONAL FLUID DYNAMICS TURBULENT TWIN JET MODEL

Seth Sheldon Lawrence

ABSTRACT

Ubiquitous application of CFD motivates the need to verify and validate CFD models and quantify uncertainty in the results. The objective of this research was to investigate the uncertainty interval over which an ANSYS Fluent CFD model predicted the axial velocity in a turbulent twin jet flow regime with 95% confidence. The modeling domain was composed of water, injected through a nozzle and into a static holding tank. This configuration was described by the American Society of Mechanical Engineers (ASME), Nuclear System Thermal Fluids Behavior (V&V30) Standard Committee as a twin jet benchmark verification and validation problem. The steady Reynolds Average Navier-Stokes (RANS) approach utilizing a realizable k-ε turbulence model was chosen. The system response quantity under consideration was the axial velocity of the flowfield in both the pre- and post-combing regions of the twin jet flow. The model input uncertainty in the jet width and spacing was treated as epistemic, with aleatory uncertainties in the mass-flow-rate and turbulence inputs for each jet. Numerical uncertainty was considered at the discretization level, through grid refinement and the Grid Convergence Index (GCI) method. Validation uncertainty was achieved through a validation procedure using experimental data. The uncertainties in the model inputs, numerics, and validation, were combined to quantify the total uncertainty in the Fluent CFD model. The results indicated that numerical uncertainty was the dominate factor in the region near the jet nozzle, located before the two jets merge together. Moving further away from the nozzle, to the region where the jets merge to form a single jet, the numerical uncertainty was reduced significantly. In this region, differences between the model and experiment resulted in a

dominant validation uncertainty. This uncertainty was observed as the consistent under prediction of axial velocity in the combined flow region. The final results offered a prediction of the total uncertainty in a 3-D Fluent CFD model; utilizing the steady-state, k-ε turbulence model approach, by applying the standard verification, validation, and uncertainty quantification techniques, to the ASME benchmark twin jet problem.

Table of Contents

List of Figure

List of Tables

Nomenclature

u, v, w	*Components of velocity in the x-, y-, z-directions (m/s)*
x, y, z	*Coordinate directions for a Cartesian system*
t	*Time (s)*
p	*Pressure (Pa)*
ν	*Kinematic viscosity of a fluid (m^2/s)*
ρ	*Density (kg/m^3)*
g	*Gravitational acceleration (m/s^2)*
u'	*Fluctuating x-direction velocity (m/s)*
$\overline{U}$	*Time-average of the x-direction velocity (m/s)*
U	*Instantaneous x-direction velocity (m/s)*
μ	*Dynamic viscosity of a fluid (kg/m-s)*
R	*Reynolds stress tensor ($kg/m\text{-}s^2$)*
k	*Turbulent kinetic energy (m^2/s^2)*
ε	*Turbulent dissipation (m^2/s^2)*
μ_t	*Turbulent viscosity (kg/m-s)*
CV	*Control volume*
CS	*Control Surface*
V	*Volume (m^3)*
$\boldsymbol{V}$	*Velocity vector (m/s)*
$\boldsymbol{A}$	*Area vector (m^2)*
Δx	*Distance between grid points for the x-direction (m)*
a	*Width of single jet (mm)*
s	*Spacing between jets (mm)*
y/a	*Nondimensional distance above the jet inlet to tank*
x/a	*Nondimensional distance from centerline between jets*
h	*Characteristic grid dimension (m)*
$\Delta y, \Delta z$	*Distance between grid points for the y-,z-direction (m)*
r	*Grid refinement factor*

$\hat{p}$	*Observed order of accuracy*
Fs	*Factor of safety*
φ_1	*Fine grid CFD solution value*
φ_2	*Medium grid CFD solution value*
φ_3	*Coarse grid CFD solution value*
ε_{32}	*Difference between coarse and medium grid solution*
ε_{21}	*Difference between medium and fine grid solution*
ΔM	*Difference between the coarse and fine grid solution*
N_1	*Fine grid*
N_2	*Medium grid*
N_3	*Coarse grid*
U_{num}	*Numerical uncertainty*
U_{input}	*Input uncertainty*
U_D	*Experimental uncertainty*
U_{val}	*Validation uncertainty*
U_{SRQ}	*Total uncertainty in system response quantity*
$\dot{m}$	*Mass flow rate of a single jet (kg/s)*
I	*Turbulence intensity of a single jet (%)*
l	*Turbulent length scale of single jet (m)*
σ	*Statistical mean value*
μ_s	*Statistical standard deviation from the mean*
s/a	*Nondimensional jet spacing ratio*

1 INTRODUCTION

1.1 Motivation

To support the energy desires of a global society, a new generation (Gen-IV) of nuclear reactors will be designed to provide energy that is sustainable. The sodium-cooled reactor is a candidate design to be used in the development of this new reactor, and uses turbulent jets in close proximity to cool the system [1]. Computational fluid dynamics (CFD) is a simulation tool that engineers use to predict the flow of these jets through numerical modeling. These CFD models include errors when solving complex fluid dynamics problems, due to numerical approximations that must be made. These errors introduce a level of uncertainty to the predictions made by a CFD model.

Pertaining to the field of CFD research and the future of the new reactor design, consider the following question. What is the uncertainty in a CFD model, used to simulate and predict the velocity and mixing of two turbulent jets in close proximity? To answer this question, the CFD code Fluent was used to simulate the flow of twin jets with turbulent mixing. The model conditions of the simulation were similar to conditions described in the new reactor design. Application of current verification and validations (V&V) techniques were utilized to quantify the uncertainty in system response quantities predicted by the simulation. Strengths and weaknesses found in applying standard V&V techniques to this complex fluid dynamics problem were found. The results presented in this research, provided insight to the capabilities of ANSYS Fluent, as a CFD tool for modeling turbulent twin jets.

1.2 Computational Fluid Dynamics

In the context of this work, computational fluid dynamics or CFD was used to numerically solve the Navier-Stokes equations that govern the behavior of a fluid in terms of mass, momentum, and energy. ANSYS Fluent is the commercial CFD software package that was used to solve a form of the Navier-Stokes equation. This form of the equations that were solved, is applicable to turbulent flows, and referred to as the Reynolds Average Navier-Stokes (RANS) equations. More details concerning the numerical

solution technique are provided in Chapter 3. The data generated by the CFD simulation was post-processed using FieldView, to visualize and analyze the outputs generated by the solver.

1.3 Modeling in CFD

The steps that were followed to setup, run a simulation, and produce results are delineated in Table 1. This oulines the iterative process upon which the CFD model of the turbulent twin jets was built and utilized throughout this work.

Table 1: Categorizing and defining the four components of the CFD workflow that was utilized in the development of the turbulent twin jet CFD simulation.

Geometry:	The domain which describes where model was defined.
Meshing:	The process of dividing the geometry into finite points where to solution to equations was solved.
Solution:	The computational process of solving the equations at each point described by the mesh.
Results:	The process of analyzing the data produced by solving the equations numerically.

1.4 Verification and Validation of CFD Models

Verification can be considered in two different ways; code verification and solution verification. Code verification was the procedure for ensuring that there were no programming errors in the CFD solver. Solution verification was used to quantifying and understand the "numerical error" in the CFD simulation. Numerical error is the consequence of the approximations that must be made by a CFD simulation to solve complex fluids flows. In order to quantify this numerical error in the CFD model, the solution was verified. The details of verifying the CFD solution are discussed in Chapter 5. Validation was the procedure used to compare the simulation results with experimental data. The steps that were used to validate the CFD model through the comparison with measured experimental values are discussed in Chapter 6.

1.5 Uncertainty in CFD Models

Quantifying the uncertainty in the CFD model was the process of determining the interval over which a system response quantity (SQR) would fall, within a specified degree of confidence. A system response

quantity was the value of any flow-related quantity that is determined by running a CFD model. Examples of typical SQRs are the velocity at a point, or at several points, the mass flow rate across a surface, the force on a surface (i.e. lift or drag), etc. Often referred to as "uncertainty quantification" (UQ), this step accounted for and combined all the errors and uncertainties that would influence the SRQ of interest. A thorough UQ of the CFD model required significant computational resources. The procedures and techniques for quantifying the uncertainty in the CFD model are discussed in Chapter 6.

1.6 High Performance Computing in CFD

The use of High Performance Computing (HPC) in CFD modeling referred to usage of parallel processing (usually on a computer "cluster" that contains many central processing units (CPUs) arranged in parallel) as a tool to solve models that would otherwise not be computationally feasible on a single computer workstation. In this work, parallelism was defined as the use of multiple computational processing units (CPUs) in parallel to work on different parts of the same problem. HPC availability factored into the computational budget, and was carefully considered in Chapter 7.

1.7 Turbulent Twin Jet Flow Problem

The motivation of this work was to quantify the uncertainty in a Fluent CFD model that predicted the velocity and turbulent mixing of twin jets in close proximity. This flow regime was representative of the conditions found in the upper plenum of an advanced liquid metal-cooled nuclear reactor [2]. A scaled twin jet experiment (working fluid was water), was conducted by Texas A&M University in December of 2015, using particle image velocimetry (PIV) and laser Doppler anemometry (LDA). This experiment provided measurements of the mean and fluctuating velocity field surrounding the twin jets. This experiment utilized high resolution measurements and provided datasets that could be used in the UQ of the Fluent CFD model. The Fluent model was designed to replicate the experimental conditions as closely as possible, by using information that documented the setup and boundary conditions. The particular configuration of this twin jet problem, was described by the American Society of Mechanical Engineers Verification and Validation in Computational Nuclear System Thermal Fluids Behavior Committee

(ASME V&V 30), and proposed as a benchmark challenge problem. Additional published information regarding the specific details of this problem can be found at: https://www.asme.org/events/vandv/program/challenge-problems as well as in Ref. [3].

First, the scaled experimental setup was modeled using Fluent, and the solution was verified. To quantify the uncertainty in the model, a reasonable confidence of 95% was selected, due to the wide range of engineering applications. The SRQ of interest that was considered in the UQ analysis was the axial velocity of the twin jet flowfield. The results of the UQ were reported with a specified 95% confidence, and represented the quantified uncertainty interval of the Fluent SRQ prediction.

1.8 Objectives of Thesis

The objectives of this research are outlined, to represent the foundation for the work presented in this document.

- Model ASME benchmark turbulent twin jet problem using the CFD solver Fluent
- Estimate the numerical uncertainty in the SRQ predicted by the CFD model
- Evaluate the input uncertainty in the SRQ predicted by the CFD model
- Enumerate the validation uncertainty in SRQ predicted by the CFD model
- Quantify the total uncertainty in predicted SRQ (axial velocity) with 95% confidence

1.9 Organization of Thesis

Chapter 2 provides a review of literature pertaining to validation and uncertainty quantification of numerical simulations. This chapter also includes literature pertaining to previous works in numerical modelling of twin jets. In Chapter 3, the details of the Fluent solver are described, to better understand how the governing equation are being solved numerically. The model approach and setup is discussed in Chapter 4, including boundary conditions and solver settings. Chapter 5 discusses the grid refinement index method used to verify the solution. All sources of uncertainty are identified in Chapter 6, including a description of the methods for quantifying all sources of uncertainty. This chapter concludes by combining all errors and uncertainties into the total uncertainty in the model prediction of the SRQ.

Chapter 7 presents the workflow utilized to combine all computational tools used in this study. Finally, the results of the research are presented and discussed along with conclusions in Chapters 8 and 9. Documentation of the model, scripts used to set-up and run the computational models, and other information required to reproduce the results presented in this thesis, are provided in the Appendices.

2 REVIEW OF LITERATURE

The prominent works pertaining to the verification, validation, and uncertainty quantification of numerical simulations, including twin jet flow models were considered. Only with great acknowledgement to the previous works mentioned here, was the research in this document possible. The literature discussed in this chapter was fundamentally integrated into the principles and decisions adopted throughout this study.

2.1 Verification Methods

The methods available for verification of a numerical simulation were found to be standardized in the American Society of Mechanical Engineers V&V20 [4], which outlined general procedures for estimating the discretization error and uncertainty using the Richardson extrapolation and the Grid Refinement Index (GCI). Richardson extrapolation has been used as a particularly useful error estimator, in that it may be applied to any method of discretization, mainly finite-volume, -difference, -element. The GCI originally suggested by Roache [5] used the observed order of accuracy obtained through grid refinement of a numerical model to determine the asymptotic behavior of the solution. If asymptotic behavior was observed, that is, the solution was converging to a fixed value proportional to the order of the solution method, the GCI value which includes a safety factor of 1.25 was shown to estimate the discretization uncertainty very well. An upper limit (observed order greater than the theoretical order) for the observed order was suggested by Roache in [6], due to the significant reduction in errors at observed values above the theoretical order. If asymptotic solution behavior was observed, the GCI method was found to be a reliable estimator of the numerical uncertainty.

In practice, asymptotic behavior is often difficult to achieve and thus estimation of the numerical discretization error and uncertainty is challenging to obtain. Work published by Eca and Hoekstra [7] documented a least-squares technique for addressing the numerical uncertainty when the solution behavior was not well behaved. This involved categorizing the convergence of the solution. For each of the classifications, a separate estimator was applied, according to the behavior of the observed order of

accuracy. This was found to be useful in determining numerical uncertainty when the GCI method was unreliably.

2.2 Developments in Uncertainty

The term *uncertainty* in the field of computational modeling was not found to be established and adopted under a universal consensus. The works presented by Roy and Oberkampf [8], [9] were taken as the current "standard" that is being adopted by many in the field. This was demonstrated in the recent publication by Lee *et al.* [10], which documented the recent updates to the role of uncertainty in computational modeling by adopting many of the practices suggested by Roy and Oberkampf.

Despite the multitude of uncertainty approaches that were found in the literature, the most widely adopted was referenced to Morgan and Henrion [11]. This approach was employed in the risk assessment communities, and looked to categorize uncertainty as either *aleatory* or *epistemic*. In a general sense, these categories attempted to quantify the various uncertainties in the form of random or systemic unknowns. Typically, random aleatory uncertainty was described by an interval with a probability distribution, while the epistemic uncertainty was said to exist on an interval with no distribution.

Classifying the uncertainty in a computational model as either aleatory or epistemic, comprised the first step in assessing the uncertainty in the model prediction. A rigorous procedure for quantifying the uncertainty in a model prediction was documented in detail by Roy and Oberkampt [9]. Additional documentation in uncertainty quantification techniques with applications in CFD was also discussed in in the ASME V&V 20-2009 standard [4]. All methods included identifying the sources of uncertainty, in the attempt to quantify the uncertainty in a numerical model prediction of a system response quantity.

2.3 Twin Jet Flow

The flow regime established when two or more jets interact at close proximity is distinctly unique from single jet flow. This is due to the interaction between multiple jets, which causes the flow from each jet to merge together and form a single jet, often referred to as *mutual entrainment*. This merging of jets is associated with a low-pressure recirculation region, located just prior to the merging of the jets, and is

entirely distinctive from a single jet. One of the earliest works that addressed this interactions between multiple jets was a 1977 publication by Marsters [12]. This article proposed an integral analysis method to model the merging process of the jets, but over-predicted the rate at which the merging occurred. Marsters concluded by suggesting a numerical method would be required to continue the work, noting a satisfactory turbulence model to be the most significant challenge.

In a more recent development, the ASME Verification and Validation in Computational Nuclear System Thermal Fluids Behavior Committee (ASME V&V30 Standard Committee) proposed a benchmark problem to address the turbulent mixing and merging of twin jets [3]. The configuration was similar to that found in an advanced liquid metal-cooled reactor, which coincided with the base motivation of this work. In 2015, a validation experiment conducted by Wang and Hassan [2][1] documented the use of laser-doppler measuring techniques to capture and report data concerning the turbulent mixing of twin parallel jets. The experiment was reported to use water as the working fluid, with an image of the setup shown in Figure 1. The corresponding measurements of the velocity field that were collected during the experiment can be seen in the form of axial velocity contours in Figure 2.

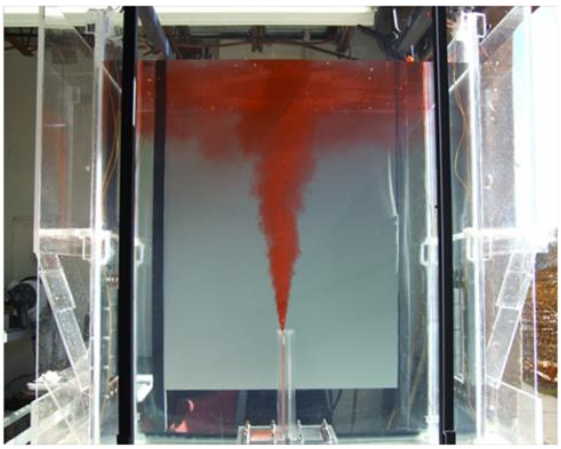


Figure 1: Image of the setup that was used to collect experimental data of the axial flow velocity to be used in the validation of numerical models. [13]

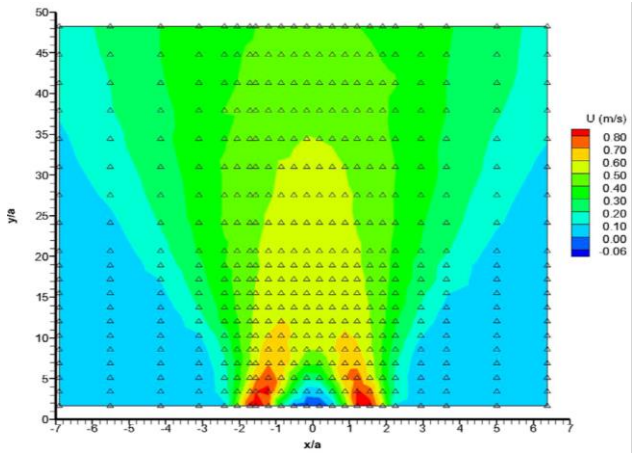


Figure 2: Contour plot of the mean axial velocity field that was obtained from the experimental setup of twin jets and measured using laser-doppler. [1]

Over the last few years, several efforts to model jets using CFD codes were documented. In a conference paper, Lu and Wang [14] presented the results using a 2-D Fluent model to predict the flow of a single

water jet that was injected vertically into a stationary tank. They demonstrated that the proximity of the wall boundary conditions to the jet would cause the jet to "lean" to the side and attach to the wall as shown in Figure 3. This result was nonphysical and attributed to nonsymmetrical vortices that developed in the 2-D numerical solution. For the outlet boundary condition, a volume-of-fluid (VOF) method was used to simulate the air-water condition at the top of the tank, but found to have no influence on the jet flow. Model results were presented and discussed. The paper did not address verification of the solution or uncertainty quantification.

In a paper by Carasik, Ruggles, and Hassan [15], a 3-D twin jet model was described using Adapco's CFD software Star-CCM+. This model used the same configuration and boundary conditions shown in Figure 1. Figure 4 shows a contour plot of the velocity field reported by this work, and though a grid refinement study was mentioned, it was not described in the paper. The solver used a steady, k-ε turbulence model approach to capture the mixing.

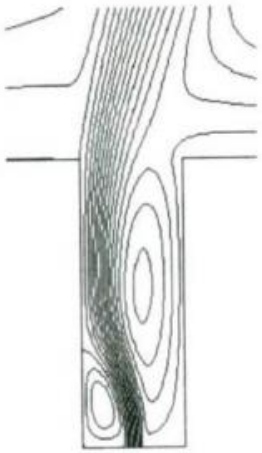


Figure 3: Image of the 2-D Fluent model of a single jet that artificially attaches to the wall boundary condition that was reported in the work of Lu and Wang. [14]

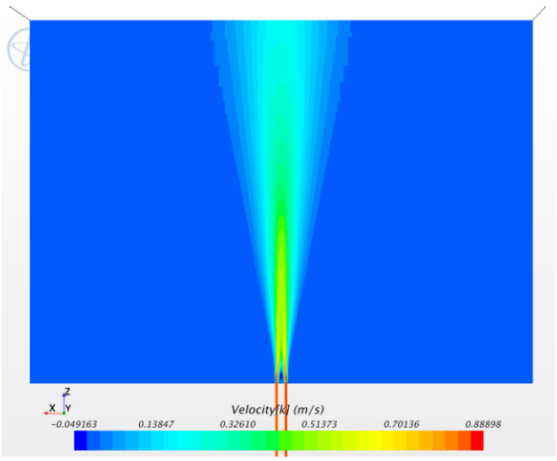


Figure 4: Contour image of the axial velocity field that was reported by a 3-D twin jet CFD model using the software Star-CCM+. [15]

Several additional publications pertaining to jets in close proximity under varying configurations and boundary conditions can be found in references [16]–[18]. While some work does exists in the CFD modeling of twin jets, very little documentation was available in the verification and uncertainty

quantification of the models. To build upon the previous research, an emphasis on quantifying the uncertainty in the numerical model prediction is required.

3 COMPUTATIONAL FLUID DYNAMICS

It was very important to understand the fundamental principles upon which a numerical CFD model is built. Understanding these principles helped to avoid potential pitfalls that may arise, due to a lack of knowledge regarding the tools that would be used. Starting with governing equations, a general overview of the procedure used to develop a turbulence model and compose a system of discretized partial differential equations was reviewed. The implication of the model assumptions and discretization accuracy was found to propagate throughout this work. This is seen later on, in the form of numerical and validation uncertainty, that will be discussed in Chapters 5, 6, 8, and 9.

3.1 Governing Equations

The full Navier-Stokes equations are a complex set of governing partial differential equations (PDEs) that describe the behavior of a fluid in terms of conserving mass, momentum, and energy. The governing equations for incompressible flow in differential form that are discretized and solved by a CFD solver, can be written as follows.

Incompressible Navier-Stokes Equations [19]:

$$continuity \qquad \frac{\partial u}{\partial x}+\frac{\partial v}{\partial y}+\frac{\partial w}{\partial z}=0 \tag{1}$$

$$x-momentum \qquad \frac{\partial u}{\partial t}+u\frac{\partial u}{\partial x}+\mathrm{v}\frac{\partial u}{\partial y}+w\frac{\partial u}{\partial z}=-\frac{1}{\rho}\frac{\partial p}{\partial x}+\nu\left[\frac{\partial^2 u}{\partial x^2}+\frac{\partial^2 u}{\partial y^2}+\frac{\partial^2 u}{\partial z^2}\right]+g_x \tag{2a}$$

$$y-momentum \qquad \frac{\partial \mathrm{v}}{\partial t}+u\frac{\partial \mathrm{v}}{\partial x}+\mathrm{v}\frac{\partial \mathrm{v}}{\partial y}+w\frac{\partial \mathrm{v}}{\partial z}=-\frac{1}{\rho}\frac{\partial p}{\partial y}+\nu\left[\frac{\partial^2 \mathrm{v}}{\partial x^2}+\frac{\partial^2 \mathrm{v}}{\partial y^2}+\frac{\partial^2 \mathrm{v}}{\partial z^2}\right]+g_y \tag{2b}$$

$$z-momentum \qquad \frac{\partial w}{\partial t}+u\frac{\partial w}{\partial x}+\mathrm{v}\frac{\partial w}{\partial y}+w\frac{\partial w}{\partial z}=-\frac{1}{\rho}\frac{\partial p}{\partial z}+\nu\left[\frac{\partial^2 w}{\partial x^2}+\frac{\partial^2 w}{\partial y^2}+\frac{\partial^2 w}{\partial z^2}\right]+g_z \tag{2c}$$

Since the flow under consideration in the twin jet problem could be considered an incompressible liquid, the energy equation would not be needed, and the four unknowns, *u*, *v*, *w*, and *p* can be solved via the four equations 1-2c. Solving the Navier-Stokes equations is typically a cornerstone of the modeling efforts in the field of Computational Fluid Mechanics. Since solutions to these equations using analytical

techniques do not exist for most problems, CFD modeling offers a unique solution technique that would not otherwise be possible. In addition to solving complex problems with no analytical solutions, CFD offers a particular advantage by offering the ability to model turbulent flows. Turbulence is associated with the time-dependent fluctuations in a flowfield that are observed in both steady and unsteady fluids problems. The space and time scale necessary to resolve these fluctuations is incredibly small, meaning the grid resolution and time step used in the solution must be very small. This can be done, and is referred to as a direct numerical simulation (DNS), however, the computational cost that accompanies such simulations is not practical. Instead of performing a DNS, CFD models such as Fluent, employ advanced averaging techniques which are able to model the turbulent fluctuations at a fraction of the computational cost. For this reason, the turbulence associated with the twin jet in this study, makes CFD a key tool in solving the problem. Fluent is able to approximate the time-dependent turbulent nature of the flow by employing a modified version of the Navier-Stokes equations that is referred to as the Reynolds Average Navier-Stokes (RANS) equations. The use of these modified RANS equations introduces an entire subset category in CFD known as turbulence modeling.

3.2 Turbulence Modeling

Turbulence modeling is the effort to model the turbulent characteristics of fluid flow without resorting to the computational expense of a DNS. In a DNS, no model is used, and the full Navier-Stokes are solved to resolve the intricate nature of the turbulence. As shown in Figure 5 and Figure 6, the RANS approach does not capture the intricate nature of the turbulence, but does provide good information about the diffusion of momentum that is caused by viscous and turbulent effects, at a fraction of the computational cost.

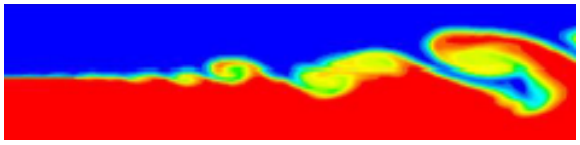


Figure 5: Contour image of a turbulent velocity flowfield that was computed using a Direct Numerical Simulation (DNS) approach. [20]

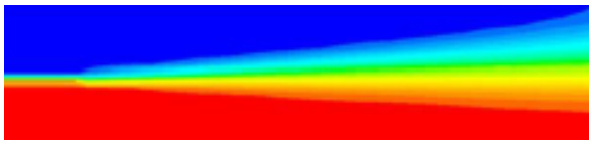


Figure 6: Contour image of a turbulent velocity flowfield that was computed using a Reynolds Average Navier-Stokes (RANS) approach. [20]

The RANS approach is achieved by understanding the behavior of turbulent flow and separating into components to simplify the problem. As shown in Figure 7, the unsteady turbulent velocity at a point can be decomposed into a mean and fluctuating component. This decomposition is applied to the entire velocity field by averaging the mean and fluctuating velocity.

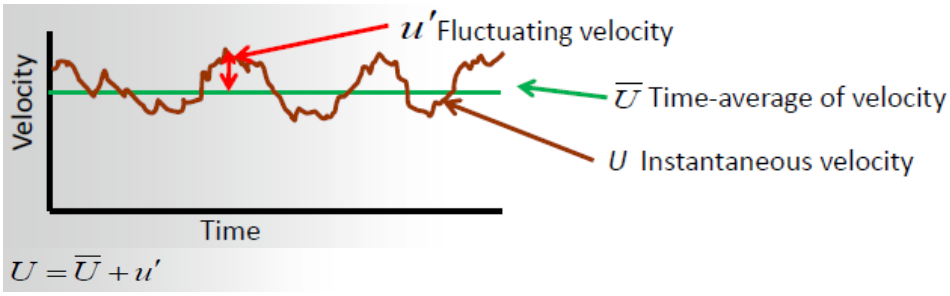


Figure 7: Mean and fluctuating components of a turbulent flow that is assumed in the RANS turbulence model approach. [20]

As shown for the x – direction in Figure 7, the mean, $\overline{U}$ and fluctuations, u' of a turbulent flowfield, are used to replace the instantaneous velocity U. This averaging process is applied to all the components of the velocity vector, u, v, w that were in the governing equations 1-2c. The result of this averaging of the velocity field is adopted and employed in the formulation of any RANS solution. The result is a full set of RANS equations that are used to model turbulent flow in place of governing Navier-Stokes discussed previously.

Incompressible Reynolds Average Navier-Stokes Equations [20]:

$$continuity \qquad \frac{\partial \bar{u}}{\partial x} + \frac{\partial \bar{v}}{\partial y} + \frac{\partial \bar{w}}{\partial z} = 0 \tag{3}$$

$$x - momentum \qquad \rho\left(\frac{\partial \bar{u}_i}{\partial t} + \bar{u}_k \frac{\partial \bar{u}_i}{\partial x_k}\right) = -\frac{\partial \bar{p}}{\partial x_i} + \frac{\partial}{\partial x_j}\left(\mu \frac{\partial \bar{u}_i}{\partial x_j}\right) + \frac{\partial R_{ij}}{\partial x_j} \tag{4a}$$

$$y - momentum \qquad \rho\left(\frac{\partial \bar{v}_i}{\partial t} + \bar{v}_k \frac{\partial \bar{v}_i}{\partial y_k}\right) = -\frac{\partial \bar{p}}{\partial y_i} + \frac{\partial}{\partial y_j}\left(\mu \frac{\partial \bar{v}_i}{\partial y_j}\right) + \frac{\partial R_{ij}}{\partial y_j} \tag{4b}$$

$$z - momentum \qquad \rho\left(\frac{\partial \bar{w}_i}{\partial t} + \bar{w}_k \frac{\partial \bar{w}_i}{\partial z_k}\right) = -\frac{\partial \bar{p}}{\partial z_i} + \frac{\partial}{\partial z_j}\left(\mu \frac{\partial \bar{w}_i}{\partial z_j}\right) + \frac{\partial R_{ij}}{\partial z_j} \tag{4c}$$

The averaging of the equations results in additional unknowns deemed the Reynolds stress tensor. This tensor R_{ij} becomes the primary focus and challenge of the RANS solution technique.

Reynolds stress tensor

$$R_{ij} = -\rho\overline{u_i' u_j'} = \begin{pmatrix} -\overline{\rho u'^2} & -\overline{\rho u' v'} & -\overline{\rho u' w'} \\ -\overline{\rho u' v'} & -\overline{\rho v'^2} & -\overline{\rho v' w'} \\ -\overline{\rho u' w'} & -\overline{\rho v' w'} & -\overline{\rho w'^2} \end{pmatrix}$$

with 6 unknowns:

$$-\overline{\rho u'^2}$$

$$-\overline{\rho u' v'}$$

$$-\overline{\rho u' w'} \tag{5}$$

$$-\overline{\rho v'^2}$$

$$-\overline{\rho v' w'}$$

$$-\overline{\rho w'^2}$$

The original governing equations had four equations and four unknowns, but the six additional unknowns introduced by the averaging, means additional equations are required to close a solution to the RANS

approach. The various turbulence models available in a RANS approach range in the number of added equations but are all attempting to close the solution by introducing additional empirically derived equations which cannot be derived from the governing principles described by Fluid Mechanics. These added equations are the result of expert judgment and experimental calibration. Fluent has two techniques available for closing the problem, the Reynolds-Stress Model (RSM) and the Eddy Viscosity Model (EVM). The EVM is utilized by the k- ε turbulence model chosen for this study will be discussed further, while the RSM will not.

Realizable k-ε Turbulence Model

The two-equation, eddy viscosity model used in the $k - \varepsilon$ turbulence model first proposed by Chien [21], makes an assumption regarding the shear stress by introducing a new term known as the turbulent viscosity, μ_T. This allows the stress tensor to be written in the following manner:

$$Boussineq\ hypothesis \qquad R_{ij} = -\rho\overline{u_i' u_j'} = \mu_T\left(\frac{\partial \bar{u}_i}{\partial x_j} + \frac{\partial \bar{u}_i}{\partial x_i}\right) - \frac{2}{3}\mu_T \frac{\partial \bar{u}_k}{\partial x_k}\delta_{ij} - \frac{2}{3}\rho k \delta_{ij} \tag{6}$$

This assumption is used to close the RANS model and results in two additional transport equations that are used to track the generation, k, and dissipation, ε, of turbulent energy in the flow.

$$k - transport\ equation \qquad \rho\frac{Dk}{Dt} = \frac{\partial}{\partial x_j}\left[\left(\mu + \frac{\mu_t}{\sigma_k}\right)\frac{\partial k}{\partial x_j}\right] + \mu_t S^2 - \rho\varepsilon \tag{7}$$

$$where \quad S = \sqrt{2S_{ij}S_{ij}}$$

$$\varepsilon - transport\ equation \qquad \frac{D\varepsilon}{Dt} = \frac{\partial}{\partial x_j}\left[\left(\mu + \frac{\mu_t}{\sigma_\varepsilon}\right)\frac{\partial \varepsilon}{\partial x_j}\right] + \frac{\varepsilon}{k}(C_{1\varepsilon}\mu_t S^2 - \rho C_{2\varepsilon}\varepsilon) \tag{8}$$

$$coefficients \qquad \sigma_k, \sigma_\varepsilon, C_{1\varepsilon}, C_{2\varepsilon} \rightarrow emperically\ derived$$

$$turbulent\ viscosity \qquad \mu_t = \rho C_\mu \frac{k^2}{\varepsilon} \tag{9}$$

From here, the turbulent kinetic energy can be calculated from the average fluctuations in all special directions, and indicates the level of turbulence due to the average velocity fluctuations computed by the turbulence model. Additional turbulence inputs to the turbulence model are used to calibrate the model for problem specific flow conditions. These inputs are used to specify the level of turbulent energy generation and dissipation in the flow. The turbulence intensity input, I, is used to specify the ratio of fluctuating velocity to the instantaneous flow velocity. The turbulent length scale input, l, is used to specify the ratio of turbulent energy generation to the dissipation, based on a characteristic length of the problem. Both of these turbulence model inputs were used in the twin jet model, and discussed further in Chapter 6.

$$turbulent\ kinetic\ energy \qquad k = \frac{1}{2}\left(\overline{u'^2} + \overline{v'^2} + \overline{w'^2}\right) \tag{10}$$

$$turbulence\ intensity \qquad I = \frac{u'}{U} \approx \frac{1}{U}\sqrt{\frac{2k}{3}} \tag{11}$$

$$turbulent\ length\ scale \qquad l = \frac{k^{2/3}}{\varepsilon} \tag{12}$$

To solve the RANS equations, they are discretized into a system of algebraic equations that can be solved numerically on a computer. The discretization technique plays a critical role when considering the order of accuracy in a CFD model, analyzed with detail in Chapter 5.

3.3 Discretization

To solve the RANS equations numerically, they must be setup as a system of algebraic equations that can be numerically computed. Fluent uses the finite volume technique to compute the flowfield, which is distinct from a finite differencing technique. This distinction is primarily focused on the form of the equations that are solved. In the finite volume method, the integral form of the equations are solved over the surface of a control volume, while the finite difference method solves the differential form of the equations at a point. The two different forms of the continuity equation are presented as an example of what each technique is addressing when solving numerically.

$Integral\ Form$ (Finite Volume)	$\frac{\partial}{\partial t}\int_{CV} \rho\, dV + \int_{CS} \rho \boldsymbol{V}\, d\boldsymbol{A} = 0$
$Differential\ Form$ (Finite Difference)	$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \boldsymbol{V}) = 0$

Fluent employs the finite volume technique and solves the integral form of the RANS equations, which involves additional integration that is not required in the differential form. In both cases, however, a discretization technique is required to approximate the solution to the equations. This is where the order of accuracy of the method used to make the approximation becomes very important. To demonstrate the principles of the differencing method, a second-order central difference scheme that is similar to that used by the Fluent solver is considered for a simple example in 1D. Figure 8 shows the setup for this example, and is used to demonstrate how the order of accuracy can be determined due to discretizing a partial differential equation into an algebraic equation that can be solved numerically.

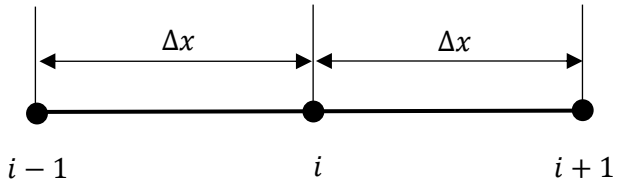


Figure 8: Example of second order accuracy in discretization of a partial differential equation using the central difference method.

To determine the order of accuracy in a central difference discretization scheme, consider how the x – direction velocity $(\partial u/\partial x)$ at point (i) can be calculated using the points behind and in front, $(i-1)$ and $(i+1)$. The order of accuracy can be demonstrated by performing a Taylor series expansion about the point at (i) as follows.

First the expansion is applied from point i to point $i+1$:

$$u_{i+1} = u_i + \left(\frac{\partial u}{\partial x}\right)_i \Delta x + \left(\frac{\partial^2 u}{\partial x^2}\right)_i \frac{(\Delta x)^2}{2} + \left(\frac{\partial^3 u}{\partial x^3}\right)_i \frac{(\Delta x)^3}{6} + \cdots \tag{13}$$

Solving for the first order derivative, one obtains,

$$\left(\frac{\partial u}{\partial x}\right)_i = \underbrace{\frac{u_{i+1} - u_i}{\Delta x}}_{\textit{Finite difference}} \underbrace{- \left(\frac{\partial^2 u}{\partial x^2}\right)_i \frac{\Delta x}{2} - \left(\frac{\partial^3 u}{\partial x^3}\right)_i \frac{(\Delta x)^2}{6} + \cdots}_{\textit{Truncation terms}} \tag{14}$$

Next, the expansion is applied from point i to $i-1$:

$$u_{i-1} = u_i - \left(\frac{\partial u}{\partial x}\right)_i \Delta x + \left(\frac{\partial^2 u}{\partial x^2}\right)_i \frac{(\Delta x)^2}{2} - \left(\frac{\partial^3 u}{\partial x^3}\right)_i \frac{(\Delta x)^3}{6} + \cdots \tag{15}$$

Again, solving for the first order derivative, one obtains,

$$\left(\frac{\partial u}{\partial x}\right)_i = \frac{u_i - u_{i-1}}{\Delta x} + \left(\frac{\partial^2 u}{\partial x^2}\right)_i \frac{\Delta x}{2} - \left(\frac{\partial^3 u}{\partial x^3}\right)_i \frac{(\Delta x)^2}{6} + \cdots \tag{16}$$

Subtracting (13) from (15):

$$\begin{aligned} u_{i+1} - u_{i-1} = &\left[u_i + \left(\frac{\partial u}{\partial x}\right)_i \Delta x + \left(\frac{\partial^2 u}{\partial x^2}\right)_i \frac{(\Delta x)^2}{2} + \left(\frac{\partial^3 u}{\partial x^3}\right)_i \frac{(\Delta x)^3}{6} + \cdots\right] \\ &- \left[u_i - \left(\frac{\partial u}{\partial x}\right)_i \Delta x + \left(\frac{\partial^2 u}{\partial x^2}\right)_i \frac{(\Delta x)^2}{2} - \left(\frac{\partial^3 u}{\partial x^3}\right)_i \frac{(\Delta x)^3}{6} + \cdots\right] \end{aligned} \tag{17}$$

Once again, solving for the first order derivative,

$$\left(\frac{\partial u}{\partial x}\right)_i = \frac{u_{i+1} - u_{i-1}}{2\Delta x} \underbrace{+ 0(\Delta x)^2}_{2^{nd}\ \textit{order accurate}} \tag{18}$$

The "truncation terms" shown in (14) are neglected and the central difference approximation for the derivative is computed using u_{i+1} and u_{i-1}. Since the truncation terms are $0(\Delta x)^2$, the method is said to be "second order" accurate.This example shows that the discretization of the PDE describing the *x*-

velocity into an algebraic equation at a single point, carries an order of accuracy due to the truncation of the series expansion. The implementation of the discretization is slightly different for the finite volume method, however, the order of accuracy that was demonstrated is in principle, the same. The process shown in this example is applied to all the terms in the RANS model, which transform the equations into a complete system of algebraic equations. The order of accuracy due to this transformation of the equations will be considered in detail in Chapter 5, and is known as the theoretical order of accuracy that will later be used to estimate the discretization error in the twin jet CFD solution.

4 MODELING TWIN JET FLOW

The purpose of this chapter is to provide specific details regarding the approach and implementation of the twin jet CFD model that will be used for verification, validation, and uncertainty quantification in Chapters 5 and 6.

4.1 Model Approach

The approach that was adopted in this CFD model was a balance between the objective to quantify the uncertainty in predicted SRQs, and the limits of the tools available for the study. The primary tools that were utilized in the twin jet CFD model are listed in Table 2.

Table 2: Details and specifications of hardware and software tools that were available and used in the development of a Fluent CFD model.

Hardware:	
Dell Precision Workstation	Windows 64-bit OS with 12 cores and 32-GB of memory. — (2) Intel Xeon 2.1 GHz processors with 6 cores each — 12 physical cores and 24 logical processes
Monsoon Cluster	Red Hat Enterprise Linux OS with 884 cores and 12-TB of memory. — Intel Xeon 2.5 GHz Haswell nodes (24 cores per node) Memory: 64 GB per socket × 2 sockets = 128 GB per node L1 instruction cache: 32 KB, private to each core L1 data cache: 32 KB, private to each core L2 cache: 256 KB, private to each core L3 cache: 30 MB, shared by 12 cores in each socket Instruction Sets: Advanced Vector Extensions (AVX2) 256-bit floating-point vector registers 16 floating-point operations per core — Interconnected via FDR Infiniband at a rate of 56 Gbps
Software:	
ANSYS Workbench 16.2	Modules include: Design Modeler, Meshing, Fluent + 120 Parallel HPC Licenses.
DAKOTA 6.5	Design Analysis Kit for Optimization and Terascale Application (DAKOTA) available from Sandia National Laboratory.
FieldView 16.1	Post-processing software for large scale CFD modeling available and provided by Intelligent Light.
MATLAB 2015b	Matrix Laboratory programming software by MathWorks.

The initial model setup and troubleshooting was performed on the Dell Precision workstation, before being scaled up and sent to the Cluster (designated *Monsoon*) for numerical computations. The solution data generated by the model was post-processed using FieldView for visualization and sent back to DAKOTA as a response quantity in the uncertainty quantification (UQ) analysis. Figure 9 shows a general overview of the workflow approach that was taken (see Chapter 7 for more details on this workflow).

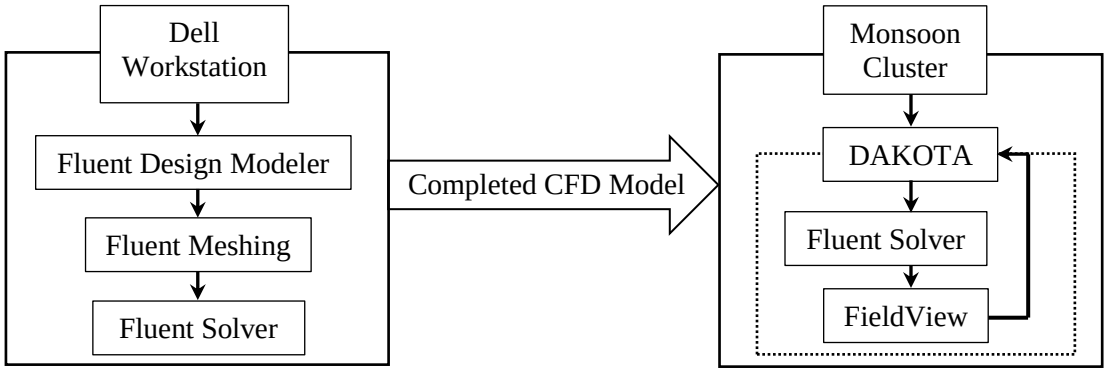
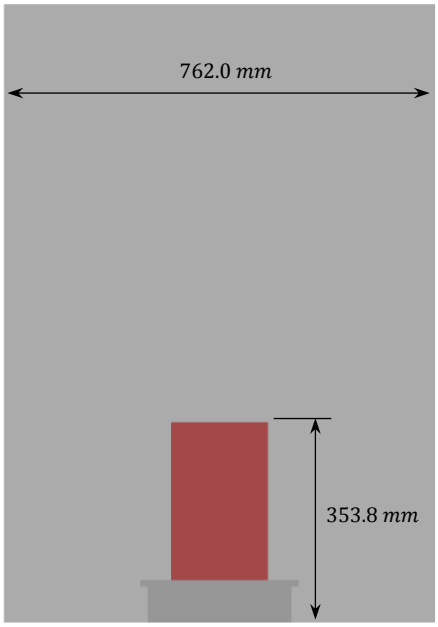


Figure 9: Workflow of the model approach showing the combined relationship of the Dell Workstation and Linux Cluster used to conduct the computational workload.

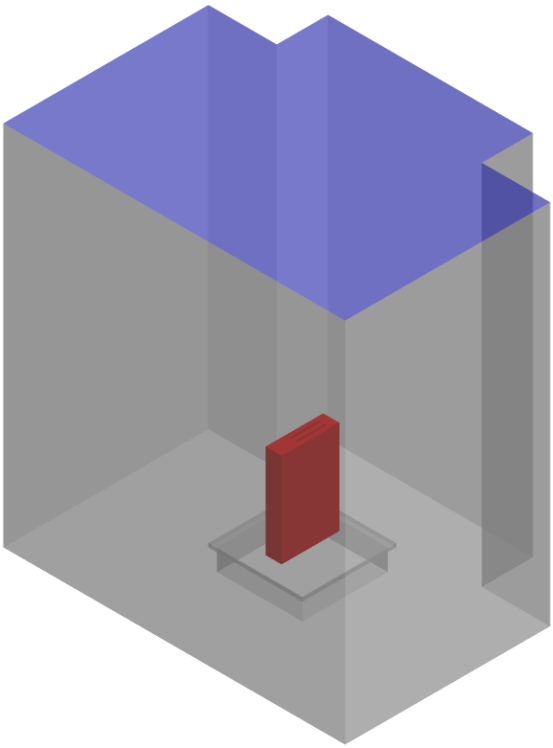
This model approach incorporated the tools available (hardware and software) with the goal of quantifying the uncertainty in system response quantities predicted by the model. The following sub-sections provide specific details pertaining to design and implementation of the model.

4.2 Geometry

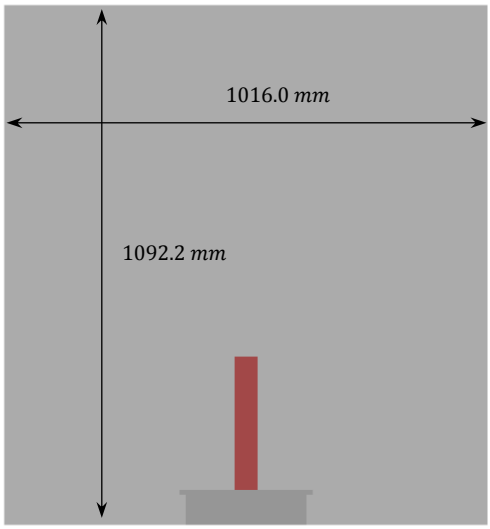
The geometry configuration of the CFD model was based upon the physical conditions of the twin jet validation experiment conducted by Wang and Hassan [1] that was first presented in Figure 1 from Chapter 2. The specification for the nozzle and tank used in the experiment were taken from a subsequent document by Wang and Hassan [2] which provided additional information regarding the geometry and configuration of the twin jets (see 0). The model geometry shown in Figure 10 and Figure 11 was created to match the experimental conditions as closely as possible (see Figure 53).



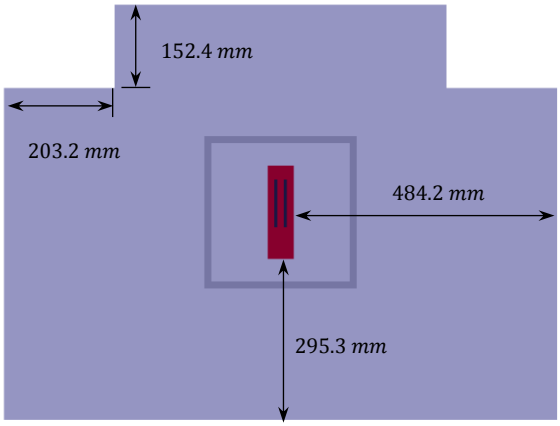
(a) side-view



(b) isometric-view

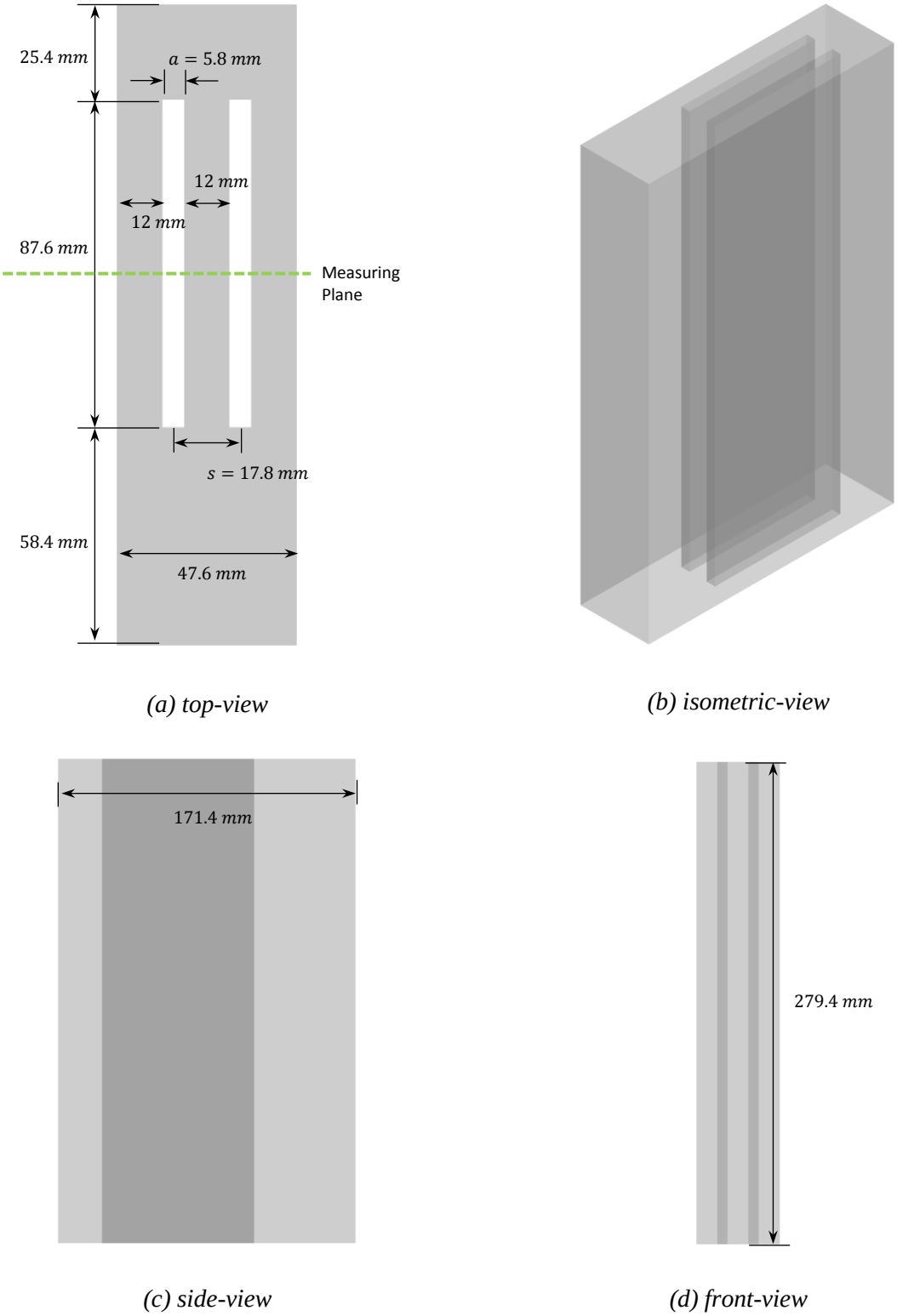


(c) front-view



(d) top-view

Figure 10: Geometry of the domain from various viewpoints including dimensions used in the model.



(a) top-view

(b) isometric-view

(c) side-view

(d) front-view

Figure 11: Geometry of the twin jet nozzle from various viewpoints. Note the nozzle width is specified by the variable "a" and the spacing between the centerline of each nozzle if denoted as "s".

The modeling domain was created using the Design Modeler module included in the ANSYS Workbench package used in this study. A simplified geometry was used for the 2-D case and can be found in Appendix A.

4.3 Meshing the Geometry

Once the geometry was finalized, the next step was to generate a mesh. The mesh or grid was designed to focus additional solution points in the region above the jet inlets. As shown in Figure 12, the mesh was uniformly spaced using hexahedral cells in the region surrounding the jet inlets. Moving away from the inlets, the mesh transitioned to non-uniform tetrahedral cells which continue to grow in size based on the proximity to the nozzle inlets. The less computationally expensive tetrahedral cells were chosen for the region further away from the nozzle, where very little flow was expected. The more accurate but computationally expensive hexahedral cells, were used to focus in on the regions where high gradients and significant change was expected. This design was chosen to reduce computational cost by limiting the use of uniformly spaced hexahedral cells to the region near the nozzle inlets.

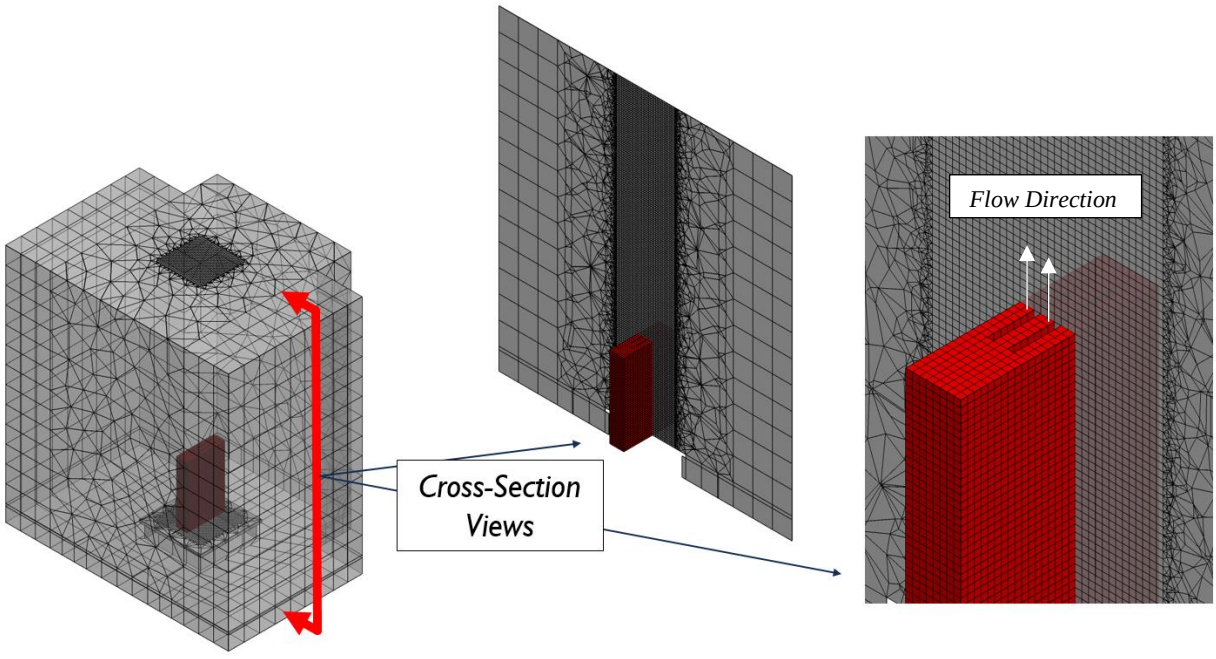


Figure 12: Cross-sectional view of the hexahedral/tetrahedral mesh with uniform spacing the region near the nozzle inlet.

4.4 Boundary Conditions

The boundary conditions utilized by the CFD model were selected in accordance with those documented in the validation experiment. A no-slip condition was enforced on the tank and nozzle walls, and the flow-rate was specified at the inlet of each jet. Since the outlet did not have a significant impact on the interaction of the twin jets [14], a pressure-outlet conditions was applied to the top of the tank. The boundary conditions and values that were used to generate the nominal results are shown in Figure 13 and listed in Table 3.

Table 3: List of conditions applied to each of the model boundaries in the Fluent model.

Type	*Value*	*Units*
Inlets:		
Mass-flow	0.385	$[kg/s]$
Turbulence Intensity	8	[%]
Turbulent Length Scale	5.8	$[mm]$
Pressure-outlet:	0	$[Pa]$
Wall:	$no-slip$	-

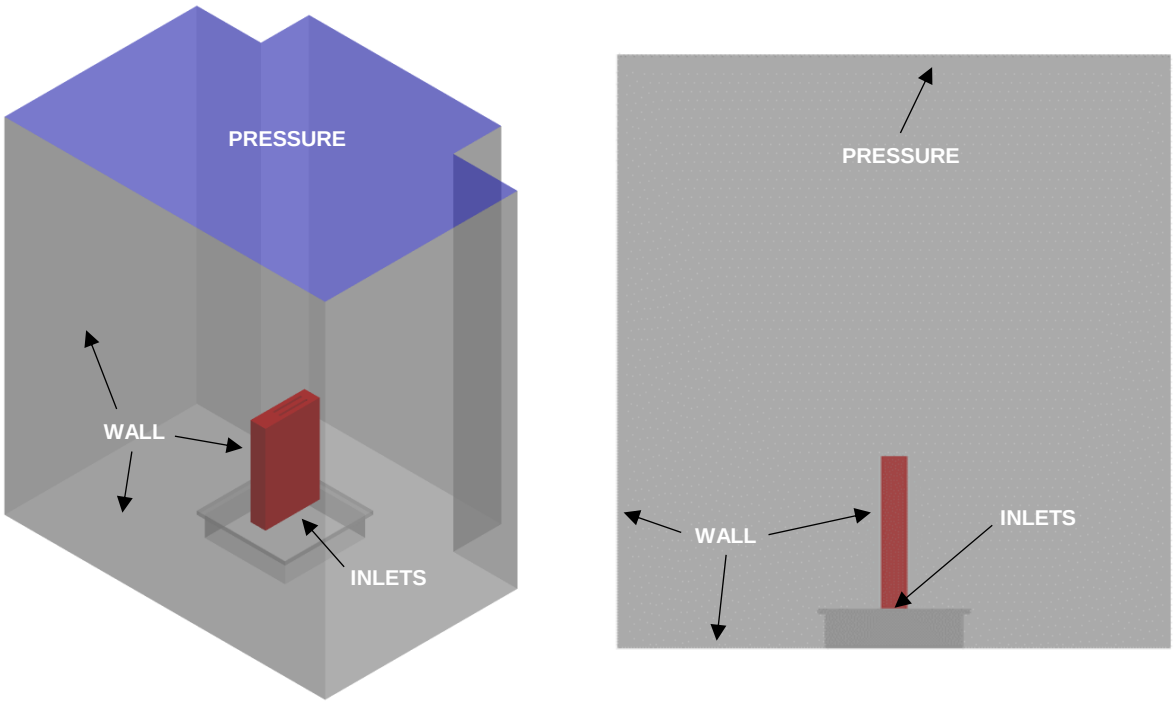


Figure 13: Boundary conditions utilized by the model with respect to the modeling domain.

4.5 System Response Quantity

For this study, the SRQ of interest was the velocity of the fluid in the axial direction at specific points in the flow. The validation experiment documented the mean axial velocity at 23 point locations along each of the two lines shown in Figure 14. The CFD model will be validated using the experimental data available at these locations. The green measuring plane represents the locations where the velocity was collected from the model and used in the verification and validation study. This plane included 4318 node locations which were used to study the flow field predicted by the model. For clarity, the coordinates of all points in the flowfield were nondimensionalized by the jet diameter, $a = 5.8mm$ (see Figure 11 for definition of "a"). The origin for the nondimensional coordinate can be seen as the nozzle inlet to the tank in Figure 14.

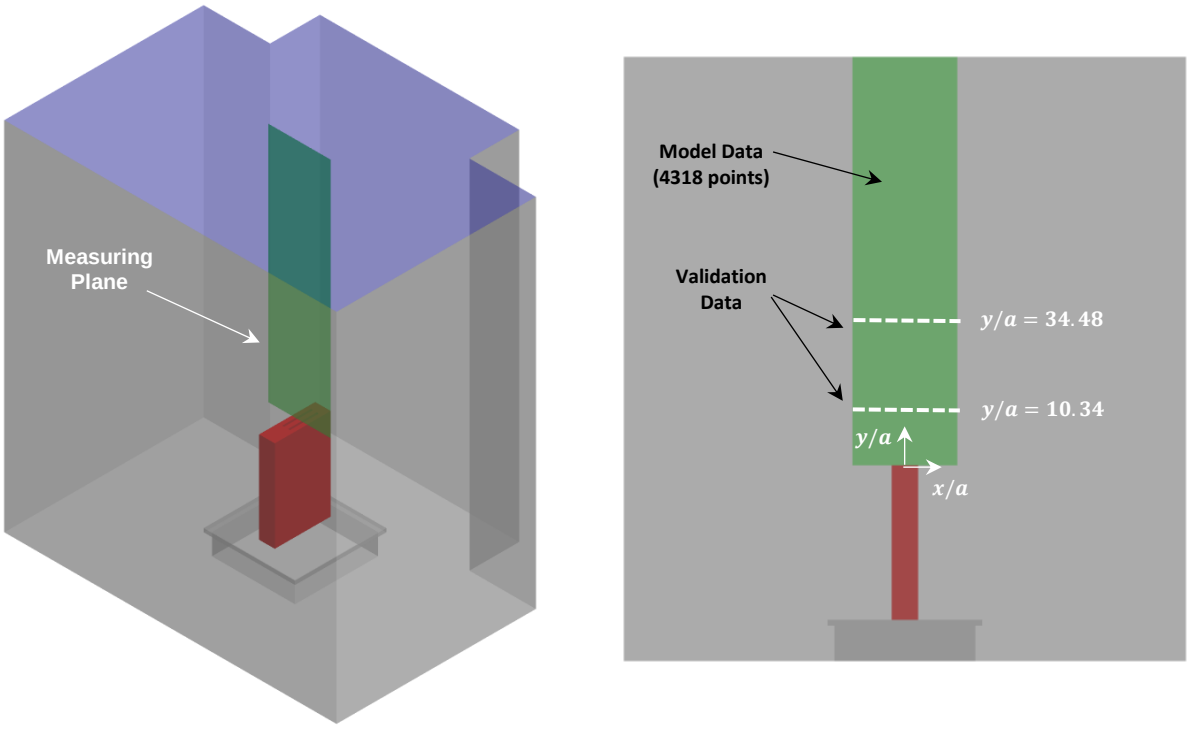


Figure 14: Location of system response quantity used in the validation and uncertainty study with respect to the modeling domain.

4.6 Convergence

Fluent reported the "residuals" pertaining to fundamental quantities being solved by the numerical model, such as mass and momentum flux, from each iteration performed. The residuals were reported based on the amount of mass or momentum entering or leaving the control volume of each cell. One indication of solution convergence was the reduction of these residuals by 3-4 orders of magnitude, or to a number that was on the order of 10^{-3} or less. Additionally, the velocity was also checked for convergence. The velocity at the centerline between the jets at a nondimensional distance of $y/a = 34.48$ was used to determine solution convergence in addition to reduction in residuals. Figure 15 demonstrates a properly converged simulation for the velocity, since the solution is no longer changing as the solver iterates. To ensure that the velocity would not change as the solver iterates, the simulation was run far past the point when the velocity no longer changed.

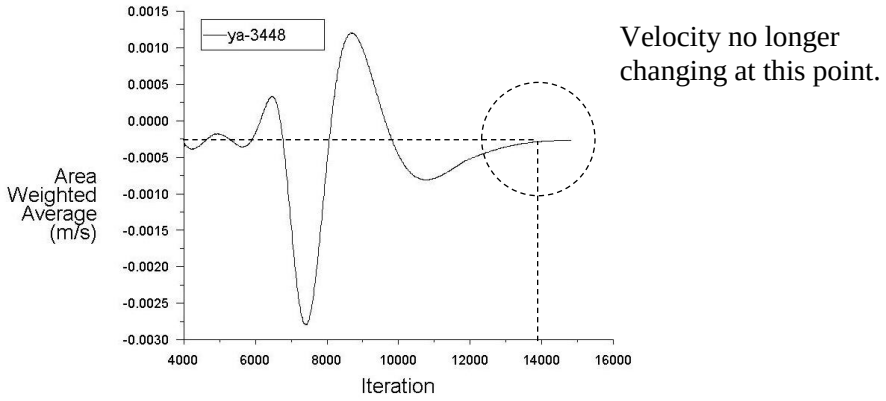


Figure 15: Example of a converged solution by monitoring the velocity at a point in the flow field.

4.7 Solver Settings

The models and settings implemented in Fluent were determined in conjunction with documentation from ANSYS and experience running the solver under various configurations. The final models and solver settings that were selected and reported here, were found to be well suited for the twin jet problem of interest. An overview of these top-level solver settings are listed in Table 4 and Table 5. For a full report summary of the specific setting including all lower level options, see Appendix G.

Table 4: Solver models chosen for the treatment in space, time and viscosity by the Fluent solver.

Models	**Settings**
Space	3D
Time	Steady
Viscous	*Realizable k-epsilon turbulence model*
	$C_\mu = 0.09; C_1 = 1.44; C_{2\varepsilon} = 1.92; \sigma_k = 1.0; \sigma_\varepsilon = 1.3$
Wall Treatment	Enhanced Wall Treatment

Despite the unsteady nature of this twin jet problem, the steady fluent solver was found to work well. The constraint introduced by this selection was that true unsteady characteristics were not captured. However, several runs using the unsteady Fluent solver did not provided better information about the average velocity field. Therefore, the steady solver was selected for this model. Fluent solved the set of RANS equation for flow and turbulence discussed in Chapter 3. The enhanced wall treatment (EWT) option was chosen to instruct the solver how to properly resolve the flow at locations in the mesh that were near a wall. This option is only required for the k-ε turbulence models, and is a robust technique that was recommended by the Fluent documentation. The second-order accuracy method was selected for all terms in the discretization procedure. Since the method of discretization was second order, the theoretical order of accuracy for the method was two, as was used as key metric in the following Chapter, when assessing the numerical error.

Table 5: Fluent solver settings which specify the order of accuracy used in the discretization discussed in Chapter 3.

Solver Settings	**Description**	**Value**
Equations Solved	Flow (RANS)	-
	Turbulence	-
Pressure-Velocity Coupling	SIMPLE	-
Discretization Scheme	Pressure	*Second Order*
	Momentum	*Second Order Upwind*
	Turbulent Kinetic Energy	*Second Order Upwind*
	Turbulent Dissipation Rate	*Second Order Upwind*

5 VERIFICATION

5.1 Code Verification

Code verification involves demonstrating that a numerical solution generated by a numerical model is able to solve a problem with a known exact solution and obtain or approach the exact answer. This typically involves problems with a relatively simple geometry and known exact analytical solutions to decide if the CFD code answer is correct. Once this has been demonstrated, the CFD code is said to be verified [4]. To verify the commercial CFD code Fluent, the relevant equations that are being solved must be identified and examined. A grid refinement study would then be performed to exercise the terms of interest and determine if the code has any errors, or deviates unexpectedly from the analytical or known solution. If issues were found with the code, modification would be made to fix the problem. The goal of code verification is to provide an evaluation of the code error and ensure there is no programming issues in the numerical solution. Since Fluent is a well-established commercial CFD code, the code verification step was deemed unnecessary in this work, which instead focused on the solution verification, validation and uncertainty quantification.

5.2 Solution Verification

Verification of the Fluent solution involved quantifying the numerical error due to discretization of the partial differential equations discussed in Chapter 3. Numerical error also includes round-off and iterative error. Round-off error is due to rounding of floating point numbers by the computer. This was reduced to a very small amount by using double precision in all the calculations, which reduced the rounding error such if could be neglected. Iterative error is due to incomplete solution iterations when computing the numerical solution. This was also reduced to a small amount by running the solution and monitoring the residuals to ensure iterations were not changing the solution result, such that it could also be neglected. By reducing the round-off and iterative error to very small values, the discretization error was considered to be the primary source of numerical error. While numerical error refers to all three sources of error that were mentioned, the numerical error was considered to be primarily due the discretization. During the

setup of the model discussed in Chapter 4, it was shown that a second order method was chosen for the discretization scheme, which meant that the theoretical order of accuracy for this model was two. As shown in Chapter 3, the theoretical order can be determined from the truncation of a Taylor series expansion of PDEs converted to a system of algebraic equations. To estimate the numerical error, a study of the grid at each point where the solution is solved was required. If the divisions between grid points are equally spaced, the numerical error due to discretization can be found by refining the grid and analyzing the relationship between the refinement and solution.

Discretization Error

To demonstrate the technique of grid refinement as a tool for estimating the discretization error, consider the simple example presented in Figure 16. In this case, the grid is refined in a uniform manner in all directions. To find the refinement factor in this example, a characteristic grid dimension h, is chosen as follows.

$$h = \sqrt{(\Delta x)^2 + (\Delta y)^2 + (\Delta z)^2} \tag{19}$$

Once h is selected, the refinement factor r, is defined based on the characteristic h for each grid level.

$$r = h_1/h_2 \tag{20}$$

In this example, the refinement factor is two, since the number of divisions are successively doubled as the grid is refined about the discrete point. The numerical model is run separately for each of the grid levels shown, and the value of the solution at the discrete point for each grid is considered.

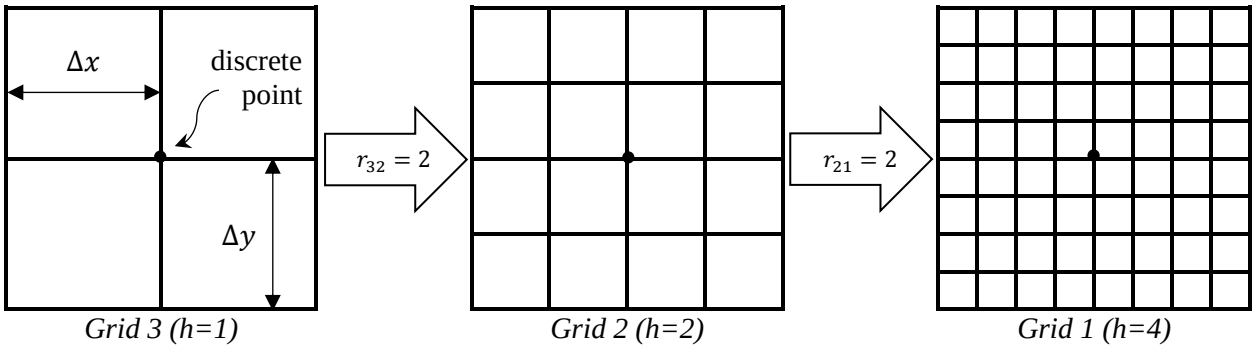


Figure 16: Example of grid refinement to determine the discretization error of a numerical solution.

The procedure for analyzing the solution between grid refinement levels and to estimate the numerical error due to discretization was described in the ASME V&V 20 standard [4]. The first step is to calculate the observed or apparent order of accuracy for the solution between grids. The observed order is distinct from the theoretical order discussed previously. The observed order is the measure of the actual order of accuracy that is exhibited or observed as the grid is refined. It is important to identify the observed order through a refinement study to compare with the theoretical order that is chosen when selecting the discretization scheme. If the observed order is similar to the theoretical order, the solution is said to be asymptotic, and the error due to discretization can be found using the Grid Convergence Index (GCI) method. On the other hand, if the observed order is much less than the theoretical order, the solution is said to be oscillatory, and the GCI cannot be used.

The procedure for finding the observed order of the Fluent CFD model was found in the manner discussed previously. Since the grid refinement index, r, chosen for this study was two, calculation of the observed (or apparent) order $\hat{p}$ was found as follows,

$$\hat{p} = \frac{\ln(\varepsilon_{32}/\varepsilon_{21})}{\ln(r_{21})} \tag{21}$$

where $\varepsilon_{32} = \varphi_3 - \varphi_2$ and $\varepsilon_{21} = \varphi_2 - \varphi_1$. Here, φ is the value of the simulation variable of interest, which in this case was the axial velocity at key locations in the domain where experimental validation data was collected. The numerical subscript on φ, refers to the refinement level of the grid, with φ_3 being the course grid, and φ_1 being the fine grid. Once a value for the observed order p was obtained, the grid convergence index was then calculated as,

$$GCI = \frac{Fs \cdot |\varphi_1 - \varphi_2|}{r_{21}^{\hat{p}} - 1} \tag{22}$$

where $Fs = 1.25$ is a factor of safety recommended by Roache in [6], based on his observations made in applying this technique to hundreds of numerical simulations. As mentioned previously, the GCI method was only applicable when the observed order was close to the theoretical order. When the observed order was not close to the theoretical, an alternative method was used. The work by Eca and Hoekstra [7]

suggested an alternative method to address the numerical error when the solution is not asymptotic. To estimate the numerical uncertainty when the observed order was less than the theoretical order, the following procedure called the EH method was employed,

$$EH = 3\Delta M \tag{23}$$

where

$$\Delta M = |\varphi_3 - \varphi_1|$$

For the purposes of estimating the numerical error in the Fluent model, either the GCI or EH method was used depending on the observed order of the three grid solutions.

$$if\ \hat{p} \geq 1 \rightarrow use\ GCI$$

$$if\ \hat{p} < 1 \rightarrow use\ EH$$

Since the theoretical order for this case was two, if the solution exhibited an order of accuracy less than one, it was considered oscillatory, and the EH method was applied. If the order was greater than one, asymptotic behavior was assumed, and the GCI method was used. The details of the grid and refinement levels used in the numerical estimation of the Fluent model are discussed next.

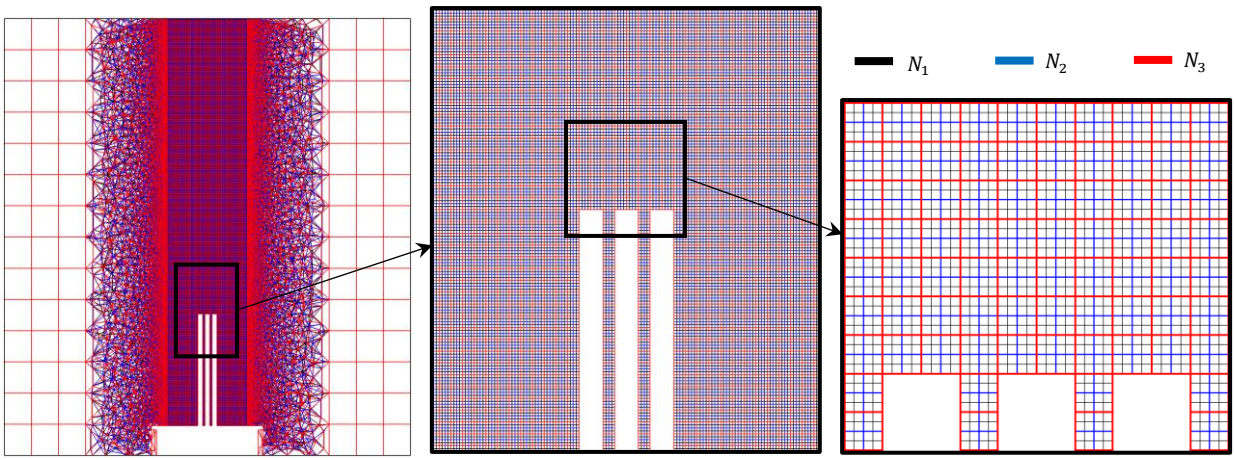
Grid Refinement

Once the geometry was finalized to the specifications shown in Chapter 4, a series of grids were generated using the meshing application available in ANSYS Workbench, i.e, ANSYS Meshing. The mesh design was chosen with the goal of estimating the numerical error in mind. To provide an estimate of the numerical error using techniques described in [4], the grid was spaced uniformly in all spatial directions as done in the example at the beginning of this chapter. As the grid was refined by two in each direction ($r = 2$), each refinement level was exactly superimposed on the last, as shown in Figure 17. This meant that each node location on the coarse mesh, shared a node with every grid refinement level that followed. This technique provided superior numerical error estimation, by reducing any additional uncertainty due to interpolation between grids. To reduce computational cost, only the region surrounding

the nozzle was refined and uniformly spaced. This was a practical choice due to the high gradients in the near jet region that was of particular interest in the model. The spacing and refinement for each mesh is detailed in Table 6, which includes the number of cells that span across each jet. Additional images of the three grid that were used to estimate the numerical uncertainty in the CFD model can be found in Appendix D.

Table 6: List of mesh details including refinement factors and number elements in each grid for the Fluent model.

Grid Number, N	Refinement Factor, r	Number of Cells	Cells Across Jet Diameter	File Size (MB)
N_3	2	~250,000	1	~40
N_2	2	~4,500,00	2	~330
N_1	2	~23,00,000	4	~1,900



(a) Full-domain *(b) Localized refinement region* *(c) Superimposed grids in near-jet region.*

Figure 17: Mesh configurations with three refinement levels used to estimate the numerical error.

6 UNCERTAINTY QUANTIFICATION

6.1 Structure of UQ Analysis

The structure that was implemented in the uncertainty analysis of the Fluent model relied heavily on the work of the ASME V&V20 [4] and the work of Roy and Oberkampf [8]. Following the procedure outlined in these documents, uncertainty in the model prediction of the SRQ was composed of the following contributors:

- Numerical Uncertainty, U_{num}
- Model Input Uncertainty, U_{input}
- Experimental Uncertainty, U_D
- Validation Uncertainty, U_{val}

Up to this point, only techniques for obtaining the numerical uncertainty were discussed in the verification of the Fluent model. Each of these four uncertainty contributions needed to be considered separately, and then combined to form the final estimation of the overall model uncertainty in a given SRQ. The total uncertainty in the Fluent model prediction of the SRQ, was computed as,

$$U_{SRQ,\%} = U_{num,\%} \pm U_{input,\%} \pm U_{val,\%} \pm U_{D,\%} \tag{24}$$

where $U_{SRQ,\%}$ represents the interval that bounds the model prediction with a specific "%" confidence level. The following sections outline the steps that were followed to quantify each of the contributions to the uncertainty in the Fluent model prediction of the axial velocity.

6.2 Input Uncertainty

Quantifying the input uncertainty was the most computationally expensive component of the uncertainty quantification. The contribution of input uncertainty, U_{input} was determined by the interval generated in the response quantity due to varying the model inputs. As mentioned in the Chapter 2, the methods for quantify uncertainty required that the uncertain parameters be classified as either aleatory or epistemic.

The explanation for the classification of input parameters to the Fluent model are documented in the following sub-sections.

Deterministic

Inputs to the model that were known with a high degree of confidence were not considered as a source of uncertainty and classified as *deterministic*. This was a necessary step to reduce the computational cost of every additional parameter considered with uncertainty. In reality, there was some degree of uncertainty in these parameters, however, judgment was used to eliminate parameters that would have no significant impact on the SRQs. This allowed for carefully considered parameters to be treated as a single fixed value, listed in Table 7. For this case, the values applied in the Fluent model for density and viscosity were treated as deterministic and were not considered as a source of input uncertainty.

Table 7: Values of the deterministic model inputs for the density and viscosity of water. [22]

Model Input	**Type**	**Value**	**Units**
Fluid density, ρ	*Deterministic*	997.5	[kg/m^3]
Fluid viscosity, ν	*Deterministic*	9.39e-7	[m^2/s]

Aleatory

The model inputs with uncertainty due to random events were classified as *aleatory*. This type of uncertainty is commonly observed while collecting data experimentally. The sensible assumption when considering measurements obtained through experiment is to apply a probability distribution function (PDF) to describe the behavior of a measurement. The PDF specifies the likelihood (probability) of a single value being measured and is based on a large number of supporting measurement samples. The inputs to the Fluent model were selected based upon mean values reported by the experiment, which were then converted to a PDF in order to describe these as aleatory uncertainties.

The nominally reported mass flow rate of each jet, $\dot{m}$, was classified as an aleatory uncertainty because it was described as "fluctuating continuously" during the experiment [1]. Despite additional information, the random fluctuation in flow rate due to the pumps and flow meter were irreducible for the given

experimental setup. To describe the uncertainty in this metric, the mass flow rate of each jet was assumed to follow a normal distribution. To generate a PDF that described the flow rate of each jet, a mean and standard deviation was required. This information was obtained using manufacturer specifications that were available for the GPR TM100 flow meter used in the experiment. The mean value reported by the experiment, and standard deviation found using the flow meter specifications, were used to generate a PDF that described the flow rate of each jet. Figure 18 shows a graphical example of a PDF generated for the mass-flow-rate.

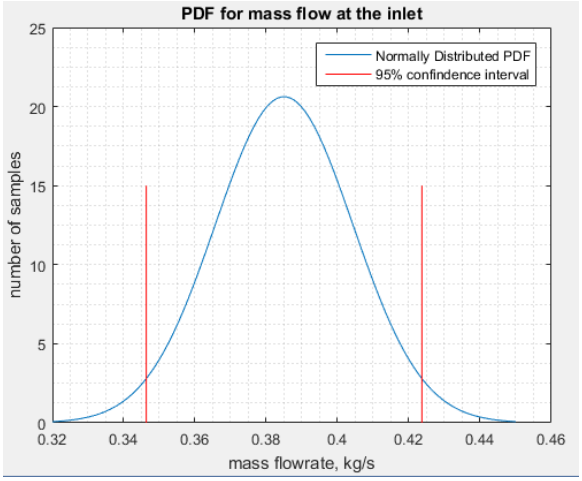


Figure 18: Probability density function for flow rate of jet inlet.

A similar procedure was adopted to generate additional PDFs that described the turbulence intensity and length scale at the jet inlet. For each of the parameters listed in Table 8, a sample was chosen for the boundary condition and a simulation was run. This resulted in a series of simulations (see chapter 7) that were conducted to quantify the uncertainty due to aleatory input uncertainty.

Table 8: List of the aleatory model input parameters that were considered using a normally distributed PDF.

Model Input	**Type**	**Mathematical Structure**	**Mean**, σ	**Stdv**, μ_s
Mass flow per jet, $\dot{m}_1$ & $\dot{m}_2$	*Aleatory*	Normally distributed PDF	0.385	0.011
Jet Turbulence Intensity, I_1 & I_2	*Aleatory*	Normally distributed PDF	8	2
Jet Turbulent Length Scale, l_1 & l_2	*Aleatory*	Normally distributed PDF	5.8	2.32

Epistemic

The model inputs with uncertainty due to a lack of knowledge were classified as *epistemic*. Unlike the aleatory classification, epistemically classified uncertainties can theoretically be eliminated entirely. This serves the purpose of including sources of uncertainty due to assumptions that must be made when modeling a physical environment. This can be demonstrated by the reporting of nominal vs. dimensional length. This distinction can be thought of as the discrepancy in the dimensions of lumber. For example, a single piece of lumber is nominally designated 2 × 4 inches, but when physically measured it is actually 1½ × 3 ¾ inches. There is no uncertainty in the actual dimension, but there is uncertainty in the analyst knowledge of the actual dimension. If additional information was available (analyst could measure the dimension), this uncertainty would be reduced, even removed. It is not be sensible to describe this uncertainty with a probability function as was done in the aleatory classification because while unknown, the true value exists. Instead, the sensible option is to specify the interval over which the true value falls with a specified confidence level based on the analyst knowledge.

Since the nominal dimensions of the twin jet validation experiment were reported, there was epistemic uncertainty in the physical jet width and jet spacing. To address this uncertainty, an interval was selected for the jet diameter and spacing that was believed to capture the true value. Listed in Table 9, these intervals are shown, along with the nondimensional jet spacing ratio s/a.

Table 9: List of the epistemic model input parameters that were considered using an interval quantity and no PDF.

Model Parameter	**Type**	**Value**	**Structure**		**Units**
Jet diameter, a	*Epistemic*	5.8	Interval quantity with no PDF	[5.7, 5.9]	[mm]
Jet spacing, s	*Epistemic*	17.8	Interval quantity with no PDF	[17.7, 17.9]	[mm]
Jet spacing ratio, s/a	*Epistemic*	3.069	Interval quantity with no PDF	[3.000, 3.140]	

A series of Fluent models were created to quantify the epistemic uncertainties and obtain the influence of this uncertainty on the SRQ by varying the jet spacing ratio over a series of nine geometry combinations. These nine s/a combinations were found by dividing the jet width a, and jet spacing s, intervals into

three discrete "bins". The three "bins" chosen for the a interval were 5.7, 5.8, and 5.9 millimeters respectively. The three additional bins that were chosen for the jet spacing interval were 17.7, 17.8, and 17.9 millimeters. All possible combinations of a and s led to nine s/a ratios, with each combination representing a unique geometry configuration of the nozzle. For each of the nine nozzle configurations, a new grid was created to reflect the unique spacing ratio, as shown in Figure 19. This distribution of jet spacing ratios was used to adequately sample the s/a interval and quantify the uncertainty by this lack of knowledge.

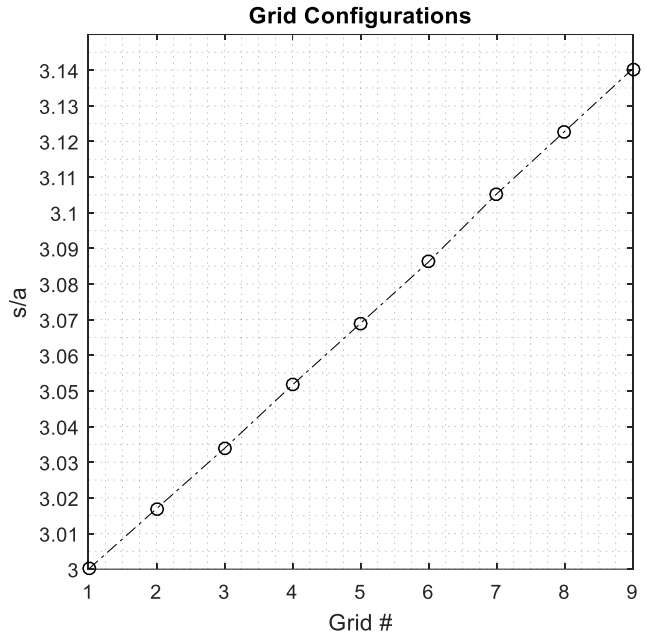


Figure 19: Series of geometry configurations generated by varying the jet spacing ratio (s/a) to obtain the impact of this input uncertainty on the SRQ.

Combing Aleatory and Epistemic Input Uncertainty

This problem was considered to have mixed input uncertainties in that both aleatory and epistemic input uncertainties were present. The cost in terms of computational resources (number of simulations) due the mixed nature of the input uncertainties was significant relative to the computing resources available. For the aleatory uncertainties, 10 values were selected using a Latin hypercube algorithm and the PDFs and then run 10 simulations and record 10 SRQs. This was then repeated on all nine jet spacing ratios to quantify the mixed input uncertainty due to both aleatory and epistemic contributions. The results of this

procedure required a total of 90 simulations that needed to be performed. Depending on the computational budget, mixed (aleatory and epistemic) uncertainty problems such as this may not be feasible, and it may be more realistic to obtain additional information and remove as many epistemic parameters as possible.

The 90 axial velocity values that were recorded as a result of running 90 total simulations, were used to statistically quantify the interval of input uncertainty with a 95% confidence level. This was achieved by determining the absolute maximum and minimum values that were recorded during all 90 simulations that were run. An example of how this was achieved is shown by Figure 20, which shows the mean axial velocity for each of the nine spacing ratios versus the cumulative 95% confidence level. The mean axial velocity values are shown as vertical lines, and correspond to each of the nine spacing ratios. The mean values were calculated using the 10 resulting simulation predictions on each grid. The largest interval due to varying the jet spacing configuration (or grid number) was identified to be grids three and four.

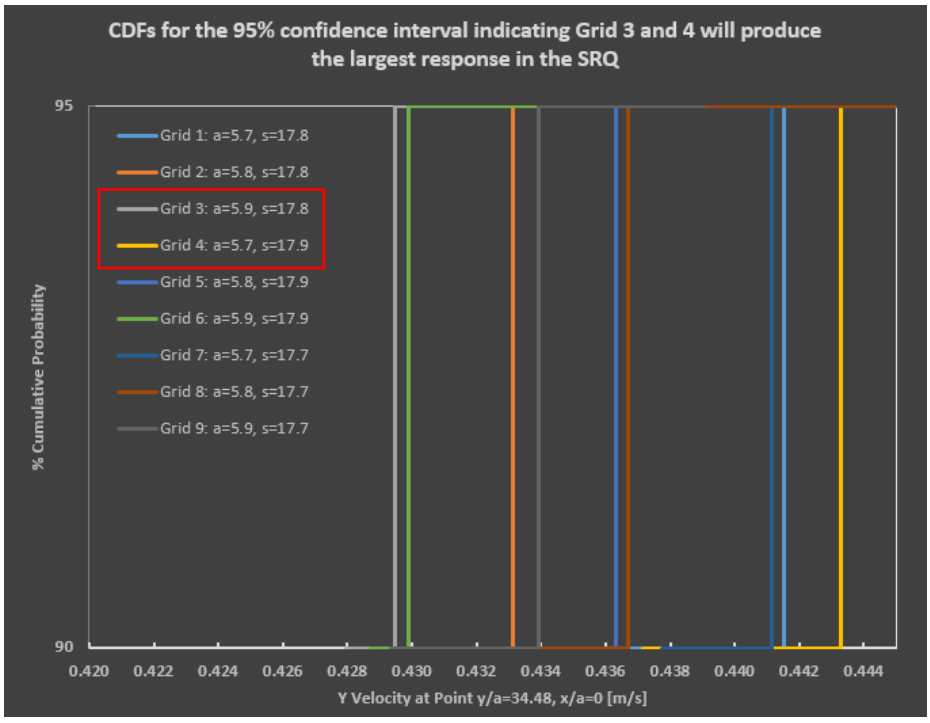


Figure 20: Example of the variation in predicted axial velocity by the model at a single point: x/a=0, y/a=0 by varying the jet spacing ratio s/a.

This visualization tool demonstrated grids 3 and 4 would produce the greatest variation in the predicted axial velocity by the model due to uncertainty in the inputs. The results of this process were able to

capture the largest range of responses due to varying the inputs, and predicted the input uncertainty interval with 95% confidence. This final interval combined the mixed aleatory and epistemic uncertainties and referred to as the p-box in [8]. Since the uncertainty was only considered at the 95% confidence level, the input uncertainty interval was renamed as the p-interval, and used as the starting point for appending the additional uncertainties discussed next. A summary of the model inputs and related uncertainties are presented in Table 10.

Table 10: Summary of the classification, mathematical structure, and values that were used in the study of the model input uncertainty.

Model Input	**Type**	**Structure**	**Mean**, σ	**Stdv**, μ_s	**Units**
Mass flow per jet, $\dot{m}_1$ & $\dot{m}_2$	*Aleatory*	Normally distributed PDF	0.385	0.011	[kg/s]
Jet Turbulence Intensity, I_1 & I_2		Normally distributed PDF	8	2	[%]
Jet Turbulent Length Scale, l_1 & l_2		Normally distributed PDF	5.8	2.32	[mm]
Jet spacing ratio, s/a	*Epistemic*	Interval with no PDF	[3.000, 3.140]		-
Fluid density, ρ	*Deterministic*	Fixe value	997.5	-	[kg/m^3]
Fluid viscosity, ν		Fixed value	9.39e-7	-	[m^2/s]

6.3 Experimental Uncertainty

The uncertainty contribution due to experimental data was obtained from the documentation published in regards to the experiment discussed in Chapter 2. The uncertainty in the experimental data for the velocity at each point was reported at 1.6% [1] and the uncertainty contribution was found using as,

$$U_D = (0.016)D \tag{25}$$

where D was the nominal experimental value reported at each location.

6.4 Validation Uncertainty

The validation uncertainty contribution was found by comparing the experimental data set to the simulation prediction. This comparison is referred to as the validation metric in [8], which takes the difference in the experiment and CFD values at the same point. The validation uncertainty was found as,

$$U_{val} = |S - D| \tag{26}$$

where S is the Fluent model prediction and D is the experimental data.

6.5 Total Uncertainty in SRQ

To express the total uncertainty in the model prediction of the SRQ, all of the uncertainties discussed previously that contributed needed to be combined. Starting with the p-interval found by the input uncertainty analysis, each additional component of uncertainty was appended. Figure 21 provides an informative visualization of the total uncertainty in the predicted velocity for a single point in the domain. Presenting the uncertainty in the manner, allowed for easily identifying which uncertainties were contributing significantly to the overall uncertainty in model prediction of the SRQ.

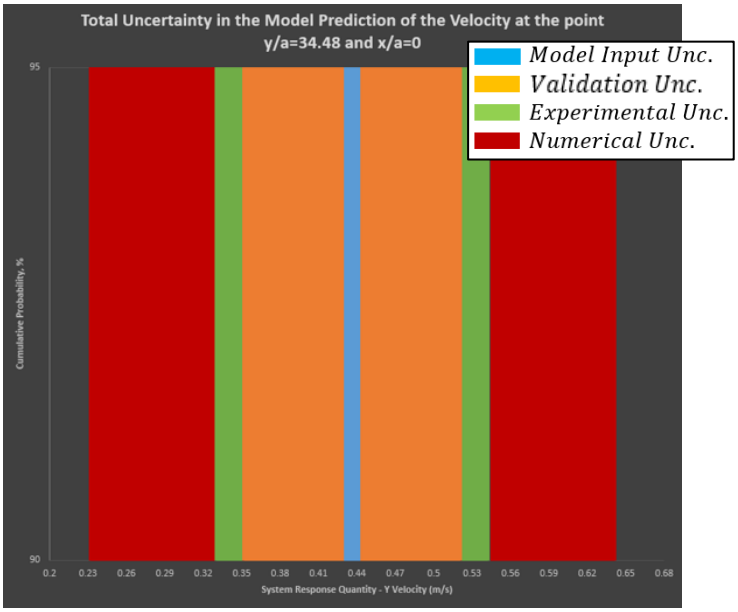


Figure 21: Total uncertainty in model prediction of SRQ due to each of the uncertainty contributions at one point: x/a=0, y/a=34.48.

7 HPC WORKFLOW

This chapter describes the details of the workflow that was used to combine the computational tools that were used in this study. The assessment and allocation of computational resources that were used is also discussed. Additional details which include the specific scripts and input files can be found in Appendix I and Appendix J. The guidelines used to create the workflow described in this chapter relied heavily on the DAKOTA 6.5 User's Guide [23] and the FieldView Reference Manual [24].

7.1 Computational Resources

The primary computational tools for this work were the ANSYS Fluent solver and Monsoon cluster, so an understanding of how to optimally utilize both was required. In theory, the time to complete a computational simulation could be reduced by allowing multiple CPUs to attack the same problem simultaneously. Instead of using a single CPU to perform all the calculations, the simulation would be divided amongst many CPUs to share the work. In some cases, however, the addition of more CPUs to the same simulation might increase computation time, instead of reducing it. Allocating multiple CPUs to a computational simulation, therefore, was expected to have a direct impact on the time required to run the simulation. To assess the computational capabilities used in this work, a performance study was conducted to determine the number of CPUs to allocate to the Fluent twin jet simulation on the Monsoon Cluster. This study revealed that the time to run the twin jet simulation would be reduced significantly by using multiple CPUs. Figure 22 shows a plot of the results obtained by conducting the performance study with the Fluent solver on Monsoon. As the number of CPUs was increased, the computational time was reduced significantly. To reduce computational time of each simulation, the decision was made to use the maximum possible number of CPUs to compute the twin jet simulation. The maximum possible CPUs was dictated by the number of Fluent licenses available for use in parallel on the Cluster, which was 120.

7.2 Automating UQ

As discussed in Chapter 6, the quantification of input uncertainty was a rigorous task and involved a significant number of simulation runs to adequately address. Assessment of the input uncertainty required

a number of simulations that depended upon the number of input parameters, and whether the inputs were classified as aleatory or epistemic. Recall, for every epistemic input uncertainty, all the aleatory uncertainties needed to be considered. The steps for setting up the input uncertainty study that were discussed in Chapter 6, defined 90 simulations that needed to be performed. Conducting the 90 simulations would not have been feasible without automation of the workflow that was first presented by Figure 9 in Chapter 4. The Dakota software offered the tools for automating the 90 simulations, but needed to be interfaced with the Fluent and FieldView software to make use of the automate features. For this reason, the Dakota tools needed to be setup to work with this specific twin jet CFD simulation. The detailed structure of how this setup was created for determining the input uncertainty in the twin jet model using Fluent and FieldView, is displayed in Figure 23.

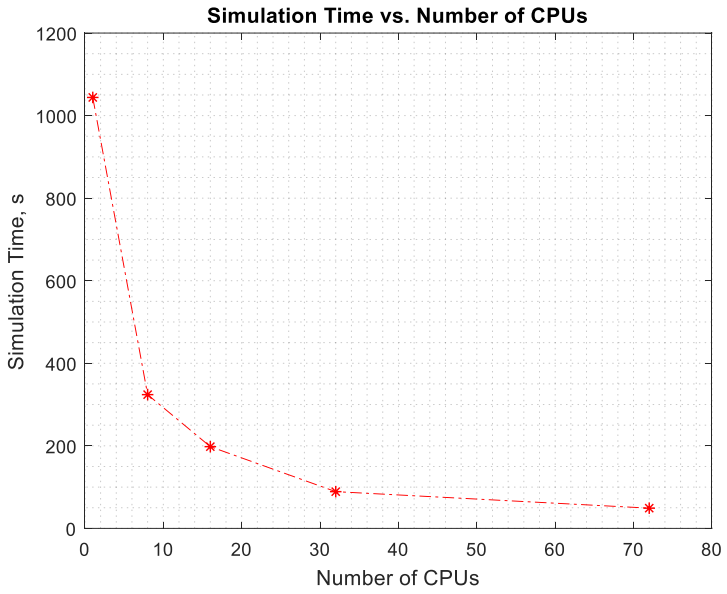


Figure 22: Computational performance of the Fluent solver on Monsoon in terms of the number of CPUs and time to complete the simulation.

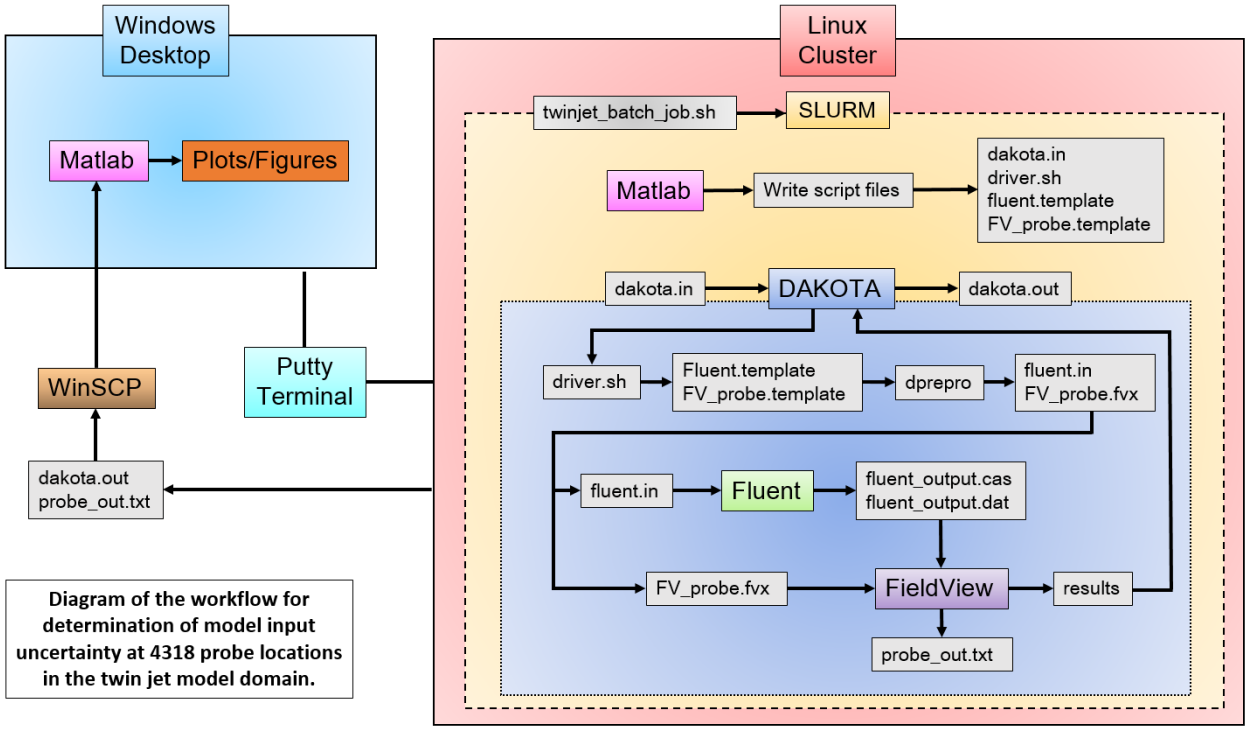


Figure 23: Workflow utilized to automate simulations on the Monsoon cluster and conduct the Dakota input uncertainty study.

Additional explanation of the process that was utilized to achieve the automated workflow described in Figure 23, is outlined in more detail in the following Steps.

Step 1 – Submit the automated study in the form of a batch job to the Linux/Monsoon Cluster

The automated workflow for establishing the input uncertainty in the twin jet simulation, began by submitting a job request to the Monsoon computer cluster. The Putty Terminal, which connects the Desktop to the Cluster, allowed for interaction with the Linux cluster using the desktop as a remote interface. Slurm was the job scheduling software on Monsoon, and decided when the job would start, and allocated computational resources such as memory and CPUs to the job. For the twin jet simulation job, Slurm was instructed to allocate 120 CPUs and 96 GB of memory. Once Slurm allocated resources and started the job, an automated sequence of events was enabled.

Step 2 – Generate all input files which provide instructions for how Dakota will run the study

First, Matlab was launched to write a series of input files that dictated how to run the input uncertainty analysis. The Matlab scripts that were used to generate these files can be found in Appendix I, and used to specify input instructions for Dakota, Fluent, and FieldView. The Dakota input file contains all the information regarding the number of simulation, uncertainty classifications, and software to use. In this case, Dakota needed to know that Fluent would be used to conduct the simulation, and FieldView would be used to collect and report the results of the simulation. To connect between Dakota, Fluent, and FieldView, there was a special file indicated as the driver. This file was in charge of carrying out the instructions that Dakota provided, which were based on the original instructions specified in the Dakota input file. The Fluent and FieldView input files contain information regarding the settings to use when running the twin jet simulation, and the locations to record results from the simulation data.

Step 3 – Modifying the input files for Fluent and FieldView before running a simulation

For this case, nine grids were available and needed to be selected to run with different combinations of model input parameters discussed in Chapter 6. To do this, the Dakota pre-processing (dprepro) tool was invoked (see Ref [23]), which made edits to the input files for Fluent and FieldView that were generated in Step 2. There is an example of these edited Fluent and FieldView input files provided in Appendix J. Once Dakota finished using dprepro to make edits to the input files for Fluent and FieldView, the workflow was ready to run a simulation.

Step 4 – Run the twin jet simulation and record the simulation results as a response

Dakota used the special driver file to instruct Fluent to launch and run a simulation using the Fluent input file that was previously generated by Matlab and edited by dprepro in Steps 2 & 3. Once Fluent was finished running the twin jet simulation, Dakota once again used the driver file to instruct FieldView to measure and record the results of the simulation. Fluent results were in the form of case and data files (.cas & .dat), which FieldView used to probe/measure the axial

velocity at 4318 locations in the measuring plane described in Chapter 4. After probing the points in the domain, FieldView would write the axial velocity data to a results file. The values in this results file, were used and referred to as the response quantity in Step 5.

Step 5 – Generate Dakota output file and run the next simulation with new input parameters

Dakota stored the response quantity provided in the results file from Step 4 in the form of a Dakota output file. This output file contained the axial velocity values associated with the specific input parameters that were used when running the twin jet simulation under the specified conditions. Once the responses were recorded in the Dakota output file, the cycle repeated by starting back at Step 3, and defining input parameters for the next simulation. Again, Dakota made edits to the Fluent input file, which provided new input parameters before the next would simulation run, and the new results were recorded.

Step 6 – View the results of the Dakota input uncertainty study from the twin jet model

Once all 90 simulation were completed, and the results recorded in the Dakota output file, Dakota performed a final statistical calculation. Using all the response data stored in the output file, Dakota calculated the 95% confidence interval for the axial jet velocity using the data from all 90 simulations. This final calculation was appended to the Dakota output file, which was then moved from the cluster to the desktop where Matlab was used for visualizing the input uncertainty.

8 RESULTS AND DISSCUSSION

8.1 Nominal Results

The twin jet CFD model results in this section were generated by a single simulation using the approach described in Chapter 4. The nominally reported experimental values that are listed in Table 3 were used as the boundary conditions, and provided a confirmation that the model approach was able to compute a solution under these conditions. This step was critical to understanding the behavior of the model at locations where the simulation results (or SRQ) needed to be considered. Recall the measuring plane that was first shown by Figure 14, and discussed as the location where the CFD model results would be collected and used in the uncertainty quantification. As shown in Figure 24, this measuring plane was used to collect the model results by slicing the computational domain across the two jets, and recording the axial velocity at 4318 points within this plane. The process of slicing the domain and recording the velocity at each point within the plane was achieved using the FieldView post-processing software. After recording the velocity at each point in this plane, FieldView was then able to use the point values to generate contours of the velocity field within the plane for visualization. Using the contour representations of the flowfield, observations of the characteristics which defined the twin jet problem were made.

Observations of the axial velocity flowfield contours showed a distinct zone with negative axial velocity, which indicated recirculation of the flow. This recirculation was confined to the zone located between the two jets, which eventually caused the jets to merge. These flow characteristics are clearly seen in Figure 25, where the two distinct jets are unmistakable, but only in the region near the nozzle. Moving away from the nozzle in the axial flow direction, each jet becomes attracted to the other, eventually merging together entirely to form a single jet. These observed characteristics were found to be intrinsically related to the structure of the twin jet model, and provided valuable insights that were applied in the next sections involving uncertainty quantification.

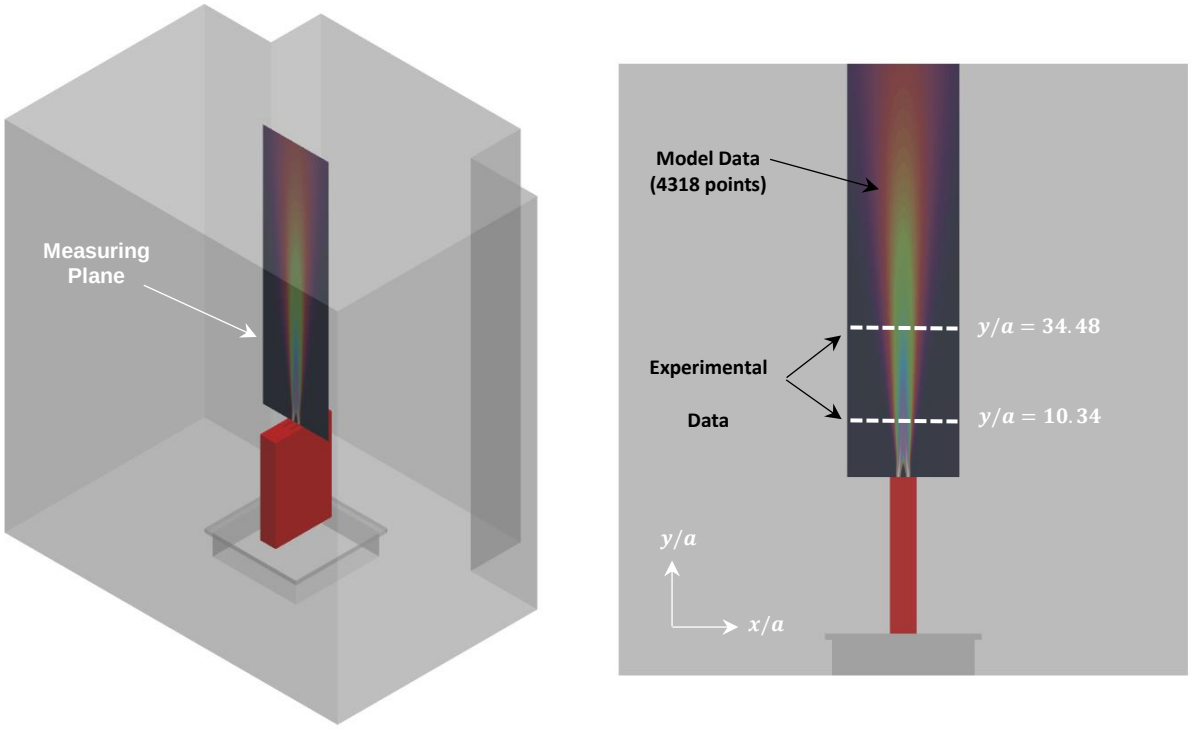


Figure 24: On the left is the measuring plane that was used to slice the domain across the two jets using FieldView. To the right is a contour of the axial velocity flowfield that was generated from the Fluent model, using the values at 4318 equally spaced model data points within the plane.

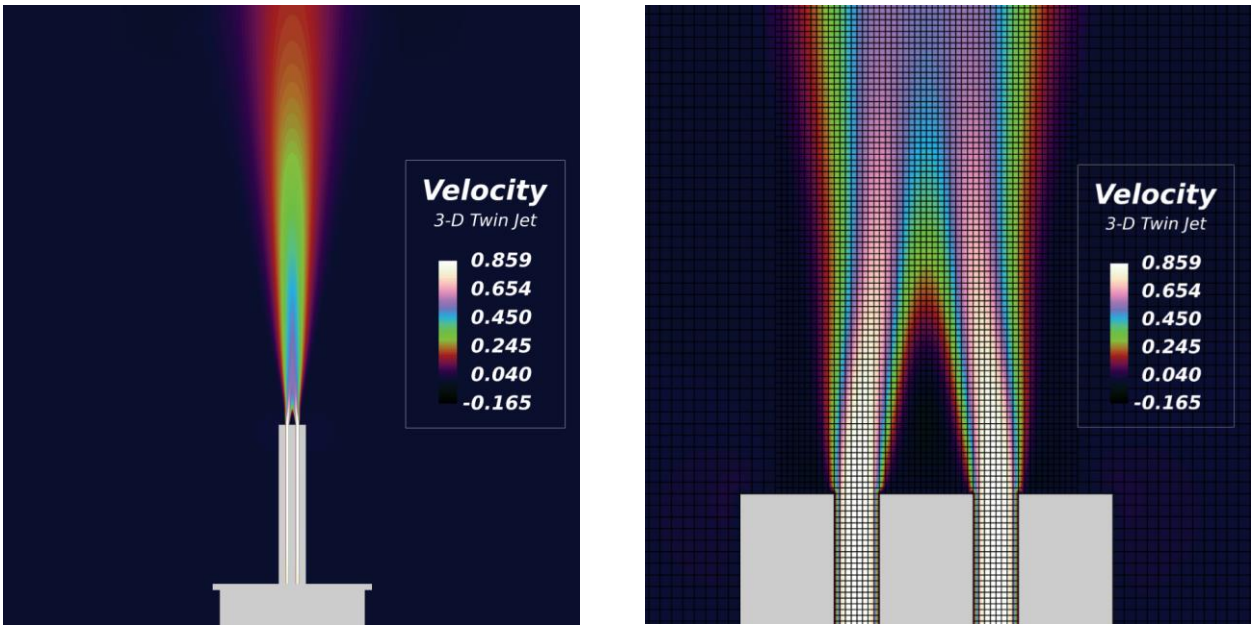


Figure 25: To the left is a contour of axial twin jet velocity generated by the CFD model using nominal conditions reported by the experiment as the boundary conditions. Shown to the right, a fine mesh used by the model to compute the solution in the region where the jets merge, is provided as an overlay on the contours for reference.

In addition to the velocity contours, Table 11 shows a list of the mean axial velocity values for both the CFD model prediction and the experiment. The values shown in this table were extracted from the model at points along the two white dashed-lines (y/a=10.34 and y/a=34.48) in Figure 24. Therefore, the values delineated in Table 11 present a direct comparison of the simulation prediction with the documented experimental data.

Table 11: List of the nominal axial velocity values predicted by the model and the corresponding experimental data at points along the lines y/a=10.34 and y/a=34.48.

axial velocity m/s	y/a:	10.34	10.34	10.34	10.34	10.34	10.34	10.34	10.34	10.34	10.34	10.34	10.34	10.34
	x/a:	−6.14	−5.13	−4.12	−3.10	−2.01	−1.04	−0	6.14	5.13	4.12	3.10	2.01	1.04
	CFD:	0.000	0.000	0.003	0.074	0.335	0.577	0.504	0.578	0.335	0.074	0.003	0.000	0.000
	Data:	0.001	0.011	0.012	0.096	0.393	0.619	0.550	0.605	0.366	0.085	0.009	0.004	0.005
axial velocity m/s	y/a:	34.48	34.48	34.48	34.48	34.48	34.48	34.48	34.48	34.48	34.48	34.48	34.48	34.48
	x/a:	−6.14	−5.13	−4.12	−3.10	−2.01	−1.04	−0	6.14	5.13	4.12	3.10	2.01	1.04
	CFD:	0.074	0.135	0.207	0.293	0.387	0.467	0.498	0.467	0.387	0.293	0.207	0.134	0.074
	Data:	0.131	0.200	0.274	0.366	0.424	0.474	0.501	0.472	0.409	0.323	0.252	0.178	0.117

8.2 Uncertainty Analysis Results

Significant preparation and planning was required to develop the twin jet CFD model to a point that uncertainty in the model results could be addressed. This planning and preparation was discussed in detail in the earlier Chapters, which culminated in the results presented in the following sub-sections. The results of quantifying the uncertainty in the model prediction of the axial velocity, using the techniques described throughout this work are presented and discussed in terms of the input, numerical, and validation uncertainty.

Input Uncertainty

The input uncertainty was due to uncertainty in the experimental conditions, leading to uncertainty in the inputs (or boundary conditions) used in the Fluent model of the experiment. Discussed in Chapter 6, the physical nature of the pumps and flow meter used in the experiment, contained inherent random variation that could not be reduced. Additionally, the geometric configuration of the two jets that were used in the experiment was also a source of uncertainty. To quantify the uncertainty that was introduced by both the

random variations and jet configuration, a total of 90 simulations were performed on a computer cluster. The details of conducting the 90 simulations were discussed in Chapter 7, and the results of these simulations are presented here.

Recall that nine jet spacing ratios were used to cover the range of uncertainty in the geometry configuration, and can be referred to in Figure 19. To describe the random variations, a normally distributed PDF was used, referred to in Figure 18. The Dakota software conducted the input uncertainty study by using a Latin-Hyper-Cube algorithm to choose 10 samples from the PDF. For each sample, a simulation was performed, and the process was repeated for each of the nine jet spacing ratios. This resulted in the 90 simulations discussed previously, and for each simulation the axial velocity was recorded and analyzed. This provided statistical quantification of the uncertainty in the axial flowfield velocity due to uncertainty in the inputs to the model. The results of the Dakota input uncertainty study provided a range or interval over which the CFD twin jet model predicted the axial velocity, instead of a single value. The predicted interval was based on the mean and standard deviation of the axial velocity reported by the Dakota study after running 10 simulations on each of the nine jet spacing ratios. The contour plots that are shown in Figure 26 represented the mean axial velocity computed by running the twin jet model using 10 different combinations of the mass-flow-rate, turbulence intensity, and turbulent length scale. After each simulation, FieldView was used to measure and record the twin jet model prediction at each of the 4318 model data points in the slicing plane discussed previously. Matlab was used to structure the statistical mean value at each of these points and generate the contour plots that are shown.

In addition to the mean axial velocity shown in Figure 26, the standard deviation was required to quantify the model prediction interval. Following the same procedure used to find the mean velocity, the Dakota study also computed the standard deviation associated with the mean. The magnitude of this deviation, with a specified 95% confidence, is shown for each of the nine jet spacing ratios in Figure 27. To generate the 95% confidence interval over which the axial velocity was predicted to fall, the magnitude of the

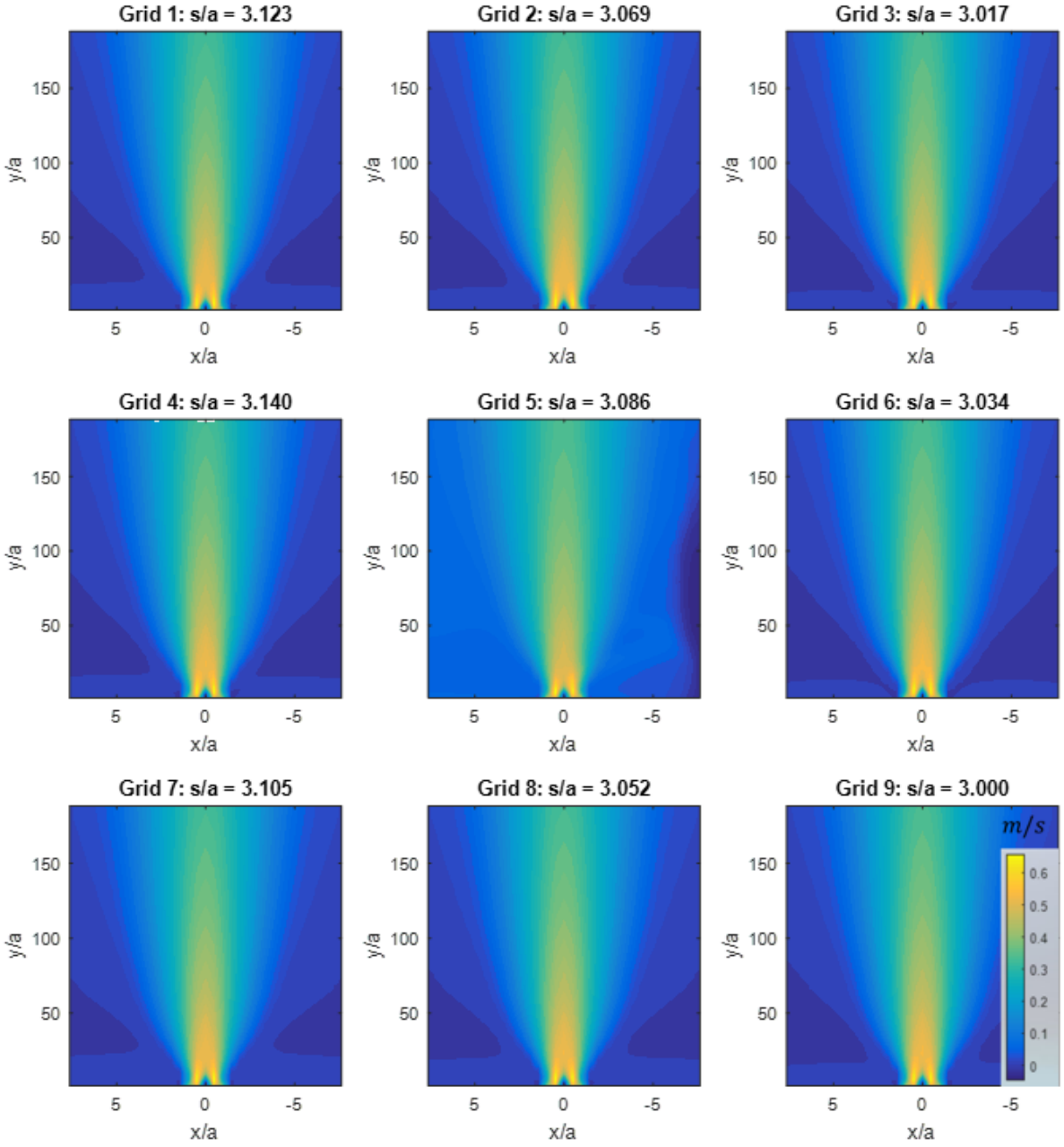


Figure 26: Contours of the twin jet model prediction for the mean axial flowfield velocity are shown for each of the nine jet spacing ratios.

deviation was both added and subtracted to/from the mean value at each of the 4318 model data points. This resulted in contours of both the lower and upper bounds, used to quantify the prediction interval of axial velocity by the model. These additional plots can be found Appendix F. It should also be noted that the response displayed by the plot indicated as Grid 5, exhibited anomalous behavior. This was found to

be the consequence of a single simulation that did not converge to a steady solution during the Dakota input study which included a total of 90 simulations. The values from this simulation were deemed and outlier, and thus removed from the results that are presented from here on. The following results are representative of the data obtained by the remaining 89 supporting simulations.

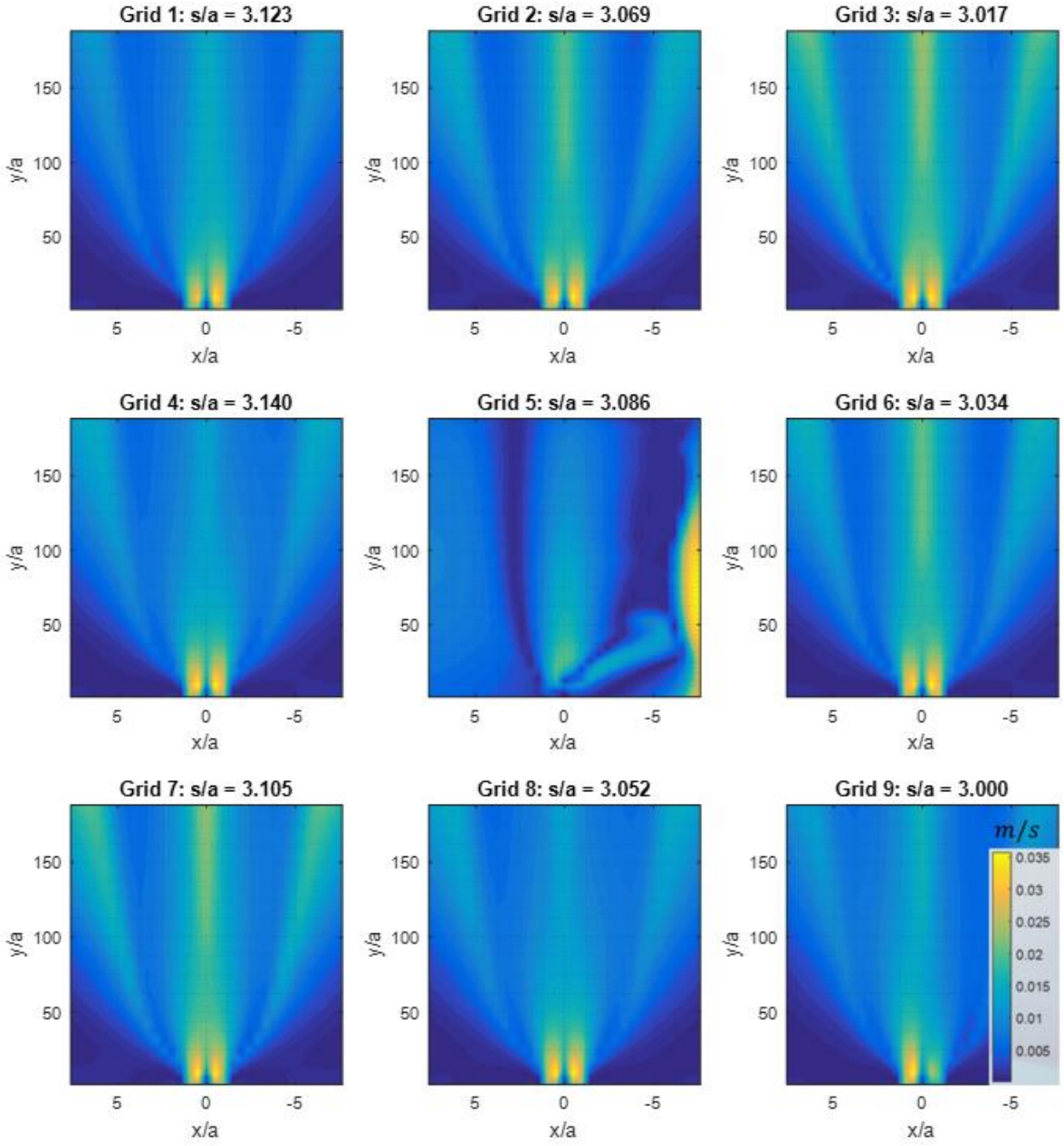


Figure 27: Contours showing the magnitude of uncertainty in the mean velocity from the twin jet model predictions for each of the nine spacing ratios tested.

Recall the two white dashed-lines in Figure 24 that were mentioned as the locations where experimental axial velocity data was available. In preparation for comparing the simulation with this experimental data at points across these two line, the twin jet CFD data was collected at points across the same lines. At each of the points along the two lines, the upper and lower simulation values were recorded. Matlab was used to generate additional plots that contained the upper and lower values at each point across the two lines, where the simulation will be compared with the experiment. The black bars shown in Figure 28, represent the upper and lower values at each point across the line, y/a=34.48. At this location in the flowfield, it was clear the two jets had completely merged together and formed a single jet. In Figure 29, the same process was applied, but instead for the points across the line, y/a=10.34. At this location, the jets had not fully merged, and two distinct peaks in the axial velocity were visible.

Once the interval of predicted axial twin jet velocities was defined across the two lines on each of the nine jet spacing ratios, a total of nine intervals existed for each point across these lines. The nine intervals at each point needed to be combined into a single interval that would represent the model prediction at each point. The combined interval could then be used to represent the CFD model prediction of the axial velocity at each point across the two line, and used for comparison with experimental data. This was achieved by considering the upper and lower bounds at each of the same points on each plot shown in Figure 28 and Figure 29. For example, only consider the single interval predicted by the Fluent model located at x/a=0 and y/a=10.34 for each of the nine plots that are shown in Figure 29. The lowest interval bound from all nine plots at this location is recorded, and then repeat for the upper bound. The combination of upper and lower bounds that were recorded in this example represents the total uncertainty interval across all nine jet spacing ratios. The results of expanding this approach to each of the points where the velocity was predicted is shown by Figure 30. To provide addition visualization of this technique, the black interval bars used to show the interval of axial velocity predicted by the simulation, were replaced by the black area-fill that is shown in Figure 31.

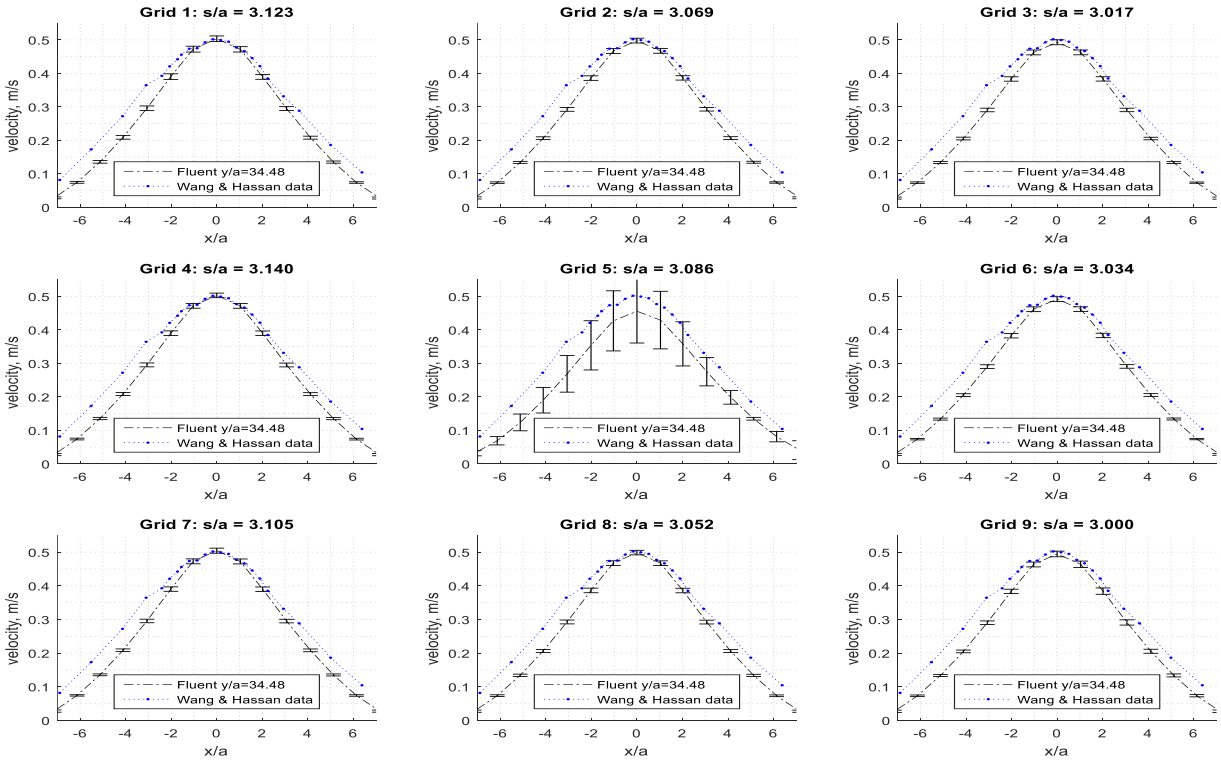


Figure 28: Interval prediction of the axial velocity from 10 simulations for the model data points across the line y/a=34.48.

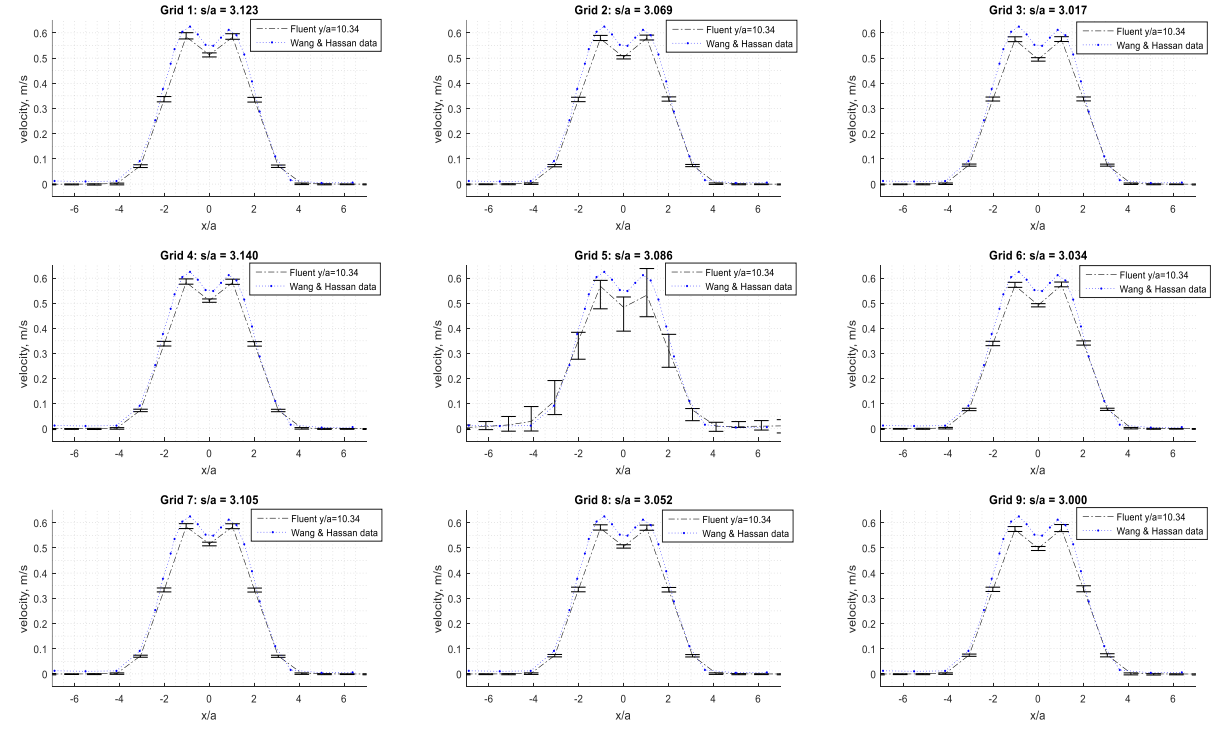


Figure 29: Interval prediction of the axial velocity from 10 simulations for the model data points across the line y/a=10.34.

This black area-fill represents the combined uncertainty prediction by the simulation due to uncertainty in the model inputs at the points across the line where the CFD model will be compared to the experimental data. To define a single 95% confidence interval over which the response quantity due to both aleatory and epistemic uncertainty in the model inputs were predicted to fall, the combination of responses that produced the largest interval was found. This ensured that the combined interval would include the largest range of possible responses by the model.

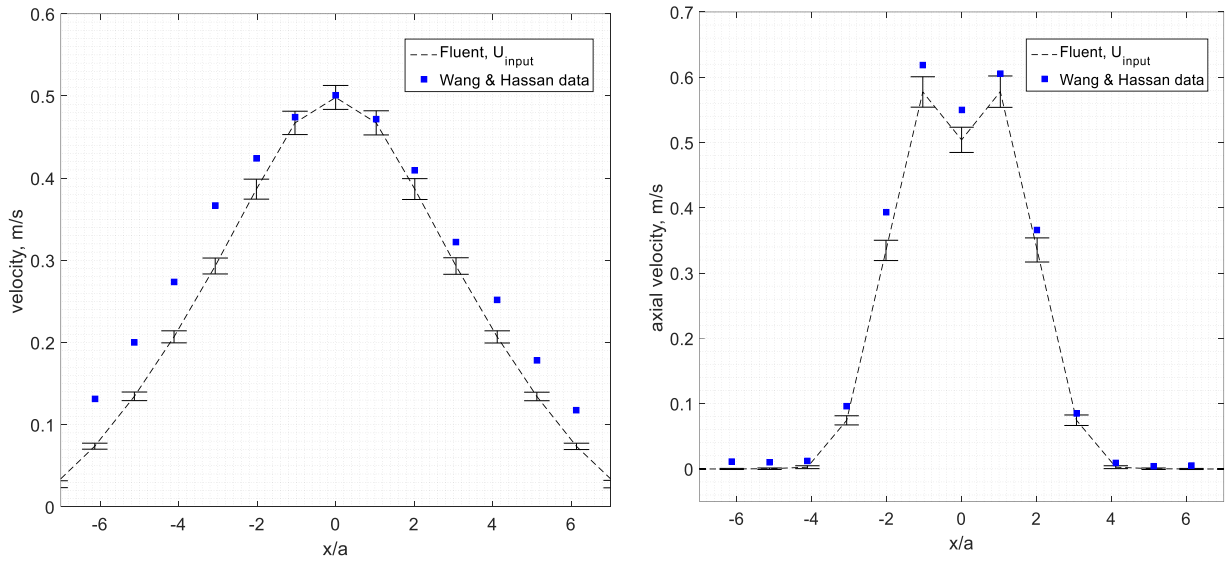


Figure 30: Predicted intervals of possible axial velocities at the validation data points, based upon the 90 simulations conducted to quantify the input uncertainties.

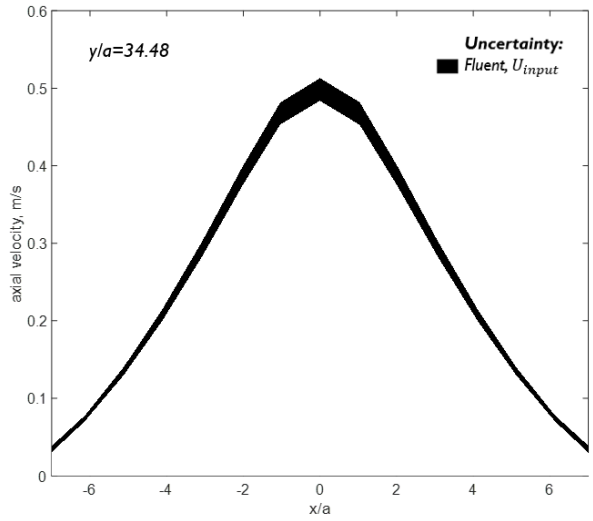


Figure 31: Visual representation of the Fluent model input uncertainty referred to as the 'p-interval' in Chapter 6.

Numerical Uncertainty

The numerical uncertainty was computed to assess the contribution of discretization error in the model prediction of axial velocity. The numerical uncertainty analysis was conducted at the same model data points discussed previously, by slicing the domain across the twin jets. The details for quantifying the numerical uncertainty in the twin jet CFD model were presented in Chapter 5. Recall that the theoretical order of accuracy for this model was two. In order to quantify the numerical uncertainty, the observed order of accuracy using the solution on three grid refinement levels was required. Using the three grids that were presented in Figure 17, a simulation was conducted to obtain a solution for each grid. At each of the 4318 model data points in the measuring plane of interest, the observed order of accuracy was computed using the three grid solution values. FieldView was used to extract and record the simulation values at the 4318 points. Matlab was used to evaluate the observed order using the recorded point values, and generate the contours shown in Figure 32. This allowed for visualizing the observed order in terms of the entire measuring plane of model data points, instead of just a single point. The contours were a particularly helpful indicator of the numerical behavior as the grid was refined. It was observed that in the region near the nozzle, located where both x/a and y/a were zero, the observed order was much less than the theoretical order of two. This indicated that as the solution was refined, the point value on each grid was not approaching an asymptotic solution value. The regions with an observed order of close to one were converging asymptotically, but not at the same order of accuracy as the theoretical order of two. These points were found to have very small differences across all three grid refinement levels. This was supported by the small numerical errors that were computed in these regions. Finally, in the regions where the observed order was equal or greater than the theoretical order, the solution was converging asymptotically at the theoretical order of accuracy. Observations of the numerical uncertainty contours indicated that relatively large numerical errors were isolated to the region near the nozzle where the solution was not converging asymptotically.

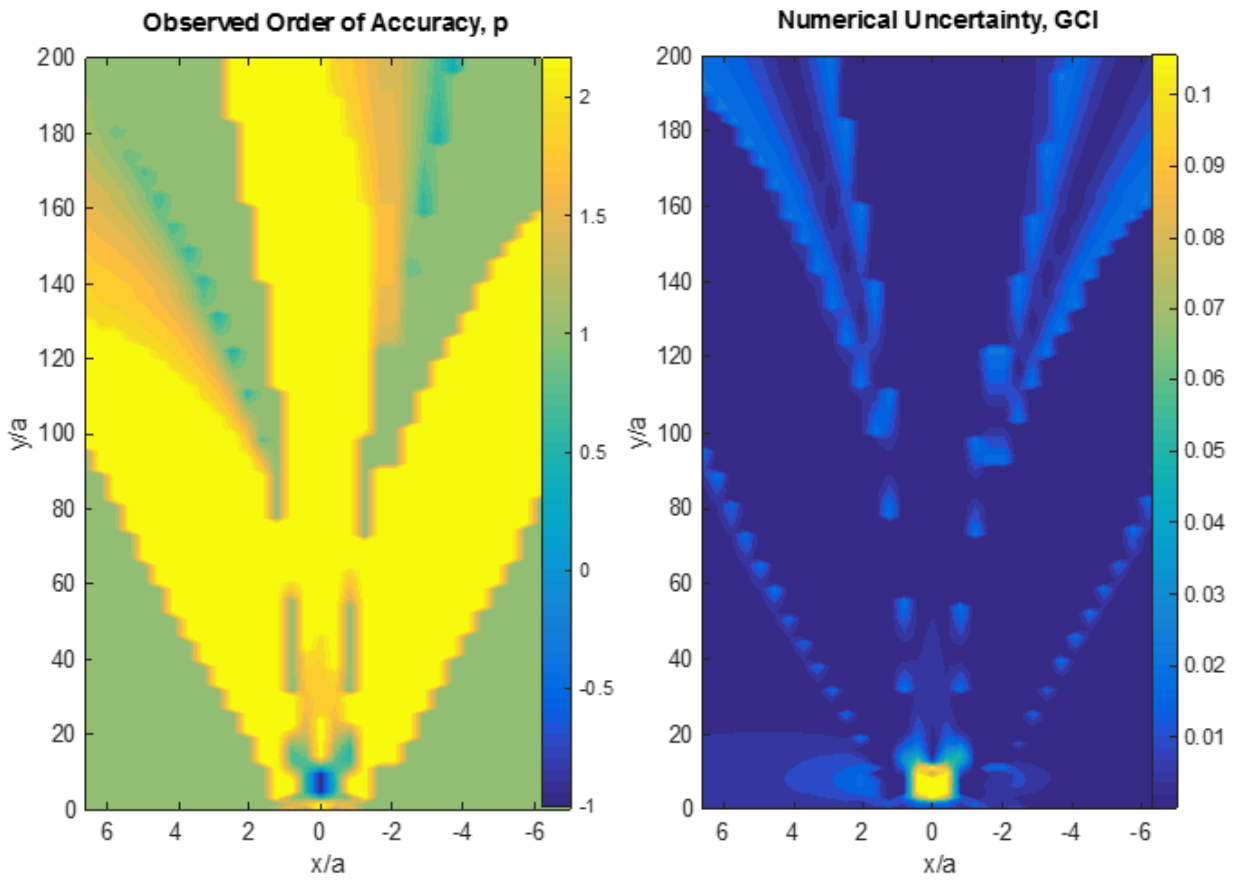


Figure 32: The left contour shows the observed order of accuracy that was computed using the solution values of three grid refinement levels at each of the 4318 model data point locations in the slicing plane across the two jets. The numerical error due to discretization was also computed and shown by the contour plot on the right which used the GCI method where applicable.

Since the theoretical order was two, the GCI method was applied to a significant portion of the points where the observed order was found to be close to the theoretical order. However, in the region near the two jet inlets, the solution was not converging at the same order as the theoretical. In this region, the numerical uncertainty was found to be significantly greater than anywhere else in the domain, and the secondary approach (discussed in Chapter 5) for computing the error was used instead of the GCI. Due to the spacing of the coarse mesh and high gradients that existed in the near jet region, it was not surprising that the discretization error dominated in this way.

Similar to the technique used in the input uncertainty study, the numerical error was collected and recorded for each of the model data points across the two dashed-lines that were show in Figure 24. The numerical error was then used to expand the uncertainty interval of the predicted axial velocity, due to

contributions of the numerical uncertainty. This interval was first presented in Figure 31, as the black area-fill region that described uncertainty in the model prediction due to the model inputs. In Figure 33, this interval of uncertainty in the model prediction of the axial velocity is expanded by including the contribution of numerical uncertainty. Indicated by the red area-fill is the numerical uncertainty that was found by considering the numerical error at the points across the line y/a=34.48. The numerical error from the points across this line were then appended to both sides of the input uncertainty interval. The combination of the red and black area-fill, represented the uncertainty in the model prediction of axial velocity, due to both the input and numerical uncertainty contributions.

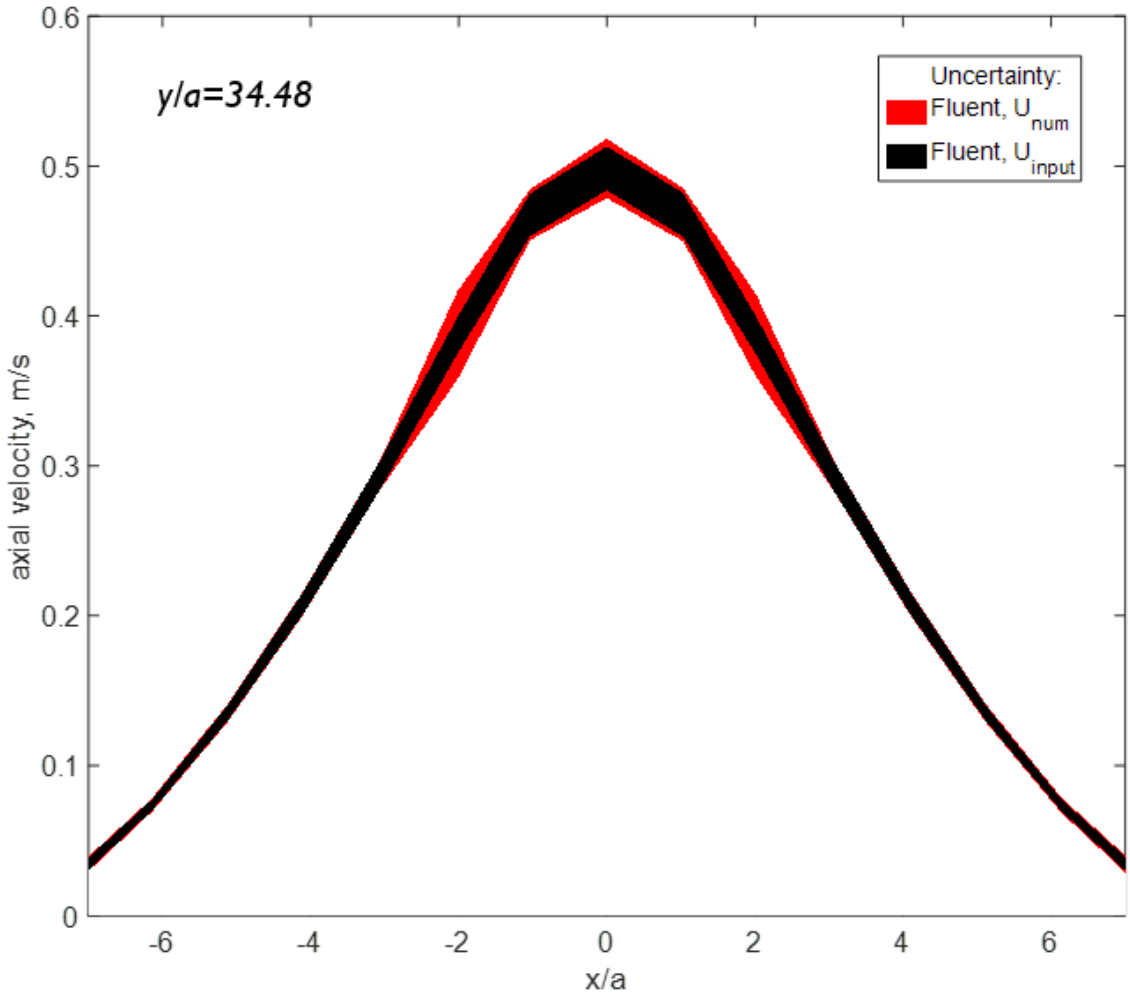


Figure 33: Visual representation of the expanding uncertainty interval by appending the numerical uncertainty to both sides of the input uncertainty shown previously in Figure 31.

Validation Uncertainty

The last contribution of uncertainty to the model prediction of axial velocity was the validation uncertainty. This contribution was found by comparing the simulation prediction with the experimental data. The simulation prediction used in the validation was not a point value, but the interval that was obtained through the Dakota input study. Quantifying the validation uncertainty was found by taking the difference in the simulation prediction interval and experimental data point. As shown in Figure 34, the differences between the edges of the input uncertainty interval and experimental data were found. These differences were designated as the validation uncertainty, shown by the purple lines. At the points where the validation uncertainty was zero, indicated points where the experimental data fell within the range captured by the input uncertainty. This corresponded with no validation uncertainty, since the interval of input uncertainty was close to the experimental values.

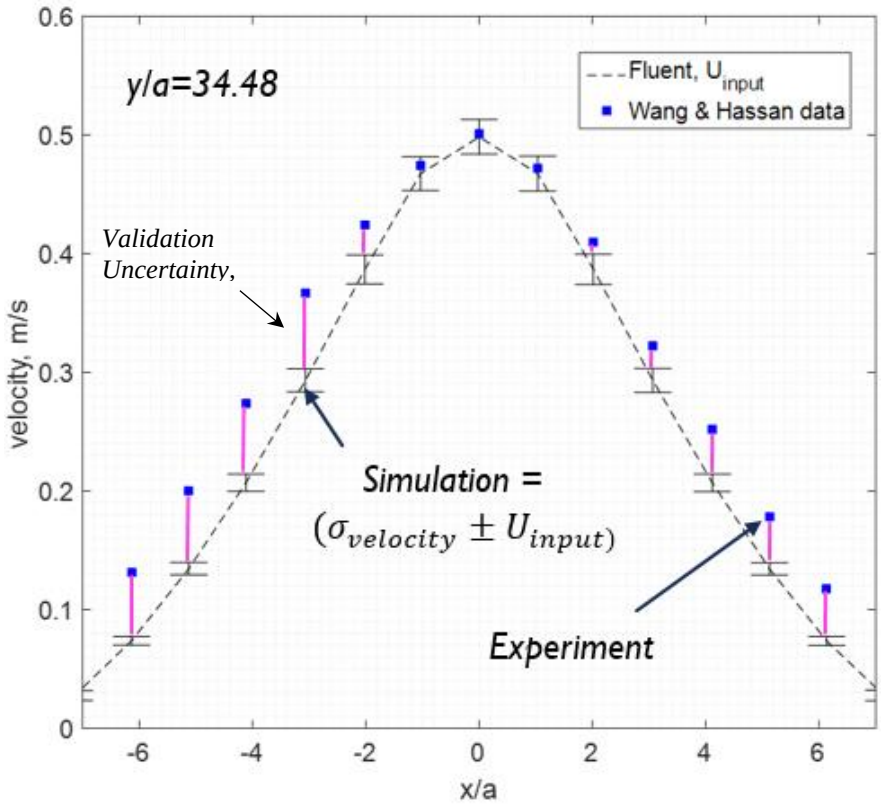


Figure 34: Validation uncertainty in the Fluent model prediction of the axial velocity indicated by the difference between simulation and experimental values.

The validation uncertainty that was computed by taking the difference in the simulation prediction and experimental data, was also appended to the expanding interval of uncertainty in the CFD model prediction of axial velocity. Shown in Figure 35, the combined uncertainty due to the contributions of input, numerical, and validation uncertainty is presented.

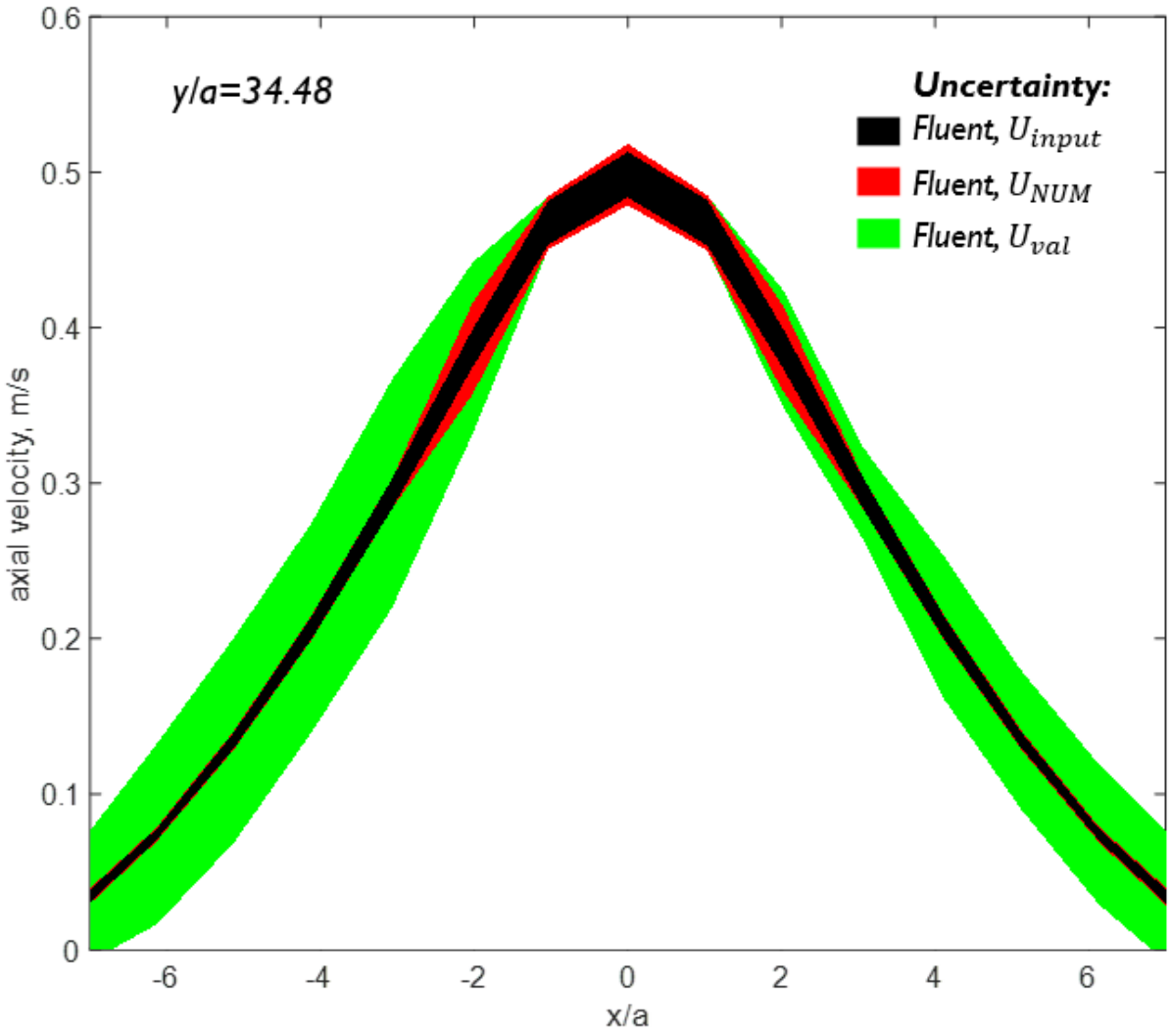


Figure 35: Contribution of validation uncertainty shown as the green area-fill region to the expanding interval of uncertainty in the CFD prediction of axial velocity.

Total Uncertainty in SRQ Prediction

The total uncertainty prediction in the CFD model of the twin jet problem included the combination of all sources of uncertainty that were discussed up to this point: input, numerical, and validation. To quantify the total uncertainty in the prediction of axial velocity, contributions from each the uncertainties were combined. As discussed in Chapter 6 and shown in the previous results, the input uncertainty interval provided the starting point, upon which the numerical and validation uncertainty was appended.

The last contribution that was discussed in Chapter 6, but not addressed until now, was uncertainty in the experimental data. This uncertainty was reported to be within 1.6% of the experimental data values. This uncertainty was appended to the final uncertainty interval in the same fashion used to append the previous contributions. Figure 36 shows the total uncertainty in the model prediction of axial velocity across the line in the domain where the jets had completely merged together, at y/a=34.48, due to input, numerical, validation, and experimental uncertainties. Figure 37 also shows the total uncertainty in the model prediction, but instead for the line where the jets were in the process of merging at y/a=10.34. In each figure, the experimental data used to quantify the validation uncertainty was included as a reference.

In Figure 36, the validation uncertainty was found to dominate in the regions further away from the twin jet centerline, which is located at x/a=0. In the region near the centerline, however, the validation uncertainty vanished entirely, and was instead, primarily attributed to the input uncertainty. The numerical uncertainty in this region of the flow was observed to be quite small, contributing very little to the total uncertainty interval. In Figure 37, the flow at y/a=10.34 was in the process of merging, and the numerical uncertainty contribution was observed to dominate. This location was closer to the nozzle, previously identified in Figure 32, as a region with larger numerical uncertainty. This was attributed to the coarse grid and high gradients located in this region of the domain.

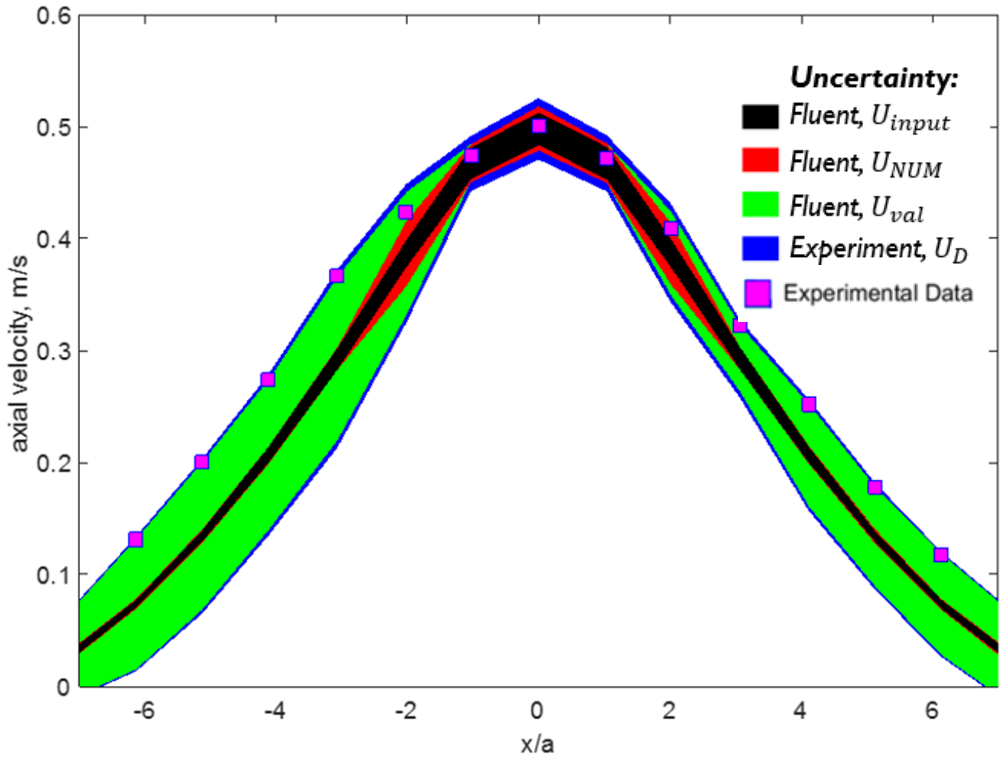


Figure 36: Total uncertainty in Model Prediction of the axial velocity at the location y/a=34.48 due to the contributions from the input, numerical, and validation uncertainty.

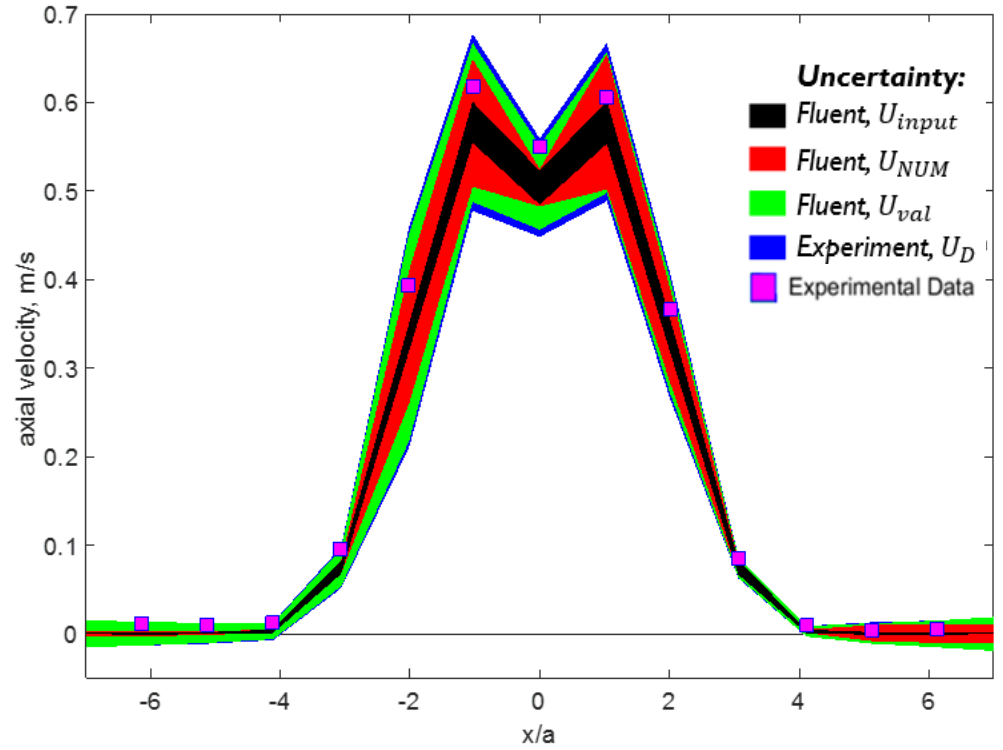


Figure 37: Total uncertainty in Model Prediction of the axial velocity at the location y/a=10.34 due to the contributions from the input, numerical, and validation uncertainty.

8.3 Discussion

Uncertainty quantification of the axial velocity predicted by the Fluent CFD model of the ASME twinjet benchmark problem was conducted using published techniques. The numerical and validation uncertainty were the most significant contributors to the overall uncertainty in the model prediction. Input uncertainty did contribute, but was notably less in magnitude, and far more consistent. In particular, the numerical uncertainty exhibited highly variable behavior, jumping from small to large intervals within just a few grid locations near the nozzle exit.

The mesh was designed to capture the numerical uncertainty due to discretization, by using uniformly spaced hexahedral cells. Due to computational limitations, the highest resolution grid that was achieved consisted of only four cells spanning each nozzle diameter. While four cells were observed to be sufficient to capture the basic structure of the flow, the coarse mesh, which only consisted of a single cell spanning the nozzle diameter, was not. The large gradients observed in the jet nozzle inlet region were not captured well by the coarse mesh. This result was observed in the large numerical errors that were found in this region and attributed to high gradients between cells. Similar to a boundary layer, the flow transitioned from static to jet velocity over a very small distance. A significant number of cells would be required to adequately resolve this transition, and explained the large variability in numerical errors observed in neighboring cells. While not surprising, the lack of cells in the transition regions produced dominating uncertainty effects by the numerical approximations of the CFD solver. If the techniques presented here were to be continued, an additional refinement grid with 8 cells spanning the jet diameter is recommended. Such a mesh was expected to result in over 100 million cells, not feasible with the tools available in this study. Based on the behavior observed, the additional refinement grid would reduce the contribution of numerical uncertainty significantly in the region near the nozzle.

To address the validation uncertainty observed in the model, a sensitivity analysis on the turbulence model selection input parameters is compulsory. The assumptions made in selecting the inputs to the $k-\varepsilon$ turbulence model were observed to have a significant influence on the velocity field. Additionally, the

Reynolds stress turbulence models available in the Fluent solver would provide additional insight and worth further investigation. Some experience with these models indicated promising results, though challenges exhibited in solution convergence were found.

9 CONCLUSIONS

The Fluent CFD software was used to model the ASME turbulent twin jet flow regime. The question regarding uncertainty in the model prediction of axial flowfield velocity was addressed through the verification and uncertainty quantification of the CFD model. First, the problem was modeled in 2- and 3-dimensions, and the results exhibited a distinct merging of the two separate jets into a single jet. The model was then subjected to a series of exercises, designed to quantify the uncertainty of the predicted axial velocity. These exercises included quantifying the numerical, input, and validation uncertainty. The numerical uncertainty was found using the Grid Convergence Index method based on the solution from three grid refinements. The input uncertainty was addressed by classifying the inputs as either aleatory or epistemic, then sampling the parameter space and mapping the model response to a 95% confidence interval. Due to the high computational cost of the input study, the workflow was automated and performed on a Linux computer cluster. Finally, the validation uncertainty was determined by comparison with published experimental data of the velocity field. The combination of the aforementioned uncertainty analysis, was used to make a final prediction of the total uncertainty in the Fluent model with 95% confidence. The techniques and procedures described and implemented provided answers to the question regarding the capabilities of the Fluent solver as a tool for modeling turbulent twin jets. The sections below state each objective of the thesis followed by the related conclusions.

Modeling the ASME Turbulent Twin Jet Model using CFD

The model approach that was discussed in Chapter 4, was used to create a Fluent CFD model of the ASME turbulent twin jet benchmark challenge problem. The CFD model was based on a published experiment configuration, that was used to collect data about the twin jet velocity flowfield. The CFD model approach was iterative, and began by defining the model geometry to be as similar as possible to the geometry that was used in the experiment. The mesh used hexahedral cells, and was created from the geometry, which defined the finite locations where the CFD solver would compute a solution. The boundary conditions for the model were also selected to agree with the experimental conditions. These

conditions included no-slip walls, mass-flow-rate specification of the jet inlets, and a pressure outlet to the model domain. The SIMPLE algorithm was chosen to resolve the coupled nature of velocity pressure. The approach chosen to address the turbulent mixing of the twin jets, was the realizable $k-\varepsilon$ turbulence model approach. The model approach also assumed steady state, and since the working fluid of the experiment was water, an incompressible assumption was made.

Quantifying the Input Uncertainty in the CFD Model

To quantify the input uncertainty in the CFD model prediction of the axial twin jet velocity, the Dakota software was used to run a total of 90 simulations using 120 CPUs on a Linux computer cluster. The inputs that were classified as aleatory due to randomness in the experiment were the mass-flow-rate of the jets, turbulence intensity, and turbulent length scale. The configuration of the nozzle in terms of the jet width and jet spacing was classified as epistemic. Each of the 90 simulations that were performed used a variation of input parameters, which provided the interval of uncertainty due to inputs. The interval that was produced by the Dakota study, provided the basis upon which additional uncertainty in the model was appended. The input uncertainty at the validation points was found to vary from a minimum of ±0.01 m/s to a maximum of ±0.05 m/s, relative to the mean velocities that ranged from 0 m/s to 0.6 m/s.

Quantifying the Numerical Uncertainty in the CFD Model

The numerical uncertainty in the CFD model prediction was quantified in terms of the discretization uncertainty. This was achieved by refining the solution using three grids, and using the value from each grid solution to compute the observed order of accuracy. Since the theoretical order of the CFD model was two, if the observed order was close to theoretical order, the GCI method was applied. In the region of the domain near the nozzle, the observed order was much less than the theoretical order, so an alternative method discussed in Chapter 5 was used. The coarse design of the mesh in the region near the nozzle that was used in the three grid refinement study, was found to contribute to large numerical uncertainty in this part of the domain. The numerical uncertainty at the validations points was found to vary from a minimum of ±0.01 m/s to a maximum of ±0.06 m/s.

Quantifying the Validation Uncertainty in the CFD Model

The final uncertainty contribution to the CFD model prediction of axial velocity was due to differences in the experimental data and the simulation prediction. This validation uncertainty, was found by taking the absolute difference in the model prediction and experimental data values at the same point in the domain. The validation uncertainty was found to dominate in the regions further away from the twin jet centerline. In the regions near the jet centerline, however, the validation uncertainty was observed to be quite small, even non-existent at some points. The validation uncertainty at the validations points was found to vary from a minimum of zero to a maximum of ±0.05 m/s.

Quantifying the Total Uncertainty in the CFD Model

The cumulative uncertainty due to model inputs, numerics, and validation were combined to quantify the total uncertainty in the CFD prediction of axial velocity. Since each of the aforementioned uncertainties contributed in some way to the CFD model results, the total uncertainty could only be found through identifying and quantifying each source of uncertainty independently. Once this was achieved, the contribution of each source of uncertainty in the model result was combined to make a final prediction of total uncertainty in the CFD model results. The total uncertainty interval that was presented in this study, was the end results of quantifying each source of uncertainty in the CFD model, and predicted a range over which the axial twin jet velocity would truly fall, with a 95% confidence level. The total uncertainty at the validations points was found to vary from a minimum of ±0.02 to a maximum of ±0.1 m/s.

Final Comments

The primary sources of uncertainty in the Fluent model prediction were the numerical and validation uncertainty. This was attributed to the coarseness of the finest mesh and ability of the turbulence model to accurately model the flow. The grid amplified numerical errors observed in the refinement study, and dominated the regions with high local gradients. To reduce the contribution of numerical uncertainty, investigation of an alternative approach to the grid design is recommended. The validation uncertainty suggested the assumptions made by the steady, realizable k-ε turbulence model approach used to solve

the twin jet problem, did not capture the full nature of the flow regime. To address the validation uncertainty, a parameter study of the turbulent model inputs is also recommended. In conclusion, the elegant structure of this rigorous uncertainty quantification, provided deep insight to the cause and source of uncertainty in the model prediction. These insights provided a direction of future work, by identifying strengths and weakness of this approach.

REFERENCES

[1] H. Wang, S. Lee, Y. A. Hassan, and A. E. Ruggles, "Laser-Doppler measurements of the turbulent mixing of two rectangular water jets impinging on a stationary pool," *Int. J. Heat Mass Transf.*, vol. 92, pp. 206–227, 2016.

[2] H. Wang and Y. A. Hassan, "Benchmark Data for Validation of Computational Simulations of Nuclear System Thermal Fluids Behavior," Texas A&M University, 2015.

[3] R. Crane, "Announcement : V & V Benchmark Problem 1 — Twin Jet Computational Fluid Dynamics (CFD) Numeric Model Validation." ASME V&V 30 Committee, 2016.

[4] "ASME V&V 20-2009 Standard for Verification and Validation in Computational Fluid Dynamics and Heat Transfer," 2009.

[5] "Perspective: A Method for Uniform Reporting of Grid Refinement Studies," *J. Fluids Eng.*, vol. 116, pp. 405–413, 1994.

[6] P. J. Roache, *Verification and Validation in Computational Science and Engineering*. Albuquerque: Hermosa Publishers, 1998.

[7] L. Eca and M. Hoekstra, "Evaluation of numerical error estimation based on grid refinement studies with the method of the manufactured solutions," *Comput. Fluids*, vol. 38, no. 8, pp. 1580–1591, 2009.

[8] C. J. Roy and W. L. Oberkampf, "A comprehensive framework for verification, validation, and uncertainty quantification in scientific computing," *Comput. Methods Appl. Mech. Eng.*, vol. 200, no. 25–28, pp. 2131–2144, 2011.

[9] W. L. Oberkampf and C. J. Roy, *Verification and Validation in Scientific Computing*. New York: Cambridge University Press, 2010.

[10] H. B. Lee *et al.*, "Development and Use of Engineering Standards for Computational Fluid Dynamics for Complex Aerospace Systems," in *46th AIAA Fluid Dynamics Conference*, 2016, no. June, pp. 1–28.

[11] M. M. Granger, M. Henrion, and M. Small, *Uncertainty: A Guide to Dealing with Uncertainty in Quantitative Risk and Policy Analysis*. Cambridge University Press, 1992.

[12] G. F. Marsters, "Interaction of Two Plane, Parallel Jets," *AIAA J.*, vol. 15, no. 12, pp. 1756–1762, 1977.

[13] M. Crosskey and A. Ruggles, "UTK Twin Jet Water Facility Computational Fluid Dynamics Validation Data Set," in *Proceedings of International Congress on Advances in Nuclear Power Plants (ICAPP '14)*, 2014.

[14] X. Lu and L. Wang, "2D Turbulent Jet Study Based on Fluent Ķ," *Symp. A Q. J. Mod. Foreign Lit.*, pp. 608–613, 2008.

[15] L. B. Carasik, A. E. Ruggles, and Y. A. Hassan, "Preliminary investigation of turbulent flow

behavior of 3-D twin jets using CFD analysis," in *Transactions of the American Nuclear Society*, 2014, vol. 110, pp. 689–692.

[16] N. E. Bunderson and B. L. Smith, "Passive mixing control of plane parallel jets," *Exp. Fluids*, vol. 39, no. 1, pp. 66–74, 2005.

[17] D. Ostheimer and Z. Yang, "A CFD Study of Twin Impinging Jets in a Cross-Flow," *Open Numer. Methods J.*, vol. 4, no. 12, pp. 24–34, 2012.

[18] N. Hnaien, S. M. Khairallah, H. Ben Aissia, and J. Jay, "Numerical Study of Interaction of Two Plane Parallel Jets," *Int. J. Eng. (IJE), Trans. A Basics Int. J. Eng. J Hnaien al. / IJE Trans. A Basics*, vol. 29, no. 10, pp. 1421–1430, 2016.

[19] J. D. Anderson, *Computational Fluid Dynamics - The Basics With Applications*. New York: McGraw-Hill Inc., 1995.

[20] ANSYS, "Fluent Documentation - Lecture 7: Turbulence Modeling," 2014.

[21] K. Y. Chien, "Predicitons of Channel and Boundary-Layer Flows with a Low-Reynolds-Number Turbulence Model," *AIAA J.*, vol. 20, no. 1, pp. 33–38, 1982.

[22] V. Vanoni, "Density and Viscosity of Water 0 ◦ C – 40 ◦ C," in *Sedimentation Engineering: Classic edition*, ASCE Library, 2006, pp. 407–409.

[23] B. M. Adams, M. S. Ebeida, M. S. Eldred, J. D. Jakeman, L. P. Swiler, and J. P. Eddy, "Dakota, A Multilevel Parallel Object-Oriented Framework for Design Optimization, Parameter Estimation, Uncertainty Quantification, and Sensitivity Analysis: Version 6.5 User's Manual," 2016.

[24] IntelligentLight, "FieldView Reference Manual SOFTWARE RELEASE VERSION 16," Rutherford, 2015.

APPENDIX

Appendix A 2D Model Approach

The geometry, mesh, and boundary conditions for the 2-D Fluent model that were used prior to the 3-D model.

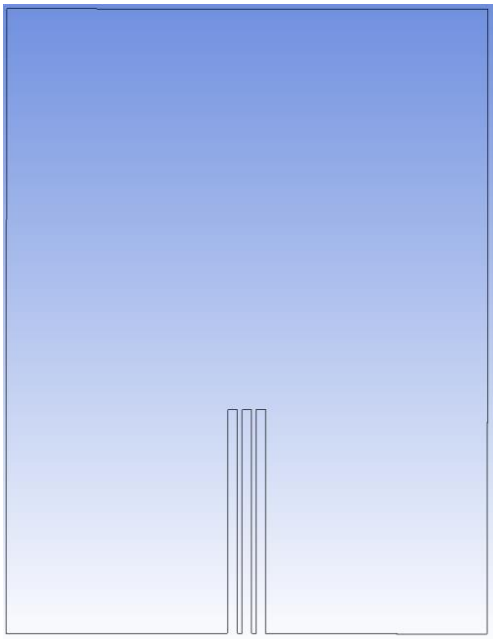


Figure 38: Image of the geometry configuration that was used to model the twin jet domain used in the 2-D CFD model.

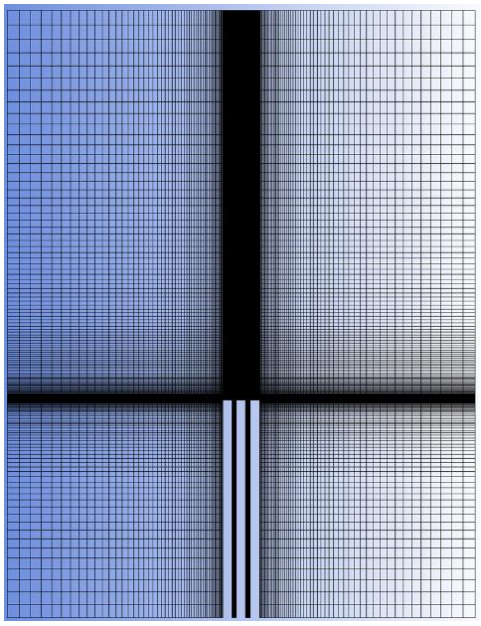


Figure 39: Image of the mesh that was created from the geometry of the twin jet domain used in the 2-D CFD model and consisting of 28,616 cells.

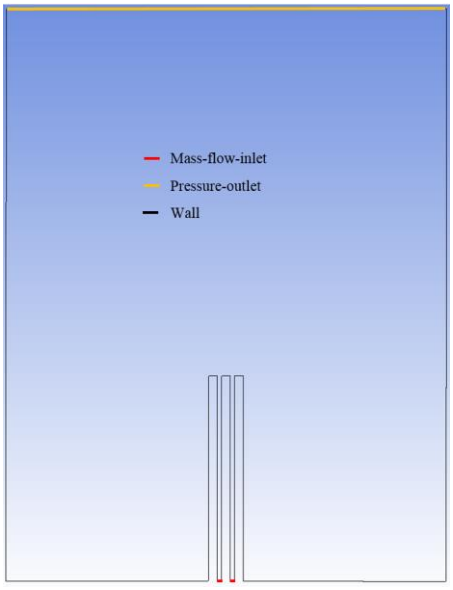


Figure 40: Image of the Boundary conditions applied to the 2-D CFD model in reference to the model domain and geometry.

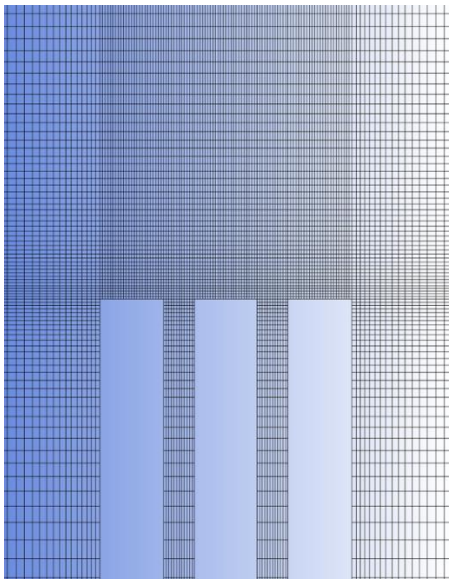


Figure 41: Visualization of the 2-D mesh zoomed-in to the regions near the nozzle to show the number of divisions across each jet.

Appendix B 2D Model Results

Some of the results obtained from the 2-D Fluent model that was implemented prior to the 3-D model are presented. shows a contour plot of the magnitude of velocity field taken from the 2-D simulation domain. Figures 1 and 2 show the velocity and pressure along the centerline for the various grid refinement levels used. Experimental data is not available for these locations currently. The processing of the solution data and images generation presented was done using the post-processing software FieldView available from Intelligent Light. All plots shown in the V&V effort were done using Matlab.

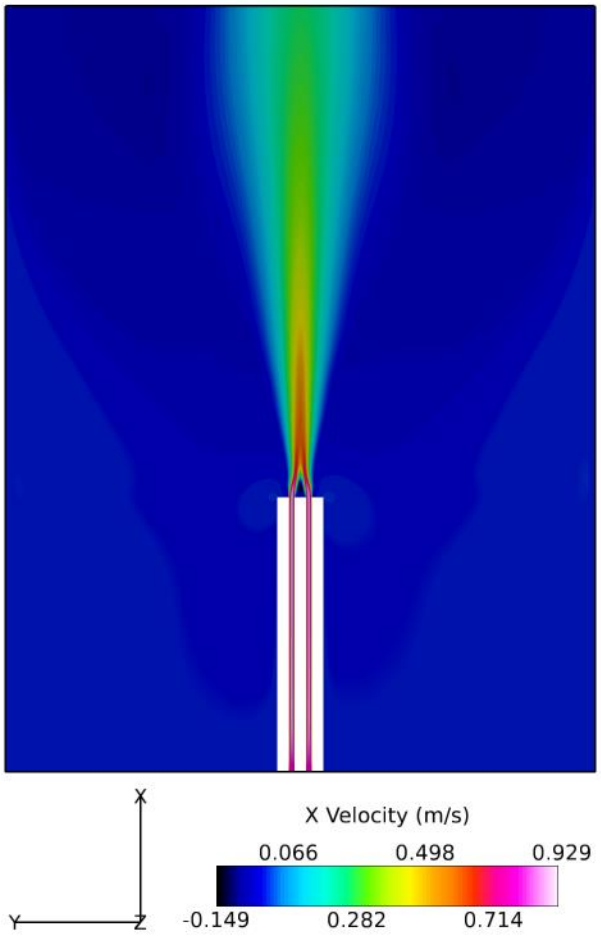


Figure 42: Velocity contour plot for the 2D Fluent model.

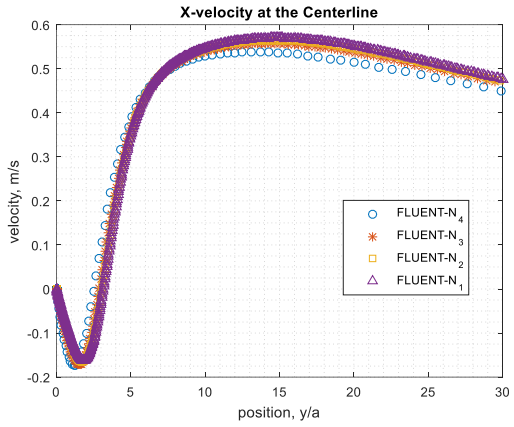


Figure 43: Velocity at the centerline for various grid refinement levels.

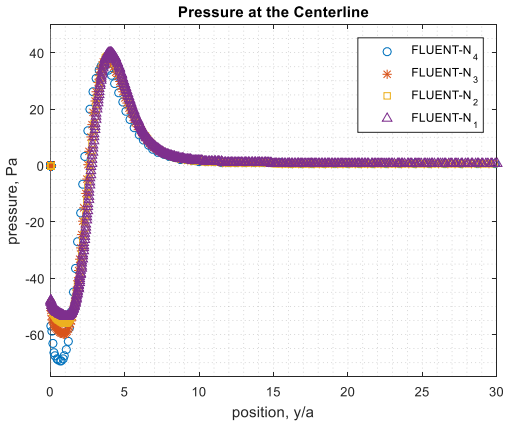


Figure 44: Pressure at the centerline for various grid refinement levels using the 2-D model.

The results of the GCI calculation for the 2-D model are tabulated for reference.

Table 12: Numerical Uncertainty computed for the 2-D Model

	$X\ velocity$ (m/s)	$Y\ velocity$ (m/s)	$Pressure$ (Pa)
$Location$	$y/a = 10.34$ $x/a = 0.665$	$y/a = 1.41$ $x/a = 1.50$	$y/a = 0$ $x/a = 0$
N_0	∞	∞	∞
N_1	1,797,952	1,797,952	1,797,952
N_2	450,112	450,112	450,112
N_3	112,840	112,840	112,840
N_4	28,416	28,416	28,416
r_{21}	2	2	2
r_{32}	2	2	2
φ_1	0.60	0.098	-47.78
φ_2	0.59	0.101	-48.85
φ_3	0.58	0.108	-50.88
φ_4	0.55	0.119	-56.78
p_{obs}^{13}	0.94	0.85	0.92
p_{obs}^{24}	1.09	0.65	1.54
φ_{ext}^{21}	0.61	0.093	-46.59
e_a^{21}	1.21%	3.82%	2.24%
e_{ext}^{21}	1.31%	5.04%	2.56%
Fs	1.25	1.25	1.25
GCI_{ext}^{21}	1.65%	6.00%	3.12%
$\varphi_1 \pm U_{num}$	$0.60 \pm 1.65\%$	$0.098 \pm 6.00\%$	$-47.78 \pm 3.12\%$
k	1.15	1.15	1.15
$\varphi_1 \pm u_{num}$	$0.60 \pm 1.44\%$	$0.098 \pm 5.21\%$	$-47.78 \pm 2.71\%$

The following Figures show a visual representation of the solution values as the mesh is refined. For this case, 0 represents an infinitely fine grid, and 4 is the coarse grid.

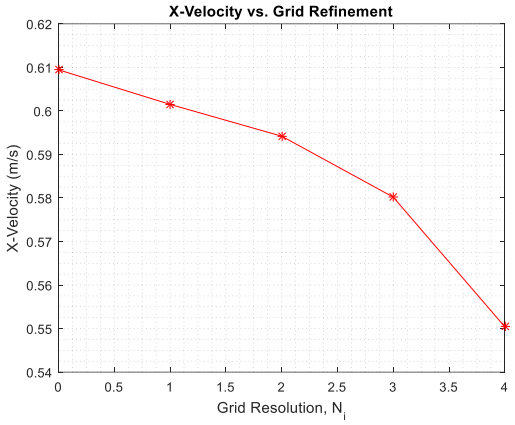


Figure 45: Grid refinement of $X\ velocity$ at the point $y/a = 10.34$ and $x/a = 0.665$.

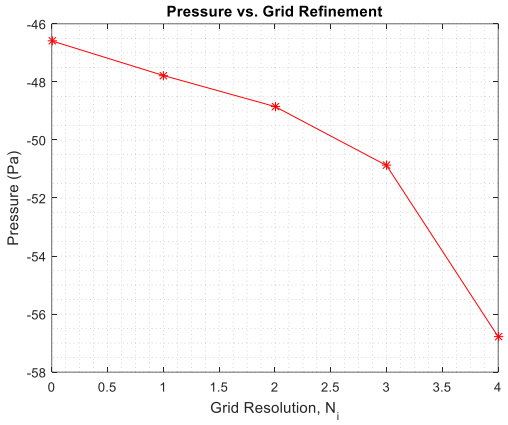


Figure 46: Grid refinement of the pressure at the point $y/a = 0$ and $x/a = 0$.

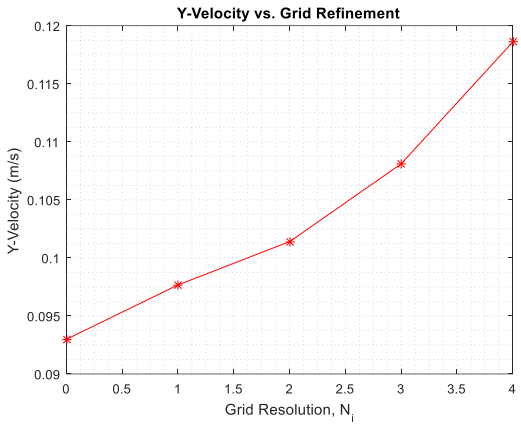


Figure 47: Grid refinement of the $X\ velocity$ at point $y/a = 1.41$ and $x/a = 1.50$.

Appendix C Additional 2D results

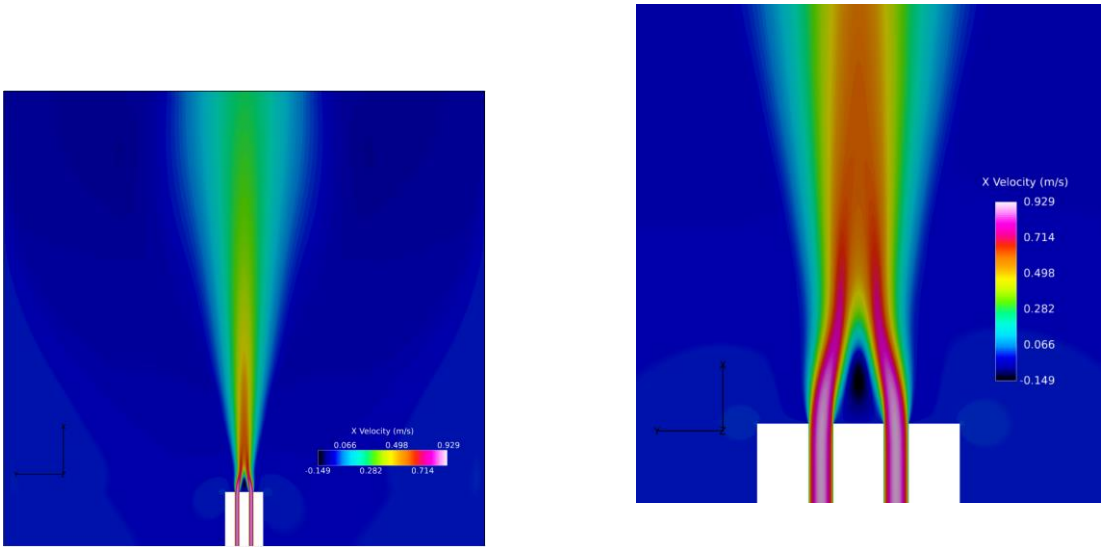


Figure 48: Contours of axial twin jet velocity flowfield that were generated using the 2-D CFD model.

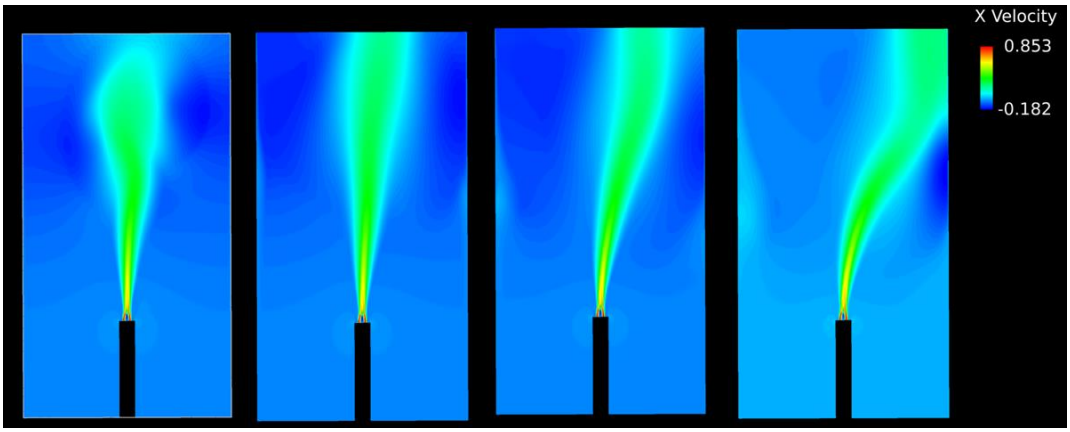


Figure 49: Non-physical behavior was observed in the 2D simulations and characterized by the attraction to the wall boundary as shown.

The behavior in Figure 49 was avoided by moving the side wall of the domain away from the jet so they would not interact.

Appendix D Additional 3D Results

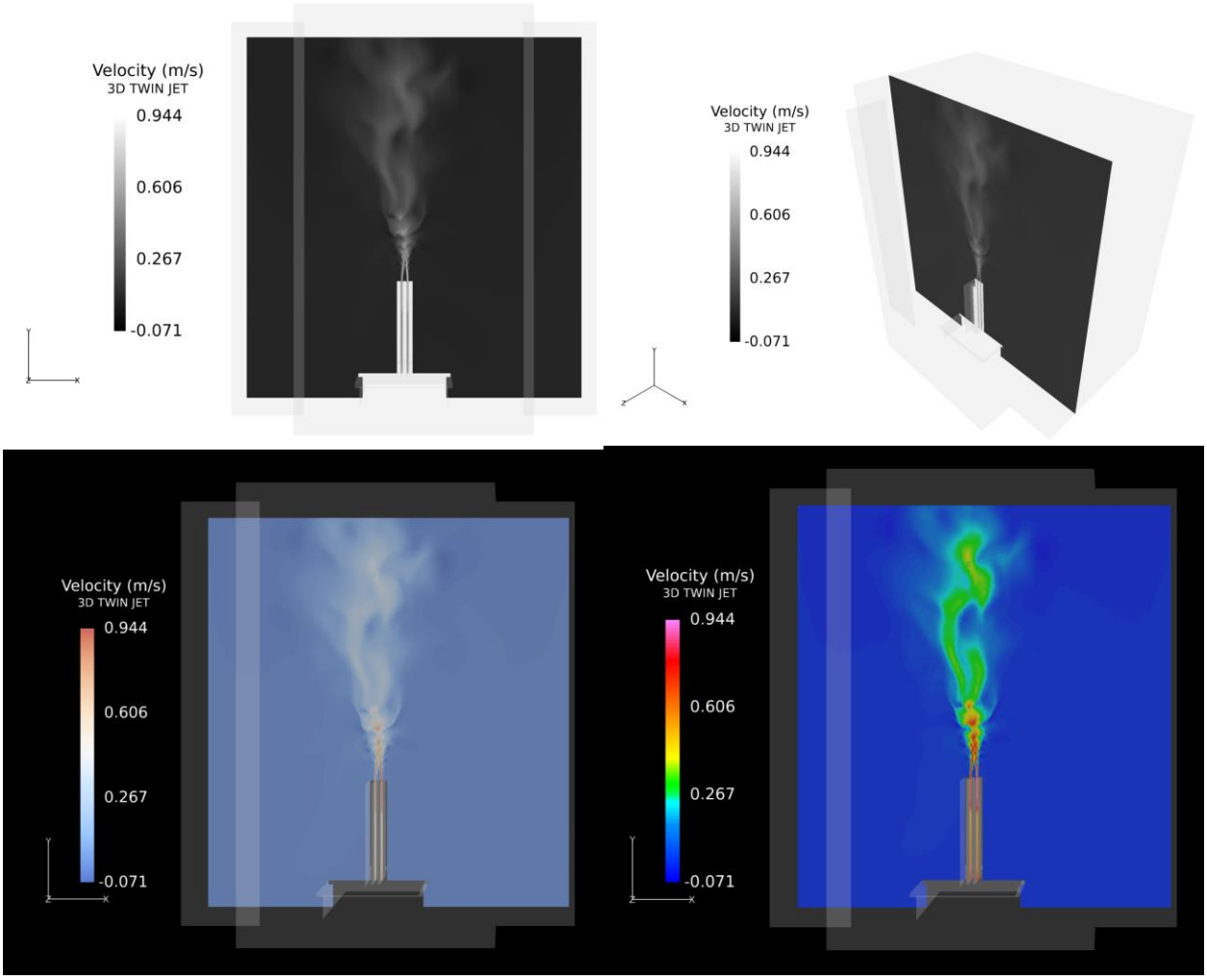


Figure 50: Visualization of velocity contours of the twin jet model that were observed in the early stages before the solution had converged. As the solver continued to iterate, the steady assumption was able to smooth out the contours and find the converged solution.

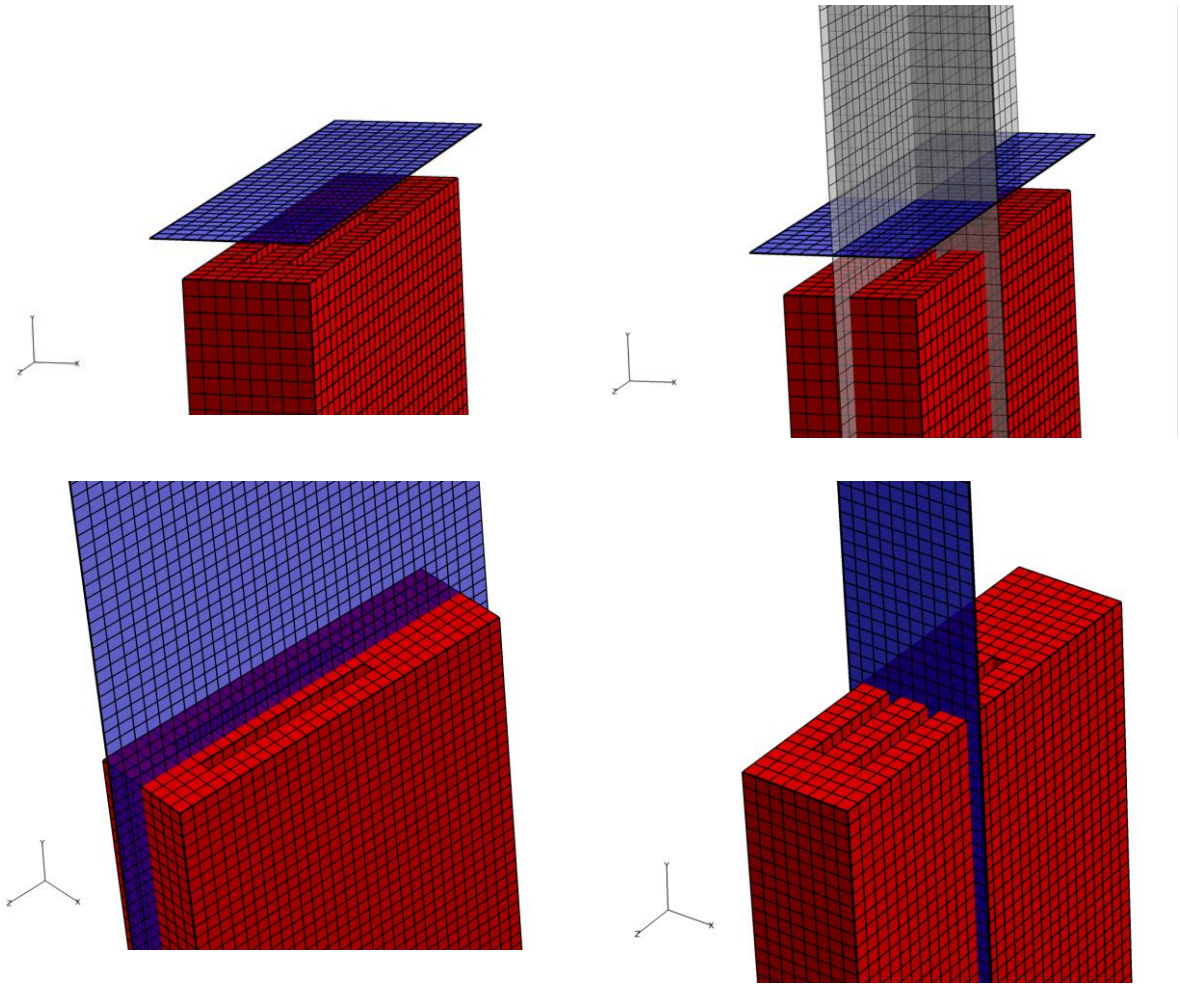


Figure 51: Images from various viewpoints to display additional information about the configuration of the grid in the region near the nozzle of the 3-D model.

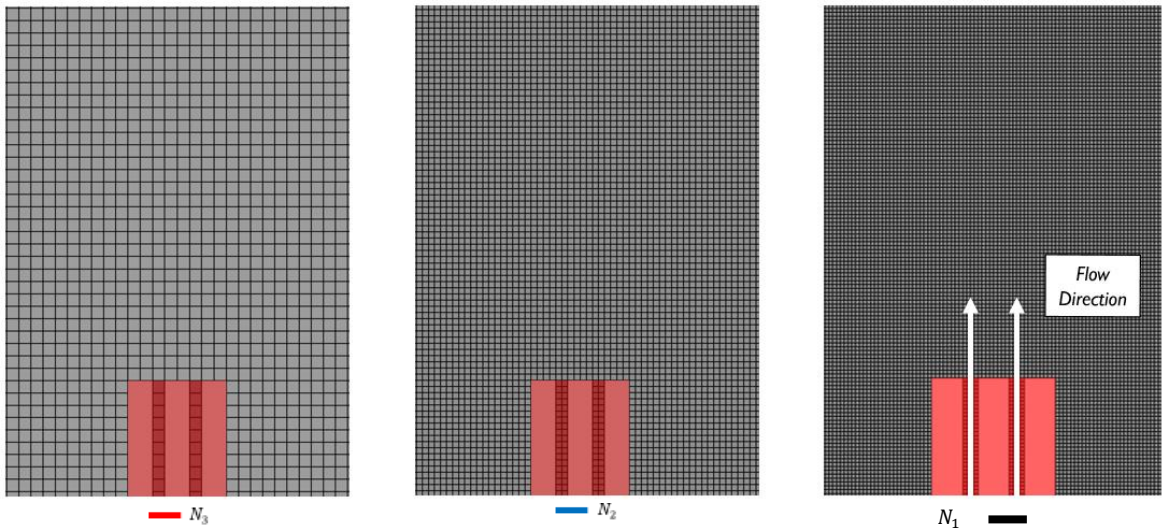


Figure 52: Images of the three grid refinements that were used in the 3-D numerical error estimation.

Appendix E Validation Experiment

Diagram of the experimental setup and nozzle geometry published by Wang and Hassan. This geometry configuration was used to create the CFD model geometry, by matching as closely as possible.

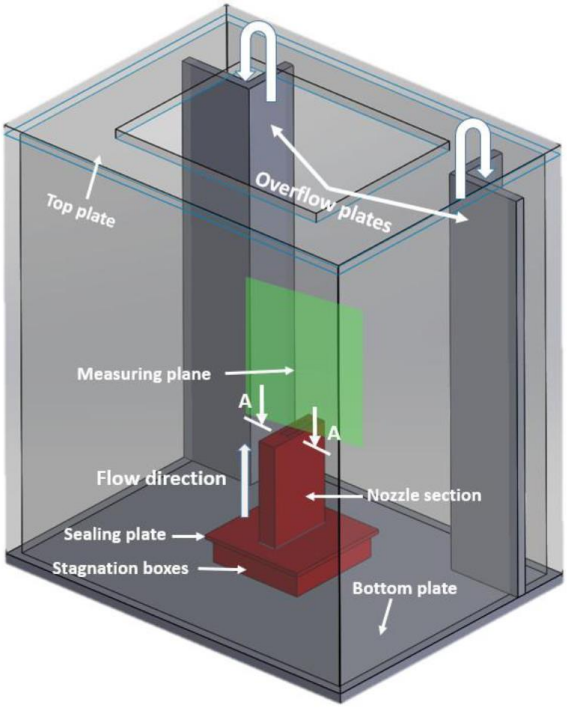


Figure 53: Experimental setup taken from Wang and Hassan. [2]

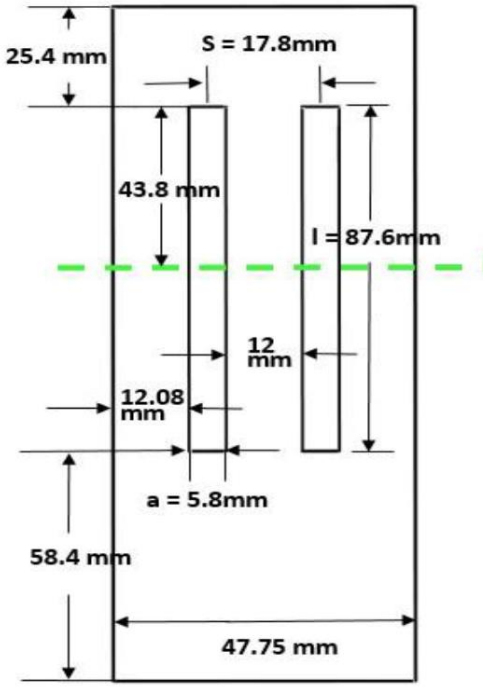


Figure 54: Nozzle specifications taken from Wang and Hassan.[2]

Appendix F Input Uncertainty Extras

The upper and lower bounds of the input uncertainty study for the 3-D twin jet are displayed in the following Figures. These bounds were used in the generation of the 95% confidence interval that was determined from the model input study.

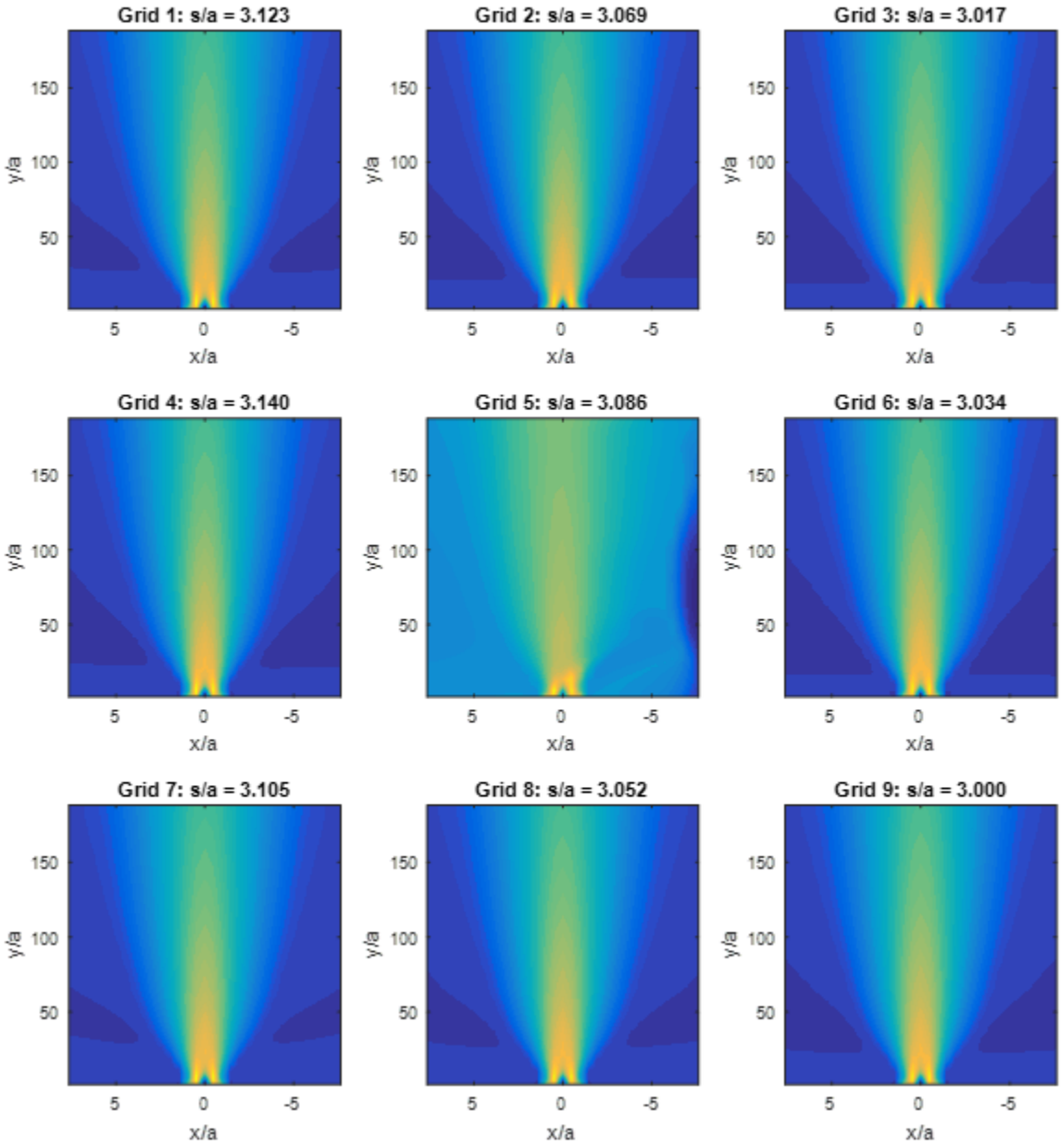


Figure 55: Lower bounds of 95% confidence interval that was obtained by conducting 90 simulations during the Dakota input study of the 3-D twin jet model. The contours represent the lower bounds of the axial velocity that was statistically computed for each of the nine jet spacing configurations.

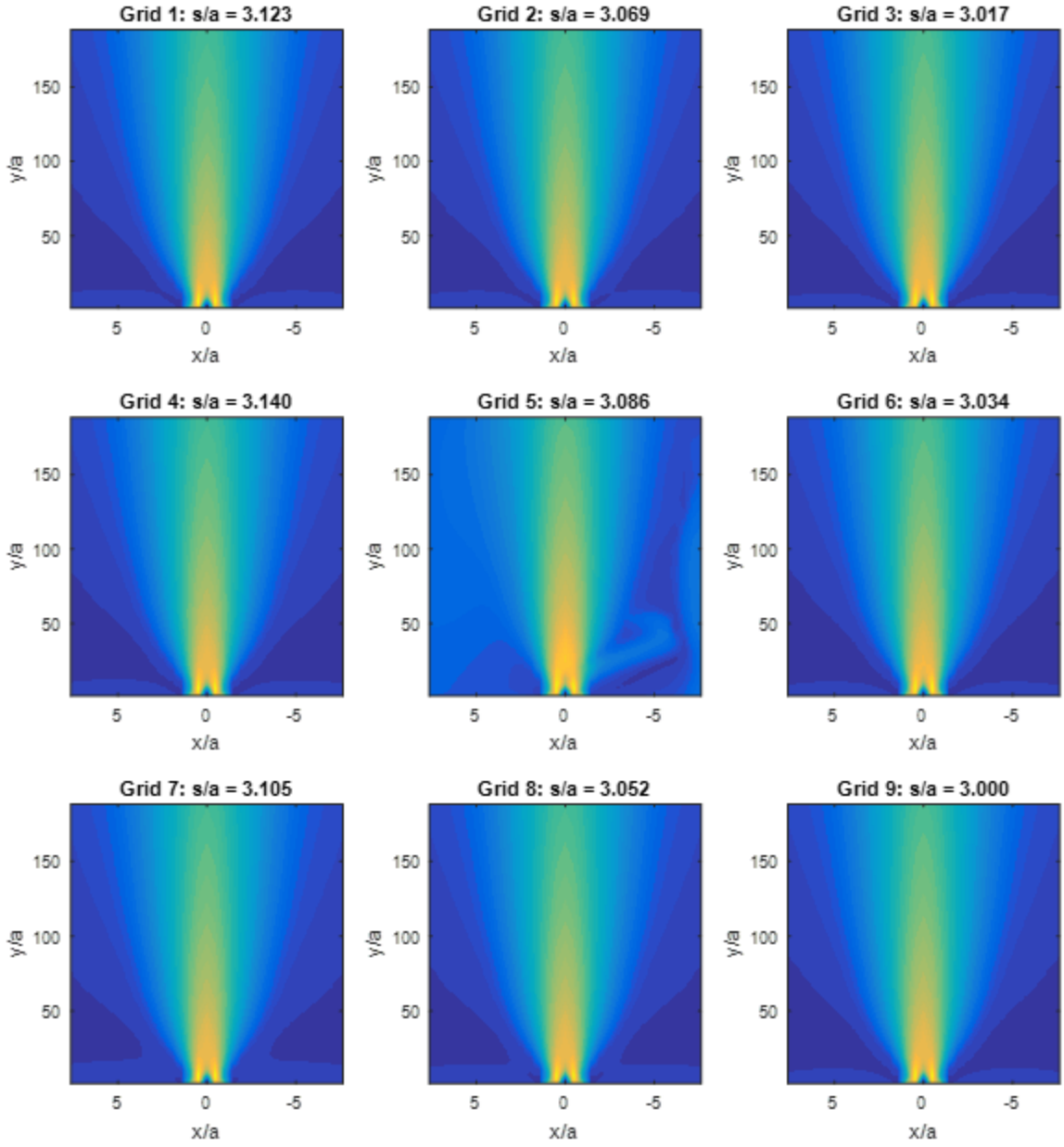


Figure 56: Upper bounds of 95% confidence interval that was obtained by conducting 90 simulations during the Dakota input study of the 3-D twin jet model. The contours represent the upper bounds of the axial velocity that was statistically computed for each of the nine jet spacing configurations.

Appendix G Report Summary – Fluent Model

The following report is a full listing of the configuration that was applied to the Fluent CFD model. The setting presented by this report, documents the actually configuration of the Fluent solver that was used to generate the results presented in the work.

```
Fluent
Version: 3d, dp, pbns, rke (3d, double precision, pressure-based, realizable
k-epsilon)
Release: 17.1.0
Title:

Models
------

   Model                         Settings
   ------------------------------------------------------------------
   Space                         3D
   Time                          Steady
   Viscous                       Realizable k-epsilon turbulence model
   Wall Treatment                Enhanced Wall Treatment
   Heat Transfer                 Disabled
   Solidification and Melting    Disabled
   Species                       Disabled
   Coupled Dispersed Phase       Disabled
   NOx Pollutants                Disabled
   SOx Pollutants                Disabled
   Soot                          Disabled
   Mercury Pollutants            Disabled

Material Properties
-------------------

   Material: water (fluid)

      Property                        Units      Method     Value(s)
      ------------------------------------------------------------
      Density                         kg/m3      constant   997.5
      Cp (Specific Heat)              j/kg-k     constant   4182
      Thermal Conductivity            w/m-k      constant   0.6
      Viscosity                       kg/m-s     constant   0.000937
      Molecular Weight                kg/kgmol   constant   18.0152
      Thermal Expansion Coefficient   1/k        constant   0
      Speed of Sound                  m/s        none       #f

   Material: aluminum (solid)

      Property               Units    Method     Value(s)
      ------------------------------------------------
      Density                kg/m3    constant   2719
      Cp (Specific Heat)     j/kg-k   constant   871
      Thermal Conductivity   w/m-k    constant   202.4
```

```
Cell Zone Conditions
--------------------

   Zones

      name      id   type
      --------------------
      fluid-1   4    fluid

   Setup Conditions

      fluid-1

         Condition                                          Value
         -------------------------------------------------------
         Frame Motion?                                      no
         Reference Frame Z-Component of Rotation-Axis       0
         Mesh Motion?                                       no

Boundary Conditions
-------------------

   Zones

      name          id   type
      ---------------------------------
      inlet_2       8    mass-flow-inlet
      inlet_1       7    mass-flow-inlet
      outlet        5    pressure-outlet
      wall-tank     6    wall
      wall-nozzle   9    wall

   Setup Conditions

      inlet_2

         Condition                          Value
         ------------------------------------------
         Mass Flow Rate (kg/s)              0.385
         Direction Specification Method     1
         Turbulent Specification Method     1
         Turbulent Intensity (%)            8
         Turbulent Length Scale (m)         0.0058

      inlet_1

         Condition                          Value
         ------------------------------------------
         Mass Flow Rate (kg/s)              0.385
         Direction Specification Method     1
         Turbulent Specification Method     1
         Turbulent Intensity (%)            8
         Turbulent Length Scale (m)         0.0058

      outlet

         Condition   Value
```

```
         -----------------

      wall-tank

         Condition                  Value
         --------------------------------
         Wall Motion                0
         Shear Boundary Condition   0

      wall-nozzle

         Condition                  Value
         --------------------------------
         Wall Motion                0
         Shear Boundary Condition   0

Solver Settings
---------------

   Equations

      Equation     Solved
      -------------------
      Flow         yes
      Turbulence   yes

   Numerics

      Numeric                           Enabled
      ---------------------------------------
      Absolute Velocity Formulation     yes

   Relaxation

      Variable                     Relaxation Factor
      ---------------------------------------------
      Density                      1
      Body Forces                  1
      Turbulent Kinetic Energy     0.8
      Turbulent Dissipation Rate   0.8
      Turbulent Viscosity          1

   Linear Solver

                                   Solver     Termination   Residual
Reduction
      Variable                     Type       Criterion     Tolerance
      ------------------------------------------------------------------------
-
      Flow                         F-Cycle    0.1
      Turbulent Kinetic Energy     Flexible   0.1           0.7
      Turbulent Dissipation Rate   Flexible   0.1           0.7

   Pressure-Velocity Coupling

      Parameter                            Value
      -----------------------------------------
```

```
   Type                                  Coupled
   Pseudo Transient                      no
   Flow Courant Number                   200
   Explicit momentum under-relaxation    0.5
   Explicit pressure under-relaxation    0.5

Discretization Scheme

   Variable                      Scheme
   ----------------------------------------------
   Pressure                      Second Order
   Momentum                      Second Order Upwind
   Turbulent Kinetic Energy      Second Order Upwind
   Turbulent Dissipation Rate    Second Order Upwind

Solution Limits

   Quantity                           Limit
   -------------------------------------
   Minimum Absolute Pressure          1
   Maximum Absolute Pressure          5e+10
   Minimum Temperature                1
   Maximum Temperature                5000
   Minimum Turb. Kinetic Energy       1e-14
   Minimum Turb. Dissipation Rate     1e-20
   Maximum Turb. Viscosity Ratio      100000
```

Appendix H Software Versions

The specific versions of all software that was used to generate the results presented are listed for future reference in the work.

- Fluent 16.2 & 17.1 from ANSYS
    - Model was generated on Windows machine using Workbench 16.2, sent to cluster for computations using Fluent 17.1.
- FieldView 15 & 16.1 from Intelligent Light
    - Some images generated using FieldView 15 on Windows machine, all processing done on the cluster used FieldView 16.1.
- DAKOTA 6.5 from Sandia National Laboratory
- Matlab 2015b from MathWorks
- PuTTY Release 0.63
- WinSCP Version 5.1.7

Appendix I Matlab Scripts

All the Matlab scripts that were used to generate the input files necessary in the Dakota input study are provided as a future reference in the work.

I.1 Observed Order and GCI

```
% ---------------------------
% -- Plot Numerical Errors
%---------------------------
clc; clear; close all
yv3=dlmread('F:\NAU Grad 2015-2017\Thesis\Model\LONG LAST\UQ\Unum\y_vel_1a.txt');
yv2=dlmread('F:\NAU Grad 2015-2017\Thesis\Model\LONG LAST\UQ\Unum\y_vel_2a.txt');
yv1=dlmread('F:\NAU Grad 2015-2017\Thesis\Model\LONG LAST\UQ\Unum\y_vel_4a.txt');
% calculate observed order of accuracy, p
e32 = abs(yv3-yv2); e21=abs(yv2-yv1);
pc=log(e32./e21)/log(2); p=zeros(34,127); GCI=p; vb=p; Unum=vb;
count = 0;
for j = 1:127
    for i = 1:34
        count=count+1;
        if abs(yv1(count)-yv3(count)) < 0.008
            p(i,j)=1;
            GCI(i,j) = 3*abs(yv1(count)-yv3(count));
            vb(i,j) = yv1(count)+((yv1(count)-yv2(count))/(2^pc(count)-1));
            Unum(i,j) = 3*abs(yv1(count)-yv3(count));
        elseif pc(count) > 2
            p(i,j) = 2.3;
            GCI(i,j) = ((1.25*e21(count))/(2^pc(count)-1))/1.15;
            vb(i,j) = yv1(count)+((yv1(count)-yv2(count))/(2^pc(count)-1));
            Unum(i,j) = 1.25*abs(yv1(count)-vb(count));
        elseif pc(count) < 1
            p(i,j) = -1;
            GCI(i,j) = 3*abs(yv1(count)-yv3(count));
            vb(i,j) = yv1(count)+((yv1(count)-yv2(count))/(2^pc(count)-1));
            Unum(i,j) = 3*abs(yv1(count)-yv3(count));
        else
           p(i,j)=pc(count);
           GCI(i,j) = ((1.25*e21(count))/(2^pc(count)-1))/1.15;
           vb(i,j) = yv1(count)+((yv1(count)-yv2(count))/(2^pc(count)-1));
           Unum(i,j) = 1.25*abs(yv1(count)-vb(count));
        end
        if GCI(i,j) > 0.1
            GCI(i,j) = 0.11;
        end
    end
end
xplot=linspace(-7,6.6,34); yplot=linspace(0,200,127);
figure(1)
contourf(yplot,xplot,p,24,'edgecolor','none')
h=colorbar(); view([270 90]);
title('Observed Order of Accuracy, p'); ylabel('x/a'); xlabel('y/a')
xlhand = get(gca,'xlabel'); set(gca,'FontSize',16)
figure(2)
contourf(yplot,xplot,GCI,24,'edgecolor','none')
h=colorbar(); view([270 90]);
title('Numerical Uncertainty, GCI');
ylabel('x/a'); xlabel('y/a'); set(gca,'FontSize',16)
yv1=reshape(yv1,[34,127]);
yv2=reshape(yv2,[34,127]);
yv3=reshape(yv3,[34,127]);
figure(3)
plot(1:127,yv1(18,:),1:127,yv2(18,:),1:127,yv3(18,:))
```

I.2 Input Uncertainty

```
% ----------------------------------------
% -- Plots of Input uncertainty from Dakota
% ----------------------------------------
clc; clear; close all; format long
np = 4318; ng=9;
fid=fopen('F:\NAU Grad 2015-2017\Thesis\Model\LONG
LAST\UQ\fine_grid_10samps_100it\dakota.out','r');
C=textscan(fid,'%s','delimiter','\n');
fclose(fid);
% moment stats for repsonses
mean_v=zeros(np,ng); stdDv=mean_v; skwns=mean_v;kurt=mean_v;
D_stat = strfind(C{1},'moment');
fid=fopen('sample_stats.txt','w');
for i=1:1:length(D_stat)-1;
if D_stat{i} > 0
    fprintf('%d\n',D_stat{i});
    fprintf(fid,'%s \n%s\n',C{1}{i},C{1}{i+1:i+np+1});
end
end
fclose(fid);
fid=fopen('sample_stats.txt','r');
for i =1:ng
file_header=textscan(fid,'%s',2,'delimiter','\n');
Cr=textscan(fid,'%s %f %f %f %f');
mean_v(:,i)=Cr{2}; stdDv(:,i)=Cr{3}; skwns(:,i)=Cr{4}; kurt(:,i)=Cr{5};
end
fclose(fid);
% mean for min, max at 95% confidence for responses
mean_min=mean_v;mean_max=mean_v;stdDv_min=mean_v; stdDv_max=mean_v; ci=mean_v;
D_ci = strfind(C{1},'95%');
fid=fopen('95_CI.txt','w');
for i=1:1:length(D_ci)-1;
if D_ci{i} > 0
    fprintf('%d\n',D_ci{i});
    fprintf(fid,'%s \n%s\n',C{1}{i},C{1}{i+1:i+np+1});
end
end
fclose(fid);
fid=fopen('95_CI.txt','r');
for i=1:ng
file_header=textscan(fid,'%s',2,'delimiter','\n');
Cr=textscan(fid,'%s %f %f %f %f');
mean_min(:,i)=Cr{2}; stdDv_min(:,i)=Cr{4};
mean_max(:,i)=Cr{3}; stdDv_max(:,i)=Cr{5};
ci(:,i) = abs(mean_max(:,i)-mean_min(:,i));
end
xmod=13:34:4318;
for i=xmod
    mean_min(i,1)=nan;
    mean_max(i,1)=nan;
    mean_v(i,1)=nan;
    ci(i,1)=nan;
end
ci(isnan(ci(:,1)),:) = [];
mean_min(isnan(mean_min(:,1)),:) = [];
mean_max(isnan(mean_max(:,1)),:) = [];
mean_v(isnan(mean_v(:,1)),:) = [];
fclose(fid);
fid=fopen('xpoints.txt','r');
C=textscan(fid,'%n%s');
fclose(fid);
```

```
x=C{1};
fid=fopen('ypoints.txt','r');
C=textscan(fid,'%n%s');
fclose(fid);
y=C{1};
Dci = zeros(length(x)-1,length(y)-1,8);
Dmin = zeros(length(x)-1,length(y)-1,8);
Dmax = zeros(length(x)-1,length(y)-1,8);
Dmean = zeros(length(x)-1,length(y)-1,8);
Dplus = zeros(length(x)-1,length(y)-1,8);
Dminus = zeros(length(x)-1,length(y)-1,8);
ngk=0; ngmod=[1:4,6:9];
for k=ngmod
    ngk=ngk+1;
    count = 0;
for j = 1:length(y)-1
    for i = 1:length(x)-1
        count = count+1;
        Dci(i,j,ngk) = ci(count,k);
        Dmin(i,j,ngk) = mean_min(count,k);
        Dmax(i,j,ngk) = mean_max(count,k);
        Dmean(i,j,ngk) = mean_v(count,k);
        Dplus(i,j,ngk) = abs(Dmax(i,j,ngk)-Dmean(i,j,ngk));
        Dminus(i,j,ngk) = abs(Dmean(i,j,ngk)-Dmin(i,j,ngk));
    end
end
end
%combined a+e ya3448
AEmean = (Dmax+Dmin)/2; Apm=abs(Dmax-Dmin)/2;
Dp33=zeros(33,8); dp33=zeros(33,1); Dm33=zeros(33,8); dm33=zeros(33,1);
AEpm33=Dp33; apm33=dp33; AEmean33=Dp33; aemeanu33=dp33;aemeanl33=dp33;
AEpm10=Dp33; apm10=dp33; AEmean10=Dp33; aemeanu10=dp33;aemeanl10=dp33;
for k=1:8
    Dp33(:,k)=Dplus(:,33,k);
    Dm33(:,k)=Dminus(:,33,k);
    AEpm33(:,k)=Apm(:,33,k);
    AEmean33(:,k)=AEmean(:,33,k);
end
% Dp33(:,5)=0; Dm33(:,5)=0;
for i=1:33
    dp33(i)=max(Dp33(i,:));
    dm33(i)=max(Dm33(i,:));
    apm33(i)=max(AEpm33(i,:));
    aemeanu33(i)=max(AEmean33(i,:));
    aemeanl33(i)=min(AEmean33(i,:));
end
epm33=(aemeanu33-aemeanl33)/2; aemeant33=(aemeanu33+aemeanl33)/2;
aepmt33=apm33+epm33;
%combined a+e ya1034
Dp10=zeros(33,9); dp10=zeros(33,1); Dm10=zeros(33,9); dm10=zeros(33,1);
for k=1:8
    Dp10(:,k)=Dplus(:,10,k);
    Dm10(:,k)=Dminus(:,10,k);
    AEpm10(:,k)=Apm(:,10,k);
    AEmean10(:,k)=AEmean(:,10,k);
end
% Dp10(:,5)=0; Dm10(:,5)=0;
for i=1:33
    dp10(i)=max(Dp10(i,:));
    dm10(i)=max(Dm10(i,:));
    apm10(i)=max(AEpm10(i,:));
    aemeanu10(i)=max(AEmean10(i,:));
    aemeanl10(i)=min(AEmean10(i,:));
```

```
end
epm10=(aemeanu10-aemeanl10)/2; aemeant10=(aemeanu10+aemeanl10)/2;
aepmt10=apm10+epm10;
% plots
c=48; % number of contours in plots
% experimental data
data_exp_y1034 = dlmread('y1034_data.txt');
data_exp_y3448 = dlmread('y3448_data.txt');
yplot = (1:127)*(1.092/127)*1000/5.8;
xplot = (-16:16)*(0.0941/34)*1000/5.8;
xin = dlmread('xpoints.txt')*1000/5.8; xaplot=[xin(1:12,1);xin(14:end,1)];
for i=1:8
figure(1)
subplot(3,3,i)
contourf(yplot,xplot,Dci(:,:,i),c,'edgecolor','none')
% title('95% Confidence Interval');
ylabel('x/a'); xlabel('y/a');
view([270 90]); grid MINOR
end
for i=1:8
figure(2)
subplot(3,3,i)
contourf(yplot,xplot,Dmin(:,:,i),c,'edgecolor','none')
% title('95% CI - Lower Bound');
ylabel('x/a');xlabel('y/a');
view([270 90])
end
for i=1:8
figure(3)
subplot(3,3,i)
contourf(yplot,xplot,Dmax(:,:,i),c,'edgecolor','none')
% title('95% CI - Upper Bound');
ylabel('x/a'); xlabel('y/a');
view([270 90])
end
for i=1:8
figure(4)
subplot(3,3,i)
contourf(yplot,xplot,Dmean(:,:,i),c,'edgecolor','none')
% title('95% CI - Mean');
ylabel('x/a'); xlabel('y/a');
% h=colorbar(); set(h, 'ylim',[min(min(mean(:,:))),max(max(mean(:,:)))]);
view([270 90])
end

figure(1); % Mag CI
subplot(3,3,1)
title('Grid 1: s/a = 3.123');
subplot(3,3,2)
title('Grid 2: s/a = 3.069');
subplot(3,3,3)
title('Grid 3: s/a = 3.017');
subplot(3,3,4)
title('Grid 4: s/a = 3.140');
subplot(3,3,5)
title('Grid 5: s/a = 3.086');
subplot(3,3,6)
title('Grid 6: s/a = 3.034');
subplot(3,3,7)
title('Grid 7: s/a = 3.105');
subplot(3,3,8)
title('Grid 8: s/a = 3.052');
subplot(3,3,9)
```

```
title('Grid 9: s/a = 3.000'); set(gca,'FontSize',16)
figure(2); % Minn CI
subplot(3,3,1)
title('Grid 1: s/a = 3.123');
subplot(3,3,2)
title('Grid 2: s/a = 3.069');
subplot(3,3,3)
title('Grid 3: s/a = 3.017');
subplot(3,3,4)
title('Grid 4: s/a = 3.140');
subplot(3,3,5)
title('Grid 5: s/a = 3.086');
subplot(3,3,6)
title('Grid 6: s/a = 3.034');
subplot(3,3,7)
title('Grid 7: s/a = 3.105');
subplot(3,3,8)
title('Grid 8: s/a = 3.052');
subplot(3,3,9)
title('Grid 9: s/a = 3.000'); set(gca,'FontSize',16)
figure(3); % Max CI
subplot(3,3,1)
title('Grid 1: s/a = 3.123');
subplot(3,3,2)
title('Grid 2: s/a = 3.069');
subplot(3,3,3)
title('Grid 3: s/a = 3.017');
subplot(3,3,4)
title('Grid 4: s/a = 3.140');
subplot(3,3,5)
title('Grid 5: s/a = 3.086');
subplot(3,3,6)
title('Grid 6: s/a = 3.034');
subplot(3,3,7)
title('Grid 7: s/a = 3.105');
subplot(3,3,8)
title('Grid 8: s/a = 3.052');
subplot(3,3,9)
title('Grid 9: s/a = 3.000'); set(gca,'FontSize',16)
figure(4); % Mean CI
subplot(3,3,1)
title('Grid 1: s/a = 3.123');
subplot(3,3,2)
title('Grid 2: s/a = 3.069');
subplot(3,3,3)
title('Grid 3: s/a = 3.017');
subplot(3,3,4)
title('Grid 4: s/a = 3.140');
subplot(3,3,5)
title('Grid 5: s/a = 3.086');
subplot(3,3,6)
title('Grid 6: s/a = 3.034');
subplot(3,3,7)
title('Grid 7: s/a = 3.105');
subplot(3,3,8)
title('Grid 8: s/a = 3.052');
subplot(3,3,9)
title('Grid 9: s/a = 3.000'); set(gca,'FontSize',16)
% Line plots at validation locations
for i=1:8
figure(5)
subplot(3,3,i); hold on
errorbar(xaplot,Dmean(:,33,i),Dminus(:,33,i),Dplus(:,33,i),'k-.');
```

```
axis([-7 7 0 0.55]); xlabel('x/a'); ylabel('velocity, m/s')
plot(data_exp_y3448(1:23),data_exp_y3448(25:47),'b.:'); grid MINOR
legend('Fluent y/a=34.48','Wang & Hassan data','Location','South')
figure(6)
subplot(3,3,i); hold on
errorbar(xaplot,Dmean(:,10,i),Dminus(:,33,i),Dplus(:,10,i),'k-.');
axis([-7 7 -0.05 0.65]); xlabel('x/a'); ylabel('velocity, m/s')
plot(data_exp_y1034(1:23),data_exp_y1034(25:47),'b.:'); grid MINOR
legend('Fluent y/a=10.34','Wang & Hassan data','Location','NorthEast')
end
subplot(3,3,1)
title('Grid 1: s/a = 3.123');
subplot(3,3,2)
title('Grid 2: s/a = 3.069');
subplot(3,3,3)
title('Grid 3: s/a = 3.017');
subplot(3,3,4)
title('Grid 4: s/a = 3.140');
subplot(3,3,5)
title('Grid 5: s/a = 3.086');
subplot(3,3,6)
title('Grid 6: s/a = 3.034');
subplot(3,3,7)
title('Grid 7: s/a = 3.105');
subplot(3,3,8)
title('Grid 8: s/a = 3.052');
subplot(3,3,9)
title('Grid 9: s/a = 3.000'); set(gca,'FontSize',16)
%combine a+e uncertainty
figure(7) %ya3448
% errorbar(xaplot,aemeant33,aepmt33,'k-.'); hold on
% errorbar(xaplot,aemeanu33,apm33,'m-.');
% errorbar(xaplot,aemeanl33,apm33,'g-.');
% axis([-7 7 0 .6])
% xlabel('x/a'); ylabel('velocity, m/s')
GCI33=[0.000540122855455
   0.000545560615137
   0.000545279821381
   0.000530638033524
   0.000471304636449
   0.000956480624154
   0.013801324646920
   0.000238557031250
   0.000423743763525
   0.000778192820850
   0.000823202591795
   0.000683078751225
   0.000433736040729
   0.000088341680844
   0.017238020896912
   0.002799810069701
   0.005204314286879
   0.003057198107195
   0.014892965555191
   0.000120740969185
   0.000590442400685
   0.000962494333029
   0.001203857622234
   0.001222469227434
   0.000836254272512
   0.000523121206518
   0.017915729200467
   0.003525960724801
```

```
   0.001608042279258
   0.001431673066691
   0.001311547122896
   0.001200973056257
   0.001095556654036]; Unum33=GCI33;
dx33=data_exp_y3448(1:23); dy33=data_exp_y3448(25:47);
dyy33 = pchip(dx33,dy33,xaplot); Umod33=ones(33,1);
for i =1:33
    if (aemeant33(i)+aepmt33(i))-dyy33(i) < 0 %|| (aemeant33(i)-aepmt33(i))-dyy(i)>0
        Umod33(i)=abs(aemeant33(i)-dyy33(i))-aepmt33(i);
    elseif (aemeant33(i)+aepmt33(i))-dyy33(i) > 0
        Umod33(i)=0;
    end
end
Ud33=dyy33*0.016;
% set(gca,'FontSize',20)
% errorbar(xaplot,aemeant33,Umod33,'m-.'); hold on; set(gca,'FontSize',20)
% Uall33=[aemeant33-aepmt33-Unum33-Umod33-
Ud33,Ud33,Umod33,Unum33,2*aepmt33,Unum33,Umod33,Ud33];
%%
Uall33=[aemeant33-aepmt33-Unum33-Umod33-
Ud33,Ud33,Umod33,Unum33,2*aepmt33,Unum33,Umod33,Ud33];
figure;
h=area(xaplot,Uall33,'LineStyle','none'); hold on
h(1).FaceColor = [1 1 1];
h(2).FaceColor = [0 0 1];
h(3).FaceColor = [0 1 0];
h(4).FaceColor = [1 0 0];
h(5).FaceColor = [0 0 0];
h(5).FaceColor = [0 0 0];
h(6).FaceColor = [1 0 0];
h(7).FaceColor = [0 1 0];
h(8).FaceColor = [0 0 1];
set(gca,'FontSize',20);
legend('Uncertainty:','Experiment, U_D','Fluent, U_m_o_d_e_l','Fluent,
U_n_u_m','Fluent, U_i_n_p_u_t','Location','NorthEast')
% errorbar(xaplot,aemeant33,Unum33,'r-.'); hold on; set(gca,'FontSize',16)
axis([-7 7 0 .6]); xlabel('x/a'); ylabel('axial velocity, m/s')
% total uncertainty in SRQ Unum+Uinput+Ud+Umodel
% Usrq33 = aepmt33+Unum33+Ud33+Umod33;
% errorbar(xaplot,aemeant33,Usrq33,'k-.')
% plot(xaplot,dyy33,'bs:');
% legend('Uncertainty:','Data, U_D','Fluent, U_m_o_d_e_l','Fluent, U_n_u_m','Fluent,
U_i_n_p_u_t','Fluent, U_S_R_Q_,_9_5_%','Wang & Hassan data','Location','NorthEast')
% legend('Uncertainty:','Fluent, U_i_n_p_u_t')
%%
Uall33=[aemeant33-aepmt33-Unum33-Umod33,Umod33,Unum33,2*aepmt33,Unum33,Umod33];
figure;
h=area(xaplot,Uall33,'LineStyle','none'); hold on
h(1).FaceColor = [1 1 1];
% h(2).FaceColor = [0 0 1];
h(2).FaceColor = [0 1 0];
h(3).FaceColor = [1 0 0];
h(4).FaceColor = [0 0 0];
h(4).FaceColor = [0 0 0];
h(5).FaceColor = [1 0 0];
h(6).FaceColor = [0 1 0];
% h(8).FaceColor = [0 0 1];
set(gca,'FontSize',20);
legend('Uncertainty:','Fluent, U_m_o_d_e_l','Fluent, U_n_u_m','Fluent,
U_i_n_p_u_t','Location','NorthEast')
% errorbar(xaplot,aemeant33,Unum33,'r-.'); hold on; set(gca,'FontSize',16)
axis([-7 7 0 .6]); xlabel('x/a'); ylabel('axial velocity, m/s')
```

```
% errorbar(xaplot,dyy,Ud33,'g-.');
% plot(xaplot,dyy33,'bs:'); grid MINOR
% title('Contributions to Total Uncertainty')
% legend('Fluent, U_i_n_p_u_t','Fluent, U_n_u_m','Fluent, U_m_o_d_e_l','Data,
U_D','Wang & Hassan data','Location','NorthEast')
% legend('Fluent, U_m_o_d_e_l','Wang & Hassan data','Location','NorthEast')
% total uncertainty in SRQ Unum+Uinput+Ud+Umodel
% Usrq33 = aepmt33+Unum33+Ud33+Umod33;
% errorbar(xaplot,aemeant33,Usrq33,'k-.')
% plot(xaplot,dyy33,'bs:');
% legend('Uncertainty:','Data, U_D','Fluent, U_m_o_d_e_l','Fluent, U_n_u_m','Fluent,
U_i_n_p_u_t','Fluent, U_S_R_Q_,_9_5_%','Wang & Hassan data','Location','NorthEast')
% legend('Uncertainty:','Fluent, U_i_n_p_u_t')
%%
figure;
Uall33=[aemeant33-aepmt33,2*aepmt33];
h=area(xaplot,Uall33,'LineStyle','none'); hold on
h(1).FaceColor = [1 1 1];
% h(2).FaceColor = [0 0 1];
% h(3).FaceColor = [0 1 0];
% h(4).FaceColor = [1 0 0];
h(2).FaceColor = [0 0 0];
h(2).FaceColor = [0 0 0];
% h(6).FaceColor = [1 0 0];
% h(7).FaceColor = [0 1 0];
% h(8).FaceColor = [0 0 1];
set(gca,'FontSize',20);
% legend('Uncertainty:','Data, U_D','Fluent, U_m_o_d_e_l','Fluent, U_n_u_m','Fluent,
U_i_n_p_u_t','Location','NorthEast')
% errorbar(xaplot,aemeant33,Unum33,'r-.'); hold on; set(gca,'FontSize',16)
axis([-7 7 0 .6]); xlabel('x/a'); ylabel('axial velocity, m/s')
% errorbar(xaplot,dyy,Ud33,'g-.');
% plot(xaplot,dyy33,'bs:'); grid MINOR
% title('Contributions to Total Uncertainty')
% legend('Fluent, U_i_n_p_u_t','Fluent, U_n_u_m','Fluent, U_m_o_d_e_l','Data,
U_D','Wang & Hassan data','Location','NorthEast')
% legend('Fluent, U_m_o_d_e_l','Wang & Hassan data','Location','NorthEast')
% total uncertainty in SRQ Unum+Uinput+Ud+Umodel
% Usrq33 = aepmt33+Unum33+Ud33+Umod33;
% errorbar(xaplot,aemeant33,Usrq33,'k-.')
% plot(xaplot,dyy33,'bs:');
% legend('Uncertainty:','Data, U_D','Fluent, U_m_o_d_e_l','Fluent, U_n_u_m','Fluent,
U_i_n_p_u_t','Fluent, U_S_R_Q_,_9_5_%','Wang & Hassan data','Location','NorthEast')
legend('Uncertainty:','Fluent, U_i_n_p_u_t')
%%
figure;
Uall33=[aemeant33-aepmt33-Unum33,Unum33,2*aepmt33,Unum33];
h=area(xaplot,Uall33,'LineStyle','none'); hold on
h(1).FaceColor = [1 1 1];
% h(2).FaceColor = [0 0 1];
% h(3).FaceColor = [0 1 0];
h(2).FaceColor = [1 0 0];
h(3).FaceColor = [0 0 0];
h(3).FaceColor = [0 0 0];
h(4).FaceColor = [1 0 0];
% h(7).FaceColor = [0 1 0];
% h(8).FaceColor = [0 0 1];
set(gca,'FontSize',20);
% legend('Uncertainty:','Data, U_D','Fluent, U_m_o_d_e_l','Fluent, U_n_u_m','Fluent,
U_i_n_p_u_t','Location','NorthEast')
% errorbar(xaplot,aemeant33,Unum33,'r-.'); hold on; set(gca,'FontSize',16)
axis([-7 7 0 .6]); xlabel('x/a'); ylabel('axial velocity, m/s')
% errorbar(xaplot,dyy,Ud33,'g-.');
```

```
% plot(xaplot,dyy33,'bs:'); grid MINOR
% title('Contributions to Total Uncertainty')
% legend('Fluent, U_i_n_p_u_t','Fluent, U_n_u_m','Fluent, U_m_o_d_e_l','Data,
U_D','Wang & Hassan data','Location','NorthEast')
% legend('Fluent, U_m_o_d_e_l','Wang & Hassan data','Location','NorthEast')
% total uncertainty in SRQ Unum+Uinput+Ud+Umodel
% Usrq33 = aepmt33+Unum33+Ud33+Umod33;
% errorbar(xaplot,aemeant33,Usrq33,'k-.')
% plot(xaplot,dyy33,'bs:');
% legend('Uncertainty:','Data, U_D','Fluent, U_m_o_d_e_l','Fluent, U_n_u_m','Fluent,
U_i_n_p_u_t','Fluent, U_S_R_Q_,_9_5_%','Wang & Hassan data','Location','NorthEast')
legend('Uncertainty:','Fluent, U_n_u_m','Fluent, U_i_n_p_u_t')
%%
figure;
%total uncertainty in SRQ Unum+Uinput+Ud+Umodel
Usrq33 = aepmt33+Unum33+Ud33+Umod33;
errorbar(xaplot,aemeant33,Usrq33,'k-.');hold on
plot(xaplot,dyy33,'bs:');
axis([-7 7 -0.05 .65]); grid MINOR; set(gca,'FontSize',16)
legend('Fluent, U_S_R_Q_,_9_5_%','Wang & Hassan data','Location','NorthEast')
xlabel('x/a'); ylabel('axial velocity, m/s')
%%
figure; %ya1034
GCI10=[0.000006114772987
   0.000070730864536
   0.000159476709086
   0.000279910862446
   0.000440115225501
   0.000651487440336
   0.000923983985558
   0.001263924292289
   0.001674085739069
   0.002030907548033
   0.002459031413309
   0.000931501155719
   0.000033024390761
   0.000073169581572
   0.059693892570055
   0.049018991366719
   0.001541099702419
   0.052145988612242
   0.034427750428556
   0.000028828383275
   0.000135201310320
   0.007861659745686
   0.009693881962448
   0.009087696962524
   0.008663614222314
   0.008094846212771
   0.007572168367915
   0.007108512043487
   0.006701156438794
   0.006346770795062
   0.006037079903763
   0.005766480288003
   0.005528983601835]; Unum10=GCI10;
dx10=data_exp_y1034(1:23); dy10=data_exp_y1034(25:47);
dyy10 = pchip(dx10,dy10,xaplot); Umod10=ones(33,1);
for i =1:33
    if (aemeant10(i)+aepmt10(i))-dyy10(i) < 0 %|| (aemeant33(i)-aepmt33(i))-dyy(i)>0
        Umod10(i)=abs(aemeant10(i)-dyy10(i))-aepmt10(i);
    elseif (aemeant10(i)+aepmt10(i))-dyy10(i) > 0
        Umod10(i)=0;
```

```
    end
end
Ud10=dyy10*0.016;
% % errorbar(xaplot,aemeant10,aepmt10,'k-.'); hold on; set(gca,'FontSize',20)
t1=aemeant10;t2=aemeant10+Umod10;
tt=[.9, .1, 0.1;
    .9, .1, 0.1;
    .9, .1, 0.1];
Uall10=[aemeant10-aepmt10-Unum10-Umod10-
Ud10,Ud10,Umod10,Unum10,2*aepmt10,Unum10,Umod10,Ud10];
h=area(xaplot,Uall10,'LineStyle','none'); hold on
h(1).FaceColor = [1 1 1];
h(2).FaceColor = [0 0 1];
h(3).FaceColor = [0 1 0];
h(4).FaceColor = [1 0 0];
h(5).FaceColor = [0 0 0];
h(5).FaceColor = [0 0 0];
h(6).FaceColor = [1 0 0];
h(7).FaceColor = [0 1 0];
h(8).FaceColor = [0 0 1];

% % errorbar(xaplot,aemeant10,Umod10,'m-.'); hold on; set(gca,'FontSize',20)
% % errorbar(xaplot,aemeant10,Unum10,'r-.'); hold on; set(gca,'FontSize',16)
axis([-7 7 -0.05 .7]); xlabel('x/a'); ylabel('axial velocity, m/s')
% % errorbar(xaplot,dyy10,Ud10,'g-.');
plot(xaplot,dyy10,'bs:');
% title('Contributions to Total Uncertainty')
set(gca,'FontSize',20);


% legend('Fluent, U_i_n_p_u_t','Wang & Hassan data','Location','NorthEast')
% total uncertainty in SRQ Unum+Uinput+Ud+Umodel
legend('Uncertainty:','Data, U_D','Fluent, U_m_o_d_e_l','Fluent, U_n_u_m','Fluent,
U_i_n_p_u_t','Location','NorthEast')
figure;
Usrq10 = aepmt10+Unum10+Ud10+Umod10;
errorbar(xaplot,aemeant10,Usrq10,'k-.')
plot(xaplot,dyy10,'bs:');
legend('Fluent, U_S_R_Q_,_9_5_%','Wang & Hassan data','Location','NorthEast')
% figure(10)
% errorbar(xaplot,aemeant10,Usrq10,'k-.');hold on
% plot(xaplot,dyy10,'bs:');
% axis([-7 7 -0.05 .7]); grid MINOR; set(gca,'FontSize',20)
% legend('Fluent, U_S_R_Q_,_9_5_%','Wang & Hassan data','Location','NorthEast')
xlabel('x/a'); ylabel('axial velocity, m/s')
```

I.3 Write driver.sh

```
function [] = write_driver(FIF,FVX,FOF,DRIVER)
driver = DRIVER; fprintf('Writing DAKOTA driver: ''%s''..... ',driver); tic;
fif=FIF; fvx=FVX; fof=FOF;
% np = NP;% 4318;
fid=fopen('driver.sh','w');
fprintf(fid,'#!/bin/bash\n');
% # $1 and $2 are special variables in bash that contain the 1st and 2nd
% # command line arguments to the script, which are the names of the
% # Dakota parameters and results files, respectively.
% ##############################################################################
fprintf(fid,'params=$1\n');
fprintf(fid,'results=$2\n');
% ##############################################################################
% ##
% ## Pre-processing Phase -- Generate/configure an input file for your simulation
% ##  by substiting in parameter values from the Dakota paramters file.
% ##
% ##############################################################################
fprintf(fid,'dprepro $params %s fluent.in\n',fif);
fprintf(fid,'dprepro --left-delimiter=CFD --right-delimiter=CFD $params %s
probe_domain.fvx\n',fvx);
% ##############################################################################
% ##
% ## Execution Phase -- Run your simulation
% ##
% ##############################################################################
fprintf(fid,'scontrol show hostname $SLURM_NODELIST > /scratch/ssl29/hosts.txt\n');
fprintf(fid,'module load ansys\n');
%fprintf(fid,'srun fluent 3ddp -g < fluent.in > fluent.log\n');
fprintf(fid,'fluent 3ddp -t $SLURM_NPROCS -mpi=openmpi -mpirun=''mpirun'' -
cnf=/scratch/ssl29/hosts.txt -pib -g < fluent.in > %s\n',fof);
% ##############################################################################
% ##
% ## Post-processing Phase -- Extract (or calculate) quantities of interest
% ##  from your simulation's output and write them to a properly-formatted
% ##  Dakota results file.
% ##
% ##############################################################################
% #cat point-probe-yvel > results.out
% #grep -m 1 "\-0.011" line-test-values > results.out
fprintf(fid,'module load fieldview\n');
fprintf(fid,'srun fv -batch -software_render -fvx /home/ssl29/UQ/probe_domain.fvx\n');
fprintf(fid,'cat /home/ssl29/UQ/yvel.txt > $results\n');
fclose(fid);
fprintf('DONE! '); toc
```

I.4 Write Dakota.in

```
function [] = write_dakota_input_script(NG,SG,S,NP,NQ,CI,SRQ,DIF)
dif = DIF; fprintf('Writing DAKOTA input file: ''%s''..... ',dif);
tic; srq = SRQ;
% Define constants:
ng = NG; % number of grids
sg = SG; % number of starting grid
s=S; % number of samples
np = NP; % number of points
nq = NQ; % number of quantities at each point
rf=nq*np; %number of response functions
ci = CI; % confidence level
%-----Script - MAIN-----
fid=fopen('dakota_uq_matlab.in','w');
fprintf(fid,'environment\n');
fprintf(fid,'  method_pointer ''pstudy''\n');
fprintf(fid,'method\n');
fprintf(fid,'  id_method ''pstudy''\n');
fprintf(fid,'  vector_parameter_study\n');
fprintf(fid,'    step_vector 1\n');
fprintf(fid,'    num_steps %d \n',ng-sg);
fprintf(fid,'  model_pointer ''outer_m''\n');
fprintf(fid,'model\n');
fprintf(fid,'  id_model ''outer_m''\n');
fprintf(fid,'  nested\n');
fprintf(fid,'    sub_method_pointer ''sampling''\n');
%       # The primary variable mapping determines which inner model variables
%       # receive inner model variable values. The number of entries must
%       # equal the number of active inner model variables.
fprintf(fid,'    primary_variable_mapping ''grid_index''\n');
%       # The primary_response_mapping matrix has one column per result that
%       # is returned by the sampling method, and one row per outer_m response.
%       # The number of results that is returned by the method depend on the method
type,
%       # method configuration, and number of inner_m responses. This is described in
%       # full detail in the keyword entry for sub_method_pointer in the Reference
%       # Manual.
fprintf(fid,'    primary_response_mapping \n');
% vm=10; yv=10; xv=10; ke=10;
% rows = 1:1:4*vm;
col = 3:3:3*np*nq;%(4*3*np+4*3*np+4*3*np+4*3*np);
% count = 0;
z=zeros(1,3*np*nq);
for i = 1:np*nq%:np%2*(np+np+np+np)
    count=col(i);
    z(1,count) = 1;
    fprintf(fid,'%d ',z(1,:));
    fprintf(fid,'\n');
    z(1,count) = 0;
end
% for i = 1:np*nq%:8*np
%     fprintf(fid,'%d ',z(i,:));
%     fprintf(fid,'\n');
% end
% fprintf(fid,'%d ',z); fprintf(fid,'\n');
fprintf(fid,'  variables_pointer ''epist_var''\n');
fprintf(fid,'  responses_pointer ''epist_resp''\n');
fprintf(fid,'variables\n');
fprintf(fid,'  id_variables ''epist_var''\n');
fprintf(fid,'  active design\n');
fprintf(fid,'  discrete_design_range 1\n');
```

```
fprintf(fid,'    descriptors ''outer_grid_index''\n');
fprintf(fid,'    lower_bounds %d\n',sg);
fprintf(fid,'    upper_bounds %d\n',ng);
fprintf(fid,'    initial_point %d\n',sg);
fprintf(fid,'responses\n');
fprintf(fid,'  id_responses ''epist_resp''\n');
fprintf(fid,'  response_functions %d \n',rf);
fprintf(fid,'  descriptors ');
for i = 1:np
    fprintf(fid,'''%s%d''\n',srq,i);% ''yv%d'' ''xv%d'' ''ke%d''\n',i,i,i,i);
end
fprintf(fid,'  no_gradients\n');
fprintf(fid,'  no_hessians\n');
fprintf(fid,'method\n');
fprintf(fid,'  id_method ''sampling''\n');
fprintf(fid,'  sampling seed 1234 samples %d\n',s);
fprintf(fid,'  model_pointer ''inner_m''\n');
fprintf(fid,'  probability_levels ');
for i = 1:np
    fprintf(fid,'%d\n',ci);%0.95 0.95 0.95\n');
end
fprintf(fid,'  output verbose\n');
fprintf(fid,'model\n');
fprintf(fid,'  id_model ''inner_m''\n');
fprintf(fid,'  single\n');
fprintf(fid,'  variables_pointer ''aleat_var''\n');
fprintf(fid,'  responses_pointer ''aleat_resp''\n');
fprintf(fid,'variables\n');
fprintf(fid,'  id_variables ''aleat_var''\n');
fprintf(fid,'  normal_uncertain 6\n');
fprintf(fid,'    descriptors    ''m1''  ''m2'' ''I1'' ''I2'' ''L1''    ''L2''\n');
fprintf(fid,'    means          0.385 0.385 8   8    0.0058  0.0058\n');
fprintf(fid,'    std_deviations 0.011 0.011 2.5 2.5  0.00116 0.00116\n');
fprintf(fid,'  discrete_state_range 1\n');
fprintf(fid,'    descriptors ''grid_index''\n');
fprintf(fid,'    lower_bounds %d\n',sg);
fprintf(fid,'    upper_bounds %d\n',ng);
fprintf(fid,'responses\n');
fprintf(fid,'  id_responses ''aleat_resp''\n');
fprintf(fid,'  response_functions %d\n',rf);
fprintf(fid,'  descriptors ');
for i = 1:np
    fprintf(fid,'''%s%d''\n',srq,i);% ''yv%d'' ''xv%d'' ''ke%d''\n',i,i,i,i);
end
fprintf(fid,'  no_gradients\n');
fprintf(fid,'  no_hessians\n');
fprintf(fid,'interface\n');
fprintf(fid,'  system\n');
fprintf(fid,'    analysis_drivers = ''driver.sh''\n');
fprintf(fid,'    parameters_file  = ''params.in''\n');
fprintf(fid,'    results_file     = ''results.out''\n');
fclose(fid);
fprintf('DONE! '); toc
```

I.5 Write FieldView input

```
function [] =
write_fieldview_script(FVX,XPTS,YPTS,ZPTS,SCALAR_FUNC,SCALAR_OUT,FCF,LOC_FCF,FDF,LOC_F
DF)
fcf=FCF; loc_fcf=LOC_FCF;
xpts=XPTS; ypts=YPTS; zpts=ZPTS; scalar_func=SCALAR_FUNC; scalar_out=SCALAR_OUT;
fdf=FDF; loc_fdf=LOC_FDF;
fvx = FVX; fprintf('Writing FIELDVIEW input file: ''%s''..... ',fvx); tic;
% ------------------------------------------------------------------------
x=dlmread(xpts); x=x(:,1);
y=dlmread(ypts); y=y(:,1);
z=dlmread(zpts);
%-------------------------------------------------------------------------
%--                          DATA INPUT
%-------------------------------------------------------------------------
fid = fopen(fvx,'w');
% %case and data
fprintf(fid,'local datasets_info_table = {}\n');
fprintf(fid,'datasets_info_table[1] = read_dataset( {\n');
fprintf(fid,'    data_format = "fluent_cas/dat_direct",\n');
fprintf(fid,'    input_parameters = {\n');
fprintf(fid,'        grid_file = {\n');
fprintf(fid,'            name = %s%sCFDgrid_index=1CFD.msh",\n',loc_fcf,fcf);
fprintf(fid,'            name = %s%sCFDgrid_index=1CFD.msh",\n',loc_fmf,fmf);
fprintf(fid,'            name = %s%s,\n',loc_fmf,fmf);
fprintf(fid,'            name = %s%s,\n',loc_fcf,fcf);
fprintf(fid,'            options = {\n');
fprintf(fid,'                input_mode = "replace",\n');
fprintf(fid,'                grid_processing = "balanced",\n');
fprintf(fid,'                boundary_only = "off"\n');
fprintf(fid,'            }\n'); % -- options
fprintf(fid,'        },\n'); % -- grid_file
fprintf(fid,'        results_file = {\n');
fprintf(fid,'            name = %s%s,\n',loc_fdf,fdf);
fprintf(fid,'            options = {\n');
fprintf(fid,'                input_mode = "replace"\n');
fprintf(fid,'            }\n'); % -- options
fprintf(fid,'        }\n'); % -- results_file
fprintf(fid,'    }\n'); % -- input_parameters
fprintf(fid,'} )\n'); % -- read_dataset
%-------------------------------------------------------------------------
%--                           COORD SURFACE
%-------------------------------------------------------------------------
fprintf(fid,'local coord_table ={}\n');
fprintf(fid,'coord_table[1] = create_coord(\n');
fprintf(fid,' {\n');
fprintf(fid,'   scalar_func = "%s",\n',scalar_func); %--scalar_func: name or "none"
fprintf(fid,'   vector_func = "none",\n'); %--vector_func: name or "none"
fprintf(fid,'   threshold_func = "none",\n'); %--threshold_func: name or "none"
fprintf(fid,'   visibility = "off",\n'); %--visibility: either "on" or "off"
fprintf(fid,'   axis = "Z",\n'); %--axis: "X", "Y", "R", "T", or "Z"\n');
fprintf(fid,'   Z_axis = {\n'); %--X_axis: Use only if axis = "X"
fprintf(fid,'     min = -0.1,\n');
fprintf(fid,'     current = 0.01358,\n');
fprintf(fid,'     max = 0.1,\n');
fprintf(fid,'    },\n');
fprintf(fid,'    scalar_range = {\n');
fprintf(fid,'            abs_max = 0.719171404838562,\n');
fprintf(fid,'            local_min = -0.006245823577046394,\n');
fprintf(fid,'            min = -0.03261079639196396,\n');
fprintf(fid,'            abs_min = -0.03261079639196396,\n');
fprintf(fid,'            max = 0.719171404838562,\n');
```

```
fprintf(fid,'            local_max = 0.7188896536827087,\n');
fprintf(fid,'            use_local = "off",\n');
fprintf(fid,'      }, -- scalar_range\n');
fprintf(fid,'   }\n');
fprintf(fid,')\n');
%-------------------------------------------------------------------------
%--                           POINT PROBE
%-------------------------------------------------------------------------
fprintf(fid,'local scalar={}\n');
fprintf(fid,'fid = openfile("%s", "w")\n',scalar_out);
count = 0;
% for j=2:length(y)
    for i=1:length(x)
        count=count+1;
%         for k=1:length(z)
          fprintf(fid,'point = {%f, %f, %f}\n',x(i),y(i),z(1));
          fprintf(fid,'probe_out = probe_current_functions(point)\n');
          fprintf(fid,'dumpall(probe_out)\n');
          fprintf(fid,'scalar[%d] = probe_out.scalar.value\n',count);
          fprintf(fid,'write(fid, scalar[%d])\n',count);
          fprintf(fid,'write(fid,"\\n")\n');
%         end
    end
% end
fprintf(fid,'closefile(fid)\n');
%-- print_dataset_table( datasets_info_table[1] )
fclose(fid);
fprintf('DONE! '); toc
% -------------------------------------------------------------------------
```

I.6 Write Fluent Input

```
function [] =
write_fieldview_script(FVX,XPTS,YPTS,ZPTS,SCALAR_FUNC,SCALAR_OUT,FDF,LOC_FDF,FMF,LOC_F
MF)
fmf=FMF; loc_fmf=LOC_FMF;
fprintf('Writing FLUENT input file: ''%s''..... ',fif); tic;
%---FLUENT input file for study of the ASME turbulent twinjet benchmark case study at
NAU-----
% Define constants:
ites = ITES; % number of iterations
rho = 997.5; % density of water
visc=0.000937; % viscosity of water
ri = 10; % reporting interval
fid=fopen('xpoints.txt','r');
C=textscan(fid,'%n%s');
fclose(fid);
x=C{1};
fid=fopen('ypoints.txt','r');
C=textscan(fid,'%n%s');
fclose(fid);
y=C{1};
%---AUTHOR: Seth Lawrence, ME-----------------------------------------------------------
--------
fid=fopen('fluent.template','w');
%-----------IMPORT Fluent case file (.cas)----------
fprintf(fid,'file/read-case /scratch/ssl29/3D_quad.cas\n');
fprintf(fid,'file/confirm-overwrite no\n');
%----------IMPORT mesh file (.msh)----------
fprintf(fid,'/file/replace-mesh /scratch/ssl29/mesh/3D_quad_4c{grid_index=1}.msh\n');
%----------DEFINE models----------
fprintf(fid,'/define/models/viscous/ke-realizable yes\n');
fprintf(fid,'/define/models/viscous/near-wall-treatment/enhanced-wall-treatment
yes\n');
%------------ DEFINE materials------------
fprintf(fid,'/define/materials/copy fluid water-liquid\n');
fprintf(fid,'/define/boundary-conditions/fluid fluid-1 yes water-liquid no no no no 0
no 0 no 0 0 no 0 no 0 no 0 no 1 no no no no\n');
fprintf(fid,'/define/materials/delete air\n');
fprintf(fid,'/define/materials/change-create water-liquid water yes constant %f no no
yes constant %f no no no yes\n',rho,visc);
%----------DEFINE boundary conditions---------
fprintf(fid,'/define/operating-conditions/operating-pressure 0\n');
fprintf(fid,'/define/boundary-conditions/modify-zones/zone-type/inlet_1 mass-flow-
inlet\n');
fprintf(fid,'/define/boundary-conditions/modify-zones/zone-type/inlet_2 mass-flow-
inlet\n');
fprintf(fid,'/define/boundary-conditions/mass-flow-inlet inlet_1 yes yes no {m1=0.385}
no 0  no yes no yes {I1=8} {L1=0.0058}\n');
fprintf(fid,'/define/boundary-conditions/mass-flow-inlet inlet_2 yes yes no {m2=0.385}
no 0  no yes no yes {I2=8} {L2=0.0058}\n');
%----------SOLVE Settings----------
fprintf(fid,'/solve/set/reporting %d\n',ri);
fprintf(fid,'/solve/set/flow-warnings no\n');
fprintf(fid,'/solve/set limiter-warnings no\n');
fprintf(fid,'/solve/set/p-v-coupling 24\n');
fprintf(fid,'/solve/set/discretization-scheme/pressure 12\n');
fprintf(fid,'/solve/set/discretization-scheme/mom 1\n');
fprintf(fid,'/solve/set/discretization-scheme/k 1\n');
fprintf(fid,'/solve/set/discretization-scheme/epsilon 1\n');
fprintf(fid,'/solve/monitors/residual/check-convergence no no no no no no no no\n');
%---initialize flow
fprintf(fid,'/solve/initialize/hyb-init\n');
```

```
%----------RUN calculation----------
fprintf(fid,'/solve/iterate %d\n',ites);
%-----------------------------------WRITE OUTPUTS----------------------------
% /report/summary no
%---WRITE fluent data output file (.dat)
fprintf(fid,'wd /scratch/ssl29/UQ/3D_quad_uq.dat\n');
%---QUIT FLUENT---
fprintf(fid,'exit\n');
fprintf(fid,'OK');
fclose(fid);
fprintf('DONE! '); toc
```

I.7 Write Slurm Job

```
function [] = write_slurm_jobscript(SJF,JN,SLF,NCPU,T,WKDR,MEM,CON,DIF,DOF)
sjf=SJF; jn=JN; slf=SLF; ncpu=NCPU; t=T; wkdr=WKDR; mem=MEM; con=CON;
dif=DIF; dof=DOF;
fprintf('Writing SLURM job file for Monsoon: ''%s''..... ',sjf); tic;
fid=fopen(sjf,'w');
fprintf(fid,'#!/bin/bash\n');
fprintf(fid,'#SBATCH --job-name=%s\n',jn);
fprintf(fid,'#SBATCH --output=%s%s\n',wkdr,slf);
fprintf(fid,'#SBATCH --ntasks=%d\n',ncpu);
fprintf(fid,'#SBATCH --time=%s\n',t);
fprintf(fid,'#SBATCH --workdir=%s\n',wkdr);
n = ceil(ncpu/24);
fprintf(fid,'#SBATCH --nodes=%d\n',n);
fprintf(fid,'#SBATCH --mem=%d\n',mem);
fprintf(fid,'#SBATCH --constraint=%s\n',con);
fprintf(fid,'chmod +x driver.sh\n');
fprintf(fid,'echo "Starting at `date`"\n');
fprintf(fid,'module load dakota\n'); %load relevant modules
fprintf(fid,'dakota -i %s -o %s\n',dif,dof); %run dakota
fprintf(fid,'echo "Ending at `date`"\n');
fclose(fid);
fprintf('DONE! '); toc
```

I.8 Write All Scripts

```
clc; clear; close all
% ------------------------------------------------------------------------
% --                     FLUENT INPUTS
% ------------------------------------------------------------------------
ites = 100;                     % number of iterations Fluent will run
fif = 'fluent.template';        % specify fluent input file (fif)
flf = 'fluent.log';             % specify fluent log file (flf)
fcf = '3D_quad_uq.cas';         % specify fluent case file (fcf)
loc_fdf = '''/home/ssl29/UQ/';
loc_fmf = '''/scratch/ssl29/UQ/';
% ------------------------------------------------------------------------
% --                     DRIVER.SH
% ------------------------------------------------------------------------
driver = 'driver.sh';           % specify dakota driver
% ------------------------------------------------------------------------
% --                     FIELDVIEW INPUTS
% ------------------------------------------------------------------------
fvx = 'probe_domain_airfoil.fvx'; % specify fieldview script file (fvx)
xpts = '/home/ssl29/xpoints.txt'; % define x-coords of points to probe
ypts = '/home/ssl29/ypoints.txt'; % define y-coords of points to probe
zpts = '/home/ssl29/zpoints.txt';  % define z-coords of points to probe
fmf = '3D_quad_4c'; % mesh file
scalar_func = 'Y Velocity';     % define scalar to probe at points
scalar_out = 'yvel.txt';      % filename to write scalar value to
% ------------------------------------------------------------------------
% --                    DAKOTA INPUTS
% ------------------------------------------------------------------------
dif = 'dakota_study.in';  % specify dakota input file (dif)
dof = 'dakota_study.out';            % file to store dakota outputs (dof)
np = 4318;  % number of points/locations in Dakota study
nq = 1;     % number of quantities at ech point in Dakota study
ng = 9;     % number of grids for epistemic uncertainty in Dakota study
sg = 1;     % initial grid used to at start of Dakota study
s=10;       % number of samples for aleatory uncertainty in Dakota study
ci = 0.95;  % confidence level in Dakota study e.g. 95%
srq = 'y_vel';   % short name for response quantity in Dakota study
% ------------------------------------------------------------------------
% --                     SLURM JOBSCRIPT INPUTS
% ------------------------------------------------------------------------
sjf = 'run_dakota_study.sh';     % specify slurm job file for monsoon (sjf)
jn = 'DAKOTA';                   % name of batch job
slf = 'dakota_study.log';        % slurm job log file (slf)
ncpu = 120;                      % number of CPUs for job (max = 120)
t = '00-02:00:00';               % how long job will run (D-H:M:S)
wkdr = '/home/ssl29/demo/';       % working directory for job
mem = 96000;                     % Memory for job (1GB = 1000)
con = 'hw';                      % constraints (hw=haswell which are fast)
% ------------------------------------------------------------------------
% --                 WRITE SCRIPTS
% ------------------------------------------------------------------------
write_fluent_input_script(ites,fif);
write_driver(fif,fvx,flf,driver);
write_fieldview_script(fvx,xpts,ypts,zpts,scalar_func,scalar_out,fdf,loc_fdf,fmf,loc_f
mf);
write_dakota_input_script(ng,sg,s,np,nq,ci,srq,dif);
write_slurm_jobscript(sjf,jn,slf,ncpu,t,wkdr,mem,con,dif,dof);
fprintf('Following files have been successfully written:\n')
fprintf('---> %s\n---> %s\n---> %s\n---> %s\n',fif,driver,fvx,dif);
fprintf('---> %s\n',sjf);
% ------------------------------------------------------------------------
```

Appendix J Input Files

The files that are automatically generated by the Matlab scripts that were presented in the previous Appendix are provided as a reference. Each file was used to bridge the workflow on the Monsoon cluster and run the full uncertainty analysis using Dakota.

J.1 Slurm Job File

```
#!/bin/bash
#SBATCH --job-name=DAKOTA
#SBATCH --output=/home/ssl29/UQ/dakota.log
#SBATCH --ntasks=120
#SBATCH --time=2:00:00
#SBATCH --workdir=/home/ssl29/UQ
#SBATCH --nodes=5
#SBATCH --mem=96000
#SBATCH --constraint=hw
echo "Starting at `date`"
# scontrol show hostname $SLURM_NODELIST > /scratch/ssl29/hosts.txt
# load relevant modules
module load dakota
# run dakota
dakota -i dakota_uq_matlab.in -o dakota.out
echo "Ending at `date`"
```

J.2 Dakota Driver File

```
#!/bin/bash
params=$1
results=$2
dprepro $params fluent.template fluent.in
dprepro --left-delimiter=CFD --right-delimiter=CFD $params probe_domain.template
probe_domain.fvx
scontrol show hostname $SLURM_NODELIST > /scratch/ssl29/hosts.txt
module load ansys
fluent 3ddp -t $SLURM_NPROCS -mpi=openmpi -mpirun='mpirun' -
cnf=/scratch/ssl29/hosts.txt -pib -g < fluent.in > fluent.log
module load fieldview
srun fv -batch -software_render -fvx /home/ssl29/UQ/probe_domain.fvx
cat /home/ssl29/UQ/yvel.txt > $results
```

J.3 Dakota Input File

```
environment
  method_pointer 'pstudy'
method
  id_method 'pstudy'
  vector_parameter_study
    step_vector 1
    num_steps 8
  model_pointer 'outer_m'
model
  id_model 'outer_m'
  nested
    sub_method_pointer 'sampling'
    primary_variable_mapping 'grid_index'
    primary_response_mapping
0 0 1
  variables_pointer 'epist_var'
  responses_pointer 'epist_resp'
variables
  id_variables 'epist_var'
  active design
  discrete_design_range 1
    descriptors 'outer_grid_index'
    lower_bounds 1
    upper_bounds 9
    initial_point 1
responses
  id_responses 'epist_resp'
  response_functions 1
  descriptors 'y_vel1'
  no_gradients
  no_hessians
method
  id_method 'sampling'
  sampling seed 1234 samples 10
  model_pointer 'inner_m'
  probability_levels 9.500000e-01
  output verbose
model
  id_model 'inner_m'
  single
  variables_pointer 'aleat_var'
  responses_pointer 'aleat_resp'
variables
  id_variables 'aleat_var'
  normal_uncertain 6
    descriptors    'm1'  'm2' 'I1' 'I2' 'L1'    'L2'
    means          0.385 0.385 8   8    0.0058  0.0058
    std_deviations 0.011 0.011 2.5 2.5  0.00116 0.00116
  discrete_state_range 1
    descriptors 'grid_index'
    lower_bounds 1
    upper_bounds 9
responses
```

```
  id_responses 'aleat_resp'
  response_functions 1
  descriptors 'y_vel1'
  no_gradients
  no_hessians
interface
  system
    analysis_drivers = 'driver.sh'
    parameters_file  = 'params.in'
    results_file     = 'results.out'
```

J.4 Fluent Input File

```
file/read-case /scratch/ssl29/3D_quad.cas
file/confirm-overwrite no
/file/replace-mesh /scratch/ssl29/mesh/3D_quad_4c{grid_index=1}.msh
/define/models/viscous/ke-realizable yes
/define/models/viscous/near-wall-treatment/enhanced-wall-treatment yes
/define/materials/copy fluid water-liquid
/define/boundary-conditions/fluid fluid-1 yes water-liquid no no no no 0 no 0 no 0 0
no 0 no 0 no 0 no 1 no no no no
/define/materials/delete air
/define/materials/change-create water-liquid water yes constant 997.500000 no no yes
constant 0.000937 no no no yes
/define/operating-conditions/operating-pressure 0
/define/boundary-conditions/modify-zones/zone-type/inlet_1 mass-flow-inlet
/define/boundary-conditions/modify-zones/zone-type/inlet_2 mass-flow-inlet
/define/boundary-conditions/mass-flow-inlet inlet_1 yes yes no {m1=0.385} no 0  no yes
no yes {I1=8} {L1=0.0058}
/define/boundary-conditions/mass-flow-inlet inlet_2 yes yes no {m2=0.385} no 0  no yes
no yes {I2=8} {L2=0.0058}
/solve/set/reporting 10
/solve/set/flow-warnings no
/solve/set limiter-warnings no
/solve/set/p-v-coupling 24
/solve/set/discretization-scheme/pressure 12
/solve/set/discretization-scheme/mom 1
/solve/set/discretization-scheme/k 1
/solve/set/discretization-scheme/epsilon 1
/solve/monitors/residual/check-convergence no no no no no no no no
/solve/initialize/hyb-init
/solve/iterate 100
wd /scratch/ssl29/UQ/3D_quad_uq.dat
exit
OK
```

J.5 FieldView FVX File

```
local datasets_info_table = {}
datasets_info_table[1] = read_dataset( {
    data_format = "fluent_cas/dat_direct",
    input_parameters = {
        grid_file = {
            name = '/scratch/ssl29/UQ/3D_quad_4cCFDgrid_index=1CFD.msh",
            options = {
                input_mode = "replace",
                grid_processing = "balanced",
                boundary_only = "off"
            }
        },
        results_file = {
            name = /home/ssl29/UQ/3D_quad_uq.dat,
            options = {
                input_mode = "replace"
            }
        }
    }
} )
local coord_table ={}
coord_table[1] = create_coord(
 {
   scalar_func = "Y Velocity",
   vector_func = "none",
   threshold_func = "none",
   visibility = "off",
   axis = "Z",
   Z_axis = {
     min = -0.1,
     current = 0.01358,
     max = 0.1,
    },
    scalar_range = {
            abs_max = 0.719171404838562,
            local_min = -0.006245823577046394,
            min = -0.03261079639196396,
            abs_min = -0.03261079639196396,
            max = 0.719171404838562,
            local_max = 0.7188896536827087,
            use_local = "off",
      }, -- scalar_range
   }
)
local scalar={}
fid = openfile("yvel.txt", "w")
point = {1.000000, 1.000000, 1.000000}
probe_out = probe_current_functions(point)
dumpall(probe_out)
scalar[1] = probe_out.scalar.value
write(fid, scalar[1])
write(fid,"\n")
closefile(fid)
```

Appendix K Experimental Data.

K.1 Mean Axial Velocity at y/a = 0

U is set to be 0 at the wall (x/a=-2.03 and x/a=2.03) based on the non-slip assumption

y/a= 0, PIV, 15Hz, time averaged

x/a	y/a	U	Urms
-2.03	0	0	N/A
-1.98276	0	0.649	0.158
-1.92845	0	0.733	0.104
-1.87415	0	0.786	0.069
-1.81985	0	0.814	0.056
-1.76554	0	0.832	0.055
-1.71124	0	0.847	0.049
-1.65694	0	0.857	0.041
-1.60263	0	0.861	0.040
-1.54833	0	0.861	0.040
-1.49403	0	0.855	0.044
-1.43972	0	0.845	0.047
-1.38542	0	0.834	0.053
-1.33111	0	0.815	0.057
-1.27681	0	0.785	0.070
-1.22251	0	0.748	0.079
-1.1682	0	0.679	0.114
-1.1139	0	0.431	0.173
-1.0596	0	0.127	0.160
-1.00529	0	0.018	0.050
-0.95099	0	-0.023	0.022
-0.89668	0	-0.001	0.013
-0.84238	0	0.000	0.009
-0.78808	0	-0.003	0.008
-0.73377	0	-0.004	0.007
-0.67947	0	-0.003	0.007
-0.62517	0	-0.002	0.006
-0.57086	0	-0.001	0.005
-0.51656	0	-0.001	0.005
-0.46225	0	-0.001	0.004
-0.40795	0	-0.001	0.003
-0.35365	0	0.000	0.003
-0.29934	0	-0.001	0.004
-0.24504	0	-0.001	0.004
-0.19074	0	0.000	0.004

-0.13643	0	-0.001	0.004
-0.08213	0	-0.001	0.004
-0.02783	0	-0.002	0.004
0.026478	0	-0.002	0.003
0.080782	0	-0.001	0.003
0.135086	0	-0.001	0.004
0.189389	0	-0.001	0.004
0.243693	0	-0.001	0.003
0.297997	0	-0.002	0.004
0.352301	0	-0.002	0.003
0.406604	0	-0.006	0.003
0.460908	0	-0.009	0.004
0.515212	0	-0.006	0.004
0.569515	0	-0.003	0.003
0.623819	0	-0.004	0.003
0.678123	0	-0.004	0.003
0.732426	0	-0.004	0.003
0.78673	0	-0.005	0.003
1.003945	0	0.002	0.032
1.058249	0	0.412	0.185
1.112552	0	0.600	0.204
1.166856	0	0.736	0.124
1.22116	0	0.804	0.083
1.275464	0	0.808	0.063
1.329767	0	0.829	0.051
1.384071	0	0.845	0.048
1.438375	0	0.852	0.043
1.492678	0	0.853	0.044
1.546982	0	0.852	0.042
1.601286	0	0.844	0.045
1.655589	0	0.832	0.054
1.709893	0	0.814	0.061
1.764197	0	0.789	0.073
1.818501	0	0.758	0.082
1.872804	0	0.714	0.106
1.927108	0	0.632	0.134
1.981412	0	0.421	0.166
2.03	0	0	N/A

K.2 Mean PIV Values at y/a = 1.41

Measured by PIV at 15 Hz, time averaged

x/a	y/a	U	V	Urms	Vrms	Reynolds stress
-1.98	1.41	0.550	0.010	0.171	0.115	-0.00569
-1.93	1.41	0.650	0.013	0.145	0.101	-0.00267
-1.87	1.41	0.717	0.019	0.122	0.084	-0.00245
-1.82	1.41	0.763	0.029	0.101	0.067	-0.00181
-1.77	1.41	0.797	0.037	0.077	0.054	-0.00104
-1.71	1.41	0.818	0.041	0.066	0.042	-0.00066
-1.66	1.41	0.833	0.044	0.057	0.037	-0.00039
-1.60	1.41	0.845	0.046	0.052	0.033	-0.00030
-1.55	1.41	0.850	0.047	0.049	0.030	-0.00018
-1.49	1.41	0.849	0.046	0.054	0.030	-0.00005
-1.44	1.41	0.843	0.045	0.053	0.032	0.00013
-1.39	1.41	0.831	0.045	0.058	0.037	0.00021
-1.33	1.41	0.814	0.046	0.067	0.045	0.00025
-1.28	1.41	0.788	0.045	0.080	0.058	0.00113
-1.22	1.41	0.749	0.049	0.093	0.070	0.00169
-1.17	1.41	0.690	0.053	0.112	0.086	0.00302
-1.11	1.41	0.601	0.049	0.134	0.105	0.00515
-1.06	1.41	0.493	0.043	0.153	0.121	0.00581
-1.01	1.41	0.389	0.028	0.160	0.129	0.00628
-0.95	1.41	0.286	0.012	0.173	0.121	0.00473
-0.90	1.41	0.192	-0.009	0.161	0.107	0.00287
-0.84	1.41	0.109	-0.023	0.125	0.092	0.00245
-0.79	1.41	0.051	-0.026	0.102	0.076	0.00201
-0.73	1.41	0.015	-0.031	0.076	0.060	0.00050
-0.68	1.41	-0.003	-0.033	0.061	0.050	0.00024
-0.63	1.41	-0.013	-0.033	0.056	0.042	0.00033
-0.57	1.41	-0.019	-0.031	0.049	0.037	0.00039
-0.52	1.41	-0.024	-0.029	0.045	0.033	0.00032
-0.46	1.41	-0.025	-0.027	0.043	0.030	0.00028
-0.41	1.41	-0.029	-0.023	0.043	0.028	0.00028
-0.35	1.41	-0.031	-0.020	0.044	0.026	0.00026
-0.30	1.41	-0.032	-0.016	0.043	0.026	0.00021
-0.25	1.41	-0.033	-0.013	0.042	0.027	0.00018
-0.19	1.41	-0.035	-0.010	0.042	0.024	0.00016
-0.14	1.41	-0.036	-0.007	0.041	0.024	0.00012
-0.08	1.41	-0.037	-0.003	0.041	0.024	0.00009
-0.03	1.41	-0.037	0.000	0.041	0.024	0.00004

0.03	1.41	-0.038	0.003	0.041	0.024	0.00000
0.08	1.41	-0.039	0.006	0.042	0.024	-0.00006
0.14	1.41	-0.037	0.009	0.041	0.024	-0.00010
0.19	1.41	-0.035	0.013	0.040	0.024	-0.00013
0.24	1.41	-0.034	0.016	0.041	0.025	-0.00013
0.30	1.41	-0.031	0.018	0.042	0.025	-0.00017
0.35	1.41	-0.030	0.022	0.044	0.027	-0.00023
0.41	1.41	-0.028	0.025	0.045	0.029	-0.00033
0.46	1.41	-0.026	0.028	0.045	0.030	-0.00033
0.52	1.41	-0.022	0.029	0.047	0.034	-0.00045
0.57	1.41	-0.016	0.032	0.052	0.038	-0.00046
0.62	1.41	-0.007	0.034	0.052	0.043	-0.00042
0.68	1.41	0.015	0.029	0.064	0.057	-0.00108
0.73	1.41	0.051	0.028	0.077	0.068	-0.00149
0.79	1.41	0.098	0.019	0.100	0.080	-0.00248
0.84	1.41	0.176	0.010	0.122	0.093	-0.00417
0.90	1.41	0.268	-0.003	0.144	0.107	-0.00672
0.95	1.41	0.380	-0.017	0.168	0.117	-0.00759
1.00	1.41	0.495	-0.031	0.166	0.106	-0.00804
1.06	1.41	0.623	-0.043	0.156	0.091	-0.00557
1.11	1.41	0.715	-0.044	0.104	0.066	-0.00258
1.17	1.41	0.761	-0.042	0.088	0.053	-0.00209
1.22	1.41	0.793	-0.044	0.067	0.040	-0.00061
1.28	1.41	0.813	-0.043	0.054	0.033	-0.00054
1.33	1.41	0.825	-0.044	0.049	0.030	-0.00025
1.38	1.41	0.834	-0.044	0.044	0.028	-0.00014
1.44	1.41	0.839	-0.044	0.044	0.028	-0.00002
1.49	1.41	0.838	-0.043	0.044	0.029	0.00009
1.55	1.41	0.832	-0.042	0.048	0.030	0.00025
1.60	1.41	0.820	-0.041	0.053	0.034	0.00039
1.66	1.41	0.805	-0.037	0.059	0.035	0.00052
1.71	1.41	0.780	-0.036	0.072	0.044	0.00116
1.76	1.41	0.751	-0.028	0.084	0.050	0.00130
1.82	1.41	0.710	-0.020	0.108	0.067	0.00260
1.87	1.41	0.636	-0.008	0.138	0.082	0.00471
1.93	1.41	0.533	-0.007	0.158	0.097	0.00452
1.98	1.41	0.432	-0.006	0.165	0.105	0.00596

K.3 Mean PIV Values at y/a = 1.74

Measured by PIV at 15 Hz, time averaged

x/a	y/a	U	V	Urms	Vrms	Reynolds stress
-1.98	1.74	0.506	0.010	0.189	0.124	-0.00466
-1.93	1.74	0.592	0.018	0.163	0.106	-0.00536
-1.87	1.74	0.669	0.025	0.145	0.094	-0.00448
-1.82	1.74	0.731	0.031	0.115	0.078	-0.00333
-1.77	1.74	0.780	0.036	0.092	0.065	-0.00205
-1.71	1.74	0.806	0.042	0.079	0.054	-0.00127
-1.66	1.74	0.827	0.045	0.063	0.044	-0.00063
-1.60	1.74	0.838	0.048	0.058	0.039	-0.00044
-1.55	1.74	0.846	0.049	0.053	0.035	-0.00016
-1.49	1.74	0.846	0.051	0.053	0.035	-0.00003
-1.44	1.74	0.842	0.052	0.057	0.036	0.00024
-1.39	1.74	0.831	0.052	0.061	0.041	0.00044
-1.33	1.74	0.816	0.056	0.068	0.045	0.00069
-1.28	1.74	0.791	0.056	0.083	0.058	0.00151
-1.22	1.74	0.754	0.061	0.096	0.073	0.00236
-1.17	1.74	0.697	0.065	0.118	0.092	0.00403
-1.11	1.74	0.622	0.066	0.134	0.101	0.00503
-1.06	1.74	0.537	0.059	0.149	0.117	0.00566
-1.01	1.74	0.434	0.043	0.160	0.124	0.00629
-0.95	1.74	0.336	0.023	0.163	0.119	0.00559
-0.90	1.74	0.250	0.006	0.170	0.117	0.00518
-0.84	1.74	0.179	-0.001	0.153	0.111	0.00350
-0.79	1.74	0.116	-0.012	0.131	0.099	0.00377
-0.73	1.74	0.066	-0.025	0.115	0.082	0.00257
-0.68	1.74	0.033	-0.030	0.093	0.076	0.00099
-0.63	1.74	0.006	-0.032	0.078	0.065	0.00050
-0.57	1.74	-0.011	-0.034	0.062	0.051	0.00058
-0.52	1.74	-0.024	-0.033	0.047	0.045	0.00032
-0.46	1.74	-0.030	-0.031	0.045	0.038	0.00025
-0.41	1.74	-0.036	-0.028	0.039	0.036	0.00016
-0.35	1.74	-0.040	-0.026	0.039	0.033	0.00012
-0.30	1.74	-0.042	-0.022	0.036	0.029	0.00013
-0.25	1.74	-0.042	-0.018	0.036	0.028	0.00011
-0.19	1.74	-0.045	-0.014	0.035	0.027	0.00008
-0.14	1.74	-0.046	-0.010	0.034	0.027	0.00007
-0.08	1.74	-0.046	-0.005	0.034	0.027	0.00004
-0.03	1.74	-0.047	-0.001	0.035	0.027	0.00002
0.03	1.74	-0.047	0.003	0.036	0.027	0.00001
0.08	1.74	-0.046	0.007	0.036	0.026	-0.00005

0.14	1.74	-0.046	0.012	0.035	0.026	-0.00005
0.19	1.74	-0.046	0.015	0.035	0.027	-0.00003
0.24	1.74	-0.044	0.020	0.036	0.027	-0.00004
0.30	1.74	-0.041	0.024	0.038	0.028	-0.00008
0.35	1.74	-0.038	0.028	0.039	0.031	-0.00006
0.41	1.74	-0.035	0.030	0.042	0.034	-0.00012
0.46	1.74	-0.031	0.032	0.044	0.039	-0.00012
0.52	1.74	-0.022	0.031	0.049	0.045	-0.00027
0.57	1.74	-0.008	0.031	0.058	0.050	-0.00042
0.62	1.74	0.017	0.025	0.069	0.063	-0.00135
0.68	1.74	0.055	0.021	0.090	0.075	-0.00176
0.73	1.74	0.108	0.013	0.110	0.087	-0.00292
0.79	1.74	0.171	0.007	0.128	0.099	-0.00370
0.84	1.74	0.245	0.000	0.142	0.107	-0.00471
0.90	1.74	0.334	-0.013	0.167	0.117	-0.00678
0.95	1.74	0.432	-0.026	0.173	0.116	-0.00718
1.00	1.74	0.536	-0.040	0.167	0.110	-0.00850
1.06	1.74	0.637	-0.045	0.141	0.094	-0.00485
1.11	1.74	0.715	-0.051	0.120	0.073	-0.00346
1.17	1.74	0.765	-0.050	0.087	0.060	-0.00195
1.22	1.74	0.799	-0.050	0.066	0.044	-0.00094
1.28	1.74	0.816	-0.050	0.059	0.040	-0.00067
1.33	1.74	0.829	-0.050	0.049	0.037	-0.00035
1.38	1.74	0.835	-0.050	0.046	0.033	-0.00015
1.44	1.74	0.837	-0.050	0.046	0.033	0.00001
1.49	1.74	0.834	-0.049	0.048	0.033	0.00016
1.55	1.74	0.826	-0.047	0.051	0.034	0.00035
1.60	1.74	0.813	-0.045	0.058	0.037	0.00047
1.66	1.74	0.794	-0.042	0.064	0.044	0.00070
1.71	1.74	0.767	-0.034	0.078	0.052	0.00100
1.76	1.74	0.728	-0.028	0.099	0.066	0.00255
1.82	1.74	0.669	-0.020	0.121	0.084	0.00379
1.87	1.74	0.588	-0.019	0.149	0.097	0.00562
1.93	1.74	0.503	-0.010	0.158	0.109	0.00679
1.98	1.74	0.409	-0.009	0.156	0.109	0.00615

K.4 Mean LDA Values at y/a = 10.34

Measured by LDA, time averaged

x/a	y/a	U	V	Urms	Vrms	Reynolds stress
-6.897	10.34	0.0123	0.0234	0.0126	0.0087	-0.00003
-5.517	10.34	0.0104	0.0269	0.0138	0.0107	-0.00003
-4.138	10.34	0.012	0.0279	0.0280	0.0271	0.00003
-3.103	10.34	0.0911	0.0198	0.0794	0.0763	-0.00177
-2.414	10.34	0.2531	-0.001	0.1150	0.0990	-0.00507
-2.069	10.34	0.3774	-0.0054	0.1234	0.1072	-0.00571
-1.724	10.34	0.4778	-0.001	0.1262	0.1062	-0.00508
-1.552	10.34	0.5353	0.0025	0.1209	0.1095	-0.00395
-1.207	10.34	0.6043	0.0153	0.1143	0.1044	-0.00093
-0.862	10.34	0.6249	0.0266	0.1158	0.1011	0.00172
-0.517	10.34	0.5938	0.0294	0.1173	0.1015	0.00356
-0.172	10.34	0.5526	0.0246	0.1048	0.1026	0.00204
0.172	10.34	0.5486	0.0161	0.1068	0.1016	-0.00156
0.517	10.34	0.581	0.0148	0.1117	0.0987	-0.00286
0.862	10.34	0.6119	0.0242	0.1095	0.0987	-0.00084
1.207	10.34	0.5906	0.0384	0.1114	0.1032	0.00127
1.552	10.34	0.5145	0.0471	0.1221	0.1090	0.00474
1.897	10.34	0.4074	0.0501	0.1237	0.1133	0.00611
2.241	10.34	0.2878	0.0412	0.1109	0.0984	0.00561
2.931	10.34	0.1101	0.0063	0.0866	0.0867	0.00327
3.621	10.34	0.0157	-0.0263	0.0443	0.0473	0.00058
5	10.34	0.0043	-0.0283	0.0185	0.0126	0.00007
6.379	10.34	0.006	-0.0243	0.0150	0.0104	0.00006

K.5 Mean LDA Values at y/a = 34.48

Measured by LDA, time averaged

x/a	y/a	U	V	Urms	Vrms	Reynolds stress
-6.897	34.48	0.0817	0.011	0.074	0.059	-0.00181
-5.517	34.48	0.1729	-0.0048	0.097	0.080	-0.00350
-4.138	34.48	0.2721	-0.0118	0.104	0.088	-0.00411
-3.103	34.48	0.3646	-0.012	0.107	0.090	-0.00502
-2.414	34.48	0.3927	-0.0052	0.109	0.086	-0.00391
-2.069	34.48	0.4211	-0.0065	0.099	0.085	-0.00329
-1.724	34.48	0.4422	0.001	0.093	0.083	-0.00279
-1.552	34.48	0.4561	-0.001	0.097	0.083	-0.00251
-1.207	34.48	0.4737	0.0082	0.093	0.079	-0.00192
-0.862	34.48	0.4752	0.0113	0.090	0.085	-0.00179
-0.517	34.48	0.4928	0.0129	0.088	0.078	-0.00151
-0.172	34.48	0.5021	0.0161	0.083	0.081	0.00000
0.172	34.48	0.4996	0.022	0.084	0.079	0.00047
0.517	34.48	0.4948	0.0197	0.086	0.077	0.00148
0.862	34.48	0.4772	0.0251	0.091	0.085	0.00244
1.207	34.48	0.4667	0.0324	0.092	0.084	0.00257
1.552	34.48	0.4459	0.0327	0.095	0.084	0.00318
1.897	34.48	0.4215	0.0335	0.098	0.086	0.00367
2.241	34.48	0.3849	0.0262	0.110	0.088	0.00462
2.931	34.48	0.3316	0.0311	0.108	0.090	0.00466
3.621	34.48	0.2883	0.037	0.102	0.089	0.00472
5	34.48	0.1863	0.0252	0.090	0.077	0.00343
6.379	34.48	0.1044	0.0068	0.078	0.068	0.00245