# SURROGATE MODELING OF FLOWS IN HEMODYNAMICS AND AERODYNAMICS USING MACHINE LEARNING

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF ENGINEERING AND SCIENCE
OF BILKENT UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR
THE DEGREE OF
MASTER OF SCIENCE
IN
MECHANICAL ENGINEERING

By
Kerem Dülger
August 2025

Surrogate Modeling of Flows in Hemodynamics and Aerodynamics Using Machine Learning
By Kerem Dülger
August 2025

We certify that we have read this thesis and that in our opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Barbaros Çetin(Advisor)

Gökberk Kabacaoğlu(Co-Advisor)

Orçun Koray Çelebi

Ali Karakuş

Approved for the Graduate School of Engineering and Science:

Orhan Arıkan
Director of the Graduate School

# ABSTRACT

## SURROGATE MODELING OF FLOWS IN HEMODYNAMICS AND AERODYNAMICS USING MACHINE LEARNING

Kerem Dülger
M.S. in Mechanical Engineering
Advisor: Barbaros Çetin
August 2025

This thesis investigates the integration of traditional physics-based simulations and modern machine learning (ML) techniques for modeling fluid dynamics in both biomedical and engineering domains. The primary motivation stems from the computational limitations of high-fidelity simulations, particularly when exploring wide parametric spaces such as stenosis geometries in arteries.

The study begins with a detailed Computational Fluid Dynamics (CFD) analysis of blood flow through idealized stenosed vessels. Stenosis geometry was parametrized using length ($L$), height ($N$), and shape exponent ($n$), and simulations were performed using both ellipsoid and superellipsoid profiles. The results demonstrated that stenosis height ($N$) has the most significant impact on hemodynamic parameters such as wall shear stress and pressure drop, with changes exceeding 300% in some configurations. However, the high computational cost associated with these simulations highlighted the need for surrogate models capable of rapid prediction.

To bridge this gap, an intermediate ML study was conducted using NACA 4- and 5-digit airfoil data. This stage served as a controlled environment to understand the ML modeling pipeline, from dataset design and input encoding to hyperparameter tuning and generalization analysis. A compact, rule-based input encoding was developed from NACA designations and flow conditions, allowing neural networks to predict aerodynamic coefficients efficiently. The insights gained here, in areas such as dataset sensitivity, activation function choice, and model complexity, provided a solid foundation for applying ML to biomedical flows.

Building on this groundwork, the final phase of the thesis applied ML models to predict hemodynamic quantities in stenosed vessels directly from geometric parameters. Various algorithms, including neural networks, support vector regression, and ensemble models, were trained on CFD-generated data. Neural networks showed the highest accuracy and strongest sensitivity to dataset size, making them particularly promising for biomedical prediction tasks. The resulting surrogate models significantly reduced computational requirements while maintaining high predictive fidelity, enabling scalable evaluations suitable for real-time clinical decision-making or large parametric studies.

Overall, this thesis demonstrates a three-phase strategy: using CFD for physical insight, NACA-based ML for model development, and stenosis-based ML for biomedical application. The result is a flexible and robust modeling framework that leverages the strengths of both simulation and data-driven methods to accelerate research in fluid dynamics.

*Keywords:* Computational Fluid Dynamics, Machine Learning, Geometry-Driven Prediction, Parametric Flow Analysis, Surrogate Modeling.

# ÖZET

# MAKİNE ÖĞRENMESİ TABANLI HEMODİNAMİK VE AERODİNAMİK AKIŞLARIN TEMSİLCİ MODELLEMESİ

Kerem Dülger
Makine Mühendisliği, Yüksek Lisans
Tez Danışmanı: Barbaros Çetin
Ağustos 2025

Bu tez, hem biyomedikal hem de mühendislik alanlarında akışkanlar dinamiğini modellemek için geleneksel fizik tabanlı simülasyonlar ile modern makine öğrenimi (ML) tekniklerinin entegrasyonunu araştırmaktadır. Araştırmanın temel motivasyonu, özellikle damar darlığı (stenoz) geometrileri gibi geniş parametre alanlarının incelendiği durumlarda, yüksek doğruluklu simülasyonların getirdiği hesaplama maliyetlerinin sınırlayıcı olmasıdır.

Çalışmanın ilk aşamasında, idealize edilmiş stenozlu damarlardaki kan akışı, Hesaplamalı Akışkanlar Dinamiği (CFD) kullanılarak detaylı biçimde analiz edilmiştir. Stenoz geometrisi; uzunluk ($L$), yükseklik ($N$) ve şekil üssü ($n$) olmak üzere üç parametreyle modellenmiş; hem elipsoid hem de süperelipsoid formlarda simülasyonlar gerçekleştirilmiştir. Sonuçlar, $N$ yüksekliğinin hemodinamik parametreler üzerinde en büyük etkiye sahip olduğunu ve bazı konfigürasyonlarda duvar kayma gerilmesinde (WSS) ve basınç farklarında %300'ün üzerinde artışa neden olduğunu göstermiştir. Ancak bu yüksek doğruluklu simülasyonlar önemli ölçüde zaman ve işlem gücü gerektirdiğinden, hızlı tahminler için yedek (surrogate) modellere ihtiyaç duyulmuştur.

Bu ihtiyaca yanıt olarak, ikinci aşamada NACA 4- ve 5-haneli hava profil verileriyle bir makine öğrenimi uygulaması geliştirilmiştir. Bu aşama, veri seti tasarımı, giriş kodlaması, hiperparametre seçimi ve genelleme analizi gibi makine öğrenimi süreçlerini öğrenmek için kontrollü bir ortam sağlamıştır. NACA profil kodlarının ve uçuş koşullarının doğrudan giriş olarak kullanıldığı kompakt bir kodlama yöntemi geliştirilmiş ve aerodinamik katsayılar (kaldırma ve moment) başarılı şekilde tahmin edilmiştir. Bu deneyim, biyomedikal akışlara geçişte kullanılacak modelleme stratejileri için sağlam bir temel oluşturmuştur.

Son aşamada, CFD'den elde edilen damar simülasyonu verileriyle eğitilen makine öğrenimi modelleri, stenoz geometrisine dayalı olarak hemodinamik çıktıları (örneğin WSS, basınç farkı) hızlı ve doğru şekilde tahmin etmiştir. Denenen modeller arasında sinir ağları (neural networks), destek vektör regresyonu (SVR) ve topluluk yöntemleri (ensemble models) yer almıştır. Sinir ağları, özellikle veri seti boyutuna karşı daha yüksek duyarlılık göstererek, biyomedikal uygulamalar için güçlü bir aday olduğunu kanıtlamıştır. Geliştirilen bu yedek modeller, hesaplama maliyetini önemli ölçüde azaltırken doğruluktan ödün vermemekte ve gerçek zamanlı klinik değerlendirmeler veya büyük ölçekli parametrik analizler için kullanılabilirlik sunmaktadır.

Genel olarak bu tez, fizik temelli analizden elde edilen içgörülerle başlayıp, hava profil verileriyle makine öğrenimi yöntemlerinin test edildiği bir geçiş aşamasıyla devam eden ve sonunda biyomedikal uygulamaya uyarlanan üç aşamalı bir strateji sunmaktadır. Ortaya konulan esnek ve güçlü modelleme çerçevesi, hem mühendislik hem de sağlık alanındaki akışkanlar dinamiği problemlerine hesaplama açısından verimli çözümler sağlamaktadır.

*Anahtar sözcükler*: Hesaplamalı Akışkanlar Dinamiği, Makine Öğrenmesi, Geometri Tabanlı Tahmin, Parametrik Akış Analizi, Yedek (Temsilci) Modelleme.

# Acknowledgement

I want to start my acknowledgments by thanking my advisor and co-advisor, Prof. Barbaros Çetin and Dr. Gökberk Kabacaoğlu, for supporting my project. I especially thank Dr. Gökberk Kabacaoğlu for guiding me from the beginning and expanding my viewpoint to the science.

I express my gratitude to Associate Prof. Ali Karakuş and Assistant Prof. Orçun Koray Çelebi for kindly accepting to be on my thesis committee and for providing me with their valuable comments on my thesis.

I also thank all of our departmental and non-departmental professors for the invaluable lessons they taught.

I gratefully acknowledge ROKETSAN and Çağatay Şahin for providing the airfoil dataset used in this study, which was crucial for my project.

Lastly, I wish to extend my thanks and deep appreciation to my family. Their unwavering support and belief in me are truly invaluable. No other source of happiness compares to the love and encouragement I receive from them. I will do my best to make them proud even more.

# Contents

**1 Introduction** **1**

1.1 Thesis Contributions and Structure . . . . . 3

**2 Traditional Computational Fluid Dynamics Case - Stenosis Analysis on a Blood Vessel** **4**

2.1 Motivation . . . . . 4

2.2 Methodology . . . . . 5

2.2.1 Governing Equations . . . . . 5

2.2.1.1 Blood Rheology . . . . . 7

2.2.2 Geometry Selection . . . . . 8

2.2.2.1 Length of Flow for Fully Developed Flow . . . . 9

2.2.2.2 Final Geometries . . . . . 11

2.2.3 Numerical Setup . . . . . 11

2.2.3.1 Boundary Conditions . . . . . 12

2.2.3.2 Solver Settings . . . . . 12

2.2.3.3 Mesh Setup . . . . . 13

2.2.3.4 Verification . . . . . 15

2.3 Results . . . . . 16

2.3.1 Comparison With Healthy Arteries . . . . . 17

2.3.2 Ellipsoid Results . . . . . 17

2.3.2.1 Effects of Length L . . . . . 18

2.3.2.2 Effects of Length N . . . . . 18

2.3.3 Superellipsoid Results . . . . . 20

2.3.3.1 Effects of order "n" . . . . . 21

2.3.3.2 Effects of Length N . . . . . 21

2.4 Conclusion . . . . . 24

**3 Introduction to Machine Learning - Machine Learning Assisted Optimal Airfoil Design at High Mach Numbers 26**

3.1 Motivation . . . . . 26

3.2 Methodology . . . . . 27

3.2.1 Workflow . . . . . 27

3.2.2 Data Set . . . . . 27

3.2.3 Input Format . . . . . 30

3.2.4 Hyperparameter Tuning . . . . . 35
3.2.4.1 Sigmoid . . . . . 37
3.2.4.2 Hyperbolic Tangent . . . . . 38
3.2.4.3 ReLU . . . . . 38
3.2.4.4 Leaky ReLU . . . . . 39
3.2.5 Support Vector Regression . . . . . 40
3.3 Results . . . . . 43
3.4 Conclusion . . . . . 44

**4 Machine Learning Assisted Blood Vessel Analysis 47**

4.1 Motivation . . . . . 47
4.2 Methodology . . . . . 49
4.2.1 Vessel Geometry . . . . . 49
4.2.2 Numerical Analysis . . . . . 50
4.2.3 Neural Network . . . . . 51
4.2.3.1 Input Format . . . . . 51
4.2.3.2 Hyperparameter Optimization . . . . . 52
4.2.4 Other Machine Learning Models . . . . . 53
4.2.4.1 Ridge . . . . . 54
4.2.4.2 Lasso . . . . . 55

4.2.4.3 K-Nearest Neighbor . . . . . 55

4.2.4.4 Decision Tree . . . . . 56

4.2.4.5 Random Forest . . . . . 57

4.2.4.6 Extreme Gradient Boosting . . . . . 57

4.3 Results . . . . . 58

4.4 Conclusion . . . . . 62

**5 Conclusion and Future Remarks 64**

5.1 Conclusion . . . . . 64

5.2 Future Remarks . . . . . 65

**A Design Points for Vessel Analysis 71**

**B Numerical Results for CFD Analysis 73**

**C Machine Learning Method Examples 76**

C.1 Support Vector Regression . . . . . 76

C.2 Ridge . . . . . 78

C.3 Lasso . . . . . 79

C.4 K-Nearest Neighbor . . . . . 80

C.5 Decision Tree . . . . . 81

C.6 Random Forest . . . . . 84

C.7 Extreme Gradient Boosting . . . . . 86

# List of Figures


2.1 (A) Velocity change on aorta within two cardiac cycle (B) Pressure change on aorta within two cardiac cycle . . . . . 6

2.2 Ellipsoid Parameter Representations . . . . . 10

2.3 (A) Ellipsoid and (B) Super-ellipsoid stenosis geometries . . . . . 11

2.4 Length of each section . . . . . 13

2.5 Line Profile Comparison of Wall Shear Stress . . . . . 16

2.6 Shear Stress Change in Various Lengths for n = 2 and Velocity = 0.52 cm/s . . . . . 19

2.7 Shear Stress Change in Various Narrowings for n = 2 and Velocity = 0.52 cm/s . . . . . 20

2.8 Wall Shear Stress in 3D model . . . . . 21

2.9 Wall Shear Stress in Various Power Index for Length = 16 mm and Velocity = 0.52 cm/s . . . . . 22

2.10 Wall Shear Stress in Various Narrowing for Length = 16 mm and Velocity = 0.52 cm/s . . . . . 23

3.1 Flow Diagram of the Neural Network Creation . . . . . 28

3.2 Distribution of (A) Drag, (B) Lift and (C) Moment Coefficients . 29

4.1 Stenosis Geometry . . . . . 50

# Chapter 1

# Introduction

Cardiovascular diseases, particularly those resulting from arterial stenosis, remain among the leading causes of mortality worldwide [1]. Stenosis, the narrowing of arteries due to plaque accumulation, disrupts normal blood flow and contributes to pathological outcomes such as stroke, thrombosis, atherosclerosis progression, and vessel rupture [2, 3]. These effects are closely related to local hemodynamic parameters, especially wall shear stress (WSS), pressure drop, and velocity field distribution, which have been identified as key biomarkers of vascular dysfunction [4, 5].

To quantify these relationships, this thesis begins with a Computational Fluid Dynamics (CFD) study of blood flow through stenosed vessels. Two classes of geometries, ellipsoid and superellipsoid, are used to model the narrowing, parameterized by $L$, the axial length of the stenosis; $N$, the severity or maximum height of the constriction; and $n$, the shape exponent, which controls the sharpness or smoothness of the superellipsoid profile.

Previous studies have shown that these geometric characteristics significantly influence flow separation, vortex formation, and wall shear localization [6, 7, 8]. However, running high-fidelity CFD simulations for a wide range of geometric configurations and flow conditions is computationally expensive; thus, they pose

a barrier to large-scale parametric analysis or real-time clinical applications[9].

To address this limitation, this thesis explores machine learning (ML) as a surrogate modeling approach. Rather than applying ML techniques directly to the complex vascular dataset, an intermediate study was first conducted using aerodynamic data as a simpler, well-characterized problem to develop and refine ML methodologies.

This preliminary machine learning study uses the NACA 4-digit and 5-digit airfoil series, where airfoil shapes are defined by rule-based identifiers encoding camber, thickness, and chordwise position[10]. Instead of using geometric coordinates or CFD meshes, the airfoil parameters themselves are encoded as structured inputs to various ML models. The goal is to predict aerodynamic coefficients, lift ($C_l$) and moment ($C_m$), under varying flight conditions such as angle of attack and Mach number. This controlled setting enables experimentation with hyperparameter tuning, input scaling, activation functions, and network depth, providing intuition for generalization, overfitting, and error metrics.

Insights gained from the NACA-based ML study were applied to the stenosis problem. Using datasets generated from the initial CFD simulations, machine learning models were trained to predict WSS and velocity fields based on stenosis geometry and inlet velocity. This surrogate modeling framework enables rapid predictions without the computational costs of solving the Navier–Stokes equations, making it suitable for use in optimization, uncertainty quantification, and potentially clinical diagnostics. By combining the physical fidelity of CFD with the speed of data-driven ML models, this approach bridges the gap between mechanistic modeling and practical application, offering a promising solution for vascular flow prediction.

## 1.1 Thesis Contributions and Structure

This thesis provides multiple contributions to the application of machine learning in the fluid dynamics domain. The study starts with an exploration of the effects of different stenosis shape parameters on cardiovascular hemodynamics. It then builds a predictive model for aerodynamic coefficient prediction of various NACA airfoil shapes under different flight conditions. Finally, it combines the two previous concepts to develop a fast predictive model for estimating blood flow properties in various stenosis cases. The structure of the thesis is as follows:

- **Chapter 2** provides details about blood flow in stenosed vessels and, using traditional Computational Fluid Dynamics analysis, investigates the effects of various stenosis shape parameters on hemodynamics.
- **Chapter 3** builds an optimized neural network model using a novel input method for predicting aerodynamic coefficients of NACA airfoils, and compares the results with a well-known machine learning model.
- **Chapter 4** combines the concepts from the previous chapters to create a stenosis database, which is used to train various machine learning models. The performance of each model is compared across different dataset sizes.
- **Chapter 5** concludes the thesis with a summary of findings and suggestions for future research directions.

# Chapter 2

# Traditional Computational Fluid Dynamics Case - Stenosis Analysis on a Blood Vessel

## 2.1 Motivation

Cardiovascular diseases, particularly those arising from arterial stenosis, remain a leading cause of mortality worldwide. Stenosis, which is the narrowing of blood vessels due to plaque accumulation, can significantly alter blood flow characteristics, leading to increases in wall shear stress (WSS) and unsteady velocities. These hemodynamic variables are not only indicators of disease presence but also play a direct role in the progression and rupture risk of atherosclerotic plaques. Accurate quantification of these variables is therefore essential for diagnosis, risk stratification, and treatment planning.

Clinical imaging techniques such as angiography or ultrasound provide valuable structural information but fall short in resolving detailed flow dynamics. In contrast, Computational Fluid Dynamics (CFD) offers a powerful, non-invasive approach to simulate blood flow through patient-specific geometries, enabling

high-resolution analysis of critical flow features such as velocity profiles, vorticity, and shear stresses. Through CFD, it becomes possible to isolate and evaluate the effects of individual geometric parameters such as stenosis length, height, and shape on flow behavior, which cannot be easily achieved through in vivo or in vitro methods, especially given the variability among patients and the challenges of replicating precise geometries in physical models.

Given its ability to provide high-resolution, quantitative insights into complex flow environments, CFD was selected as the foundational tool for this investigation into the effects of stenosis geometry. By systematically varying stenosis geometry within idealized artery models, this study leverages CFD to build a foundational understanding of how morphological features impact key hemodynamic markers. The results derived from these simulations offer a reliable basis for understanding the fundamental fluid mechanics of stenosed arteries and support further analysis of how morphological variations influence vascular health.

## 2.2 Methodology

### 2.2.1 Governing Equations

The fluid dynamics of blood flow in arteries can be modeled using the Navier–Stokes equations, which represent the conservation of mass and momentum. Generally, blood is assumed to be incompressible and laminar. However, under certain conditions, turbulent flow may be observed within the vascular system [11]. If the Reynolds number, defined in Equation 2.1, where $\rho$ is the fluid density, $\mu$ is the dynamic viscosity, $\mathbf{u}$ is the velocity field, and $\mathbf{D}$ is the hydraulic diameter, exceeds 4000, the flow becomes turbulent. However, even in large arteries, the Reynolds number typically ranges between 100 and 200. Due to the cardiac cycle, the blood ejected from the heart does not have a constant flow rate. Instead, the flow velocity follows the pressure waveform of the left ventricle over time in arterial circulation. In Figure 2.1, the relationship between ventricular

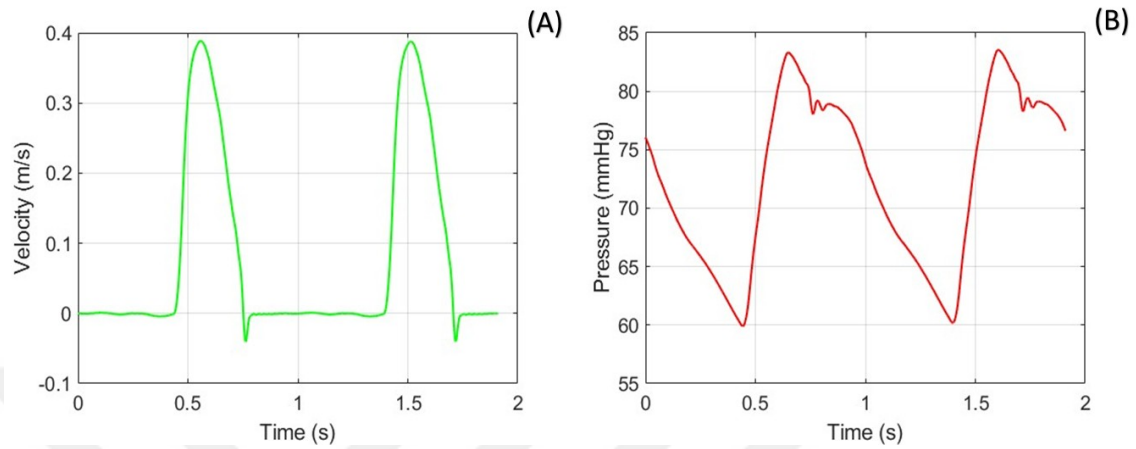


Figure 2.1: (A) Velocity change on aorta within two cardiac cycle (B) Pressure change on aorta within two cardiac cycle

pressure and blood velocity can be observed. In large vessels such as arteries and arterioles, significant pressure fluctuations can lead to rapid deceleration of blood flow. Combined with complex vessel geometries, these effects may give rise to transient turbulent-like disturbances. However, such flow behavior is not considered fully developed turbulence, as the spatial and temporal scales are small, the resulting Reynolds number remains low, and the flow is not self-sustaining. These disturbances dissipate quickly, returning the flow to a laminar state. Since this study focuses on small veins, the flow is assumed to remain laminar throughout. Based on this assumption, and the assumption of incompressibility, the Navier–Stokes equations can be expressed as follows, where $\mathbf{t}$ represents time and $\mathbf{g}$ denotes the gravitational acceleration vector.

$$Re = \frac{\rho V D}{\mu} \tag{2.1}$$

$$\begin{aligned} \nabla \cdot u &= 0 \\ \rho(\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u}) &= -\nabla \mathbf{p} + \mu \nabla^2 \mathbf{u} + \rho \mathbf{g} \end{aligned} \tag{2.2}$$

#### 2.2.1.1 Blood Rheology

Although blood is a non-Newtonian fluid, previous research has incorporated both Newtonian and non-Newtonian models to simulate blood flow. Reference [12] summarizes nearly all standard viscosity models used for predicting hemolysis. These models are categorized into four groups: (1) equation-based models with constant coefficients, (2) complex models that account for the thixotropic properties of blood, (3) population-based models, and (4) other standard models. Due to their simplicity, equation-based models were considered in this study for selecting an optimal viscosity model. The models examined include Newtonian, Carreau, Carreau–Yasuda, Casson, Power-Law, Generalized Power-Law, K–L, and Cross models [12]. Pabi *et al.* [13] also discusses prior work on the hemodynamic behavior of blood, noting that most studies utilize Newtonian, Carreau, Carreau–Yasuda, and Power-Law models. In light of these findings, and to ensure compatibility with ANSYS Fluent, We selected the Newtonian, Carreau, and Power-Law models for further analysis. The optimal model among these three was chosen by quantitatively comparing their simulation results with data reported in [14]. This decision-making process is explained in detail in the following sections.

For Newtonian fluids, the dynamic viscosity is considered constant. According to Nader *et al.* [15], the viscosity of healthy human blood ranges between 3.5 and $5.5cP$ ($10^{-3}$Pa·s. Therefore, we used the average of this range, $4.5\times10^{-3}$Pa·s, as a representative single viscosity value for all simulations. The Power-Law model is defined as:

$$\mu(\gamma) = k\gamma^{n-1} \tag{2.3}$$

where $\gamma$ is the shear rate, $k = 0.017$ is the power-law constant[16], and $n = 0.708$ is the power-law index [16] .

The Carreau model is expressed as:

$$\mu(\gamma) = \mu_\infty + (\mu_0 - \mu_\infty)[1 + (\lambda\gamma)^2]^{\frac{n-1}{2}} \tag{2.4}$$

where $\mu_\infty = 0.0035\ Pa{\cdot}s$ is the infinite shear viscosity[12], $\mu_0 = 0.056\ Pa{\cdot}s$ is the

zero-shear viscosity [16], $\lambda = 3.313005$ is the time constant[12], and $n = 0.3568$ is the power-law index[12].

We performed simulations using different viscosity models and compared their results with the reference data provided by Jedrzejczak *et al.* [14], which is based on well-established, experimentally derived empirical models [17, 18, 19, 20]. Each simulation's maximum wall shear stress (WSS) was compared against the WSS distribution reported by Jedrzejczak *et al.* [14]. Although Jedrzejczak *et al.* [14] does not provide exact numerical values for maximum WSS, it includes WSS distribution plots, which allowed for a qualitative comparison of peak WSS across geometries. The results indicated that for $N = 1$, the highest WSS values were approximately 55 Pa and 316 Pa at inlet velocities of 26 cm/s and 52 cm/s, respectively. For $N = 2$, the corresponding maximum WSS values were 8 Pa and 56 Pa. Among the viscosity models tested, the Carreau model exhibited the closest agreement with the trends presented by Jedrzejczak *et al.* [14]. According to Khan *et al.* [16], previous studies have shown that Carreau and Carreau–Yasuda models are more appropriate for modeling blood flow in stenosed arteries. In contrast, the Power-Law model tends to overestimate non-Newtonian effects, while models such as the Generalized Power-Law and modified Casson tend to underestimate them. Additionally, Khan *et al.* [16] notes that recent research increasingly favors the Carreau model for blood flow simulations. Based on these findings and the qualitative comparison with Although Jedrzejczak *et al.* [14], the Carreau model was selected for the parametric studies in this work.

### 2.2.2 Geometry Selection

The artery geometry is modeled as a straight cylindrical tube with an elliptical indentation centered along its length. The diameter is held constant at $4mm$, representing the average arterial diameter [14], and the axial distance from the center of the stenosis to each end of the artery is $2cm$. The cylinder is symmetric about its centerline. The height, length, and shape order of the stenosis are varied across simulations. To mathematically define the stenosis shape, both

ellipse and superellipsoid equations are employed. Through preliminary numerical experiments, we determined an entrance length sufficient for the flow to become fully developed before reaching the stenosis. The stenosis profile is defined using the generalized ellipse Equation 2.5:

$$\left|\frac{x}{a}\right|^n + \left|\frac{y}{b}\right|^n = 1 \tag{2.5}$$

Here, $n$ is the shape exponent, $a$ is the half-length of the stenosis, and $b$ is the half-height. As $n$ increases beyond 2, the profile transitions from an ellipse to a superellipsoid, resulting in a more rectangular shape. This transformation causes the stenosed region to occupy a larger spatial volume within the artery. In this study, we examine both elliptical ($n = 2$) and superelliptical shapes ($n = 4$, 5, and 6), along with variations in $a$ and $b$. The impact of each geometric parameter is illustrated in Figure 2.2. To facilitate interpretation of results, we redefined the geometric parameters in terms of physically meaningful quantities as follows:

$$\begin{aligned} L &= 2a \\ N &= D - (b - 2) \end{aligned} \tag{2.6}$$

where $L$ is the total length of the stenosis, $N$ represents the remaining (unclogged) vessel diameter, and $D$ is the full artery diameter. The shape exponent $n$ is retained as a parameter since it effectively captures the sharpness or bluntness of the stenosis profile.

#### 2.2.2.1 Length of Flow for Fully Developed Flow

To ensure accurate analysis of the effect of stenosis on blood flow, it is essential to eliminate any flow development effects near the stenosis. Therefore, the length of the artery is chosen such that the flow becomes fully developed before encountering the stenosed region. The required entrance length is calculated using Equation 2.7:

$$\frac{L_e}{D} = 0.06Re \tag{2.7}$$

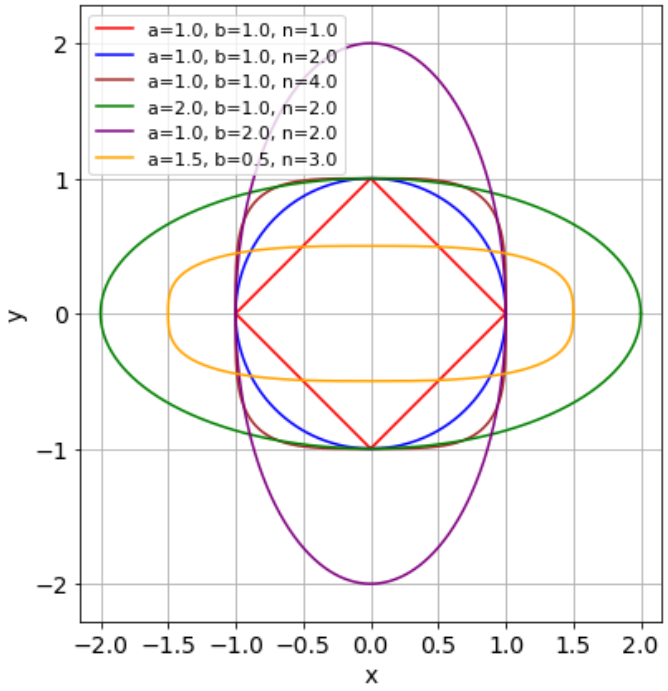


Figure 2.2: Ellipsoid Parameter Representations

where $L_e$ is the entrance length required for fully developed flow, $D = 4$mm is the diameter of the artery, and $Re$ is the Reynolds number. In this formulation, the artery diameter $D$ is used as the characteristic length in the Reynolds number calculation.

The Newtonian model was selected with an inlet velocity of 0.52 m/s. The maximum velocity was chosen because ensuring a sufficient entrance length for the highest velocity also guarantees appropriate development for lower velocities. We also verified that fully developed flow conditions were satisfied for the Power-Law and Carreau models by examining the velocity profiles in preliminary simulations. The assumption of laminar flow remains valid, as the Reynolds number at the highest velocity is approximately 480, well below the critical threshold for turbulence. Based on this, the fully developed entrance length was calculated to be 118 mm. The total artery length was set to 300 mm, which provides a 32 mm buffer between the entrance length and the start of the stenosis (given a maximum stenosis length of 20 mm). This configuration effectively eliminates any entrance effects near the stenosis.

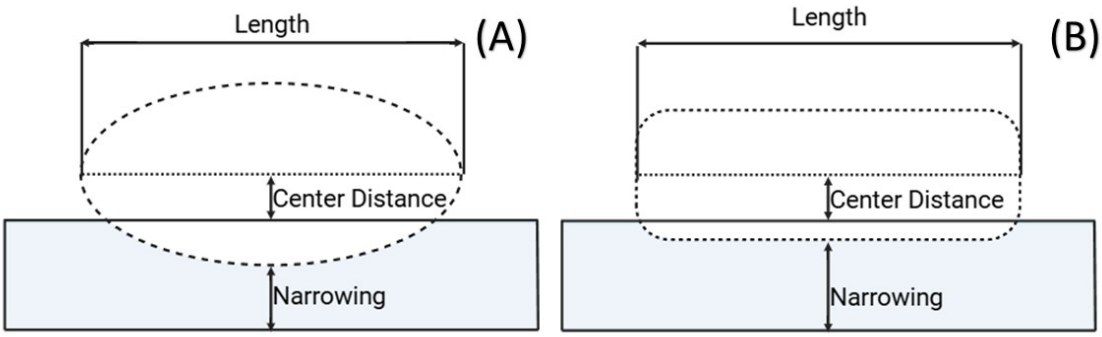


Figure 2.3: (A) Ellipsoid and (B) Super-ellipsoid stenosis geometries

#### 2.2.2.2 Final Geometries

The geometry was finalized based on the fully developed entrance length and the elliptical shape equations discussed previously. As described, the model consists of a cylindrical tube with an elliptical indentation located at its center. Figure 2.5 illustrates the geometry and its associated variables. The table that summarizes the design parameters for all simulation cases considered in this study can be found in the Appendix A.

### 2.2.3 Numerical Setup

We assumed laminar flow behavior throughout the study. This assumption is justified, as discussed earlier, since even at the highest inlet velocity, the Reynolds number remains low, approximately 480, below the threshold for turbulence. Furthermore, the vessel under investigation is not conducive to turbulent flow conditions, as noted in [11]. This assumption also simplifies the simulation process, as ANSYS Fluent supports only the Power-Law and Carreau non-Newtonian models within its laminar flow solver.

To implement the viscosity models, ANSYS Fluent requires specific parameters for each model. For the Newtonian model, only the fluid density $\rho$ and dynamic viscosity $\mu$ are needed. For the Power-Law model, in addition to the model

constants, the minimum and maximum viscosity values must be specified. The same $k$ and $n$ values defined in Equation 2.3 were used. To ensure consistency in the viscosity range between the Carreau and Power-Law models, we selected $\mu_0$ and $\mu_\infty$ from the Carreau model (given in Equation 2.4) as the maximum and minimum viscosity values for the Power-Law model. For the Carreau model, ANSYS requires the parameters $\lambda$, $\mu_0$, and $\mu_\infty$, which were provided as listed in Equation 2.4.

#### 2.2.3.1 Boundary Conditions

The boundary conditions include the inlet, outlet, and vessel walls. These conditions were defined based on the previous study in [14].

- **Inlet:** A constant velocity profile is applied. Due to vascular viscous resistance and compliance, the pulsatile nature of the flow is damped in the microcirculation [21].
- **Outlet:** A constant atmospheric pressure condition is applied.
- **Walls:** A no-slip boundary condition is applied. This condition enforces zero blood velocity at the wall surface, resulting in a velocity gradient across the vessel cross-section.

#### 2.2.3.2 Solver Settings

For the solution method, pressure–velocity coupling was handled using the Coupled scheme in ANSYS Fluent. For spatial discretization, the gradient was computed using the Least Squares Cell-Based method, pressure was discretized using a second-order scheme, and momentum equations were solved using a second-order upwind scheme. The pseudo-time stepping method was used to control the global time step. However, since the flow is steady, this setting only influences the Courant–Friedrichs–Lewy (CFL) number and does not affect the physical time

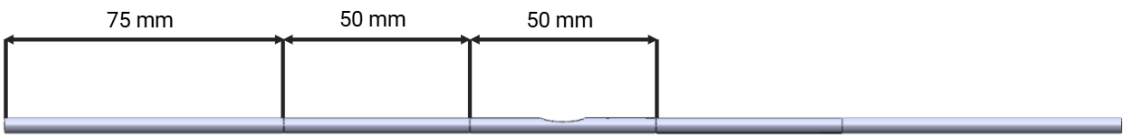


Figure 2.4: Length of each section

evolution. Convergence was assessed using the residuals of continuity and the velocity equations in the $x$, $y$, and $z$ directions. Based on preliminary tests, if the residuals fell below $10^{-4}$, the simulation was considered converged, as further iterations did not significantly change the solution. A total of 500 iterations were performed for each simulation, with a pseudo-time step size of 0.01.

#### 2.2.3.3 Mesh Setup

Meshing was performed in ANSYS Mechanical Meshing due to its convenience in allowing transitions between mesh modifications and Fluent simulations throughout the investigation. The CFD meshing mode was selected, with Fluent chosen as the solver preference. The element order was set to linear. After defining the geometry according to the fully developed flow length and the stenosis shape criteria, the total artery length was set to 300 mm, with the stenosed region occupying 50 mm. Since the stenosed segment is significantly smaller than the rest of the artery, it was computationally efficient to divide the geometry into multiple regions. This allowed a finer mesh in the stenosed region to enhance accuracy, while coarser mesh elements were used in the remaining sections to reduce computational cost for the parametric study. To achieve a smooth and gradual transition in element size, the geometry was divided into five segments: two at the ends of the cylinder, two adjacent to the center, and one in the middle containing the stenosed region.

The central segment, which contains the stenosed region, was meshed using tetrahedral elements with a size of 0.1 mm. This segment is 50mm long, corresponding to the maximum stenosis length plus a small buffer to eliminate mesh-induced artifacts in the flow. The two segments adjacent to the center

are identical, each with an element size of 0.2 mm and a length of 50 mm. The outermost segments, located at both ends of the cylinder, have a coarser element size of 0.4 mm and a length of 75 mm. The increased length and coarser mesh in the end segments enhance computational efficiency, while the gradual change in element size between the adjacent and central segments ensures a smooth mesh transition into the stenosed region, minimizing numerical diffusion and preserving flow accuracy.

Inflation layers were applied to the cylinder walls, using 5 layers with a growth rate of 1.2. Inflation layers are structured mesh layers generated adjacent to wall boundaries to accurately resolve near-wall flow behavior. They allow for finer resolution in the direction normal to the wall, enabling the mesh to capture steep velocity and shear stress gradients without requiring excessive refinement throughout the entire domain. This approach is particularly important in this study, as the stenosed region is significantly narrower than the nominal artery diameter. The number of inflation layers, element sizes in each segment, and the growth rate were optimized through mesh quality testing, specifically by monitoring skewness values. For each mesh modification, the maximum skewness was evaluated to ensure it remained below 1. Additionally, the location of the maximum skewness was examined. For example, if high skewness was observed near the stenosed wall, the inflation parameters were adjusted, such as increasing the growth rate, to reduce mesh distortion. The final mesh yielded a maximum skewness of 0.895 and an average skewness of 0.176.

Although the inflation layers and segment lengths were important for ensuring overall mesh quality, particularly by minimizing skewness and enabling smooth transitions, the most critical factor in achieving mesh accuracy was the element size, especially within the stenosed region. A mesh independence study was conducted by progressively refining the mesh in this region, with near-wall element sizes ranging from 10 mm to 0.2 mm. For each mesh configuration, the resulting maximum wall shear stress (WSS) values were compared against benchmark data from Jedrzejczak *et al.* [14]. As summarized in Table 2.1, Mesh-6 produced WSS results for all three viscosity models—Newtonian, Power-Law, and Carreau—that were nearly identical to those of Mesh-4, but with a coarser mesh in the outer

regions, resulting in a significantly lower total element count. Although Mesh-5 yielded slightly higher WSS values, the difference was not substantial enough to justify the increased computational cost. Based on this evaluation, Mesh-6 was selected as the final mesh due to its optimal balance between numerical accuracy and computational efficiency.

In the finalized mesh, the stenosed and adjacent middle regions retain the fine resolution used in Mesh-4, while the outermost segments employ slightly larger element sizes to reduce computational cost. These outer regions have minimal influence on the flow characteristics within the stenosis, so their coarsening had a negligible impact on the simulation results. The final mesh consists of 894,191 nodes and achieves a maximum skewness of 0.895 and an average skewness of 0.176, both within acceptable limits for reliable and accurate CFD simulations.

Table 2.1: Mesh Independence Analysis

| | Mesh-1 | Mesh-2 | Mesh-3 | Mesh-4 | Mesh-5 | Mesh-6 |
|---|---|---|---|---|---|---|
| Middle Section Element Size | 5.0 | 2.5 | 1.25 | 0.1 | 0.8 | 0.1 |
| Near Section Element Size | 1.0 | 0.5 | 0.25 | 0.25 | 0.2 | 0.2 |
| Outer Section Element Size | 1.5 | 0.75 | 0.375 | 0.375 | 0.3 | 0.4 |
| Newtonian Wall Shear Stress | 162 | 218 | 220 | 252 | 256 | 252 |
| Power Law Wall Shear Stress | 98.5 | 145 | 184 | 298 | 286 | 298 |
| Carreau Wall Shear Stress | 97 | 146 | 185 | 300 | 304 | 303 |

#### 2.2.3.4 Verification

To validate our method, we performed a numerical comparison with Jedrzejczak *et. al*'s [14] wall shear stress results. Since the numerical data of the article was not available, the 1-D quantitative profile extraction method was performed. A horizontal line was extracted from the 60% of the height of both vessels. The RGB color values along this line were sampled and converted to physical wall shear stress values based on the color scale. Then, both datasets were normalized for a comparison independent of scale differences. The resulting profiles were plotted

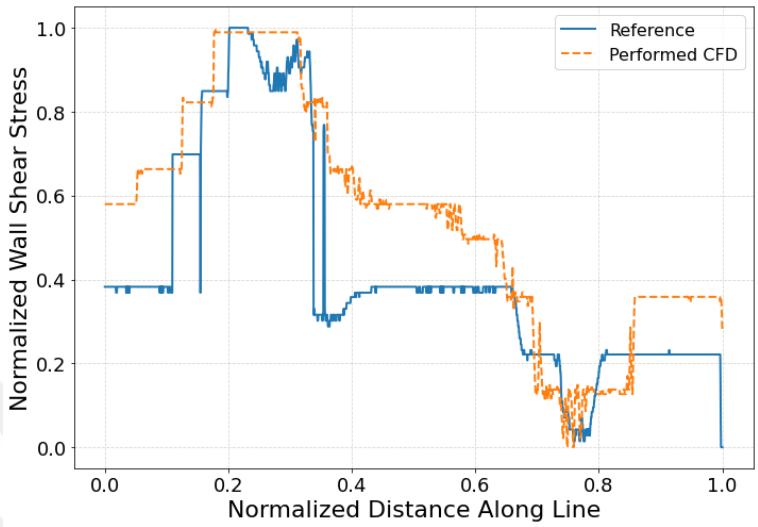


Figure 2.5: Line Profile Comparison of Wall Shear Stress

and assessed using the Pearson correlation coefficient. The Pearson correlation is used to evaluate the similarity in the distribution of wall shear stress along the line. The coefficient was found to be 0.89, indicating a strong linear agreement between the simulated and reference profiles. This high correlation supports the validity of ours CFD model in reproducing the expected shear stress distribution.

## 2.3 Results

Since the aim of this study is to investigate the effects of stenosis length, height, and shape, the results are presented by analyzing variations in the parameters $L$ and $N$ for ellipsoidal stenoses ($n = 2$), and $n$ for superellipsoidal stenoses. While $L$, and $N$, determine the geometric dimensions of the stenosis (i.e., its axial length and severity), the parameter $n$ defines the shape or sharpness of the constriction. Based on these variations, we examine changes in key hemodynamic variables: maximum wall shear stress, pressure drop, velocity, and strain rate. Among these, wall shear stress and pressure drop are emphasized, as they are widely used clinical indicators in the diagnosis of atherosclerosis. In addition to scalar values, the spatial distributions of wall shear stress, velocity, vorticity, and strain rate are visualized using Fluent. These profiles provide insights not only

into the magnitude but also into the spatial characteristics of the flow, which are influenced by stenosis geometry. In particular, changes in velocity and vorticity profiles in response to $L$, $N$, and $n$ are discussed in detail. Before analyzing the parametric effects, the results are first validated by comparing wall shear stress, pressure, velocity, vorticity, and strain rate profiles with reference data for healthy arterial flow, to ensure physiological consistency.

### 2.3.1 Comparison With Healthy Arteries

Throughout this investigation, we compared the flow characteristics of stenosed and non-stenosed (healthy) arteries. Compared to healthy arteries, stenoses are expected to increase maximum wall shear stress, peak velocity, pressure drop, and strain rate. Modeling a healthy artery and comparing its hemodynamic variables with those of stenosed arteries provides a reference point that supports the accuracy and physiological relevance of our findings. For the healthy artery, we modeled a cylindrical geometry with a diameter of $D = 4$ mm and a total length of 300 mm, consistent with the geometries used in the stenosed cases. At inlet velocities of 0.26 m/s and 0.52 m/s, the simulation results yielded the following maximum values: wall shear stress of 2.1 Pa and 4.3 Pa, velocities of 0.48 m/s and 0.925 m/s, pressure drops of 44 Pa and 85 Pa, and strain rates of 600 $s^{-1}$ and 1250 $s^{-1}$, respectively. All variables exhibited higher magnitudes in the presence of stenosis, confirming the expected physiological trend. While the literature reports typical wall shear stress values for healthy arteries in the range of 0.5–1.5 Pa [22], our results remain reasonable given the variability in blood flow velocity and vessel diameter between individuals.

### 2.3.2 Ellipsoid Results

In the ellipsoidal case, the effects of stenosis length and height are investigated by varying the parameters $L$ and $N$ while keeping the shape exponent fixed at $n = 2$. The results presented in this section correspond to these geometries. The

influence of each parameter on the hemodynamic variables is discussed in detail in the following subsections.

#### 2.3.2.1 Effects of Length L

To evaluate the effect of stenosis length on maximum wall shear stress, pressure drop, velocity, and strain rate, the parameter $L$ was varied across 12 mm, 16 mm, and 20 mm. Simulations were conducted at inlet velocities of 0.26 m/s and 0.52 m/s, and for stenosis heights $N = 1$, 2, and 3 mm, to ensure consistency and comparability with the results from Jedrzejczak *et al.* [14]. Across all $N$ values, increasing the stenosis length resulted in a decrease in maximum wall shear stress, an increase in pressure drop, and a decrease in maximum strain rate. The maximum velocity remained largely unchanged for $N = 2$ and $N = 3$, but increased noticeably for $N = 1$.

Although the decrease in maximum wall shear stress with increasing stenosis length is less pronounced for $N = 2$ and $N = 3$, it becomes more evident for the narrower case of $N = 1$. This trend is consistent with findings with Tian *et al.* [22], which also report reduced maximum wall shear stress and strain rate as the length of the stenosis decreases. This behavior can be explained by the fact that a longer stenosis leads to a smoother spatial transition in wall shear stress and its gradient [22], whereas a shorter stenosis introduces a more abrupt change in geometry, resulting in elevated shear stress concentrations. The variation in maximum wall shear stress as a function of stenosis length $L$ is illustrated in Figure 2.6.

#### 2.3.2.2 Effects of Length N

To investigate the effects of stenosis height on maximum wall shear stress, pressure drop, velocity, and strain rate, the parameter $N$—defined as the remaining open diameter through which blood can flow—was varied from 3 mm to 2 mm to 1 mm. Simulations were conducted for inlet velocities of 0.26 m/s and 0.52 m/s, and for

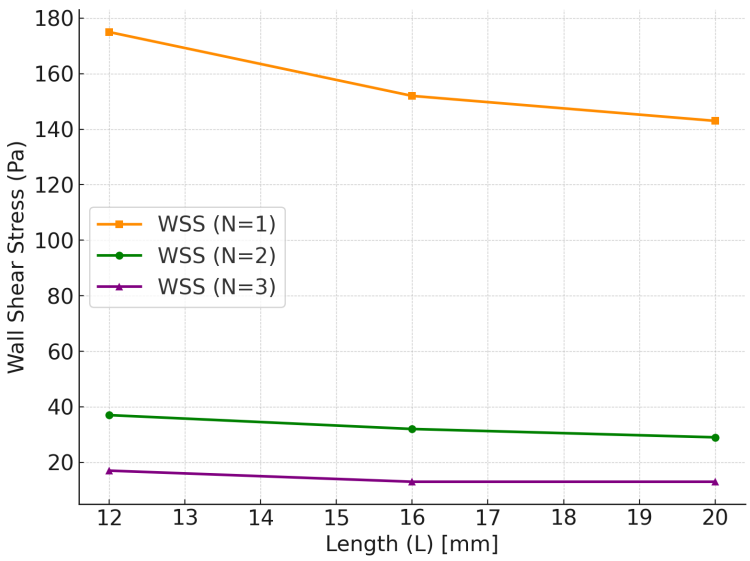


Figure 2.6: Shear Stress Change in Various Lengths for n = 2 and Velocity = 0.52 cm/s

stenosis lengths $L = 12$, 16, and 20 mm, ensuring sufficient data for comparison with the results presented by Jedrzejczak *et al.* [14]. As $N$ decreased, as the degree of narrowing increased, maximum wall shear stress, maximum velocity, pressure drop, and strain rate all increased.

Furthermore, the changes in maximum wall shear stress resulting from variations in $N$ are significantly more pronounced than those associated with changes in $L$. This indicates that $N$ has a greater impact than $L$ on both maximum wall shear stress and pressure drop. Specifically, as $N$ decreases from 3 mm to 1 mm, the maximum wall shear stress increases by approximately 929%. In contrast, decreasing $L$ from 20 mm to 12 mm results in a much smaller increase of only about 22%. These results clearly demonstrate that variations in $N$ induce substantially greater changes in hemodynamic behavior than equivalent variations in $L$.

This relationship is also discussed by Tian *et al.* [22], where similar findings are reported, further supporting the results of this study. The changes in maximum wall shear stress and pressure drop as a function of $N$ are presented in Figure 2.7.

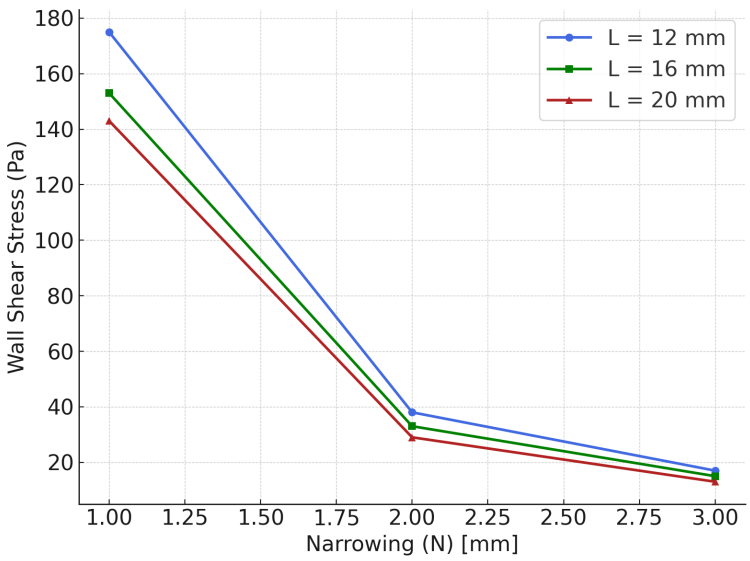


Figure 2.7: Shear Stress Change in Various Narrowings for n = 2 and Velocity = 0.52 cm/s

To validate the wall shear stress distribution, visual results obtained from Fluent were compared with those reported by Jedrzejczak *et al.* [14]. The profiles show consistent behavior: the highest wall shear stress occurs near the entrance of the stenosed region and gradually decreases downstream, away from the stenosis. This trend can be observed in Figure 2.8. The same relationship between stenosis severity ($N$) and wall shear stress is also discussed by Jedrzejczak *et al.* [14], further supporting the findings of this study.

### 2.3.3 Superellipsoid Results

The superellipsoid results include stenosed geometries with shape exponent values $n = 3$, 4, and 5, 6, and stenosis heights $N = 1$, 2, and 3 mm, while the length $L$ is held constant at 16 mm. The effects of each parameter on the hemodynamic variables are discussed in detail in the following subsections.

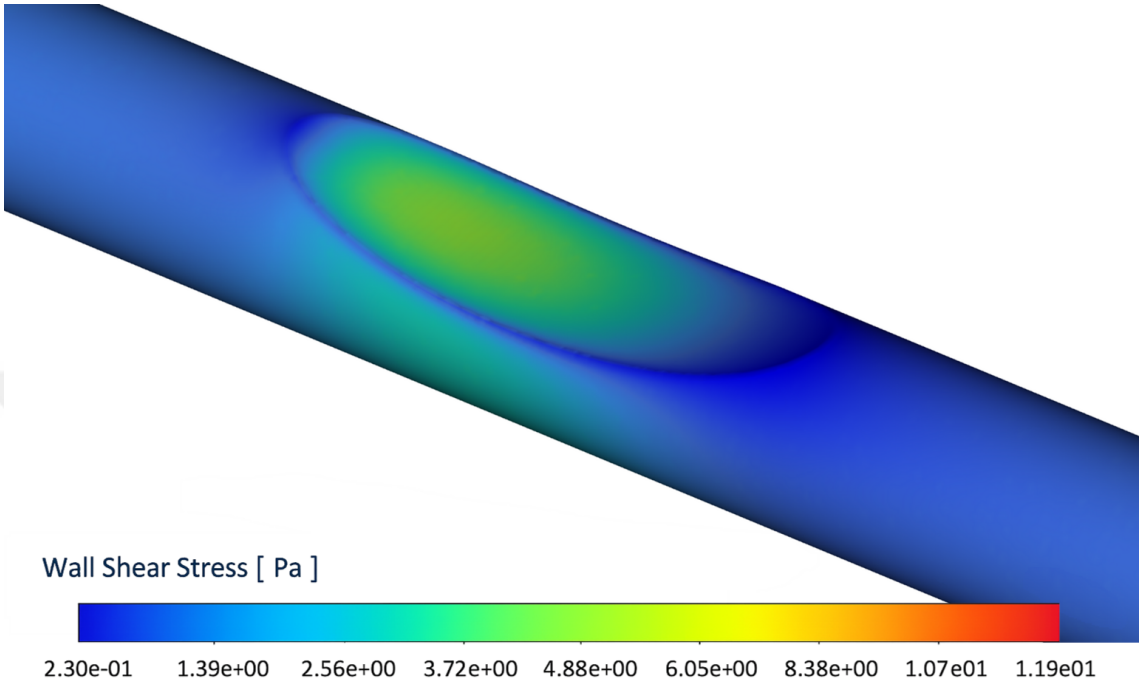


Figure 2.8: Wall Shear Stress in 3D model

#### 2.3.3.1 Effects of order "n"

To evaluate the effect of the shape exponent $n$ on maximum wall shear stress, velocity, pressure drop, and maximum strain rate, $n$ was varied while keeping $L$ and $N$ constant. The results for all stenosis heights ($N$ values) are presented in Figure 2.9. Across all $N$ values, increasing $n$ led to higher maximum wall shear stress, pressure drop, and maximum strain rate. However, no consistent or significant relationship was observed between $n$ and the maximum velocity.

#### 2.3.3.2 Effects of Length N

Since $N$ showed significant changes in the wall shear stress, velocity, pressure drop, and strain rates for the ellipsoid geometries, to have a more holistic approach, the effects of changing $N$ is also investigated for the superellipsoid geometries. For this, all the variables except $n$ and $N$ are varied for the two velocities, 0.26 and 0.52 $m/s$. For each $n$ and velocity case, as $N$ increased, the maximum

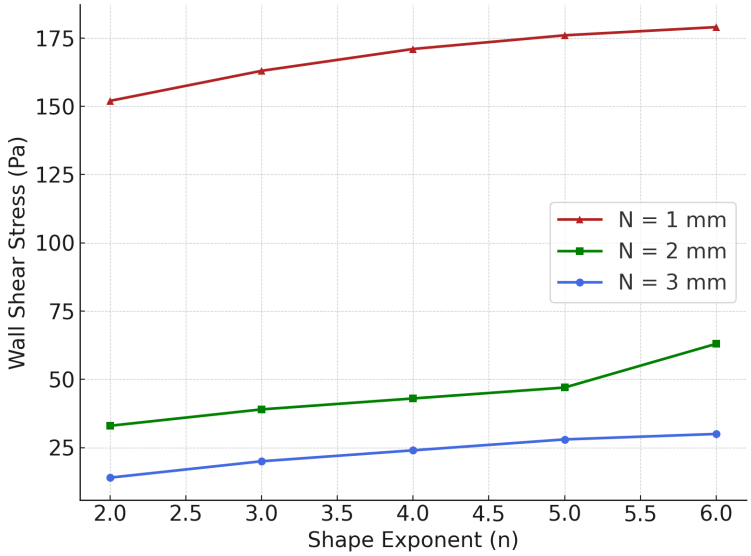


Figure 2.9: Wall Shear Stress in Various Power Index for Length = 16 mm and Velocity = 0.52 cm/s

wall shear stress, maximum velocity, pressure drop and maximum strain rate decreased. The comparison result can be seen in the Figure 2.10 This is consistent with both Tain *et al.*'s findings [22] and our findings for the elliptical geometry.

Since $N$ showed significant influence on wall shear stress, velocity, pressure drop, and strain rate in the ellipsoidal geometries, its effect was also investigated for the superellipsoidal cases to provide a more comprehensive analysis. In this part of the study, all parameters were held constant except $n$, $N$ and inlet velocities. For each combination of $n$ and inlet velocity, increasing $N$ resulted in decreased maximum wall shear stress, peak velocity, pressure drop, and maximum strain rate. The comparison results are presented in Figure 2.10. These trends are consistent with the findings of both Tian *et al.*'s [22] and our earlier results for the ellipsoidal geometries.

Furthermore, the changes in maximum wall shear stress resulting from variations in $N$ are significantly greater than those caused by changes in $n$. This indicates that $N$ has a stronger influence on maximum wall shear stress than $n$, which controls the sharpness of the stenosis geometry. Since $n$ primarily affects the smoothness of the stenosis profile, while $N$ directly determines the remaining

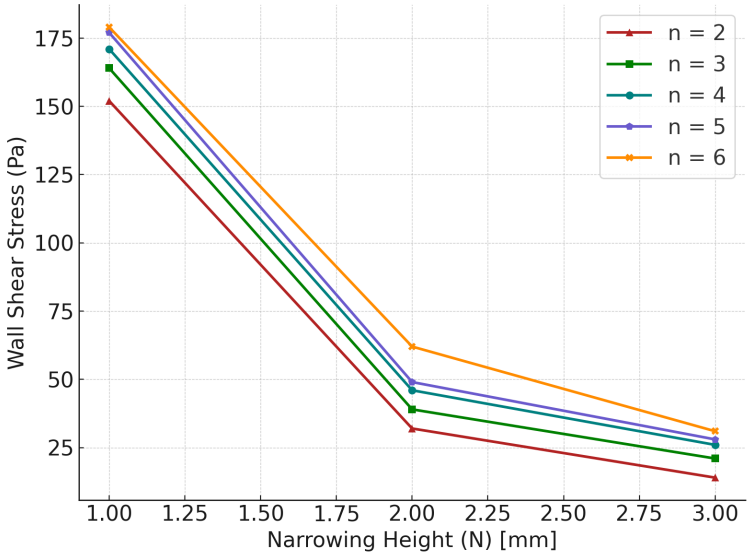


Figure 2.10: Wall Shear Stress in Various Narrowing for Length = 16 mm and Velocity = 0.52 cm/s

lumen through which blood can flow, it can be concluded that stenosis height has a more dominant effect on shear stress than shape sharpness.

The maximum wall shear stress increases by 1069%, 746%, 641%, 541%, and 493% respectively for $n = 2$ to $n = 6$ as the narrowing height $N$ decreases from 3 to 1. This results in an average increase of 698%, indicating that $N$ has a strong influence on wall shear stress. On the other hand, when the shape exponent $n$ increases from 2 to 6, the shear stress increases by 132%, 93%, and 18% for $N = 3$, 2, and 1 respectively, leading to an average increase of 81%.

Similarly, the maximum pressure drop increases by 3604%, 3240%, 3003%, 2745%, and 2684% as $N$ decreases from 3 to 1 for each $n$, with an average increase of 3055%. In contrast, increasing $n$ leads to pressure drop increases of 66%, 33%, and 25% for $N = 3$, 2, and 1 respectively, with an average increase of 42%.

These results clearly demonstrate that the narrowing height $N$ has a significantly stronger effect on both wall shear stress and pressure drop compared to the shape exponent $n$. Numerical results of each stenosis model are given in the Section 5.2.

## 2.4 Conclusion

This chapter investigated the influence of stenosis geometry which is defined by length (**L**), height (**N**), and shape exponent (**n**) on important hemodynamic variables such as wall shear stress and velocity, using CFD simulations. Both ellipsoid ($n = 2$) and superellipsoid ($n = 3$–$6$) geometries were examined under two physiologically relevant inlet velocities.

The results revealed that among the geometric parameters, stenosis height **N** had the most significant effect on all measured variables. As **N** decreased, there was a steep increase in WSS and pressure drop, with average changes exceeding 300% in some configurations. In contrast, changes in stenosis length **L** led to more moderate shifts, while the shape exponent **n** also influenced outcomes but to a lesser degree than **N**. These trends held true for both ellipsoid and superellipsoid models. The computational predictions were validated against a healthy artery model and aligned with existing literature results, especially in terms of maximum WSS localization and downstream recovery patterns.

Although these CFD based analyses provide high-fidelity insights into the fluid dynamics of stenosed vessels, they are computationally expensive and time consuming, especially when investigating a wide range of geometric and physiological parameters. This makes it impractical to rely on CFD simulations for real-time clinical assessment or large-scale parametric studies. Therefore, the development of accurate and efficient surrogate data driven models becomes essential. To address this need, the next chapter introduces a machine learning based approach to predictive hemodynamics. By learning from CFD generated data, machine learning models can approximate flow metrics such as WSS and velocity with minimal computational cost. However, before applying such models to biomedical flows, it was necessary to first gain experience in the complete pipeline of model development.

As concluded in the following chapter, building a predictive model is essential for improving the accuracy of vessel hemodynamic calculations. Recognizing the

steep learning curve associated with machine learning based modeling, We chose to first gain experience in a more approachable and well documented domain, aerodynamic coefficient prediction for airfoils which is supported by extensive datasets, established evaluation standards, and a wealth of prior research. This foundational work serves as a stepping toward adapting the same techniques for vascular flow modeling, ultimately aiming to accelerate the prediction of disease-relevant indicators in stenosed arteries.

# Chapter 3

# Introduction to Machine Learning - Machine Learning Assisted Optimal Airfoil Design at High Mach Numbers

## 3.1 Motivation

As concluded in the previous chapter, a predictive model is essential for improving the accuracy of vessel hemodynamic calculations. However, given my lack of prior experience with machine-learning–based models, it was necessary first to build foundational expertise in model selection, training, and validation. To that end, I chose to begin in a more well-documented domain, predicting aerodynamic coefficients of airfoils due to it offers mature benchmarks, publicly available datasets, and established evaluation metrics. Some of the previous work about aerodynamic coefficient prediction will be mentioned in the following sections. This environment provided the sandbox for mastering the end-to-end workflow of data preprocessing, feature engineering, and performance assessment. During this exploration, I discovered a novel feature for airfoil characterization that ease

the preprocessing while keeping the accuracy high. The identification and integration of this feature will serve as the focal point of this chapter, laying the groundwork for the eventual adaptation of these techniques to vascular stenosis prediction.

## 3.2 Methodology

### 3.2.1 Workflow

The workflow begins with obtaining CFD results and converting them into an input format suitable for training a neural network. To achieve satisfactory performance, both the input format and network architecture undergo an optimization process. These components are iteratively refined to improve accuracy while minimizing computational cost. An overview of the entire workflow is presented in Figure 3.1, and the individual steps will be described in detail in the following sections.

### 3.2.2 Data Set

The dataset includes two aerodynamic coefficients, lift and moment, for various NACA airfoils under different flow conditions, specifically varying angles of attack and Mach numbers. The angle of attack ranges from –8° to 8°, and the Mach number varies from 0.1 to 0.79. On the geometry side, the dataset comprises 28 distinct NACA 4-digit airfoils and 9 NACA 5-digit airfoils. This results in a total of 2,800 NACA 4-digit cases and 900 NACA 5-digit cases.

The drag coefficient was excluded from this study, as its distribution was heavily biased toward zero. This imbalance could lead to poor predictive precision and may hinder the learning capability of the neural network.

The data used in this study were obtained using the ANSYS Fluent CFD

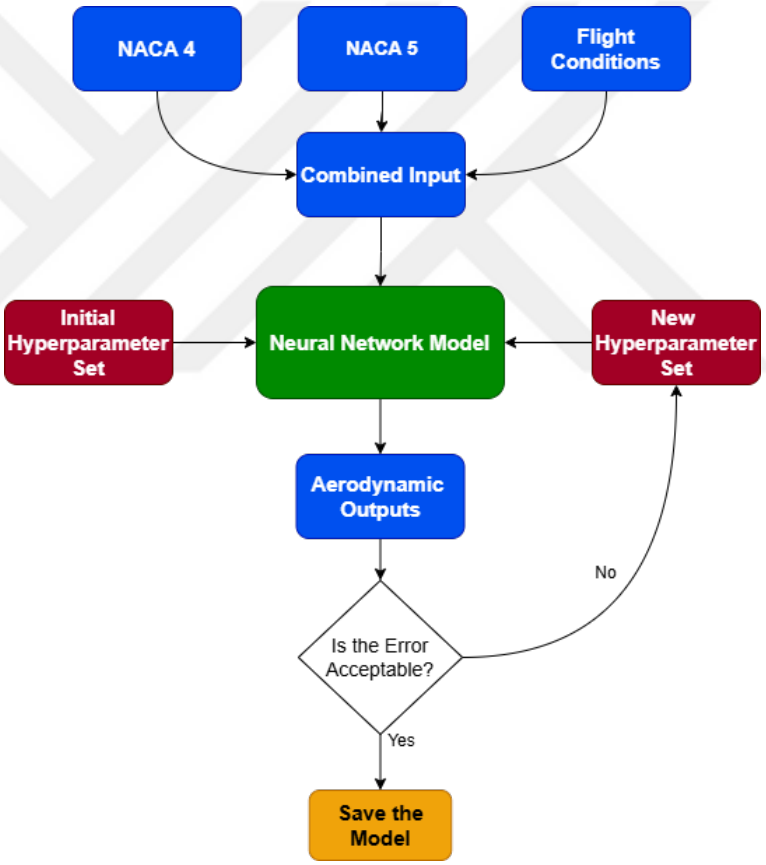


Figure 3.1: Flow Diagram of the Neural Network Creation

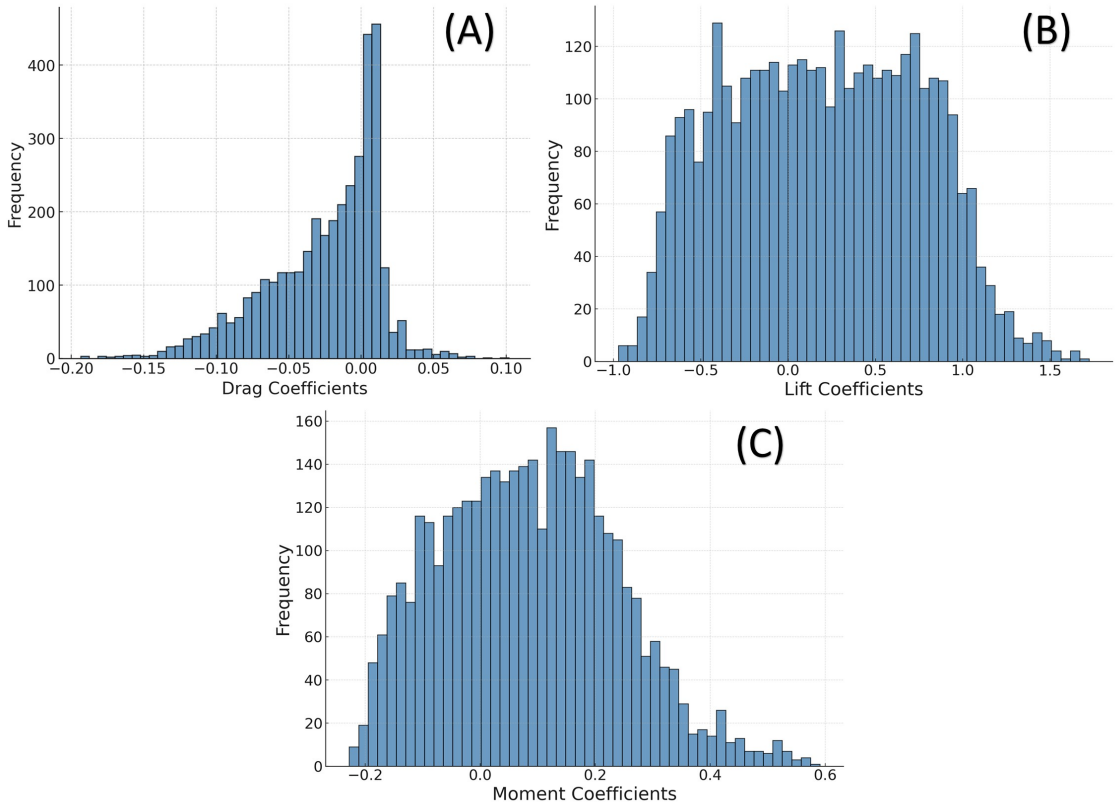


Figure 3.2: Distribution of (A) Drag, (B) Lift and (C) Moment Coefficients

solver. The SST $k$–$\omega$ turbulence model was selected due to its robustness and accuracy across a wide range of flow regimes. Since the dataset includes Mach numbers up to 0.8, operating near the transonic range, the SST $k$–$\omega$ model provides a conservative and reliable choice for capturing complex flow behavior.

The SST $k$–$\omega$ model is a two-equation turbulence model, meaning it introduces two additional transport equations to represent the turbulent characteristics of the flow. It blends the advantages of the standard $k$–$\omega$ and $k$–$\epsilon$ models using a shear stress transport (SST) function. Near the wall, it behaves like a $k$–$\omega$ model for improved accuracy in boundary layers, while in the free-stream region, it transitions to a $k$–$\epsilon$ model for better stability.

The first added equation governs the turbulent kinetic energy $k$ (Equation 3.1), and the second governs the specific dissipation rate $\omega$ (Equation 3.2). The

blending function $F_1$ controls the transition between the two models: $F_1 = 1$ near the wall and $F_1 = 0$ in the free-stream region, allowing a smooth combination of both formulations.

$$\frac{D\rho k}{Dt} = P_k - \beta^* k\omega + \frac{\partial}{\partial x_j}[(\nu + \sigma_k \nu_T)\frac{\partial k}{\partial x_j}] \tag{3.1}$$

$$\frac{D\rho \omega}{Dt} = \alpha S^2 - \beta^* \omega^2 + \frac{\partial}{\partial x_j}[(\nu + \sigma_\omega \nu_T)\frac{\partial \omega}{\partial x_j}] + 2(1 - F_1)\sigma_{\omega 2}\frac{1}{\omega}\frac{\partial k}{\partial x_i}\frac{\partial \omega}{\partial x_i} \tag{3.2}$$

The shear stress transport formulation includes a limiting mechanism to prevent excessive eddy viscosity, which can otherwise lead to non-physical solutions in adverse pressure gradient regions. This is achieved using a shear stress limiter, which controls the magnitude of turbulent viscosity and enhances the model's stability and accuracy in boundary layer flows [23, 24].

Equation 3.3 presents the shear stress transport function, where the blending function $F_2$ is used to facilitate a smooth transition between turbulence models. The $F_2$ function operates similarly to $F_1$, but is specifically designed to manage the turbulent viscosity formulation and maintain consistency in the eddy viscosity limit.

$$\nu_t = \frac{\alpha_1 k}{\max(\alpha_1 \omega, SF_2)} \tag{3.3}$$

### 3.2.3 Input Format

Geometry processing methods such as Class-Shape Transformation (CST) are often complex and computationally intensive. Therefore, in this study, we opted to use raw NACA airfoil identifiers directly, leveraging their analytical structure. NACA airfoils are defined by digit-based naming conventions that encode key aerodynamic characteristics, primarily the mean camber line, thickness distribution, and slope of curvature. Depending on the complexity of the airfoil, the number of digits can vary (e.g., NACA 4-digit vs. 5-digit series). While more recent NACA variants introduce additional parameters, the core geometric features, camber and thickness, remain central to all formulations [25].

Inspired by this foundational structure, we created a unified input matrix that combines both NACA 4-digit and 5-digit airfoils. This approach allows us to encode the geometric properties directly into the machine learning model without the need for coordinate-based preprocessing.

The NACA 4-digit airfoil naming convention encodes three primary geometric parameters into a four-digit format denoted as ABXX: The first digit (A) represents the maximum camber as a percentage of the chord length. The second digit (B) indicates the location of maximum camber from the leading edge, expressed in tenths of the chord. The last two digits (XX) define the maximum thickness of the airfoil as a percentage of the chord. For example, the NACA 2410 airfoil has a maximum camber of 2%, located at 40% of the chord length, and a maximum thickness of 10% of the chord.

To compute the full coordinates of the airfoil surface, the following standard analytical equations are used to define both the mean camber line and the thickness distribution:

1. Set the number of of wanted x coordinate from 0 to maximum chord length "l"

2. Calculate the mean curvature length using following formula:

$$y_c = \frac{A}{B^2}(2Bx - x^2) \qquad \text{for } 0 \leq x \leq p \tag{3.4}$$

$$y_c = \frac{A}{(1-B)^2}[(1-2B) + 2Bx - x^2] \qquad \text{for } p < x \leq c \tag{3.5}$$

3. Calculate the thickness calculation for both sides by using:

$$\pm y_t = \frac{XX}{0.2}(0.2969\sqrt{x} - 0.126x - 0.3516x^2 + 0.2843x^3 - 0.1015x^4) \tag{3.6}$$

4. Then calculate outer surface by

$$x_{outer} = x \pm y_t \sin\theta \tag{3.7}$$

$$y_{outer} = y_c \pm y_t \cos\theta \tag{3.8}$$

where $\tan\frac{dy_c}{dx} = \theta$

The NACA 5-digit airfoil series uses a slightly different representation than the 4-digit series, incorporating both similar and newly defined parameters. This naming convention consists of four parts, denoted as CBDXX, where each letter corresponds to a specific geometric or aerodynamic property: The first digit (C), when multiplied by $\frac{3}{20}$, gives the design lift coefficient (in tenths). The second digit (B) denotes the position of maximum camber from the leading edge, as in the NACA 4-digit series. The third digit (D) indicates the type of camber: A value of 0 refers to a normal camber line. A value of 1 indicates a reflexed camber, where the camber line curves upward near the trailing edge. Reflexed airfoils are often used to improve longitudinal stability. The last two digits (XX) define the maximum thickness as a percentage of the chord, again identical to the NACA 4-digit format.

Thus, while the position of camber and thickness specification remains consistent between NACA 4 and 5 series, the NACA 5 series introduces a direct relation to design lift and camber type, offering more control for aerodynamic performance tailoring. To compute the full airfoil coordinates for a NACA 5-digit airfoil, the corresponding camber line equation (normal or reflex) and the thickness distribution equation are used:

1. Set the number of of wanted x coordinate from 0 to maximum chord length "l"

2. Calculate the mean curvature length using following formula if the chamber is normal:

$$y_c = \frac{k_1}{6}(x^3 - 3mx^2 + m^2(3-m)x) \qquad \text{for } 0 \leq x \leq r \tag{3.9}$$

$$y_c = \frac{k_1 r^3}{6}[1-x] \qquad \text{for } r < x \leq c \tag{3.10}$$

3. Calculate the mean curvature length using following formula if the chamber is reflex:

$$y_c = \frac{k_1}{6}\left[(x-r)^3 - \frac{k_2}{k_1}(1-r)^3 x - r^3 x + r^3\right] \qquad \text{for } 0 \leq x \leq r \tag{3.11}$$

$$y_c = \frac{k_1 r^3}{6}\left[\frac{k_2}{k_1}(x-r)^3 - \frac{k_2}{k_1}(1-r)^3 x - r^3 x + r^3\right] \quad \text{for } r < x \leq c \tag{3.12}$$

4. Calculate the thickness with the same formulas as the NACA4

5. Then calculate outer surface with same formula as the NACA4

Table 3.1: Coefficients for NACA 5 - 2BDXX Series

| Digits BD | r | $k_1$ | $k_2/k_1$ |
|---|---|---|---|
| 10 | 0.0580 | 361.400 | -[1] |
| 20 | 0.1260 | 51.640 | - |
| 21 | 0.1300 | 51.990 | 0.000764 |
| 30 | 0.2025 | 15.957 | - |
| 31 | 0.2170 | 15.793 | 0.00677 |
| 40 | 0.2900 | 6.643 | - |
| 41 | 0.3180 | 6.520 | 0.0303 |
| 50 | 0.3910 | 3.230 | - |
| 51 | 0.4410 | 3.191 | 0.1355 |

[1] For normal chambers, there is no need to know $\frac{k_2}{k_1}$.

To integrate both NACA 4-digit and NACA 5-digit airfoils into a single predictive model, we focused on their shared geometric parameters: B (the location of maximum camber) and XX (the maximum thickness). In order to create a unified input representation suitable for machine learning, we devised a new 6-digit encoding format: ACBDXX. In this format: A and C represent parameters that are exclusive to either NACA4 or NACA5. B and XX are common to both series and retain their original definitions. D represents the camber type in NACA5 airfoils (0 for normal camber, 1 for reflex camber). For airfoil types that lack a specific parameter, we assign a placeholder value of –1. For example: A NACA 2410 airfoil (from the 4-digit series) is encoded as: 2 –1 4 –1 10. A NACA 25112 airfoil (from the 5-digit series) is encoded as: –1 2 5 1 12.

After encoding the airfoil geometry using the ACBDXX format, we append two additional columns representing the flight conditions, the angle of attack and the Mach number. This results in an 8-dimensional input vector for each sample, combining both geometric and aerodynamic features to serve as the complete input matrix for the neural network model.

$$\mathbf{X} = [\mathbf{A} \quad \mathbf{C} \quad \mathbf{B} \quad \mathbf{D} \quad \mathbf{XX} \quad \mathbf{Mach} \quad \alpha] \tag{3.13}$$

The entire dataset is divided into three subsets: training, validation, and test sets. Approximately 78% of the data (2900 samples) is used for the training process. To prevent overfitting, a separate validation set comprising 16% of the data (600 samples) is employed during training. This set is used to monitor the model's performance and guide weight updates without directly influencing the model's training. The remaining 6% of the data (200 samples) constitutes the test set, which is exclusively used to evaluate the performance of the finalized network architecture. Unlike the validation set, the test set is not used at any stage of training or model optimization, ensuring an unbiased assessment of the model's generalization capability.

The data for the training, validation, and test sets were organized under two distinct scenarios to evaluate the generalization capability of the model.In the first scenario, the entire dataset was randomly shuffled and then split into training, validation, and test sets. As a result, the validation and test sets contained previously seen NACA airfoil geometries, but with unseen combinations of flight conditions (angle of attack and Mach number). This ensured that the network was tested on familiar airfoil shapes under new conditions, allowing assessment of how well the model interpolates within known geometry types.

In the second scenario, the data was split based on airfoil geometries to test the model's ability to generalize to completely unseen NACA airfoils. Specifically, 5 NACA 4-digit and 3 NACA 5-digit geometries (along with all their associated flight conditions) were excluded from the training data. From these excluded geometries, 4 NACA 4 and 2 NACA 5 airfoils were used for validation, allowing the model to be evaluated during training on new geometries. Finally, the remaining 2 airfoils, 1 from NACA 4 and 1 from NACA 5, were reserved for the test set, representing airfoils the model had never encountered during training or validation. This second scenario is critical for assessing the true generalization ability of the network. By validating on unseen airfoils and testing on completely new geometries, we can determine whether the model is capable of extrapolating to novel NACA configurations, a necessary feature for practical aerodynamic experimentation.

Since lift and moment coefficients have scale difference , we standardized all these three values. Mean and standard deviation are calculated for training $\mathbf{y}$ values, outputs. Then using the Equation 3.14, we standardize the outputs. After obtaining the predicted values, they converted back using an inverse transform method given in the Equation 3.15.

$$y_i' = \frac{y_i - \mu}{\sigma} \tag{3.14}$$

$$y = y_i'\sigma + \mu \tag{3.15}$$

### 3.2.4 Hyperparameter Tuning

Different neural network architectures applied to the same dataset can exhibit varying training times and convergence behaviors. Some architectures may achieve excellent convergence but require excessively long training times, while others may converge quickly but yield suboptimal results. To strike an optimal balance between these two criteria, hyperparameter optimization is essential. Depending on the size of the dataset, different hyperparameter optimization algorithms may be employed to efficiently search the design space.

For small hyperparameter spaces, the Grid Search method is widely used to identify the best set of hyperparameters. Grid Search systematically evaluates all possible combinations from the predefined hyperparameter ranges to find the optimal set. As an exhaustive tuning method, it examines every case one by one. The key advantage of Grid Search is that it guarantees finding the optimal combination within the defined search space. Additionally, it is straightforward to implement in any neural network workflow. However, its main drawback is computational inefficiency, since it evaluates every combination, it can become extremely slow for large and complex models with many parameters.

As the size of the dataset and the hyperparameter search space increased, traditional methods like Grid Search became inefficient due to their exponential computational cost. To address this challenge, a more scalable and efficient approach was necessary. One such method is the Tree-structured Parzen Estimator

(TPE), a type of Bayesian Optimization algorithm. Unlike classical Bayesian Optimization, which models the objective function $p(y \mid x)$, TPE models two conditional probability densities:

$$l(x) = p(x \mid y < y^*) \tag{3.16}$$

$$g(x) = p(x \mid y \geq y^*) \tag{3.17}$$

Here, $y^*$ is a threshold value, often set as a quintile of the observed objective values. Initially, the algorithm samples and evaluates a set of hyperparameter configurations to build a database of performance metrics. Based on these, TPE estimates two probability distributions: one representing promising configurations ($l(x)$) and the other representing less promising ones ($g(x)$).

The next hyperparameter configuration is chosen by maximizing the ratio:

$$\frac{l(x)}{g(x)} \tag{3.18}$$

This selection criterion focuses the search on regions in the hyperparameter space that are more likely to yield high-performing models. As new observations are made, the distributions ($l(x)$) and ($g(x)$) are updated, allowing the search process to adaptively refine its focus. This makes TPE significantly more sample-efficient than exhaustive methods like Grid Search, particularly in high-dimensional or complex spaces.

For our hyperparameters, we have chosen the number of neurons, the number of hidden layers, the type of activation function, and the learning rate. To obtain a more robust and structured network, we set the number of neurons in each hidden layer to be the same. Here are the hyperparameter sets we decided on:

- Number of hidden layer: [1 to 5]
- Number of neuron: [1 to 64]
- Learning rate: [0.00001 to 0.1]

- Activation functions: [ReLU, LeakyReLU, Tanh, Sigmoid]

Both the number of hidden layers and the number of neurons were chosen carefully to ensure that the total number of parameters does not exceed the number of data points. The selected activation functions are among the most commonly used in the field, each contributing unique characteristics to the network's behavior. These activation functions were selected following an extensive literature review. The details and rationale for each will be described in the following sub-sections.

Since our network operates as a non-linear function approximator, we selected Mean Squared Error (MSE) as the loss function to quantify prediction error.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(x_i - \hat{x_i})^2 \tag{3.19}$$

Here $x_i$ is the true solution while $\hat{x_i}$ is the predicted value.

#### 3.2.4.1 Sigmoid

Most neural networks use backpropagation as the primary learning algorithm. Backpropagation relies on the computation of derivatives to update model parameters through gradient descent. If the derivatives become constant across neurons or layers, the network's ability to learn effectively is hindered. To avoid this, non-linear activation functions are employed, as they provide varying derivatives at different input values, enabling the network to approximate complex non-linear mappings.

One of the simplest non-linear activation functions is the sigmoid function, which maps inputs to the (0,1) interval. It is commonly used in binary classification tasks and logistic regression models. The sigmoid function and its derivative are defined in Equations 3.20 and 3.21, respectively [26]. In aerodynamic modeling applications, sigmoid activation functions were successfully integrated into

neural network architectures for airfoil performance prediction[27].

$$f(x) = \frac{1}{1 + e^{-x}} \tag{3.20}$$

$$\frac{df(x)}{x} = \frac{e^{x}}{(1 + e^{x})^{2}} \tag{3.21}$$

#### 3.2.4.2 Hyperbolic Tangent

The hyperbolic tangent (Tanh) activation function is another widely used non-linear function, bearing a strong resemblance to the sigmoid function in shape and mathematical formulation. However, unlike the sigmoid function, which maps inputs to the range (0, 1), the Tanh function maps inputs to the range (–1, 1). This symmetric output around zero enables the network to better capture and propagate both positive and negative input values, often resulting in faster convergence during training.The derivative of the Tanh function is also similar in shape to that of the sigmoid, but with a broader dynamic range, making it more suitable in many contexts. The mathematical definitions of the tanh activation function and its derivative are provided in Equations 3.22 and 3.23. In aerodynamic modeling and airfoil prediction tasks, the hyperbolic tangent activation function has been successfully applied by several studies [28][29][27].

$$f(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \tag{3.22}$$

$$\frac{df(x)}{x} = 1 - f^{2}(x) \tag{3.23}$$

#### 3.2.4.3 ReLU

Both the sigmoid and hyperbolic tangent activation functions suffer from a major drawback known as the vanishing gradient problem. Because these functions squash their input values into a narrow bounded range (0 to 1 for sigmoid and –1 to 1 for Tanh), their gradients tend to become very small when the input is far from zero. During backpropagation, this results in gradients that diminish

progressively across layers, which preventing effective weight updates and leading to slow or stalled learning.

To address this issue, the Rectified Linear Unit (ReLU) activation function was introduced. ReLU performs a thresholding operation, returning zero for negative inputs and the input itself for positive values, as defined in Equation 3.24. Unlike sigmoid or Tanh, ReLU is not bounded above, and its gradient is constant (1) for positive inputs. This property significantly mitigates the vanishing gradient problem, especially in deep networks, allowing for more efficient training.

However, ReLU is not without limitations. Since its derivative is zero for all negative input values, neurons can become inactive, "die", during training if their inputs are consistently negative, a phenomenon referred to as the dying ReLU problem. Despite this, ReLU remains a fundamental activation function in deep learning and has been widely used in aerodynamic neural network architectures [30][29][27][31].

$$f(x) = 0 \qquad \text{for } x < 0 \tag{3.24}$$

$$f(x) = x \qquad \text{for } x \geq 0 \tag{3.25}$$

#### 3.2.4.4 Leaky ReLU

To overcome ReLU's zero-gradient issue in the negative domain, the Leaky ReLU activation function introduces a small non-zero slope for negative input values. This modification helps keep neurons active during training, preventing them from becoming permanently inactive. The negative slope is defined by multiplying the negative input by a small constant $a$, typically in the range of 0.01. The mathematical formulation of the Leaky ReLU activation function and its derivative is given in Equation 3.26. This function has been successfully implemented in various applications, including aerodynamic modeling [27].

$$f(x) = ax \qquad \text{for } x < 0 \tag{3.26}$$

$$f(x) = x \qquad \text{for } x \geq 0 \tag{3.27}$$

### 3.2.5 Support Vector Regression

Support Vector Machine (SVM) is a supervised learning algorithm used for both classification and regression problems. It was initially developed by Vladimir Vapnik in 1992 [32]. Unlike parametric methods, SVM is a non-parametric technique, as it relies on kernel functions rather than assuming a fixed model structure. SVM is particularly effective in high-dimensional spaces, where it operates by identifying the optimal hyperplane that separates data points into distinct classes. For regression tasks, the algorithm is extended into Support Vector Regression (SVR), which modifies the original SVM framework to handle continuous output variables more effectively.

The goal of Support Vector Regression (SVR) is to predict continuous output values, rather than discrete class labels. Given a dataset, SVR aims to approximate the underlying functional relationship between input features and their corresponding outputs. To accomplish this, SVR introduces the concept of an $\epsilon$-insensitive margin, which allows the model to tolerate small deviations from the actual target values. Unlike ordinary least squares regression, which minimizes the total error between predicted and true values, SVR seeks to fit a function such that the deviations from the actual values are within a predefined margin of tolerance $\epsilon$. In this approach, errors smaller than $\epsilon$ are not penalized, effectively creating a tube of width $2\epsilon$ around the predicted function. Only the data points lying outside this $\epsilon$-tube contribute to the loss function, encouraging the model to generalize while remaining robust to minor fluctuations or noise in the data.

To allow for flexibility in fitting real-world data, Support Vector Regression (SVR) introduces slack variables, denoted by $\xi_i$ and $\xi_i^*$, which account for data points that fall outside the $\epsilon$-insensitive margin. These variables quantify the extent to which predictions violate the $\epsilon$-tube on either side of the regression function. The SVR optimization problem then becomes a trade-off between two objectives: maintaining a flat regression function, achieved by minimizing the norm $||w||^2$, and penalizing deviations beyond the $\epsilon$ margin, as measured by the slack variables. This trade-off is controlled by the regularization parameter $C$,

which governs the balance between model complexity and training error. A larger $C$ places more emphasis on minimizing the slack variables, resulting in a tighter fit to the training data (but with an increased risk of overfitting), whereas a smaller $C$ promotes a simpler model that may generalize better. Mathematically, SVR seeks to minimize the objective function in Equation 3.28, subject to the constraints in Equation 3.29. This formulation ensures that the learned function remains both simple and accurate, without deviating excessively from the target values beyond the allowed $\epsilon$ margin.

$$J(\xi^*, \xi, w) = \frac{1}{2}||w||^2 + C\sum_{i=1}^{n}(\xi_i + \xi_i^*) \tag{3.28}$$

$$y_i - w^T x_i - b \leq \epsilon + \xi_i \tag{3.29}$$

$$w^T x_i + b - y_i \leq \epsilon + \xi_i^* \tag{3.30}$$

$$\xi_i \geq 0 \tag{3.31}$$

$$\xi_i^* \geq 0 \tag{3.32}$$

In practice, Support Vector Regression can effectively handle non-linear regression tasks through the use of kernel functions. Kernels implicitly map input features into a higher-dimensional feature space, enabling SVR to perform linear regression in that transformed space, which corresponds to a non-linear fit in the original input space. This approach allows SVR to model complex, non-linear relationships without explicitly computing the transformation. Most used kernels are:

1. Linear kernel: Used for data that can be seperatable linearly.
2. Polynomial kernel: Used for modeling data that have a curved relationships.
3. Radial Basis Function (RBF) kernel: Used for most real-world problems, also it is a very highly flexible kernel so it is suitable for different type of data.

Each kernel introduces a different notion of similarity and flexibility in fitting the training data. Among these, the RBF kernel is particularly popular due to its

ability to handle highly non-linear patterns while maintaining smoothness and generalization capability.

Once the model is trained, it utilizes a subset of the training data points known as support vectors, which are the samples that lie on or outside the $\epsilon$-margin and directly influence the shape of the regression function.

One of the primary advantages of Support Vector Regression is its effectiveness in high-dimensional spaces. Thanks to the use of kernel functions, SVR can efficiently handle problems where the number of features exceeds the number of data points, without explicitly mapping the data into high-dimensional spaces. This flexibility allows SVR to adapt to both simple and highly non-linear regression tasks. Another strength of SVR is its robustness to small amounts of noise. By employing an $\epsilon$-insensitive loss function, SVR disregards minor deviations within a defined margin and focuses only on significant errors. Moreover, SVR models are inherently sparse—only a subset of the training data points, known as support vectors, contribute to the final model. This sparsity enhances memory efficiency and results in simpler, more interpretable models. Finally, the regularization mechanisms integrated into SVR contribute to strong generalization performance, making the method well-suited for handling previously unseen data.

However, SVR also has notable limitations. One major limitation is its computational complexity. Training an SVR model involves solving a quadratic optimization problem, which becomes slow and memory intensive as the number of training samples increases. This makes SVR less suitable for very large datasets. Additionally since SVR relies on distance calculations, it is highly sensitive to the scaling of features, requiring careful preprocessing such as normalization or standardization to achieve reliable results. Finally, while linear SVR models are somewhat interpretable, non-linear SVR models, especially those using complex kernels, lack straightforward interpretability, making it difficult to understand how input features influence the output.

Ultimately, SVR provides a robust and flexible approach for modeling continuous outputs, especially in scenarios with high-dimensional input spaces or when the underlying relationship between features and targets is non-linear. Its ability to generalize well and focus only on the most critical data points makes it an strong machine learning model for various applications.

## 3.3 Results

We employed Support Vector Regression to assess whether the dataset is suitable for regression modeling and to determine whether the use of a more complex method, such as a neural network, is necessary. Initially, separate SVR models were constructed for NACA4 airfoils, NACA5 airfoils, and finally for the combined dataset containing both. For each case, we followed the same data split strategies outlined previously in the Section 3.2.3.

Since SVR is inherently a single-output regression method, individual models were trained separately for each aerodynamic coefficient (lift and moment). The performance of the SVR models was evaluated using two metrics: Root Mean Squared Error (RMSE) and Mean Absolute Percentage Error (MAPE), across all scenarios.

As previously mentioned, the dataset contains a total of 2,800 NACA 4 cases. For the Support Vector Regression analysis, we used 2,400 cases for training and the remaining 400 cases for testing in Scenario-1. In Scenario-2, we excluded four unique NACA 4 airfoil shapes from the training set and used them exclusively for testing. Consequently, the training set consisted of data from 24 different NACA 4 shapes, while the test set comprised previously unseen geometries.For the NACA 5 dataset, a total of 900 cases were available. Among these, 800 cases were used for training, and the remaining 100 cases formed the test set in Scenario-1. In Scenario-2, we reserved one unique NACA 5 airfoil shape for testing, ensuring it was not present in the training data. This allowed us to evaluate the generalization capability of the SVR model on unseen airfoil geometries.

Next, we combined the NACA4 and NACA5 geometries using the encoding format described in Section 3.2.3. For Scenario-1, we merged all available NACA geometries and randomly divided the combined dataset into training and test sets. For Scenario-2, we designated four distinct NACA4 geometries and one NACA5 geometry exclusively for the test set, ensuring these airfoil shapes were not included in the training data.These two scenarios are designed to evaluate the generalization capability of the SVR model. Specifically, Scenario-2 tests whether the model can accurately predict aerodynamic coefficients for airfoil geometries it has never encountered during training, which assessing the model's effectiveness in extrapolating to new cases.

To determine the optimal neural network parameters, we employed hyperparameter optimization, as detailed in Section 3.2.4. This optimization process took approximately 3 hours and 40 minutes to complete. Based on the results, the selected network architecture consists of one input layer, four hidden layers, and one output layer. Each hidden layer contains 54 neurons, with the following activation functions applied sequentially: Tanh, ReLU, Leaky ReLU, and Sigmoid. This architecture was also utilized for Scenario-2, ensuring consistency in performance evaluation across both scenarios.

Table 3.2: Test Errors for Scenario-1

| | | Lift | | Moment | |
|---|---|---|---|---|---|
| | | **MAPE** | **RMSE** | **MAPE** | **RMSE** |
| SVR | NACA 4 | 17.26 | $7.32 \times 10^{-2}$ | 16.82 | $2.63 \times 10^{-2}$ |
| | NACA 5 | 09.09 | $3.17 \times 10^{-2}$ | 09.81 | $8.03 \times 10^{-3}$ |
| | Mixed | 18.74 | $6.64 \times 10^{-2}$ | 27.35 | $2.11 \times 10^{-2}$ |
| Neural Network | | 07.36 | $9.04 \times 10^{-4}$ | 08.16 | $6.74 \times 10^{-5}$ |

## 3.4 Conclusion

In this chapter, a foundational framework for building a neural network model from scratch was developed, using NACA airfoil data as the primary case study. The NACA dataset was chosen deliberately, as it is a well-established and widely

Table 3.3: Test Errors for Scenario-2

| | | Lift | | Moment | |
|---|---|---|---|---|---|
| | | **MAPE** | **RMSE** | **MAPE** | **RMSE** |
| SVR | NACA 4 | 13.47 | $6.73 \times 10^{-2}$ | 19.63 | $2.03 \times 10^{-2}$ |
| | NACA 5 | 07.64 | $3.15 \times 10^{-2}$ | 06.73 | $8.16 \times 10^{-3}$ |
| | Mixed | 22.71 | $6.83 \times 10^{-2}$ | 28.41 | $2.45 \times 10^{-2}$ |
| Neural Network | | 06.32 | $1.52 \times 10^{-4}$ | 07.52 | $9.74 \times 10^{-6}$ |

researched domain in the field of machine learning, allowing for meaningful comparisons and a solid basis for learning. By working with this known dataset, the goal was not only to construct an accurate predictive model but also to deeply understand the process of model development from data preparation to performance evaluation.

Key insights gained throughout this chapter highlight the importance of data distribution and dataset size. An unbiased and diverse dataset leads to better generalization, while larger datasets support more accurate models and help mitigate overfitting. By encoding airfoil names directly, the need for complex preprocessing was minimized, enabling faster iterations and facilitating rapid design evaluations for industrial applications. However, this convenience reinforces the need for a well structured input format and thoughtful model complexity management, as increasing the number of parameters without adequate data can affect the model's reliability.

Moreover, this study emphasized that hyperparameter selection and activation functions must be chosen carefully, often requiring different optimization strategies depending on dataset characteristics. These considerations become especially critical in more advanced physical scenarios, such as turbulence or biomedical flows, where the underlying physics is more complex and harder to model. Notably, the neural network consistently outperformed Support Vector Regression (SVR), especially in generalizing to unseen airfoil geometries. The neural model demonstrated robustness in capturing underlying aerodynamic patterns and provided accurate predictions even for cases not present in the training data, showcasing its potential for extrapolating to novel configurations.

This chapter serves as a learning phase, establishing a methodology in a familiar domain that will now be extended to a less explored but more application topic: stenosed blood vessels. Unlike the NACA airfoil case, the use of machine learning in stenosis modeling is not yet widespread, presenting both a challenge and an opportunity. The experience and techniques developed here will guide the next phase, where the goal is to build reliable models for biological flow prediction and ultimately support early detection of life threatening diseases such as stroke.

# Chapter 4

# Machine Learning Assisted Blood Vessel Analysis

## 4.1 Motivation

After learning how to interpret data and develop machine learning models through the NACA airfoil case, it is appropriate to return to the initial focus: stenosed vessels. Although we previously modeled various geometric configurations using CFD, it is not feasible to simulate every possible case due to time and computational constraints. To overcome this limitation, a machine learning-based approach is required to predict outcomes for previously unseen, non-simulated cases. Therefore, in this section, we explore additional machine learning methods alongside neural networks to assess their predictive capabilities for stenosis-related hemodynamic parameters.

Creating such a model holds significant potential in the healthcare field. Early detection of critical vessel conditions is essential for preventing strokes and other cardiovascular complications. Employing a predictive model provides a straightforward and efficient approach to assist in diagnosis. However, to develop an accurate and reliable predictive model, it is crucial to have access to a sufficiently

large and representative dataset of historical cases. The quality and diversity of the training data directly influence the model's performance. If the data is inadequate or unrepresentative, the predictions will deviate significantly from actual outcomes.

Since there are no previous studies directly addressing this specific topic, no existing datasets were available for use. Therefore, the dataset was generated from scratch. During this process, careful attention was paid to avoid introducing bias that could potentially degrade the performance of the predictive models.

In the following sections, the geometry of the vessel will first be examined, followed by details of the numerical simulations used to generate the data. Subsequently, the structure of the dataset and the design of the neural network models will be described. Finally, additional machine learning models will be introduced, and a comparative evaluation of all models will be presented.

## 4.2 Methodology

### 4.2.1 Vessel Geometry

Based on insights from the previous vessel analysis, the stenosis geometry was modified to be more suitable for parametric CFD analysis. Given the absence of prior studies applying machine learning to blood vessel hemodynamics, it was necessary to generate the dataset from scratch. In addition to the original vessel geometry presented in Chapter 2, a key modification was made: instead of having stenosis affect only a portion of the vessel diameter, the stenosis in this study fully occludes the diameter, as illustrated in Figure 4.1. Furthermore, the super-ellipsoid shape parameter $n$ was excluded from the parameter space. As a result, only elliptical stenosis geometries were considered in this analysis.

The geometrical parameters investigated in this study include the length of the stenosis ($L$), the degree of narrowing caused by the stenosis ($N$), and the distance from the center of the stenosis to the theoretical healthy vessel wall ($b$). The range of values of parameters are given in the below:

- Narrowing: 1.25 mm to 2.25 mm, increasing with 0.25 mm in each step.
- Length: 10 mm to 20 mm, increasing with 1 mm in each step.
- Center Distance: 0.2 mm to 3.8 mm, increasing with 0.2 mm in each step.

Similar to the methodology used in the previous analysis, the expected Reynolds number was calculated to estimate the required entrance length for fully developed flow. Using a hypothetical Reynolds number of 200, significantly higher than the expected physiological values, the maximum theoretical entrance length was determined to be approximately 64 mm. To ensure sufficient development of the flow before encountering the stenosed region, a 130 mm inlet length was adopted. The same length was mirrored on the outlet side of the vessel to maintain flow symmetry and minimize outlet boundary effects. Through a parametric study conducted in ANSYS Workbench, a total of 1030 distinct geometries

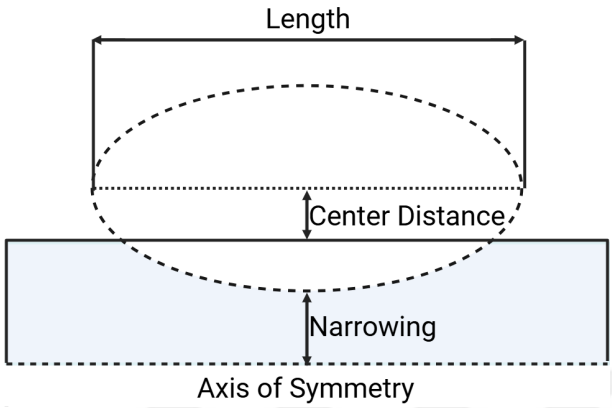


Figure 4.1: Stenosis Geometry

were generated based on various combinations of the selected stenosis parameters.

### 4.2.2 Numerical Analysis

In this analysis, a fixed inlet velocity of 0.26 m/s was used for all simulations, unlike the previous study that included two different velocity values. All other numerical settings remained consistent with the prior configurations.

Due to the slight modifications in vessel geometry compared to the earlier study, a new mesh independence analysis was required. The vessel geometry with the most severe narrowing, 1.25 mm narrowing, 0.2 mm offset distance, and 10 mm stenosis length, was selected for this analysis, as it demands the finest mesh resolution among all tested geometries. Ensuring mesh adequacy for this critical case guarantees that all other geometries with less severe constrictions are also well-resolved.

Similar to the procedure followed in Chapter 2, a series of mesh configurations were tested to find the optimal balance between accuracy and computational efficiency. The results of this analysis are presented in Table 4.1, and Mesh-8 was selected as the final mesh due to its satisfactory convergence characteristics and reasonable computational cost.

Table 4.1: Mesh Independence Analysis

| Mesh Number | Body Size | Sweep Bias | Inflation Layer Grow Rate | Shear [Pa] | Time [min] |
|---|---|---|---|---|---|
| Mesh-1 | 1 | 3 | 1.5 | 168.07 | 3 |
| Mesh-2 | 0.5 | 3 | 1.5 | 176.21 | 8 |
| Mesh-3 | 0.25 | 3 | 1.5 | 182.15 | 11 |
| Mesh-4 | 0.125 | 3 | 1.5 | 184.36 | 18 |
| Mesh-5 | 0.25 | 5 | 1.5 | 183.91 | 18 |
| Mesh-6 | 0.25 | 7 | 1.5 | 184.78 | 20 |
| Mesh-7 | 0.25 | 5 | 1.7 | 182.62 | 15 |
| Mesh-8 | 0.25 | 5 | 1.2 | 186.12 | 20 |
| Mesh-9 | 0.25 | 5 | 1.1 | 186.32 | 26 |

### 4.2.3 Neural Network

#### 4.2.3.1 Input Format

For each case, three input parameters and two output parameters were defined. The input parameters are the stenosis length $L$, the distance of the stenosis center from the vessel wall $b$, and the narrowing of the vessel $N$. These are summarized in Equation 4.1. The output parameters are the maximum velocity and the maximum wall shear stress measured at the narrowest section of the vessel. These are represented in Equation 4.2.

A total of 1030 different geometry combinations were generated. From this full dataset, two smaller subsets were randomly selected to evaluate model performance at varying data scales. Specifically, 450 and 750 geometry combinations were randomly chosen to form two additional datasets. As a result, three datasets, containing 450, 750, and 1030 samples respectively, were constructed. Each of these datasets was used to assess the learning capabilities and generalization performance of the machine learning models applied to this specific problem.

$$\mathbf{X} = [\mathbf{L} \quad \mathbf{N} \quad \mathbf{b}] \tag{4.1}$$

$$\mathbf{y} = [\tau \quad \mathbf{V}] \tag{4.2}$$

Then each data set is divided into sub sets, training set, validation set and test set. The training set contains 70% of the data, is used to train the network.

Then 20% of the data is used as validation set in the training process to improve the learning capacity of the model. Then the least 10% data is used as a test set which used to evaluate the performance of the model.

#### 4.2.3.2 Hyperparameter Optimization

For hyperparameter optimization, a similar approach was adopted as in the previous NACA airfoil study. The same optimization algorithm was employed; however, certain hyperparameters were adjusted to reflect the differences in the current problem. Specifically, due to the reduced number of input features in this study, the maximum number of neurons per layer was decreased from 64 to 32 in order to minimize the risk of overfitting. Other parameters, such as the set of activation functions, learning rate range, and optimization method, were kept consistent with the previous setup.

Optimization algorithm ran three times with different data sets, one for each batch. For each batch sample size kept at 200 for consistency. For the evaluation criteria, mean percentage error is calculated for both shear and velocity values. Then their average is taken into account as the total error. Optimization result are observed with different number of epoch, then it decided to kept at 1000 since for all the cases the training loss did not change dramatically. It was safe to say that loss was converging around 1000th epoch. The results of different epoch sizes are given in the Table 4.2.

The hyperparameter optimization algorithm was executed separately for each of the three data sets, resulting in a total of three independent optimization runs. To maintain consistency across these runs, the batch size for hyperparameter evaluation was fixed at 200 in all cases. During training, the mean squared error (MSE) was used as the loss function, calculated individually for the maximum wall shear stress and maximum velocity. The average of these two values was minimized. For model evaluation, the mean absolute percentage error (MAPE) was computed for both output parameters. The final performance of each model was assessed by taking the average of the two MAPE values, which was considered

the overall prediction error.

To determine the appropriate number of training epochs, the optimization process was initially conducted with varying epoch counts. The results indicated that the training loss exhibited negligible improvement beyond 1000 epochs, suggesting that convergence had been effectively reached. As a result, 1000 epochs was selected as the standard training duration for all models. A summary of the training loss behavior across different epoch sizes is presented in Table 4.2.

Table 4.2: Epoch Analysis

| Number of Epoch | Training Time (in minutes) | Validation Loss |
|---|---|---|
| 100 | 1.04 | 0.26 |
| 1000 | 5.3 | 0.03 |
| 10000 | 44.2 | 0.03 |

### 4.2.4 Other Machine Learning Models

In addition to Neural Networks, various alternative machine learning models, both linear and nonlinear, were implemented to evaluate their effectiveness in predicting hemodynamic parameters from the given dataset. Furthermore, ensemble methods were applied to assess the predictive capability of more complex architectures. Each model has distinct strengths and limitations, and their performance is expected to vary depending on both the target variable and the size of the dataset.

For the linear models, Ridge Regression and Lasso Regression were selected due to their ability to manage multicollinearity and perform feature selection. Among nonlinear models, the following were investigated: Support Vector Machine (SVM) with a Radial Basis Function (RBF) kernel, K-Nearest Neighbors (KNN), and Decision Trees. Additionally, two ensemble learning techniques, Random Forest and Extreme Gradient Boosting (XGBoost), were employed. Ensemble methods combine multiple learners to enhance predictive performance and robustness. As Support Vector Machine was already discussed in detail in Chapter 3, it will not be reintroduced here. Numerical examples and implementation

details for each model are provided in Appendix C.

#### 4.2.4.1 Ridge

Ridge Regression, also known as $L_2$ regularization, is one of the most widely used linear regression techniques, particularly effective in scenarios where the data exhibits multicollinearity. Multicollinearity occurs when two or more independent variables are highly linearly correlated, leading to instability in the estimation of regression coefficients. For example, to estimate fuel consumption, $y$, two input variables might be total time spent on the road, $x_1$ and total distance traveled $x_2$. In this case, the two variables are inherently related, greater distances usually require more time, which making them collinear. In such situations, the model struggles to distinguish the individual effect of each variable, and the regression coefficients become unstable, fluctuating significantly with even minor changes in the data.

The goal of any regression model is to minimize the loss function, typically the difference between the actual value $y^i$ and the predicted value $\hat{y}^i$. In classical regression approaches, the loss function is defined as:

$$J(\beta) = \sum_{i=1}^{n} (y^i - \hat{y}^i)^2 = \sum_{i=1}^{n} \left( y^i - \beta_0 - \sum_{j=1}^{p} \beta_j x_j^i \right)^2 \tag{4.3}$$

However, in the presence of multicollinearity, coefficients such as $\beta_1$ and $\beta_2$ may take on disproportionately large values, making the model overly sensitive and prone to overfitting. To address this, Ridge Regression introduces a regularization term that penalizes large coefficients. The modified loss function becomes:

$$J(\beta) = \sum_{i=1}^{n} \left( y^i - \beta_0 - \sum_{j=1}^{p} \beta_j x_j^i \right)^2 + \lambda \sum_{j=1}^{p} \beta_j^2 \tag{4.4}$$

Here, $\lambda$ is the regularization parameter, which controls the strength of the penalty. A higher $\lambda$ shrinks the coefficients more aggressively, reducing variance but potentially increasing bias.

#### 4.2.4.2 Lasso

Lasso Regression is a regression model similar to Ridge Regression, and it is also used when the dataset suffers from multicollinearity. However, Lasso offers an additional benefit: it can perform feature selection by eliminating irrelevant or redundant variables. This is particularly useful when the dataset contains many noisy or hard to interpret features.

Unlike Ridge Regression, which uses the $L_2$ norm, Lasso Regression uses the $L_1$ norm as the regularization term. This difference leads to a crucial behavior: Lasso tends to shrink some coefficients exactly to zero, which leads to removing those features entirely from the model.

The loss function for Lasso Regression is formulated as follows:

$$J(\beta) = \sum_{i=1}^{n} \left( y^i - \beta_0 - \sum_{j=1}^{p} \beta_j x_j^i \right)^2 + \lambda \sum_{j=1}^{p} |\beta_j| \tag{4.5}$$

Here, $\lambda$ is the regularization strength. Higher values of $\lambda$ lead to more aggressive feature elimination, effectively reducing model complexity and potentially improving generalization.

#### 4.2.4.3 K-Nearest Neighbor

K-Nearest Neighbors (KNN) is fundamentally different from regression models like Ridge and Lasso. In fact, KNN is not a regression model in the traditional sense—it is a non-parametric, instance-based learning algorithm.

Instance-based learning algorithms do not explicitly learn a model during training. Instead, they store the training data and make predictions based on comparisons between new inputs and the stored data. In this way, KNN predicts outcomes directly from the data rather than building a mathematical model.

When a prediction is required for a new input, KNN computes the distance

between the new input and all the points in the training set. Several distance metrics can be used for this comparison, with the most common being:

$$\begin{aligned} \text{Euclidean distance, } d(x,y) &= \sqrt{\sum_{i=1}^{n}(y_i - x_i)^2} \\ \text{Manhattan distance, } d(x,y) &= \sum_{i=1}^{n}|x_i - y_i| \end{aligned} \tag{4.6}$$

After computing these distances, the algorithm selects an integer $K$ and identifies the $K$ nearest neighbors. The output for the new input is then predicted by averaging the outputs of these $K$ neighbors. A small $K$ leads to low bias but may result in overfitting due to high variance while a large $K$ smooths the prediction but may underfit the data.

#### 4.2.4.4 Decision Tree

Decision Tree models are divided into two categories: classification and regression. The focus here is on the regression variant. This model predicts continuous numerical values, unlike the classification type, by applying simple decision rules derived from the data. The algorithm recursively splits the input space with the objective of reducing the overall prediction error. For each potential split, it partitions the data into left and right subsets, then computes the desired error metric for each output. The cost of each potential split is calculated using Equation 4.7:

$$\text{Split Cost} = \frac{n_L}{n} \cdot \text{Error}_L + \frac{n_R}{n} \cdot \text{Error}_R \tag{4.7}$$

After evaluating all possible splits, the one with the lowest cost is selected. This process continues recursively until the specified tree depth is reached.

#### 4.2.4.5 Random Forest

Random Forest is an ensemble learning method that combines multiple decision trees into a single predictive model. While it shares the foundational principles of decision trees, it introduces randomness to enhance generalization and reduce overfitting. The algorithm generates $N$ random samples from the original dataset (with replacement), each of size $N$. A separate decision tree is trained on each sample. At each split within a decision tree, a random subset of input variables is considered, rather than evaluating all features. This strategy increases diversity among trees and mitigates overfitting. For regression tasks, the final prediction is obtained by averaging the outputs of all individual trees, as expressed in Equation 4.8, where $f(x)_i$ denotes the prediction from the $i^{th}$ decision tree.

$$y = \frac{1}{N} \sum_{i=1}^{N} f(x)_i \tag{4.8}$$

#### 4.2.4.6 Extreme Gradient Boosting

Extreme Gradient Boosting (XGBoost) is an ensemble learning algorithm that, like Random Forest, aggregates multiple decision trees. However, unlike Random Forest where trees are trained independently in parallel, XGBoost builds trees sequentially. Each new tree is added with the objective of correcting the errors made by the ensemble of previously trained trees. This sequential learning framework is what distinguishes XGBoost as a boosting algorithm.

Let $l(y, \hat{y})$ represent the loss function. The final prediction of the model is expressed as the summation of predictions from all decision trees, as shown in Equation 4.9, where $f_k(x_i)$ denotes the output of the $k^{th}$ decision tree and $T$ is the total number of trees.

$$\hat{y} = \sum_{k=1}^{T} f_k(x_i) \tag{4.9}$$

At each iteration $k$ , the algorithm aims to minimize an overall objective function, which combines the training loss and a regularization term to prevent

overfitting, as shown in Equation 4.10.

$$L^k = \sum_{i=1}^{n} l(y_i, \hat{y}_i^{(k-1)} + f_k(x_i)) + \Omega(f_k) \tag{4.10}$$

The regularization term $\Omega(f_k)$, defined in Equation 4.11, penalizes model complexity by considering the number of leaf nodes $N$ and the squared leaf weights $w_j$. Here, $\gamma$ and $\lambda$ are regularization coefficients

$$\Omega(f_k) = \gamma N + \frac{1}{2}\lambda \sum_{j=1}^{N} w_j^2 \tag{4.11}$$

## 4.3 Results

For each dataset size described in Section 4.2.3.1, a separate hyperparameter optimization procedure was conducted. Both wall shear stress and velocity losses were calculated during the training process. Subsequently, the optimized models were employed to predict the outcomes on the respective test sets. The evaluation of the other machine learning models described in Section 4.2.4 was conducted using the three dataset sizes introduced in Section 4.2.3.1. For consistency, all models were trained and tested using the same number of samples. Validation sets were not employed for these models, as their hyperparameters were effectively optimized using cross-validation techniques. Instead of reserving a separate portion of the data for validation, methods such as GridSearchCV and RandomizedSearchCV internally partition the training data into multiple folds, assessing performance across these subsets. Importantly, the validation data was not incorporated into the training set, in order to ensure that all models were trained on an equivalent amount of data. Detailed model configurations for the neural networks, along with the prediction results of all machine learning models, are presented in Tables 4.3, 4.4, and 4.5, respectively.

As expected, an increase in the size of the training dataset resulted in a decrease in both training loss and prediction error. In the neural network model, using 721 training samples, a prediction error of less than 1% was achieved for both maximum wall shear stress and velocity values.

Table 4.3: Models' Hyper Parameters

| Model Name | Data Size | Learning Rate | Neuron Number |
|---|---|---|---|
| Model-1 | 450 | $1.04 \times 10^{-2}$ | 30 |
| Model-2 | 750 | $6.98 \times 10^{-3}$ | 30 |
| Model-3 | 1030 | $8.77 \times 10^{-3}$ | 31 |

Table 4.4: Models' Activation Functions

| Model Name | Activation Functions | | | | |
|---|---|---|---|---|---|
| Model-1 | Tanh | Leaky ReLU | - | - | - |
| Model-2 | ReLU | Tanh | Leaky ReLU | Tanh | |
| Model-3 | Tanh | Tanh | Tanh | Sigmoid | Sigmoid |

Similar to the neural network models, all machine learning methods demonstrated improved performance with increased training data, as evidenced by the reduction in error for both shear stress and velocity predictions. However, the magnitude of improvement varied significantly across models, indicating that the sensitivity to data volume is highly model-dependent.

Among all evaluated models, Ridge and Lasso regression exhibited the poorest performance. This outcome aligns with expectations, as these models are best suited for datasets exhibiting linear relationships and multicollinearity among features. In contrast, the dataset used in this study is highly non-linear. As previously discussed, Ridge and Lasso assume a linear mapping between inputs and outputs, as expressed in Equation 4.12. However, the target variables—maximum shear stress and maximum velocity—are governed by complex non-linear interactions with the geometric parameters, as suggested by the relationship in Equation 4.13. Moreover, these models tend to over-penalize coefficients through regularization, which may result in the suppression of features that are actually crucial for accurate predictions.

$$\hat{y} = \beta_0 + \beta_1 x_1 + \beta_2 x_2 \tag{4.12}$$

$$\text{Shear} \propto \frac{dV}{dy} \tag{4.13}$$

K-Nearest Neighbor demonstrated the third weakest performance across all

Table 4.5: Results for Data Set with Size of 450

| Model Name | Shear Error | Velocity Error |
|---|---|---|
| Ridge | 37.0% | 11.2% |
| Lasso | 107.5% | 46.8% |
| K-Nearest | 23.67% | 12.3% |
| SVR | 21.1% | 08.3% |
| Decision Tree | 02.3% | 0.3% |
| Random Forest | 02.0% | 0.3% |
| XGBoost | 01.8% | 0.2% |
| Neural Network | 03.5% | 01.0% |

Table 4.6: Results for Data Set with Size of 750

| Model Name | Shear Error | Velocity Error |
|---|---|---|
| Ridge | 30.0% | 11.0% |
| Lasso | 80.1% | 37.4% |
| K-Nearest | 18.9% | 10.5% |
| SVR | 19.2% | 08.3% |
| Decision Tree | 02.4% | 0.3% |
| Random Forest | 01.6% | 0.2% |
| XGBoost | 01.4% | 0.2% |
| Neural Network | 01.6% | 0.7% |

three dataset sizes. Although KNN is a non-parametric method capable of capturing non-linear relationships, it struggles in scenarios where the output is highly sensitive to input variations, as in this study. Moreover, in high-dimensional feature spaces or when input features exhibit strong correlations, the effectiveness of distance-based metrics significantly decrease, leading to reduced generalization capability. These challenges were especially evident in smaller training sets. While techniques such as feature scaling, dimensionality reduction, and distance-based weighting can mitigate some of these issues, their implementation demands considerable additional effort and may not fully resolve the limitations.

Support Vector Machine Regression provides a robust kernel-based framework for modeling nonlinear relationships between geometrical parameters and fluid dynamics outputs. When appropriately scaled and tuned, SVR can effectively capture smooth, continuous variations in fluid behavior resulting from geometric changes. However, its performance tends to deteriorate in the presence of sharp gradients, noise, or abrupt transitions, such as those observed in stenotic regions,

Table 4.7: Results for Data Set with Size of 1300

| Model Name | Shear Error | Velocity Error |
|---|---|---|
| Ridge | 28.9% | 11.2% |
| Lasso | 78.4% | 37.4% |
| K-Nearest | 17.5% | 09.2% |
| SVR | 18.4% | 07.7% |
| Decision Tree | 01.9% | 0.2% |
| Random Forest | 01.2% | 0.2% |
| XGBoost | 01.3% | 0.2% |
| Neural Network | 0.9% | 0.5% |

particularly when predicting wall shear stress. This limitation is evident in the model's comparative results. SVR performed significantly better in predicting velocity than shear stress. Additionally, SVR requires careful hyperparameter tuning, like neural networks, to achieve optimal performance. In this study, SVR achieved moderate accuracy, with observable improvements as the training data size increased. Nevertheless, it lagged behind ensemble methods and neural networks in terms of flexibility and robustness for modeling complex fluid–structure interactions.

Since decision trees are highly flexible, rule-based models, they are well-suited for capturing nonlinear relationships and can perform effectively in scenarios involving abrupt changes, such as sharp stenoses. Their ability to partition the input space makes them particularly capable of modeling sudden transitions in the data. However, due to their piecewise constant nature, decision trees often struggle with smooth and highly sensitive outputs. Without appropriate regularization, such as limiting tree depth or employing pruning techniques, single trees may yield unstable predictions and fail to accurately represent continuous variations, particularly for outputs like shear stress that respond gradually to geometric changes.

In the case of Random Forest, the aggregation of predictions from multiple decision trees where each capable of capturing nonlinear patterns, allowed the model to effectively represent complex interactions in the dataset while reducing overfitting. Its inherent robustness to irrelevant features, insensitivity to feature

scaling, and ability to model both abrupt transitions and smooth variations made it particularly well-suited for this problem. As expected, the model's performance improved with larger training sets, evidenced by decreasing error metrics. However, like all tree-based models, Random Forest lacks extrapolation capabilities and offers limited interpretability. This restricts its applicability in extrapolative scenarios or real-time diagnostic tasks unless supplemented with visualization or explainability tools. Therefore, the predicted values are reliable only within the range of the training data. Overall, Random Forest emerges as a dependable and accurate model for this application.

Since XGBoost constructs each decision tree to minimize the residual errors of the previous ones, it is expected to outperform standalone Decision Tree and Random Forest methods. As anticipated, XGBoost delivered the best overall performance among the traditional machine learning models, particularly in smaller data sets, where it rivaled neural networks in accuracy. However, like other tree-based models, XGBoost struggles with extrapolation and cannot reliably predict outputs for geometries outside the training data distribution. While it excels in capturing complex patterns with limited data, its performance converges as the data size grows. In contrast, neural networks continue to improve with larger datasets due to their higher representational capacity. A comprehensive comparison of all models is provided in Table 4.5.

## 4.4 Conclusion

In conclusion, this chapter sets the foundation for building a reliable predictive model for stenosed vessel cases using machine learning. The careful construction of a new, unbiased dataset was crucial to ensure that the models developed would be accurate and generalizable. Following a detailed inspection of the vessel geometries and the numerical methods used, both neural networks and alternative machine learning models were applied and compared. From the results, the importance of dataset size became evident—larger datasets led to significant improvements in predictive performance, especially for neural networks. Among

the tested models, neural networks showed a greater sensitivity to dataset size, indicating that their accuracy improves more noticeably with additional data compared to other methods. Therefore, the usage of real patient data in future work could further enhance prediction accuracy and model robustness. The insights gained from this study aim not only to advance predictive modeling in biomedical applications but also to contribute toward early detection of critical health conditions such as strokes, ultimately supporting better clinical decision-making.

# Chapter 5

# Conclusion and Future Remarks

## 5.1 Conclusion

In this thesis, three research projects were conducted to explore data-driven and physics-based modeling approaches in fluid dynamics. The first part focused on traditional CFD simulations of blood flow through stenosed vessels. By parameterizing the stenosis geometry and analyzing key hemodynamic variables such as wall shear stress, velocity profiles, and pressure drops, critical insights were obtained into how geometric factors influence disease progression. This method provided high-fidelity ground data and established a foundation for understanding vascular behavior under varying physiological conditions.

In the second part, a novel machine learning input was developed for predicting aerodynamic coefficients of NACA 4- and 5-digit airfoils. Unlike conventional methods that rely on geometric reconstruction, this study proposed a compact encoding of airfoil designations and flight conditions as direct inputs to neural networks and regression models. The resulting models demonstrated strong predictive performance with reduced computational requirements, opening a pathway for rapid aerodynamic assessments in early-stage design or prototyping tasks.

Using the knowledge gained from both prior studies, the third and final part of the thesis translated the concept of data-driven modeling to biomedical flows. Here, machine learning models were trained on CFD derived datasets of stenosed vessels to predict hemodynamic metrics directly from input parameters derived from stenosis geometry properties. The results showed that properly tuned models, especially neural networks and ensemble methods, can serve as fast and accurate surrogates for computationally expensive simulations. This approach significantly improves the feasibility of real-time clinical evaluations or large-scale parametric analyses in biomedical research.

Overall, this thesis demonstrated how traditional simulation techniques and modern machine learning can complement each other in both engineering and biomedical domains. The vessel simulations provided deep physical insights, the airfoil models showcased the power of abstraction and pattern learning, and the final vessel machine learning models illustrated how machine learning can accelerate modeling without compromising reliability. Together, these contributions form a flexible and scalable modeling framework applicable to a wide range of fluid dynamics problems.

## 5.2 Future Remarks

Both machine learning frameworks suffered from different limitations due to time and computational limitations. In NACA case more data from different airfoil geometry and flight condition combination can be generated to eliminate bias and form more balanced drag distribution. With that, not only drag coefficients be able to predicted, also availability of a third aerodynamic coefficient would help network to make better correlations between output parameters which leads to better accuracy. Also creating a new model, which would be Physic Informed Neural Network (PINN) could increase the accuracy when trained with data.

Similarly for the Stenosis, a larger database would significantly increase the accuracy as we see the increasing trend with the available data. Also since it was

aimed to use in practical applications, like clinical usage, navigating the model with a graphical user interface (GUI) would be beneficial. So an app should be created that would contain trained model. Then when input set, only geometrical parameters, giving to the system it would give the flow parameters as result within the seconds.

Also since the geometry inputs are normalized, the model can be used in any vessel geometry in geometry vise. But in large vessels, like aorta, the steady flow assumption do not hold. The flow must be taken as a pulsatile flow to mimic heartbeat. But integrating the pulsatile flow as an input to the model is not straightforward since the geometric parameters are static (constant) parameters while pulsatile flow is dynamic (periodic) input parameter. Regular dense layers cannot handle dynamic inputs. The features of the dynamic inputs must be extracted using various techniques. Recurrent networks or convolution layers can help in this case depending on the dynamic input's properties. Since pulsatile flow can be interpreted as a periodic flow, it has repeating patterns, peaks and falls. The features does not have long sequences and complex in time domain. That is why convolution layers can be a good choice to extract velocity features. Convolution layers put small filters over the input sequence, where it is periodic velocity, to detect local patterns (like peaks, slopes, or oscillations) and turn them into meaningful features. Then the extracted features will be combined in a *concatenate layer* with the static inputs to crate more complex input type. Then concatenate layer can undergo traditional neural network operations mentioned in the previous chapters.

# Bibliography


[1] W. H. Organization, "Cardiovascular diseases (cvds) — key facts," *World Health Organization Fact Sheet*, 2023.

[2] A. M. Malek, S. L. Alper, and S. Izumo, "Hemodynamic shear stress and its role in atherosclerosis," *JAMA*, vol. 282, no. 21, pp. 2035–2042, 1999.

[3] S. A. Berger and L. D. Jou, "Flows in stenotic vessels," *Annual Review of Fluid Mechanics*, vol. 32, pp. 347–382, 2000.

[4] Y. S. Chatzizisis, A. U. Coskun, M. H. Jonas, and et al., "Role of endothelial shear stress in the natural history of coronary atherosclerosis and vascular remodeling: molecular, cellular, and vascular behavior," *Circulation*, vol. 117, no. 8, pp. 1092–1103, 2008.

[5] G. W. Stone and et al., "Prediction of progression of coronary artery disease and clinical outcomes using vascular profiling of endothelial shear stress and arterial plaque characteristics," *JACC: Cardiovascular Imaging*, vol. 5, no. 12, pp. 1221–1230, 2012.

[6] M. Ojha, "Wall shear stress temporal gradient and its role in atherosclerosis and arterial remodeling," *Journal of Biomechanics*, vol. 26, no. 12, pp. 1179–1189, 1993.

[7] J. Soulis, E. Lampri, D. Papaioannou, and N. Giannoglou, "Wall shear stress in normal left coronary artery tree: influence of vessel geometry and flow conditions," *Computer Methods in Biomechanics and Biomedical Engineering*, vol. 9, no. 6, pp. 331–344, 2006.

[8] R. Mittal and J.-H. Seo, "Computational modeling of cardiac hemodynamics: current status and future outlook," *Journal of Computational Physics*, vol. 305, pp. 1065–1082, 2016.

[9] C. A. Taylor and D. A. Steinman, "Image-based modeling of blood flow and vessel wall dynamics: applications, methods and future directions," *Annals of Biomedical Engineering*, vol. 38, pp. 1188–1203, 2010.

[10] I. H. Abbott and A. E. von Doenhoff, *Theory of Wing Sections*. Dover Publications, 1959.

[11] D. A. Rubenstein, W. Yin, and M. D. Frame, *Biofluid Mechanics (Third Edition)*. Academic Press, 2015.

[12] J. Krystian, M. Łukasz, and O. Wojciech, "Model of blood rheology including hemolysis based on population balance," *Communications in Nonlinear Science and Numerical Simulation*, vol. 116, p. 106802, 2023.

[13] S. Pabi, M. K. Khan, S. K. Jain, A. K. Sen, and A. Raj, "Effect of stenotic shapes and arterial wall elasticity on the hemodynamics," *Physics of Fluids*, vol. 35, p. 101908, 10 2023.

[14] K. Jedrzejczak, Makowski, W. Orciuch, K. Wojtas, and M. Kozłowski, "Hemolysis of red blood cells in blood vessels modeled via computational fluid dynamics," *International Journal for Numerical Methods in Biomedical Engineering*, vol. 39, no. 11, p. e3699, 2023.

[15] E. Nader, S. Skinner, M. Romana, R. Fort, N. Lemonne, N. Guillot, A. Gauthier, S. Antoine-Jonville, C. Renoux, M.-D. Hardy-Dessources, E. Stauffer, P. Joly, Y. Bertrand, and P. Connes, "Blood rheology: Key parameters, impact on blood flow, role in sickle cell disease and effects of exercise," *Frontiers in Physiology*, vol. 10, 2019.

[16] P. M. Khan, A. Raj, M. I. Alam, S. Chakraborty, and S. Roy, "Prediction of vortex structures in pulsatile flow through s-bend arterial geometry with different stenosis levels," *Biocybernetics and Biomedical Engineering*, vol. 43, no. 1, pp. 298–312, 2023.

[17] M. Giersiepen, L. J. Wurzinger, R. Opitz, and H. Reul, "Estimation of shear stress related blood damage in heart valve prostheses—in vitro comparison of 25 aortic valves," *The International Journal of Artificial Organs*, vol. 13, no. 5, pp. 300–306, 1990.

[18] G. Heuser and R. Opitz, "A couette viscometer for short time shearing of blood," *Biorheology*, vol. 17, no. 1-2, pp. 17–24, 1980.

[19] G. Heuser and R. Opitz, "A couette viscometer for short time shearing of blood," *Biorheology*, vol. 17, no. 1-2, pp. 17–24, 1980.

[20] T. Zhang, M. E. Taskin, H.-B. Fang, O. M. Turan, X. Huang, Z. J. Wu, and B. P. Griffith, "Study of flow-induced hemolysis using novel couette-type blood-shearing devices," *Artificial Organs*, vol. 35, no. 12, pp. 1180–1186, 2011.

[21] Q. Pan, W. Feng, R. Wang, A. Tabuchi, P. Li, B. Nitzsche, L. Fang, W. M. Kuebler, A. R. Pries, and G. Ning, "Pulsatility damping in the microcirculation: Basic pattern and modulating factors," *Microvascular Research*, vol. 139, p. 104259, 2022.

[22] F.-B. Tian, L. Zhu, P.-W. Fok, and X.-Y. Lu, "Simulation of a pulsatile non-newtonian flow past a stenosed 2d artery with atherosclerosis," *Computers in Biology and Medicine*, vol. 43, no. 9, pp. 1098–1113, 2013.

[23] F. R. Menter, "Two-equation eddy-viscosity turbulence models for engineering applications," *AIAA journal*, vol. 32, no. 8, pp. 1598–1605, 1994.

[24] F. R. Menter, "Ten years of industrial experience with the sst turbulence model," *Turbulence, heat and mass transfer*, vol. 4, no. 1, pp. 625–632, 2003.

[25] I. H. Abbott and A. E. Von Doenhoff, *Theory of Wing Sections: Including a Summary of Airfoil Data*. New York: Dover Publications, 1959.

[26] C. E. Nwankpa, W. Ijomah, A. Gachagan, and S. Marshall, "Activation functions: Comparison of trends in practice and research for deep learning," *arXiv preprint arXiv:1811.03378*, 2018.

[27] M. A. Bouhlel, S. He, and J. R. R. A. Martins, "Scalable gradient–enhanced artificial neural networks for airfoil shape design in the subsonic and transonic regimes," *Structural and Multidisciplinary Optimization*, vol. 61, no. 4, pp. 1363–1376, 2020.

[28] H. Zhao and F. Magoules, "A review of the application of machine learning in building energy systems," *Energy Reports*, vol. 7, pp. 3431–3453, 2021.

[29] Y. Zhang, W. Sung, and D. Mavris, "Application of convolutional neural network to predict airfoil lift coefficient," 01 2018.

[30] E. Yilmaz and B. German, "A convolutional neural network approach to training predictors for airfoil performance," 06 2017.

[31] H. Moin, H. Zeeshan Iqbal Khan, S. Mobeen, and J. Riaz, "Airfoil's aerodynamic coefficients prediction using artificial neural network," in *2022 19th International Bhurban Conference on Applied Sciences and Technology (IBCAST)*, p. 175–182, IEEE, Aug. 2022.

[32] R.-E. Fan, P.-H. Chen, and C.-J. Lin, "Working set selection using second order information for training support vector machines," *Journal of Machine Learning Research*, vol. 6, pp. 1871–1918, 2005.

# Appendix A

# Design Points for Vessel Analysis

Table A.1: Parametric variables in each design point.

| Design Point | N | L | n | Velocity |
|---|---|---|---|---|
| 0 | 3 | 12 | 2 | 0.26 |
| 1 | 3 | 12 | 2 | 0.52 |
| 2 | 2 | 12 | 2 | 0.26 |
| 3 | 2 | 12 | 2 | 0.52 |
| 4 | 1 | 12 | 2 | 0.26 |
| 5 | 1 | 12 | 2 | 0.52 |
| 6 | 3 | 16 | 2 | 0.26 |
| 7 | 3 | 16 | 2 | 0.52 |
| 8 | 2 | 16 | 2 | 0.26 |
| 9 | 2 | 16 | 2 | 0.52 |
| 10 | 1 | 16 | 2 | 0.26 |
| 11 | 1 | 16 | 2 | 0.52 |
| 12 | 3 | 20 | 2 | 0.26 |
| 13 | 3 | 20 | 2 | 0.52 |
| 14 | 2 | 20 | 2 | 0.26 |
| 15 | 2 | 20 | 2 | 0.52 |

Table A.1: Parametric variables in each design point.

| Design Point | N | L | n | Velocity |
|---|---|---|---|---|
| 16 | 1 | 20 | 2 | 0.26 |
| 17 | 1 | 20 | 2 | 0.52 |
| 18 | 3 | 16 | 3 | 0.26 |
| 19 | 3 | 16 | 3 | 0.52 |
| 20 | 2 | 16 | 3 | 0.26 |
| 21 | 2 | 16 | 3 | 0.52 |
| 22 | 1 | 16 | 3 | 0.26 |
| 23 | 1 | 16 | 3 | 0.52 |
| 24 | 3 | 16 | 4 | 0.26 |
| 25 | 3 | 16 | 4 | 0.52 |
| 26 | 2 | 16 | 4 | 0.26 |
| 27 | 2 | 16 | 4 | 0.52 |
| 28 | 1 | 16 | 4 | 0.26 |
| 29 | 1 | 16 | 4 | 0.52 |
| 30 | 3 | 16 | 5 | 0.26 |
| 31 | 3 | 16 | 5 | 0.52 |
| 32 | 2 | 16 | 5 | 0.26 |
| 33 | 2 | 16 | 5 | 0.52 |
| 34 | 1 | 16 | 5 | 0.26 |
| 35 | 1 | 16 | 5 | 0.52 |
| 36 | 3 | 16 | 6 | 0.26 |
| 37 | 3 | 16 | 6 | 0.52 |
| 38 | 2 | 16 | 6 | 0.26 |
| 39 | 2 | 16 | 6 | 0.52 |
| 40 | 1 | 16 | 6 | 0.26 |
| 41 | 1 | 16 | 6 | 0.52 |

# Appendix B

# Numerical Results for CFD Analysis

Table B.1: Numerical Results for Different Stenosis Geometries

| Design Points | Max Wall Shear Stress (Pa) | Max Velocity (m/s) | Pressure Drop (Pa) | Max Strain Rate ($s^{-1}$) |
|---|---|---|---|---|
| 0 | 6.8 | 0.54 | 51 | 1750 |
| 1 | 17 | 1.025 | 140 | 4300 |
| 2 | 15 | 0.77 | 200 | 4000 |
| 3 | 37.5 | 1.45 | 634 | 10100 |
| 4 | 67.5 | 1.8 | 1564 | 18000 |
| 5 | 175 | 3.3 | 5285 | 45000 |
| 6 | 5.75 | 0.54 | 58 | 1300 |
| 7 | 13 | 1.025 | 151 | 3700 |
| 8 | 13 | 0.77 | 216 | 3450 |
| 9 | 32 | 1.45 | 663 | 8500 |
| 10 | 62.9 | 1.85 | 1693 | 16500 |
| 11 | 152 | 3.42 | 5594 | 41500 |
| 12 | 5.2 | 0.54 | 66 | 1250 |

Table B.1: Numerical Results for Different Stenosis Geometries

| Design Points | Max Wall Shear Stress (Pa) | Max Velocity (m/s) | Pressure Drop (Pa) | Max Strain Rate ($s^{-1}$) |
|---|---|---|---|---|
| 13 | 12.5 | 1.04 | 164 | 3250 |
| 14 | 11.5 | 0.79 | 235 | 3150 |
| 15 | 29 | 1.48 | 700 | 7850 |
| 16 | 57 | 1.9 | 1813 | 15500 |
| 17 | 143 | 3.5 | 5829 | 38600 |
| 18 | 7.7 | 0.545 | 71 | 2075 |
| 19 | 19.25 | 1.067 | 180 | 5370 |
| 20 | 14.5 | 0.733 | 236 | 3950 |
| 21 | 38.75 | 1.37 | 696 | 10700 |
| 22 | 63.85 | 1.87 | 2006 | 17150 |
| 23 | 163 | 3.45 | 6013 | 43150 |
| 24 | 9.5 | 0.54 | 80 | 2450 |
| 25 | 23 | 1.04 | 210 | 6470 |
| 26 | 16.5 | 0.71 | 231 | 4500 |
| 27 | 43 | 1.34 | 678 | 10250 |
| 28 | 65 | 1.71 | 2236 | 17680 |
| 29 | 170.5 | 3.12 | 6518 | 43200 |
| 30 | 10.3 | 0.54 | 85 | 2825 |
| 31 | 27.5 | 1.05 | 239 | 7525 |
| 32 | 19 | 0.76 | 282 | 5050 |
| 33 | 47 | 1.41 | 846 | 12550 |
| 34 | 67.5 | 1.78 | 2381 | 18020 |
| 35 | 176.3 | 3.31 | 6801 | 43020 |
| 36 | 11.25 | 0.542 | 93 | 3125 |
| 37 | 30.25 | 1.05 | 252 | 8455 |
| 38 | 20 | 0.75 | 295 | 5750 |
| 39 | 62 | 1.4 | 888 | 16500 |
| 40 | 72.6 | 1.76 | 2503 | 18430 |

Table B.1: Numerical Results for Different Stenosis Geometries

| Design Points | Max Wall Shear Stress (Pa) | Max Velocity (m/s) | Pressure Drop (Pa) | Max Strain Rate ($s^{-1}$) |
|---|---|---|---|---|
| 41 | 179.6 | 3.22 | 7018 | 44320 |

# Appendix C

# Machine Learning Method Examples

## C.1 Support Vector Regression

1. Assume having data set below:

| ID | $X_1$ | $X_2$ | y |
|---|---|---|---|
| 1 | 1 | 2 | 5 |
| 2 | 2 | 1 | 6 |
| 3 | 3 | 2 | 7 |
| 4 | 4 | 3 | 10 |
| 5 | 5 | 4 | 11 |

2. Kernel is choose as linear for simplicity, parameter C as 10 for moderate penalty for error and $\epsilon$ is chosen as 0.5 for small error tolerance.

3. From using linear kernel, the function SVR seeks become as below,

$$f(x_i, x_2) = w_i x_i + w_2 x_2 + b \tag{C.1}$$

4. Setting $\epsilon$ is equal to 0.5, make the model accept small error, $\pm 0.5$ is acceptable which means,

**if** $|y - y_{predict}| < \pm 0.5$ **then**
  No error penalty;
**else**
  Include $\xi_i$ and $\xi_i^*$;
**end**

5. For each sample the following predictions must satisfy for each data set:

$$y_i - \epsilon - \xi_i^* \leq w_i x_i + w_2 x_2 + b \leq y_i + \epsilon + \xi_i \tag{C.2}$$

6. After building the constraints , assume a perfect fit, where $\xi_i = \xi_i^* = 0$. So 10 inequality is obtained, which is two per sample as below.

$$\begin{aligned}
4.5 &\leq w_1(1) + w_2(2) + b \leq 5.5 \\
5.5 &\leq w_1(2) + w_2(1) + b \leq 6.5 \\
6.5 &\leq w_1(3) + w_2(2) + b \leq 7.5 \\
9.5 &\leq w_1(4) + w_2(3) + b \leq 10.5 \\
10.5 &\leq w_1(5) + w_2(4) + b \leq 11.5
\end{aligned}$$

7. Then reduce constraints by using middle values of the inequalities.

$$w_1(1) + w_2(2) + b = 5 \tag{C.3}$$
$$w_1(2) + w_2(1) + b = 6 \tag{C.4}$$
$$w_1(3) + w_2(2) + b = 7 \tag{C.5}$$
$$w_1(4) + w_2(3) + b = 10 \tag{C.6}$$
$$w_1(5) + w_2(4) + b = 11 \tag{C.7}$$
$$\tag{C.8}$$

8. Then the constraints is written as in a matrix format $AX = y$, and it solved by least squares.

$$\begin{bmatrix} 1 & 2 & 1 \\ 2 & 1 & 1 \\ 3 & 2 & 1 \\ 4 & 3 & 1 \\ 5 & 4 & 1 \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ b \end{bmatrix} = \begin{bmatrix} 5 \\ 6 \\ 7 \\ 10 \\ 11 \end{bmatrix}$$

$$X = (A^T A)^{-1} A^T y \tag{C.9}$$

9. By computing the the coefficients are obtained as $w_1 = 1$, $w_2 = 1$ and $b = 2.5$. So the model become as:

$$f(x_1, x_2) = 1x_1 + 1x_2 + 2.5 \tag{C.10}$$

10. After checking prediction if the error is in the margin decided, no slack penalty is required, the model is finalized.

11. If there were a result that do not satisfy the error margin, then its violation would become the slack variable, $\xi_i^*$.

## C.2 Ridge

1. Assume having data set below:

2. Assume the model is given in the equation C.11. And $\lambda$ is 1. Then the equation to minimize become as C.12.

$$\hat{y} = \beta_0 + \beta_1 x_1 + \beta_2 x_2 \tag{C.11}$$

$$J(\beta_0, \beta_1, \beta_2) = \sum_{i=1}^{n} (y_i - \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2})^2 + \lambda(\beta_1^2 + \beta_2^2) \tag{C.12}$$

Table C.2: Sample Input Data

| **ID** | $X_1$ | $X_2$ | y |
|---|---|---|---|
| 1 | 1 | 2 | 5 |
| 2 | 2 | 1 | 6 |
| 3 | 3 | 2 | 7 |
| 4 | 4 | 3 | 10 |
| 5 | 5 | 4 | 11 |

3. After solving the minimization equation with numerical methods, it is not possible to solve analytically due to L1 norm added, the equation become as below:

$$\hat{y} = 2.47 + 1.49 + 0.37x_2 \tag{C.13}$$

## C.3 Lasso

1. Assume having data set below:

| ID | $X_1$ | $X_2$ | y |
|---|---|---|---|
| 1 | 1 | 2 | 5 |
| 2 | 2 | 1 | 6 |
| 3 | 3 | 2 | 7 |
| 4 | 4 | 3 | 10 |
| 5 | 5 | 4 | 11 |

2. Assume the model is given in the equation C.14. And $\lambda$ is 1. Then the equation to minimize become as C.15.

$$\hat{y} = \beta_0 + \beta_1 x_1 + \beta_2 x_2 \tag{C.14}$$

$$J(\beta_0, \beta_1, \beta_2) = \sum_{i=1}^{n} (y_i - \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2})^2 + \lambda(|\beta_1| + |\beta_2|) \tag{C.15}$$

3. After solving the minimization equation with numerical methods, it is not possible to solve analytically due to L1 norm added, the equation become as below:

$$\hat{y} = 2.76 + 1.42 + 0x_2 \tag{C.16}$$

4. As told before, due to L1 term, $\beta_2$ became 0.

## C.4 K-Nearest Neighbor

1. Assume having data set below:

| ID | $X_1$ | $X_2$ | y |
|---|---|---|---|
| 1 | 1 | 2 | 5 |
| 2 | 2 | 1 | 6 |
| 3 | 3 | 2 | 7 |
| 4 | 4 | 3 | 10 |
| 5 | 5 | 4 | 11 |

2. For the distance function, *Euclidean Distance* is chosen.

3. For prediction a new sample is created with variables $X_1$ and $X_2$ as 3.2 and 2.4 respectively.

4. K will be chosen as 3.

5. Distance of predicted value to the rest of the data is calculated as using the formula below.

$$Distance = \sqrt{(X_1 - 3.2)^2 + (X_2 - 2.4)^2} \tag{C.17}$$

6. The distance become bellow as in the table:

| ID | $X_1$ | $X_2$ | y | Dsitance |
|---|---|---|---|---|
| 1 | 1 | 2 | 5 | $\sqrt{(1-3.2)^2 + (2-2.4)^2} = 2.24$ |
| 2 | 2 | 1 | 6 | $\sqrt{(2-3.2)^2 + (2-2.4)^2} = 1.84$ |
| 3 | 3 | 2 | 7 | $\sqrt{(3-3.2)^2 + (2-2.4)^2} = 0.45$ |
| 4 | 4 | 3 | 10 | $\sqrt{(4-3.2)^2 + (3-2.4)^2} = 1$ |
| 5 | 5 | 4 | 11 | $\sqrt{(5-3.2)^2 + (4-2.4)^2} = 2.41$ |

7. Then each point is sorted by the distance to the prediction point. The ID order become as [3, 4, 2, 1, 5].

8. Since the K value chosen as 3, average of the first three sample will be taken. So the prediction will be 7.67.

## C.5 Decision Tree

1. Assume having data set below:

| ID | $X_1$ | $X_2$ | y |
|---|---|---|---|
| 0 | 1 | 2 | 5 |
| 1 | 2 | 1 | 6 |

| | | | |
|---|---|---|---|
| 2 | 3 | 2 | 7 |
| 3 | 4 | 3 | 10 |
| 4 | 5 | 4 | 11 |

2. A decision tree with depth two will be built.

3. For the initial step, depth=1, all variables, $X_1$ and $X_2$, will be considered to find the best split that minimizes he loss function. For this example mean squared error is chosen as the loss function.

4. Initially trying $X_1$ ¡ 3.5, split samples as, samples with ID 1, 2, 3 goes on side with a mean value of 6 while samples 4 and 5 goes to other side with value 10.5. Then the loss is calculaed by:

$$\frac{1}{5}[(5-6)^2+(6-6)^2+(7-6)^2+(10-10.5)^2+(11-10.5)^2]=0.5 \quad \text{(C.18)}$$

5. Then variable $X_2$ used for split on the value of 2.5. In this split only sample 2 goes on side with value of 6 while the rest of the samples go to the other side with a mean value of 8.25. Then the loss function is calculated again.

$$\frac{1}{5}[(6-6)^2+(5-8.25)^2+(7-8.25)^2+(10-10.8.255)^2+(11-10.5)^2]=4.55 \quad \text{(C.19)}$$

6. From the loss function best split is found as $X_1$ ¡ 3.5. So the tree begins as below:

**if** $X_1 < 3.5$ **then**
  Go left;
**else**
  Go right;
**end**

7. After creating the first depth, second layer is needs to be constructed. Beginning with the left side, first split is created with trying to minimize the loss. Assume the best split is $X_1$, sample with ID 1 goes to one side with value of 5 while the rest of the samples on the left side (samples 2 and 3) goes the other side with a mean value of 6.5.

**if** $X_1 < 3.5$ **then**
  **if** $X_1 < 1.5$ **then**
    Predict 5
  **else**
    Predict 6.5
  **end**
**end**

8. For the right side, assume variable $X_2$ with split on 3.5 is the best split point. Then sample with ID 4 goes on one side with value of 10 while the other sample with ID 5 goes to other side with value of 11.

   From combining the dpets the final tree looks like below:

**if** $X_1 < 3.5$ **then**
  **if** $X_1 < 1.5$ **then**
    Predict 5
  **else**
    Predict 6.5
  **end**
**else**
  **if** $X_2 < 3.5$ **then**
    Predict 10
  **else**
    Predict 11
  **end**
**end**

9. For predicting a new sample, let it have $X_1$ and $X_2$ value of 2 and 1 respectively. For the initial split, $X_1$ is smaller than 3.5 so it will go to left. Then for having $X_1$ is greater than 1.5 the model will predict it as 6.5.

## C.6 Random Forest

1. Assume having data set below:

| ID | $X_1$ | $X_2$ | y |
|---|---|---|---|
| 0 | 1 | 2 | 5 |
| 1 | 2 | 1 | 5 |
| 2 | 3 | 2 | 7 |
| 3 | 4 | 3 | 11 |
| 4 | 5 | 4 | 12 |

2. Here, two decision trees will be built with having a depth of 2.

3. For the first tree the initial batch selected random. Let assume the selected sample ID's are [2, 3, 3, 4, 5]. As it seen, sample 3 selected twice and sample 1 was not selected. item Let the random selected feature be $X_1$ and split it on value 3.5.

   The samples on the on side are 2, 3 and 3 with a $X_1$ values of [6, 7, 7]. So their mean value becomes 6.67. The rest of the samples are in the other side and their mean value becomes as 11.5.

**if** $X_1 < 3.5$ **then**
  Go left;
**else**
  Go right;
**end**

4. The tree splits again, depth = 2. A new random variable is selected. Let it be $X_2$. For the left side it split on 1.5. So sample with ID 2 goes to one side with value of 6 and the two sample with ID 3 go to the other side with value 7.

```
if X2 < 1.5 then
|  Predict 6;
else
|  Predict 7;
end
```

5. For the right side again variable $X_2$ is chosen. For this one let is split on 3.5. SO the samples with goes one side with value 10 while smaple with ID goes to the other side with an value of 11.

6. So the finalized version of first tree is look like this:

```
if X1 < 3.5 then
|  if X2 < 1.5 then
|  |  Predict 6
|  else
|  |  Predict 7
|  end
else
|  if X2 < 3.5 then
|  |  Predict 10
|  else
|  |  Predict 11
|  end
end
```

7. For the next tree, another random batch from the inital data set is selected. Let assume the selected sample ID's are [1, 1, 2, 4, 5].

8. For the first split a random variable. Let the random variable be $X_2$. Split it on 2.5. So Only the sample with ID 2 goes the one side with a value of 6 while the rest of the sample goes to other side with an average value of 7.75.

9. For the dept 2's split, the left side cannot be split since it has only one sample. So it automatically predict 6. For the right side select again a random variable and let it be $X_1$ this time. It split on 3.5. So the two sample with ID 1 go one side with a value of 5, while the other two sample

with ID 4 and 5 goes to other side while having a mean value of 10.5. So he tree finalize as below.

```
if X2 < 2.5 then
 | Predict 6
else
 | if X1 < 3.5 then
 |  | Predict 5
 | else
 |  | Predict 10.5
 | end
end
```

10. For predicting a new sample, let it have $X_1$ and $X_2$ value of 3 and 2 respectively. For the initial tree $X_1$ is smaller than 3.5 so it will go to left. Then for having $X_2$ as 2 it will goes to right side since it is greater than 1.5. So the first tree predicts 7 as the output.

11. For the second tree it begins with $X_2$. Since the value is 2 and less than 2.5, it goes to left side and directly predicts the output as 6.

12. For the final prediction the average of predicted output in each tree is taken. So for this case average of 7 and 6 is taken which is 6.5

## C.7 Extreme Gradient Boosting

1. Assume having a data set with input and output x = [1, 2, 3, 4], y = [2, 4, 8, 16], respectively.

2. The model starts with calculating the initial prediction with is usually mean of outputs, $\hat{y}^0 = 7.5$ in this case. Then the residuals are calculated with respect to initial prediction.

$$r_i = y_i - \hat{y}_i = [-5.5, -3.5, 0.5, 8.5] \tag{C.20}$$

3. The first decision tree is rained to predict these calculated residuals. For the sake of the example, assume this tree makes piecewise decisions.

$$\text{If } x \leq 2.5, \text{ predict} \quad -4.5 \tag{C.21}$$

$$\text{If } x \geq 2.5, \text{ predict} \quad 4.5 \tag{C.22}$$

4. Then update the initial prediction using an arbitrary learning rate, for the sake of example make $\eta = 0.5$

$$\hat{y}^1 = \hat{y}^0 + \eta f_1(x) \tag{C.23}$$

5. From this new predictions comes as:

$$\text{For } x = 1 \textbf{ and } x = 2 \text{ , predict } 5.25 \tag{C.24}$$

$$\text{For } x = 3 \textbf{ and } x = 4 \text{ , predict } 9.75 \tag{C.25}$$

$$\tag{C.26}$$

6. Now new residuals, $r_2$ can be computed, [2 - 5.25, 4 - 5.25, 8 - 5.25, 16 - 5.25] = [-3.25, -1.25, -1.75, 6.25]

7. Having a new residual the second decision tree can be build and the process continue sequentially until the model reaches desired tolerance.