A Framework to improve Turbulence Models using Full-field Inversion and Machine Learning

by

Anand Pratap Singh

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Aerospace Engineering)
in The University of Michigan
2018

Doctoral Committee:

Associate Professor Karthik Duraisamy, Chair
Associate Professor Krzysztof J. Fidkowski
Professor Krishnakumar R. Garikipati
Professor Ken Powell

“The truth is,
most of us
discover where
we are heading
when we arrive.”

Bill Watterson
Calvin and Hobbes

Anand Pratap Singh

anandps@umich.edu

ORCID iD: 0000-0002-1954-6375

To my mother for her courage and sacrifices
and to my late father for his dreams.

ACKNOWLEDGEMENTS

I have to confess, joining the PhD program, I was not sure about what I was going to do or what I was really interested in. The only thing I was sure of was that I loved being a student. However, thanks to my advisor Prof. Karthik Duraisamy and others, my graduate work has been extremely fruitful. In all honesty, the marathon of a PhD is never entirely fun, there were both good days and bad days, but it all was completely worth it. As John Donne said, no man is an island. This work could not be possible without an amalgamation of support from a number of people. I know I cannot possibly list everyone in this section.

I could not have asked for a better advisor than Prof. Duraisamy. He is tolerant towards what I will call productive procrastination, where I could work on a projects and ideas I loved but were not directly related to my PhD thesis. He was, in hindsight, kind enough to push me at times, and brutally honest. He was extremely patient and helpful with my writing. His mentoring style had me be more scientifically independent. Above everything, I am grateful for his patience during the toughest time of my personal life.

I would like to thank the committee members, Prof. Garikipati, Prof. Fidkowski, and Prof. Powell for their guidance and helpful suggestions during the course of this work and the preparation of this document.

I was fortunate to have a great cohort of colleagues and friends who made this journey worthwhile. I would like to thank Shivaji Medida for his help getting started

with the TURNS code when I was just starting my PhD and during my internship at Altair towards the end, which really boosted my graduate work. I thank Helen for her generous help with all the coursework during the first year - most of which I was careless enough to miss, and also for her help throughout the projects. I dont know if I should thank Eric, Nick and Ayoub for all our crazy shenanigans. In all seriousness, they have been great friends. I especially thank Yaser, Eric, Nick, and Walt for meticulously proofreading this document. All of the group members have each contributed in their own way to this work and also towards my personal development. Special mention and thanks to all of my FXB friends, Yuki, Chris, Devina for all the happy, fun times and memories. Brad, Wai Lee, and Yuntao were my first office-mates who really helped me settle were always approachable and warm. They played an important role as I progressed through my graduate work.

I would have never been able to do this without the unwavering support of my family. I thank my wife Arushi, who has been a pillar of support during the entire duration. I thank my parents Kanti and Dr. Ramesh Kumar Deshwali - their inspiration and sacrifices has brought me to where I am today. I am the luckiest person to be able to look up to them. I thank my sister Dr. Akanksha Deshwali for always believing in me.

This acknowledgement section took the longest to write, I really didn't want to write it at all. In a symbolic way this section probably marks the end of my student life, however, I will always continue to learn.

Funding for this work was provided by the NASA Aeronautics Research Institute under the Leading Edge Aeronautics Research for NASA program (technical monitors: Koushik Datta and Gary Coleman) and DARPA under the EQUiPS project (technical monitor: Dr. Fariba Fahroo). Computing resources were provided by the NSF via grant MRI: Acquisition of Conflux, A Novel Platform for Data-Driven Computational Physics (technical monitor: Ed Walker).

TABLE OF CONTENTS

CHAPTER

LIST OF TABLES

Table

LIST OF FIGURES

Figure

LIST OF ABBREVIATIONS

APG	adverse pressure gradient
CV	cross-validation
DES	detached-eddy simulation
DNS	direct numerical simulation
FANN	fast artificial neural network
FIML	field-inversion and machine-learning
LES	large-eddy simulation
MAP	maximum a-posteriori
MCMC	Markov-chain Monte Carlo
ML	machine-learning
NN	neural-network
NS	Navier-Stokes
RANS	Reynolds-averaged Navier-Stokes
SA	Spalart-Allmaras
TKE	turbulence kinetic energy
UQ	uncertainty quantification

ABSTRACT

Accurate prediction of turbulent flows remains a barrier to the widespread use of computational fluid dynamics in analysis and design. Since practical wall-bounded turbulent flows involve a very wide range of length and time scales, it is intractable to resolve all relevant scales, due to limitations in computational power. The usual tools for predictions, in order of their accuracy, includes direct numerical simulation (DNS), large-eddy simulation (LES), and Reynolds-averaged Navier-Stokes (RANS) based models.

DNS and LES will continue to be prohibitively expensive for analysis of high Reynolds number wall-bounded flows for at least two more decades and for much longer for design applications. At the same time, the high-quality data generated by such simulations provides detailed information about turbulence physics in affordable problems. Experimental measurements have the potential to offer limited data in more practical regimes. However, data from simulations and experiments are mostly used for validation, but not directly in model improvement.

This thesis presents a generalized framework of data-augmented modeling, which we refer to as field-inversion and machine-learning (FIML). FIML is utilized to develop augmentations to RANS-based models using data from DNS, LES or experiments. This framework involves the solution of multiple inverse problems to infer spatial discrepancies in a baseline turbulence model by minimizing the misfit between data and predictions. Solving the inverse problem to infer the spatial discrepancy field allows the use of a wide variety and fidelity of data. Inferring the field discrepancy using this approach connects the data and the turbulence model in a manner consistent with the underlying assumptions in the baseline model. Several such discrepancy fields are used as inputs to a machine learning procedure, which in turn reconstructs corrective functional forms in terms of local flow quantities. The machine-learned discrepancy is then embedded within existing turbulence closures, resulting in a partial differential equation/machine learning hybrid, and utilized for prediction.

The field-inversion and machine-learning (FIML) framework is applied to augment the Spalart-Allmaras (SA) and the Wilcox's $k - \omega$ model and for flows involving curvature, adverse pressure gradients, and separation. The value of the framework is demonstrated by augmenting the SA model for massively separated flows over airfoil using lift data for just one airfoil. The augmented SA model is able to accurately predict the surface pressure, the point of separation and the maximum lift – even for Reynolds numbers and airfoil shapes not used for training the model. The portability of the augmented model is demonstrated by utilizing in-house finite-volume flow solver with FIML to develop augmentations and embedding them in a commercial finite-element solver. The implication is that the machine-learning (ML)-augmented model can thus be used in a fashion that is similar to present-day turbulence model.

While the results presented in this thesis are limited to turbulence modeling, the FIML framework represents a general physics-constrained data-driven paradigm that can be applied to augment models governed by partial differential equations.

CHAPTER I

Introduction

Driven by an exponential growth in computational power, reliance on simulations has grown in every scientific field and the trend is expected to continue. Computational models are increasingly being used to drive design and development of engineering products. The rise in high fidelity simulations has led to better understanding of physical processes, which is being used to fill gaps in existing knowledge and to improve lower order theories/models.

Simulations of physical systems involve the solution of mathematical equations in their discretized forms. The accuracy of the solution depends on the correctness of the underlying mathematical model and the accuracy of the discretization approach. In other words, to obtain accurate results, the right set of equations should be solved using the right techniques.

Physical modeling involves the development of a mathematical relationship to describe a system. Such relationships may be derived based on first principles. For example, the governing equations of fluid dynamics, the Navier-Stokes (NS) equations, are the embodiments of the conservation of mass, momentum, and energy. In some cases, the models are purely empirical and are estimated by fitting the model to the data. Most practical models lie between the two extremes, with a blend of first principles and a few empirical relationships. All the examples discussed in the present work belong to this last category.

The quality of the solution of a discretized model often requires a trade-off between computational power and accuracy – particularly in the simulation of turbulent flows, where the difference between the smallest and the largest flow structures can be many orders of magnitude. Currently available computational power is not sufficient to resolve all the scales in such flows, and a fully-resolved simulation will not be attainable for the next few decades. Even when viable, it will require months of runtime on the world's most powerful computers, which will restrict its utility in an iterative design process.

Accurate prediction of turbulence is important as an increase in turbulence leads to enhanced mixing, heat transfer, and momentum transfer. These are quantities of interest in many situations and can affect critical design decisions. The inability of accurately simulating turbulent fluid flow in an affordable manner has led to lower-fidelity models, where the unresolved physics (small-scale) is modeled in terms of the resolved quantities (large scale). Examples of such models include large-eddy simulation (LES) and Reynolds-averaged Navier-Stokes (RANS) based models.

This chapter highlights properties of turbulent flows which lead to modeling difficulties. Various levels and methods of turbulence modeling are discussed along with some of their strengths and weaknesses. A case is made for data-driven augmentation of turbulence models. We discuss previous and ongoing efforts of using data-driven techniques to improve the prediction of turbulence. Against this backdrop, the motivation and the contribution of this work are presented. This discussion is tailored to provide context to the current work and is by no means exhaustive. Readers are referred to [74] for a more detailed discussion of turbulence modeling.

1.1 Fluid Flow and Turbulence

Turbulent flows are characterized by seemingly random structures or eddies. A non-dimensionless quantity, Reynolds number Re, describes the capability of the flow to transition from a laminar to a turbulent state. The Reynolds number, Re, is defined as the ratio of the momentum of the flow to the viscous force and is given by

$$Re = \frac{UL}{\nu}, \tag{1.1}$$

where U and L are characteristic velocity and length scales and ν is the kinematic viscosity of the fluid.

In a high Reynolds number flow, the size of the smallest and the largest eddies can differ by many orders of magnitude. Most flows of practical interest in the aerospace community are high Reynolds number flows, which are inherently turbulent or transition to turbulence from a laminar state. The typical Reynolds number of the flow over the wing of a commercial jet, based on the wing chord, is $\mathcal{O}(10^8)$.

The smallest scales in a turbulent flow were first estimated by A.N. Kolmogorov in 1941[50]. According to the Kolmogorov hypothesis, at sufficiently high Reynolds numbers, small-scale turbulent motions are statistically isotropic and the statistics of the small-scale motion is uniquely determined by the laminar kinematic viscosity ν and specific dissipation ϵ. The associated scales, also know as Kolmogorov microscales can then be defined by dimensional analysis

$$\eta \equiv \left(\frac{\nu^3}{\epsilon}\right)^{1/4}, u_\eta \equiv (\epsilon\nu)^{1/4}, \tau_\eta \equiv \left(\frac{\nu}{\epsilon}\right)^{1/2}, \tag{1.2}$$

where, η, u_η and τ_η are the length, velocity and time scale, respectively. Further, the small scales dissipate the energy, and therefore the relationship between the larger scales and smaller scales can be estimated as $\epsilon = U^3/L$, where U and L are the large

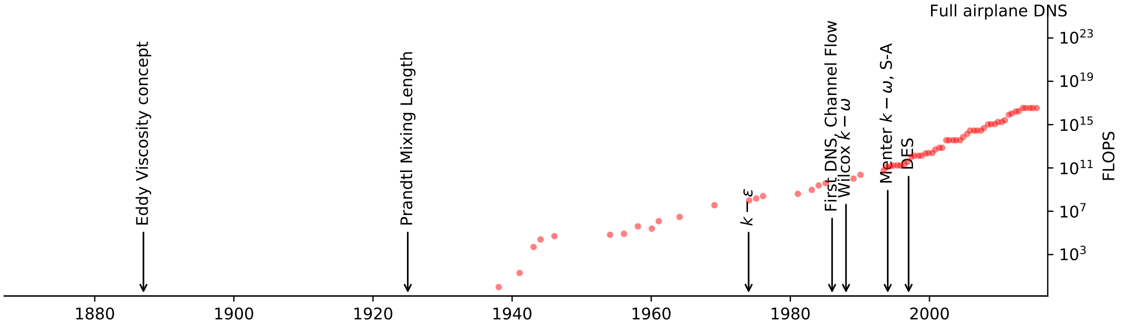


Figure 1.1: A notional time-line of development in turbulence simulations.

scale velocity and length scales. This results in the following relationships,

$$\eta/L \sim Re^{-3/4}, u_\eta/U \sim Re^{-1/4}, \tau_\eta/T \sim Re^{-1/2}. \tag{1.3}$$

The smallest length and time scales therefore depend on the Reynolds number of the flow. These range of scales makes it hard to solve the NS equations numerically with sufficient resolution for high Reynolds number flows.

1.2 Turbulence Modeling

The Navier-Stokes (NS) equations are non-linear partial differential equations and the non-linearity leads to a complex interaction between different scales of the flow. We solve the NS equations computationally, by discretizing them in some form. For this work, we focus on a mesh-based discretization.

The mesh resolution–the distance between two neighboring mesh points–governs the quality of the solution along with the numerical scheme. If all the scales in a flow are resolved, both in space and time, the solution is *fully resolved* and can be considered on par with the analytical solution[1]. However, most high Reynolds number simulations are not fully resolved because of excessive requirements of the

[1] Numerical errors associated with finite precision of the computers are always present.

computational resources. Such *under-resolved* simulations requires modeling of the unresolved scales, as they interact and affect the resolved scales via transfer of energy and momentum. An under-resolved simulation without such modeling would be inaccurate.

Models of turbulence range from fully-resolved to resolving only the ensemble average of the inherently unsteady turbulent flow. All the models of turbulent flows are however based on the NS equations. Various levels of models are discussed in the following sections.

1.2.1 Direct Numerical Simulation

In a direct numerical simulation (DNS), the NS equations are solved numerically, on a computational mesh, which is fine enough to resolve all the turbulent scales. Using the Einstein notation, the unsteady incompressible NS equations are given by,

$$\frac{\partial u_i}{\partial x_i} = 0 \tag{1.4}$$

$$\frac{\partial u_i}{\partial t} + \frac{\partial u_i u_j}{\partial x_j} = -\frac{1}{\rho}\frac{\partial p}{\partial x_i} + \nu \frac{\partial^2 u_i}{\partial x_j \partial x_j} \quad i = 1, 2, 3 \tag{1.5}$$

where, u is the velocity vector, p and ρ are the pressure, and density, respectively.

In three dimensions, in order to resolve the smallest length scale, η, the number of grid points grows as $(Re^{3/4})^3 \equiv Re^{9/4}$ (using equations 1.3). The number of time steps to resolve the time scale, τ_η, grows as $Re^{1/2}$. Therefore the effective work units increase as $\approx Re^3$. A doubling in Reynolds number requires eight times increase in the work units which translates to a proportional increase in computational time and resources. This simple estimation does not include the complexity of the algorithm used to numerically solve the NS equations, which adds to the overall complexity.

The cost associated with a DNS renders it impractical for high Reynolds number flows, such as those encountered in the Aerospace industry. DNS solutions can however, provide a very detailed description of the flow and are used to understand turbulence phenomena for simple canonical flows at low Reynolds numbers. Hence, DNS for simpler flows in channels, or pipes have been performed, and the knowledge has been extrapolated to improve lower order models[66, 53, 52, 30].

Fig. 1.2 shows an estimate of various milestones in the field of turbulence computations along with the increase in the processing power of the computers, calculated as floating-point operations per second (FLOPS). Also, marked is the processing power required for a full airplane simulation. It is expected that the milestone will not be reached for at least next few decades. Therefore, models which do not resolve all the scales will maintain their importance in engineering predictions.

Even if fully-resolved simulations are achievable, a design process which requires multiple solutions for perturbed geometry or flow conditions will maintain the need for less expensive models. In such situations, data from fully-resolved simulations could be used to augment the lower order models, and both levels of models can co-exist in symbiosis.

1.2.2 Large Eddy Simulation

In a large-eddy simulation (LES), the larger turbulent scales are resolved while the effects of the small scales are modeled. A filtering operation is defined and applied to the NS equations, which results in the following filtered continuity and momentum equations

$$\frac{\partial \bar{u}_i}{\partial x_i} = 0, \tag{1.6}$$

$$\frac{\partial \bar{u}_i}{\partial t} + \bar{u_j}\frac{\partial \bar{u}_i}{\partial x_j} = -\frac{1}{\rho}\frac{\partial \bar{p}}{\partial x_i} + \nu\frac{\partial^2 \bar{u}_i}{\partial x_j \partial x_j} - \frac{\partial \tau_{ij}}{\partial x_j}, \tag{1.7}$$

where $\bar{\phi}$ represents the filtered (or resolved) ϕ, and $\tau_{ij} = \overline{u_i u_j} - \bar{u}_i \bar{u_j}$. The filtering is typically implicitly performed by the mesh, as the resolution of the mesh governs the lowest length scale that can be resolved.

The filtered momentum equation contains a tensorial term τ_{ij}, referred to as the Smagorinsky (SGS) stress tensor, which is a function of the unresolved scales $\overline{u_i u_j}$. This leads to a closure problem. The tensor τ_{ij} is therefore modeled to close the equations. A simple model, referred to as the Smagorinsky, calculates effective viscosity ν_t such that,

$$\tau_{ij} = -2\nu_{\mathrm{t}}\bar{S}_{ij}, \tag{1.8}$$

where $\bar{S}_{ij} = \frac{1}{2}\left(\frac{\partial \bar{u}_i}{\partial x_j} + \frac{\partial \bar{u}_j}{\partial x_i}\right)$. The eddy-viscosity is assumed to be the function of mesh resolution and is given by the following relation,

$$\nu_{\mathrm{t}} = (C_s \Delta_g)^2 \sqrt{2\bar{S}_{ij}\bar{S}_{ij}}, \tag{1.9}$$

where C_s is a model constant and Δ_g is a measure of mesh spacing. The model is a function of the mesh spacing as the filtering operation depends on the mesh resolution. The filtered equations, along with the closure model are then solved numerically.

In the near-wall region, the number of required mesh points increases as $\approx Re^{1.8}$[16], which renders LES for high Reynolds number flows involving walls computationally intractable. *Wall models* have been proposed to model the near wall region to relax the resolution requirements. A robust and accurate modeling of wall effects remains a research problem. Performing a well-resolved and accurate LES requires significant computational resources and expertise, which has prevented it from becoming an industry workhorse.

1.2.3 Reynolds-averaged Navier-Stokes

RANS-based models are the most popular and widely used turbulence models. RANS-based models are relatively inexpensive, numerically robust, and require a low level of user expertise when compared to LES. Over a half-century of research and development has led to sufficiently accurate models in many flow problems.

In a RANS setting, the flow variables are decomposed as the sum of two components: ensemble average, and fluctuation. The former part is resolved both in space and time and the effects of the later part is modeled. The decomposition has the following property,

$$\phi = \bar{\phi} + \phi^{'} \tag{1.10}$$

$$\bar{\phi'} = 0 \tag{1.11}$$

where, $\bar{\phi}$ is the ensemble average of ϕ and $\phi^{'}$ is the fluctuation. The ensemble averaging operation, when applied to the NS equations, results in the following RANS equations

$$\frac{\partial \bar{u}_i}{\partial x_i} = 0, \tag{1.12}$$

$$\frac{\partial \bar{u}_i}{\partial t} + \bar{u_j} \frac{\partial \bar{u}_i}{\partial x_j} = -\frac{1}{\rho} \frac{\partial \bar{p}}{\partial x_i} + \nu \frac{\partial^2 \bar{u}_i}{\partial x_j \partial x_j} - \frac{\partial \overline{u_i' u_j'}}{\partial x_j}. \tag{1.13}$$

Similar to LES, the averaging leads to a closure problem because of the term $\overline{u_i' u_j'}$. This term requires knowledge of the fluctuations in velocity–which are not resolved– and therefore needs to be modeled in terms of the averaged quantities.

The unclosed term $\overline{u_i' u_j'}$ referred to as the Reynolds stress tensor, has some characteristics analogous to the viscous stress. The addition of this term leads to an effective increase in the diffusivity in the flow. Consequently, the term has been modeled based on analogies to kinetic theory, which leads to a model similar to molecular diffusion. It is also possible to derive transport equations for the Reynolds stresses starting with the NS equations. However, such equations are also unclosed as they require further higher order correlations (sec 1.4).

The various models for the Reynolds stresses are discussed in the following sections.

1.3 Eddy Viscosity-based Models

The simplest models for the Reynolds stresses are derived based on analogies between the Reynolds stress and the viscous stress. An effective additional viscosity, eddy-viscosity or turbulent viscosity, is defined. The Reynolds stresses are then given by a relation identical to the viscous diffusivity,

$$-\overline{u_i' u_j'} = 2\nu_t S_{ij}, \tag{1.14}$$

where S_{ij} is the mean strain rate tensor and ν_t is the eddy-viscosity. Consequently, the problem is transformed into that of estimating the eddy-viscosity instead of six Reynolds stress components.

While the analogy between the molecular viscosity and the eddy-viscosity reduces the complexity in modeling, it is built on assumptions which are not strictly valid in a turbulent flow. The molecular timescale is much smaller than the flow timescale, while the turbulence timescale and the flow timescale can be comparable. Therefore, the implicit assumption of stress being proportional to strain or equilibrium is almost never valid.

The eddy-viscosity assumption also assumes that the turbulent Reynolds stresses are isotropic, which is rarely valid (for instance, in the near-wall region). In wall-bounded flows, the blocking effects of the wall inhibits the wall-normal component of the Reynolds stresses more than the other components.

Despite overarching simplifications and its shortcomings, eddy-viscosity based models provide sufficiently accurate results for *simple* attached flows. Because of their simplicity, eddy-viscosity-based models form the workhorse of the industrial design and analysis.

In the subsequent sections, we discuss different types of eddy-viscosity models. The models range from simple algebraic relationships to those containing multiple transport equations with highly non-linear source terms. Full Reynolds stress closures and other ideas in the field of turbulence modeling are also discussed briefly. In this thesis, unless specified, the term "turbulence model" refers to an eddy-viscosity based model.

1.3.1 Algebraic Models

The simplest of turbulence models relates the eddy-viscosity to the flow properties by means of dimensional analysis. The eddy-viscosity ν_t has dimensions of velocity times length. Prandtl[77] assumed the velocity scale to be $l_m \left|\frac{\partial \bar{u}}{\partial y}\right|$, where, l_m is the turbulence length scale. Following this assumption, the eddy-viscosity can be written as

$$\nu_t = \left|\frac{\partial \bar{u}}{\partial y}\right| l_m^2. \tag{1.15}$$

The constant of proportionality is absorbed in the length scale. The mixing length l_m depends on the flow setup.

The Prandtl length scale model is *incomplete* as it requires the determination of the mixing length l_m, which can be tractable for parallel shear flows but difficult for more complex flows. Other notable algebraic models include the Cebeci-Smith [94] and the Baldwin-Lomax[5] model. Both these models have different eddy-viscosity expressions for the inner and outer layer of a boundary layer flow along with a damping function to ensure the Reynolds stresses asymptotically vanish at the wall with the correct slope.

Algebraic models lack information about the history of the flow which is known to be important. The problem can be addressed by adding history effects using transport equations as presented in the next section.

1.3.2 One Equation Models

The calculation of eddy-viscosity (Eq. 1.15) requires two scales: velocity and length. Prandtl [78] proposed the use of the turbulent kinetic energy as a measure of the velocity scale, hence replacing the relation used in the algebraic models. The governing equation of the turbulent kinetic energy can be derived by taking the trace of the Reynolds stress equations and is given by

$$\frac{\partial k}{\partial t} + u_j \frac{\partial k}{\partial x_j} = \nabla \cdot \left[\left(\nu + \frac{\nu_t}{\sigma_k} \right) \nabla k \right] + P - \epsilon \tag{1.16}$$

where $P = \tau_{t,ij} \frac{\partial u_i}{\partial x_j}$ is the production term and $\epsilon = \nu \overline{\frac{\partial u_i'}{\partial x_k} \frac{\partial u_i'}{\partial x_k}}$ is the dissipation term. σ_k is a constant which takes into account the pressure diffusion and the turbulence transport terms present in the equation of the kinetic energy. This equation is still unclosed as it requires the specification of ϵ and the Reynolds stresses $\tau_{t,ij}$. The eddy-viscosity assumption is used to define the Reynolds stress and Prandtl[78] derived an estimate $\epsilon = C_D k^{3/2}/l_m$ based on assumption of thin shear layer flow, where, C_D is a model parameter. The eddy-viscosity is then given by

$$\nu_t = c k^{1/2} l_m. \tag{1.17}$$

This model is also *incomplete* as it requires the specification of the length scale l_m similar to the algebraic models. Several variations to the original model have been proposed including [33, 38, 9]. More notable among the one-equation models is the Baldwin-Barth [4] model which does not require the specification of the length scale. A popular and notable one-equation model is the Spalart-Allmaras (SA) model which is described below.

Spalart-Allmaras model

The Spalart-Allmaras (SA) model[99] is a *complete* one equation model, which was developed primarily with attached flows in mind. The model solves for an effective eddy-viscosity $\tilde{\nu}$ and is not derived from existing turbulence kinetic energy (k) based models. The eddy-viscosity is derived from $\tilde{\nu}$. The equation for the $\tilde{\nu}$ has the following form,

$$\frac{\partial \tilde{\nu}}{\partial t} + u_j \frac{\partial \tilde{\nu}}{\partial x_j} = \nabla \cdot \left(\frac{\nu_t}{\sigma_{\tilde{\nu}}} \nabla \tilde{\nu} \right) + P - D. \tag{1.18}$$

where P and D are the production and the dissipation terms, respectively.

1.3.3 Two Equation Models

Most two equation models are based on transport equations for turbulent kinetic energy k and a second auxiliary quantity. The relationship between the dissipation and length scale is replaced by a differential equation of dissipation, which can again be derived by taking appropriate moments of the NS equations. One such form is given by

$$\frac{\partial \epsilon}{\partial t} + u_j \frac{\partial \epsilon}{\partial x_j} = \nabla \cdot \left(\frac{\nu_t}{\sigma_\epsilon} \nabla \epsilon \right) + C_{\epsilon 1} \frac{P_\epsilon}{k} - C_{\epsilon 2} \frac{\epsilon^2}{k}, \tag{1.19}$$

with eddy-viscosity given by

$$\nu_t = C_\mu k^2 / \epsilon. \tag{1.20}$$

This model is *complete* as it does not require knowledge of a specific problem except for the boundary conditions. Many models have been proposed based on the same idea with some modifications. Some popular examples of other two-equation models include Wilcox's $k - \omega$[113] which uses an equation of $\omega = \epsilon/k$ and $k - \tau$[103] where $\tau = 1/\omega$ and Menters $k - \omega$ SST[63].

$$\frac{\partial \omega}{\partial t} + u_j \frac{\partial \omega}{\partial x_j} = \nabla \cdot \left(\frac{\nu_t}{\sigma_\omega} \nabla \omega \right) + C_{\omega 1} \frac{P\omega}{k} - C_{\omega 2} \omega^2, \tag{1.21}$$

and where $\nu_t = C_\mu k/\omega$.

1.4 Reynolds Stress Closures

The exact transport equation for the Reynolds-stresses can be derived by taking appropriate moments of the NS equations. These equations are of the form,

$$\frac{\partial u_i' u_j'}{\partial t} + u_j \frac{\partial u_i' u_j'}{\partial x_j} + \frac{\partial T_{kij}}{\partial x_k} = P_{ij} + R_{ij} - \epsilon_{ij}. \tag{1.22}$$

In the Reynolds stress model, the production term is known, but the dissipation tensor, pressure-rate-of strain tensor and Reynolds stress flux require closure. These equations are usually closed with the help of an additional equation of some form of specific dissipation ϵ resulting in a total of 7 transport equations.

The Reynolds-stress models have explicit terms to characterize flows with significant streamline curvature, flows with a swirl or mean rotation. Reynolds-stress models are also anisotropic and can capture secondary flows such as the ones found in a duct. Examples of such models are LRR[31] and Wilcox's stress-ω[113].

1.5 Other Ideas

Deficiencies in linear eddy-viscosity models and second moment closures have led to more complex ideas for turbulence modeling. To combat the isotropy of the linear eddy-viscosity models Spalart et al. [98, 57] proposed a variant of the SA model with quadratic constitutive relations. The model is built on the SA model, but in place of the linear relationship between the stress and strain a non-linear expression is used, which, in principle can capture secondary flows. Transition models[51] have been proposed to replicate the transition of the flow from a laminar state to fully-turbulent. Such models are also built on existing baseline models.

The eddy-viscosity based models presented so far are based on quasi-homogenous flow assumptions. The eddy-viscosity is often calculated using turbulence kinetic energy k. However, wall-bounded flows are not homogeneous and such models predict inaccurate Reynolds stresses in the near wall region. Historically, such discrepancies have been corrected using exponential damping functions, which are based on calibration with the experimental data. Durbin [28] proposed an elliptic relaxation model $v^2 - f$. In the model, the non-homogeneous effects are modeled by solving a governing equation for a scalar measure of anisotropy (v^2), which in turn, depends on a variable f governed by an elliptic equation.

Detached-eddy simulation (DES) [101, 107] combines RANS and LES by appropriate switching between the two levels of models. In a DES, the RANS model is used in the regions where the turbulent length scale is lower than the grid resolution and LES is used in the rest of the regions. This leads to a tractable computational cost for the wall-bounded flows while resolving larger eddies away from the wall.

The presented material is not an exhaustive review of turbulence models. Other novel ideas include structure-based turbulence models[12], probability distribution function based models[49] and closures based on third and fourth order correlations of the velocity[123, 76].

1.6 A Case for Data-Driven Turbulence Modeling

Based on the discussion, it is clear that many turbulence models with varying degrees of complexity are in existence. Such a plethora of models exists because no single model is known to predict all kinds of flow accurately. Whether such a *universal* model exists is a questionable idea in itself. For long, it was predicted that LES would eventually replace RANS, but it has not yet happened. In fact, in constrast to fading away, with an increase in computational power, RANS is increasingly being used for more complicated flow cases.

There is no consensus on whether high-quality models, like full Reynolds stress closures, are viable alternatives. They have not resulted in any distinct improvement despite their theoretical advantages over eddy-viscosity models[102]. As a result, on top of baseline eddy-viscosity models, many corrections have been proposed to make the model respond to effects such as curvature and rotation[88, 93, 23], near-wall anisotropy[28], and transition from laminar to fully turbulent flows[51]. These corrections have not yet become mainstream because they perform well for a limited set of problems, and performance is only mediocre for general flows.

Despite the well-known deficiencies and empirical nature of RANS models, they are expected to remain the industry workhorse in the absence of a viable alternative.

Meanwhile, data-science has gained enormous traction in the last decade. The growth in data-science is driven by the need for automated processing, and classification of the vast amount of data generated and collected by the world wide web. Existing algorithms are improved, and new algorithms have been developed to satisfy the need of processing a large amount of data, also terms as big-data. Popular applications of such algorithms include language translation, speech recognition, and image recognition. At the same time, the increase in computing power and memory has made the application of these algorithms to process massive datasets a routine task.

Diffusion of ideas between the data-science community and the physical sciences have led to fascinating techniques of problem-solving. These early exploratory experiments have shown optimistic outcomes. Translation of ideas is not direct as the concepts from data-science have to be adapted to respect the known laws of physics. Such constraints do not exist in applications like facial recognition.

While high fidelity direct numerical simulations of turbulence, albeit for simple geometries and low Reynolds numbers, have been performed, they have not been used systematically to improve RANS models. The utility has been confined to calibrating parameters at best or to evaluate the quality of RANS solutions.

This availability of data and the growth in the field of data-science has led researchers to use data in an organized way to improve turbulence models. A brief survey of the literature in data-driven turbulence modeling is presented in the next section.

1.7 Previous Work on Data-Driven Model Improvements

The concept of using data to improve turbulence models is not new. In fact, it can be argued that turbulence modeling has always been data-driven as it involves calibration of parameters, for instance by matching the velocity log-layer or the skin friction prediction for zero pressure gradient flat-plate flows.

More sophisticated strategies have evolved over the last two decades. In 1998, Pareneix et al.[70] proposed the use of DNS datasets to evaluate the accuracy of second moment closure equations apriori by solving for one variable by fixing others to their DNS values. Based on the evaluation, the equations are modified to improve the model prediction. On the same line, Raiesi et al.[79] calculated the turbulence model variable using DNS and LES datasets for one and two-equation models. The authors report that the "...use of exact values of the turbulent kinetic energy and dissipation rate in the modeled eddy-viscosity did not improve its performance..." and "...the use of exact values of the turbulent kinetic energy deteriorates its performance...".

It can be argued that the dissipation or the turbulent kinetic energy used in the models are operational variables and not the real turbulent dissipation or the real turbulent kinetic energy. Therefore, using data to replace the terms in turbulent models directly is a futile strategy. Any correction to a model has to be suggested in a more comprehensive manner, and, it has to be consistent with all other assumptions involved with the turbulence model.

Milano and Koumoutsakos[65] used NNs to recreate the behavior of near-wall structures in a turbulent channel flow. The authors also augmented a second order wall model by using NN to represent the higher order terms as a function of only the wall quantities. Several researchers[122, 45, 29, 82, 15, 68] have used data to infer model parameters, probability distributions or joint probability distribution of the model parameters to quantify and reduce model errors. The probability distributions of the model parameters are used to predict the bounds on the model solution and provides a measure of uncertainty. Cheung et al.[15, 68] employed Bayesian model averaging[14] to calibrate model coefficients. Edeling et al.[29] used statistical inference on skin-friction and velocity data from a number of boundary layer experiments to quantify parametric model error. These methods provide insight into parametric uncertainties and address some of the deficiencies of *a priori* processing of data. However, they do not account for the model form uncertainties.

Dow and Wang[19, 20] made progress towards addressing non-parametric uncertainties by inferring the spatial structure of the eddy-viscosity required to match the DNS velocity. The discrepancy between the inferred and the $k-\omega$ eddy-viscosity was represented as a Gaussian random field and propagated to obtain uncertainty bounds on the mean flow quantities. The research group of Iaccarino[34, 39, 35] introduced ad-hoc, but realizable perturbations to the non-dimensional Reynolds stress anisotropy tensor a_{ij} to quantify structural errors in eddy-viscosity models. Tracey et al.[105] applied neural networks to large eddy simulation data to learn the functional form of the discrepancy in the eigenvalues of a_{ij} and injected these functional forms in a predictive simulation in an attempt to obtain improved predictions.

A natural precursor of the present work was performed by Tracey[106] at Stanford University. The work involved evaluating the SA turbulence model for a set of cases including flow over a flat plate, flow in a channel, and flow over an airfoil. A database of the analytical production and destruction terms are used to train their NN counterpart. The NN-based terms are then used for prediction replacing the analytical terms. While the predicted solution contains the same discrepancy present in the SA model, the exercise is a proof-of-concept of using machine learning based terms in an iterative RANS solver.

The work performed in this dissertation has been reported in a number of journals [89, 91] and conferences [90, 92, 27, 24, 26]. Several ideas have emerged, a year from the start of this dissertation (2014). They are presented below to differentiate from state of the art before 2015.

Post-2015

Xiao and co-workers[120, 110] calculated the spatial distribution of the perturbations in the anisotropy tensor $a_{ij} = \overline{u_i' u_j'} - \frac{2}{3}k\delta_{ij}$ using DNS data. The perturbations to the anisotropic tensor are calculated by transforming the eigenvalues to barycentric coordinate to ensure the realizability of the resulting perturbed stresses. The perturbations are then reconstructed using a machine learning algorithm as a function of local flow variables. The reconstructed machine-learned model, along with the RANS equations is used for prediction for a marginally difference flow setup. Xiao et al.[119] used metrics such as kernel density estimation and Mahalanobis distance to for a priori estimation of the quality of a machine-learned model.

Weatheritt[112] used evolutionary algorithms on DNS data to construct algebraic non-linear stress-strain relationships for RANS models to capture the anisotropic effect in an eddy-viscosity based model. The anisotropy tensor is evaluated using DNS data and formulated as a function of four linearly independent basis of a_{ij} and two scalar invariants. The final output of this strategy is an algebraic expression in contrast with a black-box model is most cases. The asymptotic behavior of an algebraic relationship is easy to analyze and also add to the confidence compared to a black-box model.

Ling and Templeton[54] used ML-based classifiers to ascertain regions of the flow in which commonly-used assumptions break down. Ling et al.[56] embedded the

Galilean invariance property of the Reynolds stress tensor in a neural network architecture and observed better training and prediction compared to generic neural network architecture.

Previous work on Model-form discrepancy

A popular approach to address model-form uncertainty is based on the work by Kennedy & O'Hagan[46]. In their work, the output quantity of interest is represented by

$$\mathbf{G}_d = \mathbf{G}_m(\mathbf{Q}, \lambda) + \delta(\mathbf{Q}) + \epsilon,$$

where δ is the model discrepancy, ϵ represents measurement errors, and λ contains the model hyper-parameters. Gaussian processes are assumed for $\mathbf{G}_m$, δ, and ϵ and Bayesian inference is used to infer the posterior distribution of the hyper-parameters λ. The approach can be applied to any model without the knowledge of the underlying physical system, which is both an advantage and also a disadvantage. Other concerns with this approach include the inability to discern the effects of λ and δ independently, the physically unconstrained approach of modeling δ, the dependence of the model discrepancy on the observable d, and its inability to accurately model non-Gaussian phenomena. To address these shortcomings, in this work, model discrepancies are *embedded* within the governing equations (rather than added to the *output.* Thus, physics constraints can still be satisfied using the underlying governing equations and causality can be better established.

1.8 Contributions of the Present Work

Most of the previous work for model improvement involves the use of data to calibrate parameters in existing models. However, the error in the models is structural which cannot be eliminated by recalibration. Further, all the previous work involves the use of full-field LES or DNS data. Such datasets are limited in availability and are confined to low Reynolds number flows and simple geometries. Moreover, model variables such as turbulent kinetic energy k and dissipation ϵ are not real quantities. They are just operational variables, and therefore direct extraction of these quantities from a LES or DNS have not lead to more accurate models. Past work also overlooked the existence of experimental data which can be sparse (measurements of surface pressure, lift, etc.) but are available for high Reynolds number flow and realistic geometries.

The focus of this work is to address the shortcomings of the previous work and propose a method for functional augmentation using data of varying quality and quantity. The specific contributions are as follows:

- A framework of FIML is proposed and developed to augment existing RANS based turbulence models. The framework builds on existing turbulence models, and uses inversion and ML to augment them. The existing approaches of applying machine learning connects the data directly to the model. For example, the turbulent kinetic energy (k) from DNS or LES data can be reconstructed using machine learning and be used to replace k in the $k - \omega$ model. However, the *real* k and ω in DNS and LES are different from the k and ω used in the RANS model, where they are operational quantities. In the current approach, inverse problems are solved to connect these operational quantities with data. We look for the additional correction/discrepancy term, in the context or the baseline model, which is required for the model prediction to lie closer to the data. The advantages of using inversion are two-fold:

    1. The introduced correction is consistent with existing turbulence models.
    2. Any data can be used to solve the inverse problem, from full-field velocity to integral quantity of lift. Knowledge of quantities like k and ω, which

appear in the model explicitly, is not required.

- **Proof of concept:** The framework is evaluated on three different flow setups. In the first configuration, data is synthetically generated using a known analytic curvature correction to the SA model and the FIML framework is used to reconstruct the correction. Since all the information is known for this setup, there are no uncertainties related to flow conditions, discretization and hence in data. It is shown that the complete flow field, including Reynolds-stresses, can be precisely predicted by the FIML based augmentation if the full-field velocity is used as data. The quality of the prediction reduces if skin-friction is used as the data, but skin-friction, near wall Reynolds stresses, and velocities are still predicted accurately.

- **Model improvement via ensemble data:** Wilcox's $k-\omega$ model is augmented using data from LES for a set of flows over a bump, which serves as a proxy for adverse pressure gradients. It is shown that the inverse solution for just two different bump heights can be used to reconstruct an augmented model which works on other bump heights and different incoming flow conditions. The prediction using full-field velocity data and skin-friction are found to be similar.

- **Limited data, extrapolation, portability:** The potential of the FIML framework is demonstrated by augmenting the SA model using only the data for the lift coefficient for one airfoil. Improvements in pressure, the location of separation, and maximum lift coefficient are achieved for different airfoil shapes and Reynolds number not used for training. It is shown that using pressure and lift as data results in almost identical FIML augmentation and hence prediction. This opens up the opportunity of widely available experimental data to be used for augmentation of RANS-based models.

1.9 Organization

This thesis is organized as follows:

- Chapter 2 describes the proposed framework of FIML and describes the details of the two important steps of full field inversion and ML.
- In Chapter 3, the compressible form of the governing equations and the techniques of spatial and temporal discretization are described. Details about the discrete adjoint framework are presented.
- As a proof-of-concept, in Chapter 4, the FIML framework is applied to flow in a curved channel, where the data is generated synthetically by using an analytic correction. The resulting solution and the augmented model result from the FIML are compared with the analytical correction.
- Chapter 5 presents the application of the FIML framework on adverse pressure gradient flows. Data from LES simulations are used to improve the Wilcox's $k-\omega$ model for such flows.
- Chapter 6 shows the potential of the FIML framework by using experimental data for flow over an airfoil to improve prediction over a set of airfoils over a range of Reynolds number.
- Conclusions and recommendations for future work are presented in Chapter 7.

CHAPTER II

Field Inversion and Machine Learning Framework

This chapter introduces the field-inversion and machine-learning (FIML) framework for model augmentation. The framework consists of three key steps:

1. Solution of inverse problems to infer discrepancies in an existing model by minimizing the differences between data and predictions
2. Training ML algorithms to construct a model of the inferred discrepancy, as a function of local flow solution
3. Using the ML based model of discrepancy in an iterative RANS solver for predictions.

Inverse problems are required to establish the relationship between available data and discrepancies in existing models. Machine learning algorithms are used to find patterns in the discrepancies, which can then be used for predictions. The first two steps are analogous to the (1) transformation of data into useful information, and (2) transformation of the information into knowledge. In the third step, the knowledge is used to make predictions. Fig. 2.1 shows a schematic of the complete framework, distinguishing the offline (inversion and training of machine learning model) and the online (prediction) component.

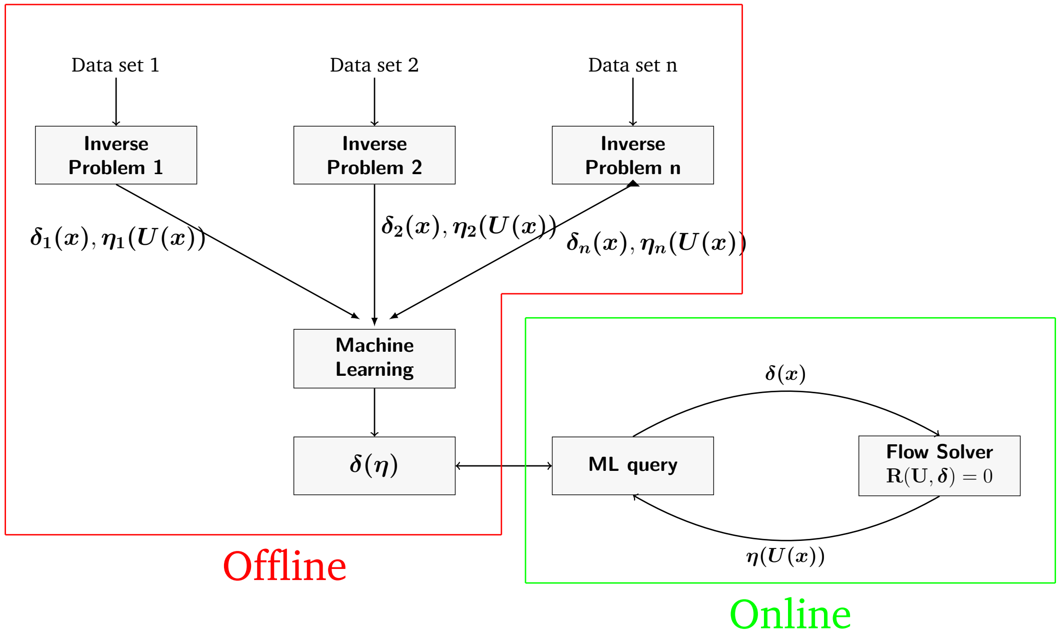


Figure 2.1: Schematic of the field-inversion and machine-learning (FIML) showing the offline components which includes inference and training of the machine learning model and the online prediction.

We start with discussing the form of discrepancies present in physical models, which is followed by a discussion to infer such discrepancies. We examine the need for inference or inversion and present two different perspectives of framing an inverse problem: Bayesian and deterministic. Details of the methodology used to solve an inverse problem are presented. An introduction to machine learning is presented followed by a discussion of important considerations when using a machine learning based model. Two machine learning algorithms used in this work, neural-network (NN) and AdaBoost, are discussed.

To preserve the generality of the approach, we restrict to a level of abstraction. We leave the implementation-related details for specific RANS models in subsequent chapters. While analogies and examples are borrowed from turbulence modeling, the approach can be applied to any physical situation.

2.1 Nature of Modeling Discrepancy

Most practical models of physical problems combine theoretical foundations with empirical calibration of a few parameters. For example, ideal gas law relates pressure, density, and temperature using a simple algebraic relationship with the gas constant as a calibrated (or estimated) parameter. A pertinent example is a differential equation based turbulence model. Such models contain source terms based on theoretical understanding of the flow physics, and these terms contain constants calibrated on canonical flows. These models suffer from both: the inadequate understanding of theory, and the errors associated with calibration. Simple recalibration can address the discrepancy with the calibration, but it is not sufficient to mitigate the missing knowledge of the theory. For example, the missing real gases effects in the ideal gas equation cannot be mitigated by using a different constant. In the context of the turbulence model equations, the discrepancies are in the **functional form** of the source terms. While data-driven parameter calibration [122, 15, 68, 29] is not new, this work focuses on the discovery of this **functional discrepancy**.

The novelty of this work lies in the introduction of discrepancy and techniques used to recover it from data. Data is used to augment and fill gaps in existing theories–which are based on decades of research–and not to replace them. The discovered form of discrepancy may be used to gain physical insight; however, sometimes it is purely a mathematical construct. An example of the latter case is a two-dimensional separated flow over an airfoil. We can recover a model form, from data, such that the model output agree with the data, but it is well known that the flow separation is an inherently three-dimensional phenomenon. In such cases, it would be overreaching to derive a physical interpretation.

2.1.1 Mathematical representation

The mathematical description of the functional discrepancy is discussed in the context of a representative model. Consider the following model:

$$R(\mathbf{Q}, \mathbf{G}) = 0, \tag{2.1}$$

$$\frac{D\mathbf{G}}{Dt} = T_1(\mathbf{Q}, \mathbf{G}) + T_2(\mathbf{Q}, \mathbf{G}) + T_3(\mathbf{Q}, \mathbf{G}), \tag{2.2}$$

where $R(\mathbf{Q}, \mathbf{G})$ is the base model equation, $\mathbf{Q}$ is the model solution, $\mathbf{G}$ is an operational variable, and T_i's are the source terms. In the turbulence modeling context, equation 2.1 refers to the RANS equations. For the SA turbulence model [99] $\mathbf{G}$ is the operational eddy viscosity, T_1 is the production term, T_2 is negative of destruction, and T_3 is the turbulent transport term. For the Wilcox's $k - \omega$ model [114], $\mathbf{G}$ is a vector containing k and ω and all the source terms are also vectors.

Applications of the presented framework are not restricted to models of the form mentioned above. However, this form is representative of turbulence models, which are the primary subject considered in this work. Among others, the formulation can be applied to models which are not differential in nature, and can also be used to introduce discrepancies in boundary conditions.

As argued in the previous section, the discrepancy in the model is functional in nature, and hence we introduce a functional correction term or a discrepancy term to correct for it. The discrepancy is required to be a function of the model solution. After the introduction of the functional correction term, $\boldsymbol{\delta}(\mathbf{Q}, \mathbf{G})$, the new model equation has the form,

$$\frac{D\mathbf{G}}{Dt} = T_1(\mathbf{Q}, \mathbf{G}) + T_2(\mathbf{Q}, \mathbf{G}) + T_3(\mathbf{Q}, \mathbf{G}) + \boldsymbol{\delta}(\mathbf{Q}, \mathbf{G}). \tag{2.3}$$

This work deals with discovering the form of the function $\boldsymbol{\delta}(\mathbf{Q}, \mathbf{G})$, such that the

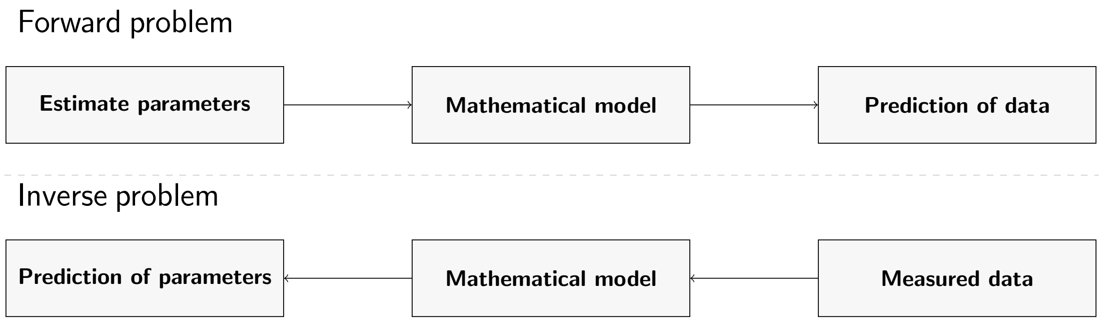


Figure 2.2: Flowchart depicting the connection between the forward and the inverse problem.

model output agrees with the data, and predictions can be improved.

We infer the spatial distribution of the $\boldsymbol{\delta}(\mathbf{Q}, \mathbf{G}) \equiv \boldsymbol{\delta}(\mathbf{x})$ required to minimize some measure of the misfit between the model output and the data. Such spatial distributions are sought for a number of problems. As solution to inverse problems are required to infer field distributions of $\boldsymbol{\delta}$, we term the process as full field inversion. For the inversion, $\boldsymbol{\delta}$ is a spatial function, but the final goal of the process is to get $\boldsymbol{\delta}$ as a function of locally non-dimensional flow features. That is achieved by training a machine learning model on many inverse solutions. The next sections cover these aspects of the FIML framework.

2.2 Full Field Inverse Problem

Inverse problems, as the name suggests, are opposite to forward problems (Fig 2.2). In a typical forward problem, a set of mathematical equations are solved conditional on a few parameters and boundary conditions. In the inverse problem, the output is known, and parameters or boundary conditions includes the unknowns. Consider a forward system of equations

$$\mathbf{u} = \mathbf{R}(\mathbf{m}) \tag{2.4}$$

where $\mathbf{m}$ includes model parameters and boundary conditions. The inverse problem involved obtaining a map $\mathbf{F}(\mathbf{u})$ such that

$$\mathbf{m} = \mathbf{F}(\mathbf{u}) \mid \mathbf{u} \approx \mathbf{u_{data}} \tag{2.5}$$

In a *full-field* inverse problem a spatial distribution for $\boldsymbol{\delta}$ (in equation 2.3) is sought in order to match the model output with data, and not a few parameters. When the model equations are discretized, the discrepancy term is transformed from infinite dimensional to finite dimensional and is defined at every grid point or cell volume depending on the type of discretization. Therefore after discretization, the problem can be seen as a parameter inference problem, with a very large number of parameters. The large numbers of unknowns add to the complexity in obtaining the solution of the inverse problem. The difficulty in obtaining the inverse solution also comes from noise in the data, non-uniqueness of the inverse solution, and non-linear nature of the forward problem.

In this work, many inverse problems are solved, to infer the discrepancy $\boldsymbol{\delta}$ in an existing model, for different flow setups.

Rationale Behind Solving an Inverse Problem

The rationale behind solving an inverse problem is not clear at first glance. In the presence of high-fidelity data, it is reasonable to calculate $\boldsymbol{\delta}$ directly using equation 2.3 by evaluating the various terms using data. However, we emphasize that the source terms present in the model are not *real* physical terms. For example, the production of eddy viscosity is not actual production–it is an operational term designed to match some criteria. Therefore, direct calculation of $\boldsymbol{\delta}(\mathbf{Q}, \mathbf{G})$ may not provide the correction required for a model to produce more accurate solutions. In fact, it may deteriorate the solution[75, 79, 70].

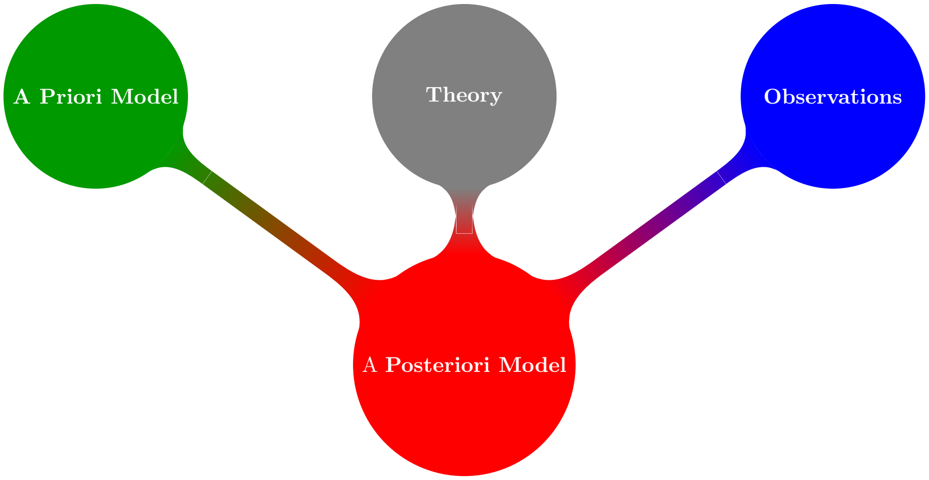


Figure 2.3: Bayesian inference involves calculation of posterior probability distribution using prior distribution and the likelihood.

Furthermore, the inverse framework provides a generalized approach of using data of various levels of fidelity. The data can be as sparse as a lift coefficient or as dense as full velocity fields. Inverse problem theoretically does not differentiate between the two cases, and therefore both can be used to infer a discrepancy field. It is, of course, expected that the effectiveness of the inferred discrepancy will be dependent on whether the data is sufficiently informative of the model discrepancy.

2.2.1 Types of Inverse Problems

An inverse problem can be formulated from two different perspectives: (1) Bayesian and (2) deterministic. The Bayesian perspective allows for a rigorous consideration and treatment of uncertainties associated with the inverse solutions. It provides a means for propagating the uncertainties in data to the inferred solution. The deterministic perspective does not account for the uncertainties. The two perspectives are discussed below in detail.

2.2.1.1 Bayesian Inverse Problem

While the majority of the computational science work centers on models which are deterministic, the models and hence the solutions always have some level of uncertainty. Uncertainties may arise from imperfect equations, boundary conditions, or numerical methods. In a Bayesian setting, both the data and the spatial discrepancy are assumed to be random variables. The objective is to infer the posterior probability distribution of the discrepancy, given a prior distribution of the discrepancy and the distribution of the data.

Equation 2.6 is the mathematical representation of the Bayes theorem, which states that the posterior probability distribution of a random variable, $\boldsymbol{\delta} \in \Re^n$, is proportional to the prior distribution of $\boldsymbol{\delta}$ times the likelihood $\mathbf{h}$, $\mathbf{h} \in \Re^m$ and $\mathbf{d} \in \Re^m$. Note that $\boldsymbol{\delta}$ is a vector and $\mathbf{d}$ is a vector.

$$p(\boldsymbol{\delta}|\mathbf{d}) \propto h(\mathbf{d}|\boldsymbol{\delta})p(\boldsymbol{\delta}), \tag{2.6}$$

The traditional process of solving this problem involves the use of sampling algorithms. In a typical sampling algorithm, samples are drawn from the prior and likelihood is evaluated, which is then used to update the posterior. Such methods suffer from the curse of dimensionality, and the number of samples required for such cases increases exponentially with the size of the $\boldsymbol{\delta}$ vector.

A further assumption of Gaussian nature of data and the prior can be used to simplify the general problem to that of obtaining a maximum a-posteriori (MAP) solution followed by calculation of the posterior covariance by inverting the Hessian of the objective function[2]. Calculation of the MAP involves solution of an optimization problem with the following objective function,

$$\begin{aligned}\boldsymbol{\delta}_{MAP} &= \arg\min_{\delta} \frac{1}{2}\left[(\mathbf{d}-\mathbf{h}(\delta))^T\mathbf{C}_{obs}^{-1}(\mathbf{d}-\mathbf{h}(\delta)) + (\boldsymbol{\delta}-\boldsymbol{\delta}_{prior})^T\mathbf{C}_{prior}^{-1}(\boldsymbol{\delta}-\boldsymbol{\delta}_{prior})\right] && (2.7)\\ &= J(\boldsymbol{\delta}), && (2.8)\end{aligned}$$

where $\mathbf{C}_{obs}$ and $\mathbf{C}_{prior}$ are the observation and the prior covariance matrices respectively. $\boldsymbol{\delta}_{prior}$ is the prior (initial) value of $\boldsymbol{\delta}$, which is always zero for our problems. Following this, the posterior covariance matrix is calculated using

$$\mathbf{C}_{posterior} = \left[\frac{d^2 J(\boldsymbol{\delta})}{d\boldsymbol{\delta} d\boldsymbol{\delta}}\right]^{-1}\Bigg|_{\boldsymbol{\delta}_{MAP}}. \qquad (2.9)$$

Once the posterior covariance is known, the samples of posterior discrepancies are evaluated by Cholesky factorization of the covariance

$$\mathbf{R}^T\mathbf{R} = \mathbf{C}_{posterior} \qquad (2.10)$$

and

$$\boldsymbol{\delta} = \boldsymbol{\delta}_{MAP} + \mathbf{R}^T\mathbf{s}, \qquad (2.11)$$

where $\mathbf{s}$ is a vector of normally distributed random numbers with a mean of zero and a standard deviation of unity.

Further assumption of constant diagonal covariances, $\mathbf{C}_{obs} = \sigma_{obs}^2\mathbf{I}$ and $\mathbf{C}_{prior} = \sigma_{prior}^2\mathbf{I}$, simplifies the objective function to[2],

$$J(\boldsymbol{\delta}) = \frac{1}{2}\left[\frac{1}{\sigma_{obs}^2}(\mathbf{d}-\mathbf{h}(\boldsymbol{\delta}))^T(\mathbf{d}-\mathbf{h}(\boldsymbol{\delta})) + \frac{1}{\sigma_{prior}^2}(\boldsymbol{\delta}-\boldsymbol{\delta}_{prior})^T(\boldsymbol{\delta}-\boldsymbol{\delta}_{prior})\right]. \qquad (2.12)$$

Turbulence models constitute complex nonlinear systems, and thus the posterior distribution and the likelihood cannot be expected to be Gaussian. The error introduced by the Gaussian assumption is challenging to estimate without the use of sampling strategies, which are very expensive for the problems considered in this work.

Parish & Duraisamy[69] studied the effects of different levels of assumptions for a synthetic heat conduction problem. While the calculation of uncertainties is a theoretically appealing process, it is typically prohibitively expensive in problems involving many parameters. The prior distribution of the parameters has a significant impact on the posterior. However, the approach of selecting the prior is not well understood and is subjective. For these reasons, we employ a deterministic strategy to frame and solve the inverse problem for the examples presented in this work.

2.2.1.2 Deterministic Inverse Problem

In a deterministic setting, the objective is to minimize a measure of discrepancy between the model output and the data. The formulation is ill-posed because of noise in the data and a high degree of freedom in the model compared to the number of data points. The problem is regularized by an additional term in the objective function. The selection of the regularization term is not unique. For this work, we use a regularization which biases the solution to lie near to the prior (initial) solution; this is a particular case of Tikhonov regularization[7] with a diagonal prior covariance matrix. The optimization problem, in this case, is given by:

$$\begin{aligned}
\boldsymbol{\delta}_{inverse} &= \arg\min_{\boldsymbol{\delta}} \frac{1}{2}\left(J_1(\boldsymbol{\delta}) + \lambda J_2(\boldsymbol{\delta})\right) && (2.13)\\
&= \arg\min_{\boldsymbol{\delta}} \frac{1}{2}\left[(\mathbf{d} - \mathbf{h}(\boldsymbol{\delta}))^T(\mathbf{d} - \mathbf{h}(\boldsymbol{\delta})) + \lambda(\boldsymbol{\delta} - \boldsymbol{\delta}_{prior})^T(\boldsymbol{\delta} - \boldsymbol{\delta}_{prior})\right] && (2.14)
\end{aligned}$$

It can be seen that this objective function is identical (except for a scaling) with the Bayesian objective function (Eq. 2.12) with diagonal co-variance and $\lambda = \sigma^2_{obs}/\sigma^2_{prior}$.

Setting a value for λ requires engineering judgment. A higher value of λ leads to

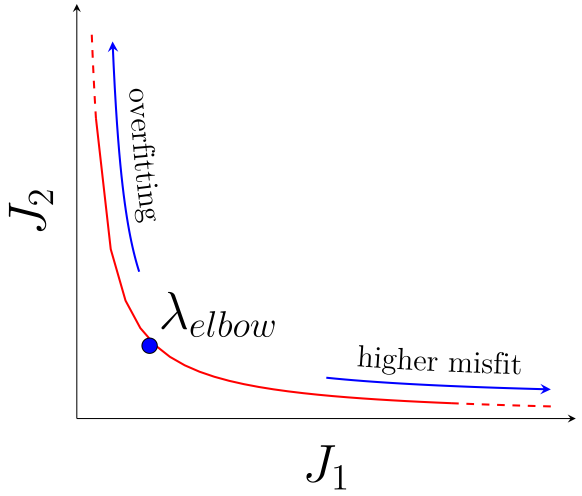


Figure 2.4: L-curves are used to fix the value of λ, but it requires solution to many inverse problems for different values of λ's.

a high misfit while a low value of λ leads to over-fitting. In case the confidence in the data is high, we can choose a very small value of λ. For other cases, λ can be selected by estimating the trust in the observation and the model. For problems without such information, the value of λ is chosen by solving multiple inverse problems for different values of λ and plotting the two parts of the objective function. This results in an "L curve" as shown in Fig. 2.4. The value corresponding to the elbow is selected, which represents a balance between over-fitting and the discrepancy.

In the examples presented in this work, the value of λ does not affect the qualitative distribution of inferred discrepancy, but it does affect the absolute values to some degree.

2.2.2 Optimization Problem and Discrete Adjoints

Recent efforts have been made to accelerate Markov-chain Monte Carlo (MCMC) by using structure-exploiting proposal densities[41, 109, 58, 11] and dimensional reduction[17, 18]. Despite the progress, many practical challenges remain before it can be used for problems of our interest, which involves tens of thousands of variables and moderately expensive RANS solver for evaluation of the forward problem.

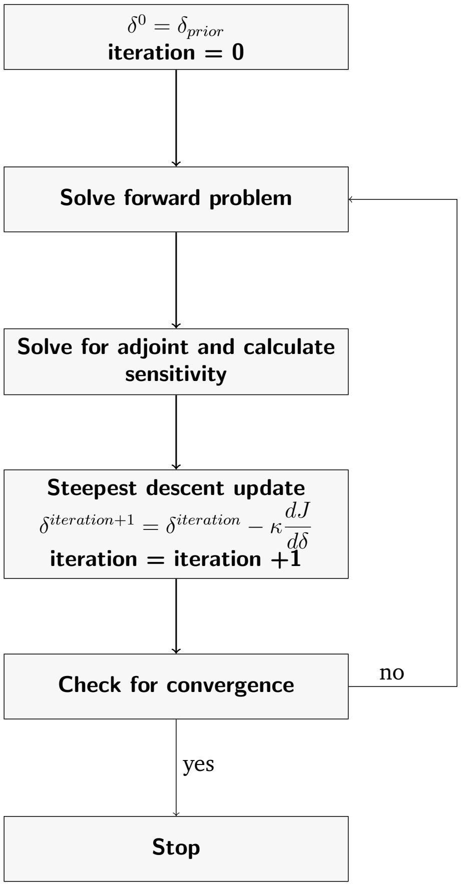

$\delta^0 = \delta_{prior}$
iteration = 0
Solve forward problem
Solve for adjoint and calculate sensitivity
Steepest descent update
$\delta^{iteration+1} = \delta^{iteration} - \kappa \frac{dJ}{d\delta}$
iteration = iteration +1
Check for convergence
no
yes
Stop

Figure 2.5: Solving an inverse problem is equivalent to solving an optimization problem with an appropriate objective function, J, and the discrepancy $\boldsymbol{\delta}$ being the design variable.

Therefore, irrespective of the choice of the problem setup, the inverse problem involves the solution of an optimization problem. We use a discrete adjoint approach to calculate the sensitivities required to solve the optimization problem efficiently. Examples presented in this work use a simple steepest-descent algorithm, because of its ease of implementation. Steepest-descent algorithm suffers from a relatively slow convergence rate, but we found it to be satisfactory for our work.

A steepest descent update is given by

$$\boldsymbol{\delta}^{n+1} = \boldsymbol{\delta}^{n} - \kappa \frac{dJ}{d\boldsymbol{\delta}} \tag{2.15}$$

where, κ represents the step size. κ can be fixed to a small value or appropriate line search [118, pg. 30-65] strategies can be used to estimate κ. The gradient of the objective function $\dfrac{dJ}{d\boldsymbol{\delta}}$ is calculated using discrete adjoint as discussed below.

2.2.2.1 Discrete Adjoint Formulation

A discrete adjoint[37] formulation is used to calculate the gradient. Discrete adjoints provide gradients which are consistent with the discretized form of the governing equations. The finite difference approximation of the gradients involves at least N forward solves, where N is the number of unknowns. The number of unknowns in our work is same as the number of grid points, which can be in the order of tens of thousands even for a two-dimensional problem. The calculation of discrete adjoint based sensitivity requires a solution of a linear system and few matrix-vector multiplications. Because of its efficiency, adjoints are a popular tool for calculation of sensitivities in the field of design and optimization. From an optimization viewpoint, the discrepancy $\boldsymbol{\delta}$ can be seen as the design variables. The derivation of the discrete adjoint formulation for the present optimization setup is discussed below.

The optimization problem is framed as that of minimizing the objective function while satisfying the governing equation. Mathematically,

$$\min_{\boldsymbol{\delta}} J(\mathbf{U}, \boldsymbol{\delta}) \tag{2.16}$$

$$\mathbf{R}(\mathbf{U}, \boldsymbol{\delta}) = 0 \tag{2.17}$$

where **R** includes the RANS equations and the turbulence model, and **U** includes the flow variables and the turbulence models variables. All the cases presented in this work are for steady flows and therefore no time derivative is present in the governing equation. The discrete adjoint formulation is derived by differentiating equations 2.16 and 2.17 with respect to the design variable δ, which results in,

$$\frac{dJ}{d\boldsymbol{\delta}} = \frac{\partial J}{\partial \mathbf{U}} \frac{d\mathbf{U}}{d\boldsymbol{\delta}} + \frac{\partial J}{\partial \boldsymbol{\delta}} \tag{2.18}$$

and

$$\frac{d\mathbf{R}}{d\boldsymbol{\delta}} = \frac{\partial \mathbf{R}}{\partial \mathbf{U}} \frac{d\mathbf{U}}{d\boldsymbol{\delta}} + \frac{\partial \mathbf{R}}{\partial \boldsymbol{\delta}} = 0 \tag{2.19}$$

$$\Rightarrow \frac{d\mathbf{U}}{d\boldsymbol{\delta}} = \left(\frac{\partial \mathbf{R}}{\partial \mathbf{U}}\right)^{-1} \left(-\frac{\partial \mathbf{R}}{\partial \boldsymbol{\delta}}\right) \tag{2.20}$$

The term $\frac{d\mathbf{U}}{d\boldsymbol{\delta}}$ is then substituted in 2.18, which gives

$$\frac{dJ}{d\boldsymbol{\delta}} = \frac{\partial J}{\partial \boldsymbol{\delta}} - \frac{\partial J}{\partial \mathbf{U}} \left(\frac{\partial \mathbf{R}}{\partial \mathbf{U}}\right)^{-1} \left(\frac{\partial \mathbf{R}}{\partial \boldsymbol{\delta}}\right) \tag{2.21}$$

Now, we define the adjoint variable Ψ as

$$\Psi = \left[\frac{\partial J}{\partial \mathbf{U}} \left(\frac{\partial \mathbf{R}}{\partial \mathbf{U}}\right)^{-1}\right]^T \tag{2.22}$$

Following which, the computation of the sensitivity of the objective function with respect to the design variables is calculated by

1. Solving the linear system

$$\left[\frac{\partial \mathbf{R}}{\partial \mathbf{U}}\right]^T \psi = \left[\frac{\partial J}{\partial \mathbf{U}}\right]^T \tag{2.23}$$

2. Computing

$$\frac{dJ}{d\boldsymbol{\delta}} = \frac{\partial J}{\partial \boldsymbol{\delta}} - \psi^T \frac{\partial \mathbf{R}}{\partial \boldsymbol{\delta}} \tag{2.24}$$

The partial derivatives in equations 2.23 and 2.24 can be calculated either using finite difference [83], complex step differentiation [60], or automatic differentiation [80]. The adjoint equation (Eq. 2.23) is independent of the design variables, therefore the cost of solving the linear system does not depend on the size of the design variables.

Next, we delve into the details of machine learning, which forms the second part of the FIML framework.

2.3 Machine Learning

Machine-learning (ML), in simple terms, constitutes a set of methods and tools to discover patterns in data and then use the patterns to predict a future or perform decision-making. Machine learning has gained popularity in recent years because of advances in computing and also a rise in quantity of data in the technology industry. For example, it is expected that by the year 2020, about 1.7 megabytes of new information will be created every second for every human being on the planet[32]. This vast quantity of data needs automatic methods of finding patterns and useful information: machine learning.

Machine-learning (ML) algorithms can be divided into three main categories: supervised learning, unsupervised learning, and reinforcement learning. Given pairs of input-output $\mathfrak{D} = \{(\boldsymbol{\eta}_i, \delta_i)\}_{i=1}^N$ supervised learning is used to learn a mapping between

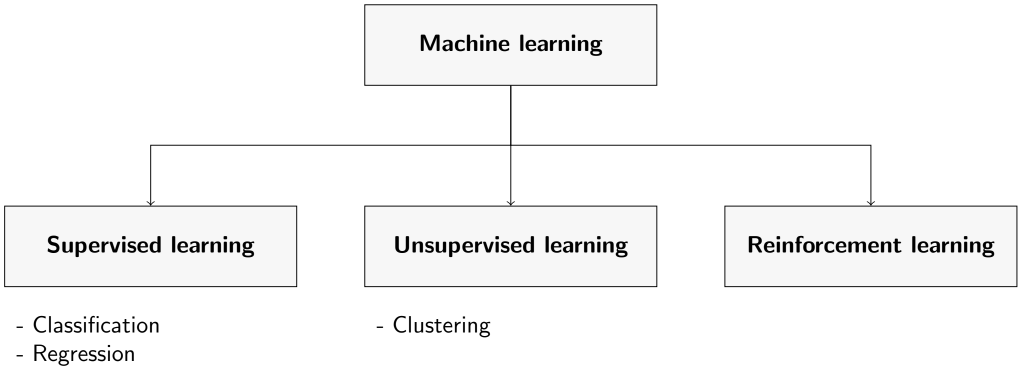


Figure 2.6: Machine learning problem can be classified as either supervised learning, unsupervised learning, or reinforcement learning.

the inputs $\boldsymbol{\eta}$ and outputs $\boldsymbol{\delta}$. The process of constructing this mapping is referred to as **training** and evaluating the built mapping or model for a given input is referred to as **testing**. Usually the input $\boldsymbol{\eta}_i$ is a vector of fixed size; they are also termed as features or attributes, and the output δ_i is a scalar. When δ_i can take values from a fixed discrete set, the problem is known as classification. Facial recognition and spam filters for email are simple examples of the classification problem. In the turbulence modeling setting Ling et al.[55] used classification algorithms and DNS data to mark region of uncertainty in a RANS simulation. When δ_i can be any real number, the problem is termed as regression. Predicting the age of a viewer watching a youtube video is an example of regression, and so is predicting turbulent kinetic energy in a fluid flow.

Unsupervised learning, on the other hand, considers no information about the output. It utilizes the input data $\mathfrak{D} = \{\boldsymbol{\eta}_i\}_{i=1}^{N}$ to extract useful information about the data. A simple example of unsupervised learning is the clustering of e-commerce users into groups based on their purchasing habits, which can then be used for targeted advertising. In turbulence, unsupervised learning can be used to categorize flows based on their characteristics. Reinforcement learning constitutes a set of methods to build models which can adapt themselves to a given situation.

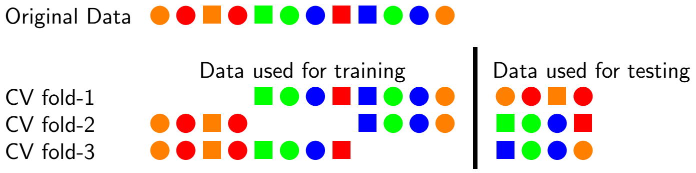


Figure 2.7: Schematic describing the process of cross-validation (CV). The example uses a 3-fold CV. The figure is adapted from `https://tex.stackexchange.com/a/154121`.

This work focuses on supervised learning, specifically regression algorithms on pairs of flow features and the model discrepancy, which is derived from the inverse solution. The goal is to build a generalized regression model of the discrepancy. The following sections discuss the various aspects of machine learning in the context of the current work.

2.3.1 Problem Setup

The inverse approach presented in the previous section results in an optimal spatial discrepancy field for a given problem setup. In predictive modeling, the problem-specific information encoded in $\boldsymbol{\delta}_1(\mathbf{x_1}), \boldsymbol{\delta}_2(\mathbf{x_2}), \ldots, \boldsymbol{\delta}_n(\mathbf{x_n})$ must be transformed into modeling knowledge. This is done by extracting the functional relationship $\boldsymbol{\delta}(\boldsymbol{\eta})$, where $\boldsymbol{\eta} = [\eta^1, \eta^2, \ldots, \eta^m]$ are input features derived from the solution. The functional relationship must be developed by considering the output of many inverse problems representative of the modeling deficiencies relevant to the predictive problem. Furthermore, as will be explained in the following, elements of the feature vector $\boldsymbol{\eta}$ are chosen to be locally nondimensional quantities such that the functional relationship $\boldsymbol{\delta}(\boldsymbol{\eta})$ is useful for different problems in which the flow-features are realizable.

2.3.2 Feature Selection

The input features $\boldsymbol{\eta}_i$ plays a crucial role in any machine learning process; they should be carefully selected to represent all the major traits of the data. Two desired properties in the selected features are their locality and non-dimensionality. These allow the use of learned discrepancy in a general setting, for example, use of a different geometry or Reynolds number than that used for training. The features should be able to characterize the discrepancy function and at the same time they should be kept to a minimal number to reduce the cost associated with training and prediction. Standard practice is to input a large number of features and use existing algorithms to rank their importance using algorithms[10]. However, using a large number of features increases the requirements on the number of data points required for training without over-fitting. We have also observed that automatic selection of features can be misleading. For instance, in a trial utilizing over 20 different flow-features including density, velocity gradient, strain rate and vorticity, a random forest-based regressor ranks density to be the most critical feature. Therefore, we use domain knowledge to select the features. Examples of such relevant features include appropriately non-dimensionalized strain-rate, vorticity, wall-distance, and eddy-viscosity.

2.3.3 Feature Normalization

Normalization of the features contained in $\boldsymbol{\eta}$ is required when there are differences in their orders of magnitude, which is often the case with flow features used in turbulence. Differences in magnitude may result in poor or slow training[44]. The features are normalized before training using

$$\eta^j_{normal} = \frac{\eta^j - \overline{\eta^j}}{\sigma_{\eta^j}} \tag{2.25}$$

where, $\overline{\eta^j}$ and σ_{η^j} are the mean and the standard deviation, respectively, of the j'th component of the flow-feature vector $\boldsymbol{\eta}$. The same normalization process is used for the testing data (features) when querying a trained ML model.

2.3.4 Cross-Validation

An important consideration when training a machine learning model is the tendency of highly flexible regression models to over-fit the data. As we do not have an infinite amount of data covering all the possible scenarios, it is possible for the model to be very accurate for the data used for training, while it may be poor in generalizing to other inputs not used for training. While some degree of over-fitting is unavoidable, its effects can be mitigated by following some simple strategies. A popular approach is to use cross-validation (CV)[48]. In CV, the training data is split into K folds, then for each $k \in \{1, \ldots, K\}$, the model is trained on all the folds except k'th and tested on the k'th. Fig 2.7 demonstrates the strategy graphically. The final error is an average of the errors over all the folds. Also, the prediction for any data point is taken to be the average of predictions of all the K ensembles of models. CV is also used to select optimal values for the hyper-parameters associated with an algorithm.

2.3.5 Measure of Training Quality

The quality of a trained ML model can be measured in a number of ways. In this work, we use coefficient of determination, which is also represented by R^2. Given a set of truth values $\{\delta_{1,true}, \delta_{2,true}, \ldots, \delta_{n,true}\}$ and corresponding predictions by the machine learning model $\{\delta_{1,pred}, \delta_{2,pred}, \ldots, \delta_{n,pred}\}$, the coefficient of determination is given by:

$$R^2 \equiv 1 - \frac{SS_{\text{res}}}{SS_{\text{tot}}} \tag{2.26}$$

where SS_{tot} is proportional to the variance of the truth values and is given by:

$$SS_{\text{tot}} = \sum_i (\delta_{i,true} - \bar{y})^2 \tag{2.27}$$

$$\bar{y} = \frac{1}{n} \sum_i \delta_{i,true} \tag{2.28}$$

and SS_{res} is sum of squares of residuals

$$SS_{\text{res}} = \sum_i (\delta_{i,true} - \delta_{i,pred})^2. \tag{2.29}$$

A perfect machine learning mode will result in $R^2 = 1$. The R^2 is a measure of quality of the ML model but whether a model is acceptable for a certain R^2 requires human judgement. Most of the results presented in this work have $R^2 > 0.9$.

2.3.6 Regression Algorithms

The algorithms to extract the relationship between flow features and the discrepancies can be as simple as linear regression. More sophisticated algorithms exist to extract the relationship. The specific algorithm selected depends on many factors including the distribution of data and the scales in the data. There is no rigorous method to determine a suitable algorithm a priori. However, they can be evaluated based on properties such as scalability, number of hyper-parameters, the cost of evaluation and ease of training.

The objective of this work is not to compare different algorithms but to use them to derive functional relationships between model discrepancies and non-dimensional flow features. The algorithms used for this work are discussed in the following sections. Other algorithms explored but not discussed here include Random Forest Regression[10] and Gaussian Processes[81].

2.3.6.1 Neural Network

The standard NN algorithm operates by constructing linear combinations of inputs and transforming them through nonlinear activation functions. The process is repeated once for the input layer and each hidden layer (marked in blue in Fig. 2.8) in the network, until the output layer is reached. Fig. 2.8 presents a sample NN.

For this sample network, the values of the hidden nodes $z_{1,1}$ through z_{1,H_1} would be constructed as

$$z_{1,j} = a_{(1)} \left(\sum_{i=1}^{3} w_{ij}^{(1)} \eta^i \right) \tag{2.30}$$

where $a_{(1)}$ and $w_{ij}^{(1)}$ are the activation function and weights associated with the first hidden layer, respectively. Similarly, the second layer of hidden nodes is constructed as

$$z_{2,j} = a_{(2)} \left(\sum_{i=1}^{H_1} w_{ij}^{(2)} z_{1,i} \right) \tag{2.31}$$

Finally, the output is given by:

$$\boldsymbol{\delta} \approx f(\boldsymbol{\eta}) = a_{(3)} \left(\sum_{i=1}^{H_2} w_{ij}^{(3)} z_{2,i} \right). \tag{2.32}$$

Some common activation functions includes the sigmoid function $a(x) = 1/(1 + e^{-x})$, hyperbolic tangent $a(x) = \tanh(x)$, and ReLU $a(x) = \max(0, x)$.

Algorithm 1 Training a Neural Network using Simple Backpropagation algorithm

```
Inputs: $\eta_1, \eta_2, \ldots, \eta_n$
Desired Outputs: $\delta_1, \delta_2, \ldots, \delta_n$
Initialize: $w_{ij}^k$ with small random values
Error function: $E(\delta_{prediction}, \delta)$

    for $t \leftarrow 1$ to $T$ do
        Evaluate $\delta_{prediction}$ (equation 2.32) and $E(\delta_{prediction}, \delta)$
        Evaluate $\frac{\partial E}{\partial w_{ij}^k}$
        $w_{ij}^k(t+1) = w_{ij}^k(t) - \eta \frac{\partial E}{\partial w_{ij}^k}$
    end for
```

A simple approach to train a neural network is the use of a gradient-based optimization method to estimate the model weights, $w_{ij}^{(n)}$, by minimizing differences between prediction and data (as shown in Algorithm 1). However, more sophisticated strategies are required to handle a large amount of data and to prevent overfitting of the model.

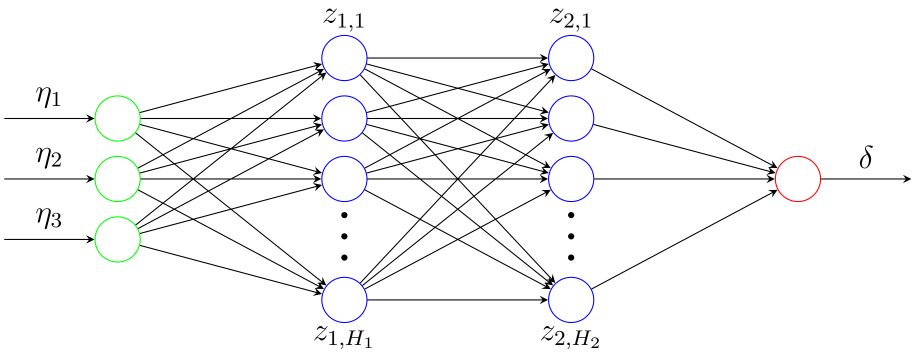


Figure 2.8: Network diagram for a feed-forward NN with three inputs, two hidden layers, and one output.

Once the weights are found, computing the output depends only on the number of hidden nodes, and not on the size of the training data. Hyperparameters of the NN method include the number of hidden layers, the number of nodes in each hidden layer, and the forms of the activation functions. The fast artificial neural network (FANN) [67] is used for this work. This library uses resilient back-propagation (RPROP) [59]. Typically, four layers (including the input and output layers) and 128 neurons (total) were employed with a sigmoid activation function.

2.3.6.2 AdaBoost

In the context of machine learning, boosting refers to the process of combining multiple *weak learners* to create a single *strong learner*. Neural-networks (NN) are classified as strong learners, whereas simple techniques which are only better than a random prediction are classified as weak learners. Examples include algorithms based on thresholds (like algorithms based on decision trees).

AdaBoost [36, 21] works by training weak learners for different sets of training data. Weights are assigned to individual training data points and updated to reflect the goodness of the fit. The initial weights are the same for all the data points. The training set is then selected from the training data, with a probability of selection being proportional to the weights. The data points with low weights are used less than the data points with higher weights. Hence, in the current iteration, the model will improve for data points with high weights. A weak learner is trained on the training set and added to the model from the previous iteration with the weight of the weak learner calculated by the error in prediction by the weak learner. The weights for the data points–for which the model performed well–are then reduced. The quality of the trained model is measured by an appropriate norm of error. The training steps are listed in Algorithm 2.

Hyperparameters for the AdaBoost method include the number of weak learners, the type of weak learner, and the measure of error (loss function). Typically, 1000 decision tree based weak learners are used with a maximum depth of 24. A square loss function is used to measure the error. A Python based library scikit-learn[73] is used to train and test the AdaBoost algorithm.

Algorithm 2 Training AdaBoost model

```
Inputs: $\eta_1, \eta_2, \ldots, \eta_n$
Desired Outputs: $\delta_1, \delta_2, \ldots, \delta_n$
Initial Weights: $w_{1,1}, w_{2,1}, \ldots, w_{n,1}$
Error Function: $E(f(x), y, i) = e^{-\delta_i f(\eta_i)}$
Weak Learners: $h\colon x \to [-1, 1]$

    for $t \leftarrow 1$ to $T$ do
        1. Find weak learner $h_t(x)$ that minimizes $\epsilon_t = \sum_{\substack{i=1 \\ h_t(\eta_i) \neq \delta_i}}^{n} w_{i,t}$
        2. Choose $\alpha_t = \frac{1}{2} \ln \left( \frac{1 - \epsilon_t}{\epsilon_t} \right)$
        3. Add to ensemble $F_t(x) = F_{t-1}(x) + \alpha_t h_t(x)$
        4. Update weights $w_{i,t+1} = w_{i,t} e^{-\delta_i \alpha_t h_t(\eta_i)}$
        5. Renormalize $w_{i,t+1}$ such that $\sum_i w_{i,t+1} = 1$
    end for
```

2.4 Embedding and Prediction

Once the machine learning model is trained on the output of the inverse problems, the augmented model has the following form,

$$\frac{D\mathbf{G}}{Dt} = T_1(\mathbf{Q}, \mathbf{G}) + T_2(\mathbf{Q}, \mathbf{G}) + T_3(\mathbf{Q}, \mathbf{G}) + \boldsymbol{\delta}(\boldsymbol{\eta}(\mathbf{Q}, \mathbf{G})). \tag{2.33}$$

The last term in the equation 2.33 involves a call to the machine learning evaluation. Except for the last term being non-analytic, there is no difference between the form of the augmented model and the baseline model. This model can be used in place of the baseline model for prediction for any setup not used for training. The quality of the prediction will depend on how well the training data represents the test flow unless there is a universal correction term $\boldsymbol{\delta}$ and we are able to recover that correction. Whether such a universal correction exists is a philosophical question in the field of turbulence. In this work, we use the augmented model for predictions on setups similar to the ones used for training, and show that it is possible to improve the model prediction for such cases.

2.5 Summary

This section summarizes the material presented in the chapter.

1. The nature of discrepancies found on physical models was discussed. The discrepancies are functional, and hence any model augmentation has to augment the functional form of existing models; calibration of model constants are not sufficient.

2. A general method of introducing functional discrepancy in existing models was discussed and the framework of FIML to discover such discrepancy was presented. The framework consists of two principal steps. First, data is used

to infer the spatial distribution of the discrepancy for many configurations by solving inverse problems. The inverse solutions are then used to construct a generalized regression model using machine learning.

3. Bayesian and deterministic approaches to frame and solve inverse problems were discussed. The optimization problem encountered in the solution of inverse problems was presented, along with discussion of the discrete adjoints required for sensitivity calculation.

4. Finally, various facets of training a machine learning algorithm were discussed followed by description of the two algorithms used for this work: neural-network (NN) and AdaBoost.

CHAPTER III

Numerics

The compressible form of the Reynolds-averaged Navier-Stokes equations are discretized using unstructured finite-volume based schemes[72, 116, 22, 25, 111]. This chapter presents details about the spatial and temporal discretization. Implementation-related aspects of the discrete adjoint framework are discussed.

3.1 Compressible RANS Equations

In Chapter I, the incompressible Navier-Stokes equations were discussed for simplicity. Fluid flow is inherently compressible and the effects of compressibility increase with the flow Mach number. Therefore, we solve the compressible form of the NS equations. The compressible form of the Reynolds-averaged Navier-Stokes equations is given by,

$$\frac{\partial \mathbf{U}}{\partial t} + \frac{\partial \mathbf{F_i}}{\partial x} + \frac{\partial \mathbf{G_i}}{\partial y} + \frac{\partial \mathbf{H_i}}{\partial z} = \frac{\partial \mathbf{F_v}}{\partial x} + \frac{\partial \mathbf{G_v}}{\partial y} + \frac{\partial \mathbf{H_v}}{\partial z} + \mathbf{S}, \tag{3.1}$$

where, $\mathbf{U}$ is the vector of conserved flow variables. $\mathbf{F_i}$, $\mathbf{G_i}$, $\mathbf{H_i}$ are the vectors representing inviscid fluxes and $\mathbf{F_v}$, $\mathbf{G_v}$, and $\mathbf{H_v}$ are the vectors representing the viscous fluxes. **S** is the source term vector. The vector **U** is given by,

$$\mathbf{U} = \left\{ \begin{matrix} \rho \\ \rho u \\ \rho v \\ \rho w \\ e \\ \rho \Phi \end{matrix} \right\}, \tag{3.2}$$

where, ρ is the density, u, v, w, are the three components of velocity, e is the total energy per unit volume, and Φ is a vector of operational variables used in the turbulence model. Pressure p is calculated using the following constitutive relation,

$$e = \frac{p}{\gamma - 1} + \frac{1}{2}\rho\left(u^2 + v^2 + w^2\right), \tag{3.3}$$

where, γ is the ratio of the heat capacities at constant pressure and volume. The inviscid and the viscous fluxes are given by,

$$\mathbf{F_i} = \left\{ \begin{matrix} \rho u \\ \rho u^2 + p \\ \rho uv \\ \rho uw \\ u(e+p) \\ \rho u \Phi \end{matrix} \right\}, \tag{3.4}$$

$$\mathbf{G_i} = \left\{ \begin{array}{c} \rho v \\ \rho v u \\ \rho v^2 + p \\ \rho v w \\ v(e+p) \\ \rho v \Phi \end{array} \right\}, \tag{3.5}$$

$$\mathbf{H_i} = \left\{ \begin{array}{c} \rho w \\ \rho w u \\ \rho w v \\ \rho w^2 + p \\ w(e+p) \\ \rho w \Phi \end{array} \right\}, \tag{3.6}$$

$$\mathbf{F_v} = \left\{ \begin{array}{c} 0 \\ \tau_{xx} \\ \tau_{yx} \\ \tau_{zx} \\ u\tau_{xx} + v\tau_{yx} + w\tau_{zx} - k\frac{\partial T}{\partial x} \\ \rho \Phi'_x \end{array} \right\}, \tag{3.7}$$

$$\mathbf{G_v} = \left\{ \begin{array}{c} 0 \\ \tau_{xy} \\ \tau_{yy} \\ \tau_{zy} \\ u\tau_{xy} + v\tau_{yy} + w\tau_{zy} - k\frac{\partial T}{\partial y} \\ \rho \Phi'_y \end{array} \right\}, \tag{3.8}$$

$$\mathbf{H_v} = \begin{Bmatrix} 0 \\ \tau_{xz} \\ \tau_{yz} \\ \tau_{zz} \\ u\tau_{xz} + v\tau_{yz} + w\tau_{zz} - k\frac{\partial T}{\partial z} \\ \rho\Phi'_z \end{Bmatrix}. \tag{3.9}$$

Following the Newtonian fluid assumption and the Boussinesq approximation for turbulence modeling, the stress tensor (τ_{ij}) is given by,

$$\tau_{ij} = 2\left(\mu + \mu_t\right)\left[\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right) - \frac{1}{3}\frac{\partial u_k}{\partial x_k}\delta_{ij}\right], \tag{3.10}$$

where, μ and μ_t are the laminar and turbulent viscosity respectively. The temperature dependence of the laminar viscosity is modeled using the power law,

$$\mu = \mu_{ref}\left(\frac{T}{T_{ref}}\right)^{2/3}, \tag{3.11}$$

where the reference conditions are specified as a part of the problem setup. Operational variables contained in Φ are used to evaluate the turbulent viscosity. The coefficient of the thermal conductivity is related to the viscosity using laminar and turbulent Prandtl numbers,

$$k = \frac{\mu C_p}{Pr} + \frac{\mu_t C_p}{Pr_t}. \tag{3.12}$$

$Pr = 0.72$ and $Pr_t = 0.9$ are used for this work. Diffusive terms for the turbulence model are of the form,

$$\Phi'_x = \frac{1}{\rho}\mu_{eff}\frac{\partial \Phi}{\partial x}, \Phi'_y = \frac{1}{\rho}\mu_{eff}\frac{\partial \Phi}{\partial y}, \Phi'_z = \frac{1}{\rho}\mu_{eff}\frac{\partial \Phi}{\partial z}. \tag{3.13}$$

The expression of μ_{eff} depends on the specific model. The RANS equations are closed using the ideal gas law,

$$p = \rho RT,$$

where R is the gas constant. No body-forces are assumed, therefore, the source term is nonzero only for the turbulence model and is given by,

$$\mathbf{S} = \begin{Bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ S_\Phi \end{Bmatrix}, \tag{3.14}$$

where S_Φ is the source term associated with the turbulence model.

3.2 Spatial Discretization

The governing equations are discretized in space using cell-centered finite-volume schemes. The governing equations are integrated over a cell-volume and the divergence theorem is applied to transform the volume integral of the flux gradients to a surface integral of fluxes. This results in the following discretized form,

$$\frac{\partial \mathbf{U}}{\partial t} + \frac{1}{V}\sum_f (\mathcal{F}_i(\mathbf{U}) - \mathcal{F}_v(\mathbf{U}))A_f + \mathbf{S} = 0, \tag{3.15}$$

where, V is the cell-volume and $\mathbf{U}$ is averaged over the cell volume and defined at the cell center. $\mathcal{F}_i$ and $\mathcal{F}_v$ are the projections of the inviscid and viscous flux vectors, respectively, on the area vector of the face f with an area of A_f. $\mathbf{S}_i$ is the source term averaged over the cell volume. The governing equation can be written in the following form,

$$\frac{\partial \mathbf{U}_i}{\partial t} = -\mathbf{R}(\mathbf{U}), \tag{3.16}$$

with the residual $\mathbf{R}(\mathbf{U})$ defined as,

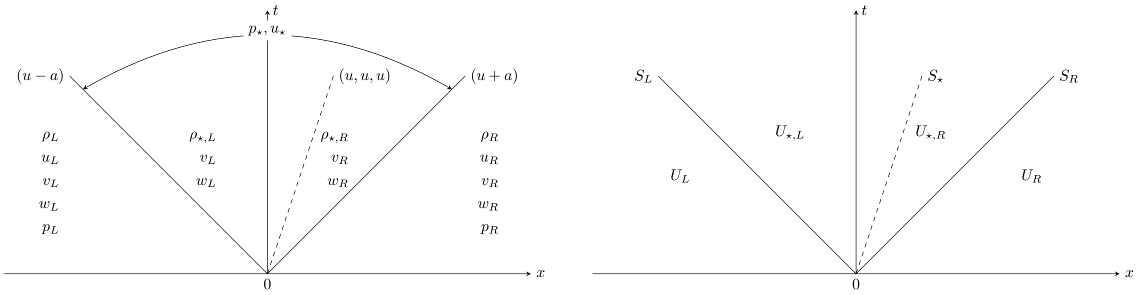
$$\mathbf{R}(\mathbf{U}) = -\frac{1}{V}\sum_f (\mathcal{F}_i(\mathbf{U}) - \mathcal{F}_v(\mathbf{U}))A_f - \mathbf{S}. \tag{3.17}$$

The fluxes are evaluated at the faces using the solution $\mathbf{U}$ defined at the cell-centers. The inviscid and viscous fluxes are evaluated differently because of the difference in the physical and mathematical nature of these fluxes.

3.2.1 Calculation of the Inviscid Fluxes

The inviscid fluxes are calculated by solving a Riemann problem at the cell interface. Given a left (subscript L) and a right (subscript R) solution state at $t = 0$, a Riemann problem involves evaluating the solution at the interface for $t > 0$. The Riemann problem has an exact solution and requires solving a non-linear algebraic equation using fixed-point iteration. In practice, the exact solution is expensive as the Riemann problem is solved at every cell interface. Therefore, approximate Riemann solvers are used, which are based on approximations of the exact eigenstructure of the interface solution.

Harten-Lax-van Leer-Contact (HLLC)[104] approximate Riemann solver is used for this work . Fig. 3.1(a) shows the structure of the exact solution of a Riemann problem with the interface at $x = 0$. Fig. 3.1(b) shows the structure for the approximate HLLC flux. The inviscid flux at the interface in HLLC is given by,



t
$p_\star, u_\star$
$(u-a)$
(u,u,u)
$(u+a)$
ρ_L u_L v_L w_L p_L
$\rho_{\star,L}$ v_L w_L
$\rho_{\star,R}$ v_R w_R
ρ_R u_R v_R w_R p_R
0
x
t
S_L
$S_\star$
S_R
$U_{\star,L}$
$U_{\star,R}$
U_L
U_R
0
x

(a) Real

(b) HLLC

Figure 3.1: Characteristics of the actual Euler equations and the HLLC approximation.

$$\mathcal{F}^{HLLC} = \begin{cases} \mathcal{F}_L & 0 \leq S_L \\ \mathcal{F}_{\star L} = \mathcal{F}_L + S_L(\mathbf{U}_{\star L} - \mathbf{U}_L) & S_L \leq 0 \leq S_\star \\ \mathcal{F}_{\star R} = \mathcal{F}_R + S_R(\mathbf{U}_{\star R} - \mathbf{U}_R) & S_\star \leq 0 \leq S_R \\ \mathcal{F}_R & 0 \geq S_R \end{cases} \tag{3.18}$$

where $\mathcal{F}_L$ and $\mathcal{F}_R$ are evaluated using the left $\mathbf{U}_L$ and right $\mathbf{U}_R$ state respectively. $S_\star$ is an approximation to the velocity of the contact discontinuity. $\mathbf{U}_{\star,L}$ and $\mathbf{U}_{\star,R}$ are the solutions to the left and right of the contact discontinuity given by,

$$\mathbf{U}_{\star K} = \rho_K \left(\frac{S_K - u_k}{S_K - S_\star}\right) \left\{ \begin{array}{c} 1 \\ S_\star \\ v_K \\ w_K \\ \frac{e_K}{\rho_K} + (S_\star - u_K)\left[S_\star + \frac{p_K}{\rho_K(S_K - u_K)}\right] \\ \Phi_K \end{array} \right\} \tag{3.19}$$

for $K = L$ or R. The velocity of the contact discontinuity is given by,

$$S_\star = \frac{p_R - p_L + \rho_L u_L(S_L - u_L) - \rho_R u_R(S_R - u_R)}{\rho_L(S_L - u_L) - \rho_R(S_R - u_R)}. \tag{3.20}$$

The fluxes are evaluated given the left and right wave speeds, which are calculated using the Roe averaging of the left and the right state. The expressions for the wave-speeds are given by,

$$S_L = \min\{\tilde{u} - \tilde{a}, u_L - a_L\}, \tag{3.21}$$

$$S_R = \min\{\tilde{u} + \tilde{a}, u_R + a_R\}. \tag{3.22}$$

The Roe average quantities are evaluated using the following relations,

$$\tilde{u} = \frac{\sqrt{\rho_R}u_R + \sqrt{\rho_L}u_L}{\sqrt{\rho_R} + \sqrt{\rho_L}}, \tag{3.23}$$

$$h_K = \frac{\gamma p_K}{\rho_K} + \frac{\gamma - 1}{2}u_K^2, \tag{3.24}$$

,

$$\tilde{h} = \frac{\sqrt{\rho_R}h_R + \sqrt{\rho_L}h_L}{\sqrt{\rho_R} + \sqrt{\rho_L}}, \tag{3.25}$$

$$\tilde{a} = \sqrt{\tilde{h} - \frac{\gamma - 1}{2}\tilde{u}^2}. \tag{3.26}$$

The accuracy of the inviscid flux given by equation 3.18 is governed by the definition of the left and the right state. A first-order accuracy is obtained by setting the left and right state as the solution at the left and the right cell. Higher order accuracy can be achieved by using information of the solution gradients and solution at the neighboring cells. A second order approximation is obtained by using the following reconstruction,

$$\phi_f^K = \phi_K + \psi_f \nabla\phi|_K \cdot \mathbf{r}_f \tag{3.27}$$

for $K = L$ or R, ϕ_K and $\nabla\phi|_K$ contains the solution and its gradient on the K^{th} side of the interface. $\mathbf{r}_f$ is a vector from the center of the cell to the center of the face. A limiter ψ_p preserves the monotonicity of the reconstruction to avoid spurious oscillations in the solution. The limiter used for this work[108] is given by,

$$\psi_f = \begin{cases} \psi\left(\frac{\delta_+}{\delta_-}\right), \delta_+ = \phi_{max} - \phi_p & \phi_f > \phi_p \\ \psi\left(\frac{\delta_+}{\delta_-}\right), \delta_+ = \phi_{min} - \phi_p & \phi_f < \phi_p \\ 1 & \phi_f = \phi_p \end{cases}, \tag{3.28}$$

where $\delta_- = \phi_f - \phi_p$, and, ϕ_f is the interpolated value of the solution at the face f.

$$\psi\left(\frac{\delta_+}{\delta_-}\right) = \frac{\frac{\delta_+^2}{\delta_-^2} + \frac{2\delta_+}{\delta_-} + \frac{\epsilon^2}{\delta_-^2}}{\frac{\delta_+^2}{\delta_-^2} + \frac{\delta_+}{\delta_-} + 2 + \frac{\epsilon^2}{\delta_-^2}}, \tag{3.29}$$

$$\epsilon = K\phi_{ref}, \tag{3.30}$$

with $K = 0.01$. The required gradients at the cell centers are calculated using the Green-Gauss theorem, which results in the follow expression,

$$\nabla\phi = \frac{1}{V}\sum_{faces_i} \phi_f \vec{\mathbf{A}}_f. \tag{3.31}$$

3.2.2 Calculation of the Viscous Fluxes

The viscous fluxes do not require special treatment because of their elliptic nature. Their evaluation requires the projection of gradient of the solution on the face area-vector, which is calculated using the following expression[47],

$$\nabla\phi|_f \cdot \mathbf{A}_f = \frac{\phi_{nbr} - \phi_p}{|\mathbf{x}_{nbr} - \mathbf{x}_p|} + \frac{1}{2}\left(\nabla\phi|_p + \nabla\phi|_{nbr}\right) \cdot \left(\mathbf{A}_f - \alpha_f \mathbf{s}_f\right) \tag{3.32}$$

where, $\mathbf{s}_f$ is a vector connecting cell center and the center of the face f and $\alpha_f = \mathbf{s}_f \cdot \mathbf{A}_f$. Neighbor ($nbr$) is the cell which shares the face f with cell p. $\mathbf{x}$'s are the cell center coordinates.

3.3 Temporal Discretization

In the semi-discrete form, the governing equation can be written as,

$$\frac{d\mathbf{U}}{dt} = -\mathbf{R}(\mathbf{U}). \tag{3.33}$$

While the evaluation of the residual $\mathbf{R}(\mathbf{U})$ is discussed in the previous section, this section deals with the scheme to evolve the solution in time. A time-marching scheme is explicit when the residual is evaluated at a known time instant. A simple time-marching scheme, forward-Euler, is given by,

$$\mathbf{U}^{n+1} = \mathbf{U}^n - \mathbf{R}(\mathbf{U}^n)\Delta t. \tag{3.34}$$

The forward-Euler has a very limited stability condition which restricts the maximum time step Δt. All the problems in this thesis are steady, therefore we use an implicit scheme which can support large time steps. A simple implicit time-marching scheme, backward-Euler, is given by,

$$\mathbf{U}^{n+1} = \mathbf{U}^n - \mathbf{R}(\mathbf{U}^{n+1})\Delta t. \tag{3.35}$$

The residual $\mathbf{R}(\mathbf{U}^{n+1})$ cannot be calculated directly as $\mathbf{U}^{n+1}$ is not known. It is therefore approximate using a Taylor series expansion around $\mathbf{U}^n$, which leads to the following expression,

$$\mathbf{U}^{n+1} = \mathbf{U}^n - \left(\mathbf{R}(\mathbf{U}^n) + \frac{\partial \mathbf{R}}{\partial \mathbf{U}} |_{\mathbf{U}^n} \Delta\mathbf{U} + \mathcal{O}(\Delta\mathbf{U}^2)\right) \Delta t. \tag{3.36}$$

Neglecting the second order terms, the expression can be simplified as,

$$\left(\frac{\mathbf{I}}{\Delta t} + \frac{\partial \mathbf{R}}{\partial \mathbf{U}}\right) \Delta \mathbf{U} = -\mathbf{R}(\mathbf{U}^n) \tag{3.37}$$

where $\mathbf{I}$ is an identity matrix and $\Delta \mathbf{U} = \mathbf{U}^{n+1} - \mathbf{U}^n$. Note that calculating $\Delta \mathbf{U}$ requires solving a sparse linear system and we need the Jacobian of the residual to build this. The inviscid and viscous flux Jacobians are evaluated exactly[72] while only the diagonal components of the Jacobian of source terms are used because of their complex nature.

Linear Solver

The *Portable Extensible Toolkit for Scientific Computation (PETSc)*[3] linear algebra library is used to solve the linear systems involved with the backward-Euler time integration and also to solve the discrete adjoint equation (3.38). PETSc provides a comprehensive set of tools for solving linear and non-linear problems. In most cases, a generalized minimal residual method (GMRES)[85] with block-Jacobi preconditioner is used. In cases with very high Jacobian matrix condition numbers, ILU or LU based preconditioners are used. Details about these algorithms can be found in [84].

3.4 Turbulence Models

A broad set of RANS based turbulence models have been implemented and used in our in-house solver. This list includes one equation: Spalart-Allmaras[99], two equation Wilcox's $k-\omega$[113], $k-\epsilon$, $k-\omega$ SST[63], Menter's $\gamma - Re_\theta$ based transition model[64], and the Wilcox full Reynolds stress closure stress-ω[113] model. While the framework used in this chapter has been utilized to enhance many turbulence models, the results presented in this thesis involves SA and Wilcox's $k-\omega$. Appendix

A contains the mathematical formulation of these two models.

3.5 Discrete Adjoint

Solution to an inverse problem requires calculation of the sensitivity of the objective function J with respect to the introduced discrepancy. As described in section 2.2.2.1, discrete adjoints[37] are used to evaluate these gradients. Calculation of discrete adjoints requires solution of the following linear system,

$$\left[\frac{\partial \mathbf{R}}{\partial \mathbf{U}}\right]^T \Psi = -\left[\frac{\partial J}{\partial \mathbf{U}}\right]^T. \tag{3.38}$$

where, Ψ is the adjoint variable, note that the dimensions of Ψ are same as that of $\mathbf{U}$. The exact Jacobian of the residual vector and gradient of the objective function is required in the adjoint equation. Once these are known, the linear system is solved using iterative methods. The presence of second-order reconstruction, flux vector splitting, and non-linear source terms in turbulence models, make the analytical evaluation of the Jacobian a laborious and error-prone task. In the following sections, other techniques of evaluating the Jacobian are discussed.

3.5.1 Finite Difference

In the finite difference formulation, partial derivatives involved in the Jacobian are evaluated by perturbing an element of the solution vector by a small value ϵ. One column of the Jacobian matrix using a forward difference formulation has the following form,

$$\frac{\partial \mathbf{R}}{\partial \mathbf{U}} \mid_{,j} = \frac{\mathbf{R}(\mathbf{U} + \epsilon e_j) - \mathbf{R}(\mathbf{U})}{\epsilon} \tag{3.39}$$

where e_j is a vector containing zeros except for the j'th element, which is one. The cost of a finite difference approximation is proportional to the size of the input vector $\mathbf{U}$. Theoretically, the error in the finite difference approximation reduces with ϵ.

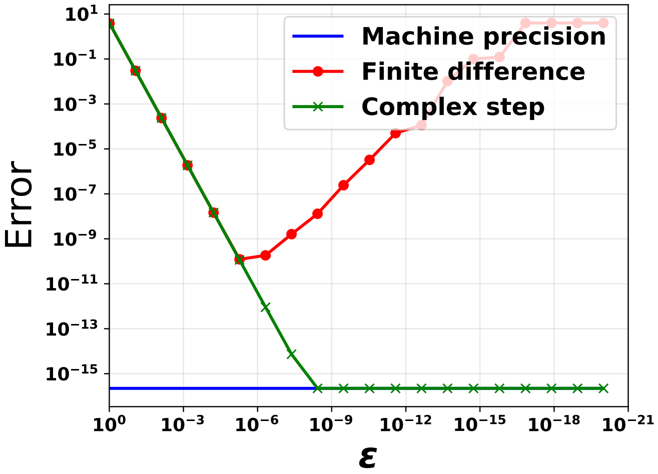


Figure 3.2: Error in the derivative calculation for a test function function $f(x) = x^4$ at $x = 1$.

However, because of the finite precision of the floating point representation, the finite difference approximation suffers from an increase in error with reduction in ϵ below a threshold value of ϵ. Fig. 3.2 shows the error for a simple test function. In cases where accuracy upto the machine precision is important, finite difference should be avoided.

3.5.2 Complex-Step Differentiation

In a complex-step differentiation[61], the function is evaluated by perturbing the solution in the imaginary space. j'th column of the Jacobian matrix is given by,

$$\frac{\partial \mathbf{R}}{\partial \mathbf{U}} |_{,j} = \mathbb{I}\mathrm{m}\left[\frac{\mathbf{R}(\mathbf{U} + \iota \epsilon e_j)}{\epsilon}\right] \tag{3.40}$$

where $\iota = \sqrt{-1}$, and $\mathbb{I}\mathrm{m}[x]$ returns the imaginary part of x. The formulation is derived by using a Taylor series expansion of $\mathbf{R}(\mathbf{U} + \iota \epsilon e_j)$ around $\mathbf{U}$. This form of approximation does not suffer from the cancellation error and therefore the error reduces with decreasing ϵ to machine precision, as shown in Fig. 3.2. Implementation of the complex-step differentiation requires the computer code to be compatible with complex-number arithmetic, which may be easy or difficult based on the programming language and architecture of the code.

3.5.3 Automatic Differentiation

In automatic differentiation[80] or algorithmic differentiation, the chain rule of differentiation is applied recursively to evaluate the derivatives. The chain rule can be applied in a forward or a backward manner resulting in two different modes. The derivatives obtained using automatic differentiation are exact as it does not involve any approximation and are precise up to the numerical precision. Because of its systematic nature, the process of calculating derivatives using algorithmic-differentiation can be automated. Various tools exist to calculate gradients and the Jacobian of a general non-linear function. The tools either generate a transformed source code[42, 6] or use operator overloading[40] to calculate the gradients. We use an operator overloading based tool, ADOL-C[40], to calculate the exact Jacobian of the residual and other partial derivatives required for the discrete adjoints.

3.6 Summary

- The compressible RANS equations are solved using a finite-volume based discretization on unstructured meshes. HLLC approximate Riemann solver is used to evaluate the inviscid flux using a second order gradient based reconstruction.
- Implicit backward-Euler temporal scheme is used to obtain the steady solution of the RANS equations. A linear algebra library PETSc is used to solve the linear systems using iterative GMRES algorithm.
- Discrete adjoints are used for the calculation of the sensitivities of the objective function to the design variables. The discrete adjoint framework is based on an

automatic-differential library, ADOL-C, which is used to calculate the required intermediate partial derivatives exactly.

CHAPTER IV

Proof-of-concept of Data-driven Turbulence Modeling

This chapter presents a proof of concept study of the efficacy of the field-inversion and machine-learning (FIML) framework. The framework is used with a synthetically generated data-sets from a new model and thus there is virtually no uncertainty in the data or the flow setup. The augmented model is then used for prediction for the same flow-setup used for training. This chapter tests the framework under a best-case scenario and attempts to establish a benchmark verification case for data-driven turbulence modeling, while the next two chapters focus on more realistic flow problems.

We consider an internal flow in a planar channel with a concave curvature. The objective is to augment the Spalart-Allmaras (SA)[99] model using synthetically generated data. The data is generated using an analytic rotation curvature correction SA–RC[100, 88] to the SA model. There are minimal uncertainties as the solver, mesh, and flow conditions are identical for both the data generation and the FIML steps. In most practical situations, measurement noise and inconsistencies are always present due to uncertainties associated with geometry, discretization, governing equations, boundary conditions, etc. Additionally, as the analytic form of the correction is known, parallels can be drawn between the FIML results and the analytic correction.

As the data is synthetically generated, full-field profiles of velocity, eddy-viscosity, and Reynolds stresses are available which can be used for model augmentation and evaluation. We compare the inverse solutions obtained using the full-field velocity to the ones obtained using only the skin friction and pressure. Multiple ML models are trained using different inverse solutions and combinations of local flow features, and are compared in a predictive setting. It is shown that the FIML based augmentation is qualitatively similar to the analytic correction used to generate the data. Turbulent flows over curved surfaces are subjected to streamwise pressure gradients and additional mean strains. The flow may be stabilized or destabilized depending on the nature of curvature[8, 71]. Surface curvature can also lead to the creation of small-scale vortical structures[86], adding complex features to the flow-field.

4.1 Curvature Correction and Turbulence Models

Production terms in the equations of Reynolds stress can intrinsically capture the rotation and curvature effects. These terms are not present in most linear eddy-viscosity based models because the trace of the Coriolis forces in the Reynolds stress equations vanishes, and hence does not contribute to the turbulence kinetic energy (TKE)[1]. The modest improvement in accuracy provided by the full Reynolds stress closure, however, has not justified the cost and complexity associated with such models. Further, full Reynolds stress closures require explicit rotation and curvature corrections to match the model output with the observations[102]. Therefore, linear eddy-viscosity closures are still preferred and have been explicitly sensitized to capture rotation and curvature effects. Examples of such proposed corrections can be found in [115, 100, 23]. In this chapter we use the SA–RC correction to the SA model for generating the data and compare it with the FIML-augmented SA model.

[1] The production of TKE is exact, but the net result with other approximations lead to inaccuracies.

The forms of the baseline SA model and the SA–RC correction are presented in the following sections.

4.1.1 Baseline SA Model

The SA model for turbulent flows utilizes a differential equation for an operational variable $\tilde{\nu}$, which is used to estimate the eddy-viscosity. The details of the model are provided in A.1. The SA governing equation has the following form,

$$\frac{D\tilde{\nu}}{Dt} = P(\tilde{\nu}, \mathbf{Q}) - D(\tilde{\nu}, \mathbf{Q}) + T(\tilde{\nu}, \mathbf{Q}), \tag{4.1}$$

where P, D, and T are the production, dissipation, and diffusion terms, respectively, and $\mathbf{Q}$ is a vector containing flow variables.

4.1.2 SA–RC Augmentation to the SA Model

The SA–RC correction introduced by Spalart & Shur[100] multiplies the production term in equation 4.1 with a correction function f_{r1}. f_{r1} is a function of two Galilean-invariant non-dimensional variables $r_\star$ and $\tilde{r}$. The model takes the following form,

$$\frac{D\tilde{\nu}}{Dt} = f_{r1}P(\tilde{\nu}, \mathbf{Q}) - D(\tilde{\nu}, \mathbf{Q}) + T(\tilde{\nu}, \mathbf{Q}), \tag{4.2}$$

$$f_{r1} = (1 + c_{r1})\frac{2r^*}{1 + r^*}\left[1 - c_{r3}\text{tan}^{-1}(c_{r2}\tilde{r})\right] - c_{r1}. \tag{4.3}$$

where $r^* = S/\omega$, $\tilde{r} = \frac{2\omega_{ik}S_{jk}}{D^4}\left(\frac{DS_{ij}}{Dt} + \left(\epsilon_{imn}S_{jn} + \epsilon jmnS_{in}\right)\Omega'_m\right)$ and $D = \frac{1}{2}\left(S^2 + \omega^2\right)$. S is the strain rate tensor and ω is the vorticity tensor. More details on the constants and intermediate functions can be found in [100].

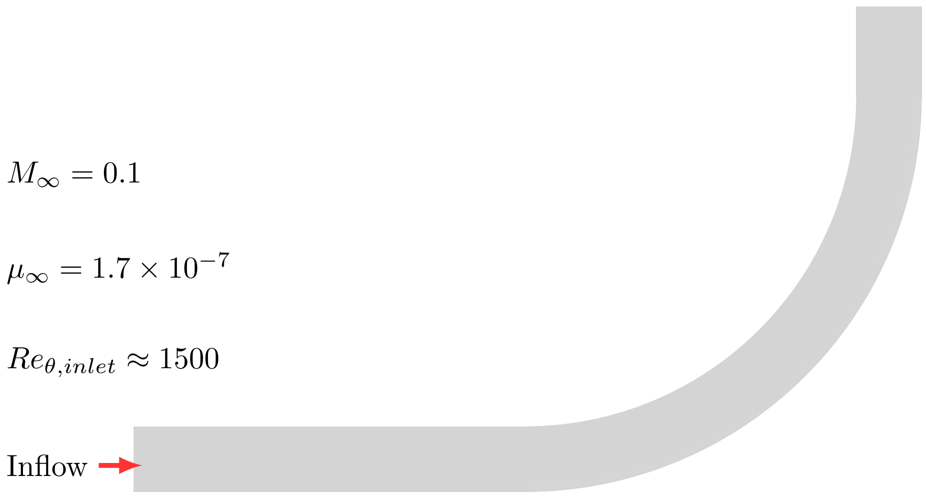
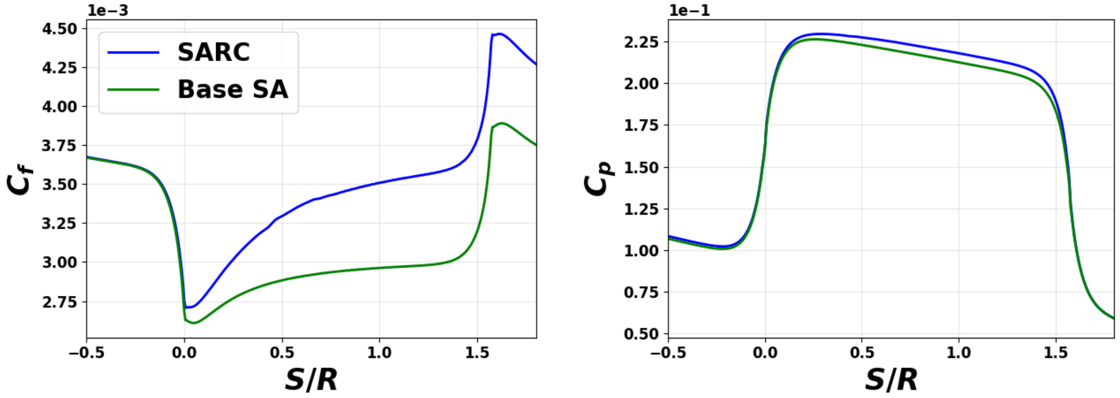

$M_\infty = 0.1$
$\mu_\infty = 1.7 \times 10^{-7}$
$Re_{\theta,inlet} \approx 1500$
Inflow

Figure 4.1: Flow setup and geometry for the concave curvature case. The inlet boundary layer is generated using a zero pressure gradient flat plate simulation.

4.2 Forward Problem

We consider flow in a planar channel with a concave curvature (Fig. 4.1). The inlet boundary layer thickness is set to match the LES by Arolla & Durbin[1]. The inlet momentum thickness Reynolds number Re_θ is approximately 1500. The incoming boundary profile is generated by running a zero pressure gradient flat plate simulation with the baseline SA model. The lower wall is set to a no-slip boundary condition, whereas the top wall is inviscid. A characteristic based boundary condition is used for the outlet. The mesh is generated by extracting a plane from the LES and skipping every other node in the streamwise direction.

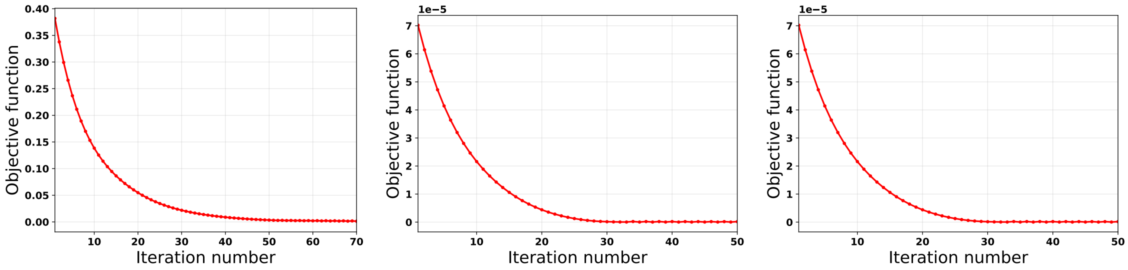
The baseline SA model and the SA–RC augmentation are used to obtain the flow solutions. Fig. 4.2 shows the skin friction and surface pressure at the lower wall using the two models. The SA–RC predictions are closer to the LES prediction while the baseline SA model fails to capture the effect of the curvature. While notable discrepancies are present between the LES and the SA–RC solution, for this chapter, the SA–RC solution is considered to be the *truth*. We use this SA–RC solution as the



(a) Skin friction

(b) Surface pressure

Figure 4.2: Comparison of the skin friction and the surface pressure coefficients at the lower wall using the LES, SA model, and SA–RC model. The skin friction predictions are much improved using SA–RC. S represents the streamwise distance.



(a) Inverse – U

(b) Inverse – C_f

(c) Inverse – C_p

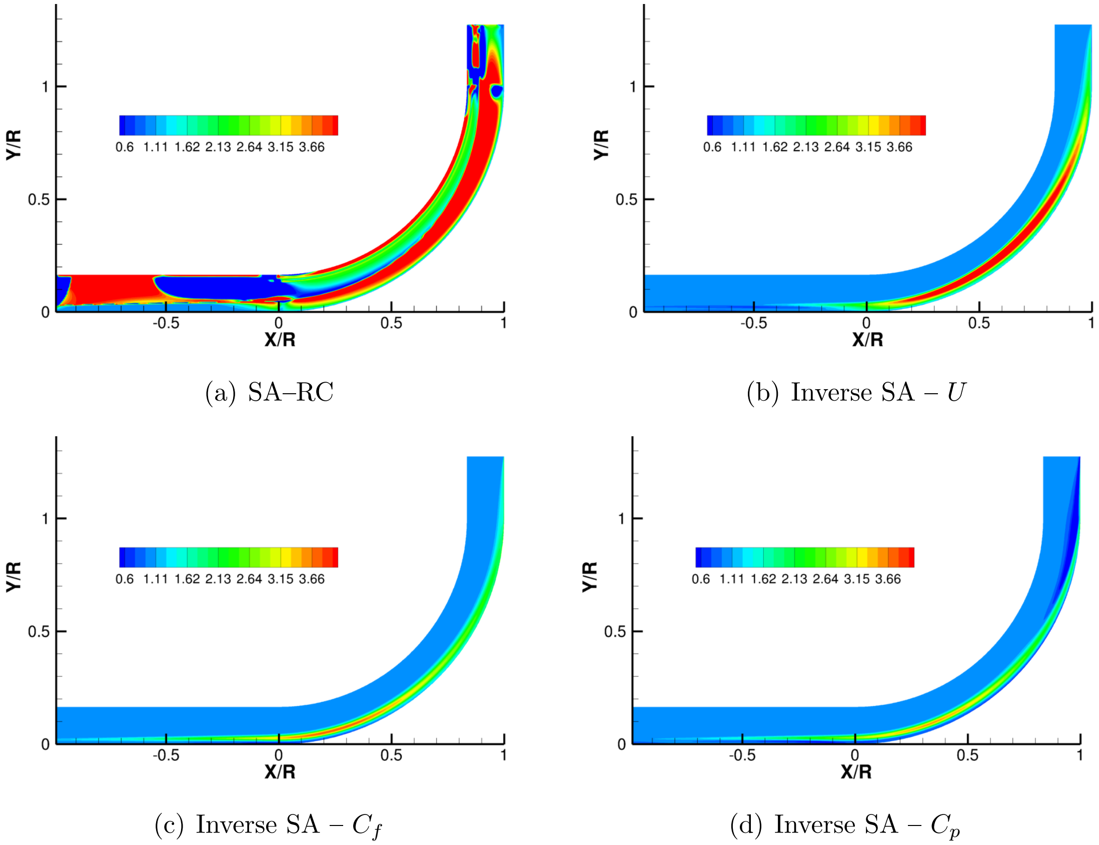
Figure 4.3: Convergence of the steepest descent algorithm for the three inverse problems.

data for the data-driven augmentation.

The next sections detail the FIML procedure applied to the present case.

4.3 Application of the FIML Framework

As described in Chapter II, the procedure for the model augmentation begins with the introduction of a functional discrepancy in an existing (or baseline) model. We add a correction function (or discrepancy), $\boldsymbol{\beta}(\mathbf{x})$, to the baseline SA model as follows



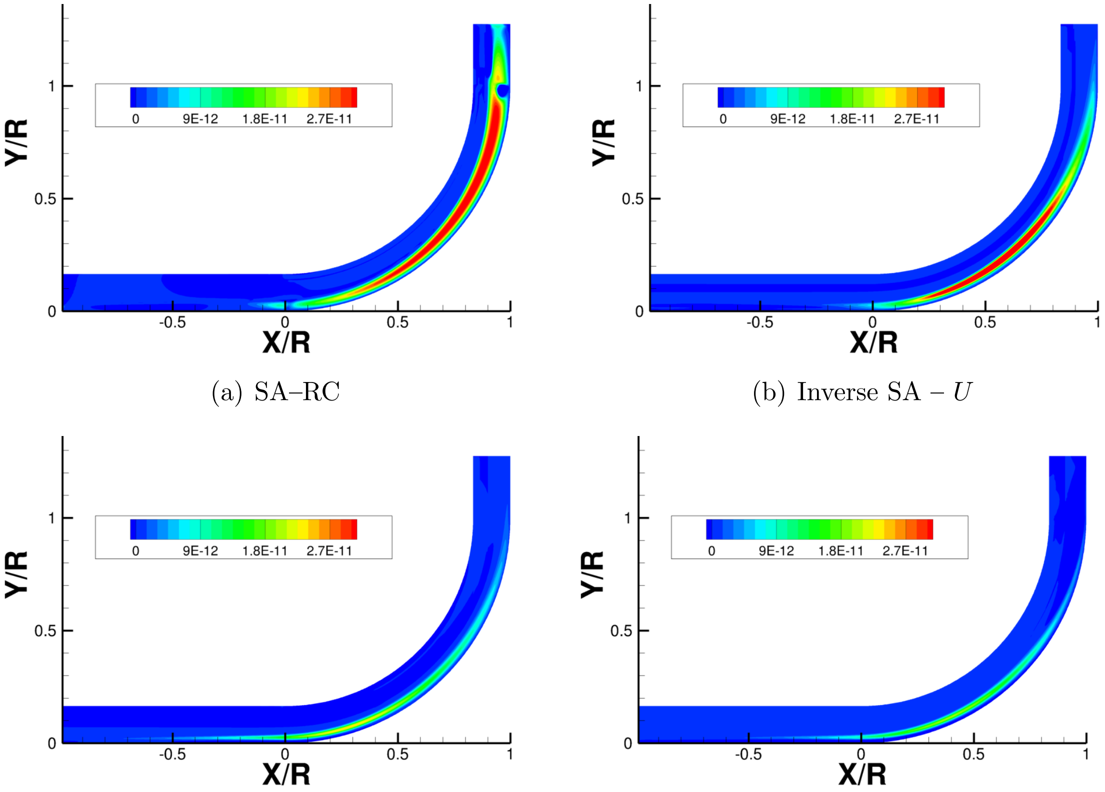
(a) SA–RC

(b) Inverse SA – U

(c) Inverse SA – C_f

(d) Inverse SA – C_p

Figure 4.4: Inferred non-dimensional discrepancy field ($\boldsymbol{\beta}(\mathbf{x})$) using three different objective functions and the equivalent term f_{r1} in the SA–RC model.



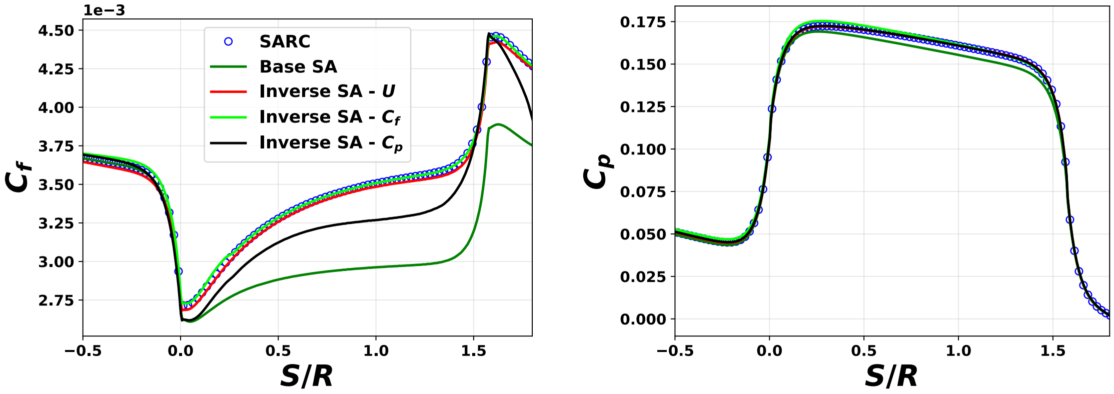
(a) SA–RC

(b) Inverse SA – U

(c) Inverse SA – C_f

(d) Inverse SA – C_p

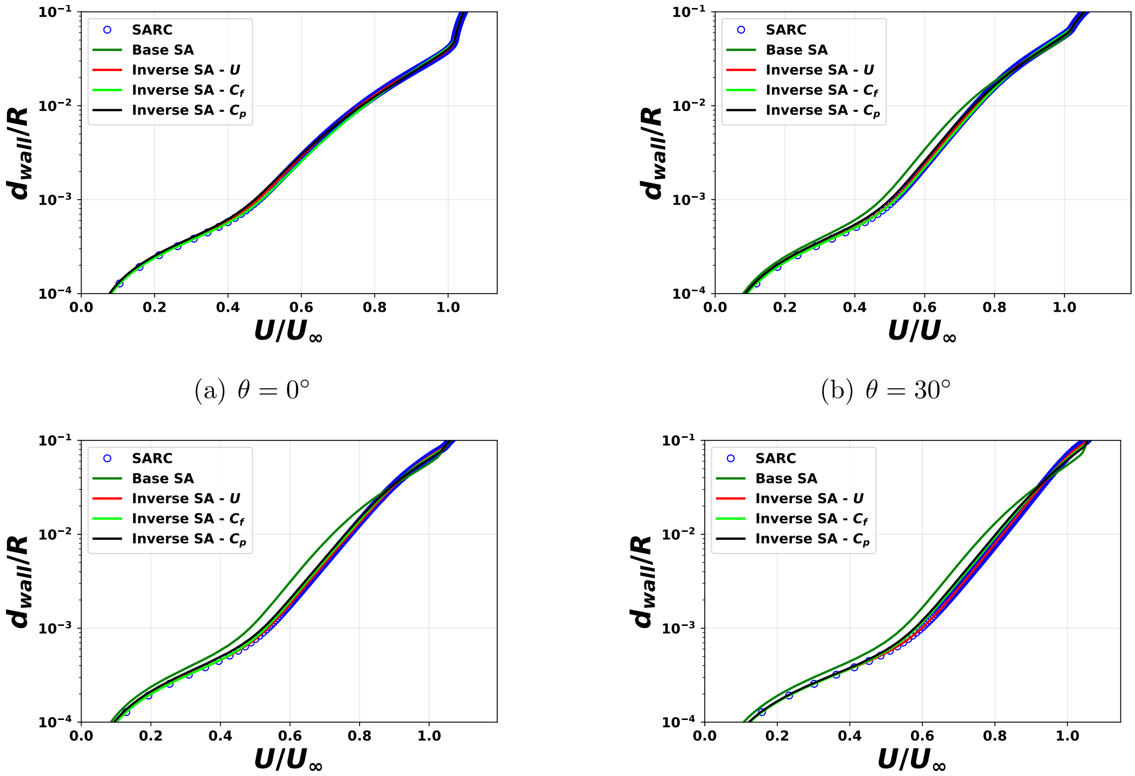
Figure 4.5: Inferred dimensional discrepancy field ($\delta(\mathbf{x})$) using three different objective functions and the equivalent term in the SA–RC model.



(a) Skin friction

(b) Surface pressure

Figure 4.6: Comparison of the skin friction and surface pressure coefficients at the lower wall using the SA–RC, baseline SA model, and inverse SA model using three different objective functions. S represents the streamwise distance.



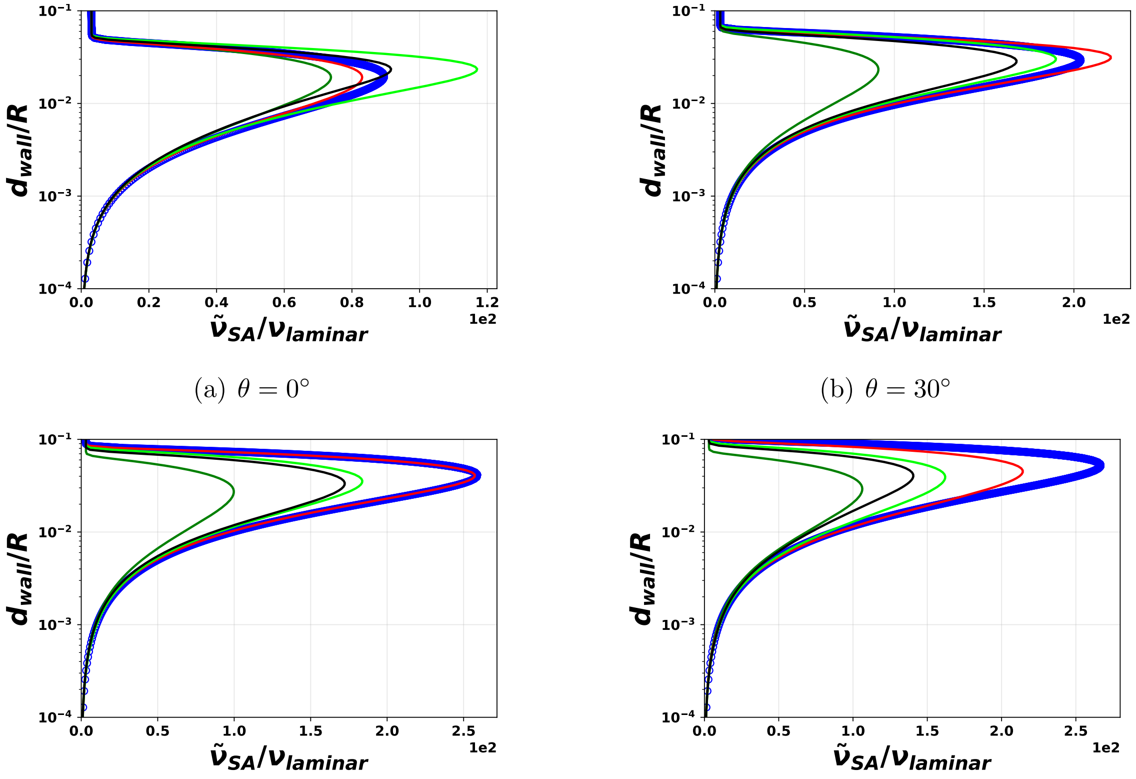
(a) $\theta = 0°$

(b) $\theta = 30°$

(c) $\theta = 60°$

(d) $\theta = 90°$

Figure 4.7: Tangential velocity profile at different streamwise locations. $\theta = 0°$ marks the onset of the curvature and $\theta = 90°$ marks its end.



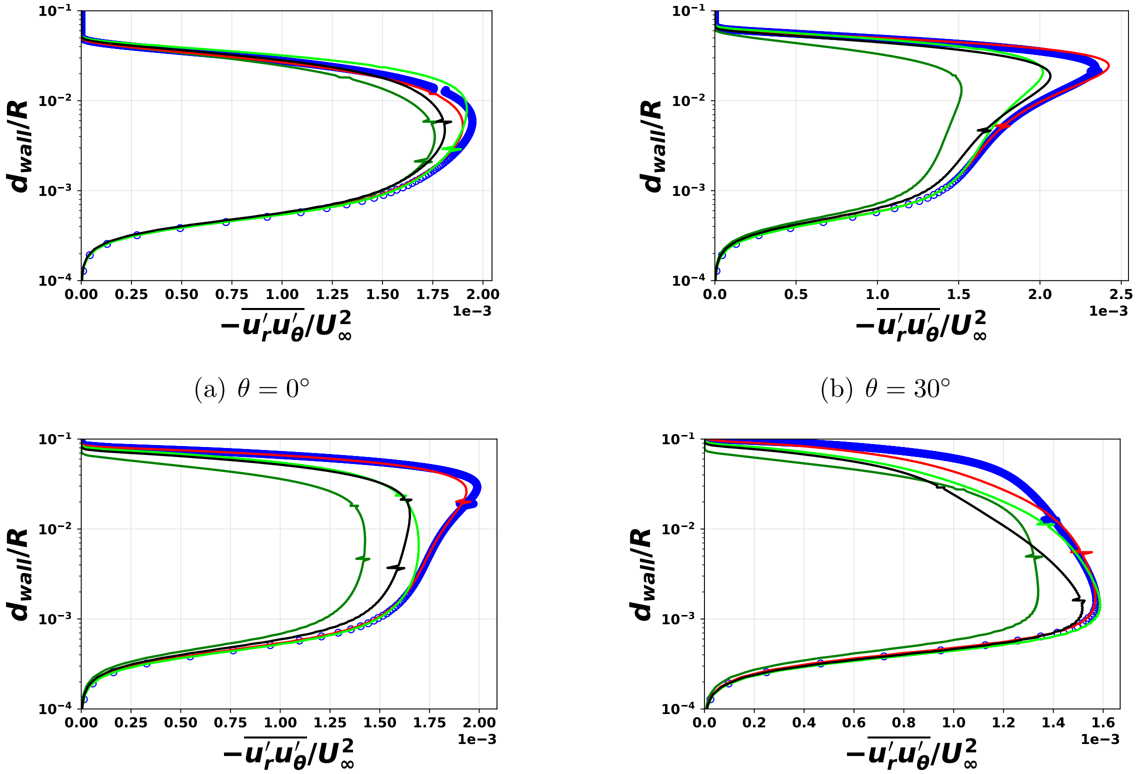
(a) $\theta = 0°$

(b) $\theta = 30°$

(c) $\theta = 60°$

(d) $\theta = 90°$

Figure 4.8: SA eddy-viscosity profile at different streamwise locations. $\theta = 0°$ marks the onset of the curvature and $\theta = 90°$ marks its end.



(a) $\theta = 0°$

(b) $\theta = 30°$

(c) $\theta = 60°$

(d) $\theta = 90°$

Figure 4.9: Reynolds shear-stress $-\overline{u'_r u'_\theta}$ profile at different streamwise locations. $\theta = 0°$ marks the onset of the curvature and $\theta = 90°$ marks its end.

$$\frac{D\tilde{\nu}}{Dt} = \boldsymbol{\beta}(\mathbf{x})P(\tilde{\nu}, \mathbf{Q}) - D(\tilde{\nu}, \mathbf{Q}) + T(\tilde{\nu}, \mathbf{Q}). \tag{4.4}$$

The introduction of the correction function $\boldsymbol{\beta}(\mathbf{x})$ is equivalent to adding a term $\boldsymbol{\delta}(\mathbf{x}) = (\boldsymbol{\beta}(\mathbf{x}) - 1)P(\tilde{\nu}, \mathbf{Q})$. The form of $\boldsymbol{\beta}(\mathbf{x})$ makes the correction function non-dimensional with a prior (baseline) value of unity. This setup also improves the conditioning of the inverse problem as the magnitude of the correction function will be $\mathcal{O}(1)$ in the entire domain.

The introduction of the discrepancy does not just modify the production term but changes the net balance of the source terms. Its form does not limit the effectiveness of the discrepancy as the effects of turbulence are not important in the regions of zero production. From equations 4.2 and 4.4, it is clear that $\boldsymbol{\beta}(\mathbf{x})$ and f_{r1} serves the same purpose and the FIML can be expected to produce a form of $\boldsymbol{\beta}(\mathbf{x})$ similar to f_{r1}.

Next, we follow the FIML procedures to discover the form of $\boldsymbol{\beta}(\mathbf{x})$ which can improve the prediction for a given flow setup.

4.3.1 Field Inversion

Inverse problems are solved to infer the spatial distribution of the discrepancy required for the model prediction to match the data. The data for the inverse problems comes from the SA–RC solution. Since the data has no uncertainty, we set a low value for the Tikhonov regularization parameter $\lambda \approx 10^{-12}$. A low value of λ reflects a high level of confidence in the data.

The quality of the inverse solution, measured as the difference in the flow-field using the inferred discrepancy and the data, is dependent on the nature of the data. In general, full-field data is expected to be better than using surface data. Moreover, the surface data is expected to be better than using integral data like lift or drag. Often the choice of data used for inversion is made by the scarcity of the data. As the data is synthetically generated, we have access to full fields and various surface quantities. Therefore for the current case, we compare the quality of the inverse solution using three different data types: (1) full-field velocity (Inverse $SA - U$), (2) lower wall skin friction (Inverse $SA - C_f$), and (3) lower wall pressure (Inverse $SA - C_p$). The objective functions for these cases have the following form,

$$J_U(\boldsymbol{\beta}) = \sum_{i=1}^{N_{cells}} \left[U_i - U_{i,SA-RC}\right]^2 + \lambda \sum_{j=1}^{N_{cells}} [\boldsymbol{\beta}_j - 1]^2, \tag{4.5}$$

$$J_{C_f}(\boldsymbol{\beta}) = \sum_{i=1}^{N_{wall}} \left[C_{f,i} - C_{f,i,SA-RC}\right]^2 + \lambda \sum_{j=1}^{N_{cells}} [\boldsymbol{\beta}_j - 1]^2, \tag{4.6}$$

$$J_{C_p}(\boldsymbol{\beta}) = \sum_{i=1}^{N_{wall}} \left[C_{p,i} - C_{p,i,SA-RC}\right]^2 + \lambda \sum_{j=1}^{N_{cells}} [\boldsymbol{\beta}_j - 1]^2. \tag{4.7}$$

While all the available data, including full-field eddy-viscosity and Reynolds stresses are used to evaluate the quality of the inverse solutions, we do not use them to inform the solution.

Steepest descent, with fixed step size, is used to solve the inverse problems and the inverse iterations are terminated when the objective function attains a steady state. Fig. 4.3 shows the convergence of the inverse iterations for all the three cases. A drop in the objective function of at least one order of magnitude is achieved in all cases.

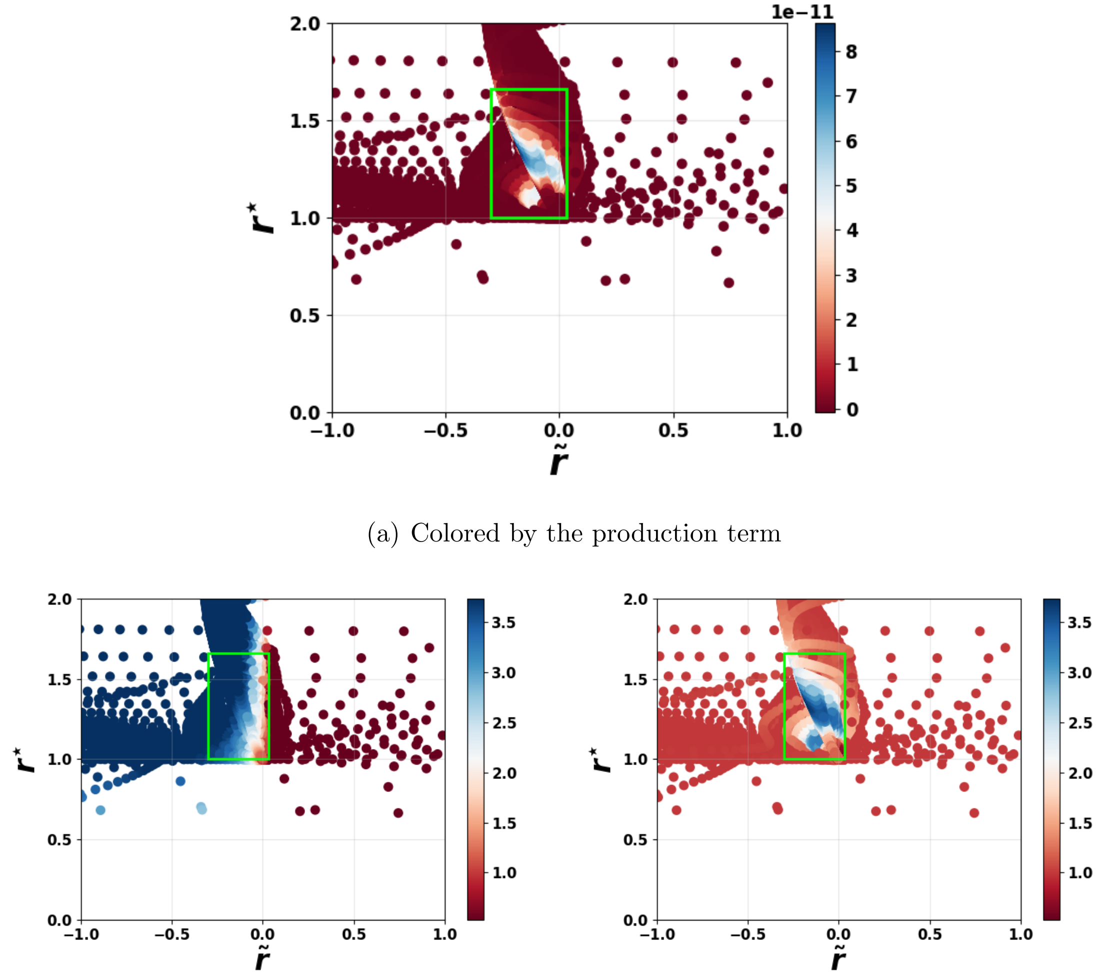
Fig. 4.4 shows the inferred correction term $\boldsymbol{\beta}$ for the three cases as well as the SA–RC correction term f_{r1}. The SA–RC correction terms takes non-zero values away from the wall, but it does not affect the solution because the production term is zero in those regions. Accordingly, we compare the net effect of the correction $\boldsymbol{\delta}$, which is $(f_{r1} - 1)P$ for the SA–RC and $(\boldsymbol{\beta} - 1)P$ for the inverse solutions. This net correction term is shown in Fig. 4.5. There are qualitative similarities between the various cases. In all the cases, the net production is increased in the curved region. This result is expected based on our physical understanding of the problem. The inferred term with full-field velocity data is closer to the SA–RC term, followed by the inverse solutions using skin friction C_f and surface pressure C_p. This is an expected trend; more data should lead to a more informed solution.

Fig. 4.6 shows the skin friction and the pressure at the lower wall for the different inversion cases. The two quantities obtained using the inferred discrepancy fields are closer to the SA–RC than the baseline SA model. Skin-friction using velocity data and skin-friction data are nearly identical to the SA–RC solution.

Fig. 4.7 shows the tangential velocity profiles at different streamwise locations. Improvements in the velocity profiles are observed for all the cases. Fig. 4.8 and 4.9 shows the SA eddy-viscosity and the Reynolds shear-stress profiles at various streamwise locations. The improvements in the Reynolds shear-stress are more pronounced than the SA eddy-viscosity. These differences are reflective of the fact that the flow solution is influenced by the derivative of Reynolds shear-stress and not the eddy-viscosity alone. The inversion using velocity recovers the Reynolds shear-stress almost perfectly at all the locations. Skin-friction based inversion consistently recovers the shear-stress in the inner layer, but this trend is not so robust in the log-layer and the outer layer. The pressure based inversion has the poorest performance.

Improvements in the auxiliary field quantities add to the confidence in the inverse solutions. It shows that the inversion is not just over-fitting the model to the data. It is especially pertinent to the cases where the surface data is used, and we observe improvements in the field quantities such as velocity and Reynolds stresses. While the quality of the three inverse solutions varies, the improvements are substantial in comparison to the baseline SA model.

The spatial form of the inferred discrepancy $\boldsymbol{\beta}$ can be used by a modeler to suggest an analytical correction to the model. However, the ability of a human-modeler to find patterns in data is limited. Therefore, we use machine-learning (ML) to reconstruct $\boldsymbol{\beta}$ as a function of the flow solution.



(a) Colored by the production term

(b) Colored by the analytic f_{r1} evaluated at these points

(c) Colored by the inferred discrepancy $\boldsymbol{\beta}$

Figure 4.10: Scatter plot of the two features used in SA–RC model. The features are evaluated at the inverse solution and are colored by: (a) Production, (b) analytic f_{r1}, and (c) inferred $\boldsymbol{\beta}$. The region of interest is enclosed in the green rectangle. Outside this region the production term is zero. Therefore, differences between f_{r1} and $\boldsymbol{\beta}$ outside this region will have minimal impact on the flow solution.

Model Label	*Training Data*	*Features, $\boldsymbol{\eta}$*
Model 1	Inverse - C_f	$\hat{r}$, $r_\star$, $\rho\lvert S\rvert d_{wall}^2/\mu_L$
Model 2	Inverse - C_f	$\rho\lvert S\rvert d_{wall}^2/\mu_L$, $\mu_T\lvert S\rvert/\tau_{wall}$, d_{wall}
Model 22	Inverse - U	$\rho\lvert S\rvert d_{wall}^2/\mu_L$, $\mu_T\lvert S\rvert/\tau_{wall}$, d_{wall}
Model 3	Inverse - C_f	$\hat{r}$, $r_\star$, $\mu_T\lvert S\rvert/\tau_{wall}$
Model 4	Analytic f_{r1}	$\hat{r}$, $r_\star$

Table 4.1: List of ML models along with the data and flow-features used for training.

4.3.2 Machine Learning Training

In the FIML framework, the ML algorithms are used to project the spatial function $\boldsymbol{\beta}(\mathbf{x})$ onto a feature space to obtain $\boldsymbol{\beta}(\boldsymbol{\eta}(\mathbf{Q}))$, which can then be used for prediction. The selection of features $\boldsymbol{\eta}$ and the data used for training the ML model have a significant impact on the quality of the ML model. In this work, features are selected on the basis of domain knowledge. Table 4.1 shows the different ML-based models with the data and the flow-features used for training. As seen in the previous section, the quality of the inverse solution depends on the data used for inversion. The same trend is expected with the ML models as the inverse solutions are used for training the models. If the training is "perfect", the prediction using an ML-based model should be identical to the inverse solution used for training.

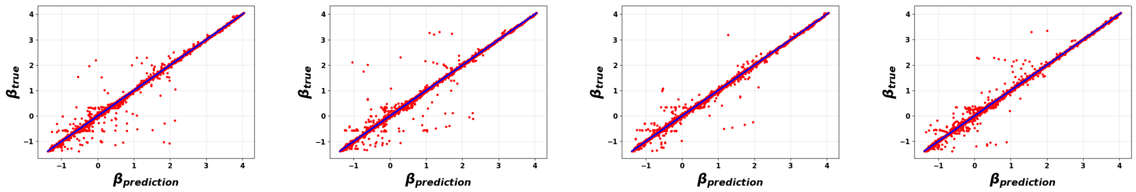
The training data for model 2 is the inverse solution obtained using skin friction C_f. The flow features are

$$\eta_1 = \rho|S|d_{wall}^2/\mu_L, \tag{4.8}$$

$$\eta_2 = \mu_T|S|/\tau_{wall}, \tag{4.9}$$

$$\eta_3 = d_{wall}, \tag{4.10}$$

where τ_{wall} and d_{wall} are the wall shear-stress and distance of the nearest wall. While other sets and combinations of features are explored, including the features used in the



(a) $CV-1$ (b) $CV-2$ (c) $CV-3$ (d) $CV-4$

Figure 4.11: Test results for Model 2 using 4-fold cross-validation and the AdaBoost algorithm.

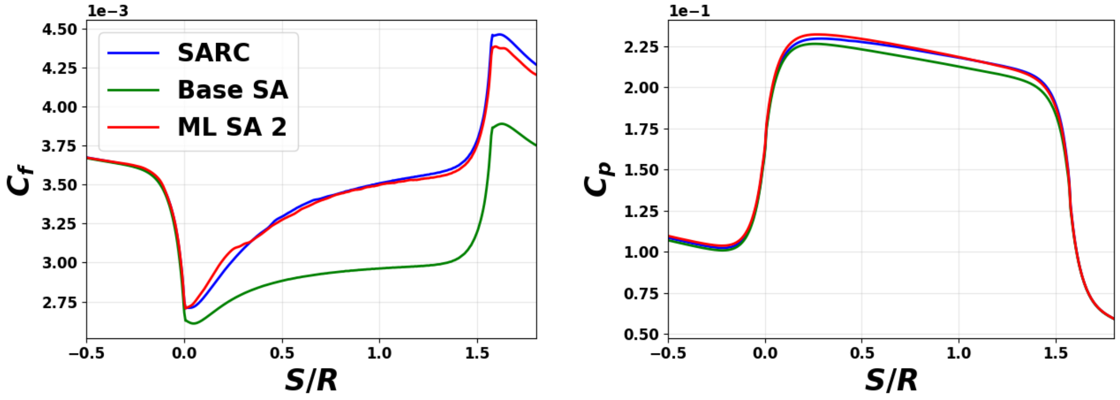
SA–RC correction ($\hat{r}$ and $r_\star$), these features produced the most optimal ML model.

The ML algorithm AdaBoost is used to find the functional $\boldsymbol{\beta}(\eta_1, \eta_2, \eta_3)$ using 4-fold cross-validation. For all the stages of cross validation, the minimum coefficient of determination for the prediction of test data points was 0.99. Fig. 4.11 shows the predictions for these test data-points; clearly it represents a good ML training.

$$\frac{D\tilde{\nu}}{Dt} = \boldsymbol{\beta}(\boldsymbol{\eta})P(\tilde{\nu}, \mathbf{Q}) - D(\tilde{\nu}, \mathbf{Q}) + T(\tilde{\nu}, \mathbf{Q}) \tag{4.11}$$

Once the ML model is trained, the baseline RANS equations along with the ML-based augmentation (Eq. 4.11) are then solved to obtain predictive results. Theoretically, it is possible to use these models for any other geometry or flow setup, however, improvements are expected for the flows which are *similar*. Since the goal in this chapter is to present the results for an idealized scenario, we use the augmentation for prediction on the same flow setup used to generate the training data.

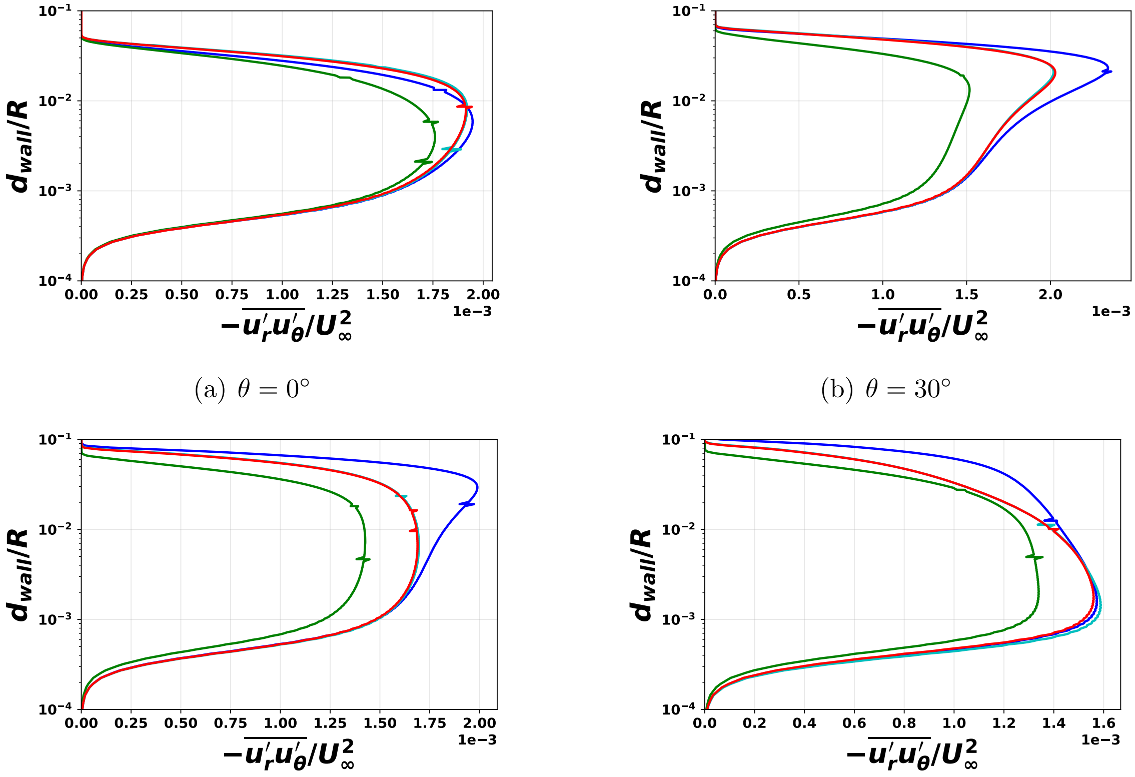
The ML augmentation interacts with the RANS equations in a non-linear and coupled manner. Even with "perfect" training, it is not necessary that the converged solution with the ML augmentation will be similar to the inverse solution used for training. It is especially true with iterative solvers as the intermediate flow-features $\boldsymbol{\eta}$ can be radically different from the steady-state features used for training. Therefore, the ultimate test of a ML-based augmentation is when it is used for prediction in an iterative solver.



(a) Skin friction

(b) Surface pressure

Figure 4.12: Comparison of the skin friction and the surface pressure coefficients at the lower wall using the SA–RC, the baseline SA model, and the ML augmented SA model 2. S represents the streamwise distance.



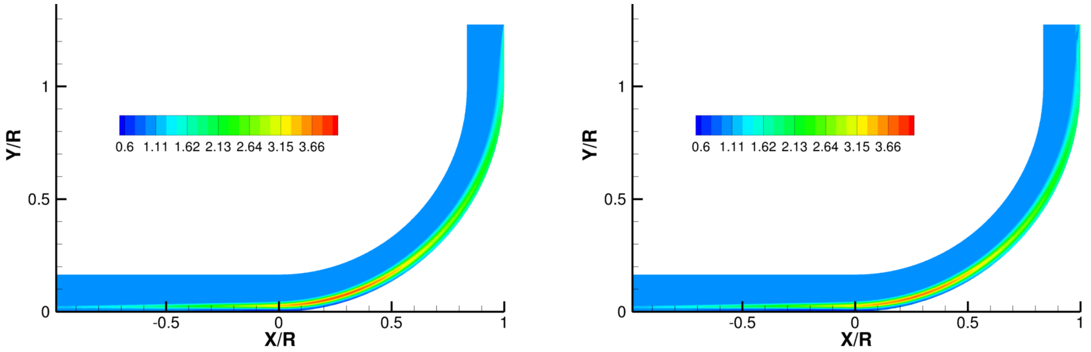
(a) $\theta = 0°$

(b) $\theta = 30°$

(c) $\theta = 60°$

(d) $\theta = 90°$

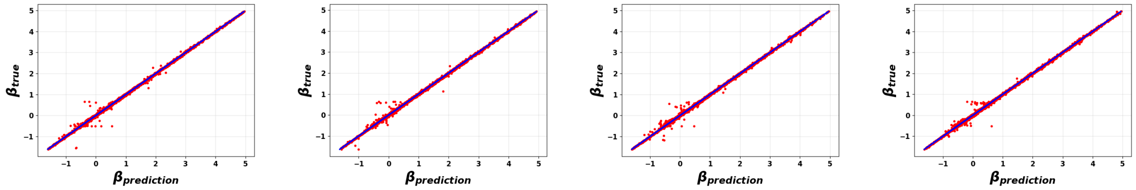
Figure 4.13: Reynolds shear-stress profile at different streamwise locations. $\theta = 0°$ marks the onset of the curvature and $\theta = 90°$ marks the end of the curvature. Legends: — SA–RC, — Base SA, — Inverse SA - C_f and — ML SA model 2.



(a) Inverse SA – C_f

(b) Model 2

Figure 4.14: Inferred discrepancy field $\boldsymbol{\beta}(\mathbf{x})$ using C_f and the ML SA model 2 predicted $\boldsymbol{\beta}(\eta)$ at the converged solution.



(a) $CV-1$

(b) $CV-2$

(c) $CV-3$

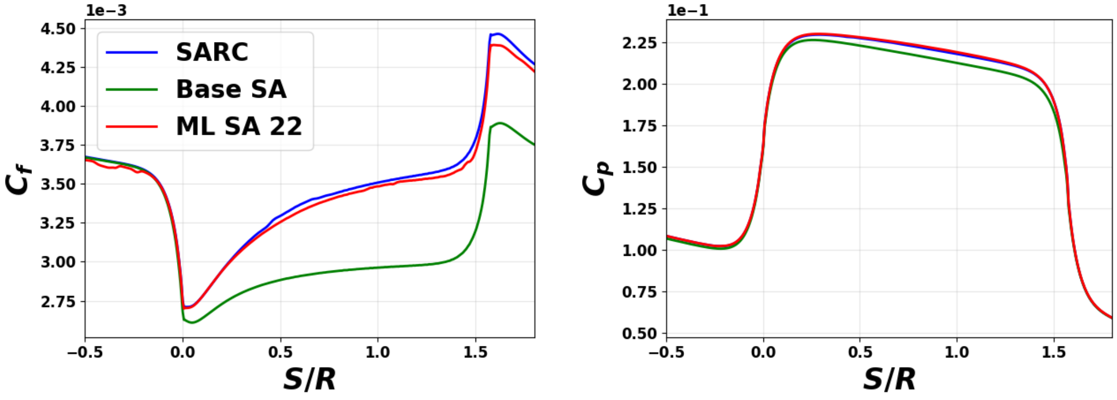
(d) $CV-4$

Figure 4.15: Test results for Model 22 using 4-fold CV and the AdaBoost algorithm.

4.3.3 Machine Learning Prediction

The ML-based models are used in an iterative solver. Fig. 4.12 shows the predicted skin friction using model 2. In most regions, the predicted skin-friction is close to the SA–RC skin-friction. The error is significantly smaller when compared to the baseline SA model. Similar improvements can be observed in the surface pressure. Fig. 4.13 shows the predicted Reynolds shear-stress at various streamwise locations. The Reynolds shear-stress predictions are almost identical to the inverse solution. Also, as with the inverse solution, the predictions are significantly accurate in comparison with the baseline model. Further, Fig. 4.14 shows the discrepancy field predicted using model 2, which is a good representation of the corresponding inverse solution.

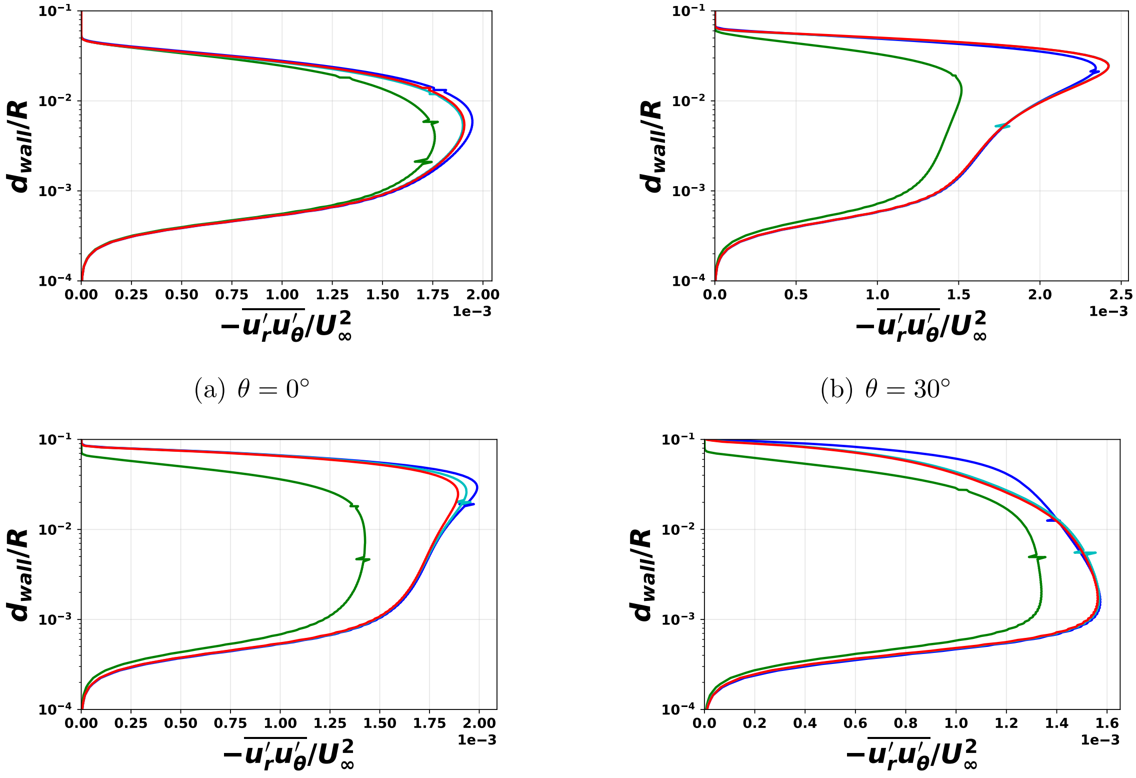
Model 22 is trained using a setup similar to the model 2. However, for this model, the inverse solution obtained using velocity is used as training data. Fig. 4.16 shows



(a) Skin friction

(b) Surface pressure

Figure 4.16: Comparison of the skin friction and the surface pressure coefficients at the lower wall using the SA–RC, baseline SA model, and ML augmented SA model 22. S represents the streamwise distance.



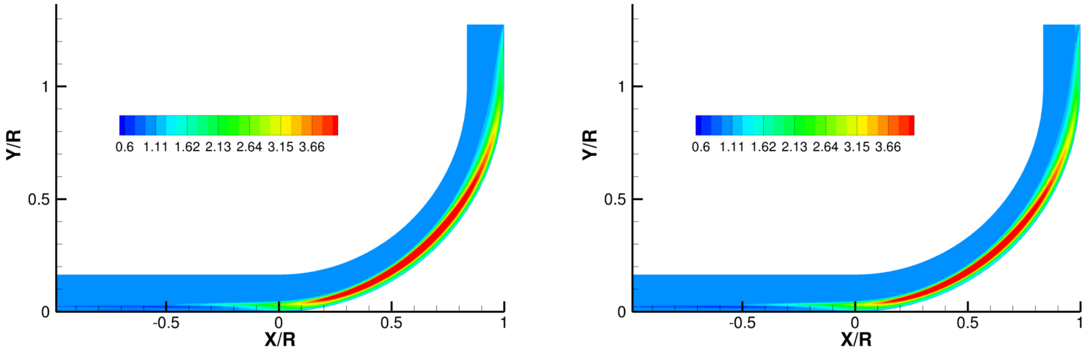
(a) $\theta = 0°$

(b) $\theta = 30°$

(c) $\theta = 60°$

(d) $\theta = 90°$

Figure 4.17: Reynolds shear-stress profile at different streamwise locations. $\theta = 0°$ marks the onset of the curvature and $\theta = 90°$ marks the end of the curvature. Legends: — SA–RC, — Base SA, — Inverse SA - U and — ML SA model 22.



(a) Inverse SA – U

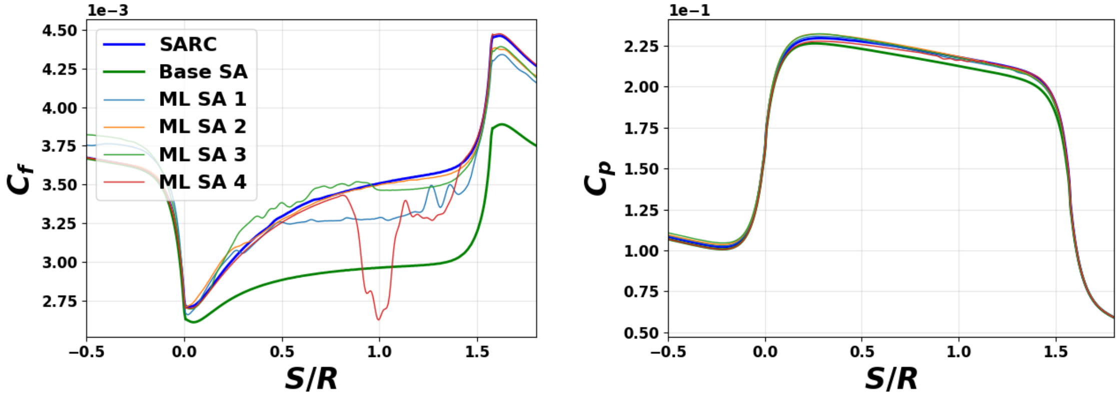
(b) Model 22

Figure 4.18: Inferred discrepancy field $\boldsymbol{\beta}(\mathbf{x})$ using U and the ML SA model 22 predicted $\boldsymbol{\beta}(\eta)$ at the converged solution.

the skin friction and pressure at the lower wall. The ML augmented model shows significant improvement in all regions. Moreover, Fig. 4.17 shows the Reynolds stress profiles at different streamwise locations. It can be seen that the prediction using the ML augmented model is almost identical to the inverse solution, which is also close to the SA–RC prediction. Fig. 4.18 shows the discrepancy field predicted using the ML-based augmentation and again it is a good representation of the inverse solution.

One of the drawbacks of model 2 and 22 is the presence of non-local quantities: τ_{wall} and d_{wall}. The non-local quantities may not be desirable in a parallel solver as it involves global communications. Further, d_{wall} in itself is not an ideal feature as it is not Galilean invariant.

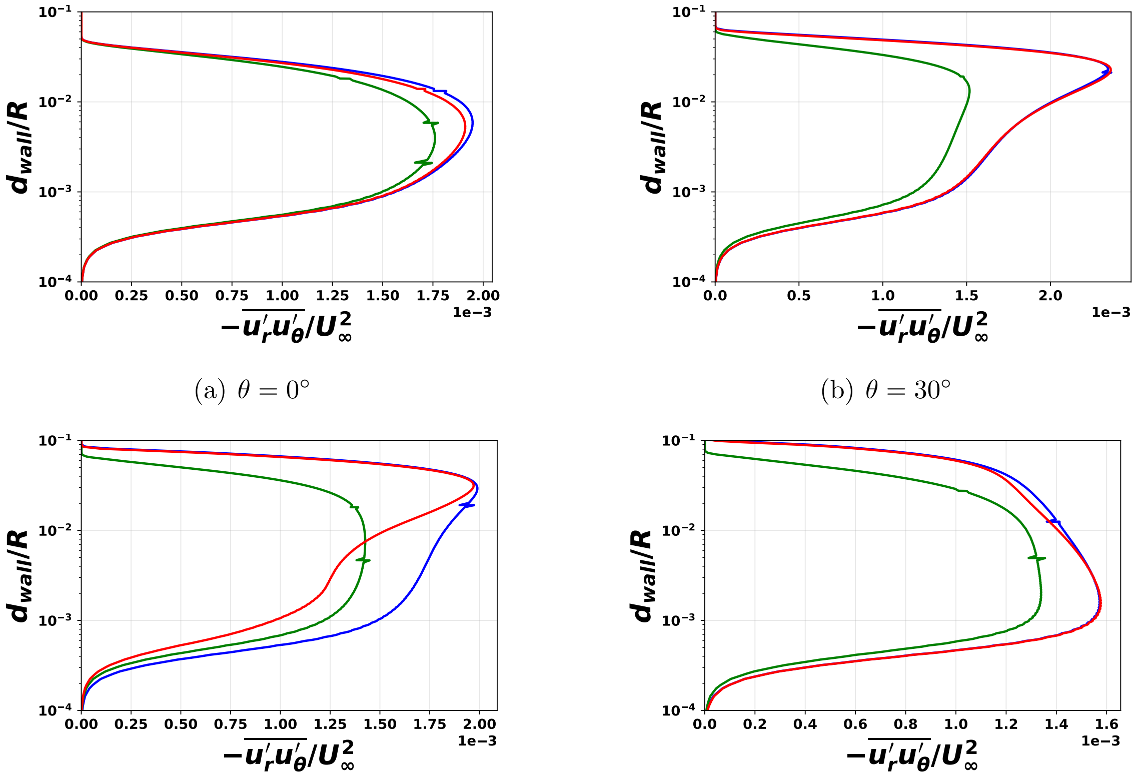
The problem of non-local features can be avoided by using features which are local and locally non-dimensional. Models 1 and 3 are trained on such flow-features. Two of the features are identical to the ones used in the SA–RC correction. A third feature is required as the two features were inadequate to train a ML model. The models are trained on the inverse solution based on skin friction. The predictions using these models are shown in Fig. 4.19. While the solution is better than the baseline model, the quality is worse than what is predicted by Model 2. It should be noted that this set of flow-features involves calculating the second derivative of velocity.



(a) Skin friction

(b) Surface pressure

Figure 4.19: Comparison of the skin friction and the surface pressure coefficients at the lower wall using the SA–RC, baseline SA model, and ensembles of various ML augmented SA model. S represents the streamwise distance.



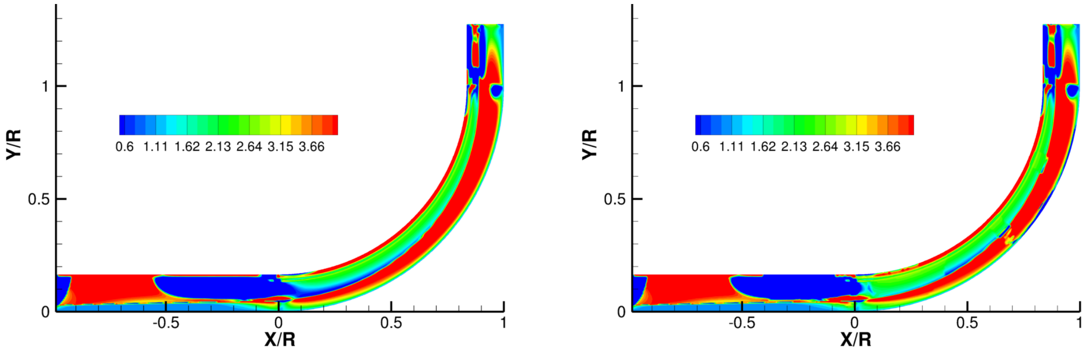
(a) $\theta = 0°$

(b) $\theta = 30°$

(c) $\theta = 60°$

(d) $\theta = 90°$

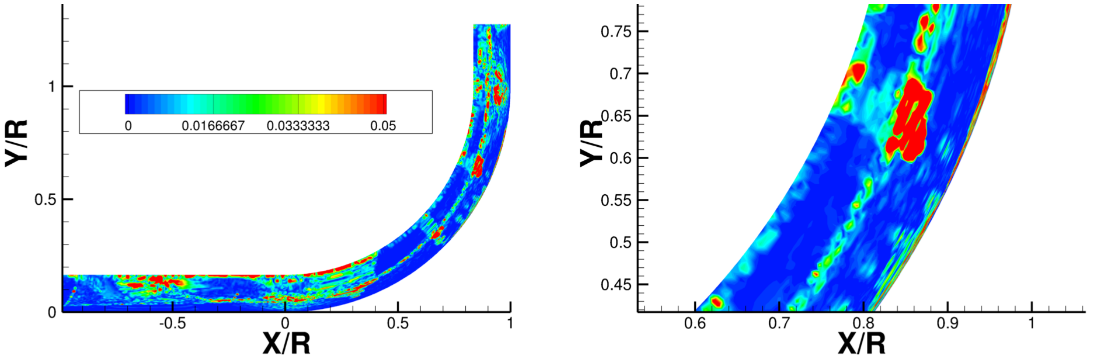
Figure 4.20: Reynolds shear-stress profile at different streamwise locations. $\theta = 0°$ marks the onset of the curvature and $\theta = 90°$ marks the end of the curvature. Legends: — SA–RC, — Base SA, and — ML SA model 4.



(a) Analytic evaluation of f_{r1}

(b) Model 4

Figure 4.21: Comparison of the analytic correction f_{r1} with the ML reconstruction of f_{r1}.



(a)

(b)

Figure 4.22: Absolute difference between the analytic correction f_{r1} and the ML reconstruction of f_{r1}.

4.3.4 Learning the Analytic Correction Without Inversion

As a final test, we use ML to learn the analytical correction function f_{r1}. The features for the training, $\hat{r}$ and $r_{\star}$, are extracted at the inverse solution however, instead of reconstructing the inverse discrepancy field, we evaluate the analytic f_{r1} at these points and reconstruct the function $f_{r1,ML}(\hat{r}, r_{star})$. We label this as model 4 and use it for prediction. The ML training for this reconstruction is almost perfect, and the coefficient of determination is greater than 0.99. Fig. 4.21 compares the contour of the analytic f_{r1} with the ML f_{r1} at the converged solution. While most of the characteristics are reproduced accurately, the model fails to provide a flow field which is similar to the SA–RC.

This failure is apparent in skin friction and pressure in Fig. 4.19 and in the Reynolds stresses in Fig. 4.20. Reynolds stresses are identical to the SA–RC at three out of four locations but differ significantly around $\theta = 60°$. This difference is due to the differences in the f_{r1} prediction in the same region as shown in Fig. 4.22. The reason for this discrepancy even with almost *perfect* ($R^2 = 0.99$) training is because of the high gradient of the function f_{r1} with respect to $\hat{r}$. Even a small imperfection in training can set f_{r1} to a non-zero value in where it should be zero, and vice versa. In comparison, the inferred $\boldsymbol{\beta}$ is a relatively well behaved and smooth function and thus less prone to such issues. This behavior also supports the claim that the real test of any ML augmentation is when it is used in an iterative RANS solver, and the quality of ML training is a necessary but not sufficient measure of the quality of the model prediction.

4.4 Summary

This section summarizes the information presented in the chapter.

1. A rotation and curvature augmentation to the SA model, SA–RC, was used to

generate data to test the FIML approach for model augmentation. The SA–RC augmentation and the discrepancy introduced in the FIML approach have the same form, and they are expected to have a similar form.

2. Multiple inverse problems were solved using full-field velocity data, surface skin friction, and surface pressure. In all the cases the inverse solutions were significantly more accurate than the baseline solutions. Velocity based inversion recovers the Reynolds stress field with significant accuracy, while the accuracy varies for the other two.

3. More comprehensive data (full-field velocity) was found to result in an inverse solution with more accurate field quantities in comparison to the surface data-based inversion. Depending on the quantity of interest, however, inference using sparse surface data can be sufficient. This is useful when only sparse experimental data is available.

4. Inverse solutions were reconstructed using machine-learning (ML) and used for prediction in the RANS solver. The prediction using ML models were found to be consistent with the corresponding inverse solution. All the ML based models showed significant improvements compared to the baseline model.

5. The SA–RC analytic correction f_{r1} was replaced with its ML reconstruction in the RANS solver. Prediction using the ML reconstruction was not on par with the analytic correction, even with perfect training. The difference was because of the relatively non-smooth nature of f_{r1} when compared to the inverse solution $\boldsymbol{\beta}$. This further emphasizes the importance of the inversion and avoiding ML training on ad-hoc functions.

CHAPTER V

Application to Adverse Pressure Gradient Flows

Turbulence models are often inaccurate in the presence of adverse pressure gradients (APG) and flow-separation. This largely limits the broader use of computational fluid dynamics in complex problems. The following two chapters focus on such flows. In this chapter, the Wilcox $k - \omega$ model[113] is augmented for flow over a smooth bump. The flow over a bump consists of regions of favorable pressure gradients followed by regions of adverse pressure gradients. The height of the bump controls the intensity of the pressure gradients. The forward part of the bump stabilizes the turbulence and reduces mixing, whereas the aft section produces the opposite effects. The equilibrium-based eddy-viscosity models do not capture the effects of adverse pressure gradients and underpredict mixing in the aft section, resulting in a lower skin friction prediction than large-eddy simulations (LES).

In this chapter, the bump height and the incoming boundary layer properties are varied, resulting in a total of 11 different cases. LES data from these cases is used to infer discrepancies in the $k-\omega$ model. A comparison of the inverse solutions informed by skin-friction and full-field velocity is presented. Machine-learning (ML) models are trained on multiple combinations of the 11 inverse solutions, which are then used for prediction.

5.1 Forward Problem

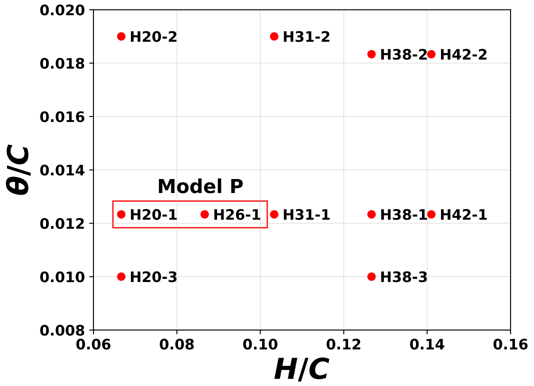
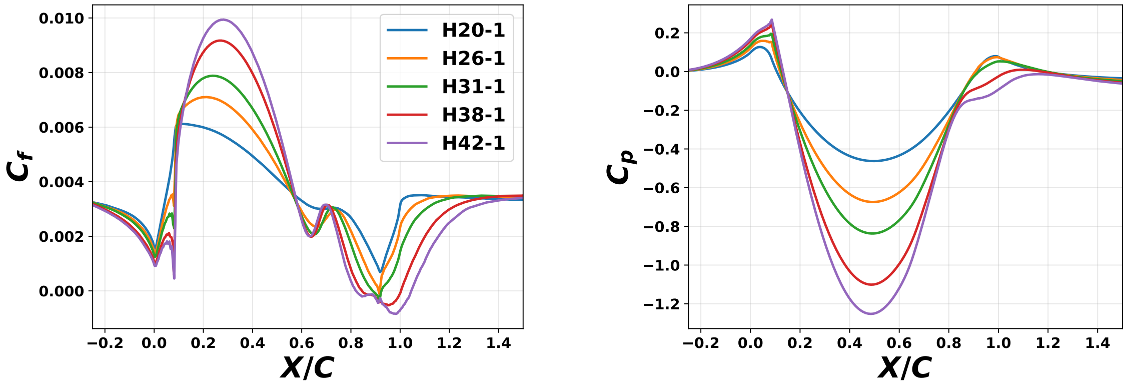


Figure 5.1: Labels for the various flow cases based on variation in the bump height and inlet momentum thickness. Inverse solutions for the cases marked in the red box are used to train model **P**.

The flow setup is based on a set of large-eddy simulations (LES) performed by Durbin et al.[90]. The problem consists of an incoming turbulent boundary layer with Re_θ varying from 2000 to 4000 and the height of the bump ranging from 7% to 14% of its length. Fig. 5.1 shows the labels of all 11 forward cases with different bump heights and momentum layer thickness. Wilcox $k-\omega$ model[113] is used as the baseline RANS model. The $k-\omega$ model is used in place of the SA model to exhibit the generality of the FIML framework. Freestream conditions are selected to match the LES. The incoming boundary layer is generated by running a zero-pressure gradient flat-plate simulation and is enforced on the left boundary. A no-slip boundary condition is used for the bottom wall, and characteristic-based freestream boundary conditions are used for the top and the right boundaries. The solution is driven to a steady state starting from a freestream initial condition.

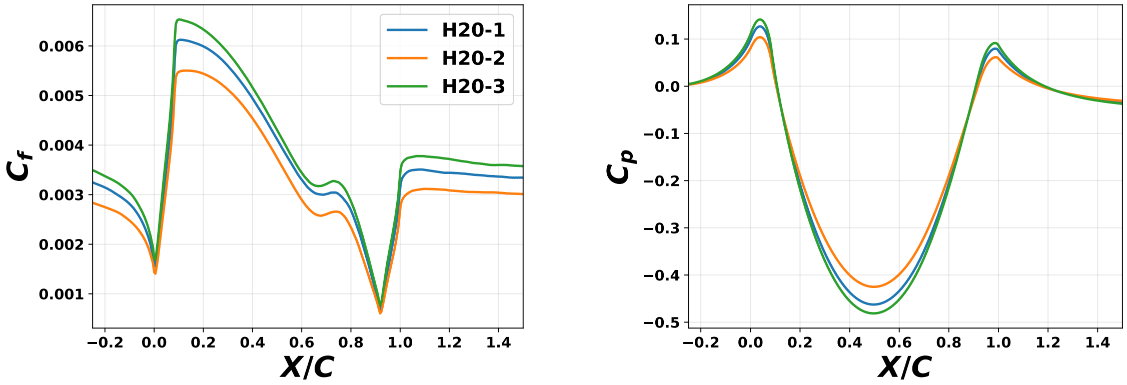
Figs. 5.2 and 5.3 show the LES skin friction and surface pressure exhibiting the differences in the flow and the adverse pressure gradient between various cases. The flow over the bump H20-1 does not separate; H26-1 is on the verge of separation; and H31-1 to H42-1 shows separation, as apparent from the skin friction (C_f) in Fig. 5.2a.



(a) Skin friction

(b) Surface pressure

Figure 5.2: Comparison of the skin friction and pressure for bumps with different bump heights from LES.



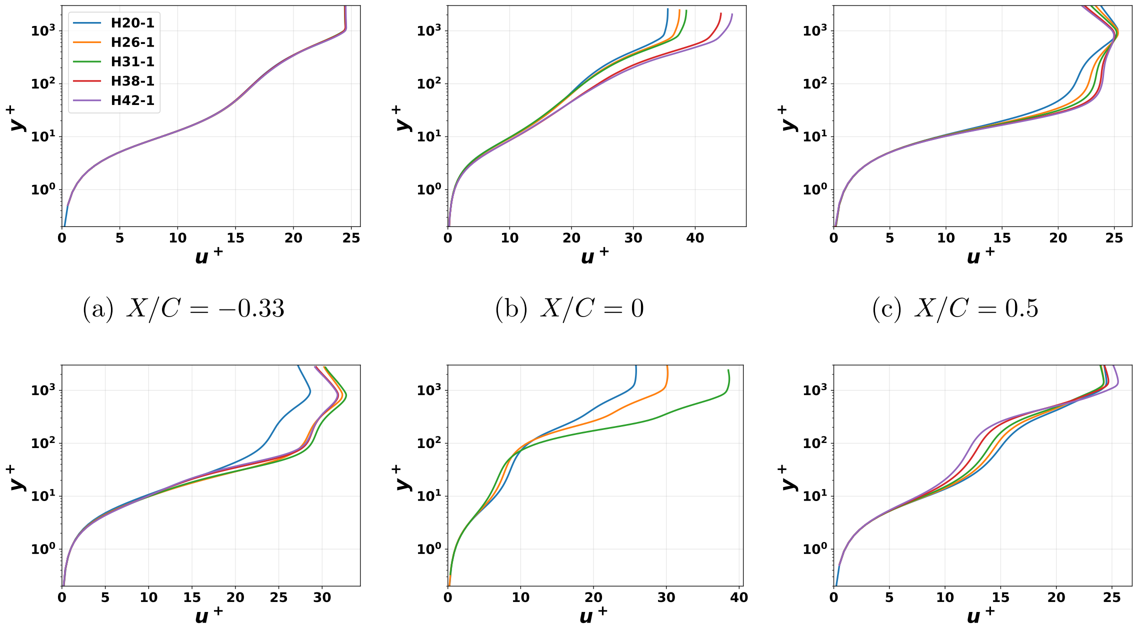
(a) Skin friction

(b) Surface pressure

Figure 5.3: Comparison of the skin friction and pressure for 3 different inlet momentum thicknesses from LES.

As the bump height is increased, C_f over the bump increases. This is attributable to a greater flow acceleration (Fig. 5.2b) over the crest. Fig. 5.4 shows the progressive changes in velocity as the bump height is increased. A similar rise in the turbulent shear-stress can be seen in Fig. 5.5. A threefold increase in turbulent shear-stress is observed advancing from H20-1 to H42-1.

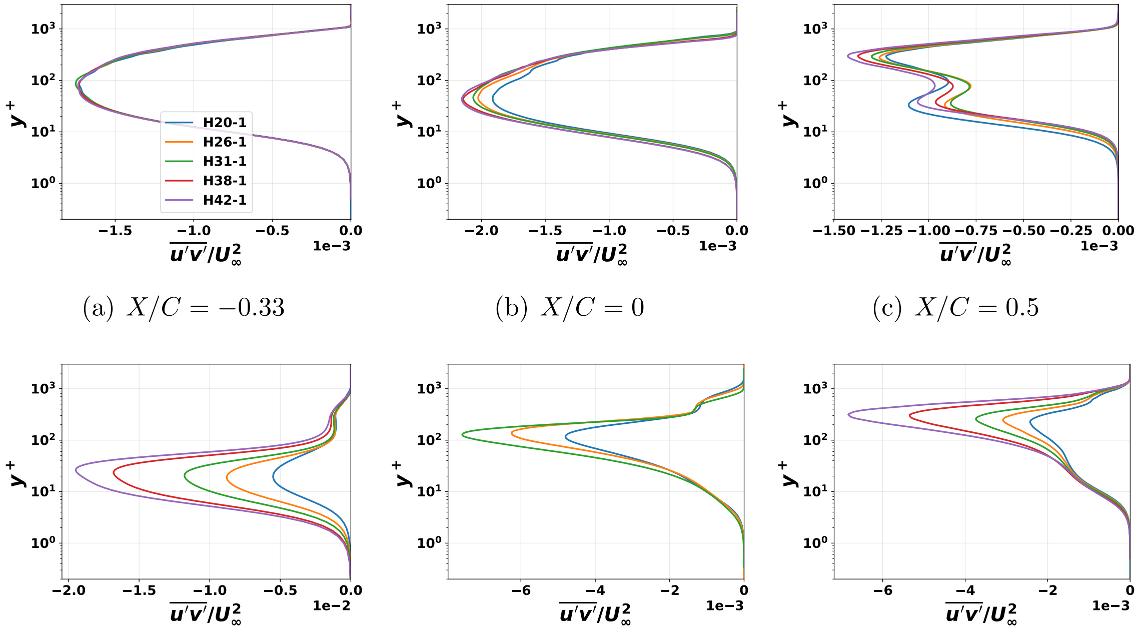
Fig. 5.6b shows that the turbulence kinetic energy (TKE) is suppressed in the favorable pressure gradient regions of the H20-1 bump and enhanced in the adverse pressure gradient regions. A similar but amplified trend can be seen over the H42-1



(a) $X/C = -0.33$ (b) $X/C = 0$ (c) $X/C = 0.5$

(d) $X/C = 0.688$ (e) $X/C = 1$ (f) $X/C = 1.33$

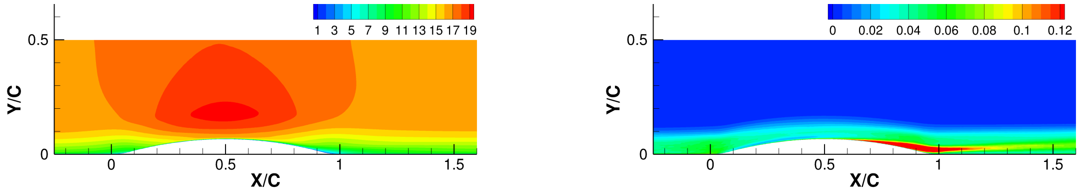
Figure 5.4: Comparison of X-velocity at different streamwise locations.



(a) $X/C = -0.33$ (b) $X/C = 0$ (c) $X/C = 0.5$

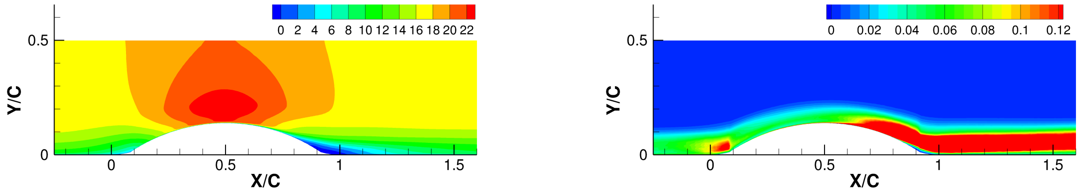
(d) $X/C = 0.688$ (e) $X/C = 1$ (f) $X/C = 1.33$

Figure 5.5: Comparison of turbulent shear-stress at different streamwise locations.



(a) U_{mean} (b) Turbulent kinetic energy

Figure 5.6: Contour plots of mean X-velocity and TKE for H20-1.



(a) U_{mean} (b) Turbulent kinetic energy

Figure 5.7: Contour plots of mean X-velocity and TKE for H42-1.

bump (Fig. 5.7b). RANS fails to capture these high TKE regions accurately. The flow separation in H42-1 case is also apparent in Fig. 5.7a, which shows a region of backflow.

5.2 Application of the FIML Framework

Transport equations for turbulence kinetic energy (TKE) k and the time scale ω are solved in the Wilcox $k-\omega$ model[113]. The discrepancy term can be introduced in either or both of these equations. Introducing discrepancy in both the equations provides more control to drive the inverse solution towards the data, but it also deteriorates the already ill-conditioned inverse problem. While there is no preference for one equation over the other, the ω equation is more ad-hoc than the k equation. Consequently, we introduce the discrepancy to the transport equation of ω. The discrepancy in the scale equation was sufficient to drive the model solution towards data.

For the baseline Wilcox $k-\omega$ model, the transport equation for ω has the following form,

$$\frac{D\omega}{Dt} = P(k, \omega, \mathbf{Q}) - D(k, \omega, \mathbf{Q}) + T(k, \omega, \mathbf{Q}), \tag{5.1}$$

where $\mathbf{Q}$ represents the Reynolds averaged conserved flow variables, P, D, and T are the production, destruction and transport terms respectively. Details about the individual terms are presented in section A.2. We introduce a spatially varying discrepancy term, $\boldsymbol{\beta}(\mathbf{x})$, to the ω transport equation, which results in

$$\frac{D\omega}{Dt} = \boldsymbol{\beta}(\mathbf{x})P(k, \omega, \mathbf{Q}) - D(k, \omega, \mathbf{Q}) + T(k, \omega, \mathbf{Q}). \tag{5.2}$$

As clarified in the previous chapter, this mode of discrepancy introduction does not restrict our ability to infer appropriate $\boldsymbol{\beta}(\mathbf{x})$ as the effect of turbulence will be negligible in the region with zero production. Inverse problems are now solved to infer the spatial distribution of the discrepancy $\boldsymbol{\beta}(\mathbf{x})$ for all 11 cases.

5.2.1 Field Inversion

Inverse problems are solved for all the cases shown in Fig. 5.1 using a skin-friction based objective function. Full-field data and surface data are available from LES, and hence for one of the cases, H20-1, we compare the inverse solutions informed by the skin-friction and the full-field velocity. Objective functions using these two datasets takes the following form

$$J(\boldsymbol{\beta}) = \sum_{i=1}^{N_{wall}} (c_{f,i} - c_{f,i,LES})^2 + \lambda \sum_{j=1}^{N_{cells}} (\boldsymbol{\beta}_j - \boldsymbol{\beta}_{j,prior}), \tag{5.3}$$

and

$$J(\boldsymbol{\beta}) = \sum_{i=1}^{N_{cells}} (u_i - u_{i,LES})^2 + \lambda \sum_{j=1}^{N_{cells}} (\boldsymbol{\beta}_j - \boldsymbol{\beta}_{j,prior}), \tag{5.4}$$

where u is the streamwise velocity and c_f is the skin friction coefficient.

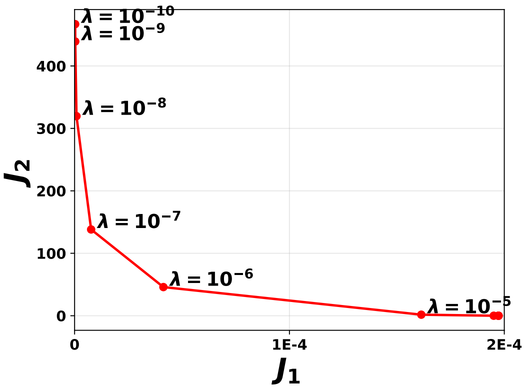
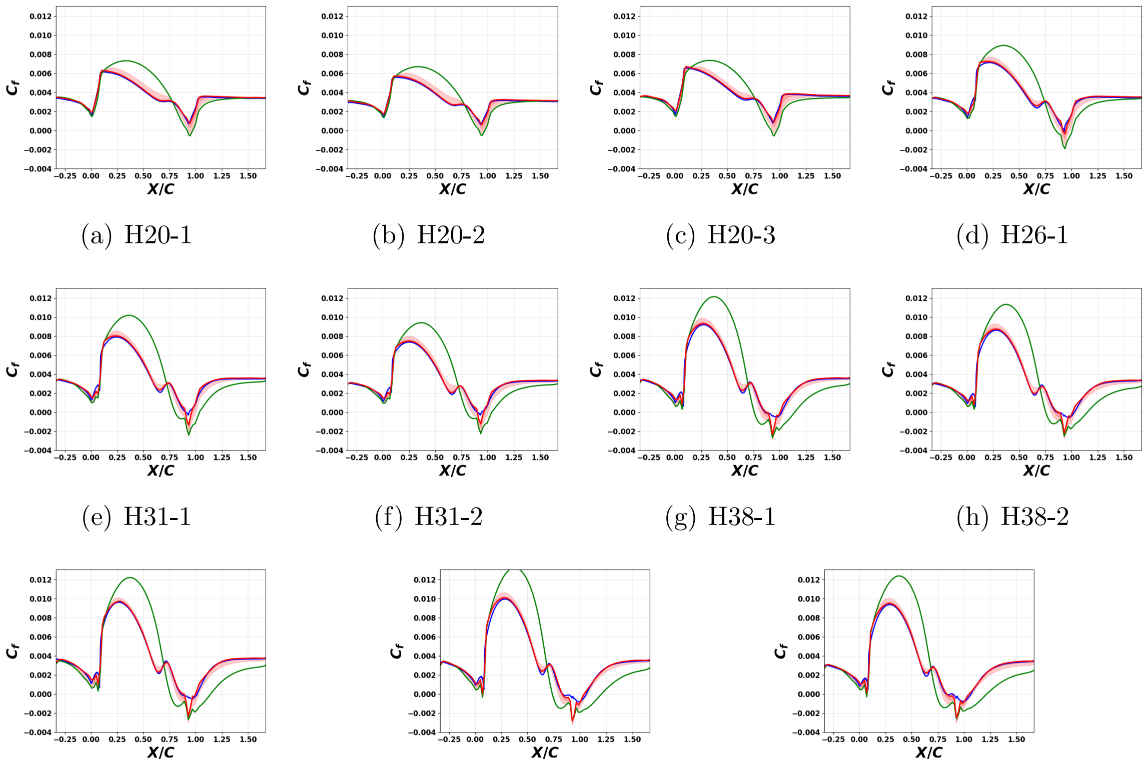


Figure 5.8: The misfit (J_1) and the regularization (J_2) terms are used to fix the regularization constant. For this representative case $\lambda_{elbow} \approx 10^{-6}$.

In the absence of any useful information regarding the uncertainty associated with the data, we solve multiple inverse problems (for each case and type of objective) to select the appropriate regularization factor λ. A very low value of λ leads to ill-conditioning (and ill-posedness) and over-fitting of the model to data, while a high λ limits model improvements. A value of λ is selected by obtaining inverse solutions for a set of λ's ($10^{-10} < \lambda < 10^{-3}$) followed by plotting the misfit term (J_1) and the regularization term (J_2) of the objective function and selecting the value of λ corresponding to the elbow. Fig. 5.8 illustrates a representative J_1 vs J_2 curve which results in $\lambda_{elbow} = 10^{-6}$.

The inferred discrepancy fields using the skin friction based inversion are shown in Fig. 5.10 and the corresponding solutions of the skin-friction are shown in Fig. 5.9. The forward section of the bump results in a favorable pressure gradient, which stabilizes the turbulent flow. The inverse solution captures this stabilization by increased production of ω (regions with $\boldsymbol{\beta}(\mathbf{x}) > 1$). The downstream section is dominated by an increase in turbulence with $\boldsymbol{\beta}(\mathbf{x}) < 1$. The increase is due to enhanced mixing which is consistent with the LES as seen in Figs. 5.6b and 5.7b. The resulting skin friction is almost identical to the LES data and displays the same degree of separation as the LES. The shaded regions in Fig. 5.9 contain a number of inverse solutions with



(a) H20-1 (b) H20-2 (c) H20-3 (d) H26-1

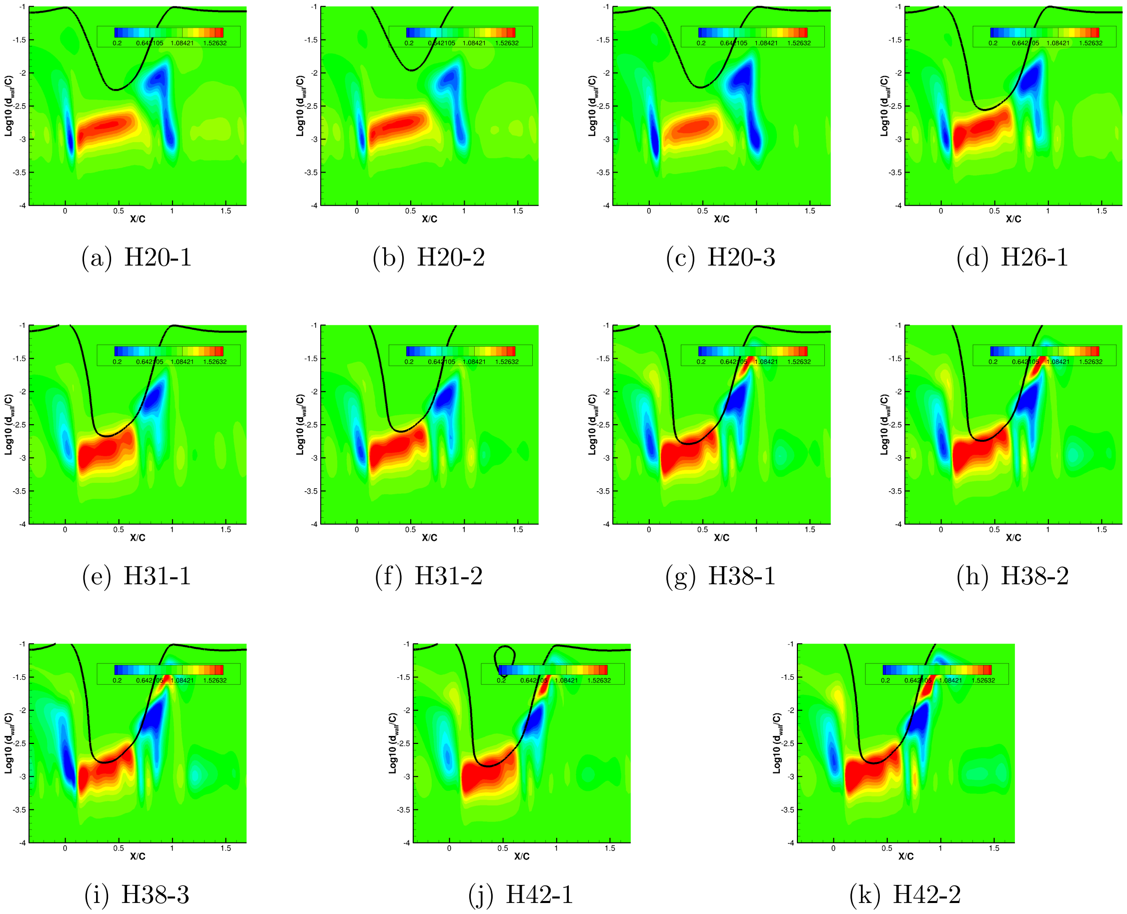
(e) H31-1 (f) H31-2 (g) H38-1 (h) H38-2

(i) H38-3 (j) H42-1 (k) H42-2

Figure 5.9: Skin friction for all the 11 cases. Shaded red region contains inverse solutions for various $10^{-10} < \lambda < 10^{-6}$. Legend: — LES, — base $k-\omega$ and — inverse $k-\omega$ using C_f.

$\lambda \in [10^{-10}, 10^{-3}]$ which are used to select the optimal value λ. The solutions are moderately sensitive to the regularization constant.

All the discrepancy fields in Fig. 5.10 have a qualitatively similar spatial distribution. This trend can be exploited to suggest an analytical correction. However, ML algorithms view data more comprehensively and can construct more complex models than a human modeler. The details about the ML training and prediction to reconstruct the discrepancy as a function of local flow quantities are discussed in the section 5.2.2.



(a) H20-1 (b) H20-2 (c) H20-3 (d) H26-1

(e) H31-1 (f) H31-2 (g) H38-1 (h) H38-2

(i) H38-3 (j) H42-1 (k) H42-2

Figure 5.10: Inferred spatial discrepancy field $\boldsymbol{\beta}(\mathbf{x})$ using skin friction data. The thick black line marks the boundary layer edge.

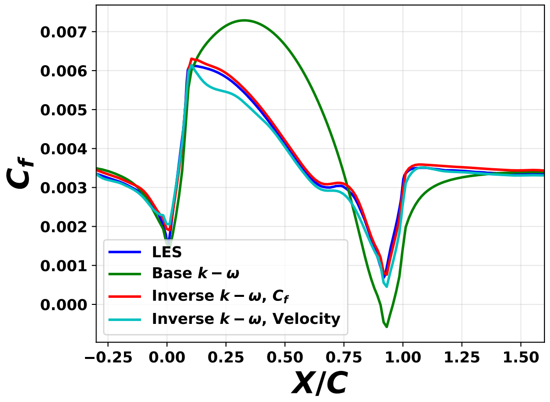
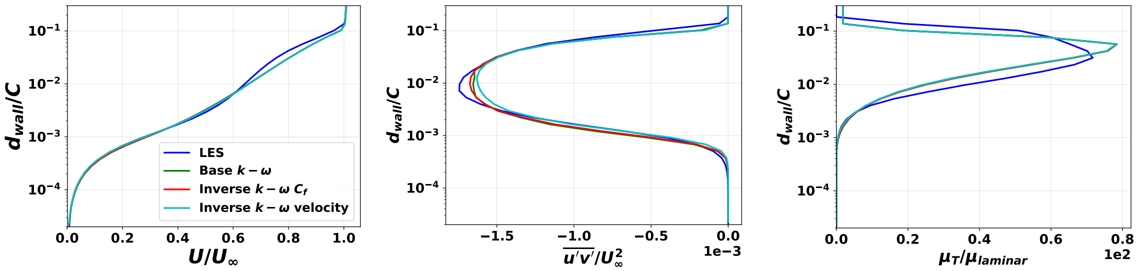


Figure 5.11: Comparison of skin friction obtained after inference using data for the skin friction and the full-field velocity.

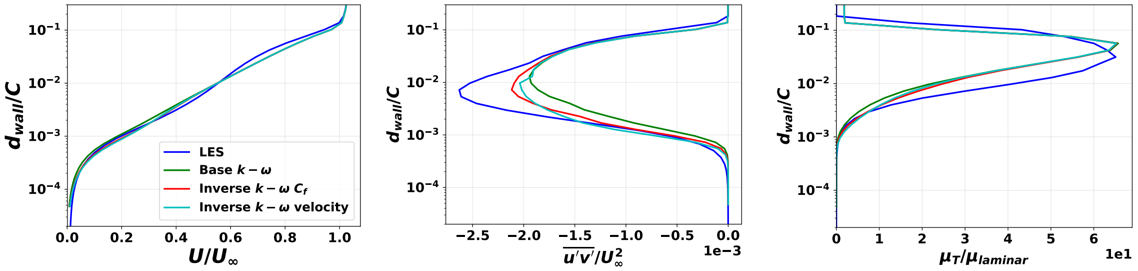
5.2.1.1 Inverse solution using C_f vs. full-field velocity

To investigate the effect of data-type used for inversion, an inverse problem was solved for H20-1 using the full-field velocity data (using the objective function in equation 5.4). Fig. 5.11 compares the effect of using skin friction and the full field velocity on the inverse skin friction. Figs. 5.12-5.17 compares the velocity, Reynolds shear-stress and eddy-viscosity profiles at different stream-wise locations. The similarity observed between the two inverse solutions justifies the usage of skin-friction data for inversion. When using skin-friction as data, improvements in the field quantities build confidence in the inverse solution. Notably, inversion doesn't just imply over-fitting the model to the data but can also infer a more "global" discrepancy.



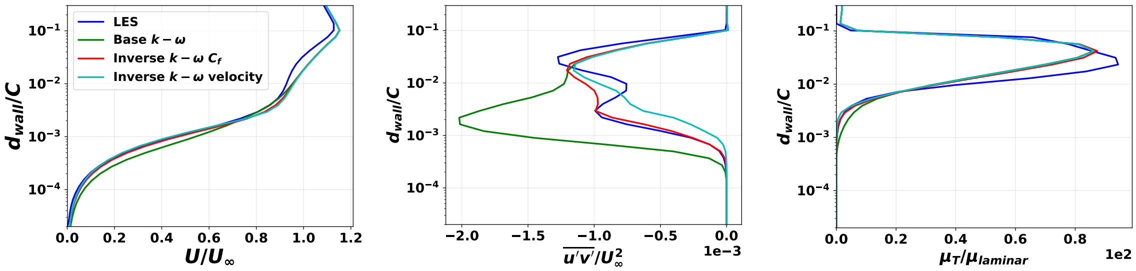
(a) Velocity (b) Reynolds shear-stress (c) eddy-viscosity

Figure 5.12: Flow solution at $X/C = -0.16$ obtained after inference using data for the skin friction and the full field velocity.



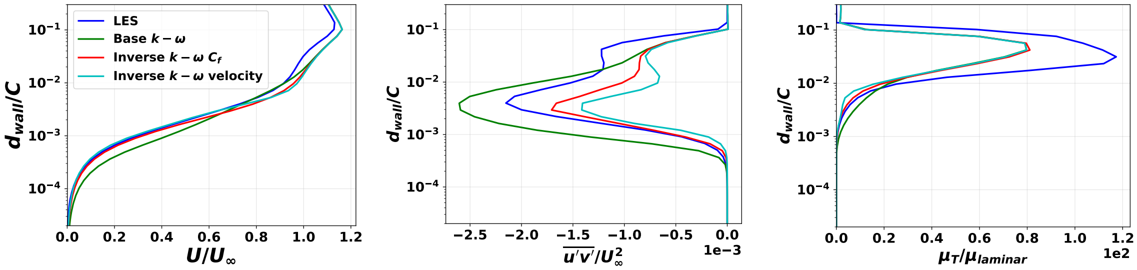
(a) Velocity (b) Reynolds shear-stress (c) eddy-viscosity

Figure 5.13: Flow solution at $X/C = 0$ obtained after inference using data for the skin friction and the full field velocity.



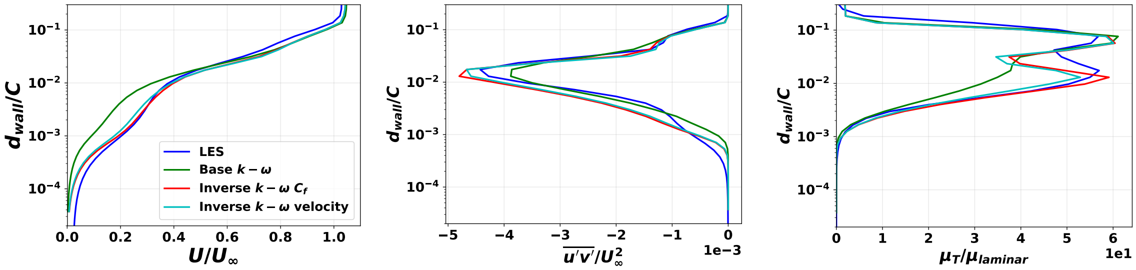
(a) Velocity (b) Reynolds shear-stress (c) eddy-viscosity

Figure 5.14: Flow solution at $X/C = 0.33$ obtained after inference using data for the skin friction and the full field velocity.



(a) Velocity (b) Reynolds shear-stress (c) eddy-viscosity

Figure 5.15: Flow solution at $X/C = 0.66$ obtained after inference using data for the skin friction and the full field velocity.

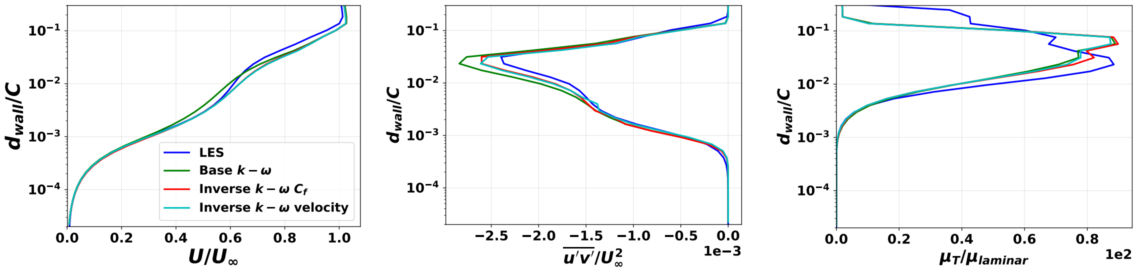


(a) Velocity (b) Reynolds shear-stress (c) eddy-viscosity

Figure 5.16: Flow solution at $X/C = 0.98$ obtained after inference using data for the skin friction and the full field velocity.

5.2.2 Machine Learning Training

In line with Chapter IV, the AdaBoost algorithm is used to reconstruct the discrepancy field as a function of local flow features using a 2-fold CV. We use the



(a) Velocity (b) Reynolds shear-stress (c) eddy-viscosity

Figure 5.17: Flow solution at $X/C = 1.31$ obtained after inference using data for the skin friction and the full field velocity.

following flow-features for training

$$\eta_1 = \rho|S|d_{wall}^2/\mu_L, \tag{5.5}$$

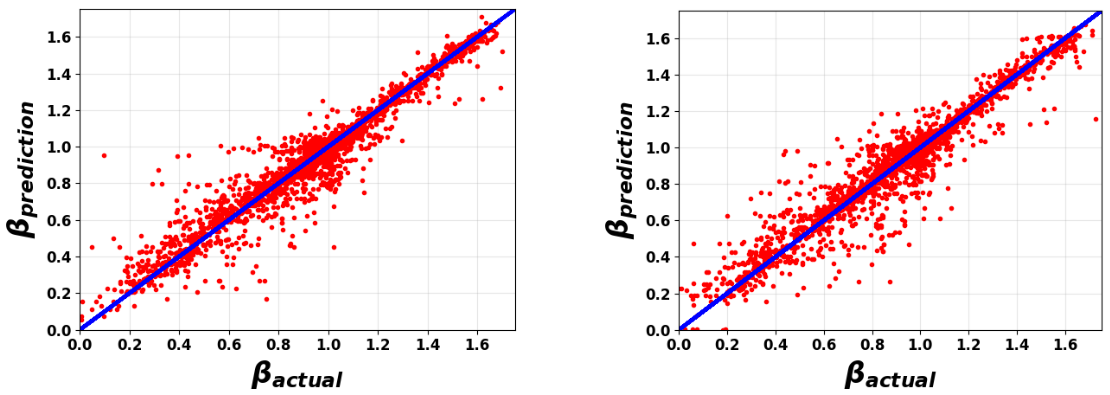
$$\eta_2 = \mu_T|S|/\tau_{wall}, \tag{5.6}$$

$$\eta_3 = d_{wall}, \tag{5.7}$$

where τ_{wall} is the shear-stress at the nearest wall, d_{wall} is the distance to the nearest wall, and $|S|$ is the local strain rate magnitude. Different combinations of the 11 inverse solutions are used to train an ensemble of models. These models are then used in the RANS solver for predictions. Fig. 5.18 shows the training plot for a model trained on inverse solutions for the H20-1 and the H26-1 case (model **P**, as marked in Fig. 5.1). Coefficients of determination R^2 for testing of the two folds are 0.92 and 0.91, which indicates a good training.

5.2.3 Machine Learning Prediction

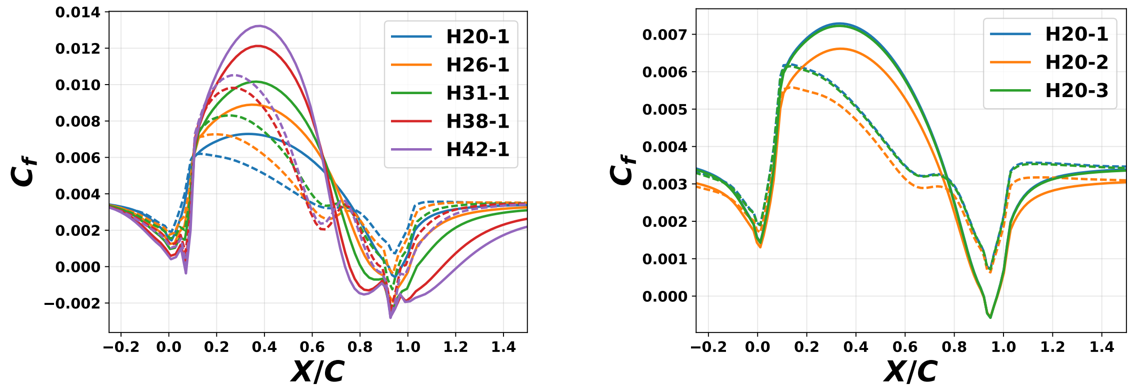
The trained ML-based models are embedded in an iterative RANS solver for predictions. The predicted skin friction for all the 11 cases are shown in Figs. 5.19 and 5.20. Figs. 5.21-5.26 show the velocity, Reynolds shear-stress, and eddy-viscosity at different streamwise locations. Model **P** results in the best prediction over all the



(a) Fold 1: Testing

(b) Fold 2: Testing

Figure 5.18: Test results for model **P** using 2-fold CV and the AdaBoost algorithm.



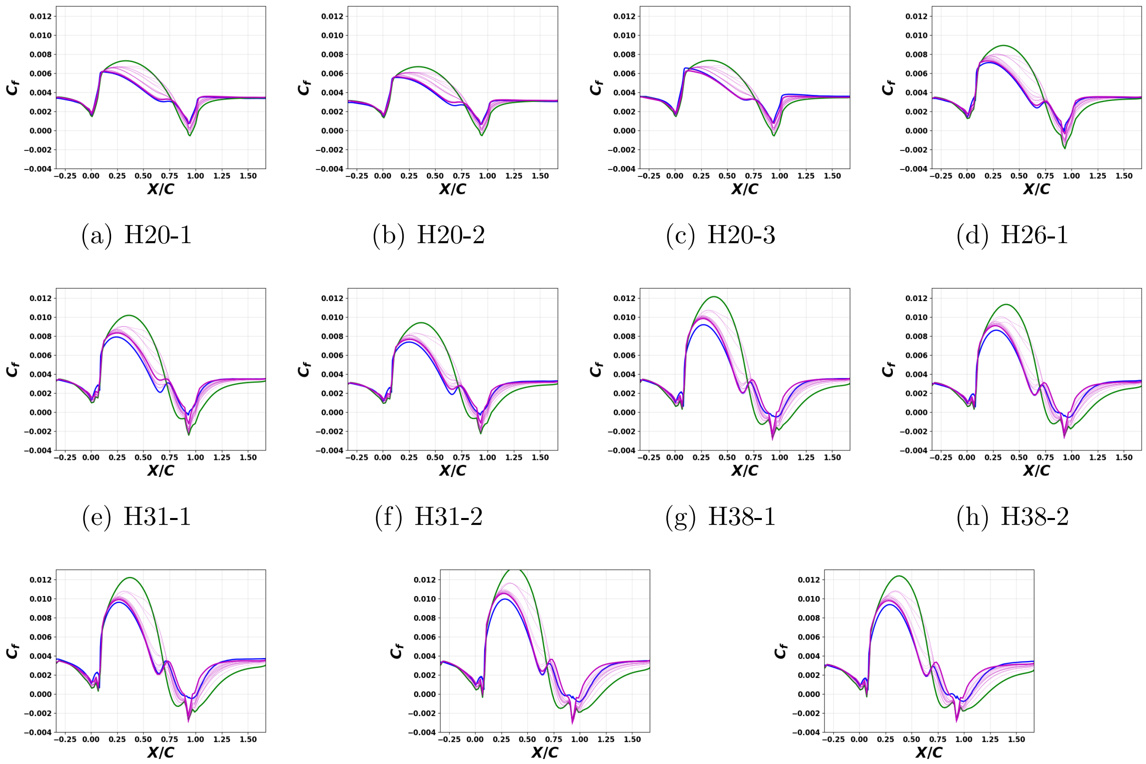
(a) Bump height variation

(b) Inlet momentum thickness variation

Figure 5.19: Skin friction predictions using the baseline $k - \omega$ (solid line) and the AdaBoost-augmented model **P** (dashed line).

cases. Just by using the skin friction data in two cases, improvements are observed in the full field quantities in the inner layer. The quality of predictions using other ML models varies but demonstrates consistent improvements over the baseline model.

A more rigorous analysis is required to determine a priori the quality of a ML model trained on a given combination. Nevertheless, some insight can be obtained by examining the pairwise scatter plots of the flow features (Fig. 5.27). The points in green, which are used to train model **P**, are a good proxy of the entire data-set shown in red. While using the entire inverse data-set would theoretically produce the most

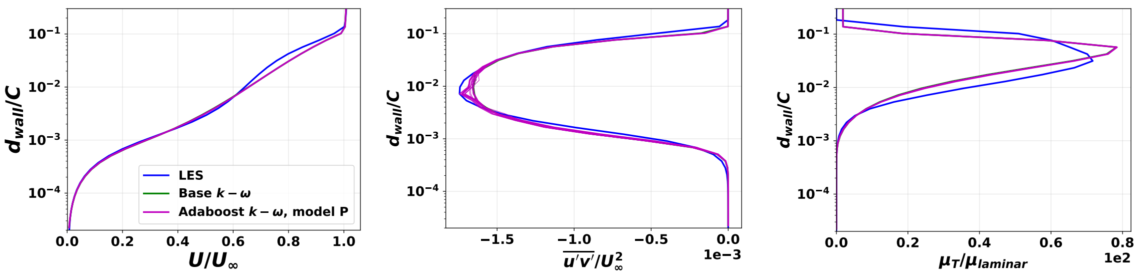


(a) H20-1 (b) H20-2 (c) H20-3 (d) H26-1

(e) H31-1 (f) H31-2 (g) H38-1 (h) H38-2

(i) H38-3 (j) H42-1 (k) H42-2

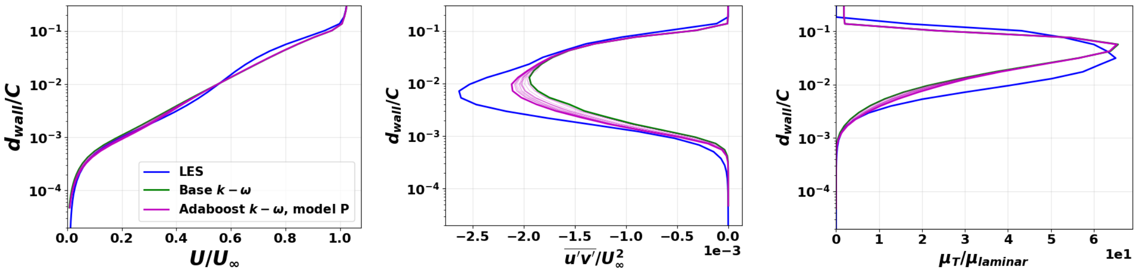
Figure 5.20: Skin friction prediction for all the 11 cases. Thin magenta lines represent predictions using an ensemble of machine-learned models trained on different combinations of the inverse solutions. Legend: — LES, — base $k-\omega$ and — AdaBoost augmented $k-\omega$ using model **P**.



(a) Velocity (b) Reynolds shear-stress (c) eddy-viscosity

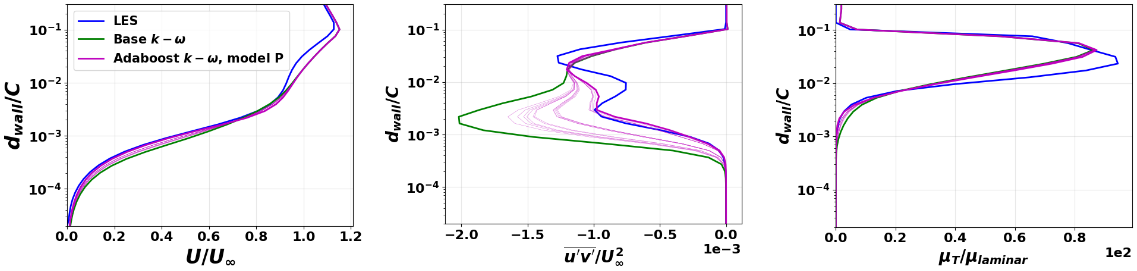
Figure 5.21: Flow solution at $X/C = -0.16$ using AdaBoost augmented model **P**. Thin magenta lines represent predictions using an ensemble of machine-learned models trained on different combinations of the inverse solutions.

optimal ML model, reducing the number of data points results in a better training of the ML model.



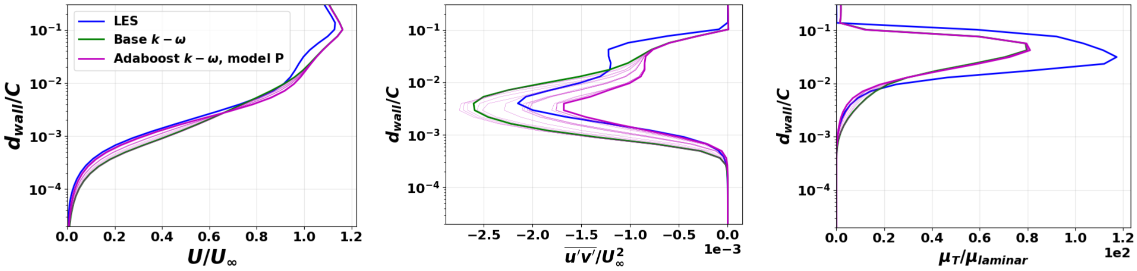
(a) Velocity (b) Reynolds shear-stress (c) eddy-viscosity

Figure 5.22: Flow solution at $X/C = 0$ using AdaBoost augmented model **P**. Thin magenta lines represent predictions using an ensemble of machine-learned models trained on different combinations of the inverse solutions.



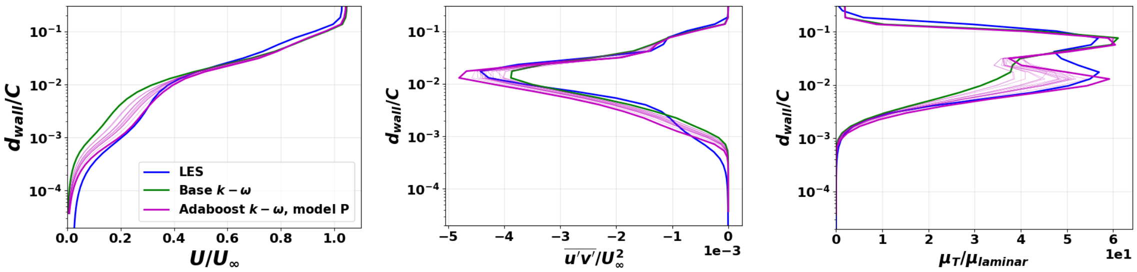
(a) Velocity (b) Reynolds shear-stress (c) eddy-viscosity

Figure 5.23: Flow solution at $X/C = 0.33$ using AdaBoost augmented model **P**. Thin magenta lines represent predictions using an ensemble of machine-learned models trained on different combinations of the inverse solutions.



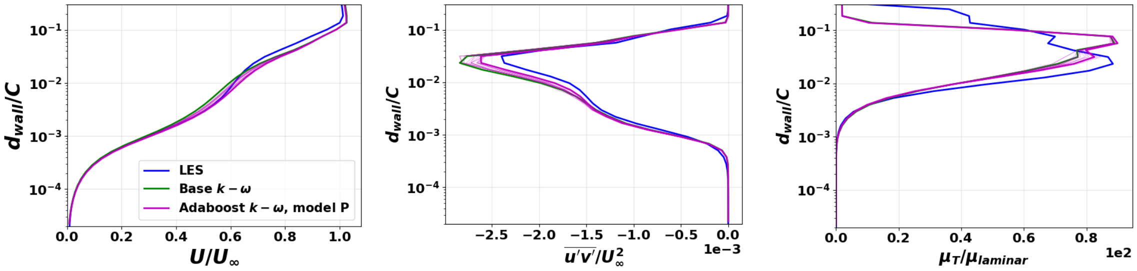
(a) Velocity (b) Reynolds shear-stress (c) eddy-viscosity

Figure 5.24: Flow solution at $X/C = 0.66$ using AdaBoost augmented model **P**. Thin magenta lines represent predictions using an ensemble of machine-learned models trained on different combinations of the inverse solutions.



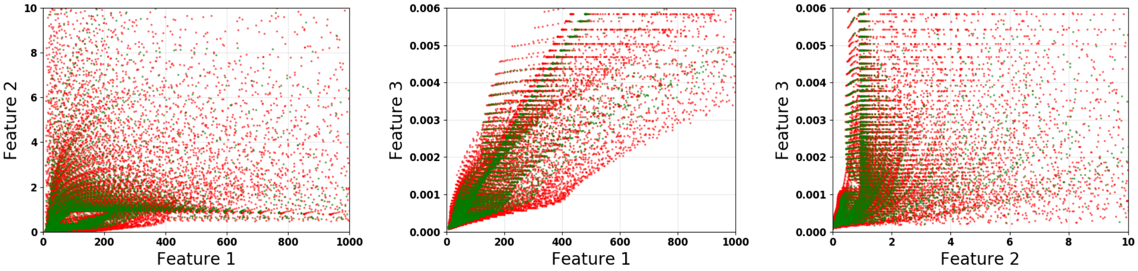
(a) Velocity (b) Reynolds shear-stress (c) eddy-viscosity

Figure 5.25: Flow solution at $X/C = 0.98$ using AdaBoost augmented model **P**. Thin magenta lines represent predictions using an ensemble of machine-learned models trained on different combinations of the inverse solutions.



(a) Velocity (b) Reynolds shear-stress (c) eddy-viscosity

Figure 5.26: Flow solution at $X/C = 1.31$ using AdaBoost augmented model **P**. Thin magenta lines represent predictions using an ensemble of machine-learned models trained on different combinations of the inverse solutions.



(a) (b) (c)

Figure 5.27: Pairwise scatter plots of the flow features used for training. Features for all the cases are shown in red and the features used to train model **P** are shown in green.

5.3 Summary

- The FIML framework was applied to augment the Wilcox $k-\omega$ model for a set of flows over bumps of varying height and inlet conditions. In this second test case, data from LES was used. In contrast to almost perfect data in the previous chapter, the LES data contains some level of uncertainty. Therefore, a more rigorous approach was taken to estimate the regularization constant.

- Inverse problems were solved using skin-friction and full-field velocity as data. Both the variations exhibit similar improvements in the inverse solution. The inferred discrepancy fields displayed universality across all the inverse cases. The discrepancy fields were also found to be consistent with the expectations based on flow physics.

- Two inverse solutions were used to train a machine-learning (ML) model $\mathbf{P}$, which was used for prediction for all the cases. The prediction using the model $\mathbf{P}$ was significantly more accurate than the baseline model. Using the skin friction data also led to notable improvements in the velocity and Reynolds shear-stress predictions.

- Ensembles of models were trained on multiple combinations of the inverse solutions. The spread in prediction using these models reflects the uncertainty associated with the ML-based augmentation based on limited data. All these models were found to be more accurate than the baseline $k-\omega$ model.

CHAPTER VI

Application to Separated Flows over Airfoils

Airfoil sections form the core elements of any lifting surface. Prediction of aerodynamic forces over airfoils is complicated by the transition of the flow from laminar to turbulent, the presence of adverse pressure gradients (APG) in the diffusive portion of the airfoil, and flow-separation at high angles of attack. Turbulent flow separation over lifting surfaces is critical to many applications, including high-lift systems, the off-design operation of new vehicles, airframe noise, wind turbines, turbomachinery flows, and combustors. A RANS turbulence modeling capability that can confidently predict separated flows in these various contexts would be a key enabling factor in the development of aerospace and energy systems of the future. The ability to accurately model the effects of strong APGs is crucial to the prediction of boundary layer separation in wall-bounded flows; however, most one- and two-equation RANS turbulence models fail to accurately predict stall onset for airfoils at high angles of attack, where strong APG is encountered. Consequently, they tend to over-predict the maximum lift and stall onset angle for a given set of flow conditions.

Fig. 6.1 shows the prediction of lift for an S809 airfoil at a Reynolds number of 2 Million. The Spalart-Allmaras (SA) turbulence model is used for the RANS prediction. At low angles of attack ($\alpha < 6^\circ$) the prediction and the experimental measurements are in agreement, but they diverge as the flow starts to separate with

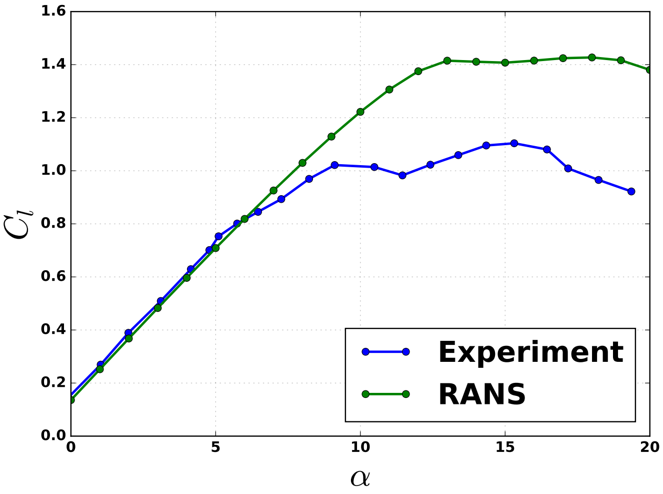


Figure 6.1: Lift vs. angle of attack plot for the S809 airfoil at a Reynolds number of 2 Million.

increasing α. Linear eddy-viscosity based models have been observed to overpredict eddy-viscosity, which results in a delayed separation. Previous works[13, 121, 117] support the observed failure of the RANS-based models to predict the onset of separation and maximum lift accurately.

In this chapter, the FIML framework is applied to augment the SA turbulence model. Experimental lift and surface pressure measurements are utilized to infer discrepancy in the SA model. Inverse problems are solved for multiple airfoils at different angles of attack and Reynolds numbers. Inverse solutions are used to train ML-based augmentations. ML-based augmentations are then used for prediction. Portability of the ML-based augmentations is demonstrated by embedding and using them for prediction in a commercial flow solver. Effects of grid refinement and a measure of uncertainty associated with ML-based augmentation are discussed.

6.1 Forward Problem

The problem setup is based on a set of experiments conducted at the National Renewable Energy Laboratory (NREL) in 1997 [95, 96, 97]. The experiments mea-

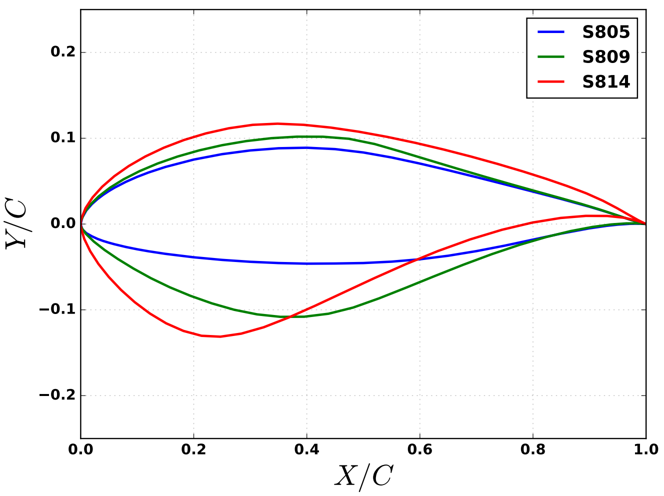


Figure 6.2: Three different airfoil shapes are used for training and testing the ML-based augmentation.

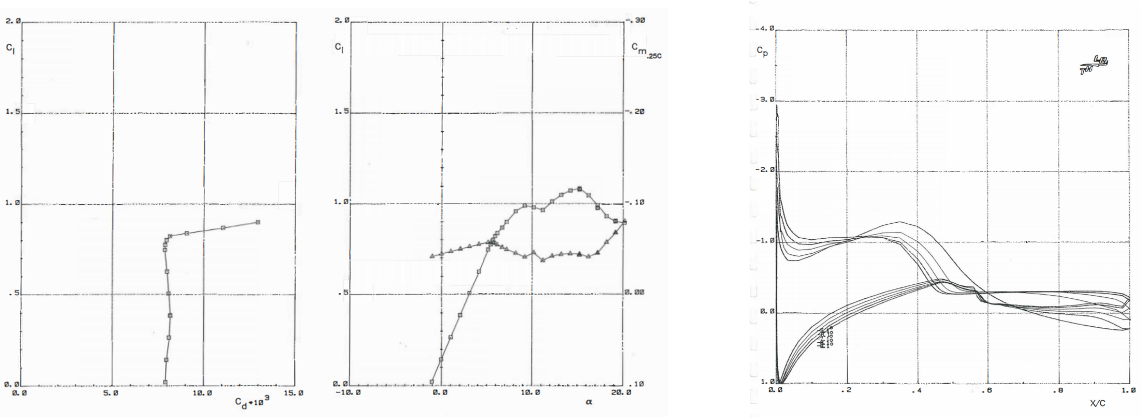


Figure 6.3: Samples of figures used to extract data for this work. These figures are reproduced from [96].

sured aerodynamics forces for three airfoil shapes (S805, S809, and S814) belonging to the Somers (S-series) family. The maximum thickness of the airfoils varies from 5% to 14% of the chord length. Pressure, lift, and drag measurements are reported for the experiments. Fig. 6.3 shows plots from the experiment reports which are used to extract the data. Errors are introduced during the data extraction process which are in addition to any unpublished.

The forward problem consists of solving for the flow around the S-series airfoils and then uses the data from the experiments for comparison and model augmentation. For

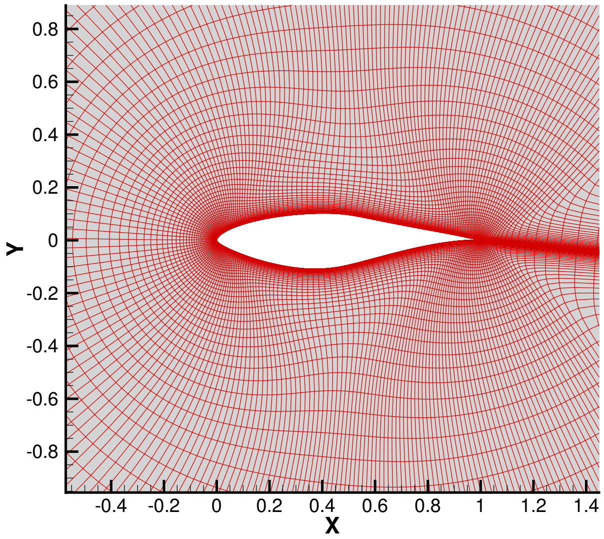


Figure 6.4: A body fitted C-mesh is used with 291 points in the wrap around direction and 111 points in the perpendicular direction.

the computations, a chord-length of unity is used, and the Reynolds number is varied from 1 to 3 Million. The mesh is resolved sufficiently to ensure a $y^+ < 1$ at the airfoil surface and 200 grid points are used in the wrap-around direction. Fig. 6.4 shows a representative mesh. A Mach number of 0.1 is used to emulate near incompressible flow. The flow solver is used to drive the flow to a steady state starting from a free-stream based initial condition.

Figs. 6.5 and 6.6 show the surface pressures and velocity fields, using the baseline SA model, for the S809 airfoil. There are significant differences between the experimental pressures and those calculated using the baseline SA model. These differences manifest in the lift output, as shown in Fig. 6.1, where the maximum error is over 40%.

The error in aerodynamic force renders RANS an ineffective airfoil design tool especially for testing at off-design conditions. Any projection based on the baseline SA solution is bound to contain inaccuracies. These shortcomings provide an opportunity for model improvements.

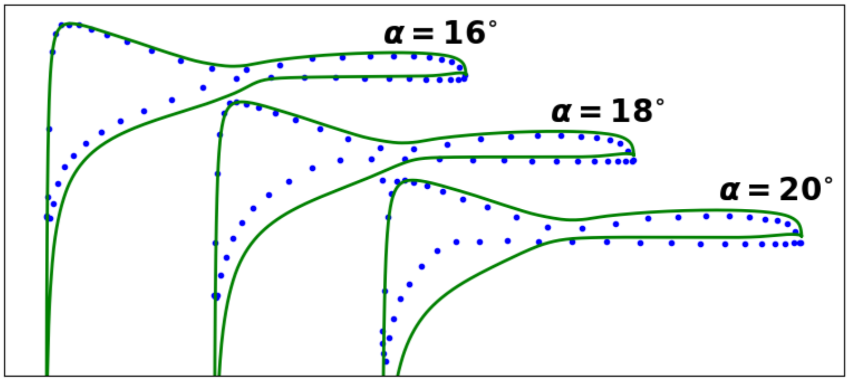
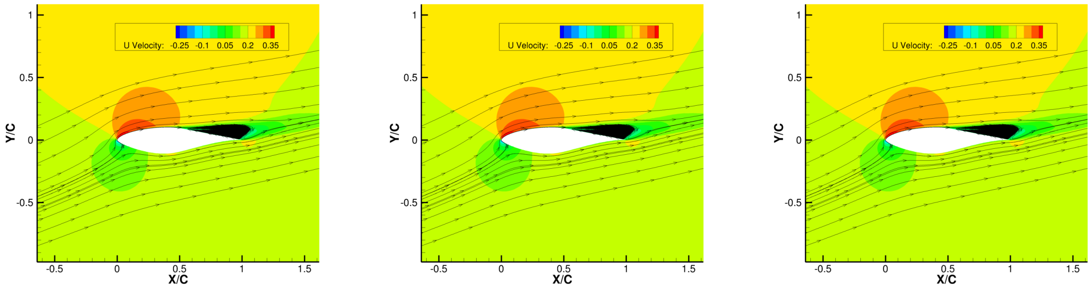


Figure 6.5: Surface pressure using the baseline SA model (green) and experiment (blue) for the S809 airfoil at $Re = 2$ Million.



(a) $\alpha = 16^\circ$

(b) $\alpha = 18^\circ$

(c) $\alpha = 20^\circ$

Figure 6.6: Velocity fields using the baseline SA model for the S809 airfoil at $Re = 2$ Million.

6.2 Application of the FIML framework

Following the setup in Chapter IV, a discrepancy function $\boldsymbol{\beta}(\mathbf{x})$ is introduced in the SA turbulence model. The SA model then takes the following form:

$$\frac{D\tilde{\nu}}{Dt} = \boldsymbol{\beta}(\mathbf{x})P(\tilde{\nu}, \mathbf{Q}) - D(\tilde{\nu}, \mathbf{Q}) + T(\tilde{\nu}, \mathbf{Q}) \tag{6.1}$$

where P, D, and T are the production, dissipation, and turbulent transport terms respectively. Inverse problems are solved to infer spatial distributions of the discrepancy. Following this, the neural-network (NN) algorithm is used to build a generalized model of the discrepancy, which is then used for prediction. The two steps are discussed in the following sections.

6.2.1 Field Inversion

A spatial distribution of the discrepancy $\boldsymbol{\beta}(\mathbf{x})$ is inferred by minimizing the difference between the prediction and the experimental data. The inverse problem provides a generalized approach to connect the model discrepancy and data, even when a direct relationship does not exist. The data is available in the form of surface pressure and lift coefficient for each angle of attack and Reynolds number combination. The objective functions, when using these two different datasets, have the following form:

$$J(\boldsymbol{\beta}) = (c_l - c_{l,exp})^2 + \lambda \sum_{j=1}^{N_{cells}} (\boldsymbol{\beta}_j - \boldsymbol{\beta}_{j,prior}), \tag{6.2}$$

and

$$J(\boldsymbol{\beta}) = \sum_{i=1}^{N_{wall}} (c_{p,i} - c_{p,i,exp})^2 + \lambda \sum_{j=1}^{N_{cells}} (\boldsymbol{\beta}_j - \boldsymbol{\beta}_{j,prior}). \tag{6.3}$$

where c_l is the lift coefficient and $c_{p,i}$ is the coefficient of pressure at the wall grid point i. The regularization factor λ is fixed to 10^{-4} for both cases. This relatively higher factor points to a low-confidence in the data in comparison with the high-quality

Airfoil	*Reynolds number (Re)*	*Angle of attack (α)*
S805	1 & 2 Million	$\{12°, 14°, 16°, 18°\}$
S809	1 & 2 Million	$\{12°, 14°, 16°, 18°, 20°\}$
S814	1 & 2 Million	$\{12°, 14°, 16°, 17°\}$

Table 6.1: List of airfoil shapes and flow-conditions for which inverse problems are solved.

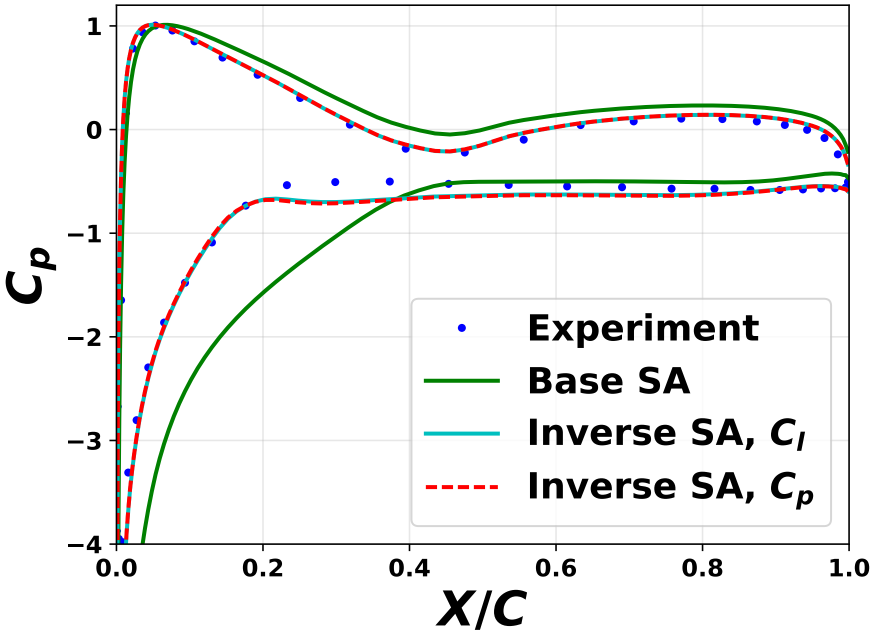
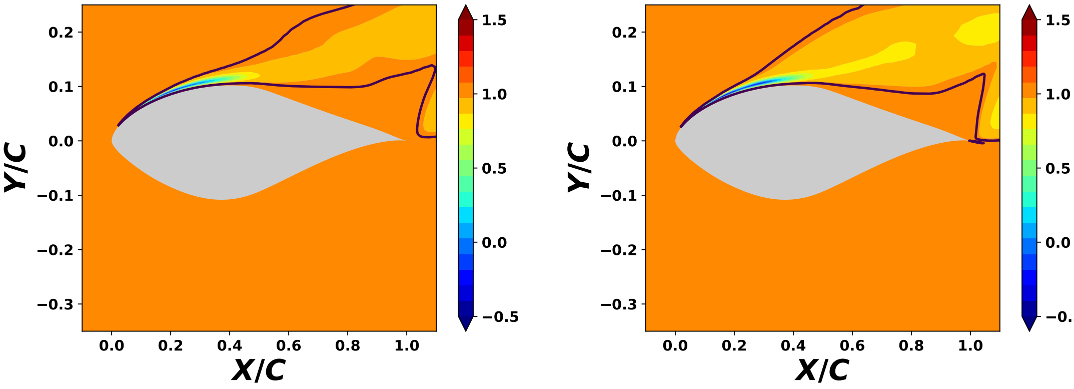


Figure 6.7: Surface pressure for the S809 airfoil at $Re = 2$ Million and $\alpha = 14°$.

synthetic and LES data used in the previous chapters. This value of λ is estimated by approximating the experimental error and the error in the prior model.

Inverse problems are solved for all three airfoils at post separation angles of attack and for two different Reynolds numbers. Inverse problems are not solved for pre-separation angles of attack as the baseline solution and the experiments are within the margin of experimental error, and any attempt to infer the $\boldsymbol{\beta}$ fields for such cases will lead to solutions dominated by numerical inaccuracies. All the inverse cases are tabulated in Table 6.1.

Fig. 6.8 shows the inferred $\boldsymbol{\beta}(\mathbf{x})$ fields for the S809 airfoil at $\alpha = 19°$ and $Re = 2$



(a) Data: pressure

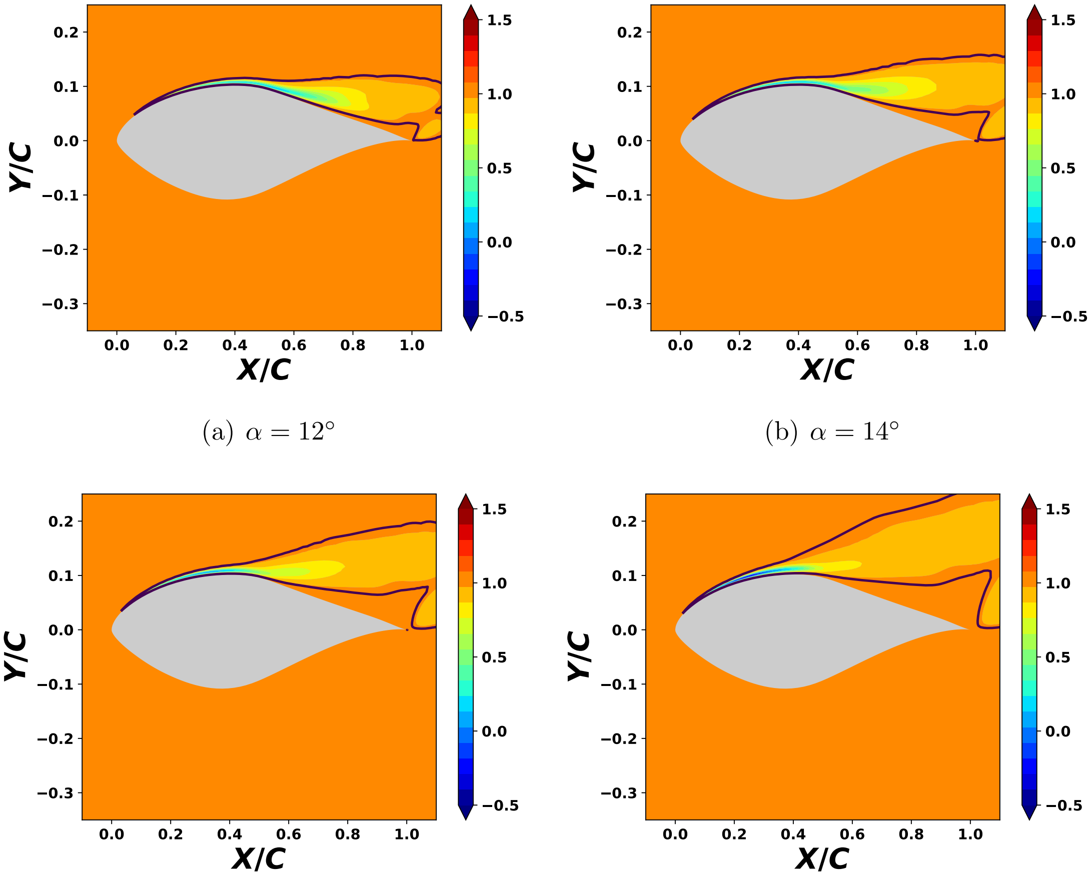
(b) Data: lift

Figure 6.8: Inferred discrepancy using different data types for the S809 airfoil at $\alpha = 19°$ and Re = 2 Million. The discrepancy deviates from the baseline value of unity only inside the region marked with the black curve.

Million using the two different datasets: surface pressure and the lift. When using data for the lift, just one number is used to infer a field of thousands of parameters. However, the two inferred fields are nearly identical. Moreover, this similarity results in near identical surface pressure distributions as shown in Fig. 6.7. This agreement can be attributed to a one to one relationship between the lift and the surface pressure for steady flows.

The similarity of the inferred field using the surface pressure and lift open up the possibility of using only the lift data to perform the inversion and hence augment a turbulence model. An enormous amount of experimental lift (and drag) data exists going back several decades and can be used for the augmentation of turbulence models. These experimental datasets are available for practical Reynolds numbers in contrast with restricted DNS or LES data.

The inverse solution is close to the experimental results (Fig. 6.7) and also predicts the point of separation accurately. The point of flow separation is approximately marked by the flattening of the surface pressure curve. Similar improvements are observed for all airfoil shapes, angles of attack, and Reynolds numbers. Fig. 6.10 shows the surface pressure profiles at different angles of attack for the S809 airfoil at



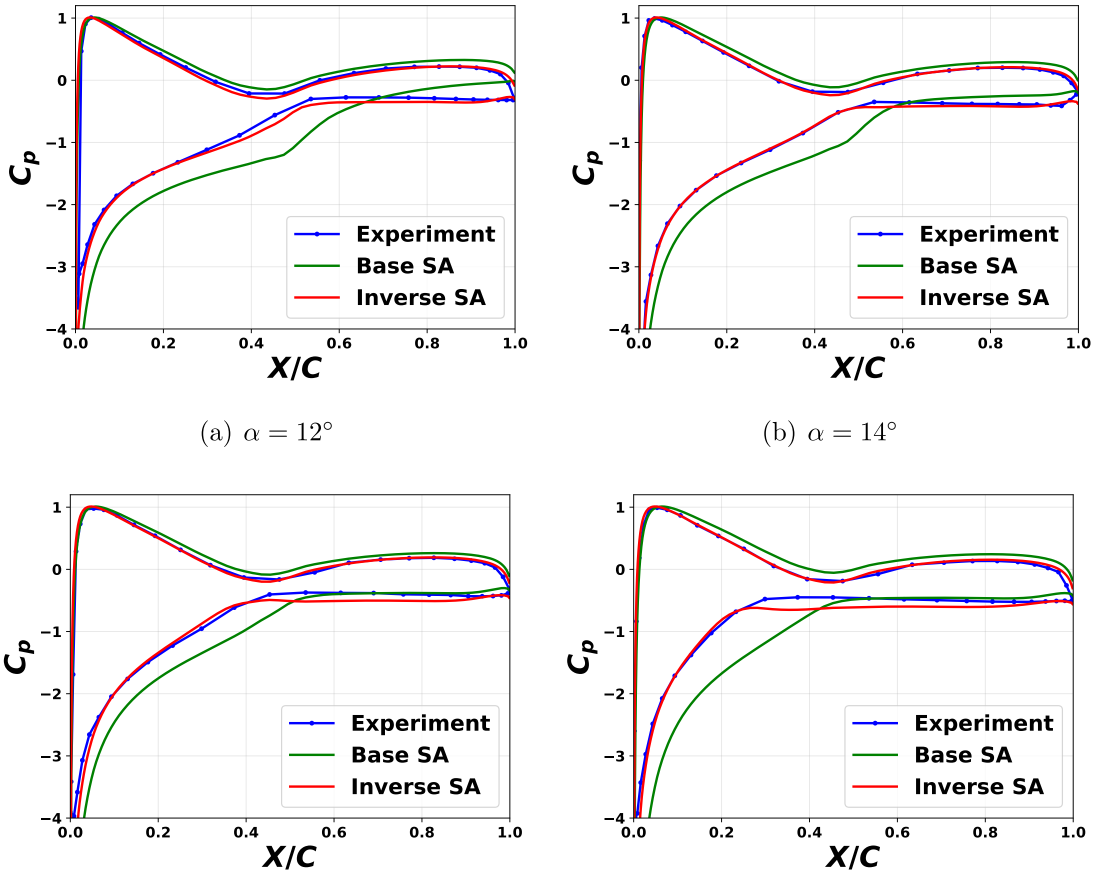
(a) $\alpha = 12°$

(b) $\alpha = 14°$

(c) $\alpha = 16°$

(d) $\alpha = 18°$

Figure 6.9: Inferred discrepancy fields for the S809 airfoil at Re = 2 Million. The discrepancy deviates from the baseline value of unity only inside the region marked with the black curve.



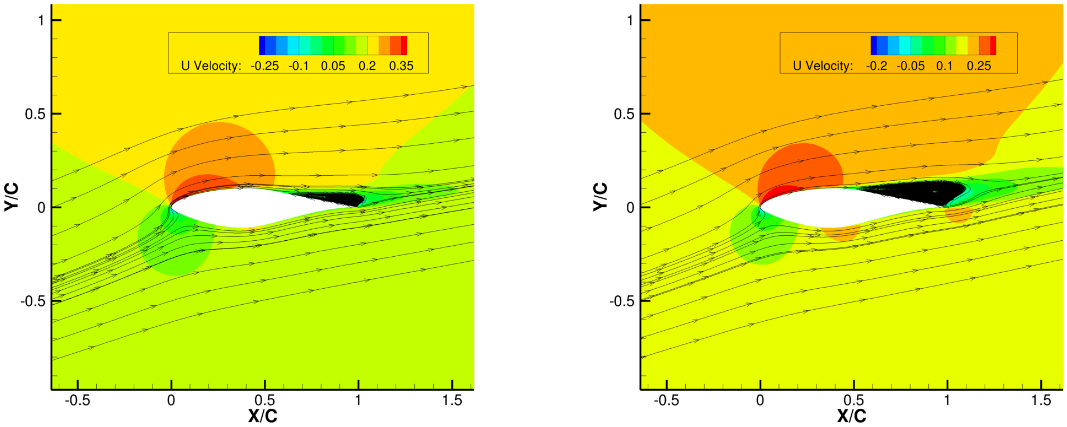
(a) $\alpha = 12°$

(b) $\alpha = 14°$

(c) $\alpha = 16°$

(d) $\alpha = 18°$

Figure 6.10: Surface pressure obtained using inferred discrepancy fields for the S809 airfoil at Re = 2 Million.



(a) Base SA

(b) Inverse SA

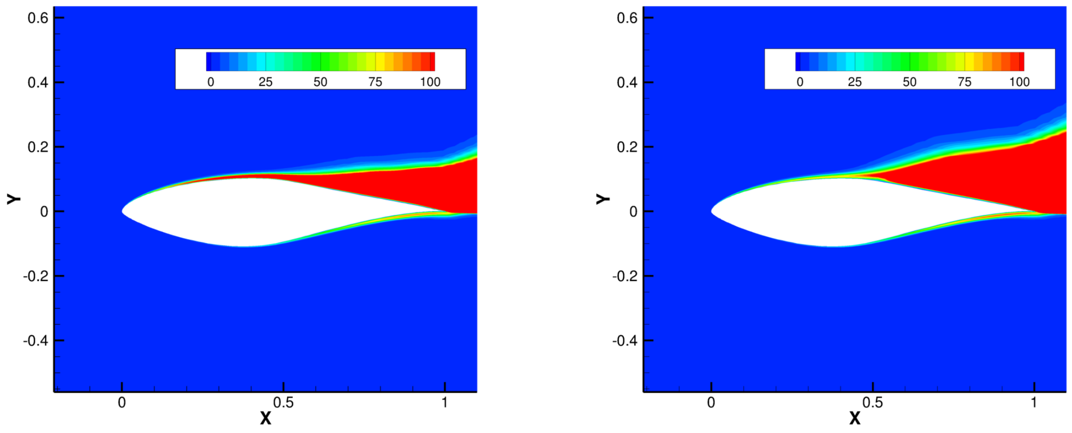
Figure 6.11: Streamline and contour plots of the X-velocity for the S809 airfoil at $Re = 2$ Million and $\alpha = 14°$.

a Reynolds number of 2 Million. Fig. 6.9 shows the inferred discrepancy fields. There are qualitative similarities between the inferred discrepancy for all the cases.

The reduction in the discrepancy field in the forward section of the airfoil results in a reduction of turbulence. The profile also bears similarities with the intermittency field used in transition models[62]. A more detailed physical interpretation of the discrepancy field is difficult as it corrects for the combined effects of transition, adverse pressure gradients, and separation, which are difficult to isolate.

Fig. 6.11 shows the velocity contours of the inverse and the base solution. The separation bubble lengths are significantly different in the two cases. The separation point is near the mid-chord, which is consistent with the experimental observation. The exact location of separation is not presented in the experiment; only a qualitative description is provided. Based on the improvements in the surface pressure and separation point, it can be argued that the overall velocity field is also improved in the inversion.

Fig. 6.12 shows the eddy-viscosity contours of the inverse and base solutions. The eddy-viscosity is reduced in the forward section, which is consistent with the transition



(a) Base SA

(b) Inverse SA

Figure 6.12: Eddy-viscosity for the S809 airfoil at $Re = 2$ Million and $\alpha = 14°$.

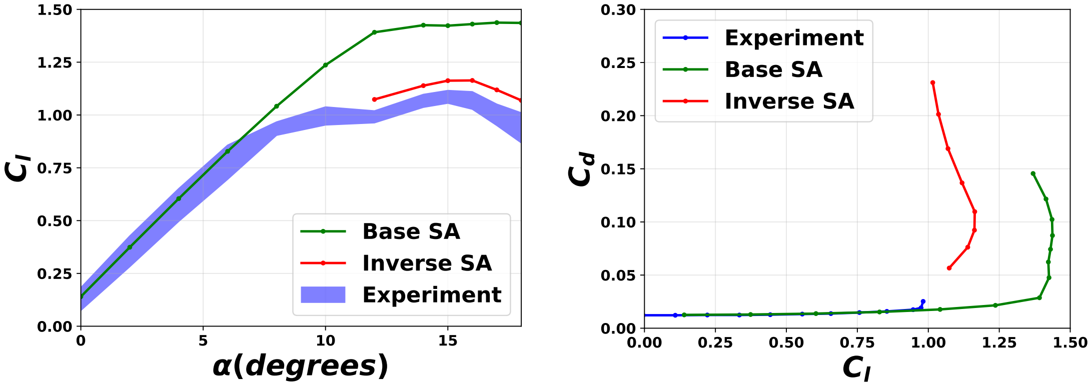
from laminar to turbulent flow. The increase in eddy-viscosity in the separation region points to enhanced mixing, which is not accurately captured by the baseline SA model.

Finally, Fig. 6.13 shows the lift and the drag comparison of the baseline SA and inverse SA with the experiments. Significant improvements are observed in the lift and the separation onset angle.

The results presented in this section show that it is possible to infer a turbulence model producing accurate results for one problem. However, it can be argued whether such a model actually replicates the relevant physics or is just producing numerical artifacts. Using the same spatial discrepancy field for a different shape or Reynolds number may not necessarily lead to an improved solution. The discrepancy fields are therefore reconstructed as a function of local flow quantities using ML and used for predictions.

6.2.2 Machine Learning Training

Machine-learning (ML) is used to reconstruct the discrepancy field as a function of locally non-dimensional flow features. The features are identical to the ones used in Chapter V. The neural-network (NN) algorithm is used in contrast with the previous



(a) C_l vs α

(b) C_d vs C_l

Figure 6.13: Lift and drag coefficients obtained using inferred discrepancy fields for the S809 airfoil at Re = 2 Million.

chapter.

All the inverse solutions or their combinations can be used as inputs to train the ML model. In the absence of a precise approach to a priori quantification of the quality of the resulting model, we train many ML models using a number of combinations and use them for prediction.

Table 6.2 shows the different combinations of inverse solutions used to train the ML models and also the labels assigned to them. Model **P** results in the most optimal predictions for a range of shapes, angles of attack, and Reynolds numbers. Model **P** contains the most extreme cases in terms of adverse pressure gradients and separation, therefore the flow-features in all remaining cases are a subset of the features used for model **P**.

All the ML models are trained using 5-fold cross-validation. Table 6.3 shows the coefficient of determination R^2 for training the model **P**. Fig. 6.14 shows the scatter of prediction and the true values for all the folds for the training and testing data. The quality of the training is not as good when compared to the previous examples. The ML training suffers from extreme values of the flow features, especially near the stagnation region. The real measure of quality, however, is when the ML model is

Model label	*Training data*
P	**S814** at $\mathbf{R}e = 1 \times 10^6, 2 \times 10^6$
1	S805 at $Re = 1 \times 10^6$
2	S805 at $Re = 2 \times 10^6$
3	S809 at $Re = 1 \times 10^6$
4	S809 at $Re = 2 \times 10^6$
5	S805 at $Re = 1 \times 10^6, 2 \times 10^6$
6	S809 at $Re = 1 \times 10^6, 2 \times 10^6$
7	S805, S809, S814 at $Re = 1 \times 10^6, 2 \times 10^6$

Table 6.2: List of ML models and the inverse solutions used for training. The main predictive model is labeled **P**.

CV-fold	*Training error*	*Validation error*
1	0.9403	0.8832
2	0.9272	0.8665
3	0.9293	0.8572
4	0.9404	0.8847
5	0.9401	0.9033

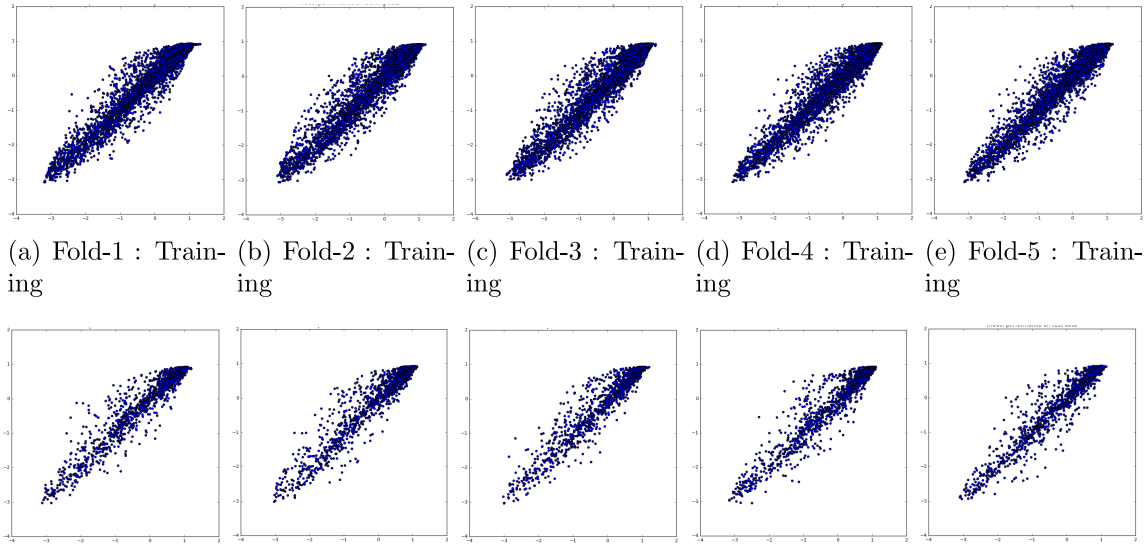
Table 6.3: Results of 5-fold cross-validation. The error metric is the coefficient of determination - R^2.

used in the iterative RANS solver for prediction.

6.2.3 Machine Learning Prediction

The trained ML models are used in an iterative RANS solver for prediction. Figs. 6.15 and 6.16 compare the inverse solution and the machine learning prediction using model **P** for the S809 airfoil at $\alpha = 14^\circ$ and $Re = 2 \times 10^6$. Except for minor differences, the predicted pressure using model **P** is in agreement with the inverse solution and the experimental data. The inverse and the model P predicted velocity field are also qualitatively similar. Overall the model **P** shows significant improvement over the baseline model in terms of lift, drag, and the location of flow-separation. The training data-set was based on assimilating lift information only.

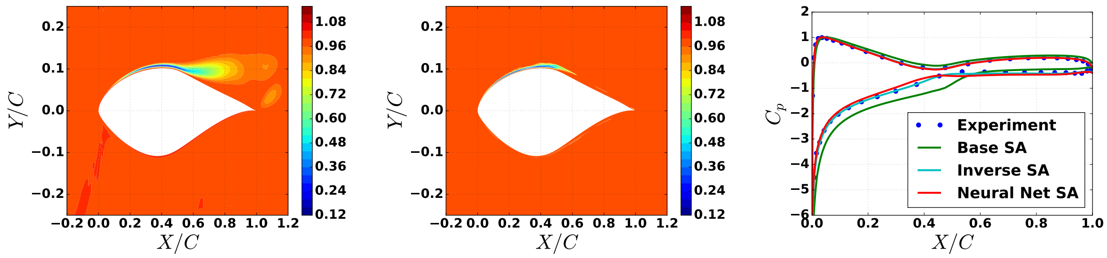
Fig. 6.17 shows the lift and drag prediction using model **P** for all Reynolds numbers



(a) Fold-1 : Training (b) Fold-2 : Training (c) Fold-3 : Training (d) Fold-4 : Training (e) Fold-5 : Training

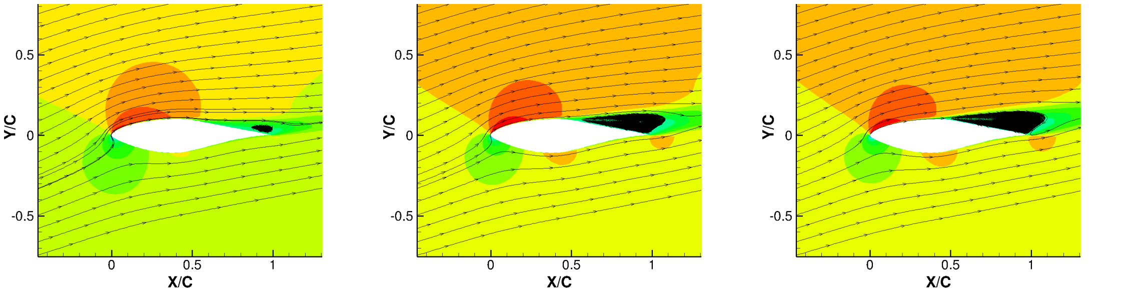
(f) Fold-1 : Testing (g) Fold-2 : Testing (h) Fold-3 : Testing (i) Fold-4 : Testing (j) Fold-5 : Testing

Figure 6.14: Neural network training for model **P**. x and y axes correspond to the true and predicted values, respectively.



(a) $\boldsymbol{\beta}(\mathbf{x})$ from inverse SA (b) $\boldsymbol{\beta}(\mathbf{U})$ from NN-augmented SA (prediction) (c) Pressure coefficient

Figure 6.15: Comparison of inverse and NN-augmented predictions (using model **P**) for S809 airfoil at $\alpha = 14°$ and $Re = 2 \times 10^6$.



(a) Base SA

(b) Inverse SA

(c) NN-augmented SA (prediction)

Figure 6.16: Streamlines and X-velocity contours for the S809 airfoil at $Re = 2 \times 10^6$ and $\alpha = 14°$.

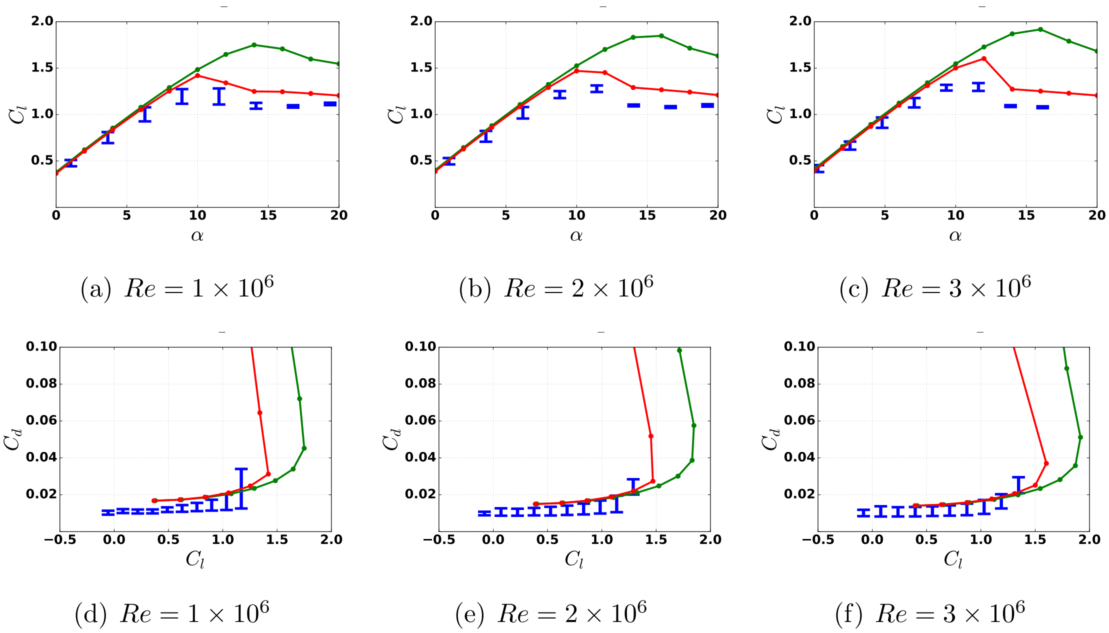
for the S814 airfoil, including $Re = 3 \times 10^6$, which was not used in the training set. Significant improvement in stall prediction is evident in the lift curve. As a consequence, the drag rise is predicted to occur at lower angles of attack than in the baseline model, a trend that is qualitatively correct. There is no evidence of deterioration of accuracy in the low angle of attack regions, where the original model is already accurate. The model **P**erforms equally well for airfoil shapes not used in the training set, i.e. S805 and S809 (Figs. 6.18, 6.19).

The improvement in the quality of the predictions is further emphasized in Figs. 6.20, 6.21, and 6.22, in which pressure predictions are shown for the S809, S805 and S814 airfoils. These results confirm that the NN-augmented model offers considerable predictive improvements in surface pressure distributions.

Fig. 6.23 shows the base SA, and the NN-augmented SA solutions for two different grid resolutions. The solutions using both models suggest that the variability in the solutions between the two grids is much smaller than the corrections introduced by the data augmentation.

6.2.4 Portability of the Trained Model

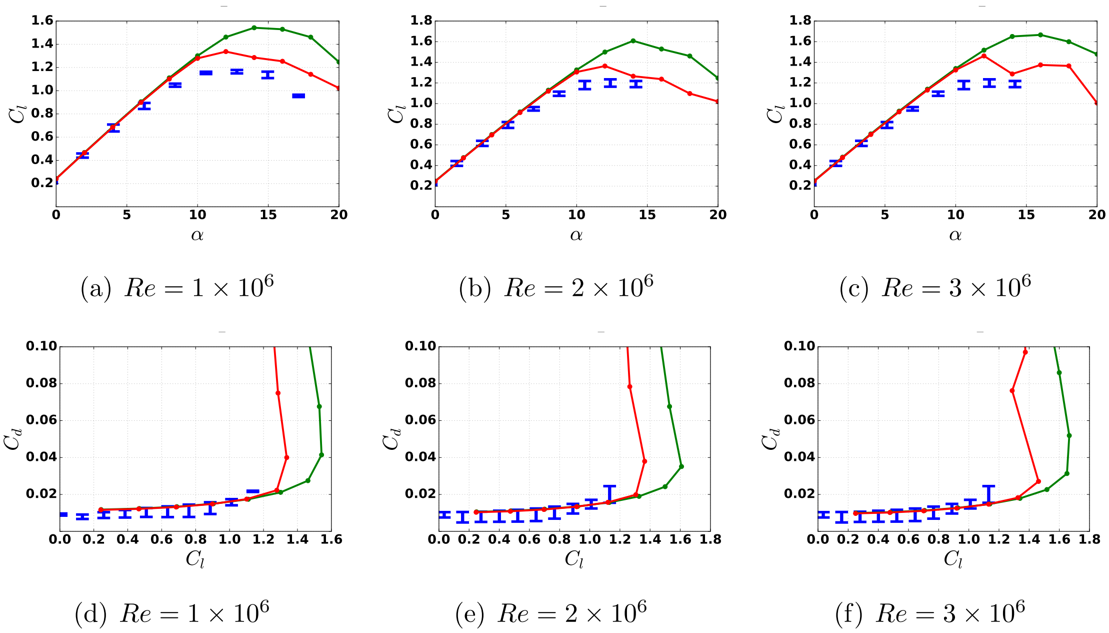
The entire modeling framework was developed and tested using an in-house unstructured finite-volume flow solver. To demonstrate the portability of this approach,



(a) $Re = 1 \times 10^6$ (b) $Re = 2 \times 10^6$ (c) $Re = 3 \times 10^6$

(d) $Re = 1 \times 10^6$ (e) $Re = 2 \times 10^6$ (f) $Re = 3 \times 10^6$

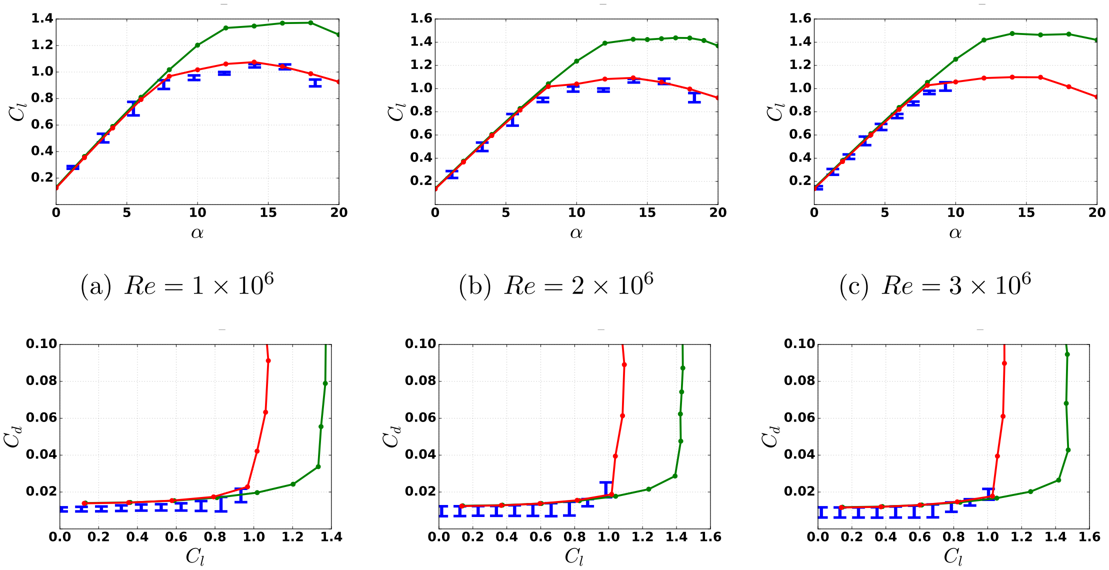
Figure 6.17: NN-augmented SA prediction for the S814 airfoil using model **P**. Legend: — Experiment, — base SA and — neural network SA model **P**.



(a) $Re = 1 \times 10^6$ (b) $Re = 2 \times 10^6$ (c) $Re = 3 \times 10^6$

(d) $Re = 1 \times 10^6$ (e) $Re = 2 \times 10^6$ (f) $Re = 3 \times 10^6$

Figure 6.18: NN-augmented SA prediction for the S805 airfoil using model **P**. Legend: — Experiment, — base SA and — neural network SA model **P**.



(a) $Re = 1 \times 10^6$

(b) $Re = 2 \times 10^6$

(c) $Re = 3 \times 10^6$

(d) $Re = 1 \times 10^6$

(e) $Re = 2 \times 10^6$

(f) $Re = 3 \times 10^6$

Figure 6.19: NN-augmented SA prediction for the S809 airfoil using model **P**. Legend: — Experiment, — base SA and — neural network SA model **P**.

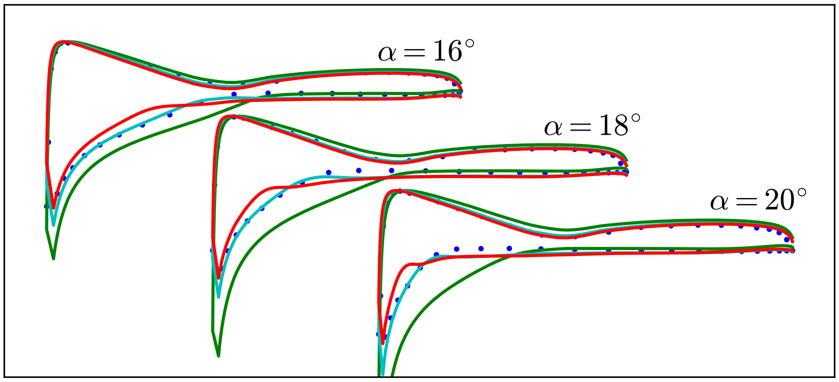


Figure 6.20: Surface pressure for the S809 airfoil at $Re = 2 \times 10^6$ and $\alpha = \{16°, 18°, 20°\}$. Refer Fig. 6.15(c) for legend. Not to scale.

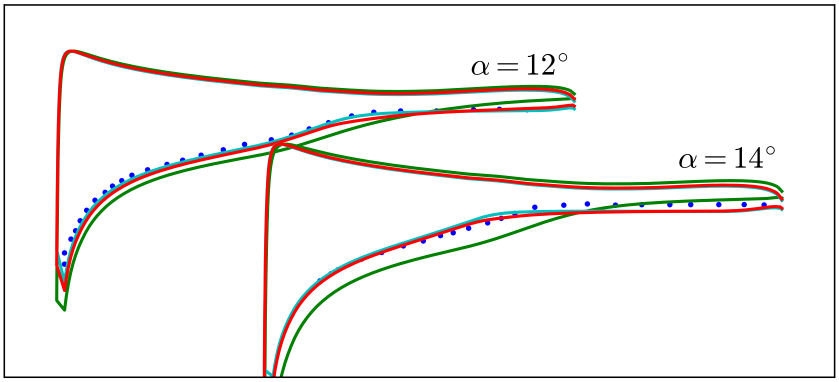


Figure 6.21: Surface pressure for the S805 airfoil at $Re = 1 \times 10^6$ and $\alpha = \{12°, 14°\}$. Refer Fig. 6.15(c) for legend. Experimental pressure is shown only for the upper surface. Not to scale.

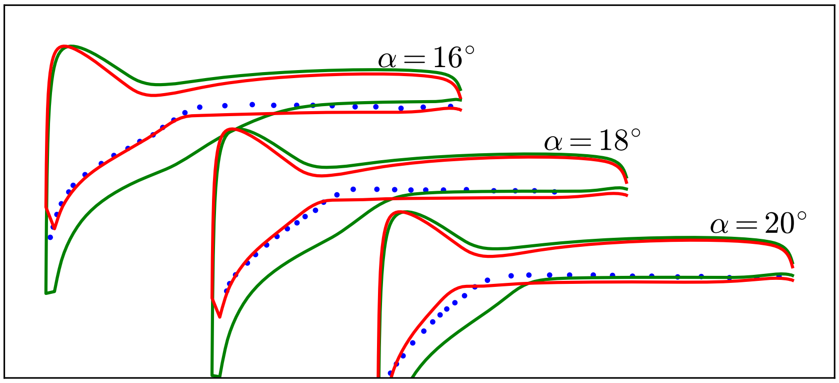
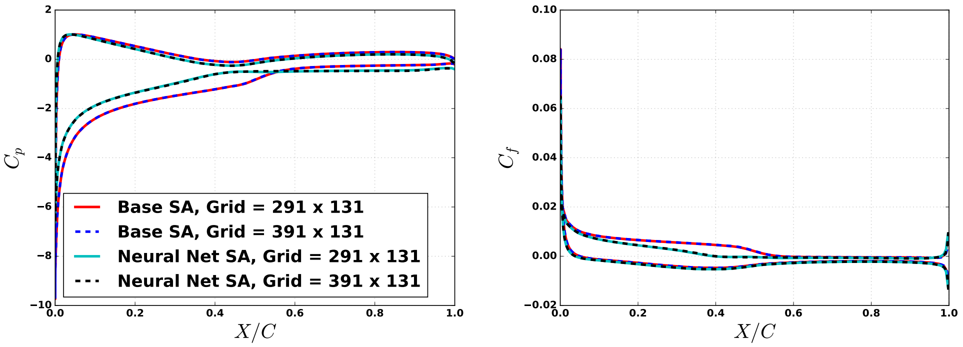


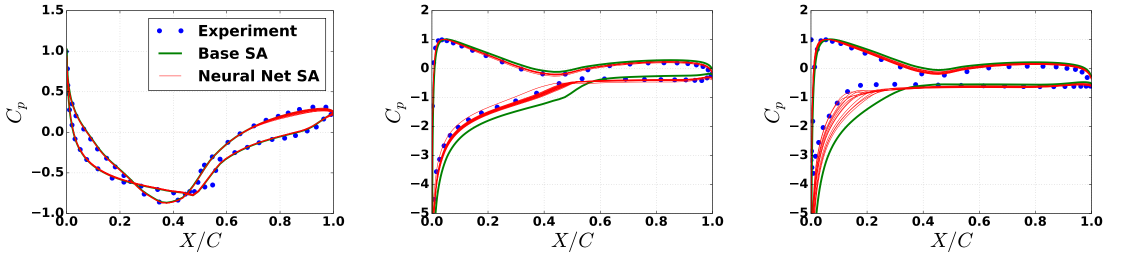
Figure 6.22: Surface pressure for the S814 airfoil at $Re = 1.5 \times 10^6$ and $\alpha = \{16°, 18°, 20°\}$. Refer Fig. 6.15(c) for legend. Experimental pressure is shown only for the upper surface. Inversion is not performed for this case. Not to scale.



(a) C_p

(b) C_f

Figure 6.23: Pressure and skin friction (using model **P**) for the S809 airfoil at $Re = 2 \times 10^6$ and $\alpha = 14°$ using grids of different spatial resolutions. Solutions of both the base SA model and the neural network augmented SA are grid converged.



(a) $\alpha = 0°$

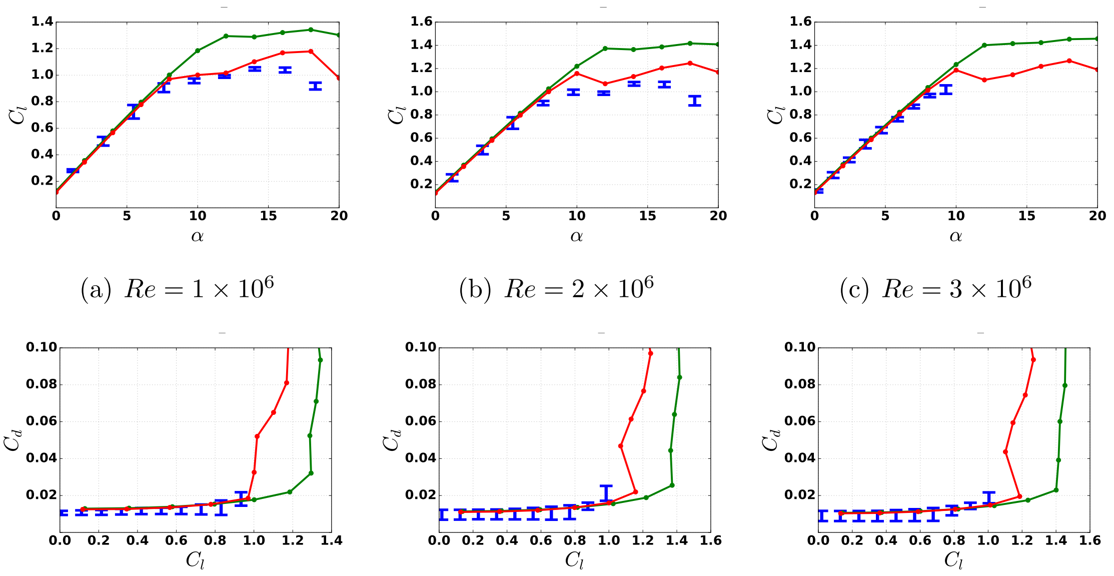
(b) $\alpha = 14°$

(c) $\alpha = 20°$

Figure 6.24: Predicted surface pressure for the S809 airfoil at $Re = 2 \times 10^6$ using 8 different NN-based models listed in table 6.2.

the NN-augmented SA model **P** is embedded into *AcuSolve*, a commercially available unstructured flow solver based on the Galerkin/Least-Squares (GLS) stabilized finite-element method [43, 87]. Therefore, developing the neural network model based on a feature set consisting of locally non-dimensional flow variables, as presented in this work, is essential for portability across flow solvers.

Fig. 6.25 shows the lift and drag coefficient predictions from *AcuSolve* for the S-809 airfoil at three Reynolds numbers. The NN-augmentation shows significant improvement in predictions and its effectiveness is comparable to that observed in the in-house code. It should be noted that the *AcuSolve* uses a variation of the SA model which corrects for the rotation and the curvature effects. These corrections are



(a) $Re = 1 \times 10^6$ (b) $Re = 2 \times 10^6$ (c) $Re = 3 \times 10^6$

(d) $Re = 1 \times 10^6$ (e) $Re = 2 \times 10^6$ (f) $Re = 3 \times 10^6$

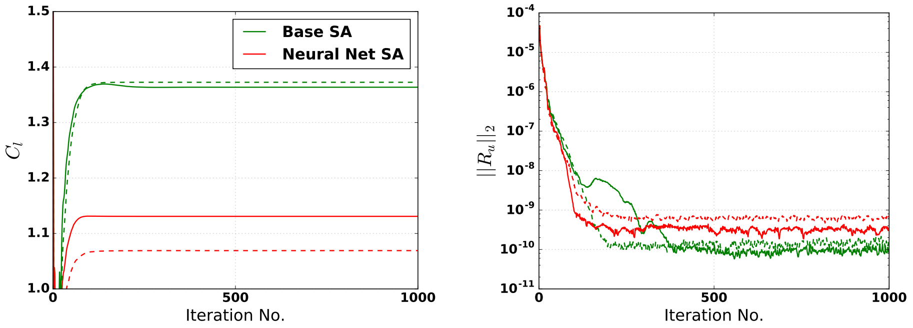
Figure 6.25: NN-augmented SA prediction using *AcuSolve* for the S809 airfoil using model **P**. Legend: — Experiment, — base SA and — neural network SA model **P**.

not used in the in-house solver and therefore the solutions from these two codes are not expected to be identical, even for the baseline model.

Fig. 6.26 shows the rate of convergence for the base SA and the NN-augmented SA for a sample problem. The initial condition is set to uniform free-stream for all runs. The NN-augmented model displays comparable convergence characteristics to the baseline model. Additional overhead exists in passing the features $\boldsymbol{\eta}$ to the NN and obtaining $\boldsymbol{\beta}$ at grid locations, this was confirmed to add less than 10% of additional compute time compared to the baseline calculation in *AcuSolve*.

6.2.5 Variation Between Models: A Measure of Uncertainty

It is desirable that any new modifications introduced into a turbulence model do not affect the solution at flow conditions for which the base model is accurate. The results suggest that the NN-augmented SA model satisfies this requirement. Fig. 6.24 shows this feature for the S809 airfoil at a Reynolds number of 2×10^6. The predicted surface pressure using neural networks trained on the different data-sets listed in



(a) Lift coefficient

(b) L2 norm of solver residual

Figure 6.26: *AcuSolve*'s convergence history for S809 airfoil at $Re = 2 \times 10^6$, $\alpha = 12^\circ$ (dashed) and $\alpha = 14^\circ$ (solid).

Table 6.2 is shown in red lines. Model augmentations show variability as is apparent in Figs. 6.24 (b) and (c). Overall, the neural network-augmented models are more accurate than the base SA model for all the cases, and more importantly, none of the NN-augmented predictions diverge from the base SA model at $\alpha = 0^\circ$. While this ensemble approach does not qualify as a formal uncertainty quantification technique, it is nevertheless a useful test to ascertain the sensitivity of the model output to the training set. If significant variabilities are revealed in the model predictions, it serves as a warning to the user that models may be operating far from the conditions in which they were trained.

Fig. 6.24 shows that the quality of the NN-augmented model is sensitive to the selection of the training-data. In this work, the best model **P** is selected by exploring several combinations of the data–sets. This observation is subjected to the uncertainty involved with the intermediate steps (feature selection, ML algorithm, etc.).

6.3 Summary

- The FIML framework was applied to augment the Spalart-Allmaras (SA) model for separated flows over airfoils. The application presented in this chapter used low fidelity experimental data in contrast to relatively high-quality data in the previous chapters. All the inverse problems used for model augmentation utilized the lift coefficient exclusively.

- The inverse solutions using lift and surface pressure were found to be nearly identical. The inverse solution using lift leads to improvements in the surface pressure and location of separation. This opens up the use of experimental lift data for augmentation of RANS models. Such data are more readily available for practical cases than high-fidelity DNS or LES data.

- The inferred discrepancy fields in all the cases bear similarities to the intermittency field used in transition models and are in line with the expectations based on the physics of the problem.

- Machine-learning (ML) was used to train an augmented model using the inverse solutions for the thickest airfoil, S814. The machine-learned model was used for the prediction of other airfoil shapes and Reynolds numbers with improved results.

- A number of ML models were trained using combinations of the inverse solutions and all the models were shown to perform significantly better than the baseline model. The ensembles of the prediction represent a way of measuring uncertainties associated with the FIML framework.

- The portability of the ML model was demonstrated by embedding the model designed using an in-house finite-volume compressible solver to a finite-element incompressible solver. Improvements in prediction similar to the ones observed in the in-house solver were produced with the finite-element code, demonstrating model portability.

CHAPTER VII

Conclusions and Future Work

7.1 Summary and Conclusions

Most popular RANS models in practice today were developed more than two decades ago. While elegant ideas such as elliptic relaxation have emerged since, their penetration has been limited because of a lack of generality in the predictive capability of a given model. As a consequence, many of the problems with RANS models which existed two decades ago continue to persist today. It remains an open question in the turbulence community whether it is possible to create a universal turbulence model, which produces acceptable results for all flow cases.

The thesis proposes a new paradigm to augment turbulence models with the goal of improving them in flow configurations of a similar class. The proposed framework of FIML uses data to infer discrepancy in existing models. These discrepancies from many problems were generalized using machine learning to propose an augmentation. The method is different from similar attempts in three significant ways:

1. Instead of directly learning from the data, the approach creates information from data via inverse problems and learns from the information. This way, consistency is established between the embedded model discrepancies and the model.

2. The method introduces a functional discrepancy rather than the discrepancy in a parameter.

3. The method can use any amount of data of any level of fidelity (as long as the data is informative for the model discrepancy), whereas most past works required high-fidelity datasets.

The use of inverse problems to infer discrepancies in existing models instead of directly calculating discrepancy from the data recognizes the fact that all the variables used in RANS based models are modeled quantities and not *real* quantities. For example, the turbulence kinetic energy k and dissipation ϵ used in RANS models are not the real k and ϵ. These variables serve to estimate the length and time scale associated with the flow, which in turn is used to estimate the eddy-viscosity. Therefore, these variables cannot be directly extracted from a DNS or LES. Further, solving an inverse problem does not require access to model related data (like k and ϵ), it is possible to solve an inverse problem with a wide range of data–from full-field velocity to integral lift. Inverse problems can also be used to propagate the uncertainty associated with the data to the model discrepancy.

The field-inversion and machine-learning framework was presented in Chapter II. Two different perspectives of framing an inverse problem were discussed. In the Bayesian setting, a probability distribution of the discrepancy is sought using data, while in a deterministic setting only the mean discrepancy is sought. The discrete adjoint formulation required to calculate the sensitivities was presented. The utility of the machine learning was examined in the context of the FIML framework. Neural-networks and Adaboost, the two machine learning algorithms used in this work were presented. Methods to prevent overfitting and to select flow features for the machine learning were discussed in the chapter.

The utility of the FIML framework was demonstrated by using it to augment the SA and the Wilcox's $k-\omega$ model for a set of flow configurations. As a proof of concept of the data-driven model augmentation, synthetically generated data was used to augment the SA model. This setup has virtually no uncertainty and tests the approach under a best-case scenario. The setup consisted of flow in a concavely curved channel. The synthetically generated data, which was based on an analytic correction to the SA model, was used to recover the form of discrepancy in the SA model. The impact of using just surface data or full-field data on the inverse solution was discussed, and it was shown that while more data is beneficial, surface data such as skin-friction can improve the velocity and Reynolds stress fields. The inverse solutions were reconstructed as functions of local flow solution using a machine-learning (ML) algorithm. The ML reconstructions were embedded in iterative RANS solver with improved predictions. ML models trained using various combinations of flow-features and inverse solutions were compared.

Chapter V utilizes data from LES of flows over bumps of varying heights to augment the Wilcox's $k-\omega$ model. Such flows include regions of flow stabilization and enhanced mixing, which is under-predicted by existing models. Full-field velocity and skin friction were used to infer the discrepancy in the $k-\omega$ model. It was shown that using only the skin-friction data, it is possible to improve the velocity fields and the Reynolds stress prediction. Ensembles of models were trained using the Adaboost algorithm and used for prediction for all the cases. Prediction using all the ML augmented models were found to be more accurate than the baseline $k-\omega$ model.

In chapter VI, experimental lift data, for a family of airfoils, was used to augment the SA model. The flow contains regions of transition, adverse pressure gradient, and separation. The discrepancy field inferred using data for lift and surface pressure were found to be similar. Using data for just lift coefficient the FIML framework can construct an augmented SA model which predicts not only lift but surface pressure and point of separation much more accurately than the baseline SA model. The portability of the augmented model was demonstrated by embedding the augmentation developed using an in-house finite volume solver and the FIML approach into a commercially available incompressible finite-element solver.

The augmentation produced by the FIML framework in all the cases was found to be consistent with the expectation based on the understanding of the flow physics.

While this thesis presents a new paradigm with optimistic results, the presented results also raise additional questions. Certainly, a better model augmentation which works for all classes of problem was not created. Whether a *universal* model exists is itself a disputed question in the RANS community. However, at the same time, the data-driven techniques offer a way to drive the existing models to a higher level of accuracy. Data-driven techniques can be particularly useful when the representative flow physics is well-captured in the data sets. This approach will be relevant in an industrial setting as the focus is typically on a class of flows and data is usually available for at least a few flow-conditions, as is domain expertise that can be used to solidify the model improvement process.

There is no consensus among the turbulence community, whether there is an ultimate barrier in turbulence modeling or that we have reached it. Results shown in the present work suggest that, if there is indeed a barrier, it either has not yet been reached or it can be moved conditional on how informative the data is to a class of flows. It has to be mentioned that the inaccurate results could be because of the functional form of the models and not necessarily due to the inadequacy of Reynolds-averaging or even the eddy viscosity assumption.

An online platform *Turbulence Modeling Gateway*[1] was created as a part of this work. All the ensembles of models used in Chapter V along with application programming interface (API) to use them are available for a user to download. The platform will eventually serve as one point place for users to upload data, run the inversion and create the ML augmented model for their application. This work marks the first

[1] http://turbgate.engin.umich.edu

step in that direction.

7.2 Suggestions for Future Work

While a universal augmentation seems a far-reaching goal, many insights can be learned in continuing the pursuit, which in turn, contribute to turbulence modeling. Some extensions of the present work and future directions are discussed below:

- **Feature selection**: In the present work, the features used for training and testing machine-learning (ML) models are selected based on the understanding of turbulence modeling. The choices served well for the cases presented in the thesis. Indeed, this set of features is by no means unique or optimal. Augmenting expert knowledge with automatic feature selection will go a long way in generalizing the approach, especially in problems where the physics is complicated. It would be desirable to have an automatic approach to select flow-features which are non-dimensional and satisfy Galilean invariance principles. Moreover, the number of features has to be minimized to limit the quantity of data required to train a ML model with a high degree of confidence.

- **Uncertainty quantification**: Uncertainty quantification (UQ) is not just a buzzword in the fluid dynamics community–the need for UQ cannot be emphasized more. Errors and uncertainties in a RANS simulation arise from many sources including flow conditions, numerical discretization and the turbulence model. In many complex flows, the uncertainty due to the turbulence model is typically the most dominant, and estimating a measure of such uncertainty will help a user weigh the prediction for decision making. In theory, uncertainties in the FIML process appear naturally. Uncertainty in the data can be propagated to the inverse solution through Bayesian inversion, which in turn can be propagated to the machine learning utilizing algorithms like Gaussian processes. In practice, it is difficult because of the computational expense associated with proper Bayesian inversion and propagation of the uncertainties. Efforts have been made by making assumptions regarding the form of the uncertainties, but more work is needed. One of the possible paths would be to construct the inferred solution in a low dimensional manifold using dimension reduction, which, may make the Bayesian inversion tractable.

- **Standards for data-driven augmentations**: The ultimate goal of this work is to highlight the potential of data-driven models in improving prediction of turbulence models and help in decision making or improve the design process. The results presented in the thesis should be considered to be a proof-of-concept of the ideas. Any practical use of the strategy would require standards and benchmarks to ensure the confidence in data-driven models. Standards regarding the format of input and output to the machine learning, a measure of confidence and standards for *easy* sharing of such models for reproducibility will be as important as the quality of the predictions. In the absence of such standards or consensus, we will be adding to the clutter which is already abundant in the turbulence modeling community.

- **Augmenting the model for multiple classes of problems**: In all the results pursued in the thesis, a turbulence model is augmented and used for prediction of flows which are similar to the one used to construct the augmentation. An extension towards a more universal augmentation would be to use different classes of problems. Any such attempts are constrained by the availability of informative data. Ideally, the Reynolds number of the cases used for inversion and the nature of discrepancy should be comparable for a good augmentation.

- **Application to more sophisticated models**: The FIML framework can be applied to more sophisticated turbulence models such as the $v^2 - f$ and non-linear eddy-viscosity models which can capture secondary flows and anisotropy in the Reynolds stresses in the near-wall region. These models contain beneficial ideas which are shadowed by calibration of few parameters. The FIML framework can be used to correct for the error due to calibrations and build on additional physics introduced by these models.

- **Application to other physical models**: While presented in the context of turbulence modeling, the FIML framework can be used to augment different physics (such as materials modeling). The framework requires no domain-specific modifications and can utilize a wide range of data-sets.

APPENDIX

APPENDIX A

Turbulence Models

Following sections contain details of the two turbulence models used in this work.

A.1 Spalart–Allmaras (SA) Model

The SA model[99] is a one equation model and it solves for the modified Eddy viscosity, $\tilde{\nu}$, which relates to kinematic Eddy viscosity ν_t as follows:

$$\nu_t = \tilde{\nu} f_{v1}; \quad f_{v1} = \frac{\chi^3}{\chi^3 + c_{v1}^3}; \quad \chi = \frac{\tilde{\nu}}{\nu}. \tag{A.1}$$

The governing equation of the SA model without the trip terms is given by,

$$\frac{\partial \rho \tilde{\nu}}{\partial t} + \frac{\partial \rho u_j \tilde{\nu}}{\partial x_j} = P - D + \frac{1}{\sigma} \left[\nabla.((\mu + \rho\tilde{\nu})\nabla\tilde{\nu}) + c_{b2}\rho(\nabla\tilde{\nu})^2 \right], \tag{A.2}$$

where, P and D are the production and destruction respectively and are given by,

$$P = c_{b1}\tilde{\Omega}\rho\tilde{\nu} \quad \text{and} \quad D = c_{w1} f_w \rho [\frac{\tilde{\nu}}{d}]^2. \tag{A.3}$$

$\tilde{\Omega}$ is a function of the vorticity magnitude, Ω, and is defined as,

$$\tilde{\Omega} = \Omega + \frac{\tilde{\nu}}{\kappa^2 d^2} f_{v2}, \quad f_{v2} = 1 - \frac{\chi}{1 + \chi f_{v1}}. \tag{A.4}$$

The function f_w is defined as,

$$f_w = g \left[\frac{1 + c_{w3}^6}{g^6 + c_{w3}^6} \right]^{\frac{1}{6}}, \quad g = r + c_{w2}(r^6 - r), \quad r = \frac{\tilde{\nu}}{\tilde{\Omega}\kappa^2 d^2}. \tag{A.5}$$

The model constants are: $c_{b1} = 0.1355, \sigma = 2/3, c_{b2} = 0.622, \kappa = 0.41, c_{w1} = c_{b1}/\kappa^2 + (1 + c_{b2})/\sigma, c_{w2} = 0.622, c_{w3} = 2.0, c_{v1} = 7.1$. The freestream boundary is set to fully turbulent with $\tilde{\nu}/\nu_\infty = 3$ and $\tilde{\nu}$ is set to zero at the no-slip walls.

A.2 Wilcox's $k - \omega$ Model

The two-equation Wilcox's $k - \omega$ model[113] solves the transport equations of the turbulence kinetic energy (k) and the dissipation (ω). The transport equations are given by,

$$\frac{\partial(\rho k)}{\partial t} + \frac{\partial(\rho u_j k)}{\partial x_j} = P - \beta^* \rho \omega k + \frac{\partial}{\partial x_j} \left[\left(\mu + \sigma_k \frac{\rho k}{\omega} \right) \frac{\partial k}{\partial x_j} \right], \tag{A.6}$$

and

$$\frac{\partial(\rho \omega)}{\partial t} + \frac{\partial(\rho u_j \omega)}{\partial x_j} = \frac{\gamma \omega}{k} P - \beta \rho \omega^2 + \frac{\partial}{\partial x_j} \left[\left(\mu + \sigma_\omega \frac{\rho k}{\omega} \right) \frac{\partial \omega}{\partial x_j} \right] + \frac{\rho \sigma_d}{\omega} \frac{\partial k}{\partial x_j} \frac{\partial \omega}{\partial x_j}. \tag{A.7}$$

The eddy-viscosity is then defined by,

$$\mu_t = \frac{\rho k}{\hat{\omega}}, \tag{A.8}$$

where,

$$\hat{\omega} = \max \left[\omega, C_{lim} \sqrt{\frac{2\overline{S}_{ij}\overline{S}_{ij}}{\beta^*}} \right]. \tag{A.9}$$

The production term is given by,

$$P = \tau_{ij} \frac{\partial u_i}{\partial x_j} \tag{A.10}$$

with,

$$\tau_{ij} = \mu_t \left(2S_{ij} - \frac{2}{3} \frac{\partial u_k}{\partial x_k} \delta_{ij} \right) - \frac{2}{3} \rho k \delta_{ij}, \tag{A.11}$$

$$S_{ij} = \frac{1}{2} \left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right), \tag{A.12}$$

and,

$$\overline{S}_{ij} = S_{ij} - \frac{1}{3} \frac{\partial u_k}{\partial x_k} \delta_{ij}. \tag{A.13}$$

The constants and the intermediate functions are given by,

$$\sigma_k = 0.6, \sigma_\omega = 0.5, \beta^* = 0.09, \gamma = \frac{13}{25}, C_{lim} = \frac{7}{8}, \beta = \beta_0 f_\beta, \beta_0 = 0.0708, \tag{A.14}$$

$$f_\beta = \frac{1 + 85\chi_\omega}{1 + 100\chi_\omega}, \tag{A.15}$$

$$\chi_\omega = \left| \frac{\Omega_{ij} \Omega_{ij} \hat{S}_{ki}}{(\beta^* \omega)^3} \right|, \tag{A.16}$$

$$\hat{S}_{ki} = S_{ki} - \frac{1}{2} \frac{\partial u_m}{\partial x_m} \delta_{ki}, \tag{A.17}$$

$$\Omega_{ij} = \frac{1}{2} \left(\frac{\partial u_i}{\partial x_j} - \frac{\partial u_j}{\partial x_i} \right), \tag{A.18}$$

and

$$\sigma_d = 0, \quad for \quad \frac{\partial k}{\partial x_j} \frac{\partial \omega}{\partial x_j} \leq 0. \tag{A.19}$$

At the viscous wall $k = 0$ and ω takes an asymptotic value of $\frac{6\nu_{wall}}{\beta_0 d^2}$. The freestream

k and ω are evaluated by specifying the freestream turbulence intensity and the ratio of the turbulent to laminar viscosity.

BIBLIOGRAPHY

BIBLIOGRAPHY

[1] Arolla, S., and P. Durbin (2015), LES of spatially developing turbulent boundary layer over a concave surface, *Journal of Turbulence*, *16*(1), 81–99.

[2] Aster, R., B. Borchers, and C. Thurber (2013), *Parameter Estimation and Inverse Problems*, 2 ed., Elsevier Inc.

[3] Balay, S., et al. (2016), PETSc users manual revision 3.7, *Tech. rep.*, Argonne National Lab.(ANL), Argonne, IL (United States).

[4] Baldwin, B., and T. Barth (1990), *A one-equation turbulence transport model for high Reynolds number wall-bounded flows*, National Aeronautics and Space Administration, Ames Research Center.

[5] Baldwin, B., and H. Lomax (1978), Thin layer approximation and algebraic model for separated turbulent flows, *257*.

[6] Bischof, C., A. Carle, G. Corliss, A. Griewank, and P. Hovland (1992), ADIFOR–generating derivative codes from fortran programs, *Scientific Programming*, *1*(1), 11–29.

[7] Bishop, C. (1995), Training with noise is equivalent to Tikhonov regularization, *Neural computation*, *7*(1), 108–116.

[8] Bradshaw, P. (1969), The analogy between streamline curvature and buoyancy in turbulent shear flow, *Journal of Fluid Mechanics*, *36*(1), 177–191.

[9] Bradshaw, P., D. Ferriss, and N. Atwell (1967), Calculation of boundary-layer development using the turbulent energy equation, *Journal of Fluid Mechanics*, *28*(3), 593–616.

[10] Breiman, L. (2001), Random forests, *Machine learning*, *45*(1), 5–32.

[11] Bui-Thanh, T., and M. Girolami (2014), Solving large-scale pde-constrained bayesian inverse problems with riemann manifold hamiltonian monte carlo, *Inverse Problems*, *30*(11), 114,014.

[12] Campos, A., K. Duraisamy, and G. Iaccarino (2016), A segregated explicit algebraic structure-based model for wall-bounded turbulent flows, *International Journal of Heat and Fluid Flow*, *61*, 284–297.

[13] Chang, Y., S. Yang, and O. Arici (1996), Flow field computation of the NREL S809 airfoil using various turbulence models, *No. CONF-960154–. American Society of Mechanical Engineers, New York, NY (United States), 1996.*

[14] Cheung, S., and J. Beck (2009), New Bayesian updating methodology for model validation and robust predictions based on data from hierarchical subsystem tests, *Earthquake Engineering Research Laboratory, Caltech*, (CaltechEERL:EERL-2008-04).

[15] Cheung, S., T. Oliver, E. E. Prudencio, S. Prudhomme, and R. Moser (2011), Bayesian uncertainty analysis with applications to turbulence modeling, *Reliability Engineering & System Safety*, *96*(9), 1137–1149.

[16] Choi, H., and P. Moin (2012), Grid-point requirements for large eddy simulation: Chapmans estimates revisited, *Physics of fluids*, *24*(1), 011,702.

[17] Constantine, P., C. Kent, and T. Bui-Thanh (2016), Accelerating markov chain monte carlo with active subspaces, *SIAM Journal on Scientific Computing*, *38*(5), A2779–A2805.

[18] Cui, T., J. Martin, Y. Marzouk, A. Solonen, and A. Spantini (2014), Likelihood-informed dimension reduction for nonlinear inverse problems, *Inverse Problems*, *30*(11), 114,015.

[19] Dow, E., and Q. Wang (2011), Uncertainty quantification of structural uncertainties in RANS simulations of complex flows, *20th AIAA Computational Fluid Dynamics Conference Honolulu, Hawaii (AIAA 2011-3865)*, doi: 10.2514/6.2011-3865.

[20] Dow, E., and Q. Wang (2011), Quantification of structural uncertainties in the $k-\omega$ turbulence model, *52nd AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference, Denver, Colorado (AIAA 2011-1762)*, doi:10.2514/6.2011-1762.

[21] Drucker, H. (1997), Improving regressors using boosting techniques, *ICML*, *97*, 107–115.

[22] Duraisamy, K., and J. Alonso (2012), Adjoint based techniques for uncertainty quantification in turbulent flows with combustion, *42nd AIAA Fluid Dynamics Conference and Exhibit*, p. 2711.

[23] Duraisamy, K., and G. Iaccarino (2005), Curvature correction and application of the v^2-f turbulence model to tip vortex flows, *Center for Turbulence Research Annual Research Briefs*, pp. 157–168.

[24] Duraisamy, K., and A. Singh (2016), Informing turbulence closures with computational and experimental data, *54th AIAA Aerospace Sciences Meeting*, p. 1556.

[25] Duraisamy, K., J. Alonso, F. Palacios, and P. Chandrashekar (2010), Error estimation for high speed flows using continuous and discrete adjoints, *48th AIAA Aerospace Sciences Meeting*, pp. 2010–128.

[26] Duraisamy, K., Z. Zhang, and A. Singh (2015), New approaches in turbulence and transition modeling using data-driven techniques, *53rd AIAA Aerospace Sciences Meeting, 1284*.

[27] Duraisamy, K., A. Singh, and S. Pan (2017), Augmentation of turbulence models using field inversion and machine learning, *55th AIAA Aerospace Sciences Meeting*, p. 0993.

[28] Durbin, P. (1991), Near-wall turbulence closure modeling without "damping functions", *Theoretical and Computational Fluid Dynamics*, *3*(1), 1–13.

[29] Edeling, W., P. Cinnella, R. Dwight, and H. Bijl (2014), Bayesian estimates of parameter variability in the $k - \epsilon$ turbulence model, *Journal of Computational Physics*, *258*, 73–94.

[30] Eggels, J., F. Unger, M. Weiss, J. Westerweel, R. Adrian, R. Friedrich, and F. Nieuwstadt (1994), Fully developed turbulent pipe flow: a comparison between direct numerical simulation and experiment, *Journal of Fluid Mechanics*, *268*, 175–210.

[31] Eisfeld, B., and O. Brodersen (2005), Advanced turbulence modelling and stress analysis for the DLR-F6 configuration, *AIAA Paper*, *4727*(2005), 436.

[32] EMC2 (2012), IDC digital universe study: Big data, bigger digital shadows and biggest growth in the far east, *http://www.whizpr.be/upload/medialab/21/company/Media_Presentation_2012_DigiUniverseFINAL1.pdf accessed on Oct 3, 2017.*

[33] Emmons, H. (1954), Shear flow turbulence, *Journal of Applied Mechanics-Transactions of the ASME*, *21*(3), 283–283.

[34] Emory, M., R. Pecnik, and G. Iaccarino (2011), Modeling structural uncertainties in Reynolds-Averaged computations of shock/boundary layer interactions, *49th AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition Orlando, Florida (AIAA 2011-479)*, doi: 10.2514/6.2011-479.

[35] Emory, M., J. Larsson, and G. Iaccarino (2013), Modeling of structural uncertainties in Reynolds-averaged Navier-Stokes closures, *Physics of Fluids*, *25*(11), 110,822.

[36] Freund, Y., and R. Schapire (1995), A desicion-theoretic generalization of online learning and an application to boosting, *European conference on computational learning theory*, pp. 23–37.

[37] Giles, M., and N. Pierce (2000), An introduction to the adjoint approach to design, *Flow, turbulence and combustion*, *65*(3-4), 393–415.

[38] Glushko, G. (1966), Turbulent boundary layer on a flat plate in an incompressible fluid, *Tech. rep.*, Technical Information And Library Services Ministry Of Aviation London (England).

[39] Gorle, C., M. Emory, and G. Iaccarino (2012), RANS modeling of turbulent mixing for a jet in supersonic cross flow: model evaluation and uncertainty quantification, *Proceedings of the Seventh International Symposium on Turbulence, Heat and Mass Transfer (ICHMT Digital Library Online, Palermo, Italy, 2012)*.

[40] Griewank, A., D. Juedes, and J. Utke (1996), Algorithm 755: ADOL-C: a package for the automatic differentiation of algorithms written in C/C++, *ACM Transactions on Mathematical Software (TOMS)*, *22*(2), 131–167.

[41] Haario, H., M. Laine, A. Mira, and E. Saksman (2006), Dram: efficient adaptive mcmc, *Statistics and Computing*, *16*(4), 339–354.

[42] Hascoet, L., and V. Pascual (2013), The tapenade automatic differentiation tool: principles, model, and specification, *ACM Transactions on Mathematical Software (TOMS)*, *39*(3), 20.

[43] Hughes, T., L. France, and G. Hulbert (1989), A new finite element formulation for computational fluid dynamics. VIII: The Galerkin/least-squares method for advective-diffusive equations, *Computer Methods in Applied Mechanics and Engineering*, *73*(2), 173–189.

[44] Ioffe, S., and C. Szegedy (2015), Batch normalization: Accelerating deep network training by reducing internal covariate shift, *International Conference on Machine Learning*, pp. 448–456.

[45] Kato, H., and S. Obayashi (2014), Data assimilation for turbulent flows, *16th AIAA Non-Deterministic Approaches Conference, AIAA SciTech, (AIAA 2014-1177)*, doi:10.2514/6.2014-1177.

[46] Kennedy, M., and A. O'Hagan (2001), Bayesian calibration of computer models, *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, *63*(3), 425–464.

[47] Kim, S.-E., B. Makarov, and D. Caraeni (2003), A multi-dimensional linear reconstruction scheme for arbitrary unstructured grids, *AIAA paper*, *3990*, 2003.

[48] Kohavi, R. (1995), A study of cross-validation and bootstrap for accuracy estimation and model selection, *Proceedings of the 14th International Joint Conference on Artificial Intelligence - Volume 2*, pp. 1137–1143.

[49] Kollmann, W. (1990), The pdf approach to turbulent flow, *Theoretical and Computational Fluid Dynamics*, *1*(5), 249–285.

[50] Kolmogorov, A. (1941), The local structure of turbulence in incompressible viscous fluid for very large Reynolds numbers, *Dokl. Akad. Nauk SSSR*, *30*(4), 299–303.

[51] Langtry, R. (2006), A correlation-based transition model using local variables for unstructured parallelized CFD codes.

[52] Le, H., P. Moin, and J. Kim (1997), Direct numerical simulation of turbulent flow over a backward-facing step, *Journal of fluid mechanics*, *330*, 349–374.

[53] Lee, M., and R. Moser (2015), Direct numerical simulation of turbulent channel flow up to $Re_\tau = 5200$, *Journal of Fluid Mechanics*, *774*, 395–415.

[54] Ling, J., and J. Templeton (2015), Evaluation of machine learning algorithms for prediction of regions of high Reynolds averaged Navier Stokes uncertainty, *Physics of Fluids*, *27*(8), 085,103.

[55] Ling, J., R. Jones, and J. Templeton (2016), Machine learning strategies for systems with invariance properties, *Journal of Computational Physics*, *318*, 22 – 35, doi:https://doi.org/10.1016/j.jcp.2016.05.003.

[56] Ling, J., A. Kurzawski, and J. Templeton (2016), Reynolds averaged turbulence modelling using deep neural networks with embedded invariance, *Journal of Fluid Mechanics*, *807*, 155–166.

[57] Mani, M., D. Babcock, C. Winkler, and P. Spalart (2013), Predictions of a supersonic turbulent flow in a square duct, *AIAA Paper*, *860*, 2013.

[58] Martin, J., L. Wilcox, C. Burstedde, and O. Ghattas (2012), A stochastic newton mcmc method for large-scale statistical inverse problems with application to seismic inversion, *SIAM Journal on Scientific Computing*, *34*(3), A1460–A1487.

[59] Martin, R., and B. Heinrich (1993), A direct adaptive method for faster backpropagation learning: The RPROP algorithm, *IEEE Iinternational conference on Neural Networks*, pp. 586–591.

[60] Martins, J., P. Sturdza, and J. Alonso (2003), The complex-step derivative approximation, *ACM Transactions on Mathematical Software (TOMS)*, *29*(3), 245–262.

[61] Martins, J., P. Sturdza, and J. Alonso (2003), The complex-step derivative approximation, *ACM Transactions on Mathematical Software (TOMS)*, *29*(3), 245–262.

[62] Medida, S. (2014), Correlation-based transition modeling for external aerodynamic flows, Ph.D. thesis, University of Maryland, College Park.

[63] Menter, F. (1992), Improved two-equation $k - \omega$ turbulence models for aerodynamic flows.

[64] Menter, F., R. Langtry, and S. Völker (2006), Transition modelling for general purpose CFD codes, *Flow, turbulence and combustion*, *77*(1), 277–303.

[65] Milano, M., and P. Koumoutsakos (2002), Neural network modeling for near wall turbulent flow, *Journal of Computational Physics*, *182*(1), 1–26.

[66] Moser, R., J. Kim, and N. Mansour (1999), Direct numerical simulation of turbulent channel flow up to $Re_\tau = 590$, *Physics of fluids*, *11*(4), 943–945.

[67] Nissen, S. (2003), Implementation of a fast artificial neural network library (FANN), `http://leenissen.dk/fann/`, *Department of Computer Science, University of Copenhagen*.

[68] Oliver, T., and R. Moser (2011), Bayesian uncertainty quantification applied to RANS turbulence models, *Journal of Physics: Conference Series*, *318*(4), 042,032.

[69] Parish, E., and K. Duraisamy (2016), A paradigm for data-driven predictive modeling using field inversion and machine learning, *Journal of Computational Physics*, *305*, 758–774.

[70] Parneix, S., D. Laurence, and P. Durbin (1998), A procedure for using DNS databases, *Journal of fluids engineering*, *120*(1), 40–47.

[71] Patel, V., and F. Sotiropoulos (1997), Longitudinal curvature effects in turbulent boundary layers, *Progress in Aerospace Sciences*, *33*(1-2), 1–70.

[72] Pecnik, R., V. Terrapon, F. Ham, G. Iaccarino, and H. Pitsch (2012), Reynolds-averaged Navier-Stokes simulations of the HyShot II scramjet, *AIAA journal*, *50*(8), 1717–1732.

[73] Pedregosa, F., et al. (2011), Scikit-learn: Machine learning in Python, *Journal of Machine Learning Research*, *12*, 2825–2830.

[74] Pope, S. (2001), *Turbulent flows*, IOP Publishing.

[75] Poroseva, S., and S. Murman (2014), Velocity/pressure-gradient correlations in a FORANS approach to turbulence modeling, *44th AIAA Fluid Dynamics Conference, AIAA Aviation, (AIAA 2014-2207)*, doi:10.2514/6.2014-2207.

[76] Poroseva, S., B. Kaiser, J. Sillero, and S. Murman (2015), Validation of a closing procedure for fourth-order rans turbulence models with dns data in an incompressible zero-pressure-gradient turbulent boundary layer, *International Journal of Heat and Fluid Flow*, *56*, 71–79.

[77] Prandtl, L. (1925), Bericht über untersuchungen zur ausgebildeten turbulenz, *Zeitschrift fur angew. Math. u. Mechanik*, *5*(2), 136–139.

[78] Prandtl, L., and K. Wieghardt (1947), *Über ein neues Formelsystem für die ausgebildete Turbulenz*, Vandenhoeck & Ruprecht.

[79] Raiesi, H., U. Piomelli, and A. Pollard (2011), Evaluation of turbulence models using direct numerical and large-eddy simulation data, *Journal of Fluids Engineering*, *133*(2), 021,203.

[80] Rall, L. (1981), Automatic differentiation: Techniques and applications.

[81] Rasmussen, C., and C. K. Williams (2006), *Gaussian processes for machine learning*, vol. 1, MIT press Cambridge.

[82] Ray, J., S. Lefantzi, S. Arunajatesan, and L. DeChant (2015), Bayesian calibration of a RANS model with a complex response surface - a case study with Jet-in-Crossflow configuration, *45th AIAA Fluid Dynamics Conference Dallas, TX (AIAA 2015-2784)*, doi:10.2514/6.2015-2784.

[83] Richardson, C. (1954), *An introduction to the calculus of finite differences*, Van Nostrand.

[84] Saad, Y. (2003), *Iterative methods for sparse linear systems*, SIAM.

[85] Saad, Y., and M. Schultz (1986), GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems, *SIAM Journal on scientific and statistical computing*, *7*(3), 856–869.

[86] Saric, W. (1994), Görtler vortices, *Annual Review of Fluid Mechanics*, *26*(1), 379–409.

[87] Shakib, F., T. Hughes, and Z. Johan (1991), A new finite element formulation for computational fluid dynamics. X: The compressible Euler and Navier-Stokes equations, *Computer Methods in Applied Mechanics and Engineering*, *89*(1–3), 141–219.

[88] Shur, M., M. Strelets, A. Travin, and P. Spalart (2000), Turbulence modeling in rotating and curved channels: assessing the spalart-shur correction, *AIAA journal*, *38*(5), 784–792.

[89] Singh, A., and K. Duraisamy (2016), Using field inversion to quantify functional errors in turbulence closures, *Physics of Fluids (1994-present)*, *28*(4), 045,110.

[90] Singh, A., R. Matai, K. Duraisamy, and P. Durbin (2017), Data-driven augmentation of turbulence models for adverse pressure gradient flows, *23rd AIAA Computational Fluid Dynamics Conference*, p. 3626.

[91] Singh, A., S. Medida, and K. Duraisamy (2017), Machine-learning-augmented predictive modeling of turbulent separated flows over airfoils, *AIAA Journal*.

[92] Singh, A., S. Pan, and K. Duraisamy (2017), Characterizing and improving predictive accuracy in shock-turbulent boundary layer interactions using data-driven models, *55th AIAA Aerospace Sciences Meeting*, p. 0314.

[93] Smirnov, P., and F. Menter (2009), Sensitization of the SST turbulence model to rotation and curvature by applying the Spalart–Shur correction term, *Journal of Turbomachinery*, *131*(4), 041,010.

[94] Smith, A., and T. Cebeci (1967), Numerical solution of the turbulent-boundary-layer equations, *Tech. rep.*, Douglas Aircraft Co., Long Beach CA, Aircraft Division.

[95] Somers, D. (1997), Design and experimental results for the S805 airfoil, *NREL Report, NREL/SR-440-6917*.

[96] Somers, D. (1997), Design and experimental results for the S809 airfoil, *NREL Report, NREL/SR-440-6918*.

[97] Somers, D. (1997), Design and experimental results for the S814 airfoil, *NREL Report, NREL/SR-440-6919*.

[98] Spalart, P. (2000), Strategies for turbulence modelling and simulations, *International Journal of Heat and Fluid Flow*, *21*(3), 252–263.

[99] Spalart, P., and S. Allmaras (1992), A one-equation turbulence model for aerodynamic flows, *30th Aerospace Sciences Meeting and Exhibit Reno, NV*, doi: 10.2514/6.1992-439.

[100] Spalart, P., and M. Shur (1997), On the sensitization of turbulence models to rotation and curvature, *Aerospace Science and Technology*, *1*(5), 297–302.

[101] Spalart, P., W. Jou, M. Strelets, and S. Allmaras (1997), Comments on the feasibility of LES for wings, and on a hybrid RANS/LES approach, *Advances in DNS/LES*, *1*, 4–8.

[102] Spalart, P. R. (2015), Philosophies and fallacies in turbulence modeling, *Progress in Aerospace Sciences*, *74*(Supplement C), 1 – 15, doi:https://doi.org/10.1016/j.paerosci.2014.12.004.

[103] Speziale, C., R. Abid, and E. Anderson (1992), Critical evaluation of two-equation models for near-wall turbulence, *AIAA journal*, *30*(2), 324–331.

[104] Toro, E., M. Spruce, and W. Speares (1994), Restoration of the contact surface in the hll-riemann solver, *Shock waves*, *4*(1), 25–34.

[105] Tracey, B., K. Duraisamy, and J. Alonso (2013), Application of supervised learning to quantify uncertainties in turbulence and combustion modeling, *51st AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition Grapevine (Dallas/Ft. Worth Region), Texas (AIAA 2013-259)*, doi:10.2514/6.2013-259.

[106] Tracey, B. D. (2015), Machine learning for model uncertainties in turbulence models and monte carlo integral approximation, Ph.D. thesis, Stanford University.

[107] Travin, A., M. Shur, M. Strelets, and P. Spalart (2002), Physical and numerical upgrades in the detached-eddy simulation of complex turbulent flows, pp. 239–254, Springer.

[108] Venkatakrishnan, V. (1995), Convergence to steady state solutions of the euler equations on unstructured grids with limiters, *Journal of computational physics*, *118*(1), 120–130.

[109] Vrugt, J., C. Ter Braak, C. Diks, B. Robinson, J. Hyman, and D. Higdon (2009), Accelerating markov chain monte carlo simulation by differential evolution with self-adaptive randomized subspace sampling, *International Journal of Nonlinear Sciences and Numerical Simulation*, *10*(3), 273–290.

[110] Wang, J.-X., J. Wu, and H. Xiao (2017), Physics-informed machine learning approach for reconstructing Reynolds stress modeling discrepancies based on DNS data, *Physical Review Fluids*, *2*(3), 034,603.

[111] Wang, Q., K. Duraisamy, J. Alonso, and G. Iaccarino (2012), Risk assessment of scramjet unstart using adjoint-based sampling methods, *AIAA journal*, *50*(3), 581–592.

[112] Weatheritt, J. (2015), The development of data driven approaches to further turbulence closures, Ph.D. thesis, University of Southampton.

[113] Wilcox, D. (1998), *Turbulence modeling for CFD*, vol. 2, DCW industries La Canada, CA.

[114] Wilcox, D. (2008), Formulation of the $k - \omega$ turbulence model revisited, *AIAA Journal*, *46*(11), 2823–2838.

[115] Wilcox, D., and T. Chambers (1977), Streamline curvature effects on turbulent boundary layers, *AIAA J*, *15*(4), 574–580.

[116] Witteveen, J., K. Duraisamy, and G. Iaccarino (2011), Uncertainty quantification and error estimation in scramjet simulation, *17th AIAA International Space Planes and Hypersonic Systems and Technologies Conference*, pp. 11–14.

[117] Wolfe, W., and S. Ochs (1997), CFD calculations of S809 aerodynamic characteristics, *Aerospace Sciences Meetings*, doi:10.2514/6.1997-973.

[118] Wright, S., and J. Nocedal (1999), Numerical optimization, *Springer Science*, *35*(67-68), 7.

[119] Wu, J.-L., J.-X. Wang, H. Xiao, and J. Ling (2017), A priori assessment of prediction confidence for data-driven turbulence modeling, *Flow, Turbulence and Combustion*, pp. 1–22.

[120] Xiao, H., J. Wu, J. Wang, R. Sun, and C. Roy (2016), Quantifying and reducing model-form uncertainties in Reynolds-averaged Navier-Stokes simulations: A data-driven, physics-informed Bayesian approach, *Journal of Computational Physics*, *324*, 115–136.

[121] Yang, S., Y. Chang, and O. Arici (1994), Incompressible Navier-Stokes computation of the NREL airfoils using a symmetric total variational diminishing scheme, *Journal of solar energy engineering*, *116*(4), 174–182.

[122] Yarlanki, S., B. Rajendran, and H. Hamann (2012), Estimation of turbulence closure coefficients for data centers using machine learning algorithms, *13th InterSociety Conference on Thermal and Thermomechanical Phenomena in Electronic Systems*, pp. 38–42, doi:10.1109/ITHERM.2012.6231411.

[123] Zilitinkevich, S., V. M. Gryanik, V. Lykossov, and D. Mironov (1999), Third-order transport and nonlocal turbulence closures for convective boundary layers, *Journal of the atmospheric sciences*, *56*(19), 3463–3477.