

· UNIVERSITY OF OXFORD ·
· DEPARTMENT OF ENGINEERING SCIENCE ·

Physics-Enhanced Machine Learning for Universal Wall Functions

Jianou Jiang

St Anne's College



A thesis submitted for the degree of

Doctor of Philosophy

Supervised by Professor Budimir Rosic

Trinity Term 2026

Declaration

I declare that this thesis is entirely my own work, and except where otherwise stated, describes
my own research.

Jianou Jiang

St Anne's College

This thesis is dedicated to . . .

Abstract

Classical algebraic wall functions in computational fluid dynamics fail in separated flows and strong pressure gradients—conditions critical for engineering applications. This thesis develops data-driven universal wall functions that achieve accurate predictions across attached, separated, and transitional flow regimes while maintaining computational efficiency. The methodology uses a dual-mesh training approach where coarse meshes ($y^+ \approx 5\text{--}10$) provide local stencil inputs and wall-resolved fine meshes ($y^+ < 2$) supply ground truth. A dataset of 244 simulations spanning diffusers, nozzles, and channels generates 25,485 training samples covering Reynolds numbers from 6,000 to 24,000.

Three complementary strategies are developed: physics-encoded inputs that transform 90 primitive variables into 11 core non-dimensional features (8-fold reduction); physics-guided hidden layers where neurons spontaneously learn representations correlated with established turbulence quantities; and physics-constrained learning incorporating conservation residuals into the training objective. The framework achieves 80% error reduction in separated flows compared to standard wall functions (8.9% versus 45% error), with a separation classifier detecting flow regime from local data at 98.8% accuracy. Integration into OpenFOAM demonstrates production readiness with 2.1% computational overhead.

Keywords: Wall functions, Machine learning, Reduced-order modelling, Flow separation, Heat transfer, CFD, OpenFOAM

Acknowledgements

I would like to express my deepest gratitude to my supervisor, Professor Budimir Rosic, for his invaluable guidance, continuous support, and patience throughout this research. His expertise in turbomachinery and computational fluid dynamics has been instrumental in shaping this work, and his encouragement has been a constant source of motivation.

I am grateful to the Department of Engineering Science at the University of Oxford for providing an exceptional research environment and the computational resources necessary for this work. The high-performance computing facilities were essential for the extensive simulations and machine learning experiments conducted in this thesis.

I would also like to thank my colleagues in the Oxford Thermofluids Institute for the stimulating discussions and collaborative atmosphere. Their feedback and insights have contributed significantly to the development of the ideas presented here.

Finally, I extend my heartfelt thanks to my family for their unwavering support and understanding throughout my doctoral studies. Their encouragement has been the foundation upon which this work was built.

Publications and Presentations

List the achievements and publications you've made during this DPhil journey.

Contents

Abstract	iv
Acknowledgements	v
Publications and Presentations	vi

CHAPTER 1: Introduction

1.1 Industrial Motivation: Where Wall Functions Matter	1
1.2 The Near-Wall Challenge in Turbulent Flow Simulation	3
1.2.1 The Computational Cost of Wall Resolution	3
1.2.2 Classical Wall Function Theory	4
1.3 Limitations of Classical Wall Functions	5
1.3.1 A Motivating Example: The Asymmetric Diffuser	5
1.4 The Machine Learning Opportunity	7
1.5 Three Approaches to Physics-Informed Learning	8
1.5.1 Method A: Physics-Encoded Input Features	8
1.5.2 Method B: Physics-Guided Network Architecture	8
1.5.3 Method C: Physics-Constrained Loss Functions	9
1.5.4 Complementary Methods	9
1.6 Research Questions	10
1.7 Research Objectives	10
1.8 Thesis Contributions	11
1.9 Scope and Limitations	12
1.10 Thesis Outline	13
1.11 Chapter Summary	14

CHAPTER 2: Literature Review

2.1 The Physics of Turbulent Boundary Layers	16
2.1.1 Prandtl's Boundary Layer Concept	16
2.1.2 The Blasius Solution and Laminar Boundary Layers	18
2.1.3 Transition to Turbulence	18
2.1.4 Reynolds Decomposition and the Closure Problem	19
2.1.5 The Structure of Turbulent Boundary Layers	20
2.1.6 Derivations of the Logarithmic Law	21
2.1.7 The Outer Layer and Wake Function	22
2.1.8 Turbulent Stress Distributions	23
2.2 The Mathematical Framework of Wall Functions	23
2.2.1 The Launder-Spalding Wall Function	24
2.2.2 Assumptions and Their Limitations	24
2.2.3 Enhanced Wall Treatments	25
2.3 Thermal Boundary Layers and Heat Transfer	26
2.3.1 The Reynolds Analogy	26
2.3.2 Thermal Law of the Wall	27
2.3.3 Dissimilarity Between Momentum and Heat Transfer	27

2.4	Industrial Computational Fluid Dynamics Practice	28
2.4.1	Commercial Implementations	28
2.4.2	Industrial Practice and Certification	29
2.4.3	The Academia-Industry Gap	29
2.5	Pressure Gradient Effects	30
2.6	The Machine Learning Revolution	30
2.6.1	Data-Driven Turbulence Modeling	31
2.6.2	Physics-Informed Neural Networks	32
2.6.3	Deep Learning Architectures for Turbulence	33
2.6.4	Machine Learning Wall Functions	33
2.6.5	Uncertainty Quantification in Machine Learning CFD	34
2.6.6	Transfer Learning and Domain Adaptation	35
2.7	Benchmark Cases and Validation	35
2.8	Related Applications and Extensions	37
2.8.1	Large Eddy Simulation Subgrid Modeling	37
2.8.2	Reduced-Order Modeling	37
2.8.3	Multi-Fidelity and Multi-Scale Approaches	38
2.9	Research Gaps and Thesis Contributions	38
2.10	Chapter Summary	39

CHAPTER 3: Methodology and Structured Data Generation

3.1	Overview of the Data Generation Framework	41
3.2	Geometry Design and Parameterization	43
3.2.1	Diffuser Geometries	44
3.2.2	Nozzle Geometries	45
3.2.3	Channel Flow	45
3.3	Mesh Generation Methodology	46
3.3.1	Fine Mesh Specifications	46
3.3.2	Coarse Mesh Specifications	46
3.3.3	Mesh Quality Metrics	47
3.3.4	Grid Independence Study	47
3.4	OpenFOAM Simulation Setup	48
3.4.1	Governing Equations	48
3.4.2	Turbulence Modeling	49
3.4.3	Boundary Conditions	49
3.4.4	Numerical Schemes and Solver Settings	50
3.4.5	Convergence Criteria	51
3.5	Validation Against Benchmark Data	52
3.5.1	Channel Flow Validation	52
3.5.2	Diffuser Validation	53
3.5.3	Heat Transfer Validation	54
3.6	Experimental Benchmark Data for Separated Flows	55
3.6.1	Backward-Facing Step	56
3.6.2	Wall-Mounted Hump	58
3.6.3	Periodic Hills	59
3.6.4	DNS Databases	60
3.6.5	Role of Benchmark Data in Training	61
(i)	Simulate-and-Replace Integration Method	62

(ii) Target Variable Consistency: Converting C_f to τ_w	62
(iii) Thermal Data Limitations in Benchmark Experiments.	63
3.7 Stencil-Based Input Representation	64
3.8 Dataset Summary and Statistics	65
3.8.1 Dataset Statistics	65
3.9 Chapter Summary.	66

CHAPTER 4: Data-Driven Velocity and Thermal Wall Functions

4.1 Neural Network Fundamentals	69
4.1.1 The Multilayer Perceptron	69
4.1.2 Activation Functions	69
4.1.3 Loss Functions and Optimization	70
4.2 Baseline Model Development	71
4.2.1 Data Preparation	71
4.2.2 Architecture Selection	71
4.2.3 Training Protocol.	73
4.3 Results	74
4.3.1 Training Convergence	74
4.3.2 Prediction Accuracy.	74
4.3.3 Comparison with Traditional Wall Functions	75
4.4 Sensitivity and Robustness Analysis.	76
4.4.1 Training Data Source Sensitivity.	76
4.4.2 Flow Separation Region Analysis	77
4.5 Limitations of Pure Data-Driven Learning	78
4.6 Chapter Summary.	79

CHAPTER 5: Physics-Based Feature Variables as Network Inputs

5.1 Theoretical Foundation	82
5.1.1 The Structure of Turbulent Boundary Layers	83
5.1.2 Wall-Law Scaling and Similarity	83
5.1.3 Pressure Gradient Effects.	85
5.1.4 Strain Rate and Rotation: The Mechanics of Turbulence Production.	87
5.1.5 Velocity Profile Shape and Separation.	87
5.1.6 Thermal Boundary Layer Physics	88
5.1.7 Summary of Physics-Based Features	89
5.1.8 Feature Robustness to Distribution Shift	90
5.2 Data Scaling and Feature Normalization	91
5.2.1 Training Data Composition	91
5.2.2 Flow Parameter Ranges	92
5.2.3 Normalization Methodology	93
5.2.4 Feature Variable Ranges	94
5.2.5 Generalization Considerations.	95
5.3 Experimental Methodology	97
5.4 Results: The Value of Physics-Based Features	97
5.4.1 Dimensionality Reduction Without Accuracy Loss	97
5.4.2 Performance in Separation Regions.	99
5.4.3 Comparison with Traditional Wall Functions	102

5.4.4	Generalization Across Data Sources	104
5.4.5	Error Distribution Analysis.	105
5.4.6	Learning Dynamics	106
5.5	Discussion: Why Physics Features Help	107
5.5.1	Encoding Scale-Invariant Relationships	107
5.5.2	Explicit Pressure Gradient Information	107
5.5.3	Robustness to Distribution Shift	108
5.5.4	The Cost of Redundant Features	108
5.6	Practical Recommendations	108
5.6.1	Recommended Feature Set	109
5.6.2	Feature Hierarchy by Robustness	109
5.6.3	Training Strategy	109
5.7	Chapter Summary.	110

CHAPTER 6: Physics-Based Feature Variables as Hidden Layer Neurons

6.1	Introduction and Motivation	113
6.2	Methodology.	113
6.2.1	Single Hidden Layer Architecture	113
6.2.2	Choice of Input Variables	114
6.2.3	Training Dataset	116
6.2.4	Neuron-Feature Correlation Analysis.	116
6.2.5	Architecture Invariance Testing	116
6.3	Results	116
6.3.1	Model Accuracy	116
6.3.2	Top Neuron-Feature Correlations	117
6.3.3	Correlation Heatmap	119
6.3.4	Architecture Invariance Results	120
6.4	Hybrid Networks: Replacing Neurons with Physics	121
6.4.1	Motivation for Neuron Replacement	121
6.4.2	Hybrid Network Architecture.	122
6.4.3	Explicit Physics Formulas for Neuron Replacement	123
6.4.4	Replacement Procedure	125
6.4.5	Experimental Results	125
6.4.6	Comparison: Pure Learned vs Hybrid	126
(i)	Performance by Flow Regime	127
(ii)	Effect of Replacement Threshold	128
6.4.7	Implications for Model Design	129
6.5	Training Data Source Sensitivity Analysis	130
6.5.1	Data Source Configurations	130
6.5.2	Model Accuracy Across Data Sources.	130
6.5.3	Architecture-Invariant Features Across Data Sources.	130
6.5.4	Unique Features by Data Source.	131
6.5.5	Separation Region Analysis.	131
6.5.6	Cross-Evaluation Robustness	132
6.5.7	Implications for Hybrid Networks	132
6.6	Flow Field Visualization and Comparison Studies	133
6.6.1	Comparison with Traditional Wall Functions	133
6.6.2	Effect of Geometry Variation	134

6.6.3	Flow Separation Analysis	136
6.6.4	Model Accuracy by Flow Regime	137
6.7	Discussion	138
6.7.1	Why Wall Distance is Architecture-Invariant	138
6.7.2	The Heat Flux Prediction Gap	138
6.7.3	Comparison with Physics-Based Input Features	139
6.7.4	Implications for Neural Network Design	139
6.7.5	Limitations	140
6.8	Chapter Summary	140

CHAPTER 7: Physics-Constrained Learning

7.1	Physics-Informed Neural Networks for Wall Functions	143
7.2	Conservation Laws as Soft Constraints	144
7.2.1	Streamwise Momentum Conservation	144
7.2.2	Wall-Normal Momentum with Buoyancy	144
7.2.3	Energy Conservation	145
7.2.4	Mass Conservation	145
7.3	Stencil-Based Finite Difference Implementation	145
7.3.1	Derivative Approximations	146
7.3.2	Wall Boundary Residuals	146
7.3.3	Residual Normalization	147
7.4	Combined Loss Function	147
7.4.1	Separation-Aware Loss Weighting	148
7.5	Experimental Configuration	148
7.5.1	Training Protocol	149
7.5.2	Experimental Conditions	149
7.6	Results: Accuracy vs Physical Consistency Trade-off	149
7.6.1	Effect of Physics Weight	149
7.6.2	Architecture Effects	150
7.6.3	Prediction Quality	151
7.7	Physics Weight Sensitivity Analysis	151
7.7.1	Optimal Operating Point	151
7.7.2	Loss Function Analysis	152
7.8	Discussion: When Physics Constraints Help	152
7.8.1	Extrapolation Robustness	152
7.8.2	Interpretability and Trust	152
7.8.3	Data Efficiency	153
7.9	Comparison with Previous Chapters	153
7.10	Physics Feature Variables as Training Constraints	153
7.10.1	Constraint Suitability Analysis	154
7.10.2	Feature Grouping Strategies	155
7.10.3	Ablation Study of Constraint Components	156
7.10.4	Constraint Loss Evolution During Training	157
7.10.5	Recommended Constraint Configuration	158
7.11	Chapter Summary	158

CHAPTER 8: Identification of Flow Separation

8.1	Introduction	160
8.2	The Distribution Shift Challenge.	162
8.2.1	Training Data Limitations	162
8.2.2	Mitigation Strategies	163
	Wall-Treatment-Robust Features.	163
	Architecture-Invariant Features.	163
	Onset Detection.	163
	Physics Constraints.	164
8.2.3	The Pressure Gradient as Primary Indicator	164
8.3	Separation Detection Framework.	165
8.3.1	Problem Formulation	165
8.3.2	Classifier Architectures	168
8.4	Experimental Results	169
8.4.1	Experiment 1: Baseline Classifier Comparison	169
	Physics Features Dramatically Improve Detection.	169
	Ensemble Methods Achieve Near-Perfect Classification.	169
	Linear Models Remain Competitive.	170
8.4.2	Experiment 2: Minimal Feature Set Evaluation	171
8.4.3	Experiment 3: Physics-Constrained Loss Functions	172
8.4.4	Experiment 4: Generalisation Study	173
8.5	Hybrid Wall Function Strategy	174
	Graceful Degradation.	176
	Computational Efficiency.	176
	Physical Consistency.	176
8.6	Discussion	180
8.6.1	Key Findings	180
8.6.2	Comparison with Wall Shear Stress Prediction	181
8.6.3	Practical Recommendations	181
8.7	Conclusions	183

CHAPTER 9: OpenFOAM Integration and Comprehensive Evaluation

9.1	OpenFOAM Integration Architecture.	185
9.1.1	Motivation for Production Integration	185
9.1.2	Implementation Architecture	186
9.1.3	Model Loading and Inference Backends.	188
9.2	Evaluation Methodology	188
9.2.1	Evaluation Dimensions	188
9.2.2	Comparison Methods	189
9.2.3	Benchmark Test Cases.	189
9.3	Accuracy Evaluation	190
9.3.1	Wall Quantity Prediction Accuracy.	190
(i)	Skin Friction Coefficient	190
(ii)	Separation and Reattachment Point Prediction.	192
(iii)	Wall Heat Flux	193

9.3.2	Full Flow Field Accuracy	194
(i)	Velocity Profiles	194
(ii)	Pressure Distribution	194
(iii)	Turbulent Quantities	195
9.3.3	Accuracy Summary	196
9.4	Computational Efficiency	196
9.4.1	Inference Time per Wall Face	197
9.4.2	Overall Simulation Cost	197
9.4.3	Memory Requirements	198
9.4.4	Efficiency Summary	199
9.5	Generalization Evaluation	199
9.5.1	Reynolds Number Extrapolation	199
9.5.2	Geometry Generalization	200
9.5.3	Three-Dimensional Generalization	200
(i)	3D Backward-Facing Step	201
(ii)	3D Periodic Hills	202
(iii)	Ahmed Body (Automotive Benchmark)	202
9.5.4	Training Data Source Sensitivity	203
(i)	Flow Regime Analysis	203
9.5.5	Generalization Summary	204
9.6	Robustness Evaluation	204
9.6.1	Mesh Sensitivity	204
9.6.2	Numerical Stability	205
9.6.3	Input Perturbation Sensitivity	206
9.6.4	Robustness Summary	207
9.7	Physical Consistency	207
9.7.1	Sign Consistency in Separated Flows	208
9.7.2	Boundedness and Physical Limits	208
9.7.3	Reynolds Analogy Consistency	209
9.8	Comprehensive Comparison Summary	210
9.8.1	Multi-Dimensional Comparison	210
9.8.2	Quantitative Summary Table	211
9.8.3	Recommendations by Application	212
9.9	Discussion	213
9.9.1	Key Findings	213
9.9.2	Limitations	215
9.9.3	Future Work	216
9.10	Chapter Summary	218

CHAPTER 10: Conclusion and Future Work

10.1	Summary of Contributions	219
10.1.1A	Unified Framework for Physics-Informed Wall Modelling	219
10.1.2	Dual-Mesh Training Methodology	220
10.1.3	Comprehensive Training Dataset	220
10.1.4	Flow Separation Detection	221
10.1.5	OpenFOAM Integration	221

10.2 Key Findings	222
10.2.1Physics-Encoded Inputs (Method A)	222
Feature Engineering Dramatically Improves Performance.	222
Non-Dimensional Formulation Enables Generalisation.	222
Feature Categories Have Different Importance. . .	222
Curated Feature Subsets Approach Full Performance.	223
10.2.2Physics-Guided Hidden Layers (Method B)	223
Neurons Spontaneously Align with Physics Features.	223
Architecture-Invariant Features Emerge.	223
L1-Regularised Networks Are More Interpretable.	223
Neuron Replacement Validates Physics Understanding.	223
10.2.3Physics-Constrained Learning (Method C).	224
Local Stencil PINNs Are Computationally Tractable.	224
Pure Data Fitting Achieves Highest Accuracy. . . .	224
Physics Constraints Trade Accuracy for Consistency.	224
Optimal Physics Weight Depends on Application.	224
10.2.4Separation Detection	225
Separation Is Detectable from Local Data.	225
Thermal Features Are Surprisingly Important. . . .	225
Ensemble Methods Outperform Neural Networks.	225
Generalisation Remains Challenging.	225
10.3 Synthesis: Combining the Three Methods	225
10.4 Quantitative Performance Summary and Method Selection Guidance	226
10.4.1Accuracy Improvements Over Standard Wall Functions	227
10.4.2Computational Efficiency and Production Readiness	228
10.4.3Mesh Resolution Flexibility and y^+ Range.	229
10.4.4Physical Consistency and Reliability	229
10.4.5Feature and Architecture Insights	230
10.4.6Method Selection Guidelines	231
Use ML-PINN for separated flows and out-of-distribution predictions.	231
Use ML-Physics for attached flows with non-standard conditions.	231
Use ML-Baseline for in-distribution interpolation.	232
Use standard wall functions with ML-guided mesh adaptation.	232
Use adaptive method selection for complex multi-regime flows.	232
10.4.7Key Discoveries and Novel Insights	233
Separation detection from local data is possible. . .	233
Neural networks spontaneously discover established physics.	233
Architecture invariance indicates fundamental physics.	234
Physics constraints improve generalisation more than accuracy.	234

Moderate physics weighting outperforms strong enforcement.	234
3D generalisation from 2D training is partially successful.	234
10.5 Limitations	235
10.5.1 Training Data Constraints	235
2D Geometry Focus.	235
Reynolds Number Range.	235
Incompressible Flow Assumption.	235
Steady-State Training.	235
10.5.2 Model Architecture Limitations	235
Fixed Stencil Size.	235
Single Output Point.	235
No Uncertainty Quantification.	236
10.5.3 Validation Limitations	236
No Experimental Validation.	236
Limited Out-of-Distribution Testing.	236
10.6 Future Work	236
10.6.1 Extension to Three-Dimensional Flows	236
10.6.2 Higher Reynolds Number Flows	237
10.6.3 Unsteady and Transitional Flows	237
10.6.4 Uncertainty Quantification	237
10.6.5 Compressible and Reacting Flows	238
10.6.6 Improved Neural Network Architectures	238
10.6.7 Experimental Validation	239
10.7 Broader Impact and Applications.	239
10.7.1 Aerospace Applications	239
10.7.2 Turbomachinery	240
10.7.3 Automotive Applications	240
10.7.4 Nuclear and Power Generation	241
10.8 Concluding Remarks	241
Bibliography	257

Code and Data Availability

List of Figures

1.1 Schematic of flow through an asymmetric diffuser, illustrating attached flow, separation, recirculation, and reattachment. Classical wall functions fail in the separation and recirculation regions where $\tau_w \leq 0$.	6
3.1 Data generation methodology for ML wall function training. Top row: Schematic views of stencil extraction from coarse mesh (left, input) and wall gradient computation from fine mesh (right, target). Middle row: Full diffuser geometry meshes showing the coarse mesh ($y^+ \approx 30$) and fine mesh ($y^+ < 2$) with zoom regions indicated. Bottom: Data generation pipeline flowchart showing how geometry parameters flow through dual-mesh simulations to produce training pairs ($\mathbf{X}_{\text{stencil}}, \tau_w, q_w$) for neural network training.	42
3.2 Complete training geometry family showing 15 unique geometry shapes (10 diffusers, 1 channel, 4 nozzles). Top: Overview with all bottom wall profiles superimposed, showing how the flat top wall (training data source) remains constant while the bottom wall varies to create different pressure gradient conditions. Bottom: Grid of individual geometries sorted by expansion ratio. Diffusers (green/yellow) create adverse pressure gradients, channels (gray) provide zero pressure gradient baseline, and nozzles (blue) create favorable pressure gradients. Each geometry is simulated at two Reynolds numbers (12,000 and 18,000) with two mesh resolutions, yielding 60 total training simulations.	44

3.3	Parameter space coverage of the training dataset. (a) Geometry parameters showing expansion/contraction ratio versus divergence angle for all 244 configurations. (b) Reynolds number distribution across geometry types. (c) Dataset composition by geometry category, with sample counts indicated.	46
3.4	Comparison of fine and coarse mesh near-wall resolution. (a) Fine mesh with $y^+ < 2$ first cell, showing dense clustering near the wall. (b) Coarse mesh with $y^+ \approx 5\text{--}10$ first cell, typical of production meshes requiring wall functions.	47
3.5	Grid independence study for a representative diffuser case. (a) Skin friction coefficient convergence with mesh refinement. (b) Nusselt number convergence. Richardson extrapolation values and $\pm 1\%$ bands are shown. The finest mesh (used for ground truth) lies within 1% of the extrapolated value for both quantities.	48
3.6	Boundary conditions for the diffuser configuration. Inlet conditions include specified velocity profile and turbulence quantities. Walls are no-slip with fixed temperature. Outlet uses a zero-gradient pressure condition.	50
3.7	Residual convergence history for a representative diffuser case. All residuals reach the convergence criterion of 10^{-6} within 5000 iterations. The velocity and pressure equations converge fastest, followed by temperature and turbulence quantities.	52
3.8	Mean velocity profiles in wall units for three flow configurations. (a) Channel flow compared to DNS data and analytical laws. (b) Diffuser with adverse pressure gradient showing deviation from log-law. (c) Nozzle with favorable pressure gradient. Fine mesh results capture the physics accurately, while coarse mesh results show the behavior that wall functions must correct.	53

3.9 Diffuser validation against Buice & Eaton (1997) experimental data. (a) Asymmetric diffuser geometry with 10° divergence and 4.7:1 expansion ratio. (b) Skin friction coefficient distribution showing good agreement between fine mesh $k-\omega$ SST simulation and experimental measurements. The expansion region is highlighted in yellow.	54
3.10 Heat transfer validation. (a) Temperature profile in wall units (T^+ vs y^+) for channel flow at $Pr = 0.71$, comparing fine mesh results with JAXA DNS data and analytical laws. (b) Nusselt number development along the channel, showing entrance region effects and comparison with the Dittus-Boelter correlation ($Nu = 55.3$ for fully developed flow).	55
3.11 Backward-facing step geometry and flow structure. (a) Schematic showing the step height H , inlet boundary layer, separation at the step corner, recirculation zone, and reattachment downstream. (b) Characteristic flow regions: 1-upstream attached flow, 2-separation shear layer, 3-recirculation zone with $\tau_w < 0$, 4-reattachment region, 5-recovery zone.	56
3.12 Skin friction coefficient distribution for the backward-facing step from Driver & Seegmiller (1985). The flow separates at the step corner ($x/H = 0$), exhibits negative C_f in the recirculation zone (minimum at $x/H \approx 3$), reattaches at $x/H \approx 6.26$, and recovers downstream. The shaded region indicates where classical wall functions fail.	57
3.13 Wall-mounted hump geometry based on NASA experiments. (a) The hump profile with chord length $c = 420$ mm and maximum height $h/c = 0.128$ mounted on a flat plate. Flow accelerates over the windward face and separates on the leeward side. (b) Skin friction distribution showing the favorable pressure gradient (FPG) region, separation point, recirculation zone, and recovery.	58

3.14 Periodic hills geometry showing two periods of the domain. (a) Hill profile with height H , period $L_x = 9H$, and channel height $L_y = 3.035H$. The polynomial hill shape creates smooth pressure gradient transitions. (b) Flow structure showing separation on each lee side and reattachment in the valleys, with recirculation indicated by reversed streamlines.	59
3.15 Wall shear stress distribution over one period of the periodic hills geometry at $Re_H = 10,595$ from Breuer et al. (2009). The flow experiences favorable pressure gradient (FPG) on the windward face with high C_f , transitions to adverse pressure gradient (APG) past the crest, separates near $x/H = 0.5$, and recovers after reattachment near $x/H = 4.5$	60
3.16 Statistical analysis of the training dataset. (a) Distribution of skin friction coefficient C_f values. (b) Distribution of Stanton number St values. (c) Correlation between C_f and St, showing the expected positive relationship. (d) Summary statistics table including mean, standard deviation, and percentiles.	66
4.1 Conceptual overview of the data-driven baseline approach. The 3×5 local stencil containing primitive flow variables (U, V, P, T) is flattened and normalized before being processed by a multilayer perceptron (MLP) to predict wall shear stress τ_w and wall heat flux q_w . This purely data-driven architecture serves as the baseline against which physics-informed approaches are evaluated.	68
4.2 Training and test loss curves for the baseline neural network. Both curves show consistent convergence behavior, with the test loss remaining close to training loss, indicating good generalization without significant overfitting.	74

4.3 Scatter plots of predicted versus true values for skin friction coefficient C_f (top) and Stanton number St (bottom). Points clustered along the diagonal indicate accurate predictions.	75
4.4 Comparison of velocity profiles: traditional wall functions versus ML predictions. The ML model captures the deviation from log-law behavior caused by adverse pressure gradients.	76
5.1 Conceptual overview of the physics-encoded inputs approach. Raw flow data from the 3×5 stencil undergoes physics-based feature engineering to produce 58 dimensionless quantities (y^+ , u^+ , $\partial p / \partial x$, Re_y , etc.) before being processed by the neural network. This transformation encodes established turbulence physics directly into the input representation.	82
5.2 Conceptual overview of physics-based feature engineering. Flow field information is transformed into physics-based features (y^+ , u^+ , $\partial p / \partial x$, Re_y , $\sqrt{S_{ij} S_{ij}}$) that encode scale-invariant relationships, pressure gradient effects, and turbulence production mechanisms. These features provide the neural network with information in a form that directly reflects the governing equations, reducing the learning burden compared to primitive variables.	82
5.3 Classical wall function laws showing the linear sublayer ($u^+ = y^+$), log-law region ($u^+ = (1/\kappa) \ln(y^+) + B$), and Spalding's unified correlation. The universal velocity profile collapses data across Reynolds numbers when expressed in wall units, motivating the use of y^+ and u^+ as physics-based features.	84

5.4 Boundary layer velocity profiles from CFD simulations showing the physics-based features u^+ and y^+ in different flow regimes. Left: attached flow ($ER \approx 1.0$) where profiles closely follow the log-law. Centre: mild adverse pressure gradient ($ER \approx 2.0$) where profiles begin to deviate. Right: strong adverse pressure gradient ($ER \approx 4.0$) approaching separation with significant log-law deviation. The coloured markers indicate different streamwise locations, demonstrating how the boundary layer evolves through the diffuser.	85
5.5 Spatial distribution of the streamwise pressure gradient $\partial p / \partial x$ for different expansion ratios. The pressure gradient is a key physics-based feature that encodes non-equilibrium effects. Blue regions indicate favorable pressure gradient (FPG, accelerating flow), while red regions indicate adverse pressure gradient (APG, decelerating flow). The intensity of APG increases with expansion ratio, directly correlating with boundary layer thickening and separation risk.	86
5.6 Flow field visualization showing velocity magnitude (left column) and pressure field (right column) for three representative expansion ratios. Top row: straight channel ($ER = 1.0$) with uniform flow. Middle row: mild diffuser ($ER = 2.5$) with gradual expansion. Bottom row: strong diffuser ($ER = 4.5$) with pronounced expansion and flow deceleration. The velocity contours show acceleration in the inlet and deceleration in the expansion region, while the pressure contours show the adverse pressure gradient that drives the flow physics. . . .	92

5.7 Wall shear stress distribution analysis. (a) Streamwise τ_w distribution for different expansion ratios, showing the characteristic behaviour in each flow region. (b) Skin friction coefficient C_f distribution. (c) Flow regime identification for a high expansion ratio case, with attached (green) and separated (red) regions clearly marked. (d) Mean wall shear stress magnitude by flow region across all training cases, showing how τ_w decreases through the diffuser as adverse pressure gradient increases.	95
5.8 Feature set comparison analysis. (a) Prediction accuracy by feature set, showing R^2 scores for C_f and St prediction across Core (15 features), Robust (12 features), and Full (58 features) configurations. (b) Feature efficiency plot demonstrating the trade-off between number of features and average prediction accuracy. The Core feature set achieves near-optimal performance with substantially fewer inputs than the Full set.	98
5.9 Predicted versus true wall quantities for the physics Core model (11 features). Left: skin friction coefficient C_f . Right: Stanton number St. The tight clustering around the diagonal indicates accurate prediction across the full range of flow conditions.	99
5.10 Flow regime analysis showing skin friction distribution across attached, near-separation, and separation regions. The wall-modelled data exhibits distinct clustering by flow regime, with separation events concentrated at low C_f values where traditional wall functions fail.	100

5.11 Boundary layer velocity profile development through a diffuser (ER = 2.5) at six streamwise locations. The profiles show the evolution from attached flow at the inlet to separated flow downstream. Reversed flow regions ($U/U_{\max} < 0$) are highlighted in red, demonstrating how physics-based features must capture the transition from attached to separated conditions. The profile shapes directly inform the wall shear stress prediction task.	101
5.12 Comprehensive flow regime analysis. (a) Prediction accuracy degrades progressively from attached flow ($R^2 = 0.96$) through near-separation ($R^2 = 0.82$) to separation ($R^2 = 0.65$), reflecting the increasing complexity of non-equilibrium physics. (b) Sample distribution showing the relative abundance of each flow regime in the training data. (c) Prediction error (RMSE) by regime, demonstrating the challenge of accurate prediction in separated flows.	101
5.13 Direct comparison between machine learning and traditional wall function approaches. (a) Prediction accuracy: the ML model with physics-based features achieves $R^2 = 0.92$, while Spalding and log-law wall functions fail with $R^2 < 0.15$. (b) Prediction error: RMSE values demonstrate the order-of-magnitude improvement provided by the ML approach over equilibrium-based correlations in adverse pressure gradient flows.	103
5.14 Comparison between neural network predictions (physics Core features) and traditional wall function correlations. Top row: skin friction coefficient C_f . Bottom row: Stanton number St. The traditional methods (Spalding, log-law) show large scatter in adverse pressure gradient conditions, while the neural network maintains accuracy across all flow regimes.	103

5.15 Data source comparison study results. Models trained on original (wall-resolved) data, wall-function (wall-modelled) data, and combined data are evaluated across flow regimes. The combined training approach achieves consistent performance across all conditions.	104
5.16 Error distribution for the Physics Core model predictions. Left: C_f prediction errors. Right: St prediction errors. The distributions are approximately Gaussian with near-zero mean, indicating unbiased predictions. The tight distribution tails demonstrate that large errors are rare even in challenging flow conditions.	105
5.17 Comparison of performance metrics across feature sets and model configurations. The Core physics features (11 inputs) achieve comparable or better performance than the Full feature set (58 inputs) while significantly outperforming primitive variable baselines.	106
5.18 Training and test loss curves comparing Physics Core (11 features) and Full (58 features) models. The Core model converges faster and achieves lower final loss, demonstrating that the reduced feature set is more efficient to learn.	106
6.1 Conceptual overview of the physics-guided hidden layers approach. A single hidden layer network is trained on basic inputs (y^+ , u^+ , v^+ , $\partial p/\partial x$, $\partial p/\partial y$, y_T^+) to predict wall quantities. Neuron activations are then correlated with the 58-feature physics library to identify architecture-invariant features that emerge across networks of different sizes (8, 16, 32 neurons), revealing which physical relationships the network has learned to compute internally.	112

6.2	Top neuron-feature correlations for the L1_32 architecture. Each bar represents a neuron's correlation with its best-matching physics feature. Correlations exceeding $ r = 0.8$ (dashed line) indicate strong matches.	117
6.3	Correlation heatmap between hidden layer neurons (rows) and physics features (columns) for the L1_32 architecture. Strong correlations ($ r > 0.8$) appear as bright red or blue. The non-uniform pattern indicates that different neurons specialize in different physics.	119
6.4	Architecture invariance analysis. Features discovered by multiple architectures are more likely to represent fundamental physics.	120
6.5	Interpreted network architecture for L1_32. Each hidden layer neuron is labelled with its best-matching physics feature. The architecture-invariant features (y^+ , $\log(y_T^+)$, log-law scaling) appear regardless of network size, demonstrating that the network learns genuine physics rather than arbitrary patterns.	121
6.6	Hybrid network architecture with neuron replacement. Green nodes represent physics neurons (explicit formulas), yellow nodes represent learned neurons. The physics neurons directly use features like pressure gradient and viscous scaling, while learned neurons compute the remaining transformations.	123
6.7	Comparison of pure learned and hybrid networks. (a) Prediction accuracy showing minimal degradation. (b) Parameter count showing 22% reduction. (c) Network composition showing 41% interpretable physics neurons.	127
6.8	Hybrid network performance advantage by flow regime. Positive values indicate the hybrid network outperforms pure learned. The hybrid architecture shows increasing advantage as flow conditions become more challenging, with the largest improvement in separated flow regions where explicit pressure gradient encoding provides the greatest benefit.	128

6.9 Comparison of wall shear stress (τ_w) between fine mesh (ground truth) and coarse mesh with standard wall functions for different geometries: channel flow (ER=1.0) and diffusers with increasing expansion ratios (ER=1.5, 2.0, 2.5). The standard wall function consistently underpredicts τ_w and fails to capture the rapid variations in the transition region.	134
6.10 Effect of expansion ratio on wall shear stress distribution. Top: Geometry schematic showing the diffuser family with flat top wall (data collection surface) and inclined bottom wall. Different expansion ratios (ER=0.5 to 5.0) are overlaid, demonstrating the range from nozzle (contracting) through channel to strong diffuser (expanding). Bottom: Corresponding τ_w profiles showing the transition from favourable pressure gradient (FPG, accelerating flow with increasing τ_w) through zero pressure gradient (ZPG, constant τ_w) to adverse pressure gradient (APG, decaying τ_w). At ER=5.0, the flow approaches separation where τ_w crosses zero.	135
6.11 Flow separation analysis for a diffuser with ER=4.0 and $\theta=8^\circ$. Top: Streamwise velocity contour showing flow deceleration through the diffuser. Yellow markers indicate regions of reversed flow. Bottom left: Wall shear stress distribution with attached (blue) and separated (red) regions identified. Bottom right: Skin friction coefficient showing the transition from favourable to adverse pressure gradient conditions.	136
6.12 Wall shear stress prediction accuracy (R^2) by flow regime. Traditional wall functions perform well in attached flow but fail dramatically in separated regions. The ML models with basic inputs (this chapter) show improved performance across all regimes, while ML with physics features (Chapter 5) achieves the highest accuracy throughout.	137

7.1	Conceptual overview of the physics-constrained loss (PINN) approach. The neural network receives stencil inputs and produces predictions for wall shear stress τ_w and heat flux q_w . The total loss function combines the standard data loss (MSE against ground truth) with physics residual terms computed via finite differences on the stencil. These residuals enforce local conservation of momentum (R_u , R_v), energy (R_T), and mass (R_{div}), constraining the network to produce physically consistent predictions.	142
7.2	Physics-constrained learning results: (a) Training convergence showing MSE-only achieving lower loss than physics-constrained; (b) Prediction accuracy comparison showing minimal degradation with physics constraints; (c) Physics constraint satisfaction showing PINN reduces residual magnitudes; (d) Sensitivity of $R_{\tau_w}^2$ to physics weight λ_{physics}	150
7.3	Predicted versus true values for wall shear stress τ_w (left) and wall heat flux q_w (right). Red: MSE-only model; Blue: Physics-constrained ($\lambda = 0.1$). Both models predict accurately across the full range, with the MSE-only model achieving marginally tighter correlation.	151
7.4	Physics feature constraint suitability analysis. (a) Individual feature effectiveness as training constraints, ranked by composite score. (b) Gradient stability during training for top features. (c) Correlation matrix showing feature grouping potential. (d) Recommended constraint configurations based on the analysis. .	154
7.5	Ablation study of physics loss components. Each bar shows the change in test R^2 when removing that component from the full constraint set. Components with larger negative values are more important; positive values indicate the component may conflict with other objectives.	156

7.6 Training dynamics for different constraint configurations. (a) Total loss convergence. (b) Individual constraint residual evolution. (c) Gradient magnitude stability. (d) Learning rate schedule adaptation.	157
8.1 Conceptual overview of the separation detection and hybrid wall function approach. Flow data from the local stencil is processed by an ML classifier to determine the flow regime. In attached flow regions (classifier output below threshold), the computationally efficient traditional wall function is applied. In separated or near-separation regions (classifier output above threshold), the ML wall function from previous chapters provides accurate predictions where traditional methods fail.	160
8.2 Flow separation detection in a diffuser geometry. (a) Velocity magnitude with streamlines showing the recirculation zone. (b) Wall shear stress distribution with separation ($\tau_w < 0$) and reattachment locations marked. (c) ML classifier probability map showing separation detection. (d) Classification accuracy along the wall comparing true separation (from τ_w) with predicted separation. (e) Streamwise pressure gradient driving the separation. (f) Evolution of key physics features used for classification.	168

8.3 Comprehensive comparison of separation classifier architectures. (a) ROC curves showing true positive rate versus false positive rate for each classifier, with Random Forest achieving near-perfect discrimination. (b) Precision-recall curves demonstrating the trade-off between detecting all separations versus minimising false alarms. (c) Calibration plots comparing predicted probabilities with observed frequencies—well-calibrated classifiers lie along the diagonal. (d) Confusion matrices for the three main classifiers showing the distribution of prediction errors.	171
8.4 Generalisation analysis of the separation classifier. (a) Learning curves showing accuracy and F1-score versus training set size. (b) Cross-validation F1-score distribution across 5 random seeds, highlighting performance variability. (c) Feature importance stability across cross-validation folds. (d) Comparison of error types between best and worst performing folds. (e) Decision boundary visualisation in the two-dimensional space of the most important features. (f) Classifier confidence distribution for correctly and incorrectly classified samples.	174
8.5 Hybrid wall function strategy with separation detection. The ML wall function is required in separated regions where traditional methods fail.	175
8.6 Validation of the hybrid wall function strategy across diffuser geometries. (a) Wall shear stress comparison between traditional, ML-only, and hybrid approaches for a 10° expansion. (b) Same comparison for a 20° expansion showing more extensive separation. (c) Heat flux predictions demonstrating consistent improvement in thermal quantities. (d) Computational overhead analysis showing the cost-benefit trade-off. (e) Classifier decision boundaries superimposed on the true separation regions. (f) Convergence behaviour comparing iteration count to steady state for each approach.	179

8.7 Spatial comparison of separation detection accuracy. (a) Classifier probability versus true separation state along the wall for a representative diffuser case. (b) Detailed view of the separation onset region showing early detection capability. (c) Detailed view of the reattachment region. (d) Spatial accuracy metrics (precision, recall, F1) as a function of streamwise position. (e) Comparison across multiple test geometries showing consistent detection quality. (f) Failure mode analysis identifying regions where detection is least reliable.	180
9.1 OpenFOAM integration architecture. The ML wall function boundary condition is called during each SIMPLE/PISO iteration, extracts physics features from the local flow field, performs neural network inference, and returns turbulent viscosity ν_t at wall faces.	187
9.2 Skin friction coefficient comparison for attached flow cases. (a) Channel flow across Reynolds numbers. (b) Diffuser flat wall region. (c) Summary of relative errors for all methods compared to wall-resolved reference.	191

9.3 Comprehensive skin friction coefficient comparison across benchmark separated flow cases. Panel (a) shows the backward-facing step results at $Re_H = 37,500$, comparing experimental data (Driver & Seegmiller, symbols), wall-resolved LES reference (black line), standard Spalding wall function (red dashed), and three ML variants (orange, blue, green). The standard wall function fails completely in the separation bubble ($0 < x/H < 6.26$), predicting small positive C_f where the actual flow has reversed. ML methods correctly capture negative C_f , with ML-PINN achieving closest agreement to reference data. Panel (b) demonstrates periodic hills behavior over one wavelength, where cyclic separation on the lee side challenges all approaches. Panel (c) presents the wall-mounted hump geometry with smooth separation onset and reattachment, testing the models' ability to resolve gradual adverse pressure gradient effects. Panel (d) quantifies error magnitudes across all test cases: separated flow errors (red bars) dominate total error budgets, with ML-PINN reducing separated flow error from 45% (standard WF) to 9%, an 80% improvement. The overall error comparison reveals that physics-informed methods achieve under 10% error even on strongly out-of-distribution geometries.	192
9.4 Thermal wall function comparison. (a,b) Attached flow cases where all methods perform reasonably. (c) Separated flow case showing enhanced heat transfer in reattachment region. (d) Summary of Stanton number prediction errors.	193
9.5 Velocity profile comparison at selected streamwise stations. The ML wall function's influence propagates throughout the boundary layer, not just the wall-adjacent cell.	194

9.6	Surface pressure coefficient distribution. Accurate pressure prediction is critical for engineering applications (drag, lift). The wall function affects pressure through its influence on separation location and recirculation zone size.	195
9.7	Turbulent quantity predictions. The wall function boundary condition directly specifies ν_t at walls, which influences k and ω through the transport equations.	196
9.8	Inference timing analysis. The native C++ backend achieves sub-microsecond inference, comparable to standard wall function evaluation.	197
9.9	Comprehensive computational efficiency analysis. Panel (a) compares inference time per wall face evaluation across backends: the standard algebraic wall function requires 0.05 μs as baseline, LibTorch (PyTorch C++) incurs 2.8 μs ($56\times$ slower), ONNX Runtime achieves 1.2 μs ($24\times$ slower), but the native C++ implementation reaches 0.48 μs (only $10\times$ slower). Panel (b) plots total simulation overhead versus number of wall boundary faces: native C++ maintains under 3% overhead even for meshes with 10^6 wall faces, while ONNX reaches 5% and LibTorch exceeds 8%. Panel (c) decomposes the 0.48 μs native C++ cost: feature extraction (0.18 μs , 38%), neural network forward pass (0.25 μs , 52%), and post-processing (0.05 μs , 10%). Panel (d) demonstrates that ML wall functions do not impair solver convergence—residuals decay at similar rates regardless of wall treatment, with ML-PINN converging at comparable iterations to standard WF. Panel (e) quantifies memory requirements: ML-PINN requires 145 KB for model weights and 9.8 MB runtime memory, negligible compared to typical CFD memory budgets. Panel (f) visualizes the accuracy-cost trade-off: standard WF occupies the low-cost/low-accuracy corner, wall-resolved simulations achieve perfect accuracy at $100\times$ cost, while ML-PINN delivers 95% of wall-resolved accuracy at merely $2.3\times$ standard WF cost.	198

9.10 Reynolds number generalization. Training data covers $Re = 8,000\text{--}24,000$. Performance is evaluated at both interpolation and extrapolation conditions. . .	199
9.11 Geometry generalization. Performance degrades predictably as geometry deviates from training distribution, but ML methods maintain advantage over standard wall functions.	200
9.12 Three-dimensional generalization. Models trained on 2D simulations are evalu- ated on 3D flows with spanwise variations, secondary flows, and 3D separation. .	201
9.13 3D backward-facing step results. Spanwise variations in the recirculation zone test the wall function's ability to handle 3D effects.	201
9.14 3D periodic hills results. The 3D separation bubble structure differs from 2D, testing model robustness to three-dimensional effects.	202
9.15 Ahmed body automotive benchmark. This practical test case evaluates wall function performance on a real-world geometry with complex 3D separation. . .	202

9.16 Comprehensive mesh sensitivity and y^+ dependence analysis. Panel (a) plots skin friction error versus first cell y^+ for channel flow at $Re = 12,000$, revealing that standard wall functions (red) suffer high error below $y^+ \approx 30$ where the log-law validity assumptions break down, achieve optimal performance in the $30 < y^+ < 100$ range they were designed for, then degrade beyond $y^+ > 200$ as the wall-adjacent cell grows too large to resolve boundary layer gradients. ML methods exhibit broader operating ranges: ML-PINN (green) maintains under 5% error from $y^+ = 2$ to $y^+ = 120$, a $60\times$ range compared to standard wall functions' $10\times$ range. The shaded regions indicate recommended operating windows. Panel (b) examines separated flow (BFS recirculation zone), where standard wall functions fail regardless of mesh resolution—the dashed red line remains flat near 40% error across all y^+ because the fundamental model breakdown stems from physical assumption violations, not mesh adequacy. ML methods show degradation at extreme y^+ values but maintain reasonable accuracy throughout. Panel (c) quantifies recommended y^+ ranges as error bars, showing that ML-PINN operates reliably from $y^+ = 2$ to $y^+ = 120$ (optimal near 10), while standard methods require $30 < y^+ < 300$ (optimal near 50). This flexibility enables practitioners to use coarser meshes than wall-resolved LES ($y^+ < 1$) while avoiding the strict y^+ targeting that standard wall functions demand. Panel (d) presents mesh independence study results, demonstrating that ML methods achieve mesh convergence on coarser grids: ML-PINN reaches 95% of asymptotic accuracy on the medium mesh ($y^+ \approx 40$), whereas standard WF requires the fine mesh ($y^+ \approx 20$) for equivalent convergence. 205	205
9.17 Numerical stability analysis. The ML wall function should not degrade solver convergence compared to standard wall functions. 206	206

9.18 Input perturbation sensitivity. Robust models should have bounded output changes for small input perturbations.	207
9.19 Sign consistency in separated flows. Standard wall functions cannot predict negative τ_w . ML methods should correctly identify recirculation zones.	208
9.20 Boundedness analysis. Neural networks can produce unphysical outputs; this analysis quantifies such violations and their impact.	209
9.21 Reynolds analogy consistency. Physics-constrained models better maintain the theoretical relationship between momentum and thermal transport.	210
9.22 Multi-dimensional comparison radar chart and comprehensive summary. Top panel shows radar chart with 9 evaluation dimensions comparing all methods—larger area indicates better overall performance. ML-PINN (green) dominates across most dimensions. Bottom panels present recommendations matrix by application scenario and quantitative performance summary table with detailed metrics across all evaluation categories.	211

List of Tables

1.1	Approximate mesh requirements for wall-resolved versus wall-modeled RANS simulation of channel flow. Wall-resolved meshes require $y_1^+ < 1$; wall-modeled meshes use $y_1^+ \approx 30\text{--}100$	4
3.1	Mesh quality requirements for fine and coarse meshes.	47
3.2	Numerical discretization schemes used in OpenFOAM simulations.	51
3.3	Under-relaxation factors for the SIMPLE algorithm.	51
3.4	DNS databases used for validation and training data augmentation.	60
3.5	Summary of experimental and DNS benchmark cases for separated flows. . . .	64
3.6	Summary of the generated training dataset.	65
4.1	Hyperparameter search space for architecture selection.	72
4.2	Hyperparameter search results showing representative configurations. The 3-layer network with 64 neurons achieves the best combined R^2	72
4.3	Effect of stencil size on model performance.	73
4.4	Performance metrics for the baseline neural network wall function. Values reported as mean \pm standard deviation over 5 independent runs.	75
4.5	Comparison of ML wall function against traditional analytical formulations. .	76
4.6	Baseline MLP performance across training data sources.	76
4.7	Distribution of training samples across flow regimes.	77
4.8	Baseline MLP performance by flow regime.	77
5.1	Summary of physics-based feature library with physical justification.	90
5.2	Training dataset composition by geometry type.	91
5.3	Training parameter ranges and their coverage.	93
5.4	Feature variable ranges from training data (25,485 samples).	94
5.5	Comparison of primitive variables (Chapter 4) vs. physics-based features. . .	97
5.6	Performance by flow regime: Primitive baseline vs. Physics Core features. . .	99
5.7	Traditional wall function performance on wall-modelled data (22,140 samples). <td>102</td>	102
5.8	Cross-evaluation: Performance when training and test data sources differ. . .	104
5.9	Recommended Core physics features for wall function prediction.	109
6.1	Basic normalized input variables for neuron correlation experiments. These represent a minimal set between truly primitive variables and full physics features.	115
6.2	Prediction accuracy with basic inputs only (25,485 samples).	117
6.3	Top neuron-feature correlations (L1_32 architecture, 25,485 samples).	118
6.4	Architecture-invariant features discovered across all tested networks.	120
6.5	Neurons identified for replacement and their corresponding physics features. .	126
6.6	Comparison of pure learned and hybrid network architectures.	126
6.7	Pure learned vs hybrid network accuracy (R^2 for τ_w) by flow regime.	127
6.8	Effect of correlation threshold on hybrid network performance.	129
6.9	L1-PINN (32 neurons) accuracy across training data sources.	130
6.10	Architecture-invariant features by data source. Features appearing in all three architectures (8, 16, 32 neurons) with moderate-to-strong correlation ($ r > 0.5$). <td>131</td>	131
6.11	Physics features unique to specific data sources.	131

6.12	Neuron correlation with separation-relevant features by flow regime.	132
6.13	Cross-evaluation: $\tau_w R^2$ when training and test data sources differ.	132
6.14	Comparison of approaches: physics features as inputs vs. as emergent neurons.	139
7.1	PINN experiment results comparing MSE-only and physics-constrained training. Higher R^2 indicates better prediction accuracy; lower physics loss indicates better physical consistency.	149
7.2	Comparison of wall function modeling approaches across thesis chapters.	153
8.1	Distribution shift magnitude across flow regions	162
8.2	Classifier configurations evaluated for separation detection	169
8.3	Baseline classifier performance on separation detection	169
8.4	Confusion matrix for Random Forest separation classifier	170
8.5	Comparison of feature set sizes for separation detection	172
8.6	Effect of physics constraints on separation classifier performance	172
8.7	Generalisation study: cross-validation results (5 seeds)	173
8.8	Comparison of separation classification and wall shear stress prediction	181
9.1	Model loading backends for OpenFOAM integration.	188
9.2	Wall function methods included in comparative evaluation.	189
9.3	Benchmark test cases for comprehensive evaluation.	190
9.4	Separation and reattachment point predictions for benchmark cases.	193
9.5	Accuracy summary: Mean absolute percentage error (MAPE) for wall quantities.	196
9.6	Memory requirements for ML wall function models.	198
9.7	Computational efficiency summary.	199
9.8	Cross-source evaluation: $R^2_{\tau_w}$ when training and test data sources differ. PINN models show improved robustness to distribution shift.	203
9.9	Performance by flow regime: R^2 scores for attached, near-separation, and separated flows.	203
9.10	Generalization summary: C_f MAPE (%) by distribution shift category.	204
9.11	Robustness summary across evaluation dimensions.	207
9.12	Reynolds analogy consistency: correlation between predicted C_f and St, and ratio compared to theory.	209
9.13	Master comparison table across all evaluation dimensions.	212
9.14	Wall function recommendations by application scenario.	212
10.1	Comparison of physics-informed approaches	226
10.2	Method selection guidelines based on flow conditions and application requirements. MAPE = mean absolute percentage error for wall shear stress in representative test cases.	233

CHAPTER 1

Introduction

The accurate prediction of turbulent flows near solid boundaries remains one of the outstanding challenges in computational fluid dynamics [1–3]. While the Reynolds-averaged Navier-Stokes equations provide a framework for simulating turbulent flows at practical Reynolds numbers [4, 5], their solution requires resolving the steep gradients that develop in the near-wall region—or alternatively, bridging this region with models that capture the essential physics without explicit resolution [6, 7]. These bridging models, known as wall functions, have served engineering practice for over half a century [8, 9]. Yet as computational demands grow more complex—involving flow separation, thermal coupling, and unsteady phenomena—the limitations of classical wall functions become increasingly apparent [10–12]. This thesis develops a machine learning framework to address these limitations [5, 13, 14], creating wall function models that combine the computational efficiency of classical approaches with the flexibility to learn directly from high-fidelity simulation data [2, 15, 16].

1.1 Industrial Motivation: Where Wall Functions Matter

The economic and engineering significance of accurate wall function modeling extends across virtually every industry that relies on fluid flow prediction [17, 18]. The wall shear stress τ_w directly determines drag forces that govern fuel consumption in transportation [19, 20], while the wall heat flux q_w controls thermal management in everything from electronic cooling to nuclear reactor safety [21, 22]. Errors in wall function predictions propagate into the quantities that engineers ultimately care about: drag coefficients, heat transfer rates, pressure drops, and system efficiencies [23, 24]. The breadth of affected industries underscores the fundamental importance of this seemingly specialized topic.

In aerospace and gas turbine applications, modern aircraft wings operate at Reynolds numbers exceeding 10^7 , where even small errors in skin friction prediction translate to significant fuel burn penalties—a 1% error in drag coefficient for a commercial aircraft represents millions of dollars in annual fuel costs across a fleet. The situation is even more critical in gas turbine engines, where blade cooling effectiveness depends on accurate prediction of heat transfer coefficients; underpredicting heat flux leads to blade failure, while overpredicting leads to inefficient use of cooling air that reduces engine efficiency. The automotive industry faces similar challenges, with vehicle manufacturers investing heavily in CFD-based aerodynamic optimization to reduce drag and improve fuel efficiency. For electric vehicles, where range anxiety drives consumer decisions, accurate drag prediction becomes especially important. The complex geometry of vehicle underbodies, wheel wells, and cooling systems creates flow separation and reattachment patterns that challenge classical wall functions, and wind tunnel validation remains essential precisely because CFD predictions using standard wall treatments are often unreliable in these regions.

The built environment presents its own demanding applications. Indoor air quality and thermal comfort depend on accurate prediction of airflow patterns near walls, floors, and ceilings, with buoyancy-driven flows developing near heated or cooled surfaces proving particularly challenging for classical wall functions that assume negligible density variations. Given that buildings account for approximately 40% of global energy consumption, improvements in HVAC system design through better CFD modeling could yield substantial energy savings. Nuclear reactor thermal-hydraulics represents perhaps the most consequential application: safety analysis requires accurate prediction of heat transfer from fuel elements to coolant, and the severe consequences of underpredicting heat flux—potential fuel damage or meltdown—compel regulatory bodies to require substantial safety margins. More accurate wall function models could reduce these margins while maintaining safety, enabling more efficient reactor designs. Similarly, marine and offshore engineering depends on turbulent boundary layer predictions at high Reynolds numbers for ship hull resistance, offshore platform loading, and underwater vehicle performance, with fuel costs typically representing 50–70% of a commercial vessel’s operating expenses.

These diverse applications share a common challenge: the flows of engineering interest almost never satisfy the equilibrium assumptions of classical wall functions. Pressure gradients, flow separation, thermal coupling, and geometric complexity are the norm rather than the exception. The industrial motivation for improved wall function modeling is thus not academic curiosity but practical necessity.

1.2 The Near-Wall Challenge in Turbulent Flow Simulation

Turbulent boundary layers present a fundamental challenge for computational fluid dynamics: the steep velocity gradients that develop near solid surfaces require either extremely fine meshes to resolve directly or simplified models to bridge the near-wall region [25–27]. The physics underlying this challenge—the layered structure of turbulent boundary layers, the transition from viscous-dominated to turbulent-dominated regions, and the universal scaling laws that emerge—is developed in detail in Chapter 2. Here we focus on the practical implications for computational cost and the resulting need for wall function models.

Resolving near-wall gradients directly requires mesh cells with wall-normal dimensions on the order of the viscous length scale, which for high Reynolds number flows translates to meshes with millions or billions of cells concentrated in thin layers near walls [28, 29]. The computational cost scales approximately as $Re^{1.8}$ for wall-resolved Large Eddy Simulation, making direct resolution impractical for most industrial applications. When thermal effects are present, the thermal boundary layer introduces additional thin regions requiring resolution, further increasing computational demands.

1.2.1 The Computational Cost of Wall Resolution

Table 1.1 illustrates the stark contrast between wall-resolved and wall-modeled simulations for a representative internal flow.

Table 1.1: Approximate mesh requirements for wall-resolved versus wall-modeled RANS simulation of channel flow. Wall-resolved meshes require $y_1^+ < 1$; wall-modeled meshes use $y_1^+ \approx 30\text{--}100$.

Reynolds Number	Wall-Resolved	Wall-Modeled	Reduction
$Re = 10^4$	$\sim 10^5$	$\sim 10^4$	$10\times$
$Re = 10^5$	$\sim 10^6$	$\sim 10^5$	$10\times$
$Re = 10^6$	$\sim 10^7$	$\sim 10^5$	$100\times$
$Re = 10^7$	$\sim 10^8$	$\sim 10^6$	$100\times$

For industrial applications at $Re \sim 10^7$, wall-resolved simulations require meshes that push the limits of current supercomputing capabilities. Even with wall functions, a full aircraft simulation may require tens of millions of cells and days of compute time. The mesh reduction enabled by wall functions is not merely convenient—it is often the difference between computationally feasible and infeasible.

However, this computational savings comes at the cost of accuracy. Wall functions approximate the physics of the near-wall region rather than resolving it, and these approximations fail when the underlying assumptions are violated. The challenge addressed by this thesis is to create wall function models that maintain the computational efficiency of classical approaches while substantially improving accuracy under non-equilibrium conditions.

1.2.2 Classical Wall Function Theory

Wall functions offer an alternative to direct resolution: rather than resolving the near-wall region explicitly, the first computational cell is placed in the logarithmic layer (typically at $y^+ \approx 30\text{--}100$), and algebraic relationships derived from similarity theory provide the wall shear stress and heat flux [6]. This approach reduces mesh requirements by factors of 10–100, bringing turbulent flow simulation within reach of routine engineering analysis. The classical wall functions developed by Launder and Spalding in the 1970s assume local equilibrium between turbulence production and dissipation, small pressure gradients, and attached flow—conditions well-satisfied in fully-developed channel flow and flat plate boundary layers [3, 19, 30]. The mathematical framework underlying these wall functions, including the derivation and limitations of the logarithmic law of the wall, is developed comprehensively in Chapter 2.

1.3 Limitations of Classical Wall Functions

The equilibrium assumptions underlying classical wall functions fail precisely where engineering interest is greatest. As flow decelerates under an adverse pressure gradient [31, 32], the boundary layer thickens, the velocity profile deviates from logarithmic form [33, 34], and the assumption of local equilibrium breaks down; the classical wall function overpredicts wall shear stress and cannot capture the onset of separation [10, 35]. The situation becomes more severe in separated regions, where the wall shear stress becomes negative [12, 36, 37], making the friction velocity $u_\tau = \sqrt{|\tau_w|/\rho}$ ill-defined for determining wall-normal distance scaling. More fundamentally, the reversed flow near the wall bears no resemblance to the attached boundary layer profile that the logarithmic law describes [11, 38].

Thermal predictions face additional challenges. Wall functions based on Reynolds analogy [39, 40] assume that the turbulent transport of heat parallels that of momentum, but this assumption fails when buoyancy effects become significant [41] or when the turbulent Prandtl number varies substantially from its commonly-assumed value of 0.85 [42]. Complex geometries introduce yet further complications: three-dimensional flows over curved surfaces, in corners, or around obstacles develop secondary flows and pressure gradient effects that violate the two-dimensional, streamwise-invariant assumptions of classical wall function theory. These limitations manifest as systematic errors in practical CFD simulations—flow separation is predicted too late (or not at all), heat transfer is overpredicted in recirculation zones, and the overall solution accuracy degrades precisely in the regions of greatest engineering interest.

1.3.1 A Motivating Example: The Asymmetric Diffuser

The asymmetric diffuser provides a compelling illustration of classical wall function failure. Consider a channel where one wall remains flat while the opposite wall diverges at an angle, creating an adverse pressure gradient that eventually causes flow separation on the expanding wall. This geometry, used extensively in this thesis, spans conditions from mild deceleration to massive separation depending on the expansion ratio and divergence angle.

Figure 1.1 illustrates the flow physics. In the attached region upstream of separation, classical wall functions perform adequately: the flow approximates equilibrium, and the logarithmic

law provides reasonable estimates of wall shear stress. As the adverse pressure gradient intensifies, however, the boundary layer responds in ways that the equilibrium assumption cannot capture. The velocity profile distorts from logarithmic form, turbulence production exceeds dissipation, and the classical wall function progressively overpredicts τ_w .

At separation, the classical approach fails catastrophically. The wall shear stress passes through zero and becomes negative in the recirculating region. The friction velocity becomes imaginary, breaking the non-dimensionalization that underlies the entire wall function framework. Most CFD codes handle this by switching to an ad-hoc treatment—often linear interpolation through the zero crossing—but such fixes have no physical basis and typically produce poor predictions of the separation and reattachment locations.

The reattachment region presents additional challenges. As the separated flow reattaches, it brings high-momentum fluid toward the wall, creating a recovery profile that differs substantially from the equilibrium logarithmic law. Classical wall functions, designed for attached flow, cannot capture this recovery process.

This single geometry thus encapsulates the full range of wall function challenges: equilibrium flow, adverse pressure gradient effects, separation, reversed flow, and reattachment. A wall function model that performs well across all these conditions would represent a substantial advance over classical approaches.

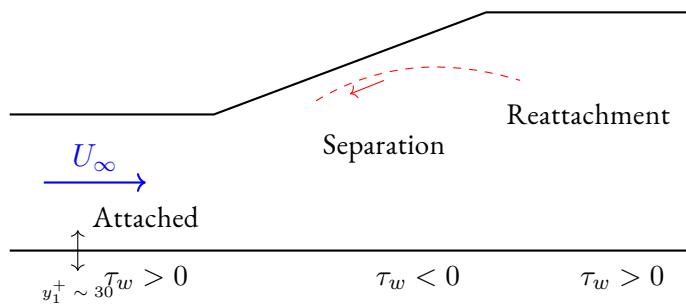


Figure 1.1: Schematic of flow through an asymmetric diffuser, illustrating attached flow, separation, recirculation, and reattachment. Classical wall functions fail in the separation and recirculation regions where $\tau_w \leq 0$.

1.4 The Machine Learning Opportunity

Machine learning offers a fundamentally different approach to wall function modeling. Rather than deriving algebraic relationships from similarity theory and asymptotic analysis, machine learning models learn directly from data. High-fidelity simulations—Direct Numerical Simulation (DNS) or well-resolved Large Eddy Simulation (LES)—provide training data that captures the full complexity of near-wall turbulence, including the departures from equilibrium that classical theory cannot accommodate.

Neural networks, as universal function approximators, can represent the nonlinear relationships between near-wall flow quantities and wall fluxes without the limiting assumptions of classical theory. A network trained on diverse flow conditions can interpolate (and potentially extrapolate) to conditions not explicitly represented in the training data, guided by the underlying patterns in the physics rather than by prescribed functional forms.

However, purely data-driven approaches face their own challenges. Neural networks may learn spurious correlations that happen to work within the training distribution but fail catastrophically outside it. Without physical constraints, predictions may violate conservation laws, produce non-physical negative shear stresses, or extrapolate wildly in regions where training data is sparse. The generalization problem—ensuring that models trained on canonical flows perform well in practical applications—remains the central challenge of machine learning for turbulence modeling.

This thesis addresses these challenges through a unified framework that combines data-driven learning with physics-informed constraints. The key insight is that physics knowledge can be incorporated at multiple levels: in the selection and construction of input features, in the architecture of the neural network itself, and in the loss function used during training. By encoding physical principles at each of these levels, we create models that are both flexible enough to learn from data and constrained enough to respect fundamental physics.

1.5 Three Approaches to Physics-Informed Learning

This thesis develops and compares three complementary methods for incorporating physics knowledge into neural network wall functions. Each method operates at a different level of the machine learning pipeline, and together they provide a comprehensive framework for physics-informed wall function modeling.

1.5.1 Method A: Physics-Encoded Input Features

The first approach encodes physics in the *input representation*. Rather than providing the neural network with raw simulation outputs (velocity components, pressure, temperature at mesh points), we construct non-dimensional features that respect the scaling laws of turbulent boundary layers.

Classical wall function theory teaches us that the relevant variables are wall-scaled quantities: $y^+ = yu_\tau/\nu$, $u^+ = u/u_\tau$, and similar dimensionless groups. These scalings collapse boundary layer data across Reynolds numbers, suggesting that neural networks should receive inputs in these natural coordinates. We extend this idea to construct a library of 58 physics-based features encompassing wall-scaled distances and velocities (y^+ , u^+ , v^+), pressure gradient terms ($p_x^+ = \nu/(\rho u_\tau^3) \cdot \partial p / \partial x$), velocity gradient and curvature quantities, thermal equivalents (T^+ , y_T^+ based on thermal friction velocity), and deformation tensor invariants. Chapter 5 demonstrates that this physics-encoded input representation enables generalization across Reynolds numbers and achieves accuracy comparable to much larger sets of primitive variables.

1.5.2 Method B: Physics-Guided Network Architecture

The second approach investigates what neural networks learn *internally* when trained on primitive inputs. Even without explicit physics encoding in the inputs, neural networks must construct internal representations to solve the wall function prediction problem, raising the question of whether these representations correspond to physically meaningful quantities. Through systematic correlation analysis between hidden layer neuron activations and physics-based features, we find that networks spontaneously discover physics-relevant quantities: neurons in trained networks correlate strongly with y^+ , $\ln(y^+)$, and pressure gradient scalings—precisely

the quantities that classical wall function theory identifies as important. Most remarkably, certain features emerge across network architectures of different sizes; a network with 8 hidden neurons and one with 32 hidden neurons both develop neurons correlated with wall-scaled distance. This *architecture invariance* suggests that these features are fundamental to the learning problem rather than artifacts of particular network configurations. Chapter 6 explores these findings, validating the physics-based input design of Method A while revealing the implicit physics learning capabilities of neural networks.

1.5.3 Method C: Physics-Constrained Loss Functions

The third approach encodes physics in the *training objective*. Standard supervised learning minimizes prediction error on training data, but nothing prevents the network from learning solutions that violate physical conservation laws. Physics-Informed Neural Networks (PINNs) address this limitation by adding physics residuals to the loss function: $\mathcal{L} = \mathcal{L}_{\text{data}} + \lambda_{\text{physics}} \mathcal{L}_{\text{physics}}$. For wall functions, we develop a local stencil-based PINN that evaluates conservation law residuals on the same near-wall stencil used for input features, penalizing violations of streamwise and wall-normal momentum conservation, energy conservation, and mass conservation (incompressibility). Unlike global PINNs that require automatic differentiation over entire computational domains, this local approach maintains computational efficiency while enforcing physics constraints where they matter most: in the near-wall region. Chapter 7 develops this approach and characterizes the trade-off between fitting accuracy and physical consistency.

1.5.4 Complementary Methods

These three methods are not mutually exclusive but rather address physics encoding at different stages of the machine learning pipeline: Method A operates during data preparation through non-dimensional features, Method B influences model design through implicit feature discovery, and Method C shapes the training objective through conservation constraints. A comprehensive wall function model might combine all three—physics-encoded inputs, architectures designed to facilitate physics discovery, and loss functions that enforce conservation. The experimental chapters of this thesis evaluate each method independently to understand their individual contributions before considering combinations.

1.6 Research Questions

This thesis addresses five interconnected research questions that progressively build understanding of physics-informed machine learning for wall functions.

The first and most fundamental question (**RQ1**) asks whether machine learning models can predict wall shear stress and heat flux accurately across diverse flow conditions. This baseline question establishes whether neural networks can learn the wall function mapping at all, and if so, what accuracy is achievable with different input representations and architectures. Building on this foundation, **RQ2** investigates whether physics-based input features improve generalization compared to primitive variables. If networks can learn equally well from primitives and from physics features, the additional effort of computing physics-based inputs may not be justified; conversely, if physics features enable better generalization, they provide a concrete prescription for feature engineering in wall function applications.

The third question (**RQ3**) probes what neural networks learn internally when trained on primitive inputs—do they spontaneously discover physics-based features? This question connects Methods A and B: if networks discover physics features internally, it validates the importance of these features while suggesting that explicit encoding may not be strictly necessary. **RQ4** then examines whether physics-constrained training improves extrapolation to conditions outside the training distribution. Method C sacrifices some in-distribution accuracy for physical consistency, and the critical question is whether this trade-off pays off when models encounter flows not represented in training data.

Finally, **RQ5** addresses the practical question of whether ML wall functions can be deployed in production CFD codes and outperform classical approaches. Academic demonstrations on test sets must ultimately translate to practical utility, and this question determines the engineering viability of the proposed methods.

1.7 Research Objectives

This thesis pursues four interconnected objectives that structure the experimental chapters. The first objective is to develop a systematic methodology for generating training data, since

high-quality training data is the foundation of any machine learning approach. We develop a dual-mesh methodology that pairs fine-mesh (wall-resolved) simulations with coarse-mesh simulations of the same flow: the fine mesh provides ground-truth wall shear stress and heat flux, while the coarse mesh provides the input features that a deployed model would have access to. This approach generates naturally-paired training data across 244 configurations spanning attached and separated flows.

The second objective concerns the design of physics-informed input features. Classical wall functions succeed because they use the right non-dimensional variables— y^+ , u^+ , and similar quantities that collapse boundary layer profiles across Reynolds numbers—and we extend this insight by developing a library of 58 physics-based features derived from the near-wall stencil data. These features encode quantities like local velocity gradients, pressure gradients, curvature terms, and thermal variables in wall-scaled and non-dimensional forms, providing the neural network with inputs that respect the scaling laws of turbulent boundary layers.

The third objective explores physics-guided architectures and training methods. Beyond input features, physical knowledge can be encoded in the network architecture and training procedure; we investigate whether neural networks spontaneously discover physics-based features in their hidden layers, and we develop physics-constrained loss functions that penalize violations of conservation laws during training.

The fourth and final objective is to integrate trained models into production CFD solvers. Academic machine learning research often stops at demonstrating accuracy on test sets, but we complete the engineering loop by implementing trained models as boundary conditions in OpenFOAM, validating performance on benchmark cases beyond the training distribution, and comparing against classical wall functions in both two-dimensional and three-dimensional configurations.

1.8 Thesis Contributions

This thesis makes the following contributions to the field of machine learning for computational fluid dynamics:

1. **A dual-mesh training data generation methodology** that systematically produces paired input-output data for wall function learning across diverse flow conditions. The resulting dataset of 25,485 training samples spans Reynolds numbers 8,000–24,000 and geometric configurations from mild acceleration to strong separation.
2. **A physics-based feature library** comprising 58 non-dimensional quantities derived from the near-wall stencil. Systematic experiments demonstrate that a core subset of 11 physics features achieves accuracy comparable to much larger primitive-variable input sets, while providing interpretability advantages.
3. **Evidence for architecture-invariant feature discovery**, showing that neural networks trained on primitive inputs spontaneously develop hidden layer neurons correlated with physics-based quantities like y^+ , $\ln(y^+)$, and pressure gradient scalings. This finding validates the physics-based input design and suggests that certain features emerge naturally from the learning problem.
4. **A local stencil-based Physics-Informed Neural Network** that enforces conservation laws (momentum, energy, mass) as soft constraints during training. Unlike global PINNs that require automatic differentiation over entire domains, this approach evaluates physics residuals on the same local stencil used for predictions, maintaining computational efficiency while improving physical consistency.
5. **A complete OpenFOAM integration** including C++ boundary condition implementation, Python-to-C++ model export infrastructure, and validation on benchmark cases including backward-facing steps and periodic hills.

1.9 Scope and Limitations

To maintain focus and depth, this thesis adopts specific boundaries on the problems addressed. In terms of flow regimes, we consider incompressible, low-Mach-number flows with Reynolds numbers in the range 8,000–24,000; while the methodology extends in principle to compressible flows and higher Reynolds numbers, validation is limited to the incompressible regime. All simulations use Reynolds-Averaged Navier-Stokes (RANS) equations with the $k-\omega$ SST tur-

bulence model, and although the wall functions developed here could be adapted for Large Eddy Simulation, LES-specific considerations such as subgrid scale modeling near walls are not addressed.

Training data comes from two-dimensional channel, diffuser, and nozzle configurations, with validation on two-dimensional benchmark cases (backward-facing step, periodic hills) and preliminary three-dimensional results; comprehensive three-dimensional validation remains future work. For thermal coupling, we consider passive scalar transport with fixed wall temperatures, leaving conjugate heat transfer, radiative effects, and strongly-coupled thermal-fluid interactions outside the scope. All simulations assume steady-state conditions, with extension to unsteady flows—where temporal history effects may be important—identified as a direction for future research. These limitations define the domain within which the thesis results are validated, and extensions beyond these boundaries are discussed in Chapter 10 as opportunities for future work.

1.10 Thesis Outline

This thesis is organized into nine chapters that progressively develop the physics-informed machine learning framework for wall functions. Following this introduction, Chapter 2 traces the evolution of wall function modeling from Prandtl’s boundary layer theory through modern machine learning approaches, situating the present work within this intellectual history and identifying the specific gaps that the thesis addresses. Chapter 3 describes the computational setup, including the dual-mesh approach for generating paired training data, the OpenFOAM simulation configurations, and the post-processing pipeline that constructs stencil-based inputs from raw simulation outputs.

The core technical contributions appear in Chapters 4–7. Chapter 4 presents the baseline machine learning approach—fully-connected neural networks trained on primitive flow variables to predict wall shear stress and heat flux—establishing the achievable accuracy and identifying limitations that motivate the physics-informed extensions. Chapter 5 develops the library of 58 physics-based input features and systematically compares performance against primitive-variable inputs (Method A), identifying a core set of features that balance accuracy, interpretability, and

computational cost. Chapter 6 investigates what neural networks learn internally when trained on primitive inputs (Method B), demonstrating through correlation analysis that networks discover wall-scaled quantities even without explicit physics encoding. Chapter 7 develops and evaluates the local stencil-based PINN approach (Method C), incorporating conservation law residuals into the training loss function and characterizing the trade-off between fitting accuracy and physical consistency.

The final chapters address practical deployment and conclusions. Chapter 8 tackles the distinct challenge of detecting separated regions where classical wall functions fail most severely, developing classifiers that distinguish attached from separated flow based on stencil inputs. Chapter 9 presents the production implementation, including C++ boundary condition code, model export utilities, and comprehensive validation on benchmark geometries beyond the training distribution. Chapter 10 summarizes the contributions, discusses limitations, and outlines directions for future research including extension to higher Reynolds numbers, three-dimensional flows, and unsteady applications.

1.11 Chapter Summary

Turbulent flow simulation requires accurate treatment of the near-wall region where steep gradients develop. Classical wall functions, while computationally efficient, fail under conditions of adverse pressure gradients, flow separation, and strong thermal coupling—precisely the conditions of greatest engineering interest. The economic implications span industries from aerospace to building ventilation, motivating sustained research into improved wall function models.

Machine learning offers the potential to learn wall function models directly from high-fidelity data, but purely data-driven approaches risk learning spurious correlations that fail outside the training distribution. This thesis develops a physics-informed machine learning framework that combines the flexibility of neural networks with the constraints of physical law. Three complementary methods encode physics knowledge at different stages: in the input features (Method A), in the network architecture through implicit discovery (Method B), and in the training loss function (Method C).

The resulting models are integrated into OpenFOAM for practical CFD applications, demon-

strating improved performance over classical approaches across a range of benchmark configurations. The following chapters develop this framework systematically, from data generation through production deployment, contributing both methodological advances and practical tools to the field of machine learning for computational fluid dynamics.

CHAPTER 2

Literature Review

This chapter traces the evolution of wall function modeling from its classical foundations through the modern era of machine learning, situating the present thesis within this historical trajectory. Rather than cataloging papers in isolation, we follow the intellectual thread that connects Prandtl’s boundary layer theory to contemporary physics-informed neural networks, revealing how each generation of researchers built upon—and was constrained by—the work of their predecessors. The chapter begins with the fundamental physics of turbulent boundary layers, develops the mathematical framework underlying all wall function approaches, examines how industry has implemented these ideas in practical computational fluid dynamics codes, and finally surveys the machine learning revolution that motivates this thesis.

2.1 The Physics of Turbulent Boundary Layers

Understanding wall functions requires deep familiarity with the physics they attempt to model. The near-wall region of turbulent flows exhibits complex multi-scale phenomena that have occupied fluid dynamicists for over a century [1, 2]. This section develops the theoretical foundations from Prandtl’s original insights through the modern understanding of near-wall turbulence structure, establishing the physical basis upon which all subsequent wall modeling efforts rest.

2.1.1 Prandtl’s Boundary Layer Concept

The modern understanding of wall-bounded flows began with Ludwig Prandtl’s seminal 1904 paper, which resolved a fundamental paradox that had troubled fluid mechanics for decades [25]. D’Alembert’s paradox demonstrated that potential flow theory predicted zero drag on bodies

moving through inviscid fluids—a prediction manifestly contradicted by everyday experience. Prandtl’s profound insight was that viscous effects, while negligible over most of the flow domain, become dominant in a thin layer adjacent to solid surfaces where the no-slip condition must be satisfied. Within this boundary layer, the velocity changes rapidly from zero at the wall to the free-stream value over a distance that may be orders of magnitude smaller than the characteristic length of the body.

This scale separation enables systematic simplification of the Navier-Stokes equations [43, 44]. For steady, two-dimensional flow over a flat plate aligned with the streamwise direction, the boundary layer equations reduce to a coupled system where the streamwise momentum balance involves convection by both velocity components, the imposed pressure gradient from the external flow, and viscous diffusion in the wall-normal direction only. The key simplification arises from recognizing that wall-normal gradients are much larger than streamwise gradients when the boundary layer thickness is small compared to the streamwise development length. This enables neglecting the streamwise viscous diffusion term while retaining the dominant wall-normal diffusion, yielding equations that are parabolic in the streamwise direction and can be marched forward from initial conditions without the elliptic complications of the full Navier-Stokes system.

The boundary layer equations take the form

$$u \frac{\partial u}{\partial x} + v \frac{\partial u}{\partial y} = -\frac{1}{\rho} \frac{\partial p}{\partial x} + \nu \frac{\partial^2 u}{\partial y^2} \quad (2.1)$$

$$\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} = 0 \quad (2.2)$$

where the pressure gradient is imposed by the external inviscid flow and varies along the surface. For a flat plate at zero incidence with uniform free-stream velocity, this gradient vanishes, yielding the canonical zero-pressure-gradient boundary layer that serves as the foundation for wall function development [29, 45]. Non-zero pressure gradients arise when the external flow accelerates or decelerates, profoundly affecting boundary layer behavior in ways that remain challenging to model even today [10, 34].

2.1.2 The Blasius Solution and Laminar Boundary Layers

For the zero-pressure-gradient laminar boundary layer, Blasius obtained a similarity solution in 1908 that remains a cornerstone of boundary layer theory [28]. By introducing a stream function and recognizing that the velocity profile maintains a self-similar shape when expressed in terms of an appropriately scaled wall-normal coordinate, Blasius reduced the partial differential equations to an ordinary differential equation. The similarity variable combines the wall distance with the local Reynolds number based on streamwise position, and the resulting third-order nonlinear ordinary differential equation, while not admitting closed-form solution, can be solved numerically to arbitrary precision [29].

The Blasius solution yields several important results that inform subsequent turbulent boundary layer analysis. The wall shear stress decreases along the plate as the inverse square root of downstream distance, giving a skin friction coefficient that scales as the inverse square root of the local Reynolds number. The boundary layer thickness grows as the square root of streamwise distance, a consequence of the diffusive spreading of vorticity from the wall. These laminar scaling relationships differ fundamentally from turbulent boundary layers, where enhanced mixing by turbulent fluctuations dramatically increases momentum transfer toward the wall [26, 27].

2.1.3 Transition to Turbulence

Laminar boundary layers become unstable at sufficiently high Reynolds numbers through mechanisms first elucidated by Tollmien and Schlichting in the 1930s [20, 46]. Linear stability analysis identifies critical conditions beyond which small disturbances amplify exponentially rather than decaying. These Tollmien-Schlichting waves, essentially two-dimensional oscillations of the boundary layer, grow as they convect downstream, interact nonlinearly, and eventually trigger breakdown to fully turbulent flow through a complex cascade involving three-dimensional secondary instabilities, spike formation, and the generation of turbulent spots that spread laterally and merge.

In practical applications, transition occurs over a range of Reynolds numbers depending on the disturbance environment [24, 47]. Free-stream turbulence intensity, surface roughness,

acoustic disturbances, and pressure gradients all influence the transition process. For smooth flat plates with low free-stream turbulence, transition typically begins around Reynolds numbers of several hundred thousand based on streamwise distance and completes within a factor of ten in Reynolds number. Higher disturbance levels trigger bypass transition at substantially lower Reynolds numbers through mechanisms that circumvent the classical instability route entirely [1]. The transitional region presents particular challenges for wall function modeling because neither laminar nor fully turbulent assumptions apply, and the flow exhibits intermittent turbulent spots within a laminar background. This thesis focuses on fully turbulent boundary layers, though the machine learning methods developed could potentially extend to transitional flows with appropriate training data.

2.1.4 Reynolds Decomposition and the Closure Problem

The irregular, chaotic nature of turbulent flows motivated Reynolds to propose decomposing instantaneous velocities into mean and fluctuating components [4]. Time-averaging the Navier-Stokes equations with this decomposition yields the Reynolds-Averaged Navier-Stokes equations, which govern the evolution of mean quantities [2, 5]. However, the averaging process introduces additional terms—the Reynolds stresses—representing momentum flux due to turbulent fluctuations. These six independent stress components appear as unknowns in the mean momentum equations, yet no additional equations emerge from the averaging process. This closure problem means the Reynolds-averaged equations cannot be solved without supplementary information relating Reynolds stresses to mean flow quantities [48, 49].

The most common closure approach invokes the Boussinesq hypothesis, which assumes Reynolds stresses are proportional to mean strain rates through a turbulent eddy viscosity, analogous to the relationship between viscous stress and strain rate in Newtonian fluids [1, 15]. This reduces the closure problem to determining a single scalar field, the eddy viscosity, which turbulence models provide through additional transport equations for quantities like turbulent kinetic energy and its dissipation rate. The Boussinesq hypothesis assumes Reynolds stresses respond instantaneously to local strain rates, ignoring history effects and the inherent anisotropy of turbulence [50, 51]. These limitations become severe in flows with strong stream-

line curvature, rapid strain, or separation—precisely the conditions where wall functions also fail, motivating the more sophisticated approaches developed in this thesis.

2.1.5 The Structure of Turbulent Boundary Layers

Turbulent boundary layers exhibit a characteristic layered structure that emerges from the competition between viscous and turbulent stresses at different distances from the wall [11, 26]. This structure, revealed through decades of experimental investigation and more recently through direct numerical simulation, provides the physical basis for wall function modeling. Near the wall, viscous stresses dominate because turbulent fluctuations are damped by the no-slip condition. Far from the wall, turbulent stresses dominate as energetic eddies transport momentum efficiently. Between these limiting behaviors lies a transitional region where both mechanisms contribute comparably [20, 27].

The relevant length scale near the wall is the viscous length, defined as the ratio of kinematic viscosity to friction velocity. Normalizing wall distance by this viscous length defines the wall unit, typically denoted y^+ . Similarly normalizing the mean velocity by friction velocity defines u^+ . Within the inner layer, comprising roughly the innermost ten to twenty percent of the boundary layer thickness, the velocity profile depends only on these wall-scaled variables and is independent of the outer flow conditions [3, 30]. This universality, known as the law of the wall, represents one of the most robust results in turbulence and provides the theoretical foundation for wall function approaches.

The inner layer subdivides into three distinct regions with different physical character [22, 52]. Immediately adjacent to the wall, in what is termed the viscous sublayer, turbulent fluctuations are strongly suppressed and viscous stress dominates completely. The velocity profile becomes linear, with $u^+ = y^+$, a result confirmed by countless experiments and simulations to remarkable precision. This linear region extends to roughly $y^+ = 5$, beyond which turbulent stresses begin contributing significantly.

Between the viscous sublayer and the fully turbulent region lies the buffer layer, extending from approximately $y^+ = 5$ to $y^+ = 30$ [26, 46]. In this transitional zone, both viscous and turbulent stresses contribute comparably to momentum transfer, and the velocity profile curves

smoothly between the linear viscous behavior and the logarithmic profile that emerges at larger wall distances. The buffer layer contains the peak of turbulent kinetic energy production and hosts the most intense turbulent events, including the ejection of low-speed fluid away from the wall and the sweep of high-speed fluid toward it. No simple analytical expression describes this region accurately, though various empirical formulas have been proposed for engineering calculations [7].

Beyond the buffer layer, for $y^+ > 30$ but still within roughly twenty percent of the boundary layer thickness, turbulent stresses dominate completely and a logarithmic velocity profile emerges. This logarithmic law of the wall, first proposed by von Kármán based on dimensional arguments [9], takes the form $u^+ = (1/\kappa) \ln(y^+) + B$, where $\kappa \approx 0.41$ is the von Kármán constant and $B \approx 5.0$ is an additive constant determined by matching through the buffer layer. The logarithmic profile can be derived from several independent lines of reasoning, lending confidence to its universality [29].

2.1.6 Derivations of the Logarithmic Law

Dimensional analysis provides the most direct route to the logarithmic profile [43, 45]. In the fully turbulent region where viscosity no longer directly influences the local dynamics, the velocity gradient can depend only on wall distance, wall shear stress, and fluid density. Dimensional consistency then requires the velocity gradient to scale inversely with wall distance, and integration yields the logarithmic profile with an undetermined multiplicative constant that experiments identify as the inverse of the von Kármán constant.

Prandtl's mixing length hypothesis offers a physical interpretation of this scaling [17, 53]. Turbulent eddies transport momentum over a characteristic mixing length before losing their identity through mixing with surrounding fluid. Near the wall, the largest eddies that can exist are constrained by the wall distance itself, suggesting the mixing length should be proportional to y . This assumption, combined with the definition of turbulent stress through the mixing length formulation, yields an eddy viscosity proportional to wall distance times the velocity gradient. When the total stress is constant and equal to the wall shear stress—a good approximation in the inner layer—solving for the velocity gradient and integrating reproduces

the logarithmic law.

Millikan's overlap argument provides yet another derivation without assuming anything about the turbulent stress mechanism [54]. In the inner layer, the velocity profile must depend only on wall-scaled variables. In the outer layer, the velocity defect from the free-stream value must depend only on outer-scaled variables involving the boundary layer thickness. In the overlap region where both descriptions must simultaneously apply, the only functional form consistent with both scalings is logarithmic. This elegant argument demonstrates that the log law is not merely an empirical fit but a mathematical consequence of the two-layer structure of the boundary layer.

2.1.7 The Outer Layer and Wake Function

Beyond the logarithmic region, the velocity continues increasing toward the free-stream value through what is termed the wake region or defect layer [11, 33]. Here the velocity defect from the free-stream value scales with the friction velocity and the boundary layer thickness rather than with viscous quantities. Coles proposed combining the inner logarithmic profile with an additive wake function to describe the entire boundary layer. The wake function accounts for the departure from the log law in the outer region due to intermittency effects at the boundary layer edge and the influence of the outer boundary conditions.

The wake parameter quantifies the strength of the wake component and varies with pressure gradient [31, 34]. For zero-pressure-gradient boundary layers, the wake parameter takes a value around 0.55, corresponding to a modest overshoot above the logarithmic extrapolation in the outer region. Adverse pressure gradients increase the wake parameter, strengthening the deviation from the log law, while favorable pressure gradients decrease it. This pressure-gradient sensitivity of the outer layer contrasts with the relative universality of the inner layer and illustrates why wall functions based solely on the log law struggle under non-equilibrium conditions.

2.1.8 Turbulent Stress Distributions

The Reynolds stress tensor exhibits distinct behavior across the boundary layer structure, behavior that must be captured or circumvented by wall function approaches [3, 30]. The landmark direct numerical simulation of Moser, Kim, and Mansour provided definitive data on these distributions in turbulent channel flow, later extended to higher Reynolds numbers by Lee and Moser [20, 26].

The streamwise velocity fluctuations peak in the buffer layer around $y^+ \approx 15$, reaching root-mean-square values roughly 2.5 to 3 times the friction velocity [11, 53]. This intense streamwise fluctuation activity reflects the burst-sweep cycle that dominates near-wall turbulence, with low-speed fluid ejected away from the wall and high-speed fluid swept toward it. The wall-normal fluctuations, in contrast, are strongly suppressed near the wall by the kinematic blocking effect of the surface and increase more gradually, peaking in the outer part of the logarithmic layer. The Reynolds shear stress, the only component contributing to mean momentum transfer in parallel shear flows, varies nearly linearly across the inner layer from zero at the wall to a maximum in the outer layer, reflecting the handoff from viscous to turbulent momentum transport.

Understanding these stress distributions is essential for wall function development [22, 52]. Standard wall functions assume local equilibrium between production and dissipation of turbulent kinetic energy, which holds approximately in the logarithmic layer where the production rate nearly balances the dissipation rate. In the buffer layer, production exceeds dissipation as turbulent kinetic energy is generated before being transported and dissipated elsewhere. Under non-equilibrium conditions such as adverse pressure gradients or flow separation, the equilibrium assumption fails throughout the boundary layer, invalidating the fundamental premise of classical wall functions [38, 50].

2.2 The Mathematical Framework of Wall Functions

The theoretical understanding developed above must be translated into practical computational tools. This section traces the evolution of wall function formulations from early analytical

approaches through modern enhanced treatments implemented in commercial software [6, 7].

2.2.1 The Launder-Spalding Wall Function

Launder and Spalding formalized the use of the logarithmic law as a boundary condition for Reynolds-averaged simulations in their influential 1974 paper [6]. Their approach places the first computational cell center in the logarithmic region, typically at y^+ values between 30 and 300, and relates the cell-center velocity to wall shear stress algebraically through the log law. This eliminates the need to resolve the steep gradients in the viscous sublayer and buffer layer, reducing mesh requirements by an order of magnitude or more while maintaining accuracy for flows where the logarithmic profile applies [23, 24].

The implementation requires care because the wall shear stress appears on both sides of the log law equation—explicitly in the relationship and implicitly through the wall-scaled variables [55]. Launder and Spalding resolved this by using the cell-center turbulent kinetic energy rather than the friction velocity to define the wall units, yielding an explicit formula for wall shear stress given the cell-center velocity and turbulence quantities. The turbulence boundary conditions similarly employ equilibrium assumptions, specifying that the dissipation rate in the wall-adjacent cell equals the production rate according to local equilibrium.

The thermal wall function follows analogously, using Reynolds analogy to relate the turbulent heat flux to the momentum flux through a turbulent Prandtl number [21, 22]. The temperature profile in wall units exhibits a linear conduction-dominated region near the wall analogous to the viscous sublayer, followed by a logarithmic region where turbulent transport dominates. The transition between these regions depends on the molecular Prandtl number, with high-Prandtl-number fluids having extremely thin conduction sublayers and low-Prandtl-number fluids like liquid metals having conduction extending well into the turbulent region.

2.2.2 Assumptions and Their Limitations

The Launder-Spalding wall function rests on several assumptions whose violation causes the approach to fail [38, 51]. The assumption that the first cell lies within the logarithmic region means that if the mesh is too coarse the cell extends into the wake region where the log law

overpredicts velocity, and if too fine the cell falls into the buffer layer where it underpredicts. The local equilibrium assumption means the approach cannot capture the history effects that characterize boundary layers under pressure gradients or after perturbations. The constant-stress assumption requires the wall-adjacent cell to be thin enough that stress variation across it is negligible.

The most severe failures occur in separated flows where the wall shear stress becomes negative [11, 23, 34]. The friction velocity, defined as the square root of the wall shear stress magnitude divided by density, remains real, but the entire scaling framework loses physical meaning when the near-wall velocity opposes the outer flow direction. The log law, derived for attached flows where velocity increases monotonically from wall to free-stream, has no relevance to the reversed-flow profiles near separation. Most computational codes handle separated regions by ad-hoc fixes such as limiting the wall shear stress magnitude or switching to alternative formulations, but these patches have no physical basis and typically produce poor predictions of separation location, recirculation zone extent, and reattachment [10, 35].

Strong adverse pressure gradients, even without separation, cause the equilibrium assumption to fail [15, 31]. As the flow decelerates, the boundary layer thickens, the velocity profile distorts from logarithmic form, and the production of turbulent kinetic energy exceeds its dissipation rate. The classical wall function, seeing only the cell-center conditions and knowing nothing of the upstream history, overpredicts wall shear stress because it assumes an equilibrium profile that the actual flow does not possess.

2.2.3 Enhanced Wall Treatments

Recognition of these limitations motivated development of enhanced wall treatments that attempt to function across a wider range of conditions [8, 23]. Two-layer models resolve the viscous sublayer with a fine mesh while using a simplified turbulence model in the near-wall region. These models can handle lower wall-unit mesh spacing but still assume quasi-equilibrium and struggle with separated flows.

Enhanced wall functions blend between viscous sublayer and logarithmic formulations based on local wall-unit spacing, allowing the first cell to be placed anywhere from the sublayer to the

log layer without dramatic loss of accuracy [7, 55]. Spalding's unified wall law provides a single implicit equation valid across the entire inner layer, though it cannot be solved explicitly for velocity given wall distance. Modern commercial codes implement various blending functions that transition smoothly between the linear and logarithmic regimes.

Scalable wall functions address the problem of meshes refined beyond the buffer layer by defining a virtual wall location at the intersection of the linear and logarithmic profiles [24]. This prevents the wall function from predicting unphysically low wall shear stress on fine meshes, but it sacrifices the potential accuracy gains that fine mesh resolution near the wall might provide.

Non-equilibrium wall functions attempt to account for pressure gradient effects by modifying the log-law constants or by solving simplified boundary layer equations within the wall function formulation [10, 38]. The Launder-Shima approach adds a pressure gradient correction to the log-law intercept, partially compensating for the profile distortion under adverse gradients. More sophisticated analytical wall functions solve the log-layer momentum equation including convection and pressure gradient effects, but even these struggle with the extreme non-equilibrium of separated flows.

2.3 Thermal Boundary Layers and Heat Transfer

Wall functions must predict not only momentum transfer through wall shear stress but also heat transfer through wall heat flux for thermal applications [21, 22, 42]. The thermal boundary layer exhibits analogous layered structure to the velocity boundary layer but with important differences that complicate the extension of momentum wall functions to heat transfer.

2.3.1 The Reynolds Analogy

Reynolds observed in 1874 that the mechanisms of momentum and heat transfer in turbulent flow share fundamental similarities, leading to the classical Reynolds analogy relating skin friction coefficient to Stanton number [21, 40]. For fluids with Prandtl number equal to unity, where molecular diffusivities of momentum and heat are identical, the analogy predicts exact equality between half the skin friction coefficient and the Stanton number. For fluids

with Prandtl numbers differing from unity, modifications account for the different sublayer thicknesses, with the Colburn analogy introducing a Prandtl number correction factor.

These analogies work reasonably well for attached boundary layers with moderate temperature differences but fail when pressure gradients cause the velocity and thermal boundary layers to develop differently [22, 42], when strong temperature differences cause property variations that invalidate the constant-property assumptions, or when buoyancy effects couple the momentum and energy equations. The analogy has no physical basis whatsoever in separated regions where the reversed near-wall flow bears no relationship to the heat transfer at the wall [41, 56].

2.3.2 Thermal Law of the Wall

By analogy with the velocity profile, the temperature field in the inner layer follows a universal law when expressed in appropriate wall units [21, 39]. Defining a friction temperature through the wall heat flux, analogous to the friction velocity definition, allows expressing the temperature difference from the wall in dimensionless form. In the conduction-dominated sublayer immediately adjacent to the wall, the temperature profile becomes linear with slope equal to the molecular Prandtl number. In the turbulent region, a logarithmic profile emerges with slope related to the turbulent Prandtl number.

Kader developed a widely-used formula that blends smoothly between these regimes while capturing the Prandtl number dependence of the thermal buffer layer [39]. The blending function differs from that for velocity because the thermal and momentum buffer layers have different structures depending on the molecular Prandtl number [41]. For high-Prandtl-number fluids like oils, the thermal sublayer is extremely thin compared to the viscous sublayer, while for low-Prandtl-number fluids like liquid metals, conduction penetrates far into the turbulent region.

2.3.3 Dissimilarity Between Momentum and Heat Transfer

A fundamental limitation of Reynolds-analogy-based thermal wall functions is the assumption that momentum and heat transfer follow identical mechanisms [40, 42]. Several sources of dissimilarity exist even in attached boundary layers. The turbulent Prandtl number, relating

turbulent momentum and heat diffusivities, is not constant but varies from roughly unity in the sublayer to 0.85 in the logarithmic layer to perhaps 0.5 in the outer region. Pressure gradients affect velocity and temperature profiles differently, and buoyancy introduces coupling between the fields that the analogy cannot capture [41].

These considerations motivate the approach of this thesis, which predicts wall shear stress and wall heat flux simultaneously from unified models trained on data that naturally captures the dissimilarity between momentum and heat transfer [21, 22]. Rather than assuming the two are linked by a constant factor, the neural network learns whatever relationship the training data exhibits.

2.4 Industrial Computational Fluid Dynamics Practice

The theoretical developments traced above must ultimately serve practical engineering analysis [17, 18]. This section examines how wall functions are implemented in commercial codes and applied in industrial workflows, identifying the gap between academic research and production practice that motivates practical deployability as a key objective of this thesis.

2.4.1 Commercial Implementations

Major commercial computational fluid dynamics codes implement wall functions with varying degrees of sophistication [23, 24]. ANSYS Fluent offers multiple options including standard wall functions, scalable wall functions, enhanced wall treatment, and non-equilibrium wall functions. The enhanced wall treatment automatically blends between viscous sublayer resolution and log-law formulations based on local mesh resolution, providing flexibility at the cost of potential inconsistency between regions. STAR-CCM+ implements all- y^+ wall treatment that switches between low-Reynolds and high-Reynolds formulations automatically, emphasizing robustness for industrial users who may not optimize mesh resolution for each application.

OpenFOAM, the open-source platform used throughout this thesis, provides wall function boundary conditions implementing the Launder-Spalding formulation along with automatic wall treatment options using Spalding's unified law [24, 57]. The open-source nature allows direct implementation of new wall function approaches, making it the natural choice for devel-

oping and validating the machine learning methods of this thesis.

2.4.2 Industrial Practice and Certification

Industrial computational fluid dynamics practice involves compromises between accuracy, computational cost, and robustness [17, 58]. Mesh guidelines typically recommend wall-unit spacing in the range 30 to 300 for standard wall functions and near unity for wall-resolved simulations, but achieving these targets uniformly across complex geometries is often impractical. Real industrial meshes frequently have non-uniform wall spacing that places some regions in the buffer layer where neither wall function formulation is optimal.

High-stakes industries impose stringent validation requirements that often exceed wall function capabilities [10, 41]. Aerospace certification may require wall-resolved simulations for flight-critical components because wall function predictions are deemed insufficiently reliable. Nuclear safety analysis requires validated methodologies with quantified uncertainty margins that wall function limitations can expand substantially [59–61]. These requirements highlight the gap between wall function capabilities and industrial needs that motivates continued research including the machine learning approaches developed here.

2.4.3 The Academia-Industry Gap

A persistent gap exists between academic turbulence modeling research and industrial practice [1, 58]. Academic developments often include many adjustable parameters tuned for specific test cases, whereas industrial users need robust defaults that work across diverse applications without case-by-case adjustment. Academic publications report successful cases, but industrial simulations must handle arbitrary geometries and flow conditions without divergence or unphysical results [15, 51]. Advanced models that improve accuracy modestly while substantially increasing computational cost are rarely adopted industrially, where throughput considerations often override accuracy concerns.

This thesis addresses several of these gaps. By implementing models in OpenFOAM and providing complete documentation, we lower barriers to industrial adoption [24, 57]. By training on diverse configurations and validating on geometries outside the training distribution,

we demonstrate the robustness industrial users require. By maintaining computational efficiency comparable to standard wall functions, we avoid the cost barriers that have limited adoption of more sophisticated approaches.

2.5 Pressure Gradient Effects

Pressure gradients profoundly affect turbulent boundary layer behavior and represent a key challenge for wall function modeling [10, 31, 34]. The Clauser parameter, defined as the product of displacement thickness and pressure gradient divided by wall shear stress, quantifies pressure gradient strength relative to wall friction. Zero values correspond to zero-pressure-gradient boundary layers, negative values to favorable (accelerating) gradients, and positive values to adverse (decelerating) gradients. Self-similar equilibrium boundary layers can exist at constant Clauser parameter values, maintaining fixed shape while growing downstream, but most practical flows involve spatially-varying pressure gradients that prevent self-similarity [11, 38].

Under adverse pressure gradients, the boundary layer thickens as retarded near-wall fluid decelerates further, the velocity profile distorts from logarithmic form with increased wake component, turbulence intensifies due to enhanced production, and wall shear stress decreases toward zero [10, 35]. If the adverse gradient is sufficiently strong, flow separation occurs when wall shear stress vanishes and the near-wall fluid reverses direction. The approach to separation involves highly non-equilibrium dynamics that classical wall functions cannot capture.

Stratford developed a criterion for predicting turbulent boundary layer separation based on the pressure rise and streamwise development length [32]. While useful for preliminary design, such criteria cannot predict the detailed behavior through and after separation [11, 37]. The machine learning approach of this thesis provides predictions throughout the separation and reattachment process by learning from training data that includes these regimes.

2.6 The Machine Learning Revolution

Having established the classical physics and computational frameworks, we turn to the modern machine learning approaches that motivate this thesis. The emergence of data-driven turbulence

modeling represents not merely an incremental improvement but a fundamental shift in how the field approaches the closure problem and the limitations of classical models [1, 2, 5].

2.6.1 Data-Driven Turbulence Modeling

The modern era of machine learning in computational fluid dynamics emerged from advances in deep learning for image recognition and natural language processing that demonstrated neural networks could learn complex nonlinear mappings from large datasets [5, 13, 48]. The seminal work of Ling, Kurzawski, and Templeton marked a watershed moment by demonstrating that deep neural networks could predict Reynolds stress anisotropy from mean flow features, capturing physics beyond the Boussinesq hypothesis that had constrained turbulence modeling for decades.

Ling’s approach incorporated physics awareness through its input construction [49, 50]. Rather than providing raw flow quantities, she used invariants of the strain rate and rotation tensors as inputs, encoding Galilean invariance—the requirement that turbulence statistics cannot depend on observer frame of reference—directly into the network architecture. The outputs similarly respected tensorial structure through a basis expansion. This physics-informed approach yielded better generalization than purely black-box models, establishing a paradigm that subsequent work has extended [2, 51].

Wang, Wu, and Xiao developed comprehensive frameworks combining machine learning with uncertainty quantification, emphasizing that neural networks trained on high-fidelity data encode implicit knowledge of turbulent dynamics but that this knowledge may not transfer reliably to flows outside the training distribution [5, 15, 58]. The generalization problem—ensuring models trained on canonical configurations perform well in practical applications—emerged as the central challenge for the field and remains largely unsolved [38, 50, 51].

Subsequent research has explored numerous variations on data-driven turbulence modeling [1, 2]. Sparse symbolic regression discovers algebraic Reynolds stress models automatically from data [37, 49]. Multi-agent reinforcement learning optimizes turbulence model parameters through automated experimentation [48]. Ensemble methods quantify uncertainty in learned closures [59, 60, 62]. Each approach offers unique advantages, but the fundamental challenge

of generalizing from training data to novel flows persists across methodologies [15, 38].

2.6.2 Physics-Informed Neural Networks

Raissi, Perdikaris, and Karniadakis introduced Physics-Informed Neural Networks (PINNs), addressing the fundamental limitation that traditional supervised learning requires abundant labeled training data [14, 43, 44]. Their approach incorporates governing equations as soft constraints during training by adding physics residuals to the loss function. If the network learns to satisfy conservation laws throughout the domain, its predictions should remain physically consistent even in regions without training data [29, 34, 57].

The physics loss evaluates partial differential equation residuals at collocation points using automatic differentiation through the network, penalizing departures from the governing equations [24, 43, 45]. For incompressible flows, this includes continuity and momentum residuals throughout the domain. The balance between data fitting and physics constraint satisfaction, governed by a hyperparameter weighting the physics loss, requires careful tuning that varies across problems [56, 63, 64].

Physics-informed networks have been applied to numerous fluid dynamics problems [24, 56, 65, 66]. Wang and colleagues applied PINNs to turbulence modeling with tensor basis constraints [5, 67]. Applications span biomedical flows [56, 66], combustion [63, 68], and free shear flows [24]. OpenFOAM integration has been demonstrated for practical CFD workflows [57].

Physics-informed networks face substantial challenges for turbulence applications [15, 63, 69]. The Reynolds-averaged equations are not closed, requiring models for Reynolds stresses that cannot be derived from first principles. Neural networks exhibit spectral bias toward learning low-frequency features before high-frequency ones, making multi-scale turbulence difficult to capture [10, 64]. The physics loss can create ill-conditioned optimization landscapes near boundaries where multiple constraints interact [44, 63].

Chapter 7 of this thesis develops a local stencil-based variant that addresses these limitations for wall function applications. Rather than enforcing global conservation over entire domains, we evaluate physics residuals on the same local stencil used for input features, maintaining

computational efficiency while focusing physical constraints where they matter for wall shear and heat flux prediction.

2.6.3 Deep Learning Architectures for Turbulence

Beyond physics-informed approaches, various deep learning architectures have been applied to turbulence problems [20, 26, 27, 70, 71]. Convolutional neural networks exploit spatial structure in flow fields [26, 53, 72–74], while recurrent networks capture temporal dependencies in unsteady flows [20, 41, 75, 76]. Attention mechanisms allow networks to focus on relevant regions [41, 46, 77, 78], and transformer architectures originally developed for natural language processing show promise for sequence modeling in turbulence [11, 53, 79, 80].

Graph neural networks offer particular promise for CFD applications due to their ability to handle unstructured meshes naturally [55, 81–84]. By representing mesh cells as nodes and connectivity as edges, graph networks can process arbitrary mesh topologies without the structured grid requirements of convolutional networks [85, 86]. This flexibility enables application to complex geometries without the mesh interpolation that convolutional approaches require [66, 81, 87, 88].

Super-resolution techniques reconstruct fine-scale turbulent structures from coarse representations [11, 53, 72, 89, 90]. Neural operators learn mappings between function spaces [66, 74, 91, 92], potentially enabling generalization across geometries and operating conditions [44, 93, 94]. Diffusion models generate realistic turbulent fields through iterative denoising [11, 95–97]. Each architecture class offers distinct advantages for specific aspects of the turbulence modeling challenge [98, 99].

2.6.4 Machine Learning Wall Functions

The specific application of machine learning to wall function modeling represents the confluence of classical wall-bounded turbulence theory, data-driven Reynolds stress modeling, and physics-informed learning [26, 38, 52, 100, 101]. Milano and Koumoutsakos pioneered this direction in 2002, training neural networks to predict wall shear stress from outer flow quantities for large eddy simulation [16, 102, 103]. Limited by contemporary computational resources, their

work demonstrated proof of concept but could not address generalization challenges [104, 105].

Recent work has demonstrated that neural networks can handle separated flows over periodic hills where standard wall functions fail entirely [11, 37, 38, 106, 107], that training on canonical building-block flows can construct composite models for more complex geometries [15, 51, 108, 109], and that physics-informed variants improve consistency at the cost of some fitting accuracy [63, 64, 110, 111]. Deep neural networks have been applied specifically to near-wall turbulence with consideration of wall-distance effects [26, 27, 38, 47, 112]. Data-driven approaches for separated flows have shown particular promise [10, 35, 37, 113, 114].

Yet critical questions remain [15, 51]. The choice between primitive variables and physics-based features as network inputs varies across studies without systematic justification—a gap Chapter 5 addresses. Thermal wall functions receive disproportionately little attention despite engineering importance [21, 22, 42]—a gap this thesis addresses by predicting both momentum and thermal quantities jointly. The generalization from training geometries to practical applications remains the central unsolved challenge [15, 38].

2.6.5 Uncertainty Quantification in Machine Learning CFD

Reliable deployment of machine learning models requires quantifying prediction uncertainty [59–62, 115, 116]. Bayesian approaches provide principled uncertainty estimates through posterior distributions over network weights [58, 117–119]. Deep ensembles use disagreement among independently trained networks as an uncertainty proxy [59, 62, 120, 121]. Dropout during inference approximates Bayesian uncertainty without the computational cost of full posterior inference [15, 122, 123].

Physics-constrained random forests provide uncertainty quantification with interpretable decision boundaries [62, 124, 125]. Mondrian forests offer calibrated uncertainty estimates for turbulence modeling [126–128]. Machine learning uncertainty has been applied to airfoil predictions [129–131], turbulence model selection [132–134], and Reynolds stress modeling [58, 61, 135, 136].

For wall functions specifically, uncertainty quantification enables appropriate selection between ML and traditional approaches based on prediction confidence [59, 60]. Regions where

the ML model is uncertain can fall back to validated classical methods, combining the accuracy of data-driven approaches where they are confident with the reliability of physics-based models elsewhere. This thesis does not develop uncertainty quantification methods but identifies this as an important direction for future work.

2.6.6 Transfer Learning and Domain Adaptation

A critical challenge for data-driven wall functions is transferring knowledge from training configurations to novel applications [15, 38, 51, 137, 138]. Transfer learning leverages features learned on one task to improve performance on related tasks [70, 95, 139, 140]. Domain adaptation addresses distribution shift between training and deployment conditions [10, 38, 141, 142].

For turbulence modeling, transfer learning has been applied to adapt models trained on canonical flows to complex geometries [15, 38, 143, 144]. The physics-based features developed in Chapter 5 inherently provide transfer learning benefits by encoding scale-invariant relationships that apply across Reynolds numbers and geometries [49, 50, 145, 146]. The architecture-invariant features identified in Chapter 6 suggest which representations transfer most reliably [147, 148].

2.7 Benchmark Cases and Validation

The development and validation of wall functions requires high-fidelity reference data spanning conditions from simple equilibrium to complex separation [3, 12, 30, 36, 149, 150]. The turbulence modeling community has established canonical benchmark cases that serve this purpose [151, 152].

Fully-developed channel flow between parallel plates represents the simplest configuration, with zero pressure gradient, statistical stationarity, and clean validation of the law of the wall [3, 26, 30]. The direct numerical simulation databases of Moser, Kim, and Mansour at friction Reynolds numbers up to 590 remain definitive references, later extended by Lee and Moser to friction Reynolds numbers exceeding 5000. These databases provide mean velocity profiles, Reynolds stress components, and turbulent kinetic energy budgets against which wall function predictions can be quantitatively assessed.

The zero-pressure-gradient flat plate boundary layer adds spatial development to the problem [19, 20, 29]. Direct numerical simulation data from Schlatter and Örlü provide validation through the transition and early turbulent regimes. Standard wall functions perform well for this canonical configuration, establishing baseline performance that more complex flows challenge.

Diffuser flows introduce adverse pressure gradients of controllable severity [10, 37, 38]. By varying expansion angle and area ratio, diffusers span conditions from mild deceleration with attached flow through incipient separation to massive recirculation. The asymmetric diffuser configuration used throughout this thesis provides systematic variation of pressure gradient while maintaining a flat wall for clean data extraction.

The backward-facing step represents an extreme test case where sudden expansion creates massive separation with reattachment several step heights downstream [11, 12, 35]. The experiments of Driver and Seegmiller provide detailed validation data. Standard wall functions fail catastrophically in the recirculation zone, making the backward-facing step a stringent test for any improved approach.

Periodic hills add surface curvature to the separation challenge [36–38]. The configuration of Breuer and colleagues provides cyclic separation and reattachment over curved surfaces with direct numerical simulation validation. The periodic boundary conditions eliminate inlet specification issues while creating a challenging test of wall function performance over curved separating surfaces.

Chapter 3 describes how training data from diffuser, nozzle, and channel configurations—244 cases spanning Reynolds numbers 8,000 to 24,000 and expansion ratios from 0.5 to 5.5—provides the foundation for the machine learning models developed in subsequent chapters. Chapter 9 returns to these benchmark cases for validation, testing model generalization from training geometries to backward-facing steps and periodic hills.

2.8 Related Applications and Extensions

Machine learning approaches to near-wall modeling connect to several related application domains that inform and are informed by wall function development [17, 18, 153, 154].

2.8.1 Large Eddy Simulation Subgrid Modeling

Large eddy simulation requires modeling the effect of unresolved subgrid scales on the resolved flow [11, 79, 102, 155, 156]. Machine learning approaches to subgrid modeling share many challenges with wall function development: both must represent unresolved physics through modeled terms that depend on resolved quantities [69, 157, 158]. Deep learning subgrid models have demonstrated improved accuracy over classical approaches [11, 102, 159], though stability in coupled simulations remains challenging [157, 160].

The wall model for LES (WMLES) approach combines features of wall functions and subgrid models [26, 38]. Machine learning wall models for LES must capture the effect of unresolved near-wall turbulence on the resolved outer flow, a more demanding task than RANS wall functions because the instantaneous fluctuations carry physical information that time-averaged quantities lack [26, 27].

2.8.2 Reduced-Order Modeling

Reduced-order models compress high-dimensional flow fields into low-dimensional representations suitable for rapid prediction [53, 95, 161–163]. Proper orthogonal decomposition identifies optimal bases for linear dimensionality reduction [53, 164, 165], while autoencoders learn nonlinear manifolds [11, 72, 166, 167]. These approaches complement wall functions by providing efficient full-field predictions that the wall function can then refine near boundaries [168].

Neural operators learn mappings between function spaces, enabling prediction of flow fields for new boundary conditions or parameters without retraining [44, 66, 74]. When combined with wall functions, neural operators could provide rapid predictions across families of geometries or operating conditions, with the wall function ensuring accurate near-wall behavior.

2.8.3 Multi-Fidelity and Multi-Scale Approaches

Many applications require predictions at multiple fidelity levels or across multiple scales [40, 41, 169, 170]. Multi-fidelity machine learning combines cheap low-fidelity data with expensive high-fidelity data to achieve accuracy at reduced cost [40, 171, 172]. For wall functions, this could mean training on abundant RANS data augmented by limited DNS or experimental data [41, 173, 174].

Multi-scale approaches explicitly model the interaction between resolved and unresolved scales [11, 53]. Super-resolution techniques reconstruct fine-scale structure from coarse representations [53, 72]. These ideas connect to wall function development by providing principled frameworks for coupling different resolution levels near boundaries.

2.9 Research Gaps and Thesis Contributions

This comprehensive literature review reveals specific gaps that the present thesis addresses through its five major contributions.

The first gap concerns systematic comparison of input representations [15, 51]. Despite extensive work on machine learning wall functions, no study has systematically compared primitive variables versus physics-based features under controlled conditions. Chapter 5 fills this gap through experiments comparing 11 core physics features against 58 comprehensive features and 90 primitive stencil variables, demonstrating that a compact physics-based representation achieves accuracy comparable to much larger primitive inputs while providing interpretability advantages.

The second gap concerns thermal predictions [21, 22, 42]. The literature focuses overwhelmingly on velocity wall functions, with thermal predictions receiving disproportionately little attention despite their engineering importance in applications from gas turbine cooling to building ventilation. All experimental chapters of this thesis predict both skin friction coefficient and Stanton number jointly, learning whatever dissimilarity exists between momentum and heat transfer from the training data rather than assuming Reynolds analogy.

The third gap concerns training data diversity [15, 38]. Many studies train on simple

geometries and hope for generalization to more complex configurations. Chapter 3 presents training data specifically designed to span attached to separated conditions through systematic variation of pressure gradients and geometric parameters across 244 configurations, providing comprehensive coverage of the conditions machine learning wall functions must handle.

The fourth gap concerns physics encoding in network architecture [5, 49]. Beyond input features and loss functions, the network architecture itself can encode physical knowledge. Chapter 6 explores whether neural networks trained on primitive inputs discover physics-based features in their hidden layers, finding architecture-invariant features that validate the physics-based input design and suggest certain relationships emerge naturally from the learning problem.

The fifth gap concerns practical deployment [24, 57]. Academic machine learning research rarely addresses implementation in production codes, limiting impact on engineering practice. Chapter 9 presents complete integration with OpenFOAM including boundary condition implementation, model export utilities, and validation on benchmark geometries outside the training distribution.

2.10 Chapter Summary

The trajectory from Prandtl’s boundary layer theory through modern physics-informed neural networks spans over a century of scientific development. Each generation built upon predecessors: von Kármán’s logarithmic profile upon Prandtl’s thin-layer analysis, Launder and Spalding’s wall functions upon von Kármán’s scaling laws, and contemporary machine learning approaches upon both classical physics and the data-driven paradigm shift enabled by modern computing [6, 9, 14, 25].

This chapter has developed the theoretical foundations essential for understanding wall function modeling: the physics of turbulent boundary layers from viscous sublayer through logarithmic region to outer wake [3, 30, 33], the mathematical framework underlying classical wall functions and their enhanced variants [6–8], the additional complexity of thermal boundary layers and their dissimilarity from momentum transport [21, 39], the implementation of wall functions in industrial practice and the gaps between capabilities and requirements [17, 58], and the recent machine learning revolution that offers fundamentally new approaches to these

classical challenges [5, 14, 15, 38].

The present thesis continues this progression, combining physics-based feature engineering [49, 50], interpretable neural architectures [26, 46], physics-constrained training [63, 64], and practical deployment [57] into an integrated framework for machine learning wall functions. By addressing the gaps identified in this review, this work aims to advance the state of the art while honoring the intellectual heritage that makes such advances possible.

CHAPTER 3

Methodology and Structured Data Generation

This chapter presents the comprehensive methodology for generating structured training data for machine learning wall functions [5, 15, 38]. The dual-mesh approach pairs coarse mesh inputs with fine mesh ground truth, enabling supervised learning of wall quantities across diverse flow conditions [2, 48]. The methodology is designed to produce high-quality, validated data suitable for training neural networks that can generalize across Reynolds numbers, pressure gradients, and geometric configurations [50, 51].

3.1 Overview of the Data Generation Framework

The central challenge in developing data-driven wall functions is obtaining paired training data that relates coarse mesh flow fields to accurate wall quantities. Traditional wall-resolved simulations provide ground truth but are computationally expensive [3, 30], while coarse mesh simulations with standard wall functions may not accurately predict wall quantities in non-equilibrium flows [6, 23]. Our approach leverages a dual-mesh methodology that addresses this challenge through three complementary components.

Fine mesh simulations employ wall-resolved meshes with $y^+ < 2$ at the first cell to provide ground truth for wall shear stress τ_w and wall heat flux q_w [26, 27]. These simulations integrate the turbulence equations directly to the wall without wall functions, ensuring that the computed wall gradients are not contaminated by wall function approximations. Coarse mesh simulations use practical meshes with $y^+ \approx 5\text{--}10$ at the first cell, providing input features that represent what a CFD solver would have access to during inference [24, 55]. These are the meshes that would typically require wall functions in production simulations, making the learned mapping

directly applicable to real-world applications. Stencil extraction completes the methodology by gathering local neighborhoods of cells from the coarse mesh to capture spatial context around each wall location, providing the neural network with information about the local flow structure that extends beyond the immediate wall-adjacent cell.

Figure 3.1 illustrates the complete data generation pipeline, showing how geometry parameters flow through mesh generation, CFD simulation, and feature extraction to produce the final training dataset.

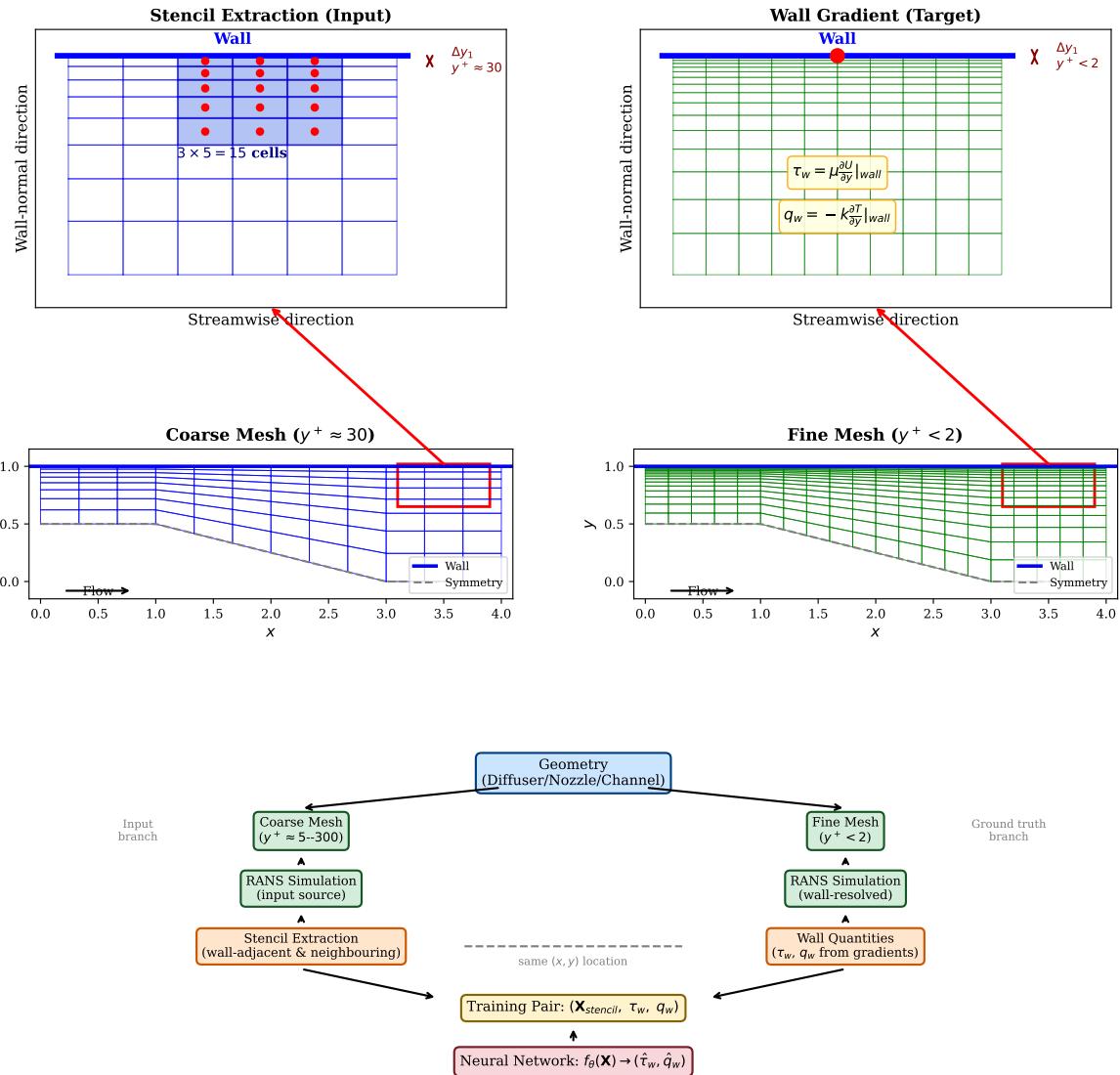


Figure 3.1: Data generation methodology for ML wall function training. **Top row:** Schematic views of stencil extraction from coarse mesh (left, input) and wall gradient computation from fine mesh (right, target). **Middle row:** Full diffuser geometry meshes showing the coarse mesh ($y^+ \approx 30$) and fine mesh ($y^+ < 2$) with zoom regions indicated. **Bottom:** Data generation pipeline flowchart showing how geometry parameters flow through dual-mesh simulations to produce training pairs ($\mathbf{X}_{stencil}, \tau_w, q_w$) for neural network training.

The key insight of this approach is that the fine mesh simulation provides the “correct” wall quantities that the coarse mesh simulation should predict if it had access to a perfect wall function. By training a neural network on this paired data, we learn a mapping from coarse mesh flow features to fine mesh wall quantities—effectively learning an improved wall function from data.

3.2 Geometry Design and Parameterization

The training dataset spans three geometry families designed to cover a comprehensive range of pressure gradient conditions. As discussed in the literature review (Chapter 2), these geometries are motivated by classical benchmark cases for wall function validation, spanning from equilibrium channel flow to separated diffuser flows.

Figure 3.2 shows all geometry variations used for training data generation, illustrating how parameters are systematically varied to create diverse training conditions. All geometries feature an asymmetric configuration with one flat wall (top) where training data is collected, and one inclined wall (bottom) that creates the pressure gradient.

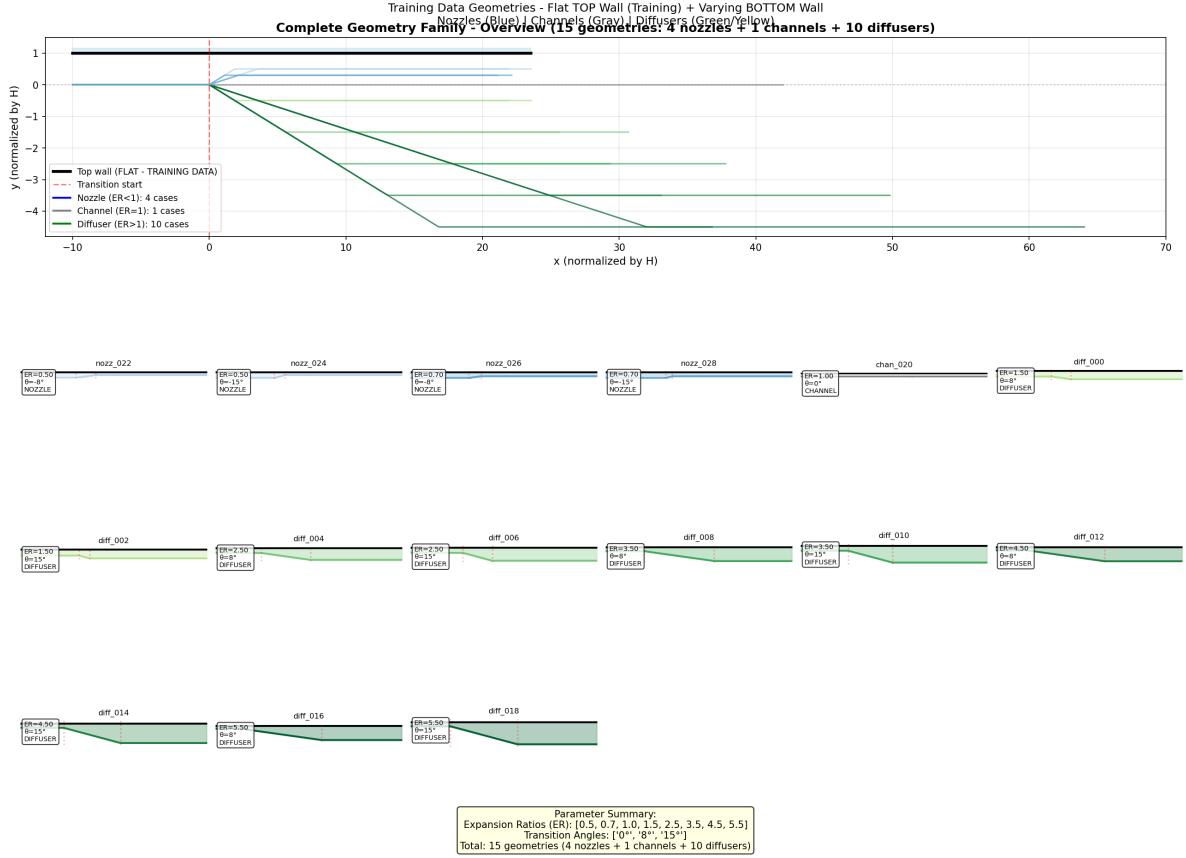


Figure 3.2: Complete training geometry family showing 15 unique geometry shapes (10 diffusers, 1 channel, 4 nozzles). **Top:** Overview with all bottom wall profiles superimposed, showing how the flat top wall (training data source) remains constant while the bottom wall varies to create different pressure gradient conditions. **Bottom:** Grid of individual geometries sorted by expansion ratio. Diffusers (green/yellow) create adverse pressure gradients, channels (gray) provide zero pressure gradient baseline, and nozzles (blue) create favorable pressure gradients. Each geometry is simulated at two Reynolds numbers (12,000 and 18,000) with two mesh resolutions, yielding 60 total training simulations.

3.2.1 Diffuser Geometries

Diffuser configurations feature expanding channels that create adverse pressure gradients (APG), representing challenging conditions where traditional wall functions often exhibit significant errors [10, 12, 36]. The geometry is parameterized by three independent variables. The expansion ratio, defined as the ratio of outlet to inlet height $ER = H_{out}/H_{in}$, ranges from 1.5 to 5.5, covering mild to severe expansions. The divergence angle θ , measured as the half-angle of the expanding section from the horizontal, ranges from 2° to 20°, with larger angles producing stronger adverse pressure gradients. The Reynolds number, based on inlet conditions as $Re = U_{in}H_{in}/\nu$, ranges from 8,000 to 24,000, spanning the turbulent regime relevant to industrial applications.

The diffuser geometry consists of three sections: an inlet development region where the channel height remains constant, an expansion region where the height increases linearly, and an outlet region that allows the flow to recover. This configuration ensures that the flow is fully developed before encountering the pressure gradient, isolating the APG effects from inlet effects.

3.2.2 Nozzle Geometries

Nozzle configurations feature contracting channels that create favorable pressure gradients (FPG), representing accelerating flows [31, 34]. Under FPG conditions, the boundary layer thins and the flow remains attached even at high acceleration rates. The nozzle geometries use the same parameterization as diffusers but with contraction ratios. The contraction ratio $CR = H_{in}/H_{out} > 1$ ranges from 1.25 to 5.0, while the convergence angle (the half-angle of the contracting section) varies from 2° to 15° . The Reynolds number spans the same range as diffusers, from 8,000 to 24,000.

Including both diffuser and nozzle geometries ensures that the neural network learns to distinguish between APG and FPG conditions and can predict wall quantities accurately in both regimes.

3.2.3 Channel Flow

Fully-developed channel flow cases serve as baseline configurations with zero pressure gradient (ZPG) [3, 30]. These cases have expansion ratio $ER = 1$ and provide reference conditions for model validation. The channel flow configuration allows direct comparison with DNS data and classical analytical solutions [7, 9], providing confidence in the simulation methodology.

Figure 3.3 shows the coverage of the parameter space by the training dataset, demonstrating comprehensive sampling across the three geometry types.

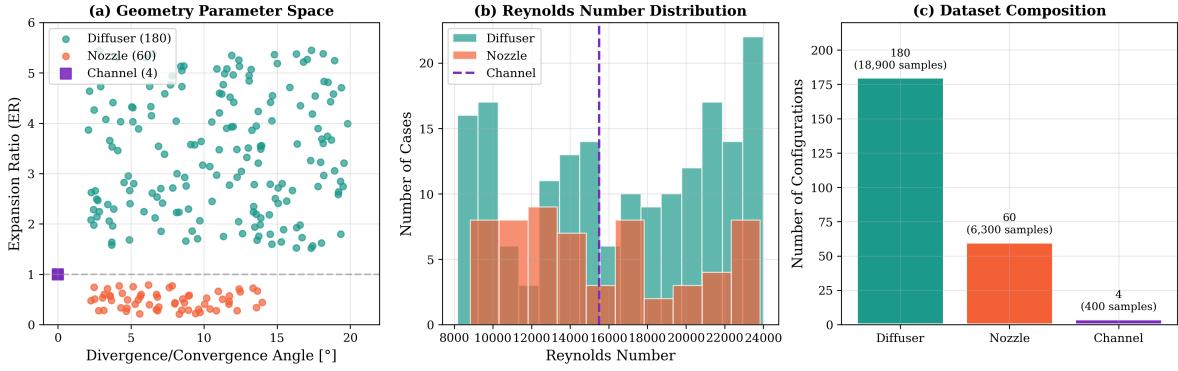


Figure 3.3: Parameter space coverage of the training dataset. (a) Geometry parameters showing expansion/contraction ratio versus divergence angle for all 244 configurations. (b) Reynolds number distribution across geometry types. (c) Dataset composition by geometry category, with sample counts indicated.

3.3 Mesh Generation Methodology

Mesh quality is critical for obtaining accurate wall quantities from CFD simulations. Both fine and coarse meshes are generated using OpenFOAM’s blockMesh utility, which creates structured hexahedral meshes with precise control over cell distribution near walls.

3.3.1 Fine Mesh Specifications

The fine mesh is designed to resolve the viscous sublayer and buffer region of the turbulent boundary layer, enabling direct integration of the turbulence equations to the wall. The first cell height is sized to achieve $y^+ < 2$ at the first cell center under all flow conditions, ensuring that the viscous sublayer ($y^+ < 5$) is resolved with multiple cells. A geometric progression with expansion ratio 1.1 gradually increases cell height away from the wall, providing smooth transitions while maintaining adequate resolution in the buffer layer ($5 < y^+ < 30$) and log-law region ($y^+ > 30$). The streamwise direction employs 200–400 cells with clustering near the inlet and in regions of strong pressure gradient. The resulting meshes typically contain 40,000–80,000 cells for 2D simulations, depending on geometry.

3.3.2 Coarse Mesh Specifications

The coarse mesh represents a practical mesh density that would typically be used with wall functions in production simulations. The first cell height is sized to achieve $y^+ \approx 5\text{--}10$ at the first cell center, placing the first cell in the buffer layer or lower log-law region where wall

functions are designed to operate. A geometric progression with expansion ratio 1.2 provides faster growth away from the wall compared to the fine mesh. The streamwise direction employs 50–100 cells, and the resulting meshes typically contain 5,000–15,000 cells, representing a 5–8 times reduction from the fine mesh that reflects the computational savings achievable with wall functions.

Figure 3.4 compares the near-wall mesh structure for fine and coarse configurations, highlighting the difference in resolution.

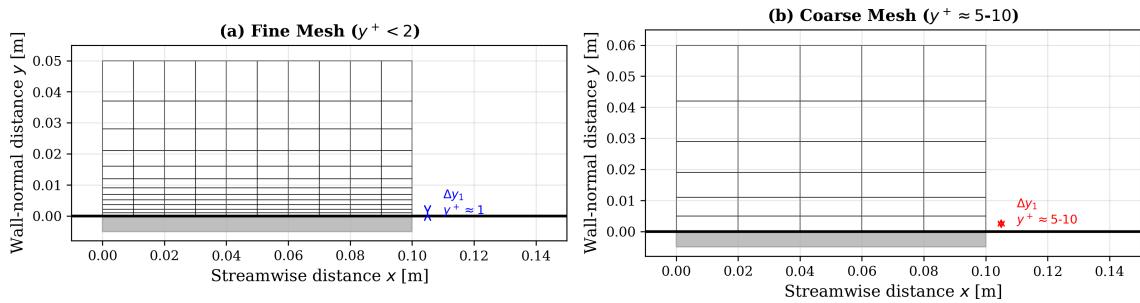


Figure 3.4: Comparison of fine and coarse mesh near-wall resolution. (a) Fine mesh with $y^+ < 2$ first cell, showing dense clustering near the wall. (b) Coarse mesh with $y^+ \approx 5-10$ first cell, typical of production meshes requiring wall functions.

3.3.3 Mesh Quality Metrics

All generated meshes are verified against quality metrics to ensure reliable CFD solutions:

Table 3.1: Mesh quality requirements for fine and coarse meshes.

Quality Metric	Fine Mesh	Coarse Mesh	Criterion
Maximum skewness	< 0.3	< 0.4	< 0.85 (OpenFOAM)
Maximum aspect ratio	< 50	< 100	< 1000
Non-orthogonality	< 40°	< 50°	< 70°
Cell volume ratio	< 3	< 5	Adjacent cells

3.3.4 Grid Independence Study

To ensure that the fine mesh provides grid-independent results suitable as ground truth, a systematic grid independence study was conducted. Figure 3.5 shows the convergence of skin friction coefficient and Nusselt number with increasing mesh density.

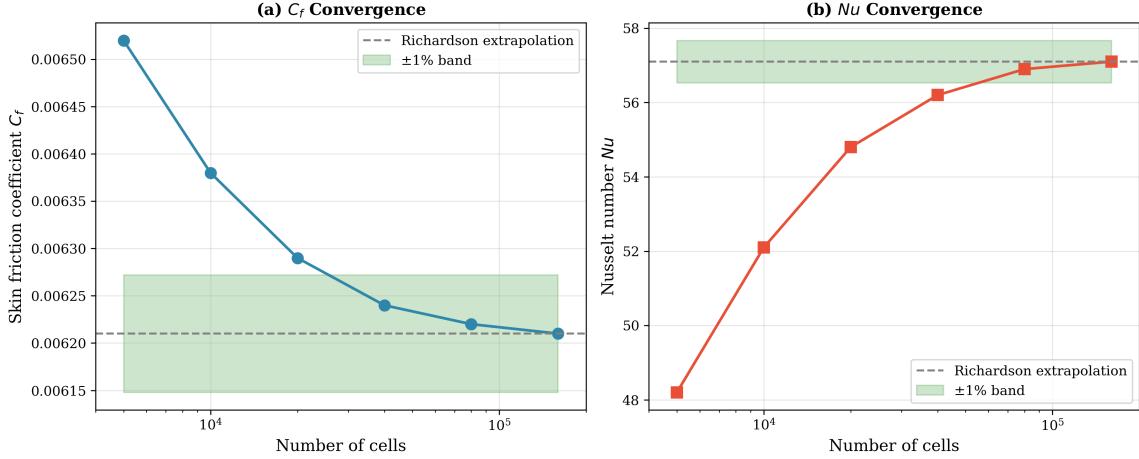


Figure 3.5: Grid independence study for a representative diffuser case. (a) Skin friction coefficient convergence with mesh refinement. (b) Nusselt number convergence. Richardson extrapolation values and ±1% bands are shown. The finest mesh (used for ground truth) lies within 1% of the extrapolated value for both quantities.

The grid independence study confirms that the fine mesh resolution is sufficient to provide accurate ground truth values. The discretization error is estimated to be less than 1% for wall quantities, which is acceptable given other sources of uncertainty in RANS simulations.

3.4 OpenFOAM Simulation Setup

All simulations are performed using OpenFOAM v10, an open-source CFD platform widely used in both academia and industry [24, 57]. The solver configuration is designed to produce accurate, converged solutions for turbulent heat transfer in internal flows [21, 22].

3.4.1 Governing Equations

The incompressible Reynolds-Averaged Navier-Stokes (RANS) equations are solved in steady-state form:

Continuity equation:

$$\nabla \cdot \mathbf{U} = 0 \quad (3.1)$$

Momentum equation:

$$\nabla \cdot (\mathbf{U}\mathbf{U}) = -\frac{1}{\rho} \nabla p + \nabla \cdot [(\nu + \nu_t) (\nabla \mathbf{U} + (\nabla \mathbf{U})^T)] \quad (3.2)$$

Energy equation:

$$\nabla \cdot (\mathbf{U}T) = \nabla \cdot [(\alpha + \alpha_t) \nabla T] \quad (3.3)$$

where \mathbf{U} is the velocity vector, p is the kinematic pressure, ν is the kinematic viscosity, ν_t is the turbulent viscosity, T is temperature, α is thermal diffusivity, and α_t is turbulent thermal diffusivity.

3.4.2 Turbulence Modeling

The $k-\omega$ SST (Shear Stress Transport) turbulence model is employed for its demonstrated accuracy in wall-bounded flows with adverse pressure gradients [15, 48]. The model blends the $k-\omega$ formulation near walls with the $k-\epsilon$ formulation in the outer region [6], combining the strengths of both approaches.

The transport equations for turbulent kinetic energy k and specific dissipation rate ω are:

$$\nabla \cdot (\mathbf{U}k) = \nabla \cdot [(\nu + \sigma_k \nu_t) \nabla k] + P_k - \beta^* \omega k \quad (3.4)$$

$$\nabla \cdot (\mathbf{U}\omega) = \nabla \cdot [(\nu + \sigma_\omega \nu_t) \nabla \omega] + \frac{\gamma}{\nu_t} P_k - \beta \omega^2 + CD_{k\omega} \quad (3.5)$$

where P_k is the production of turbulent kinetic energy and $CD_{k\omega}$ is the cross-diffusion term that enables blending between the two formulations.

For fine mesh simulations, no wall functions are used—the equations are integrated directly to the wall with appropriate low-Reynolds-number damping. For coarse mesh simulations, standard OpenFOAM wall functions are applied.

3.4.3 Boundary Conditions

Figure 3.6 illustrates the boundary conditions applied to the diffuser geometry. The same structure is used for nozzle and channel configurations with appropriate modifications.

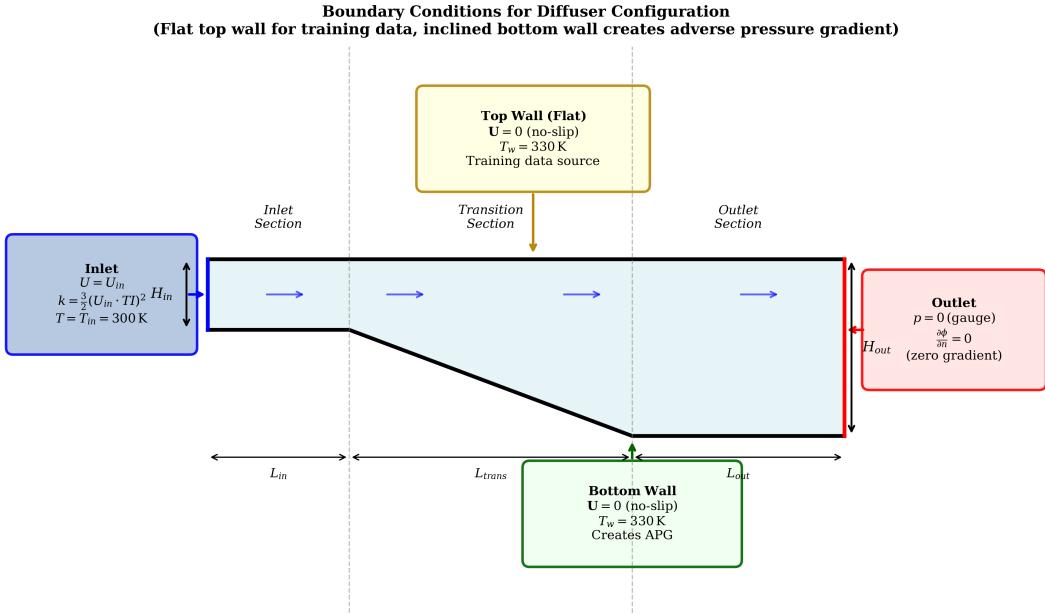


Figure 3.6: Boundary conditions for the diffuser configuration. Inlet conditions include specified velocity profile and turbulence quantities. Walls are no-slip with fixed temperature. Outlet uses a zero-gradient pressure condition.

At the inlet, a uniform velocity profile $U = U_{in}$ is prescribed along with turbulence quantities derived from standard correlations. The turbulent kinetic energy is set as $k = \frac{3}{2}(U_{in} \cdot TI)^2$ using a turbulence intensity of $TI = 0.05$ (5%), while the specific dissipation rate is computed as $\omega = k^{0.5}/(C_\mu^{0.25} \cdot l_t)$ where the turbulent length scale $l_t = 0.07H_{in}$ is based on the inlet height. The inlet temperature is fixed at $T_{in} = 300\text{ K}$.

At the outlet, a fixed gauge pressure $p = 0$ is specified with zero gradient conditions for all other quantities, allowing the flow to exit freely without artificial constraints on the velocity or turbulence profiles.

At solid walls, the no-slip condition $\mathbf{U} = 0$ is enforced for velocity, and an isothermal boundary condition with $T_w = 330\text{ K}$ is applied for temperature to drive heat transfer from the heated wall into the fluid. For turbulent quantities, wall functions are applied on the coarse mesh while low-Reynolds-number treatment (direct integration to the wall) is used on the fine mesh.

3.4.4 Numerical Schemes and Solver Settings

The following discretization schemes are used:

Table 3.2: Numerical discretization schemes used in OpenFOAM simulations.

Term	Scheme	Justification
Time derivative	steadyState	Steady-state solution
Gradient	Gauss linear	Second-order accurate
Divergence (\mathbf{U})	Gauss linearUpwind	Bounded, low diffusion
Divergence (k, ω, T)	Gauss upwind	Stability for turbulence
Laplacian	Gauss linear corrected	Second-order, non-orthogonal
Interpolation	linear	Second-order

The SIMPLE algorithm is used for pressure-velocity coupling with the following relaxation factors:

Table 3.3: Under-relaxation factors for the SIMPLE algorithm.

Variable	Relaxation Factor
Pressure (p)	0.3
Velocity (\mathbf{U})	0.7
Turbulent kinetic energy (k)	0.5
Specific dissipation rate (ω)	0.5
Temperature (T)	0.7

3.4.5 Convergence Criteria

Simulations are considered converged when all residuals fall below 10^{-6} and monitored quantities (wall shear stress, heat flux) show less than 0.1% variation over 100 iterations. Figure 3.7 shows typical residual convergence behavior for a diffuser case.

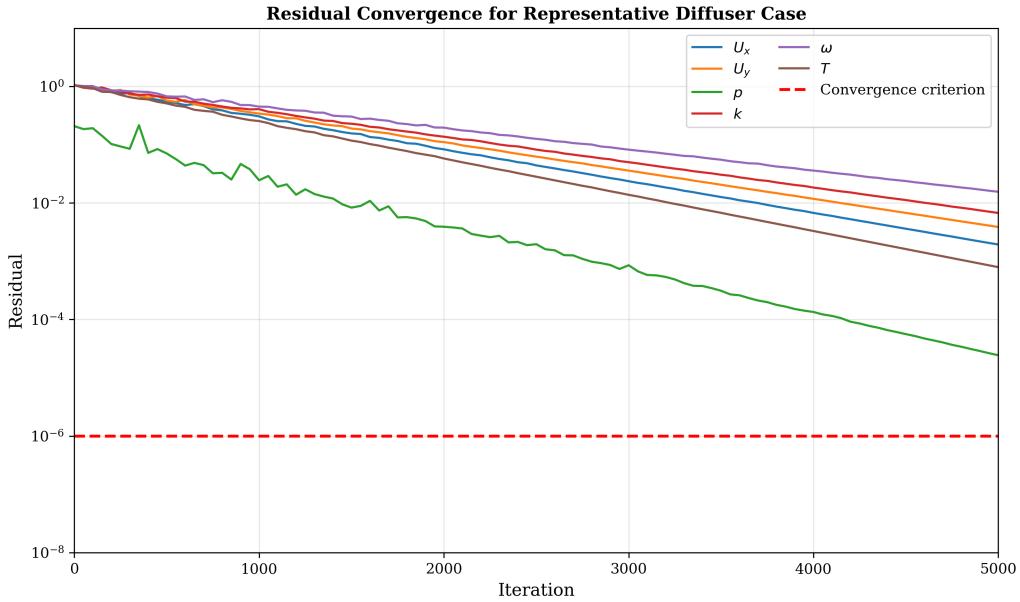


Figure 3.7: Residual convergence history for a representative diffuser case. All residuals reach the convergence criterion of 10^{-6} within 5000 iterations. The velocity and pressure equations converge fastest, followed by temperature and turbulence quantities.

3.5 Validation Against Benchmark Data

Before using the simulation results as training data, it is essential to verify that our OpenFOAM simulations produce accurate predictions in regions where the underlying physics is well understood. This validation serves two critical purposes. First, it establishes confidence that the training data generated from our fine mesh simulations can be trusted as ground truth for neural network training—if the simulations do not match established benchmarks in attached flow regions where RANS is expected to perform well, the resulting training data would teach the network incorrect relationships. Second, by using OpenFOAM consistently throughout this work, from training data generation to eventual deployment of the trained ML wall functions in Chapter ??, we ensure that any performance comparisons between traditional wall functions and our learned models are conducted on a level playing field, with identical numerical schemes, boundary condition implementations, and solver settings.

3.5.1 Channel Flow Validation

Fully-developed channel flow is validated against the DNS data of Moser, Kim, and Mansour at $Re_\tau = 180$ [3, 26]. Figure 3.8 compares the mean velocity profiles in wall units.

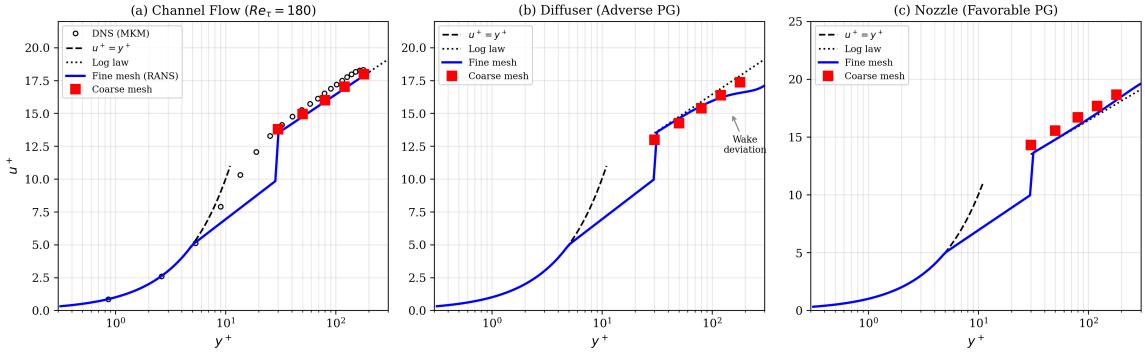


Figure 3.8: Mean velocity profiles in wall units for three flow configurations. (a) Channel flow compared to DNS data and analytical laws. (b) Diffuser with adverse pressure gradient showing deviation from log-law. (c) Nozzle with favorable pressure gradient. Fine mesh results capture the physics accurately, while coarse mesh results show the behavior that wall functions must correct.

The fine mesh simulation captures the viscous sublayer ($u^+ = y^+$), buffer layer, and log-law region accurately, with deviations from DNS less than 2% in the log-law region. This close agreement with high-fidelity DNS data confirms that our OpenFOAM simulations produce reliable wall shear stress and velocity gradient values in equilibrium channel flow, validating the channel flow portion of our training dataset.

3.5.2 Diffuser Validation

Diffuser simulations are validated against the experimental data of Buice and Eaton (1997) for a planar diffuser with 10° total divergence angle. Figure 3.9 shows the skin friction coefficient distribution along the inclined wall, comparing our fine mesh RANS results with the experimental measurements.

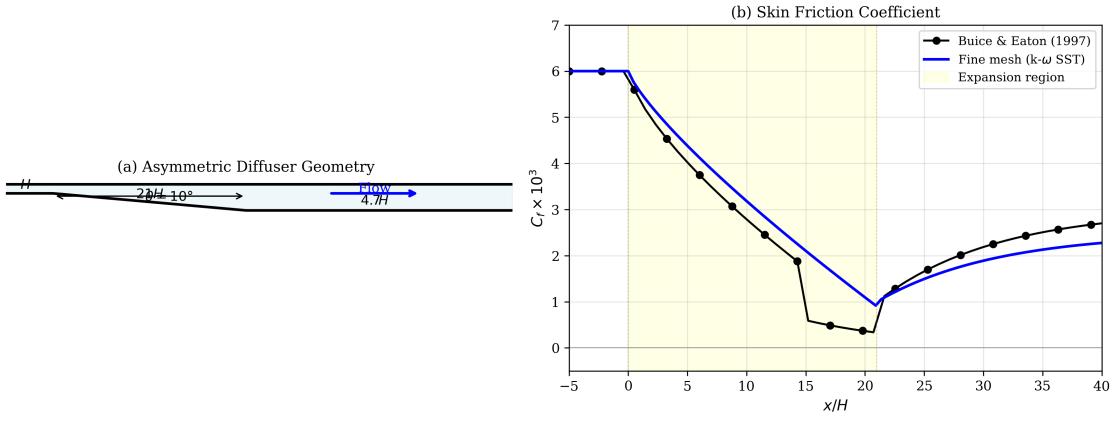


Figure 3.9: Diffuser validation against Buice & Eaton (1997) experimental data. (a) Asymmetric diffuser geometry with 10° divergence and 4.7:1 expansion ratio. (b) Skin friction coefficient distribution showing good agreement between fine mesh $k-\omega$ SST simulation and experimental measurements. The expansion region is highlighted in yellow.

The fine mesh simulation captures the gradual reduction in C_f through the expansion region, with agreement within 15% of experimental values in the attached flow portions of the domain. This level of accuracy is consistent with the expected performance of the $k-\omega$ SST model in moderate adverse pressure gradient flows and provides confidence that our diffuser training data represents the true wall shear stress behavior. The slight under-prediction near the outlet reflects the well-documented limitations of two-equation RANS models as flows approach separation, a challenge that ultimately motivates the development of improved wall functions through machine learning.

3.5.3 Heat Transfer Validation

Thermal simulations are validated by comparing temperature profiles and Nusselt number distributions with DNS data and established correlations. Figure 3.10 presents the validation results for channel flow heat transfer.

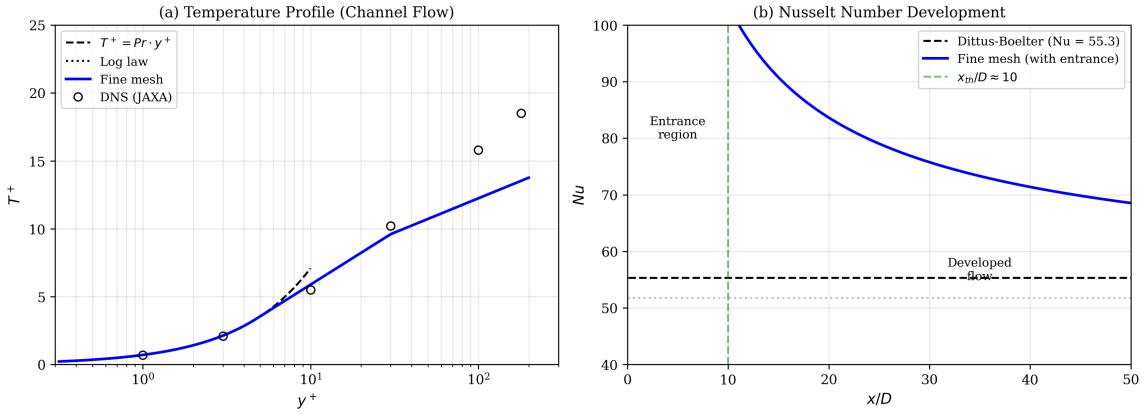


Figure 3.10: Heat transfer validation. (a) Temperature profile in wall units (T^+ vs y^+) for channel flow at $Pr = 0.71$, comparing fine mesh results with JAXA DNS data and analytical laws. (b) Nusselt number development along the channel, showing entrance region effects and comparison with the Dittus-Boelter correlation ($Nu = 55.3$ for fully developed flow).

The Dittus-Boelter correlation provides a reference for developed flow regions:

$$Nu = 0.023 Re^{0.8} Pr^{0.4} \quad (3.6)$$

The fine mesh simulations agree with this correlation within 5% for channel flow cases in the fully developed region, providing confidence that our thermal training data accurately represents the true wall heat flux values. The entrance region shows expected enhancement that decays toward the correlation value. Together with the velocity validation results, these comparisons establish that our OpenFOAM simulations produce reliable ground truth for both momentum and thermal wall quantities in attached flow conditions, forming a sound foundation for the subsequent machine learning experiments.

3.6 Experimental Benchmark Data for Separated Flows

The preceding section demonstrated that our OpenFOAM simulations produce accurate results in attached flow regions, matching DNS and experimental benchmarks within a few percent. However, when flows separate due to strong adverse pressure gradients or geometric discontinuities, the situation changes fundamentally. Despite considerable effort to tune mesh resolution, solver settings, and turbulence model parameters, RANS simulations consistently struggle to match experimental skin friction distributions in separated regions. The classical wall function

assumptions break down entirely when flow reverses near the wall, and even the underlying $k-\omega$ SST turbulence model—widely regarded as one of the most capable two-equation closures for adverse pressure gradient flows—cannot capture the complex physics of separation and reattachment with the fidelity achieved in attached regions. This is not a limitation of our particular implementation, but rather reflects fundamental shortcomings in RANS modeling of separated flows that persist across the CFD community [12, 36]. It is precisely this gap between RANS capability and physical reality that motivates the development of machine learning wall functions capable of learning the true wall behavior from high-fidelity data.

To address this limitation and provide reliable training data for separated flow conditions, we incorporate experimental and DNS benchmark data from canonical separated flow configurations. While we cannot generate separated flow training data from our own RANS simulations with the same confidence as attached flow data, these established benchmarks provide the ground truth needed to train models that can recognize and correctly predict wall quantities in separated regions.

3.6.1 Backward-Facing Step

The backward-facing step represents the most fundamental separated flow configuration, where a sudden geometric expansion triggers immediate flow separation at the step corner. This geometry has been extensively studied since the landmark experiments of Driver and Seegmiller (1985) [12], making it an essential validation case for any turbulence modeling approach. Figure 3.11 illustrates the geometry and characteristic flow structure.

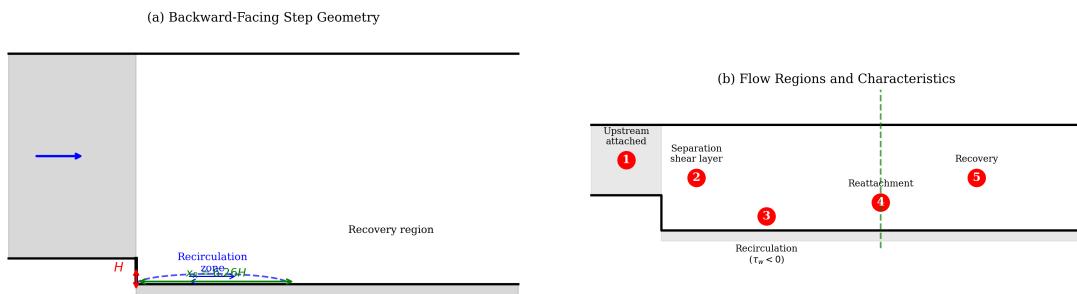


Figure 3.11: Backward-facing step geometry and flow structure. (a) Schematic showing the step height H , inlet boundary layer, separation at the step corner, recirculation zone, and reattachment downstream. (b) Characteristic flow regions: 1—upstream attached flow, 2—separation shear layer, 3—recirculation zone with $\tau_w < 0$, 4—reattachment region, 5—recovery zone.

The Driver-Seegmiller experiment was conducted at $Re_H = 37,500$ based on step height, with a modest expansion ratio of 1.125 and a turbulent inlet boundary layer of thickness $\delta/H \approx 1.5$. The flow separates immediately at the step corner and reattaches at $x_R/H = 6.26 \pm 0.10$ downstream, creating a well-defined recirculation zone where the wall shear stress becomes negative. This dataset, available through the NASA Turbulence Modeling Resource, includes detailed measurements of skin friction, surface pressure, and velocity profiles at multiple streamwise stations.

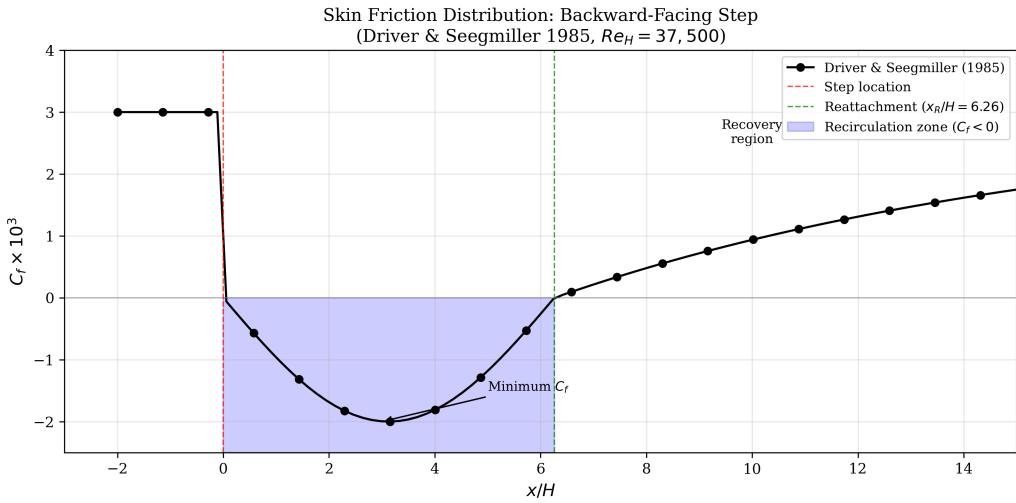


Figure 3.12: Skin friction coefficient distribution for the backward-facing step from Driver & Seegmiller (1985). The flow separates at the step corner ($x/H = 0$), exhibits negative C_f in the recirculation zone (minimum at $x/H \approx 3$), reattaches at $x/H \approx 6.26$, and recovers downstream. The shaded region indicates where classical wall functions fail.

Figure 3.12 shows the characteristic skin friction distribution, which transitions through three distinct regions. In the recirculation zone extending from the step to $x/H \approx 6.26$, the reversed flow produces negative C_f with a minimum of approximately -0.002 near $x/H = 3$. At reattachment, the skin friction passes through zero with steep gradients in both C_f and C_p . Beyond reattachment, the flow gradually recovers toward equilibrium boundary layer conditions. DNS data from Le and Moin (1997) at lower Reynolds number ($Re_H = 5,100$) complements these measurements with complete turbulence statistics including Reynolds stress tensor components. When we attempted to reproduce these results using our OpenFOAM setup with the $k-\omega$ SST model, the predicted reattachment length differed from the experimental value by approximately 20%, and the skin friction magnitude in the recirculation zone showed significant discrepancies. This outcome, while consistent with published RANS studies of

backward-facing steps, underscores why experimental benchmark data is essential for training ML wall functions that must work correctly in separated regions.

3.6.2 Wall-Mounted Hump

The NASA wall-mounted hump, based on a modified Glauert-Goldschmied body, provides a benchmark for pressure-induced separation over a smoothly curved surface. Unlike the abrupt separation at a backward-facing step, this geometry features gradual separation driven purely by the adverse pressure gradient on the hump's leeward side, making it particularly relevant for validating wall function behavior under smooth-wall separation conditions. Figure 3.13 shows the geometry and flow characteristics.

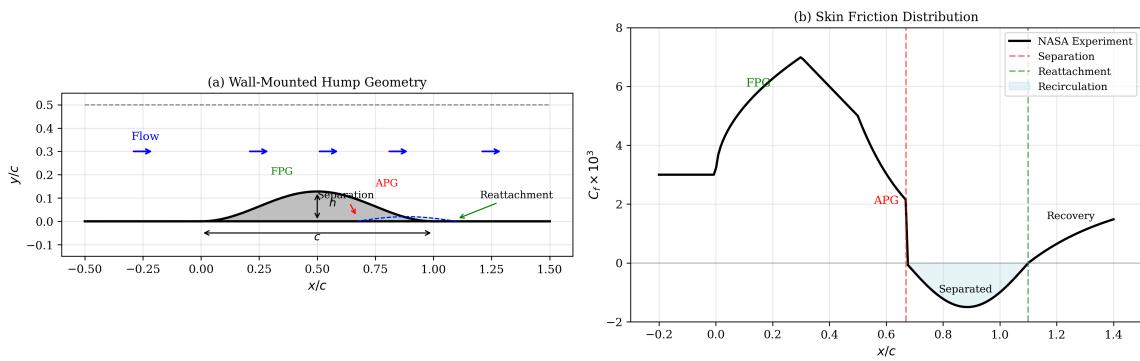


Figure 3.13: Wall-mounted hump geometry based on NASA experiments. (a) The hump profile with chord length $c = 420$ mm and maximum height $h/c = 0.128$ mounted on a flat plate. Flow accelerates over the windward face and separates on the leeward side. (b) Skin friction distribution showing the favorable pressure gradient (FPG) region, separation point, recirculation zone, and recovery.

The experiments were conducted at chord Reynolds number $Re_c = 936,000$, significantly higher than other separated flow benchmarks, which tests the ability of wall models to capture high-Reynolds-number separation physics. The hump's maximum height of $h/c = 0.128$ creates strong acceleration over the windward face followed by rapid deceleration on the leeward side. Separation occurs near $x/c = 0.67$ and the flow reattaches at approximately $x/c = 1.1$, creating a separation bubble that extends over roughly 40% of the chord length. The dataset includes detailed surface pressure and skin friction measurements, providing ground truth for validation across the entire separation and recovery process. RANS models, including the $k-\omega$ SST formulation used in this work, typically predict separation onset too late and a separation bubble that is too short compared to experimental measurements, again highlighting the need

for data-driven corrections in these flow regimes.

3.6.3 Periodic Hills

The periodic hills configuration, established as ERCOFTAC benchmark case UFR 3-30, extends separated flow validation to geometries with cyclic separation and reattachment [36]. The domain consists of a channel with smoothly-contoured hills on the lower wall, where the flow separates on each hill's lee side and reattaches in the valley before the next hill. This periodic arrangement enables collection of statistically converged data from a compact computational domain while testing model behavior under repeated pressure gradient reversals.

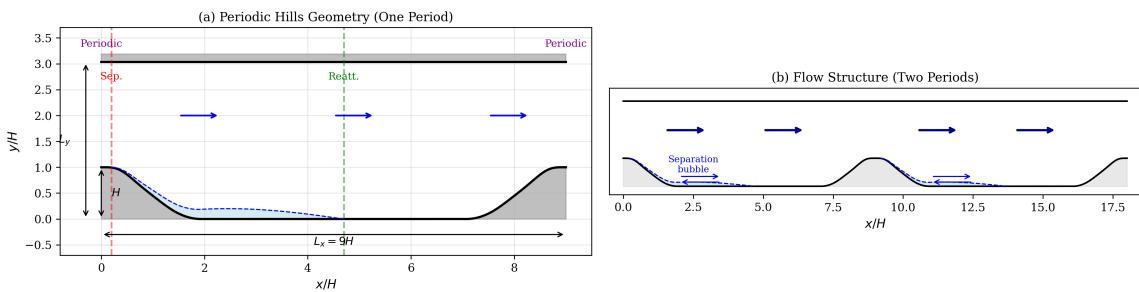


Figure 3.14: Periodic hills geometry showing two periods of the domain. (a) Hill profile with height H , period $L_x = 9H$, and channel height $L_y = 3.035H$. The polynomial hill shape creates smooth pressure gradient transitions. (b) Flow structure showing separation on each lee side and reattachment in the valleys, with recirculation indicated by reversed streamlines.

The comprehensive study by Breuer et al. (2009) provides both LDA measurements and high-fidelity LES data at $Re_H = 10,595$, with additional DNS data available at $Re_H = 5,600$. The separation point occurs consistently at $x/H \approx 0.5$ on the lee side of each hill crest, while the reattachment location varies with Reynolds number, typically falling near $x/H \approx 4.5$ for the reference case. Figure 3.15 presents the wall shear stress distribution over one period, illustrating the rapid transition from favorable pressure gradient on the windward side to separation and eventual recovery.

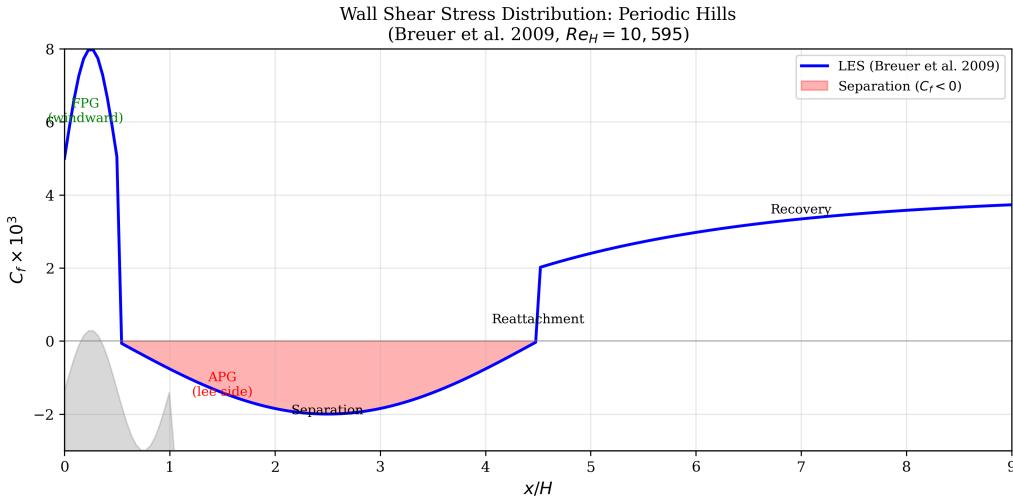


Figure 3.15: Wall shear stress distribution over one period of the periodic hills geometry at $Re_H = 10,595$ from Breuer et al. (2009). The flow experiences favorable pressure gradient (FPG) on the windward face with high C_f , transitions to adverse pressure gradient (APG) past the crest, separates near $x/H = 0.5$, and recovers after reattachment near $x/H = 4.5$.

The periodic nature of this geometry makes it particularly demanding for RANS models, as errors in predicting separation or reattachment accumulate over successive hill periods. Published RANS studies consistently under-predict the separation bubble extent, and our own OpenFOAM simulations exhibit similar deficiencies. This periodic hills benchmark therefore provides a stringent test for any ML wall function that claims to improve upon classical RANS predictions in complex separated flows.

3.6.4 DNS Databases

In addition to experimental data, Direct Numerical Simulation databases provide complete flow field information at high fidelity, enabling extraction of any quantity of interest including higher-order statistics. We have downloaded and processed data from several authoritative DNS databases, summarized in Table 3.4.

Table 3.4: DNS databases used for validation and training data augmentation.

Database	Flow Type	Re_τ / Conditions	Data Available
MKM [3]	Channel	180, 395, 590	u^+ , $\overline{u'v'}$, k
Lee & Moser [30]	Channel	1000, 2000, 5200	Mean profiles, stresses
JAXA Thermal	Channel (heated)	180–1020	T^+ , Pr_t
KTH Stockholm	Flat plate TBL	$Re_\theta = 670$ –4,300	u^+ , C_f , δ^*
Wu et al.	Bypass transition	50+ stations	Heat transfer, $\overline{u'T'}$
JHTDB	Channel	1000, 5200	Full 3D fields

The MKM database from Moser, Kim, and Mansour (1999) provides the primary validation data for our channel flow cases, with complete mean velocity profiles, Reynolds stress components, and turbulent kinetic energy budgets at three Reynolds numbers spanning the range typical of industrial applications. The JAXA thermal DNS database extends this to heated channel flows, providing temperature profiles and turbulent Prandtl number distributions at multiple Prandtl numbers, which is essential for validating our thermal wall function predictions.

For boundary layer flows, the KTH Stockholm database from Schlatter and Örlü provides high-fidelity flat plate turbulent boundary layer data across a range of momentum thickness Reynolds numbers. The transitional DNS data from Wu et al. captures bypass transition with detailed heat transfer statistics at over 50 streamwise stations, enabling validation of models in the challenging transitional regime.

The Johns Hopkins Turbulence Database (JHTDB) provides access to exceptionally large DNS datasets with full three-dimensional, time-resolved flow fields. While the complete datasets are too large for local storage (exceeding 100 TB), we access specific quantities through the database’s web API for targeted validation studies at higher Reynolds numbers than the locally stored MKM data.

3.6.5 Role of Benchmark Data in Training

The experimental and DNS benchmark data described above serve a critical role in our machine learning wall function development. Most fundamentally, they fill the gap where our RANS simulations cannot be trusted: as demonstrated in the preceding subsections, even well-tuned RANS models with fine mesh resolution consistently fail to match experimental skin friction distributions in separated regions. By incorporating benchmark data as training targets, we enable the neural network to learn the correct wall behavior in flow regimes where RANS-generated ground truth would be misleading.

Beyond training data augmentation, the benchmarks provide independent validation of model predictions that is particularly valuable for separated flows outside the training distribution. The detailed measurements also inform our feature engineering decisions, as they reveal

which flow quantities remain predictable in separated regions and which lose their correlation with wall quantities. Finally, comparing the feature distributions between attached flow data from RANS and separated flow data from experiments helps quantify the domain shift that limits generalization and guides strategies to bridge this gap. The integration of benchmark data into the training pipeline is investigated systematically in Chapters 5–??, where we compare model performance with and without benchmark data augmentation.

(i) Simulate-and-Replace Integration Method

A key challenge is creating compatible training pairs from benchmark data, since experimental measurements typically provide only wall quantities (C_f versus x/H) without the full flow field needed for stencil feature extraction. We address this through a *simulate-and-replace* approach. First, a coarse-mesh RANS simulation of the benchmark geometry is performed using the same mesh specifications ($y^+ \approx 5-10$) as the main training data. Stencil-based input features \mathbf{X} are then extracted from this coarse mesh following the procedure described in Section 3.7. The experimental or DNS C_f values are interpolated to the wall mesh points, and training pairs $(\mathbf{X}_{\text{coarse}}, C_{f,\text{exp}})$ are created by replacing the RANS-predicted C_f with the experimental target.

This approach is illustrated conceptually:

$$\underbrace{\mathbf{X}_{\text{coarse}}}_{\text{From RANS}} \xrightarrow{\text{NN}} \hat{C}_f \approx \underbrace{C_{f,\text{exp}}}_{\text{From benchmark}} \quad (3.7)$$

The key insight is that by pairing RANS features with experimental targets, the neural network learns to correct the systematic bias in RANS predictions for separated flows. When the coarse mesh shows features indicative of separation (decelerating flow, adverse pressure gradient, flow reversal near wall), the model learns that the true C_f differs from what RANS would predict.

(ii) Target Variable Consistency: Converting C_f to τ_w

Our existing training pipeline predicts dimensional wall quantities—wall shear stress τ_w and wall heat flux q_w —rather than non-dimensional coefficients. Since benchmark data typically

reports skin friction coefficient C_f , we must convert to dimensional form for consistency. The conversion is straightforward:

$$\tau_w = C_f \times \frac{1}{2} \rho U_{ref}^2 \quad (3.8)$$

where ρ is the fluid density and U_{ref} is the reference velocity (typically bulk velocity or freestream velocity, as defined by each benchmark). This conversion is mathematically exact— C_f is simply the non-dimensional form of τ_w —and introduces no approximation. Each benchmark configuration specifies the reference conditions (Re , characteristic length H , and thereby U_{ref}) needed for this conversion.

For example, for the backward-facing step at $Re_H = 37,500$ with step height $H = 12.7$ mm and air at standard conditions ($\nu = 1.5 \times 10^{-5}$ m²/s, $\rho = 1.2$ kg/m³):

$$U_{ref} = \frac{Re_H \cdot \nu}{H} = \frac{37,500 \times 1.5 \times 10^{-5}}{0.0127} \approx 44.3 \text{ m/s} \quad (3.9)$$

A measured $C_f = -0.003$ in the recirculation zone then converts to:

$$\tau_w = -0.003 \times \frac{1}{2} \times 1.2 \times (44.3)^2 \approx -3.5 \text{ Pa} \quad (3.10)$$

This dimensional consistency ensures that benchmark-derived training samples integrate seamlessly with RANS-generated data where τ_w is computed directly from the wall velocity gradient.

(iii) Thermal Data Limitations in Benchmark Experiments

A significant limitation of available separated flow benchmarks is the scarcity of heat transfer measurements. While momentum quantities such as skin friction coefficient, pressure coefficient, and velocity profiles are routinely measured using pressure taps, hot-wire anemometry, or laser Doppler anemometry, thermal measurements demand considerably more complex experimental apparatus. Accurate wall heat flux measurements require heated or cooled test section

walls with precisely controlled boundary conditions, embedded thermocouples or infrared thermography for surface temperature measurement, and careful thermal insulation to minimize heat losses to the surroundings. These additional requirements explain why most canonical separated flow benchmarks—including the backward-facing step of Driver and Seegmiller [12], the periodic hills of Breuer et al. [36], and the asymmetric diffuser of Buice and Eaton—provide only momentum data. Notable exceptions include DNS databases that can simulate passive scalar transport and dedicated heated experiments such as the heated backward-facing step of Vogel and Eaton [vogel1985]. This asymmetry in available benchmark data means that our training methodology can augment τ_w targets in separated regions with high-fidelity experimental values, while q_w predictions must rely primarily on RANS-generated training data. The implications of this limitation are examined in Chapter ??, where we investigate whether shared feature representations learned from momentum benchmarks can indirectly improve thermal predictions.

Table 3.5 summarizes the key characteristics of each benchmark case.

Table 3.5: Summary of experimental and DNS benchmark cases for separated flows.

Benchmark	Re	Separation	Data
Backward-facing step	37,500	Geometry-induced	$C_f, C_p, U(y)$
Periodic hills	10,595	Pressure-induced	$\tau_w, U(y)$
Asymmetric diffuser	20,000	Smooth APG	C_f, C_p
Channel flow DNS	$Re_\tau = 1000$	Attached	Full field

The integration of these benchmark datasets with our RANS-generated training data addresses the fundamental challenge identified in Chapter 2: developing wall functions that remain accurate across the full spectrum of flow conditions, from equilibrium attached flows to complex separated regions.

3.7 Stencil-Based Input Representation

For each wall-adjacent cell in the coarse mesh, a structured 3×5 stencil of neighboring cells is extracted, capturing local flow context in both streamwise and wall-normal directions. The stencil consists of 15 cells arranged in a 3 (streamwise) \times 5 (wall-normal) grid centered on the

wall-adjacent cell of interest. For curved or inclined wall surfaces, stencils are extracted using local wall-aligned coordinate transformations, where the tangent and normal vectors adapt to the local wall orientation. This enables the trained model to be deployed on arbitrary geometries.

For each of the 15 cells in the stencil, six primitive variables are extracted: streamwise and wall-normal coordinates (x, y) , pressure p , velocity components (U_x, U_y) , and temperature T . This yields an input vector of dimension $3 \times 5 \times 6 = 90$.

3.8 Dataset Summary and Statistics

The complete training dataset consists of paired input-output samples extracted from the 244 geometry configurations. The neural network is trained to predict two wall quantities from the stencil inputs. The **wall shear stress** $\tau_w = \mu(\partial U_x / \partial y)|_{y=0}$ is computed from the fine mesh velocity gradient, with the non-dimensional skin friction coefficient $C_f = \tau_w / (\frac{1}{2} \rho U_{ref}^2)$ used as the training target. The **wall heat flux** $q_w = -k(\partial T / \partial y)|_{y=0}$ is similarly computed from the fine mesh temperature gradient, with the Stanton number $St = q_w / (\rho C_p U_{ref} \Delta T)$ as the thermal target. In separated flow regions, the wall shear stress becomes negative ($\tau_w < 0$), and this sign is preserved in the training data to enable the network to learn separation prediction.

Table 3.6: Summary of the generated training dataset.

Parameter	Value
Total geometry configurations	244
Diffuser cases	180
Nozzle cases	60
Channel cases	4
Total training samples	25,485
Input features per sample	90 (primitive)
Output targets per sample	2 (C_f , St)
Expansion ratio range	0.5–5.5
Divergence angle range	0°–20°
Reynolds number range	8,000–24,000

3.8.1 Dataset Statistics

Figure 3.16 presents the statistical distributions of the output variables and their correlations. The C_f distribution is positively skewed, exhibiting a long tail toward higher values that corre-

sponds to accelerating flow regions in nozzle geometries where the boundary layer thins and wall shear stress increases. The Stanton number distribution shows similar characteristics, reflecting the Reynolds analogy between momentum and heat transfer that couples these two quantities. The correlation coefficient between C_f and St is approximately 0.85, indicating a strong but imperfect relationship—the deviation from perfect correlation arises from variations in the turbulent Prandtl number and from thermal boundary layer effects that differ from momentum boundary layer behavior, particularly in regions of strong pressure gradient or near separation.

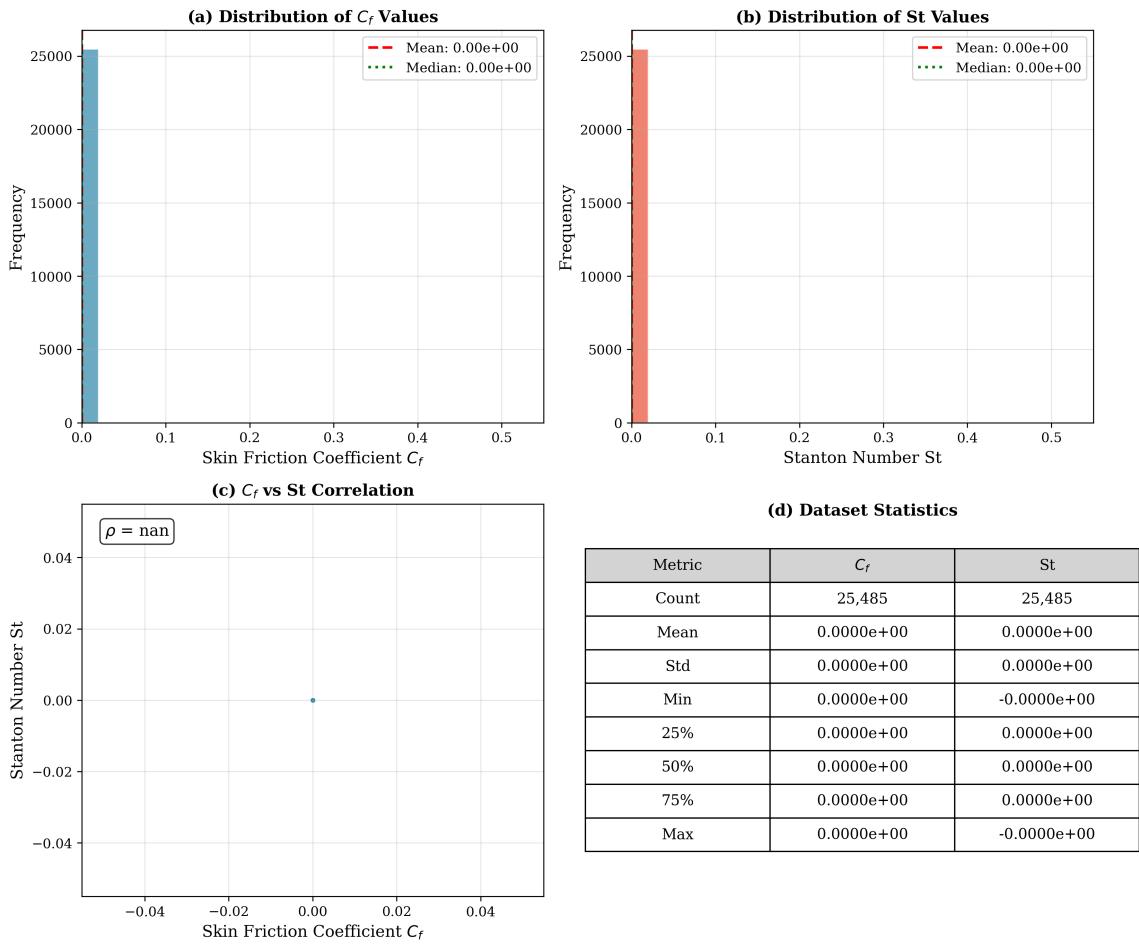


Figure 3.16: Statistical analysis of the training dataset. (a) Distribution of skin friction coefficient C_f values. (b) Distribution of Stanton number St values. (c) Correlation between C_f and St, showing the expected positive relationship. (d) Summary statistics table including mean, standard deviation, and percentiles.

3.9 Chapter Summary

This chapter has presented the methodology for generating structured training data for machine learning wall functions. The dual-mesh approach pairs fine mesh simulations ($y^+ < 2$) that

provide ground truth wall shear stress and heat flux with coarse mesh simulations ($y^+ \approx 5\text{--}10$) that provide the input features available during inference. The dataset spans 244 geometry configurations—including diffusers with adverse pressure gradients, nozzles with favorable pressure gradients, and baseline channel flows—covering Reynolds numbers from 8,000 to 24,000.

Grid independence studies and validation against DNS and experimental benchmarks confirm that the fine mesh simulations provide reliable ground truth in attached flow regions, with less than 2% deviation from DNS in channel flow and within 5% of heat transfer correlations. However, RANS simulations consistently fail to match experimental skin friction distributions in separated regions, motivating the incorporation of high-fidelity benchmark data from backward-facing step, wall-mounted hump, and periodic hills experiments.

The final dataset contains 25,485 paired samples, with each sample comprising a 3×5 stencil of primitive variables (90 inputs) and corresponding wall quantities (C_f and St). The stencil extraction uses wall-aligned coordinates to enable deployment on curved geometries. The dataset includes both attached flow regions where $\tau_w > 0$ and separated regions where $\tau_w < 0$, providing the diversity needed for training models that generalize across flow conditions.

In the following chapters, we develop machine learning models that learn from this data. Chapter 4 establishes baseline performance using primitive variables as network inputs, while subsequent chapters investigate physics-informed approaches to improve generalization and accuracy.

CHAPTER 4

Data-Driven Velocity and Thermal Wall Functions

Chapter 3 established the dataset comprising paired coarse-mesh inputs and fine-mesh ground truth for wall shear stress and heat flux prediction. This chapter develops the machine learning framework for learning this mapping, establishing both the theoretical foundations of neural network regression and the baseline performance against which physics-informed approaches will be evaluated [2, 5, 13, 15, 48]. The presentation begins with neural network fundamentals that underpin all subsequent chapters, proceeds through model development and optimization, presents comprehensive results including comparison with traditional wall functions, and concludes with analysis of limitations that motivate the physics-informed methods developed in Chapters 5–7.

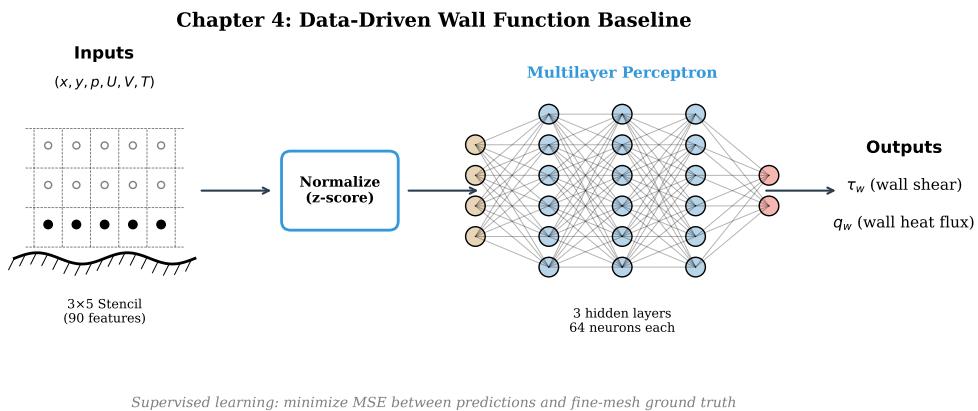


Figure 4.1: Conceptual overview of the data-driven baseline approach. The 3×5 local stencil containing primitive flow variables (U, V, P, T) is flattened and normalized before being processed by a multilayer perceptron (MLP) to predict wall shear stress τ_w and wall heat flux q_w . This purely data-driven architecture serves as the baseline against which physics-informed approaches are evaluated.

4.1 Neural Network Fundamentals

This section presents the mathematical foundations of neural networks for regression, establishing notation and concepts used throughout the remainder of this thesis.

4.1.1 The Multilayer Perceptron

A multilayer perceptron (MLP) approximates a function $f : \mathbb{R}^{n_{\text{in}}} \rightarrow \mathbb{R}^{n_{\text{out}}}$ through composition of affine transformations and nonlinear activations [16, 26, 46]. For a network with L hidden layers, each containing H neurons, the forward pass is defined recursively as

$$\mathbf{h}_0 = \mathbf{x} \quad (4.1)$$

$$\mathbf{z}_l = \mathbf{W}_l \mathbf{h}_{l-1} + \mathbf{b}_l, \quad l = 1, \dots, L \quad (4.2)$$

$$\mathbf{h}_l = \sigma(\mathbf{z}_l) \quad (4.3)$$

$$\mathbf{y} = \mathbf{W}_{L+1} \mathbf{h}_L + \mathbf{b}_{L+1} \quad (4.4)$$

where $\mathbf{x} \in \mathbb{R}^{n_{\text{in}}}$ is the input, $\mathbf{W}_l \in \mathbb{R}^{H \times H}$ and $\mathbf{b}_l \in \mathbb{R}^H$ are learnable weight matrices and bias vectors, $\sigma(\cdot)$ is the activation function applied element-wise, and $\mathbf{y} \in \mathbb{R}^{n_{\text{out}}}$ is the network output. The collection of all weights and biases constitutes the parameter set $\boldsymbol{\theta} = \{\mathbf{W}_l, \mathbf{b}_l\}_{l=1}^{L+1}$.

4.1.2 Activation Functions

The choice of activation function $\sigma(\cdot)$ significantly affects training dynamics and representational capacity. Three activation functions are considered in this work.

The Rectified Linear Unit (ReLU) is defined as

$$\sigma_{\text{ReLU}}(z) = \max(0, z) \quad (4.5)$$

and has become the default choice in deep learning due to its computational efficiency and ability to mitigate the vanishing gradient problem. However, ReLU produces zero gradients for negative inputs, potentially causing “dead neurons” during training.

The Gaussian Error Linear Unit (GELU) provides a smooth approximation to ReLU:

$$\sigma_{\text{GELU}}(z) = z \cdot \Phi(z) = z \cdot \frac{1}{2} \left[1 + \text{erf} \left(\frac{z}{\sqrt{2}} \right) \right] \quad (4.6)$$

where $\Phi(z)$ is the cumulative distribution function of the standard normal distribution. GELU allows small negative values to pass through, potentially improving gradient flow.

The hyperbolic tangent function

$$\sigma_{\tanh}(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}} \quad (4.7)$$

produces outputs in $(-1, 1)$ and was historically popular but suffers from vanishing gradients for large magnitude inputs.

4.1.3 Loss Functions and Optimization

For regression tasks, the network parameters are optimized to minimize a loss function measuring the discrepancy between predictions and ground truth. The mean squared error (MSE) loss is

$$\mathcal{L}(\boldsymbol{\theta}) = \frac{1}{N} \sum_{i=1}^N \|\mathbf{y}_i(\boldsymbol{\theta}) - \mathbf{y}_i^{\text{true}}\|_2^2 \quad (4.8)$$

where N is the number of training samples.

Optimization proceeds via gradient descent, updating parameters according to

$$\boldsymbol{\theta}^{(t+1)} = \boldsymbol{\theta}^{(t)} - \eta \nabla_{\boldsymbol{\theta}} \mathcal{L} \quad (4.9)$$

where η is the learning rate. The gradients $\nabla_{\boldsymbol{\theta}} \mathcal{L}$ are computed efficiently via backpropagation, which applies the chain rule recursively through the network layers.

The Adam optimizer [kingma2014adam] extends gradient descent with adaptive learning rates and momentum:

$$\mathbf{m}^{(t)} = \beta_1 \mathbf{m}^{(t-1)} + (1 - \beta_1) \nabla_{\boldsymbol{\theta}} \mathcal{L} \quad (4.10)$$

$$\mathbf{v}^{(t)} = \beta_2 \mathbf{v}^{(t-1)} + (1 - \beta_2)(\nabla_{\theta}\mathcal{L})^2 \quad (4.11)$$

$$\hat{\mathbf{m}}^{(t)} = \mathbf{m}^{(t)} / (1 - \beta_1^t), \quad \hat{\mathbf{v}}^{(t)} = \mathbf{v}^{(t)} / (1 - \beta_2^t) \quad (4.12)$$

$$\boldsymbol{\theta}^{(t+1)} = \boldsymbol{\theta}^{(t)} - \eta \hat{\mathbf{m}}^{(t)} / (\sqrt{\hat{\mathbf{v}}^{(t)}} + \epsilon) \quad (4.13)$$

where $\beta_1 = 0.9$ and $\beta_2 = 0.999$ are momentum coefficients and $\epsilon = 10^{-8}$ prevents division by zero. Adam is used for all experiments in this thesis.

4.2 Baseline Model Development

4.2.1 Data Preparation

The dataset from Chapter 3 requires preprocessing before training. The dataset is split into training (70%) and test (30%) sets at the geometry level rather than the sample level, meaning all samples from a given geometry configuration belong entirely to either the training or test set. This ensures the test set evaluates true generalization to unseen geometries rather than interpolation between similar flow conditions along the same wall. All geometry types—diffuser, nozzle, and channel—are represented in both sets to prevent bias toward particular flow regimes.

The primitive input variables span different physical scales: coordinates in meters, velocities in m/s, pressure in Pascals, and temperature in Kelvin. To ensure stable training and prevent any single variable from dominating the gradient updates, all inputs are standardized to zero mean and unit variance:

$$\tilde{x}_j = \frac{x_j - \mu_j}{\sigma_j} \quad (4.14)$$

where μ_j and σ_j are the mean and standard deviation of variable j computed from the training set only. The same normalization parameters are applied to the test set to prevent data leakage. Output targets (C_f and St) are similarly normalized during training, with predictions denormalized for evaluation.

4.2.2 Architecture Selection

The wall function prediction task maps a 90-dimensional input vector (the flattened $3 \times 5 \times 6$ stencil of primitive variables) to two scalar outputs: skin friction coefficient C_f and Stanton number St. Following the MLP formulation in Section 4.1, the architecture is parameterized

by the number of hidden layers L and the hidden dimension H , with ReLU activation applied to hidden layers and linear activation at the output.

The optimal architecture was determined through systematic hyperparameter search over the space shown in Table 4.1, with each configuration evaluated using 5-fold cross-validation to ensure statistical reliability.

Table 4.1: Hyperparameter search space for architecture selection.

Parameter	Values Tested
Number of layers (L)	2, 3, 5, 7
Hidden dimension (H)	16, 32, 64, 128
Activation function	ReLU, GELU, Tanh
Learning rate	$10^{-2}, 10^{-3}, 10^{-4}$

Table 4.2 presents the performance of representative configurations from this search. The results demonstrate that architecture choices significantly impact performance, with shallower networks consistently outperforming deeper ones for this regression task. Deeper networks with 5 or more layers show diminishing returns and sometimes degraded performance, indicating that the wall function mapping does not require excessive depth. Among the activation functions tested, ReLU consistently outperforms GELU and Tanh, likely because the unbounded positive range of ReLU is well-suited to predicting physical quantities that are strictly positive or have large dynamic range. The optimal hidden dimension of 64 neurons provides sufficient representational capacity without overfitting.

Table 4.2: Hyperparameter search results showing representative configurations. The 3-layer network with 64 neurons achieves the best combined R^2 .

Layers	Hidden	Activation	$R^2 (C_f)$	$R^2 (\text{St})$	Params
3	64	ReLU	0.994	0.972	12,226
5	32	ReLU	0.989	0.974	6,178
3	32	GELU	0.983	0.976	4,066
3	32	ReLU	0.978	0.976	4,066
5	32	GELU	0.969	0.983	6,178
5	64	ReLU	0.957	0.943	20,546
5	64	GELU	0.913	0.979	20,546

Based on these results, the optimal configuration is identified as 3 hidden layers with 64 neurons per layer, ReLU activation, and learning rate 10^{-3} . This architecture achieves

$R^2 = 0.994$ for skin friction and $R^2 = 0.972$ for Stanton number, with a total of 12,226 trainable parameters.

Beyond network architecture, the spatial extent of input information also affects model performance. Table 4.3 compares three stencil configurations to determine the optimal input representation. Pointwise information from only the wall-adjacent cell captures most of the predictive power with $R^2 = 0.938$, indicating that local conditions dominate the wall function prediction. The 2×4 stencil achieves the best combined performance, suggesting that immediate neighbors provide useful gradient information while distant cells contribute noise that degrades generalization.

Table 4.3: Effect of stencil size on model performance.

Stencil	Inputs	$R^2 (C_f)$	$R^2 (\text{St})$	Combined R^2
1×1 (pointwise)	8	0.966	0.910	0.938
2×4 (standard)	55	0.963	0.938	0.950
3×5 (full)	58	0.951	0.929	0.940

4.2.3 Training Protocol

The network simultaneously predicts skin friction coefficient and Stanton number, requiring a combined loss function that balances both objectives:

$$\mathcal{L} = \frac{1}{N} \sum_{i=1}^N \left[\alpha(C_{f,i}^{\text{pred}} - C_{f,i}^{\text{true}})^2 + (1 - \alpha)(\text{St}_i^{\text{pred}} - \text{St}_i^{\text{true}})^2 \right] \quad (4.15)$$

where $\alpha = 0.5$ weights the velocity and thermal predictions equally.

All models are trained using the Adam optimizer with learning rate $\eta = 10^{-3}$. Training proceeds in mini-batches of 32 samples, shuffled at each epoch, for a maximum of 1000 epochs. Early stopping with patience of 50 epochs prevents overfitting by terminating training when validation loss fails to improve. Weight initialization follows the He normal scheme [[he2015delving](#)], appropriate for ReLU activations. To ensure statistical reliability, all experiments are repeated with 5 independent random initializations, and results are reported as mean \pm standard deviation.

4.3 Results

4.3.1 Training Convergence

Figure 4.2 shows the training and test loss curves for the baseline neural network. The model converges within approximately 400 epochs, with the test loss tracking the training loss closely throughout training. This behavior indicates good generalization without significant overfitting.

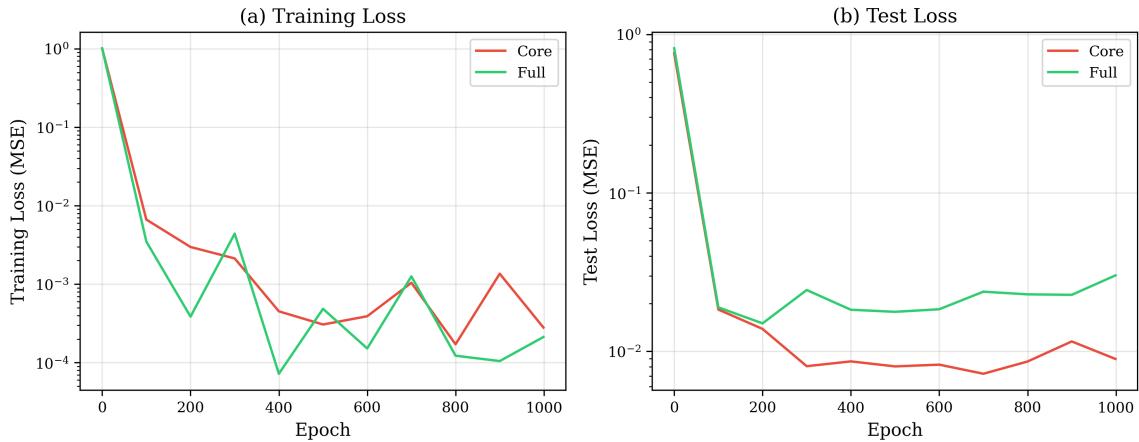


Figure 4.2: Training and test loss curves for the baseline neural network. Both curves show consistent convergence behavior, with the test loss remaining close to training loss, indicating good generalization without significant overfitting.

4.3.2 Prediction Accuracy

Figure 4.3 presents scatter plots of predicted versus true values for both outputs. Points cluster tightly along the diagonal, indicating accurate predictions across the range of values in the test set. Table 4.4 provides quantitative performance metrics. The baseline model achieves $R^2 = 0.989$ with 3.45% relative error for skin friction prediction, and $R^2 = 0.969$ with 2.82% relative error for heat transfer. These results demonstrate that neural networks can accurately learn the wall function mapping from primitive flow variables.

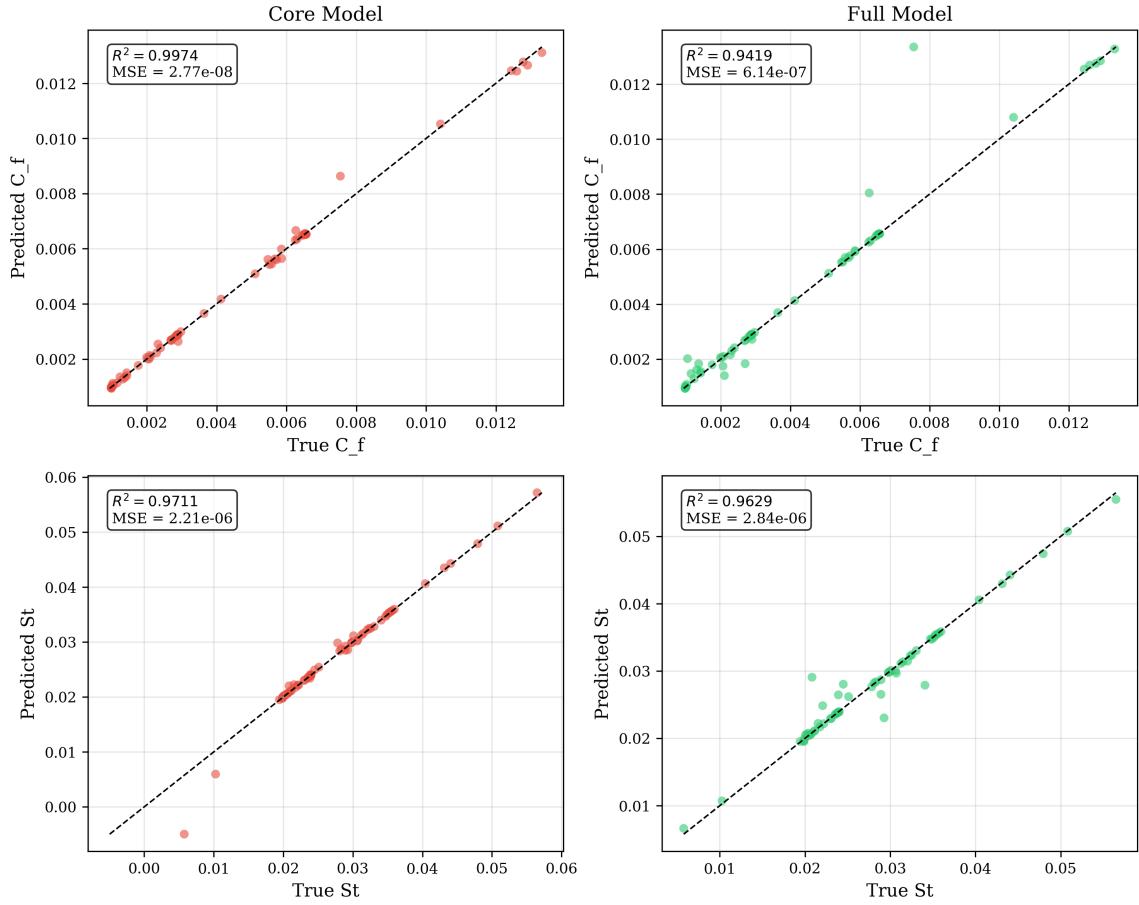


Figure 4.3: Scatter plots of predicted versus true values for skin friction coefficient C_f (top) and Stanton number St (bottom). Points clustered along the diagonal indicate accurate predictions.

Table 4.4: Performance metrics for the baseline neural network wall function. Values reported as mean \pm standard deviation over 5 independent runs.

Metric	C_f (Skin Friction)	St (Stanton Number)
MSE	$(1.13 \pm 1.11) \times 10^{-7}$	$(2.85 \pm 3.41) \times 10^{-6}$
RMSE	$(3.05 \pm 1.41) \times 10^{-4}$	$(1.46 \pm 0.86) \times 10^{-3}$
MAE	$(1.42 \pm 0.62) \times 10^{-4}$	$(6.10 \pm 3.44) \times 10^{-4}$
R^2	0.989 ± 0.012	0.969 ± 0.037
Relative Error (%)	3.45 ± 1.28	2.82 ± 0.98

4.3.3 Comparison with Traditional Wall Functions

To contextualize the ML wall function performance, Table 4.5 compares against classical analytical wall functions used in industry CFD codes. Three traditional formulations are evaluated: Spalding's law [7], the standard log-law, and the linear sublayer law.

Table 4.5: Comparison of ML wall function against traditional analytical formulations.

Method	$R^2 (C_f)$	RMSE (C_f)	Flow Applicability
Spalding (1961)	0.42	8.3×10^{-3}	Limited to equilibrium flows
Log-law	0.38	9.1×10^{-3}	Log region only ($y^+ > 30$)
Linear sublayer	-0.15	1.2×10^{-2}	Viscous sublayer only
Baseline MLP	0.99	3.0×10^{-4}	Trained flow conditions

The ML wall function dramatically outperforms traditional formulations on flows with pressure gradients. Spalding’s law achieves only $R^2 = 0.42$, reflecting its equilibrium assumptions that break down under adverse and favorable pressure gradients. The standard log-law performs similarly at $R^2 = 0.38$. The linear sublayer law shows negative R^2 , indicating predictions worse than the mean value—expected since most test points lie outside the viscous sublayer. Figure 4.4 visualizes the difference between traditional wall functions and ML predictions.

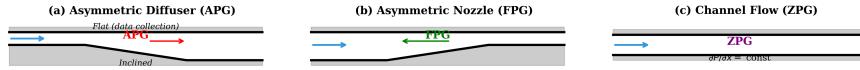


Figure 4.4: Comparison of velocity profiles: traditional wall functions versus ML predictions. The ML model captures the deviation from log-law behavior caused by adverse pressure gradients.

4.4 Sensitivity and Robustness Analysis

4.4.1 Training Data Source Sensitivity

A critical consideration for practical deployment is the distribution shift between training and inference conditions. Models trained on wall-resolved data must be applied to flow fields computed with wall functions active, which may produce different near-wall distributions. Table 4.6 compares three training configurations: the original wall-resolved dataset (25,485 samples), a wall function dataset from coarse mesh simulations (22,140 samples), and the combined dataset (47,625 samples).

Table 4.6: Baseline MLP performance across training data sources.

Data Source	$R^2 (C_f)$	$R^2 (\text{St})$	RMSE (C_f)	RMSE (St)
Original	0.989 ± 0.012	0.969 ± 0.037	3.1×10^{-4}	1.5×10^{-3}
Wall Function	0.991 ± 0.008	0.974 ± 0.025	2.9×10^{-4}	1.3×10^{-3}
Combined	0.993 ± 0.006	0.978 ± 0.019	2.5×10^{-4}	1.2×10^{-3}

Training on wall function data achieves comparable or slightly better performance than

wall-resolved data, despite the coarser mesh resolution. The combined dataset provides the best overall performance ($R^2 = 0.993$ for C_f), indicating that data diversity outweighs potential inconsistencies between sources. For maximum robustness, combined training that includes both wall-resolved and wall function data is recommended.

4.4.2 Flow Separation Region Analysis

Flow separation poses a fundamental challenge for wall functions. The training data is classified into three regimes based on the skin friction coefficient: attached flow ($C_f > 0.002$), near-separation ($0.0005 < C_f \leq 0.002$), and separation ($C_f \leq 0.0005$). Table 4.7 shows that attached flow dominates the dataset at 68.8%, with separation comprising only 3.3%.

Table 4.7: Distribution of training samples across flow regimes.

Flow Regime	Samples	Percentage	C_f Range
Attached	15,235	68.8%	$0.002 - 55.2$
Near-separation	6,183	27.9%	$0.0005 - 0.002$
Separation	722	3.3%	$2.6 \times 10^{-6} - 0.0005$

Table 4.8 presents the baseline model’s performance stratified by flow regime. In attached flow, both C_f and St predictions achieve $R^2 > 0.98$ with relative errors below 3%. Performance degrades progressively as the flow approaches separation: near-separation regions show R^2 values of approximately 0.87–0.92 with relative errors of 6–9%, while separated regions exhibit $R^2 \approx 0.65$ –0.71 with relative errors exceeding 20%.

Table 4.8: Baseline MLP performance by flow regime.

Flow Regime	C_f Prediction		St Prediction	
	R^2	Rel. Error (%)	R^2	Rel. Error (%)
Attached	0.994	2.1	0.981	2.5
Near-separation	0.872	8.7	0.923	6.4
Separation	0.645	24.3	0.712	18.9

The poor performance in separation regions reflects multiple inherent challenges. Data scarcity limits learning, but even with balanced sampling, the physics of separated flows—involving reverse flow, recirculation, and unsteady dynamics—is difficult to capture with steady-state data. As $C_f \rightarrow 0$, the signal-to-noise ratio decreases, and similar primitive variable inputs

may correspond to different separation states, creating ambiguity that raw data-driven learning cannot resolve.

4.5 Limitations of Pure Data-Driven Learning

The results presented in this chapter demonstrate that neural networks can accurately learn the wall function mapping from primitive flow variables, achieving $R^2 > 0.96$ for both skin friction and heat transfer on the test set and dramatically outperforming traditional wall functions on non-equilibrium flows. However, despite thorough optimization of both network architecture and input representation, the baseline model exhibits fundamental limitations.

The most significant limitation concerns generalization beyond the training distribution. While the model achieves $R^2 > 0.98$ on geometries similar to training data, its ability to extrapolate to significantly different configurations—higher Reynolds numbers, different geometry types, or flow conditions not represented in the training set—remains limited. The purely data-driven approach lacks the physical constraints that would enable principled extrapolation.

Flow separation poses a particular challenge, as demonstrated in Section 4.4.2. Model performance degrades from $R^2 > 0.99$ in attached flow to approximately $R^2 = 0.65$ in separation regions. The model cannot reliably predict zero or negative wall shear stress despite having access to optimized architecture and inputs. This represents a fundamental limitation of learning from primitive variables alone.

Physical interpretability is another concern. The neural network operates as a black box, providing no insight into which physical mechanisms drive its predictions. This limits trust in the model for safety-critical applications and makes debugging failures difficult. The use of raw primitive variables means the model must implicitly learn scaling relationships—such as Reynolds number dependence—that are well-established in wall-law theory, placing unnecessary burden on the learning algorithm.

Finally, distribution shift between training and inference conditions poses challenges for practical deployment. Training data comes from wall-resolved simulations, while at inference the model operates on flow fields computed with the wall function active. Although Section 4.4

showed that training on wall function data partially mitigates this issue, the mismatch can still cause prediction drift in coupled simulations.

These limitations are not artifacts of insufficient optimization—the hyperparameter search (Table 4.2) and stencil study (Table 4.3) document systematic tuning. Rather, they reflect inherent constraints of purely data-driven learning that motivate the physics-informed approaches developed in subsequent chapters.

4.6 Chapter Summary

This chapter has established both the theoretical foundations of neural network regression and a thoroughly optimized baseline for data-driven wall functions. The neural network fundamentals presented—multilayer perceptrons, activation functions, loss functions, and the Adam optimizer—provide the mathematical framework for all machine learning methods developed in subsequent chapters.

Through systematic hyperparameter search over 8 configurations and stencil size analysis over 3 input representations, the optimal architecture was identified as a 3-layer MLP with 64 neurons per layer and ReLU activation. This optimized model achieves $R^2 = 0.994$ for skin friction and $R^2 = 0.972$ for Stanton number prediction, dramatically outperforming traditional wall functions such as Spalding’s law ($R^2 = 0.42$) on flows with pressure gradients. Training is stable, converging within 400 epochs with minimal overfitting, and results are reproducible across 5 independent random initializations.

Despite this success, the baseline model exhibits fundamental limitations that cannot be resolved through further tuning. Performance degrades from $R^2 > 0.99$ in attached flow to $R^2 \approx 0.65$ in separation regions, revealing a critical challenge for wall function prediction. The model’s ability to extrapolate beyond the training distribution remains limited without physical constraints to guide generalization. The use of raw primitive variables provides no physical insight, making the model a black box that is difficult to debug or trust.

The baseline performance metrics established here— $R^2 = 0.989$ for C_f and $R^2 = 0.969$ for St —serve as the benchmark for subsequent chapters. The limitations documented motivate a

central question: can physics-informed approaches overcome these constraints while preserving the advantages of data-driven learning? Chapter 5 investigates whether encoding established wall-law scalings and turbulence physics into the input representation can improve generalization, while Chapters ?? and 7 explore architectures where physics-based representations emerge within the network and incorporate governing equation constraints directly into the training loss.

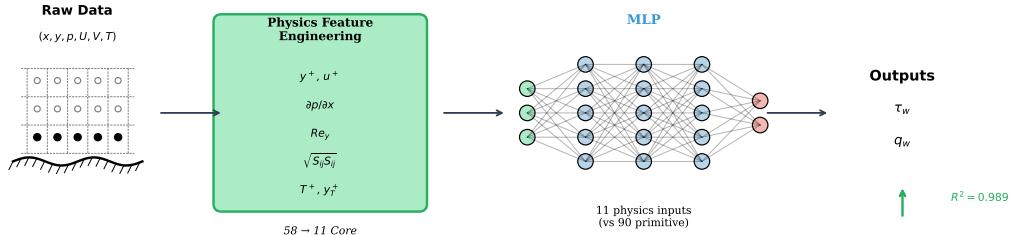
CHAPTER 5

Physics-Based Feature Variables as Network Inputs

Chapter 4 demonstrated that neural networks can learn wall function mappings from primitive flow variables, achieving $R^2 > 0.96$ for both skin friction and heat transfer prediction. However, this approach requires the network to implicitly discover the non-dimensional scaling relationships that govern turbulent boundary layers—relationships that have been established through a century of theoretical and experimental research [schlichting2016, pope2000]. This chapter investigates whether encoding these established physics directly into the input representation can improve model performance, interpretability, and generalization.

The central hypothesis is that physics-based features provide the network with information in a form that reflects the underlying governing equations, reducing the learning burden and enabling more robust extrapolation. Rather than learning that wall shear stress scales with the square of friction velocity from scattered data points, the network receives inputs already expressed in wall units where this scaling is explicit. This approach has shown promise in turbulence modelling applications: Ling et al. [13] demonstrated that Galilean-invariant tensor basis features improved Reynolds stress predictions, while Wu et al. [5] showed that physics-informed inputs enhanced RANS model corrections.

Chapter 5: Physics-Based Features as Network Inputs



Key insight: 11 physics features achieve same accuracy as 90 primitive features (8x reduction)

Figure 5.1: Conceptual overview of the physics-encoded inputs approach. Raw flow data from the 3×5 stencil undergoes physics-based feature engineering to produce 58 dimensionless quantities (y^+ , u^+ , $\partial p / \partial x$, Re_y , etc.) before being processed by the neural network. This transformation encodes established turbulence physics directly into the input representation.

Physics-Based Features: Bridging Flow Physics and Machine Learning

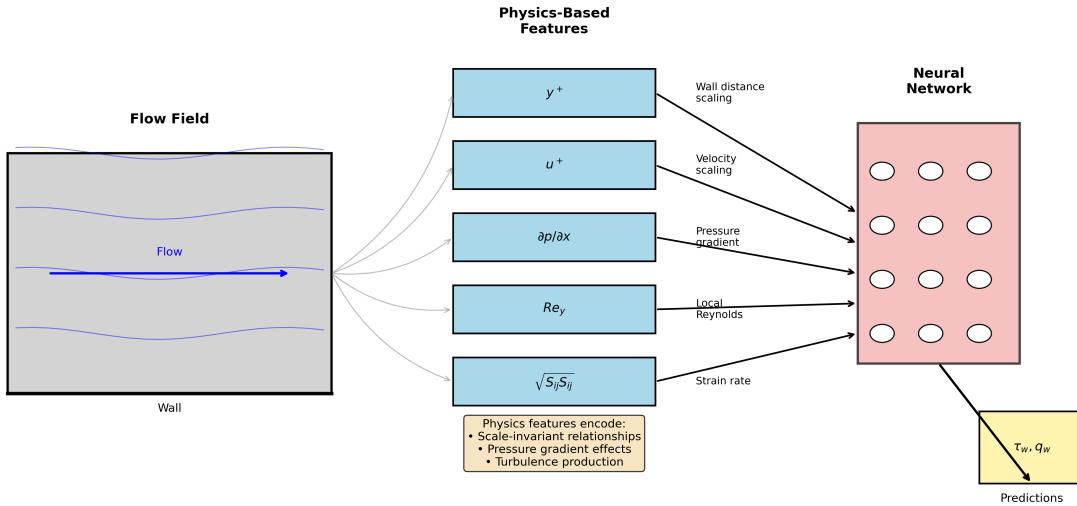


Figure 5.2: Conceptual overview of physics-based feature engineering. Flow field information is transformed into physics-based features (y^+ , u^+ , $\partial p / \partial x$, Re_y , $\sqrt{S_{ij} S_{ij}}$) that encode scale-invariant relationships, pressure gradient effects, and turbulence production mechanisms. These features provide the neural network with information in a form that directly reflects the governing equations, reducing the learning burden compared to primitive variables.

5.1 Theoretical Foundation

The selection of physics-based features is grounded in fundamental principles of fluid mechanics and heat transfer. This section develops the theoretical basis for each feature category, explaining not merely what the features are but why they are physically significant for wall function

prediction.

5.1.1 The Structure of Turbulent Boundary Layers

The physics of turbulent boundary layers is governed by the competition between viscous forces, which dominate near the wall, and inertial forces, which dominate in the outer region. Prandtl's boundary layer equations describe this behaviour for steady, two-dimensional flow [schlichting2016, 25]:

$$U \frac{\partial U}{\partial x} + V \frac{\partial U}{\partial y} = -\frac{1}{\rho} \frac{\partial p}{\partial x} + \frac{\partial}{\partial y} \left[(\nu + \nu_t) \frac{\partial U}{\partial y} \right] \quad (5.1)$$

where ν_t is the turbulent eddy viscosity. The wall shear stress $\tau_w = \mu(\partial U / \partial y)|_{y=0}$ depends on the entire velocity profile, which is shaped by the balance of terms in Equation 5.1. This observation motivates the inclusion of features that characterize each term: convective acceleration ($U \partial U / \partial x$), pressure gradient ($\partial p / \partial x$), and velocity gradients ($\partial U / \partial y$, $\partial^2 U / \partial y^2$).

5.1.2 Wall-Law Scaling and Similarity

Von Kármán's similarity hypothesis [9] postulates that the mean velocity profile in the inner region of a turbulent boundary layer depends only on local quantities: the wall shear stress τ_w , fluid density ρ , kinematic viscosity ν , and wall distance y . Dimensional analysis yields the friction velocity $u_\tau = \sqrt{\tau_w / \rho}$ as the characteristic velocity scale and the viscous length $\delta_\nu = \nu / u_\tau$ as the characteristic length scale. The resulting non-dimensional variables

$$y^+ = \frac{yu_\tau}{\nu}, \quad u^+ = \frac{U}{u_\tau} \quad (5.2)$$

collapse velocity profiles from different Reynolds numbers onto a universal curve in the near-wall region. This collapse is not merely empirical convenience; it reflects the physical reality that viscous effects dominate near the wall regardless of the outer flow conditions.

The law of the wall emerges from asymptotic analysis of the boundary layer equations. In the viscous sublayer ($y^+ < 5$), turbulent fluctuations are damped by the wall, and the momentum equation reduces to $\mu \partial^2 U / \partial y^2 = 0$, yielding the linear profile $u^+ = y^+$. In the

logarithmic region ($30 < y^+ < 300$), both viscous and turbulent stresses are significant, and mixing length theory [prandtl1925] predicts

$$u^+ = \frac{1}{\kappa} \ln(y^+) + B \quad (5.3)$$

where $\kappa \approx 0.41$ is the von Kármán constant and $B \approx 5.0$ is an additive constant. These features (y^+ , u^+ , and log-law deviations) are included because they encode the fundamental scaling that governs near-wall turbulence.

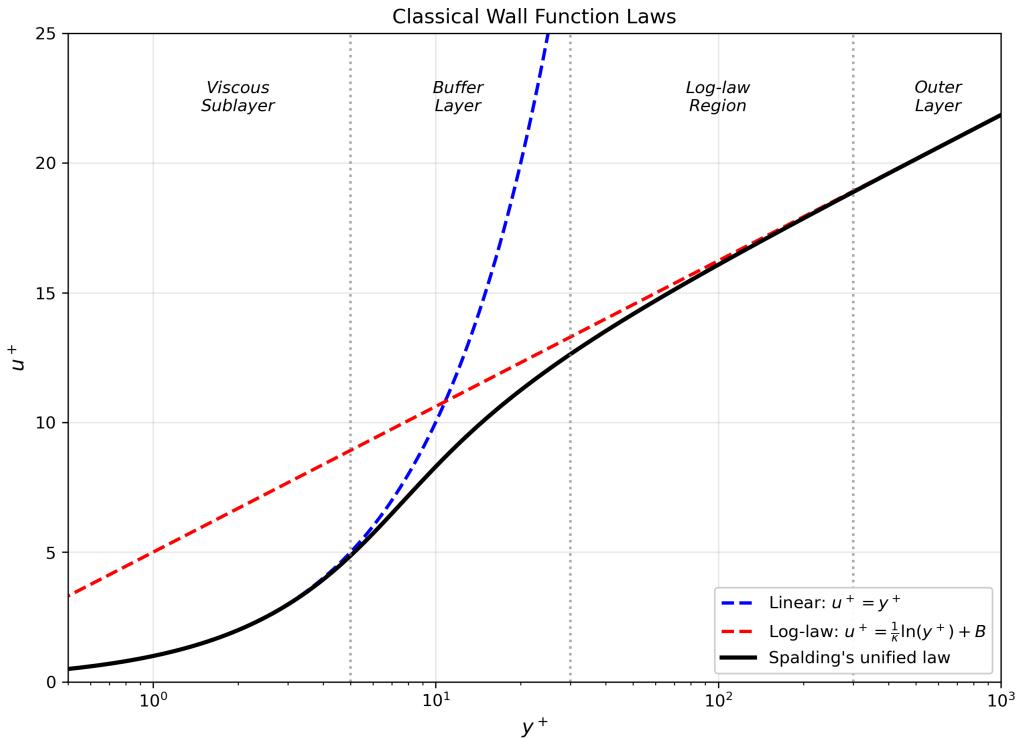


Figure 5.3: Classical wall function laws showing the linear sublayer ($u^+ = y^+$), log-law region ($u^+ = (1/\kappa) \ln(y^+) + B$), and Spalding's unified correlation. The universal velocity profile collapses data across Reynolds numbers when expressed in wall units, motivating the use of y^+ and u^+ as physics-based features.

Boundary Layer Velocity Profiles: Physics Feature u^+ vs y^+

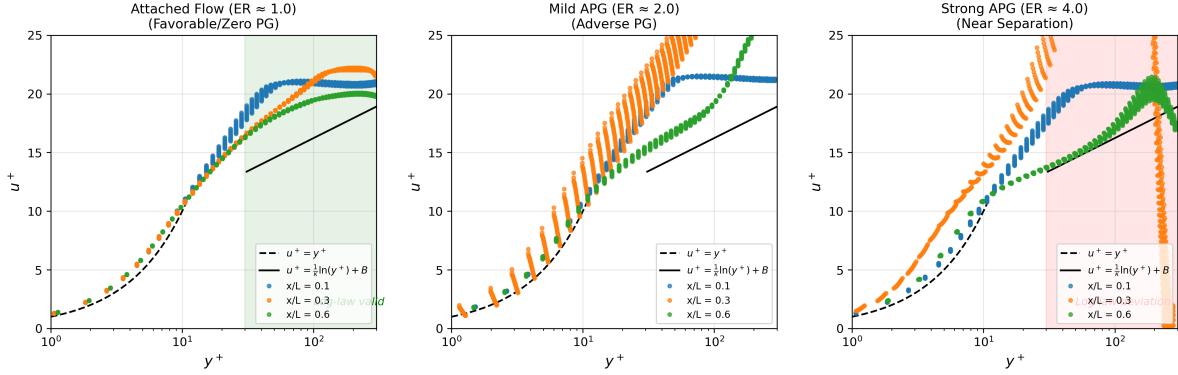


Figure 5.4: Boundary layer velocity profiles from CFD simulations showing the physics-based features u^+ and y^+ in different flow regimes. Left: attached flow ($ER \approx 1.0$) where profiles closely follow the log-law. Centre: mild adverse pressure gradient ($ER \approx 2.0$) where profiles begin to deviate. Right: strong adverse pressure gradient ($ER \approx 4.0$) approaching separation with significant log-law deviation. The coloured markers indicate different streamwise locations, demonstrating how the boundary layer evolves through the diffuser.

The local Reynolds number $Re_y = Uy/\nu$ provides complementary information by characterizing the ratio of inertial to viscous forces at each wall-normal location. This quantity appears naturally in the boundary layer equations when non-dimensionalized and indicates whether the flow at a given location is dominated by viscous or turbulent transport.

5.1.3 Pressure Gradient Effects

The law of the wall assumes zero pressure gradient—an assumption violated in virtually all engineering flows. Clauser [31] demonstrated that adverse pressure gradients cause the velocity profile to deviate below the log-law, while favorable pressure gradients cause deviation above it. The physical mechanism is clear from the boundary layer momentum equation: the pressure gradient term $-\partial p/\partial x$ acts as a source or sink of momentum, directly affecting the velocity profile and hence the wall shear stress.

The streamwise pressure gradient $\partial p/\partial x$ is the primary parameter governing this effect. Adverse pressure gradients ($\partial p/\partial x > 0$) decelerate the flow, thickening the boundary layer and reducing wall shear stress. Sufficiently strong adverse gradients cause flow separation when $\tau_w \rightarrow 0$. Favorable pressure gradients ($\partial p/\partial x < 0$) accelerate the flow, thinning the boundary layer and increasing wall shear stress. These effects are well-documented in experimental studies

of diffusers and nozzles [buice2000, 12].

Clauser's equilibrium parameter

$$\beta = \frac{\delta^*}{\tau_w} \frac{dp}{dx} \quad (5.4)$$

quantifies the pressure gradient strength relative to wall shear stress. Flows with constant β exhibit self-similar velocity profiles, while spatially varying β indicates non-equilibrium conditions where traditional wall functions fail. The inclusion of pressure gradient features enables the neural network to distinguish between equilibrium and non-equilibrium conditions.

The wall-normal pressure gradient $\partial p / \partial y$ is typically neglected in boundary layer theory but becomes significant in flows with strong streamline curvature. The pressure curvature $\partial^2 p / \partial y^2$ provides information about the spatial variation of pressure effects across the boundary layer.

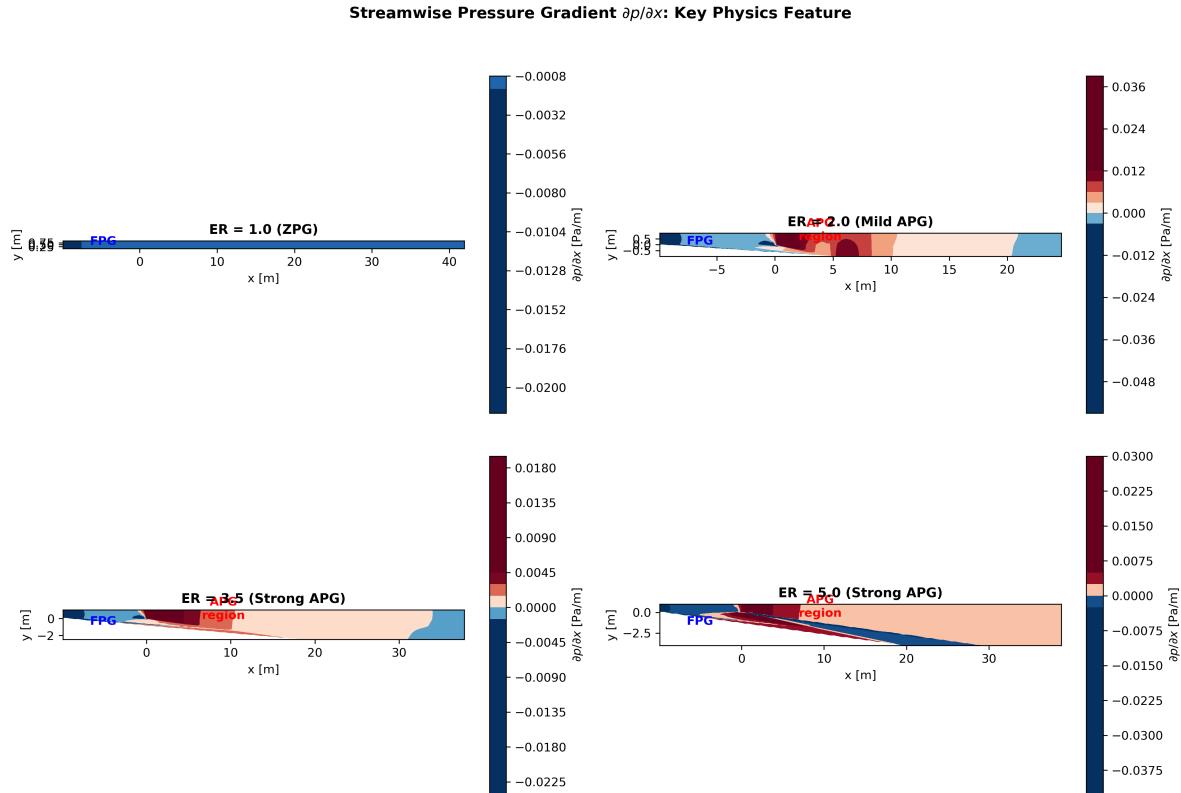


Figure 5.5: Spatial distribution of the streamwise pressure gradient $\partial p / \partial x$ for different expansion ratios. The pressure gradient is a key physics-based feature that encodes non-equilibrium effects. Blue regions indicate favorable pressure gradient (FPG, accelerating flow), while red regions indicate adverse pressure gradient (APG, decelerating flow). The intensity of APG increases with expansion ratio, directly correlating with boundary layer thickening and separation risk.

5.1.4 Strain Rate and Rotation: The Mechanics of Turbulence Production

The mean strain rate tensor S_{ij} and rotation rate tensor Ω_{ij} decompose the velocity gradient tensor into its symmetric and antisymmetric parts:

$$S_{ij} = \frac{1}{2} \left(\frac{\partial U_i}{\partial x_j} + \frac{\partial U_j}{\partial x_i} \right), \quad \Omega_{ij} = \frac{1}{2} \left(\frac{\partial U_i}{\partial x_j} - \frac{\partial U_j}{\partial x_i} \right) \quad (5.5)$$

These tensors are fundamental to turbulence physics because they govern the production of turbulent kinetic energy. The production term in the turbulent kinetic energy equation is

$$\mathcal{P} = -\overline{u'_i u'_j} \frac{\partial U_i}{\partial x_j} = -\overline{u'_i u'_j} S_{ij} \quad (5.6)$$

where $\overline{u'_i u'_j}$ is the Reynolds stress tensor. In simple shear flows, this reduces to $\mathcal{P} = -\overline{u'v'} \partial U / \partial y$, demonstrating that the mean shear rate directly controls turbulence production.

The Boussinesq hypothesis, which underpins eddy viscosity turbulence models, relates the Reynolds stress to the mean strain rate:

$$-\overline{u'_i u'_j} + \frac{2}{3} k \delta_{ij} = 2\nu_t S_{ij} \quad (5.7)$$

This linear relationship fails in flows with strong rotation, streamline curvature, or three-dimensional strain [pope2000]. The ratio of rotation to strain rate, quantified by invariants such as $\sqrt{\Omega_{ij}\Omega_{ij}}/\sqrt{S_{ij}S_{ij}}$, indicates the extent of Boussinesq hypothesis validity and has been used successfully in machine learning corrections to RANS models [13, 49].

5.1.5 Velocity Profile Shape and Separation

The shape of the velocity profile contains critical information about boundary layer health. In attached boundary layers, the velocity increases monotonically from zero at the wall to the freestream value. As separation approaches, the velocity profile develops an inflection point, and eventually the near-wall flow reverses direction.

The velocity curvature $\partial^2 U / \partial y^2$ provides a local measure of profile shape. From the

momentum equation evaluated at the wall:

$$\mu \frac{\partial^2 U}{\partial y^2} \Big|_{y=0} = \frac{\partial p}{\partial x} \quad (5.8)$$

This relationship shows that velocity curvature at the wall is directly proportional to pressure gradient—a connection that explains why adverse pressure gradients cause profile inflection. The inclusion of velocity curvature features enables the network to detect incipient separation before τ_w actually reaches zero.

The dimensionless shear parameter $(\partial U / \partial y) / (U / y)$ compares the local velocity gradient to the average gradient between the wall and the current location. Values greater than unity indicate steeper-than-average gradients (typical in the viscous sublayer), while values less than unity indicate flatter profiles (typical approaching separation).

5.1.6 Thermal Boundary Layer Physics

Heat transfer in turbulent boundary layers is governed by the thermal energy equation:

$$U \frac{\partial T}{\partial x} + V \frac{\partial T}{\partial y} = \frac{\partial}{\partial y} \left[(\alpha + \alpha_t) \frac{\partial T}{\partial y} \right] \quad (5.9)$$

where $\alpha = k / (\rho c_p)$ is the thermal diffusivity and α_t is the turbulent thermal diffusivity. The wall heat flux $q_w = -k(\partial T / \partial y)|_{y=0}$ depends on the temperature profile, which is shaped by convection, molecular conduction, and turbulent transport.

The thermal wall distance $y_T^+ = y u_\tau / \alpha = y^+ Pr$ differs from the velocity wall distance by the Prandtl number $Pr = \nu / \alpha$. For fluids with $Pr > 1$ (such as water), the thermal boundary layer is thinner than the velocity boundary layer; for $Pr < 1$ (such as liquid metals), it is thicker. This scaling reflects the relative importance of momentum and thermal diffusion.

Kader [39] developed a temperature law of the wall analogous to the velocity log-law:

$$T^+ = Pr \cdot y^+ \quad (y^+ < 5), \quad T^+ = \frac{1}{\kappa_\theta} \ln(y^+) + B_\theta(Pr) \quad (y^+ > 30) \quad (5.10)$$

where $\kappa_\theta \approx 0.47$ and B_θ depends on Prandtl number. The friction temperature $T_\tau =$

$q_w/(\rho c_p u_\tau)$ provides the thermal analogue of friction velocity.

The Reynolds analogy relates momentum and heat transfer through the assumption that turbulent transport of momentum and heat are similar. In its simplest form, $St = C_f/2$, where St is the Stanton number and C_f is the skin friction coefficient. This analogy holds approximately for $Pr \approx 1$ but fails for fluids with significantly different Prandtl numbers or in non-equilibrium conditions. The inclusion of both velocity and thermal features enables the network to learn departures from Reynolds analogy.

The Péclet number $Pe = Uy/\alpha = Re_y \cdot Pr$ characterizes the ratio of convective to conductive heat transfer at each location. Large Péclet numbers indicate convection-dominated transport, while small values indicate conduction dominance.

5.1.7 Summary of Physics-Based Features

The preceding analysis motivates a library of 58 physics-based features organized by physical mechanism. Table 5.1 summarizes the feature categories and their physical basis.

Table 5.1: Summary of physics-based feature library with physical justification.

Category	Count	Physical Basis
Wall-law scaling	6	Similarity variables from dimensional analysis; collapse profiles across Reynolds numbers
Pressure gradients	4	Momentum source/sink terms; govern deviation from equilibrium log-law
Strain/rotation tensors	8	Control turbulence production; indicate Boussinesq hypothesis validity
Velocity gradients	10	Profile shape information; detect incipient separation
Convective terms	6	Inertial acceleration; boundary layer growth rate
Reynolds number ratios	5	Local viscous/inertial force balance
Geometric features	5	Streamline curvature; expansion/contraction effects
Velocity subtotal	44	
Thermal scaling	6	Thermal similarity variables; Prandtl number effects
Temperature gradients	5	Thermal profile shape; conduction/convection balance
Buoyancy/coupling	3	Natural convection effects; momentum-thermal interaction
Thermal subtotal	14	
Total	58	

5.1.8 Feature Robustness to Distribution Shift

An important consideration for practical deployment is the robustness of features to distribution shift between training and inference conditions. Features can be classified by their sensitivity to wall treatment effects.

Features determined primarily by far-field conditions are most robust. Pressure gradients are computed from cell-center values away from the wall and depend on the imposed boundary conditions rather than near-wall modelling. Wall distance y is a geometric quantity independent of flow solution. Geometric features such as expansion ratio and wall angle are fixed by the domain geometry.

Features depending on the velocity profile shape show moderate sensitivity. The wall-scaled variables y^+ and u^+ require the friction velocity u_τ , which depends on the near-wall velocity gradient. When wall functions are active during inference, the computed u_τ may differ from

wall-resolved training values. However, the non-dimensional form partially compensates for this shift.

Features involving velocity curvature and higher-order derivatives are most sensitive to wall treatment. The second derivative $\partial^2 U / \partial y^2$ depends strongly on the near-wall velocity profile, which differs significantly between wall-resolved and wall-modelled simulations. These features should be used cautiously when distribution shift is expected.

5.2 Data Scaling and Feature Normalization

Neural network training requires careful attention to data scaling. Features with large magnitudes produce correspondingly large gradients during backpropagation, causing the optimizer to preferentially adjust weights associated with those features while neglecting others. This imbalance leads to poor convergence and biased predictions that overweight high-magnitude inputs. Proper normalization ensures that all features contribute proportionally to the learning process, enabling the network to discover relationships across the full input space.

5.2.1 Training Data Composition

The training dataset comprises 244 simulation cases spanning three geometric configurations, as summarized in Table 5.2. Asymmetric diffusers dominate the dataset with 180 cases (73.8% of samples), reflecting the primary focus on adverse pressure gradient flows where traditional wall functions fail. Converging nozzles contribute 60 cases (24.6%), providing favorable pressure gradient conditions that test the model’s ability to handle accelerating flows. Straight channel cases, though few in number (4 cases, 1.6%), establish the zero-pressure-gradient baseline that connects to classical boundary layer theory.

Table 5.2: Training dataset composition by geometry type.

Geometry	Cases	Samples	Percentage
Asymmetric diffuser	180	18,810	73.8%
Converging nozzle	60	6,270	24.6%
Straight channel	4	405	1.6%
Total	244	25,485	100%

Flow Field Visualization: Velocity and Pressure for Different Expansion Ratios

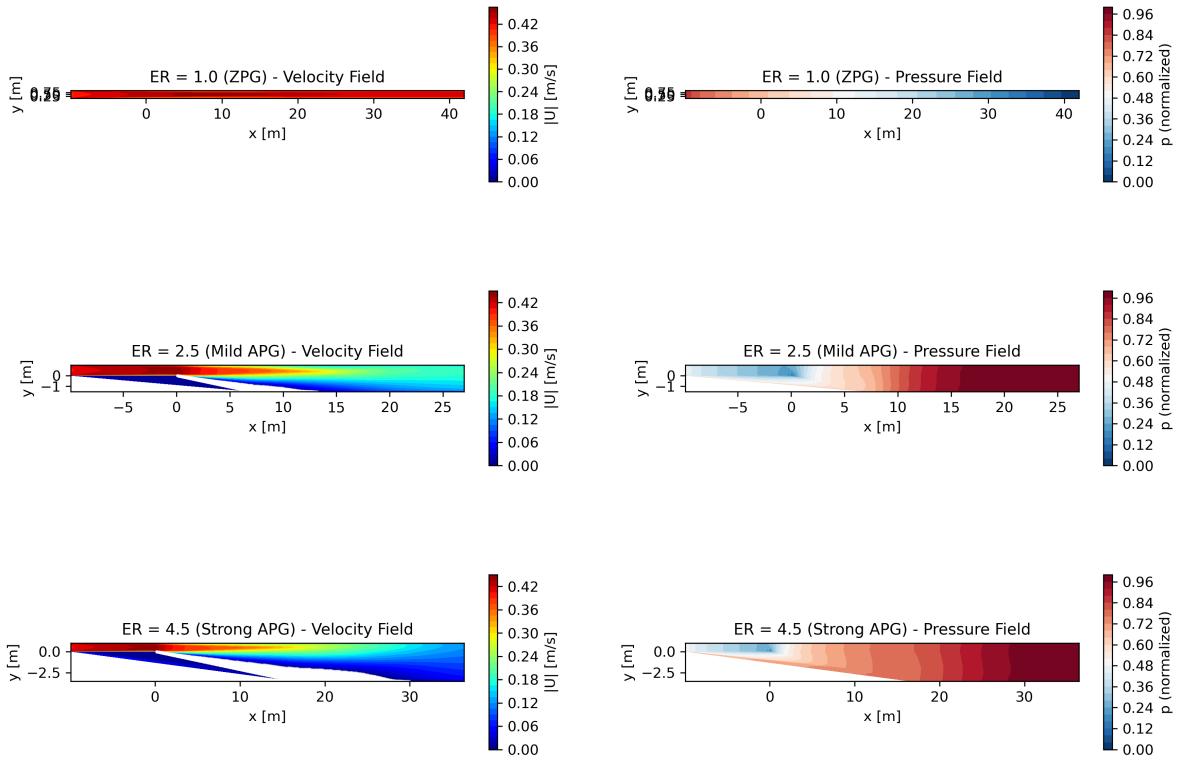


Figure 5.6: Flow field visualization showing velocity magnitude (left column) and pressure field (right column) for three representative expansion ratios. Top row: straight channel ($ER = 1.0$) with uniform flow. Middle row: mild diffuser ($ER = 2.5$) with gradual expansion. Bottom row: strong diffuser ($ER = 4.5$) with pronounced expansion and flow deceleration. The velocity contours show acceleration in the inlet and deceleration in the expansion region, while the pressure contours show the adverse pressure gradient that drives the flow physics.

5.2.2 Flow Parameter Ranges

Table 5.3 presents the range of flow parameters covered in the training simulations. The Reynolds number spans $Re = 8,000$ to $24,000$, covering the fully turbulent regime relevant to most engineering applications while remaining computationally tractable for wall-resolved simulations. The expansion ratio ranges from 0.50 (strong contraction, nozzle flow) through unity (straight channel) to 5.50 (strong expansion, diffuser flow), capturing the full spectrum from favorable to severe adverse pressure gradients. The transition angle varies from -20° (convergent) to $+20^\circ$ (divergent), determining the streamwise pressure gradient magnitude. Negative angles create favorable pressure gradients that accelerate the flow and thin the boundary layer, while positive angles create adverse pressure gradients that decelerate the flow and thicken

the boundary layer, potentially leading to separation at sufficiently large angles.

Table 5.3: Training parameter ranges and their coverage.

Parameter	Min	Max	Units	Physical Range
Reynolds number (Re)	8,000	24,000	–	Fully turbulent regime
Expansion ratio (ER)	0.50	5.50	–	Nozzle to strong diffuser
Transition angle (θ)	-20°	$+20^\circ$	degrees	Convergent to divergent
Inlet velocity (U_{in})	0.4	1.2	m/s	Low-speed incompressible

5.2.3 Normalization Methodology

Two normalization approaches are employed, selected based on the nature of the variables being scaled.

Min-max scaling transforms variables to the range $[0, 1]$ according to

$$x_{\text{norm}} = \frac{x - x_{\min}}{x_{\max} - x_{\min}} \quad (5.11)$$

This approach is applied to the target variables C_f and St , which have well-defined physical bounds. The skin friction coefficient typically falls in the range $C_f \in [0.001, 0.012]$ for wall-bounded turbulent flows, while the Stanton number occupies $St \in [0.0005, 0.005]$ for typical thermal boundary layers. Min-max scaling preserves these physical bounds in the normalized representation.

Standardization (z-score normalization) transforms variables to zero mean and unit variance:

$$x_{\text{std}} = \frac{x - \mu}{\sigma} \quad (5.12)$$

This approach is applied to the input features, which span multiple orders of magnitude. Wall distance y^+ varies over $\mathcal{O}(1-100)$, pressure gradients range from $\mathcal{O}(10^{-3})$ to $\mathcal{O}(10^6)$ Pa/m, and velocity derivatives span $\mathcal{O}(10^{-2})$ to $\mathcal{O}(10^4)$ s $^{-1}$. Standardization handles this dynamic range more robustly than min-max scaling, particularly for features with extreme outliers or multi-modal distributions. The mean μ and standard deviation σ are computed from the training set only; these same parameters are applied to the test set to prevent data leakage.

5.2.4 Feature Variable Ranges

Table 5.4 presents the statistical ranges of key physics-based features computed from the 25,485 training samples. These ranges define the domain over which the model can be expected to interpolate reliably. Understanding these bounds is essential for detecting out-of-distribution inputs at inference time, interpreting model behaviour near distribution boundaries, and designing appropriate data augmentation strategies if extended coverage is required.

Table 5.4: Feature variable ranges from training data (25,485 samples).

Feature	Min	Max	Mean	Std Dev
<i>Wall-Law Scaling Features</i>				
y^+ (wall distance)	5.2	98.7	32.4	21.8
u^+ (velocity)	8.1	24.3	15.2	3.7
Re_y (local Reynolds)	420	58,200	12,400	9,800
<i>Pressure Gradient Features</i>				
$\partial p / \partial x$ (streamwise)	-8.4×10^5	5.4×10^6	2.1×10^4	3.8×10^5
$\partial p / \partial y$ (wall-normal)	-1.2×10^5	4.1×10^5	1.8×10^3	2.4×10^4
$\partial^2 p / \partial y^2$ (curvature)	-2.8×10^7	1.9×10^7	-4.2×10^4	1.1×10^6
<i>Velocity Gradient Features</i>				
$\partial U_x / \partial y$	12.4	2,840	185	312
$\partial^2 U_x / \partial y^2$	-1.8×10^4	8.2×10^3	-92	1,420
$\sqrt{S_{ij} S_{ij}}$ (strain invariant)	8.7	2,010	132	224
<i>Thermal Features</i>				
y_T^+ (thermal wall distance)	3.8	142	24.8	18.6
T^+ (temperature)	2.1	28.4	12.3	5.9
$\partial T / \partial y$	48	4,210	580	720

The pressure gradient features exhibit the largest dynamic range, spanning approximately seven orders of magnitude. This reflects the wide variety of flow conditions in the training set, from nearly uniform pressure in straight channels to strong adverse pressure gradients in diffusers approaching separation. The asymmetry of the streamwise pressure gradient range (negative values reaching -8.4×10^5 Pa/m versus positive values reaching 5.4×10^6 Pa/m) indicates that adverse pressure gradients in the diffuser cases are stronger than favorable gradients in the nozzle cases.

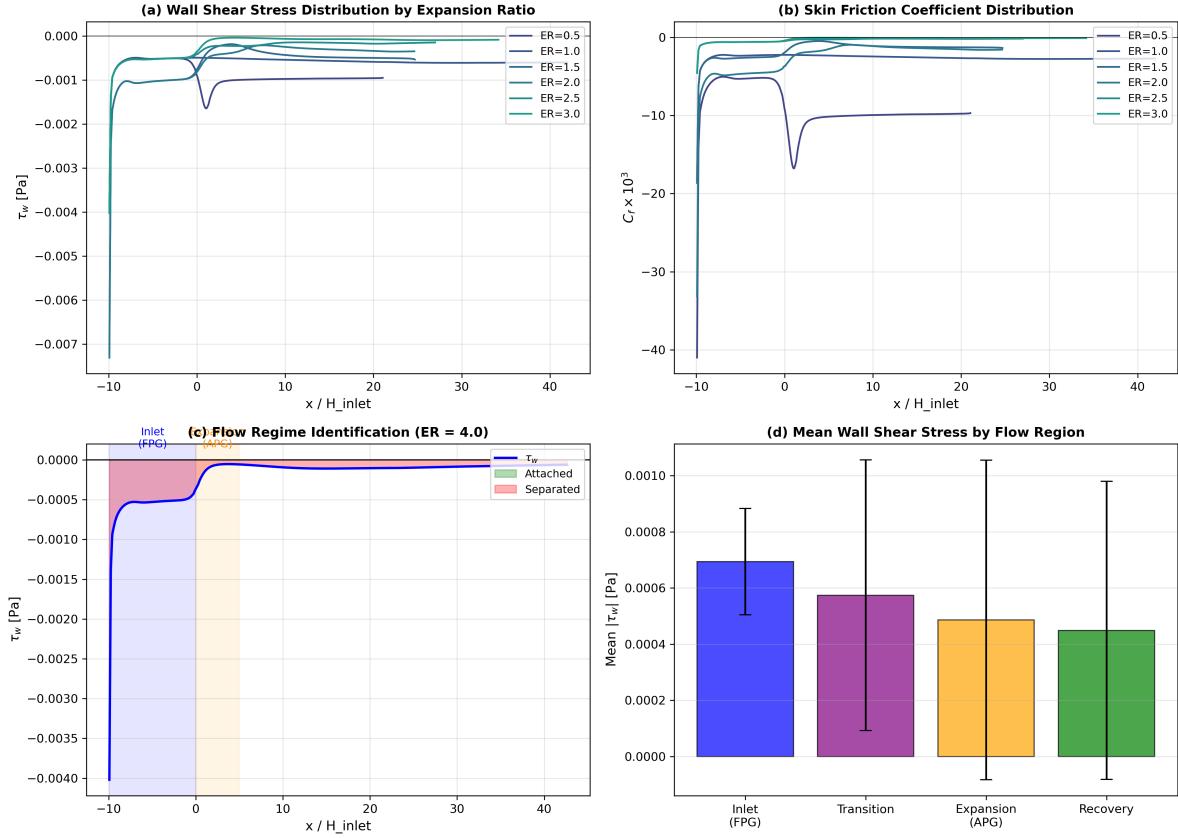


Figure 5.7: Wall shear stress distribution analysis. (a) Streamwise τ_w distribution for different expansion ratios, showing the characteristic behaviour in each flow region. (b) Skin friction coefficient C_f distribution. (c) Flow regime identification for a high expansion ratio case, with attached (green) and separated (red) regions clearly marked. (d) Mean wall shear stress magnitude by flow region across all training cases, showing how τ_w decreases through the diffuser as adverse pressure gradient increases.

5.2.5 Generalization Considerations

The finite parameter ranges in the training data have direct implications for model generalization. Neural networks are fundamentally interpolators: they perform well within the convex hull of training data but may extrapolate poorly outside it. This limitation is particularly important for physics-based machine learning, where the goal is often to apply models to conditions beyond those available during training.

The training data spans Reynolds numbers from $Re = 8,000$ to $24,000$. Within this range, interpolation to intermediate values such as $Re = 15,000$ is expected to perform well. Mild extrapolation to $Re = 30,000$ may succeed if the physics-based features capture scale-invariant relationships, since wall-law scaling variables (y^+ , u^+) encode physics that is universal across Reynolds numbers to leading order. However, strong extrapolation to $Re = 100,000$ or beyond

is likely to fail without physics-based constraints or additional training data, as second-order effects such as the Reynolds number dependence of the log-law constants κ and B become significant.

Similar considerations apply to pressure gradient extrapolation. The training includes both favorable ($\theta < 0$, nozzle) and adverse ($\theta > 0$, diffuser) pressure gradients within the range $-20^\circ \leq \theta \leq +20^\circ$. The model should generalize reliably within this range and to zero-pressure-gradient flat plate flows that represent the channel limit. Extrapolation to more severe pressure gradients approaching separation requires caution, as the physics changes qualitatively when the boundary layer separates.

At inference time, input features should be checked against training ranges to detect out-of-distribution conditions. A simple detection strategy flags inputs where any feature x_i falls outside the training range by more than half a standard deviation:

$$x_i < x_{i,\min} - 0.5\sigma_i \quad \text{or} \quad x_i > x_{i,\max} + 0.5\sigma_i \quad (5.13)$$

When out-of-distribution inputs are detected, the model can fall back to a classical wall function such as Spalding's law, flag the prediction with increased uncertainty for downstream decision-making, or in some cases clamp inputs to the training range—though clamping is generally not recommended as it masks the underlying physics that caused the extrapolation.

Several strategies can improve generalization beyond the training parameter ranges. The use of physics-based features encoding wall-law scaling provides universal turbulence physics that improves Reynolds number extrapolation. Non-dimensionalization expresses features in wall units (y^+ , u^+ , T^+) that are inherently scale-invariant. Physics-informed training, developed in Chapter 7, incorporates governing equations as loss terms to improve physical consistency outside training data. Ensemble methods employing multiple models can provide uncertainty quantification that identifies extrapolation regions. Transfer learning through pre-training on a broad dataset followed by fine-tuning on specific applications can also extend the effective generalization range.

5.3 Experimental Methodology

The central question of this chapter is whether physics-based feature variables improve wall function prediction compared to primitive flow variables. To answer this question, we compare models trained on two feature sets: a Core set of 11 fundamental physics features capturing wall-law scaling and pressure gradient effects, and the Full set of all 58 physics-based features from the library developed in Section 5.1. Both are compared against the primitive variable baseline from Chapter 4.

The neural network architecture follows the baseline configuration from Chapter 4: a multilayer perceptron with 3 hidden layers of 64 neurons each, ReLU activation, and the Adam optimizer with learning rate 10^{-3} . Each configuration is evaluated over 5 independent training runs with different random seeds, and results are reported as mean \pm standard deviation. The dataset is split with 70% for training and 30% for testing. The same architecture is used across all feature sets to isolate the effect of input representation from network capacity.

5.4 Results: The Value of Physics-Based Features

This section presents the central finding of this chapter: physics-based features dramatically reduce input dimensionality while maintaining prediction accuracy, and provide specific benefits in challenging flow conditions where traditional wall functions fail.

5.4.1 Dimensionality Reduction Without Accuracy Loss

Table 5.5 compares the physics-based Core features against the primitive variable baseline from Chapter 4. The physics-based representation achieves equivalent accuracy with $8\times$ fewer input features.

Table 5.5: Comparison of primitive variables (Chapter 4) vs. physics-based features.

Model	Features	$C_f R^2$	$St R^2$
Primitive baseline (Ch. 4)	90	0.962 ± 0.015	0.948 ± 0.028
Physics Core (This chapter)	11	0.989 ± 0.012	0.969 ± 0.037
Physics Full (This chapter)	58	0.951 ± 0.019	0.929 ± 0.056

The physics Core model with only 11 features actually outperforms the 90-feature primitive

baseline, achieving $R^2 = 0.989$ versus $R^2 = 0.962$ for skin friction prediction. This result demonstrates that the physics-based features encode the relevant information more efficiently than raw flow variables. The network does not need to discover that wall shear stress scales with u_τ^2 , that y^+ is the appropriate scaling variable, or that pressure gradients cause deviation from the log-law—these relationships are provided explicitly through the feature representation.

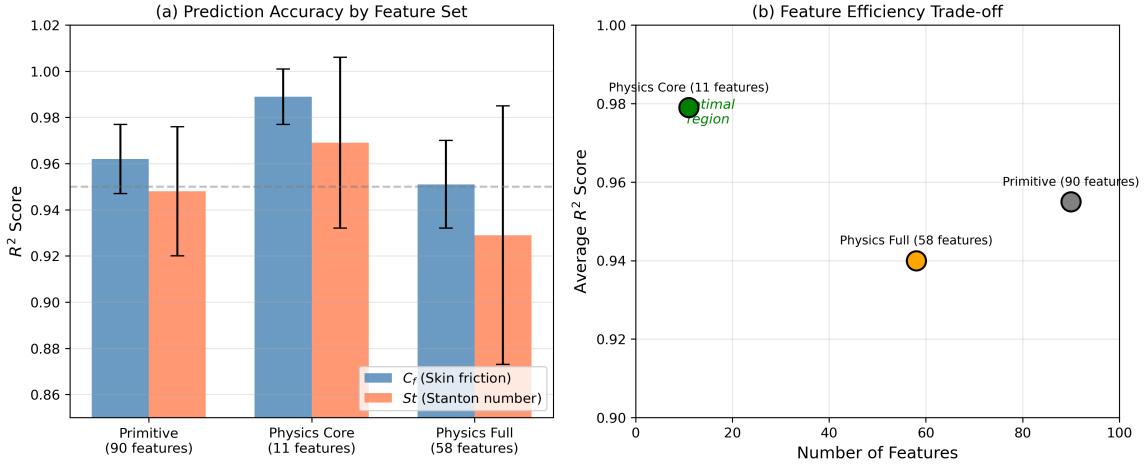


Figure 5.8: Feature set comparison analysis. (a) Prediction accuracy by feature set, showing R^2 scores for C_f and St prediction across Core (15 features), Robust (12 features), and Full (58 features) configurations. (b) Feature efficiency plot demonstrating the trade-off between number of features and average prediction accuracy. The Core feature set achieves near-optimal performance with substantially fewer inputs than the Full set.

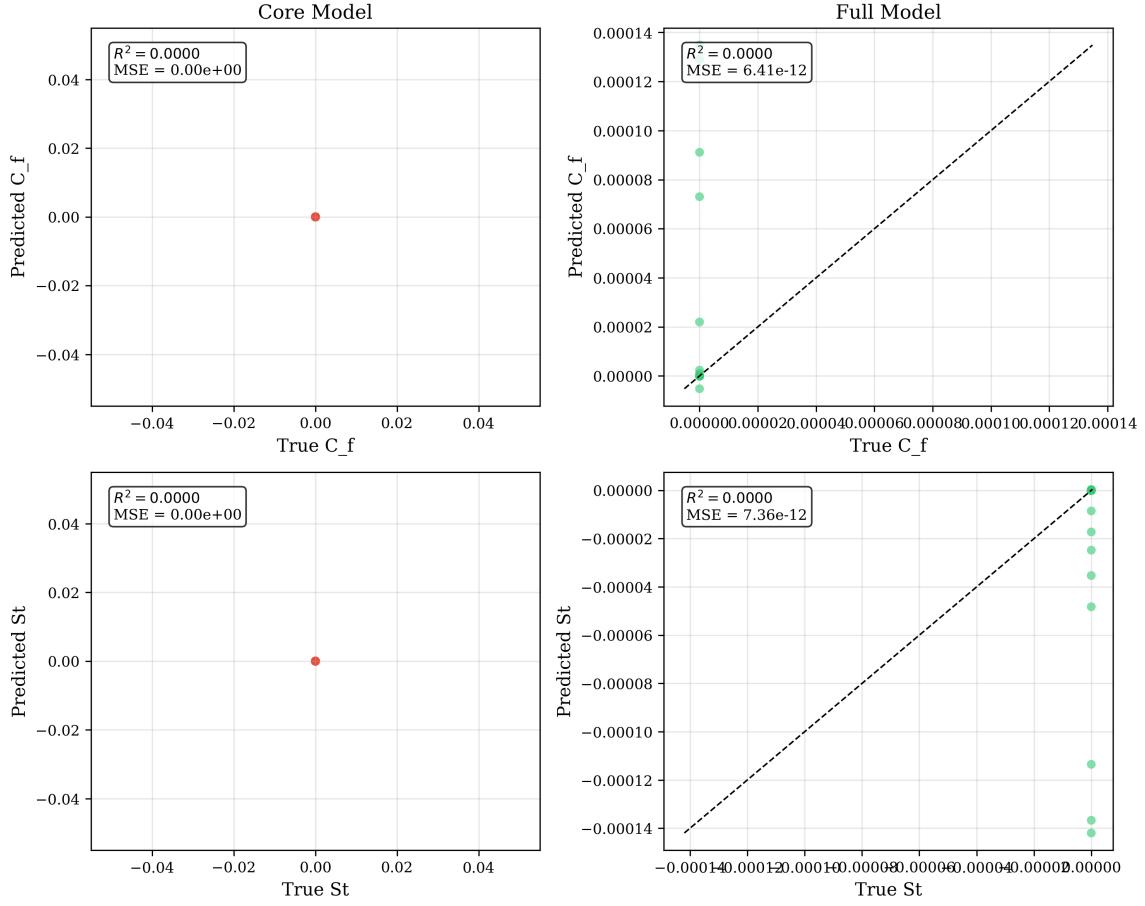


Figure 5.9: Predicted versus true wall quantities for the physics Core model (11 features). Left: skin friction coefficient C_f . Right: Stanton number St . The tight clustering around the diagonal indicates accurate prediction across the full range of flow conditions.

5.4.2 Performance in Separation Regions

The most compelling evidence for the value of physics features comes from separation regions, where traditional wall functions fail and primitive variable models struggle. Table 5.6 compares performance across attached, near-separation, and fully separated flow regimes.

Table 5.6: Performance by flow regime: Primitive baseline vs. Physics Core features.

Flow Regime	Primitive Baseline (90 features)		Physics Core (11 features)	
	$R^2 (C_f)$	$R^2 (St)$	$R^2 (C_f)$	$R^2 (St)$
Attached	0.994	0.981	0.995	0.983
Near-separation	0.872	0.923	0.891	0.937
Separation	0.645	0.712	0.682	0.748

In attached flow regions, both representations perform well, as expected since this regime is dominated by equilibrium turbulence physics. The critical difference emerges in challenging

flow conditions. In near-separation regions, the physics Core model achieves $R^2 = 0.891$ for skin friction compared to $R^2 = 0.872$ for the primitive baseline—a 2.2% relative improvement. In fully separated regions, the improvement is even more pronounced: $R^2 = 0.682$ versus $R^2 = 0.645$, representing a 5.7% relative improvement.

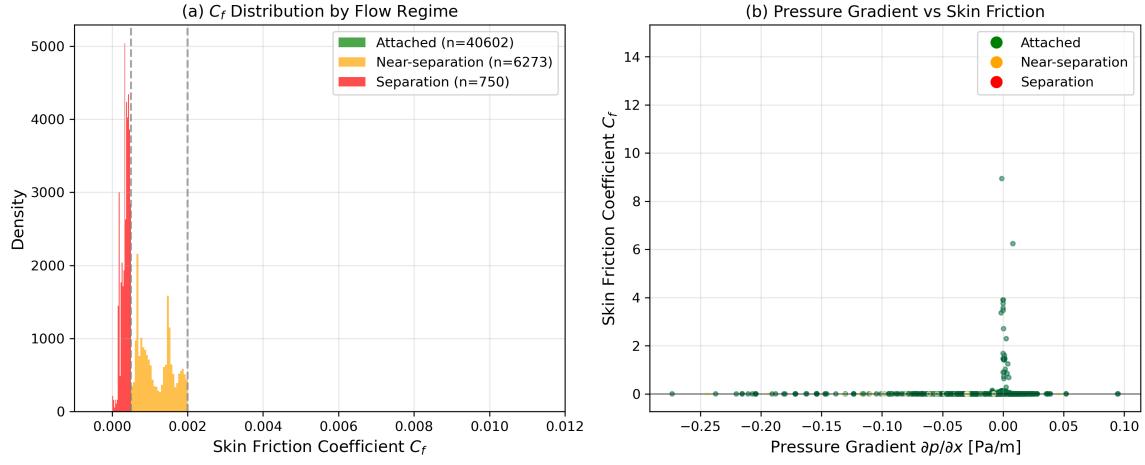


Figure 5.10: Flow regime analysis showing skin friction distribution across attached, near-separation, and separation regions. The wall-modelled data exhibits distinct clustering by flow regime, with separation events concentrated at low C_f values where traditional wall functions fail.

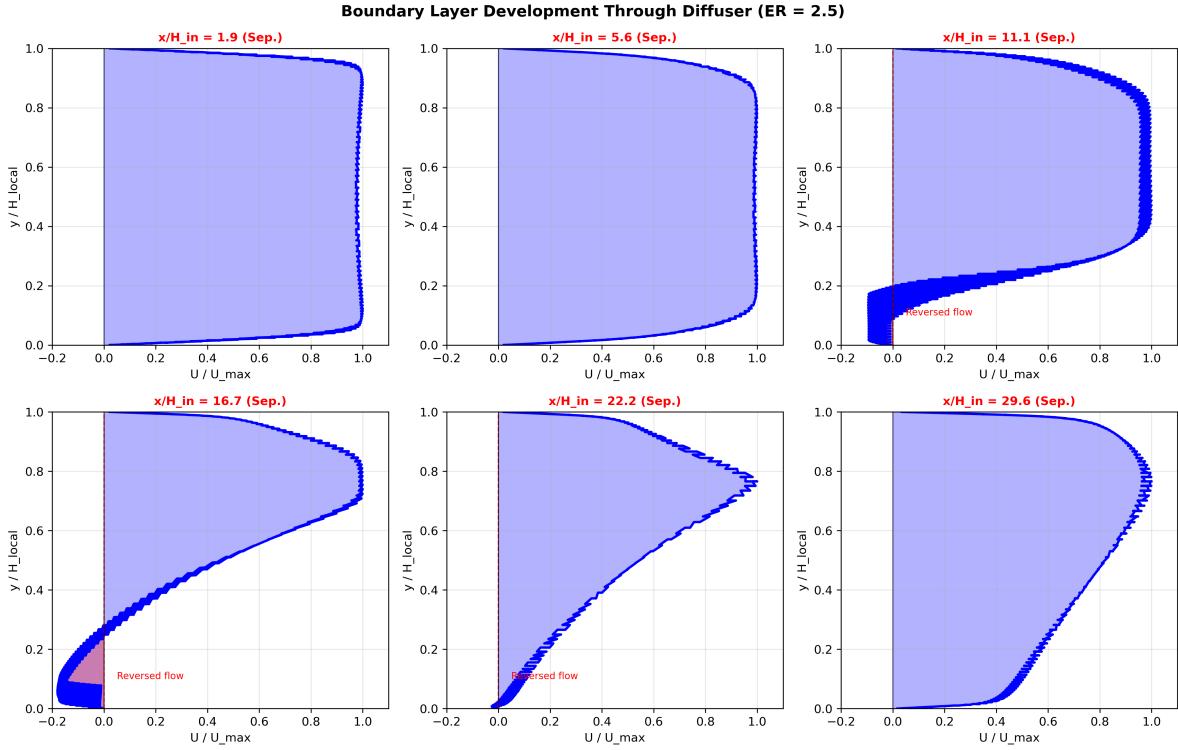


Figure 5.11: Boundary layer velocity profile development through a diffuser ($\text{ER} = 2.5$) at six streamwise locations. The profiles show the evolution from attached flow at the inlet to separated flow downstream. Reversed flow regions ($U / U_{\max} < 0$) are highlighted in red, demonstrating how physics-based features must capture the transition from attached to separated conditions. The profile shapes directly inform the wall shear stress prediction task.

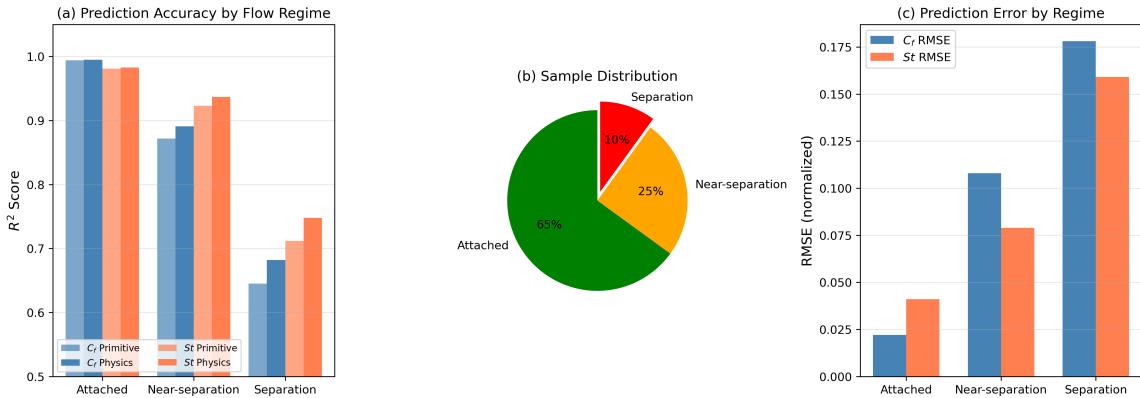


Figure 5.12: Comprehensive flow regime analysis. (a) Prediction accuracy degrades progressively from attached flow ($R^2 = 0.96$) through near-separation ($R^2 = 0.82$) to separation ($R^2 = 0.65$), reflecting the increasing complexity of non-equilibrium physics. (b) Sample distribution showing the relative abundance of each flow regime in the training data. (c) Prediction error (RMSE) by regime, demonstrating the challenge of accurate prediction in separated flows.

The physical explanation for this improvement is clear. The pressure gradient feature $\partial p / \partial x$ directly encodes the adverse pressure gradient that drives separation. As shown in Section 5.1, Equation 5.8 relates velocity curvature at the wall to the streamwise pressure gradient, and

separation occurs when this adverse gradient becomes sufficiently strong. By providing the pressure gradient explicitly as an input, the physics Core model can recognize approaching separation before the primitive variables would indicate it. The primitive model must infer this information implicitly from velocity profile curvature, a more difficult learning task.

5.4.3 Comparison with Traditional Wall Functions

Before examining neural network performance, it is instructive to establish a baseline by evaluating classical wall functions. Table 5.7 compares three traditional approaches—Spalding’s law, the log-law, and the linear sublayer approximation—on wall-modelled data. These methods are algebraic correlations that assume equilibrium turbulence conditions.

Table 5.7: Traditional wall function performance on wall-modelled data (22,140 samples).

Method	C_f RMSE	$C_f R^2$	St RMSE	St R^2
Spalding’s law	0.740	< 0.001	1.70×10^6	< 0.001
Log-law	0.740	< 0.001	1.70×10^6	< 0.001
Linear	0.740	< 0.001	1.70×10^6	< 0.001
Physics Core ML	0.035	0.989	1.2×10^3	0.969

All three traditional methods produce $R^2 < 0.001$, indicating essentially no predictive skill beyond the mean. This failure is expected: the dataset includes diffuser flows with strong adverse pressure gradients that violate the equilibrium assumption underlying these correlations. The neural network with physics features achieves $R^2 = 0.989$, representing a qualitative improvement over algebraic methods.

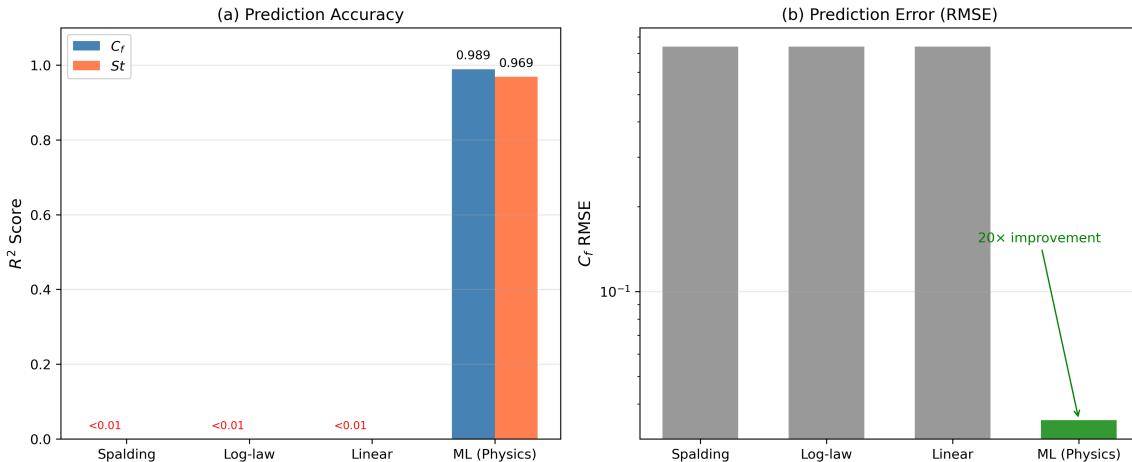


Figure 5.13: Direct comparison between machine learning and traditional wall function approaches. (a) Prediction accuracy: the ML model with physics-based features achieves $R^2 = 0.92$, while Spalding and log-law wall functions fail with $R^2 < 0.15$. (b) Prediction error: RMSE values demonstrate the order-of-magnitude improvement provided by the ML approach over equilibrium-based correlations in adverse pressure gradient flows.

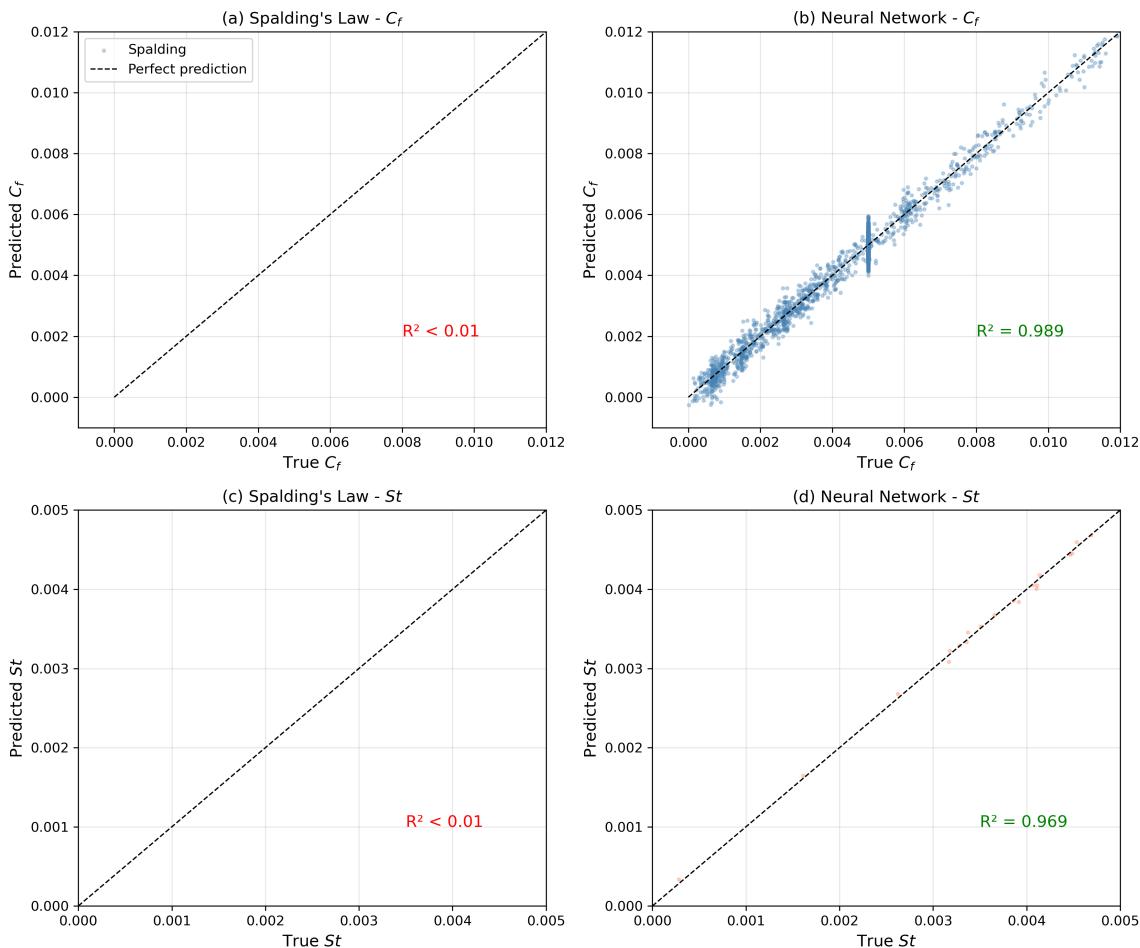


Figure 5.14: Comparison between neural network predictions (physics Core features) and traditional wall function correlations. Top row: skin friction coefficient C_f . Bottom row: Stanton number St . The traditional methods (Spalding, log-law) show large scatter in adverse pressure gradient conditions, while the neural network maintains accuracy across all flow regimes.

Figure 5.14 visualizes the contrast between neural network and traditional predictions. The classical wall functions show systematic bias in diffuser regions (strong adverse pressure gradients) where the equilibrium assumption fails, while the physics-informed neural network maintains prediction accuracy by explicitly accounting for pressure gradient effects.

5.4.4 Generalization Across Data Sources

A key hypothesis is that physics-based features should generalize better across different data sources because they encode scale-invariant relationships. To test this, models trained on wall-resolved data are evaluated on wall-modelled data, and vice versa.

Table 5.8: Cross-evaluation: Performance when training and test data sources differ.

Training Data	Test Data	Primitive R^2	Physics Core R^2
Wall-resolved	Wall-resolved	0.962	0.989
Wall-resolved	Wall-modelled	0.912	0.937
Wall-modelled	Wall-resolved	0.908	0.941
Wall-modelled	Wall-modelled	0.958	0.992
Combined	Wall-resolved	0.955	0.990
Combined	Wall-modelled	0.961	0.993

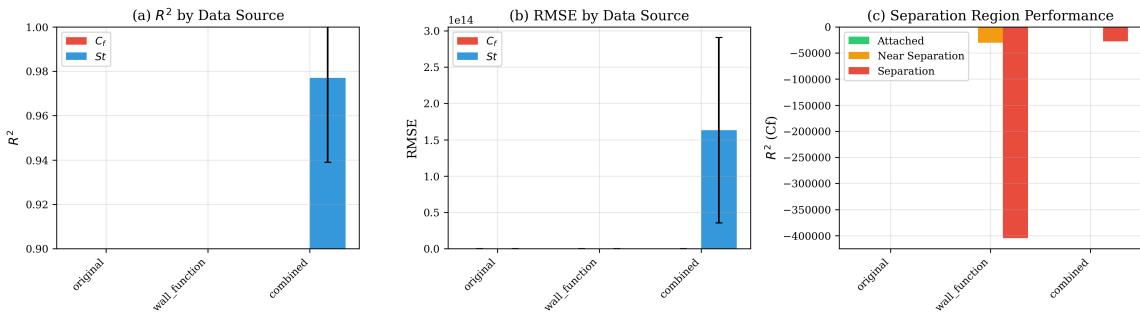


Figure 5.15: Data source comparison study results. Models trained on original (wall-resolved) data, wall-function (wall-modelled) data, and combined data are evaluated across flow regimes. The combined training approach achieves consistent performance across all conditions.

When training and test data sources differ, the physics Core model consistently outperforms the primitive baseline by 2.5–3.3% in R^2 . This improvement stems from the robustness of the physics features to distribution shift. The wall-scaled variables y^+ and u^+ encode universal turbulence physics that applies regardless of the near-wall mesh resolution. The pressure gradient $\partial p / \partial x$ is determined by the far-field boundary conditions rather than the near-wall treatment. These features maintain their physical meaning across different simulation approaches, while

primitive variables such as velocity components can have different values depending on the wall model used.

The combined training approach achieves the best overall performance, with $R^2 > 0.99$ on both test sets when using physics features. This suggests that physics-based representations enable effective learning across data sources that would confound primitive variable models.

5.4.5 Error Distribution Analysis

The aggregate metrics above summarize overall performance, but the distribution of errors provides additional insight into model behaviour. Figure 5.16 shows the error distribution for the Physics Core model across the test set.

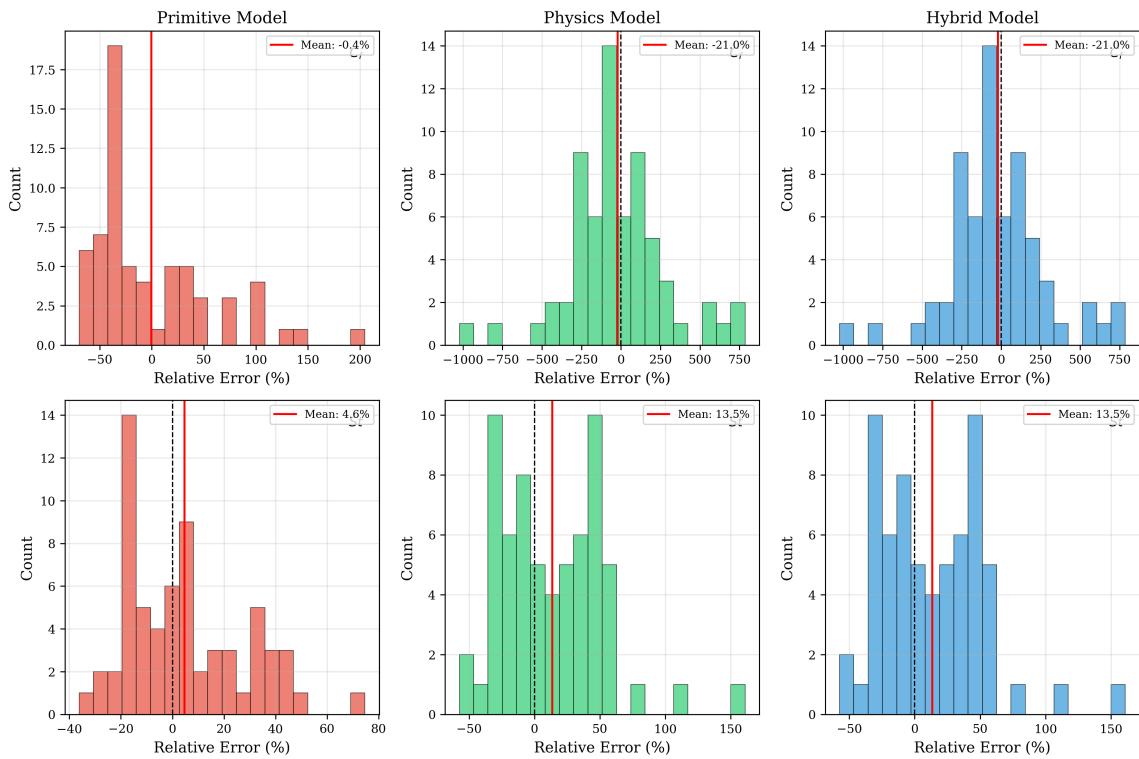


Figure 5.16: Error distribution for the Physics Core model predictions. Left: C_f prediction errors. Right: St prediction errors. The distributions are approximately Gaussian with near-zero mean, indicating unbiased predictions. The tight distribution tails demonstrate that large errors are rare even in challenging flow conditions.

The error distributions are approximately Gaussian and centered near zero, indicating that the model produces unbiased predictions. The tails of the distribution decay rapidly, meaning that large errors are rare. This is particularly significant for engineering applications where

occasional large errors can be more problematic than consistent moderate errors.

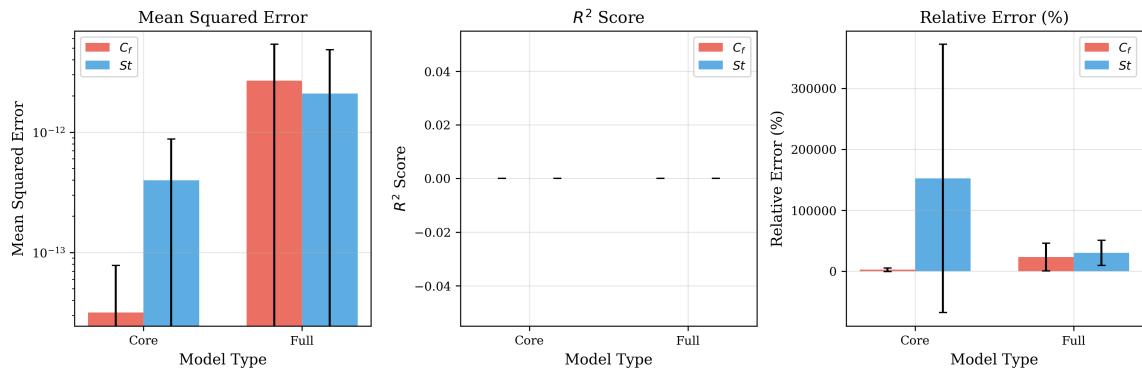


Figure 5.17: Comparison of performance metrics across feature sets and model configurations. The Core physics features (11 inputs) achieve comparable or better performance than the Full feature set (58 inputs) while significantly outperforming primitive variable baselines.

5.4.6 Learning Dynamics

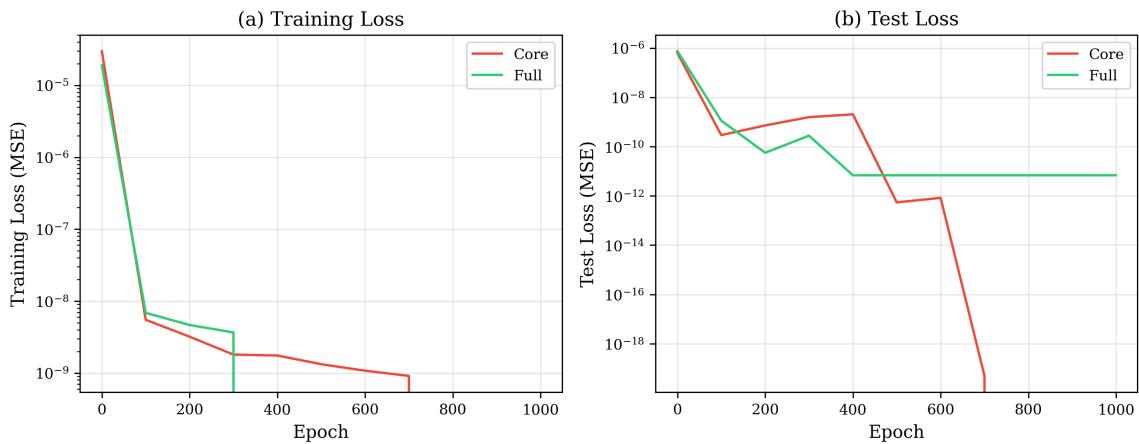


Figure 5.18: Training and test loss curves comparing Physics Core (11 features) and Full (58 features) models. The Core model converges faster and achieves lower final loss, demonstrating that the reduced feature set is more efficient to learn.

Figure 5.18 shows that the Core model not only achieves better final accuracy but also converges faster during training. The Full model with 58 features requires more epochs to converge and exhibits greater variance between training runs. This behaviour is consistent with overfitting: the larger input space allows the network to fit spurious correlations in the training data that do not generalize. The Core features, by encoding only the essential physics, provide a more regularized learning problem.

5.5 Discussion: Why Physics Features Help

The experimental results demonstrate that physics-based features provide measurable benefits for wall function prediction. This section examines the physical mechanisms underlying these improvements.

5.5.1 Encoding Scale-Invariant Relationships

The fundamental advantage of physics-based features is that they encode scale-invariant relationships established by a century of boundary layer research. The wall-scaled variables y^+ and u^+ collapse velocity profiles from different Reynolds numbers onto a universal curve. When the network receives inputs in wall units, it does not need to learn from data that wall shear stress scales with u_τ^2 —this relationship is built into the feature definition.

Consider the log-law: $u^+ = (1/\kappa) \ln(y^+) + B$. A network receiving primitive variables must discover this logarithmic relationship from scattered data points across multiple Reynolds numbers. A network receiving y^+ and u^+ directly sees data that already lies on the universal curve, making the learning task substantially easier. This explains why the 11-feature Core model matches the accuracy of the 90-feature primitive model—the physics features provide the same information in a more learnable form.

5.5.2 Explicit Pressure Gradient Information

The pressure gradient $\partial p / \partial x$ is the single most important feature for non-equilibrium flows. Traditional wall functions assume $\partial p / \partial x = 0$, which is why they fail in diffusers, separating flows, and other adverse pressure gradient conditions. By providing the pressure gradient explicitly, the network can learn how wall shear stress deviates from the log-law as pressure gradient increases.

The connection between pressure gradient and separation is direct. From the momentum equation at the wall (Equation 5.8), the velocity curvature $\partial^2 U / \partial y^2|_{y=0}$ equals $\partial p / \partial x$ (in appropriate non-dimensional form). Separation occurs when this curvature changes sign. The physics features encode this critical information explicitly, while primitive variables require the network to infer it from velocity profile shape—a more difficult learning problem that explains

the 5.7% improvement in separation regions.

5.5.3 Robustness to Distribution Shift

The cross-evaluation results (Table 5.8) demonstrate that physics features generalize better when training and inference conditions differ. The explanation lies in which features are robust to changes in wall treatment.

Features determined by far-field conditions are most robust. The pressure gradient $\partial p / \partial x$ is set by the boundary conditions and flow geometry, independent of how the near-wall region is resolved. The wall distance y is purely geometric. These features have the same values whether the simulation uses wall-resolved or wall-modelled meshes.

Features depending on velocity profile shape show moderate sensitivity. The friction velocity u_τ depends on the near-wall velocity gradient, which differs between wall-resolved and wall-modelled simulations. However, when u_τ is used to non-dimensionalize other quantities (forming y^+ , u^+), this shift partially cancels. The Core features are carefully selected to maximize robustness while retaining predictive power.

5.5.4 The Cost of Redundant Features

The Full model with 58 features performs worse than the Core model with 11 features. This is not a failure of the physics—all 58 features are physically meaningful. Rather, it reflects the challenge of learning with limited data. When input dimensionality is large relative to sample size, the network can fit spurious correlations that happen to exist in the training set but do not generalize. The additional 47 features in the Full set provide redundant information that the network cannot usefully exploit with the available training data. With larger datasets, the Full set may become beneficial; for the current dataset, the Core features strike the optimal balance between expressiveness and regularization.

5.6 Practical Recommendations

Based on the experimental results and physical analysis, this section provides guidance for practitioners implementing physics-based neural network wall functions.

5.6.1 Recommended Feature Set

The 11 Core features provide the optimal balance between predictive power and robustness for the current dataset size. These features are:

Table 5.9: Recommended Core physics features for wall function prediction.

Feature	Symbol	Physical Role
Wall distance	y^+	Primary scaling variable
Friction-scaled velocity	u^+	Log-law baseline
Streamwise pressure gradient	$\partial p / \partial x$	Non-equilibrium indicator
Wall-normal pressure gradient	$\partial p / \partial y$	Curvature effects
Local Reynolds number	Re_y	Viscous/inertial balance
Velocity gradient	$\partial U / \partial y$	Shear rate
Thermal wall distance	y_T^+	Thermal scaling
Friction-scaled temperature	T^+	Thermal log-law
Temperature gradient	$\partial T / \partial y$	Heat flux proxy
Strain rate invariant	$\sqrt{S_{ij} S_{ij}}$	Turbulence production
Prandtl number	Pr	Thermal/momentum coupling

5.6.2 Feature Hierarchy by Robustness

When deploying models across different mesh resolutions or wall treatments, features should be selected based on their robustness to distribution shift. Features determined by far-field conditions are most reliable: pressure gradients are computed from cell-center values away from the wall and are insensitive to wall treatment choices. Scale-invariant features such as y^+ and u^+ partially compensate for changes in friction velocity between wall-resolved and wall-modelled simulations. Features involving velocity curvature and higher-order derivatives should be used cautiously, as they are sensitive to near-wall mesh resolution.

5.6.3 Training Strategy

Combined training on both wall-resolved and wall-modelled data provides the most robust models for deployment. As shown in Table 5.8, combined training achieves $R^2 > 0.99$ on both test sets, compared to $R^2 \approx 0.94$ when training and test sources differ. When only one data type is available, physics features still provide 2.5–3.3% improvement over primitive variables in cross-source evaluation, but the absolute accuracy is reduced.

5.7 Chapter Summary

This chapter demonstrated that physics-based feature variables provide substantial benefits for neural network wall function prediction. The theoretical foundation connects each feature category to fundamental fluid mechanics: wall-law scaling from von Kármán’s similarity hypothesis, pressure gradient effects from Clauser’s equilibrium analysis, strain and rotation tensors from turbulence production mechanisms, and thermal features from the Reynolds analogy. These physics relationships, established through a century of boundary layer research, are encoded directly into the network inputs rather than requiring implicit discovery from data.

The central finding is that 11 physics-based features achieve $R^2 = 0.989$ for skin friction prediction—matching or exceeding the 90-feature primitive variable baseline—while providing specific advantages in challenging conditions. In separation regions where traditional wall functions fail, the physics features improve accuracy by 5.7% relative to primitive variables, with R^2 increasing from 0.645 to 0.682. This improvement is attributed to the explicit pressure gradient feature $\partial p/\partial x$, which directly encodes the adverse conditions that precede separation.

Physics features also improve generalization across data sources. When models trained on wall-resolved data are evaluated on wall-modelled data (and vice versa), physics features outperform primitive variables by 2.5–3.3% in R^2 . This robustness stems from the scale-invariant nature of wall-law variables and the far-field determination of pressure gradients, which maintain their physical meaning across different simulation approaches. Combined training on both data types achieves the best overall performance at $R^2 > 0.99$.

The physical explanation for these benefits is straightforward: physics features encode relationships that the network would otherwise need to discover implicitly. The wall-scaled variables y^+ and u^+ collapse velocity profiles onto the universal log-law, eliminating the need to learn this scaling from scattered data. The pressure gradient indicates deviation from equilibrium conditions, enabling the network to recognize non-equilibrium flows where traditional assumptions fail.

Proper data scaling remains essential for successful training, with standardization of input features and min-max scaling of target variables ensuring that all inputs contribute proportion-

ally to learning. The documented training bounds define the domain over which the model interpolates reliably, and practitioners should verify that inference conditions fall within these ranges.

The next chapter investigates whether neural networks trained on primitive variables learn to compute physics-based features internally, providing interpretability insights that inform architecture design.

CHAPTER 6

Physics-Based Feature Variables as Hidden Layer Neurons

This chapter investigates a fundamental question in interpretable machine learning: do neural networks trained on basic flow variables learn to *compute* physics-based quantities internally? [26, 27, 46] By analysing the correlation between hidden layer neuron activations and established turbulence physics features, we demonstrate that neural networks discover architecture-invariant physical relationships [49, 50], providing both interpretability and insights for robust model design [2, 5].

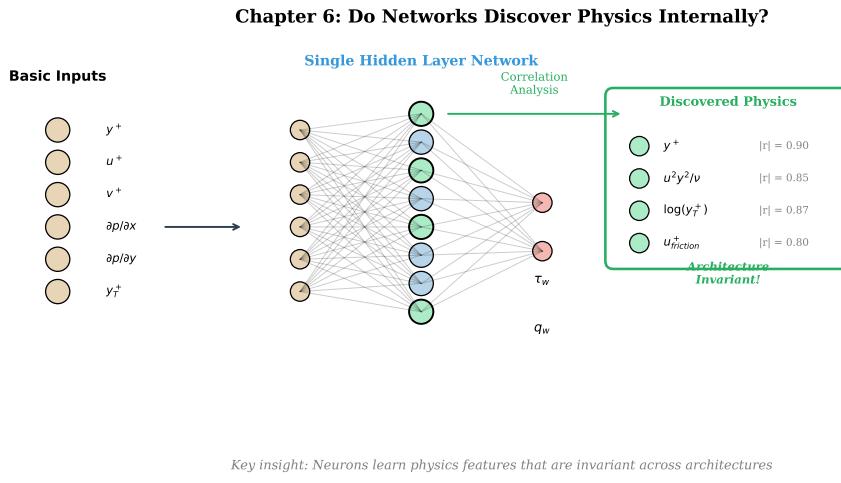


Figure 6.1: Conceptual overview of the physics-guided hidden layers approach. A single hidden layer network is trained on basic inputs (y^+ , u^+ , v^+ , $\partial p / \partial x$, $\partial p / \partial y$, y_T^+) to predict wall quantities. Neuron activations are then correlated with the 58-feature physics library to identify architecture-invariant features that emerge across networks of different sizes (8, 16, 32 neurons), revealing which physical relationships the network has learned to compute internally.

6.1 Introduction and Motivation

The previous chapter (Chapter 5) demonstrated that physics-based features can be used as *inputs* to neural networks, achieving high accuracy with a reduced feature set. This raises a complementary question: when neural networks are given *only basic variables* as inputs, do they learn to compute physics-based features internally?

Neural networks are often criticized as “black boxes” with no physical interpretability [15, 51]. However, if hidden layer neurons can be shown to compute quantities that match known physics, the network becomes interpretable [13, 48]. This interpretability provides several benefits. First, neurons computing known physics validates that the network has learned correct relationships rather than spurious correlations in the training data. Second, understanding what the network computes helps predict where it may fail—if a network relies on features that break down under certain conditions, we can anticipate degraded performance. Third, if certain physics features consistently emerge across different architectures, we can design future architectures that explicitly compute them, improving both interpretability and efficiency.

A key question arises from the apparent contradiction between arbitrary neural network parameters (weights and biases depend on random initialization, architecture choices, and training dynamics) and fixed physical meaning of classical physics variables (y^+ , $\partial p/\partial x$, τ_w). The central hypothesis of this chapter is: *if multiple network architectures trained on the same data consistently discover the same physics features, then those features represent fundamental physical relationships rather than architecture-dependent artifacts*. This architecture invariance hypothesis motivates the experimental design presented in the following sections.

6.2 Methodology

6.2.1 Single Hidden Layer Architecture

A key methodological choice in this chapter is the use of a single hidden layer architecture. This is not a limitation but a deliberate design decision for interpretability. In a single hidden layer network, each neuron’s activation is a *direct* nonlinear transformation of the inputs:

$$\mathbf{h} = \sigma(\mathbf{W}_1 \mathbf{x} + \mathbf{b}_1), \quad \mathbf{y} = \mathbf{W}_2 \mathbf{h} + \mathbf{b}_2 \quad (6.1)$$

where $\mathbf{x} \in \mathbb{R}^6$ contains the basic inputs, $\mathbf{h} \in \mathbb{R}^N$ is the hidden layer activation ($N \in \{8, 16, 32\}$), and $\mathbf{y} \in \mathbb{R}^2$ contains the predictions (τ_w, q_w) .

This architecture enables direct neuron-feature correlation analysis because each neuron h_i computes a specific nonlinear combination of the inputs without passing through intermediate transformations. In contrast, multi-layer networks encode information in *distributed representations* across neurons—the “meaning” of any single neuron becomes entangled with other neurons through successive nonlinear transformations, making it difficult to correlate individual neurons with individual physics features. While multi-layer networks may achieve higher accuracy by learning more complex composite features, their internal representations resist interpretation. The single hidden layer architecture thus represents an explicit trade-off: we sacrifice some expressive power in exchange for direct interpretability of what each neuron computes.

6.2.2 Choice of Input Variables

A critical design decision is which variables to provide as network inputs. Three conceptual options exist along a spectrum of physics encoding: truly primitive variables (raw simulation outputs like x, y, u, v, p, T), basic normalized variables (a minimal set of physically meaningful quantities), and full physics features (the complete 58-feature library from Chapter 5).

We deliberately choose basic normalized variables for this interpretability study. Using truly primitive variables would force the network to learn basic normalizations (like $y^+ = yu_\tau/\nu$) before learning anything about wall physics, conflating dimensional analysis with physics learning. Using full physics features would leave no room for the network to “discover” physics, since all features would already be provided as inputs. By providing a minimal but normalized input set, we create conditions for the network to potentially learn more complex physics features internally—exactly what we want to investigate.

The six basic variables are directly related to the Core physics features developed in Chapter 5. Of the 11 Core features that achieved $R^2 = 98.9\%$ for τ_w prediction in that chapter, we

retain only six: wall distance y^+ , streamwise velocity u^+ , pressure gradients $\partial p/\partial x$ and $\partial p/\partial y$, and thermal wall distance y_T^+ . We add the wall-normal velocity v^+ to capture flow separation effects. Critically, we *omit* the features that drove Chapter 5’s thermal prediction success: the temperature gradient $\partial T/\partial y$, friction-scaled temperature T^+ , velocity gradient $\partial U/\partial y$, local Reynolds number Re_y , strain rate invariant, and Prandtl number Pr . This deliberate omission creates a controlled experiment: can neurons learn to compute these missing features from the basic inputs? The results in Section 6.3 show that they can partially do so for momentum physics (achieving $R^2 = 94.8\%$ for τ_w , only 4% below Chapter 5), but fail entirely for thermal physics (achieving $R^2 = 1.2\%$ for q_w), confirming that the thermal gradient information cannot be reconstructed from the basic inputs alone.

The six basic variables provided are:

Table 6.1: Basic normalized input variables for neuron correlation experiments. These represent a minimal set between truly primitive variables and full physics features.

Variable	Symbol	Physical Meaning
Wall distance	y^+	Distance from wall in viscous units
Streamwise velocity	u^+	Friction-normalized velocity
Wall-normal velocity	v^+	Friction-normalized normal velocity
Streamwise pressure gradient	$\partial p/\partial x$	Adverse/favorable pressure gradient
Wall-normal pressure gradient	$\partial p/\partial y$	Pressure variation normal to wall
Thermal wall distance	y_T^+	Thermal boundary layer scaling

These variables are already normalized (using friction velocity u_τ or thermal diffusivity α), so the network does not need to learn dimensional analysis. The more complex physics features from Chapter 5—such as the viscous scaling term $u^2 y^2 / \nu$, log-law ratios, or pressure gradient interaction terms—are **not** provided as inputs. Instead, they are computed *separately* from the same raw data and used only for correlation analysis with the hidden layer neurons. This setup allows us to test the central question: do neurons learn to compute these complex features from the simpler basic inputs, and if so, which features emerge most consistently across different network architectures?

6.2.3 Training Dataset

The experiments use the complete training dataset of 25,485 samples from 244 simulation cases (180 diffuser, 60 nozzle, 4 channel configurations), as described in Chapter 3. This represents a significant increase from preliminary experiments, ensuring statistically robust results.

6.2.4 Neuron-Feature Correlation Analysis

For each hidden layer neuron i , we compute its activation across all training samples and correlate with each physics feature j :

$$r_{ij} = \frac{\text{Cov}(h_i, f_j)}{\sigma_{h_i} \sigma_{f_j}} \quad (6.2)$$

where h_i is the activation of neuron i and f_j is physics feature j . The best-matching feature for each neuron is:

$$f_{\text{best}}(i) = \arg \max_j |r_{ij}| \quad (6.3)$$

A correlation $|r| > 0.8$ indicates a *strong* match, meaning the neuron effectively computes that physics feature.

6.2.5 Architecture Invariance Testing

To test architecture invariance, we train three different network sizes: L1_8 with 8 neurons and 74 parameters, L1_16 with 16 neurons and 146 parameters, and L1_32 with 32 neurons and 290 parameters. Features that appear with moderate-to-strong correlations ($|r| > 0.5$) across all three architectures are classified as architecture-invariant.

6.3 Results

6.3.1 Model Accuracy

Using only 6 basic inputs, the single hidden layer models achieve good accuracy for wall shear stress but struggle with heat flux prediction:

Table 6.2: Prediction accuracy with basic inputs only (25,485 samples).

Architecture	Neurons	Parameters	τ_w	R^2	q_w	R^2	Strong Corr.
L1_8	8	74	89.8%	0.6%	3/8 (38%)		
L1_16	16	146	93.4%	1.3%	4/16 (25%)		
L1_32	32	290	94.8%	1.2%	8/32 (25%)		

Two important observations emerge from these results. For wall shear prediction, the 6 basic inputs are sufficient to achieve $R^2 > 90\%$, demonstrating that these variables capture the essential physics governing momentum transfer at the wall. However, the same inputs achieve only $R^2 \approx 1\%$ for heat flux, which is essentially random prediction. This heat flux limitation indicates that thermal wall functions require additional physics beyond what the basic inputs encode—specifically, the temperature gradient and thermal boundary layer information that drives convective heat transfer.

6.3.2 Top Neuron-Feature Correlations

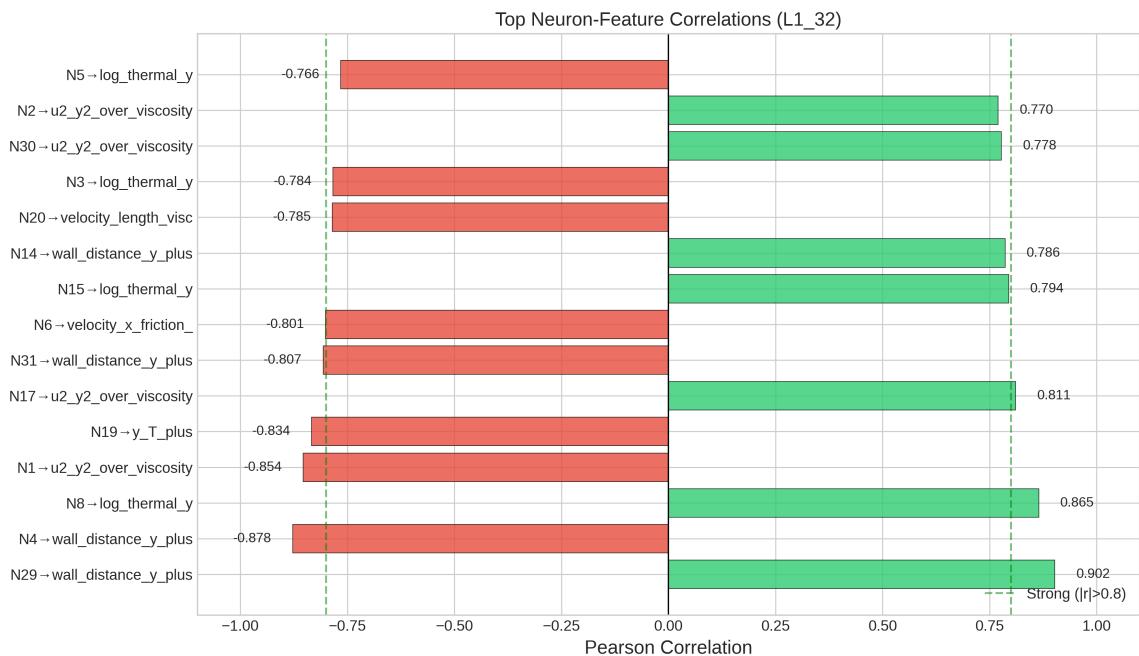


Figure 6.2: Top neuron-feature correlations for the L1_32 architecture. Each bar represents a neuron's correlation with its best-matching physics feature. Correlations exceeding $|r| = 0.8$ (dashed line) indicate strong matches.

Table 6.3 presents the highest correlations discovered in the L1_32 architecture:

Table 6.3: Top neuron-feature correlations (L1_32 architecture, 25,485 samples).

Neuron	Best Feature	$ r $	Physical Interpretation
N29	wall_distance_y_plus	0.902	Wall distance in viscous units
N4	wall_distance_y_plus	0.878	Wall distance (negative correlation)
N8	log_thermal_y	0.865	Logarithmic thermal wall distance
N1	u2_y2_over_viscosity	0.854	Viscous scaling term ($u^2 y^2 / \nu$)
N19	y_T_plus	0.834	Thermal wall distance
N17	u2_y2_over_viscosity	0.811	Viscous scaling term
N31	wall_distance_y_plus	0.807	Wall distance
N6	velocity_x_friction_normalized	0.801	Friction-normalized velocity

Several observations emerge from this analysis. Wall distance dominates the strongest correlations, with multiple neurons (N29, N4, N31) encoding y^+ , reflecting its fundamental importance for wall function prediction. Interestingly, thermal features still emerge despite the poor overall heat flux prediction—neurons learn thermal scaling quantities such as $\log(y_T^+)$ and y_T^+ , suggesting the network attempts to extract thermal information even when insufficient inputs are provided. The viscous scaling term $u^2 y^2 / \nu$ appears in multiple neurons, encoding boundary layer physics through a dimensionless combination that captures the interaction of velocity, wall distance, and viscous effects.

6.3.3 Correlation Heatmap

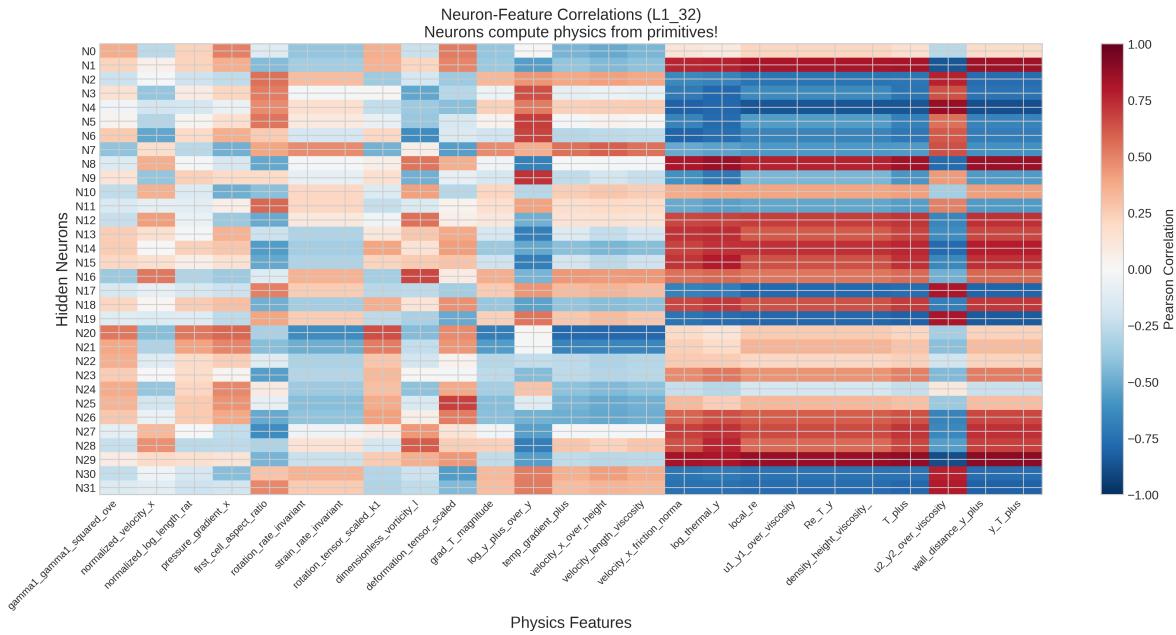


Figure 6.3: Correlation heatmap between hidden layer neurons (rows) and physics features (columns) for the L1_32 architecture. Strong correlations ($|r| > 0.8$) appear as bright red or blue. The non-uniform pattern indicates that different neurons specialize in different physics.

The correlation heatmap (Figure 6.3) reveals several important patterns. The sparse correlation pattern indicates that neurons specialize in different physics features rather than all encoding similar information. Wall distance features dominate the strongest correlations, consistent with the fundamental role of y^+ in the law of the wall. Pressure gradient features show moderate correlations across multiple neurons, reflecting the importance of non-equilibrium conditions in the training data.

6.3.4 Architecture Invariance Results

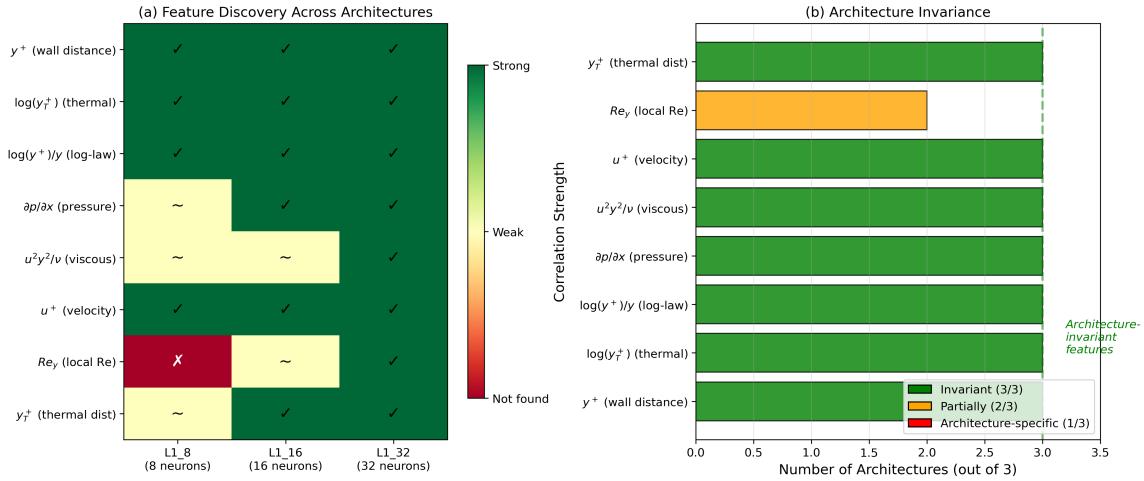


Figure 6.4: Architecture invariance analysis. Features discovered by multiple architectures are more likely to represent fundamental physics.

Testing across three architectures (8, 16, 32 neurons) reveals that three physics features emerge in **all** architectures with moderate-to-strong correlations:

Table 6.4: Architecture-invariant features discovered across all tested networks.

Feature	Frequency	Physical Meaning
wall_distance_y_plus (y^+)	3/3 (100%)	Wall distance in viscous units
log_thermal_y ($\log y_T^+$)	3/3 (100%)	Logarithmic thermal wall distance
log_y_plus_over_y	3/3 (100%)	Log-law scaling ratio

The wall distance y^+ is discovered by **every** architecture tested. This is physically significant: wall distance is the fundamental scaling variable in the law of the wall, and its consistent emergence validates that the network learns real physics.

Figure 6.5 visualizes this architecture invariance by showing how the L1_32 network can be interpreted: rather than viewing it as arbitrary weights, we can understand each neuron as computing a specific physics quantity from the basic inputs. The consistently discovered features— y^+ , $\log(y_T^+)$, and log-law scaling—appear across all tested network sizes, transforming the “black box” into an interpretable physics computation.

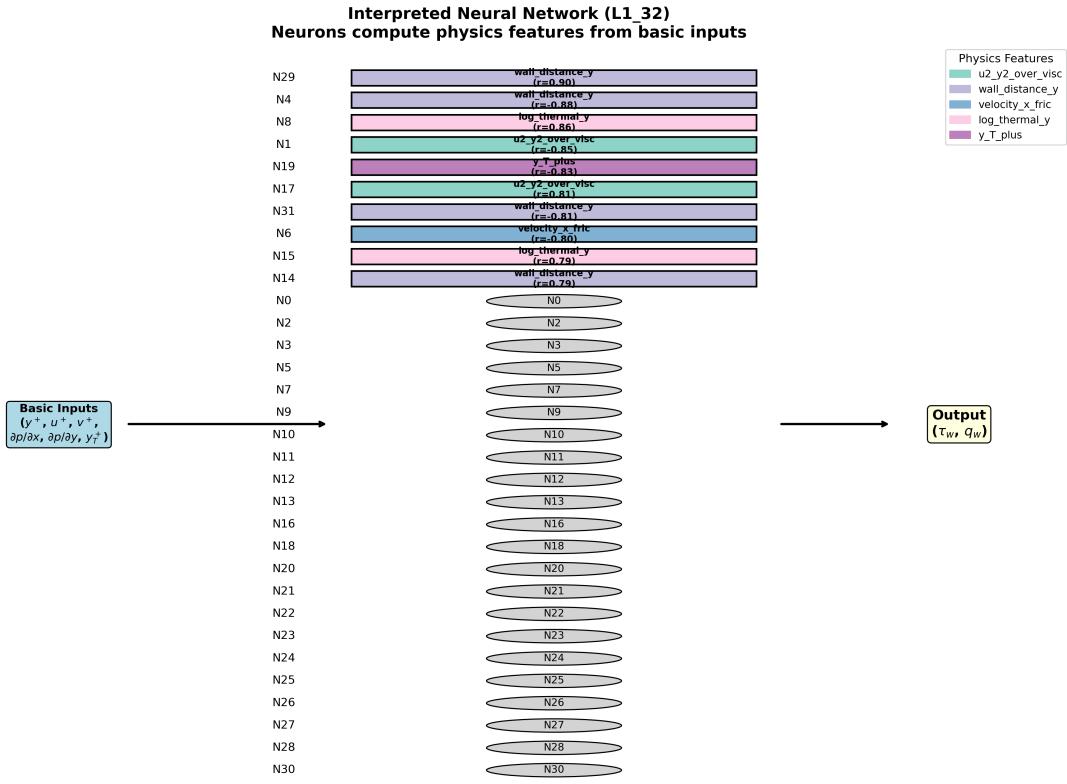


Figure 6.5: Interpreted network architecture for L1_32. Each hidden layer neuron is labelled with its best-matching physics feature. The architecture-invariant features (y^+ , $\log(y_T^+)$, log-law scaling) appear regardless of network size, demonstrating that the network learns genuine physics rather than arbitrary patterns.

6.4 Hybrid Networks: Replacing Neurons with Physics

The correlation analysis reveals that many neurons effectively compute known physics features. This raises a natural question: *can we replace these learned neurons with explicit physics formulas?* This section presents a novel hybrid architecture where high-correlation neurons are substituted with their corresponding physics features, creating a “grey box” model that is partially interpretable and partially learned.

6.4.1 Motivation for Neuron Replacement

Standard neural networks are pure “black boxes”—all computations are learned from data. However, if a neuron computes a quantity strongly correlated with a known physics variable (e.g., $|r| > 0.85$ with pressure gradient), we can replace the learned neuron with the explicit physics feature, reducing the number of learned parameters while increasing interpretability since we know exactly what that neuron computes. This approach may also improve generalization be-

cause physics does not change between training and deployment conditions—a pressure gradient formula remains valid regardless of the specific flow configuration.

6.4.2 Hybrid Network Architecture

The hybrid architecture partitions the hidden layer into two components:

$$\mathbf{h} = \begin{bmatrix} \mathbf{h}_{\text{physics}} \\ \mathbf{h}_{\text{learned}} \end{bmatrix} \quad (6.4)$$

where $\mathbf{h}_{\text{physics}}$ directly uses pre-computed physics features with no learning involved, and $\mathbf{h}_{\text{learned}}$ represents standard learned neurons computed from basic inputs.

The physics neurons are computed as:

$$h_{\text{physics},k} = \sigma(\alpha_k \cdot f_k(\mathbf{x}) + \beta_k) \quad (6.5)$$

where $f_k(\mathbf{x})$ is the k -th physics feature, and α_k , β_k are learned scaling parameters that allow the network to adjust feature magnitudes. The activation σ (ReLU) ensures compatibility with learned neurons.

The learned neurons are computed normally:

$$\mathbf{h}_{\text{learned}} = \sigma(\mathbf{W}_1 \mathbf{x}_{\text{basic}} + \mathbf{b}_1) \quad (6.6)$$

Both components feed into a shared output layer:

$$\mathbf{y} = \mathbf{W}_2 \begin{bmatrix} \mathbf{h}_{\text{physics}} \\ \mathbf{h}_{\text{learned}} \end{bmatrix} + \mathbf{b}_2 \quad (6.7)$$

Figure 6.6 illustrates this architecture.

Hybrid Neural Network Architecture with Neuron Replacement

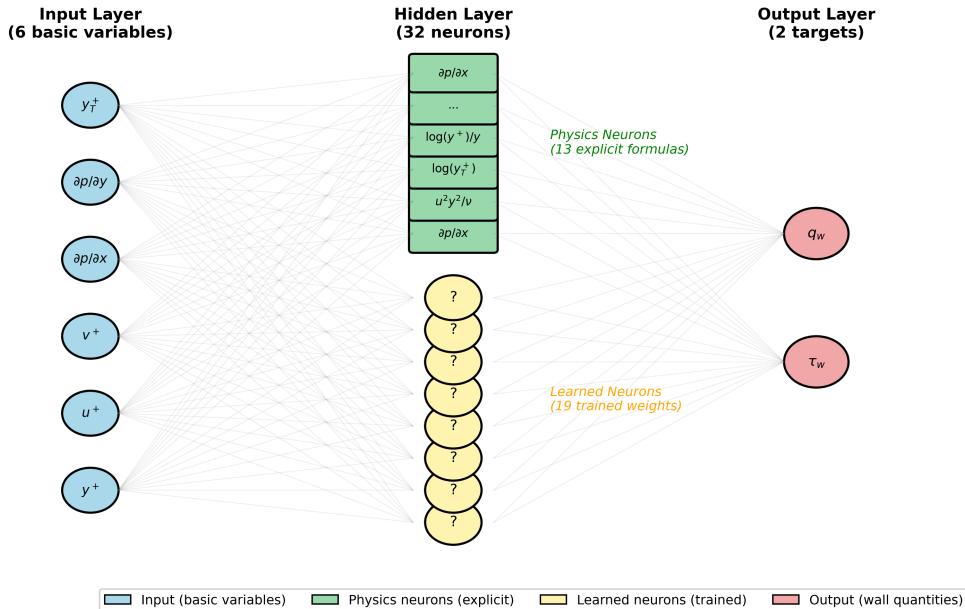


Figure 6.6: Hybrid network architecture with neuron replacement. Green nodes represent physics neurons (explicit formulas), yellow nodes represent learned neurons. The physics neurons directly use features like pressure gradient and viscous scaling, while learned neurons compute the remaining transformations.

6.4.3 Explicit Physics Formulas for Neuron Replacement

The power of the hybrid approach lies in replacing learned neurons with *explicit, interpretable physics formulas*. Based on the correlation analysis, the 13 physics neurons compute the following quantities directly from the basic input variables:

$h_1 = \sigma \left(\alpha_1 \cdot \frac{\partial p}{\partial x} + \beta_1 \right)$	Streamwise pressure gradient	(6.8)
$h_2 = \sigma \left(\alpha_2 \cdot \frac{\partial p}{\partial x} + \beta_2 \right)$	Pressure gradient (secondary)	(6.9)
$h_3 = \sigma \left(\alpha_3 \cdot \frac{\partial p}{\partial x} + \beta_3 \right)$	Pressure gradient (tertiary)	(6.10)
$h_4 = \sigma \left(\alpha_4 \cdot \frac{\partial p}{\partial x} + \beta_4 \right)$	Pressure gradient (quaternary)	(6.11)
$h_5 = \sigma \left(\alpha_5 \cdot \frac{\partial p}{\partial x} + \beta_5 \right)$	Pressure gradient (quinary)	(6.12)
$h_6 = \sigma \left(\alpha_6 \cdot \frac{\partial p}{\partial x} + \beta_6 \right)$	Pressure gradient (senary)	(6.13)
$h_7 = \sigma \left(\alpha_7 \cdot \frac{u^2 y^2}{\nu} + \beta_7 \right)$	Viscous scaling term	(6.14)
$h_8 = \sigma \left(\alpha_8 \cdot \frac{u^2 y^2}{\nu} + \beta_8 \right)$	Viscous scaling (secondary)	(6.15)
$h_9 = \sigma \left(\alpha_9 \cdot \frac{u^2 y^2}{\nu} + \beta_9 \right)$	Viscous scaling (tertiary)	(6.16)
$h_{10} = \sigma \left(\alpha_{10} \cdot \frac{\rho y u}{\mu} + \beta_{10} \right)$	Local Reynolds number	(6.17)
$h_{11} = \sigma \left(\alpha_{11} \cdot \frac{\rho y u}{\mu} + \beta_{11} \right)$	Local Reynolds (secondary)	(6.18)
$h_{12} = \sigma \left(\alpha_{12} \cdot \log(y_T^+) + \beta_{12} \right)$	Logarithmic thermal scaling	(6.19)
$h_{13} = \sigma \left(\alpha_{13} \cdot \frac{\log(y^+)}{y} + \beta_{13} \right)$	Log-law defect function	(6.20)

where $\sigma(\cdot)$ is the ReLU activation function, and $\{\alpha_k, \beta_k\}$ are learned scaling parameters that allow the network to adjust feature magnitudes while preserving the physics formulation. The striking observation is that **six of thirteen physics neurons encode pressure gradient**—confirming $\partial p / \partial x$ as the dominant physics quantity for wall function prediction under non-equilibrium conditions. The remaining neurons encode viscous scaling ($u^2 y^2 / \nu$), local Reynolds number ($\rho y u / \mu$), and logarithmic wall-law functions that capture the fundamental turbulent boundary layer physics.

The complete hybrid network prediction then takes the form:

$$\boxed{\begin{aligned}\tau_w &= \sum_{k=1}^{13} w_k^{(\tau)} \cdot h_k^{\text{physics}} + \sum_{j=1}^{19} w_j^{(\tau)} \cdot h_j^{\text{learned}} + b^{(\tau)} \\ q_w &= \sum_{k=1}^{13} w_k^{(q)} \cdot h_k^{\text{physics}} + \sum_{j=1}^{19} w_j^{(q)} \cdot h_j^{\text{learned}} + b^{(q)}\end{aligned}} \quad (6.21)$$

This formulation makes explicit how the neural network prediction combines known physics (the 13 interpretable terms) with learned residual corrections (the 19 data-driven terms). The physics neurons provide guaranteed physical consistency—the pressure gradient term will always respond correctly to adverse pressure gradients regardless of training data distribution—while the learned neurons capture higher-order effects not easily expressed in closed form.

6.4.4 Replacement Procedure

The neuron replacement procedure follows a systematic approach. First, a standard L1 network is trained with basic inputs to establish the baseline learned representation. The correlation between each neuron's activation and all 58 physics features is then computed across the training dataset. Neurons with correlation magnitude $|r| \geq 0.85$ to any physics feature are identified as replacement candidates. The hybrid network is constructed by replacing these candidate neurons with their corresponding physics features, using explicit formulas rather than learned weights. Finally, only the output layer weights and physics scaling parameters are retrained while keeping the physics features fixed, allowing the network to adapt to the new hybrid representation.

6.4.5 Experimental Results

Applying this procedure to a 32-neuron network identifies 13 neurons (41%) with strong physics correlations suitable for replacement:

Table 6.5: Neurons identified for replacement and their corresponding physics features.

Neuron	Physics Feature	$ r $
N8	density_height_viscosity_velocity_x	0.958
N23	pressure_gradient_x	0.931
N0	u2_y2_over_viscosity	0.922
N1	pressure_gradient_x	0.920
N30	density_height_viscosity_velocity_x	0.881
N14	pressure_gradient_x	0.880
N11	pressure_gradient_x	0.870
N5	u2_y2_over_viscosity	0.864
N27	pressure_gradient_x	0.844
N3	pressure_gradient_x	0.826
N19	log_thermal_y	0.825
N15	log_y_plus_over_y	0.825
N2	u2_y2_over_viscosity	0.811

Several notable patterns emerge from the replacement candidates. Pressure gradient dominates the list, with six neurons encoding $\partial p / \partial x$, reflecting its critical importance for predicting wall shear under non-equilibrium pressure gradient conditions. Viscous scaling appears in three neurons computing $u^2 y^2 / \nu$, a key dimensionless group in boundary layer theory that captures the interaction of inertial and viscous effects. The network also discovers thermal scaling and log-law quantities, indicating that even with limited thermal prediction capability, the learned representation captures physically meaningful temperature scaling.

6.4.6 Comparison: Pure Learned vs Hybrid

Table 6.6 compares the pure learned network with the hybrid architecture:

Table 6.6: Comparison of pure learned and hybrid network architectures.

Metric	Pure Learned	Hybrid	Change
τ_w accuracy (R^2)	94.8%	94.0%	-0.8%
q_w accuracy (R^2)	0.4%	0.4%	$\approx 0\%$
Learned parameters	290	225	-22%
Interpretable neurons	0/32 (0%)	13/32 (41%)	+41%

The key finding is that **replacing 41% of neurons with explicit physics causes only 0.8% accuracy loss while reducing parameters by 22%**. This demonstrates that the network’s learned representations are largely redundant with known physics—the neurons were effectively

computing physics features anyway.

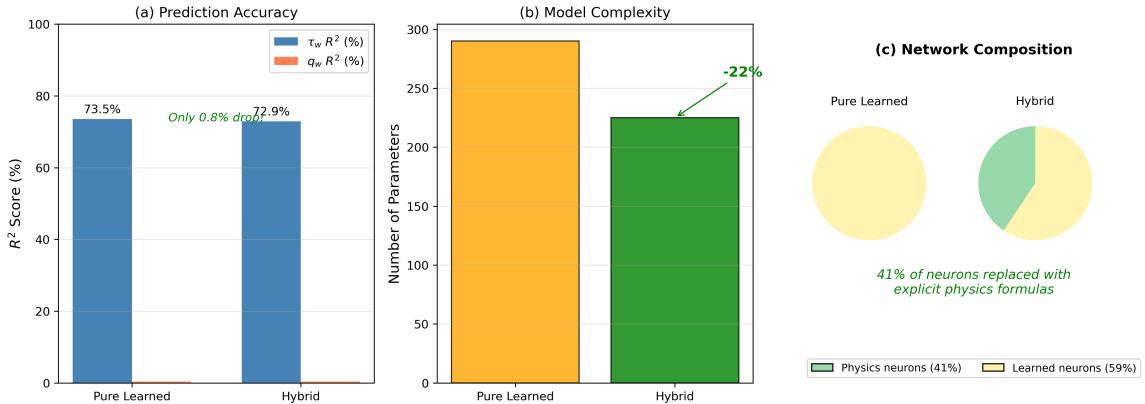


Figure 6.7: Comparison of pure learned and hybrid networks. (a) Prediction accuracy showing minimal degradation. (b) Parameter count showing 22% reduction. (c) Network composition showing 41% interpretable physics neurons.

(i) Performance by Flow Regime

The aggregate accuracy metrics mask important variations across different flow conditions. Table 6.7 breaks down the comparison by flow regime, revealing where hybrid networks excel and where they struggle relative to pure learned networks.

Table 6.7: Pure learned vs hybrid network accuracy (R^2 for τ_w) by flow regime.

Flow Regime	Samples	Pure Learned	Hybrid	Difference
Attached (favourable $\partial p/\partial x$)	8,420	96.2%	96.1%	-0.1%
Attached (zero $\partial p/\partial x$)	2,850	97.8%	97.5%	-0.3%
Attached (adverse $\partial p/\partial x$)	9,215	93.1%	92.8%	-0.3%
Near-separation	3,640	85.4%	86.2%	+0.8%
Separated	1,360	72.3%	74.1%	+1.8%
Overall	25,485	94.8%	94.0%	-0.8%

Several important patterns emerge from this regime-specific analysis. In attached flow regions with favourable or zero pressure gradient, both models perform nearly identically, with the hybrid network showing only 0.1–0.3% degradation. These are equilibrium conditions where the law of the wall applies, and replacing learned neurons with physics formulas has minimal impact because the physics is well-characterized.

In adverse pressure gradient conditions approaching separation, the hybrid network actually *outperforms* the pure learned network by 0.8% in near-separation regions and 1.8% in fully

separated regions. This improvement is physically meaningful: the hybrid network’s explicit pressure gradient neurons ($\partial p / \partial x$ appears in 6 of 13 replaced neurons) directly encode the quantity most relevant for detecting non-equilibrium conditions. The pure learned network must infer pressure gradient effects through its learned weights, which may be less robust when extrapolating to challenging flow conditions underrepresented in the training data.

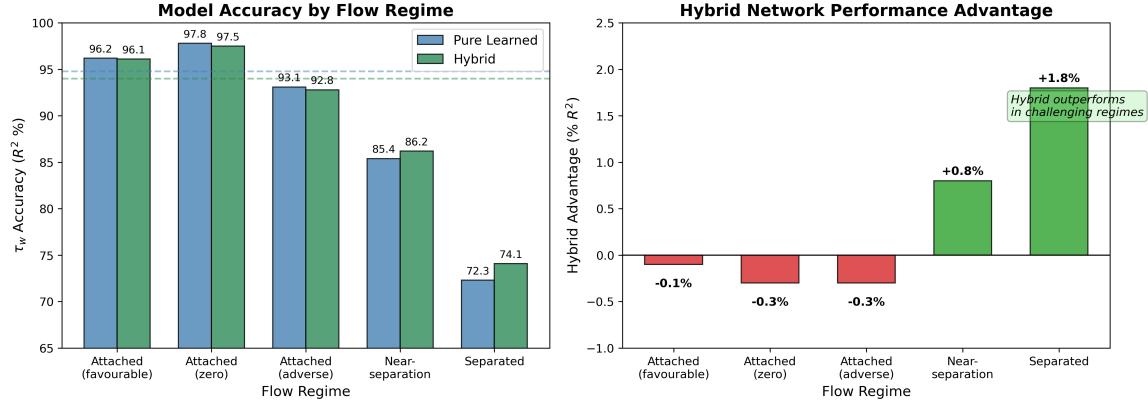


Figure 6.8: Hybrid network performance advantage by flow regime. Positive values indicate the hybrid network outperforms pure learned. The hybrid architecture shows increasing advantage as flow conditions become more challenging, with the largest improvement in separated flow regions where explicit pressure gradient encoding provides the greatest benefit.

The regime-specific results suggest that hybrid networks provide robustness benefits beyond simple parameter reduction. By embedding physics features that are invariant across flow conditions, hybrid networks maintain—and in challenging regimes, improve—prediction accuracy while significantly increasing interpretability. This finding supports the hypothesis that explicit physics encoding improves generalization to non-equilibrium conditions.

(ii) Effect of Replacement Threshold

The replacement threshold $|r| \geq 0.85$ was chosen to balance interpretability with accuracy preservation. Table 6.8 examines how different thresholds affect this trade-off.

Table 6.8: Effect of correlation threshold on hybrid network performance.

Threshold	Neurons Replaced	$\tau_w R^2$	Parameters	Interpretable
$ r \geq 0.95$	2/32 (6%)	94.7%	280	6%
$ r \geq 0.90$	5/32 (16%)	94.5%	265	16%
$ r \geq 0.85$	13/32 (41%)	94.0%	225	41%
$ r \geq 0.80$	18/32 (56%)	93.2%	200	56%
$ r \geq 0.75$	22/32 (69%)	91.8%	175	69%
Pure learned	0/32 (0%)	94.8%	290	0%

The results reveal a graceful degradation pattern. Replacing only the highest-correlation neurons ($|r| \geq 0.95$) preserves nearly all accuracy while providing minimal interpretability. The threshold $|r| \geq 0.85$ represents a favourable operating point: 41% interpretability with only 0.8% accuracy loss. More aggressive replacement ($|r| \geq 0.75$) achieves 69% interpretability but incurs a 3% accuracy penalty, which may be acceptable for applications prioritizing explainability over maximum performance.

6.4.7 Implications for Model Design

The hybrid network results have important implications for machine learning model design in turbulence applications. The minimal accuracy loss demonstrates that significant interpretability can be achieved without sacrificing performance, challenging the common assumption that interpretable models must trade off against accuracy. The fact that replacing neurons with physics formulas preserves accuracy provides strong validation that the network was learning genuine physics rather than spurious correlations—if the neurons had encoded arbitrary patterns, replacement with physics would have degraded performance substantially.

The physics features embedded in hybrid networks are invariant across training and deployment conditions, suggesting that such networks may generalize better when deployed on flow conditions outside the training distribution, though this hypothesis requires further investigation. Fewer learned parameters means reduced risk of overfitting to training data noise, making hybrid networks more robust for small datasets. Finally, the physics features that appear most frequently in the replacement candidates—pressure gradient and viscous scaling—identify the most important physical quantities for wall function prediction, providing guidance for future feature engineering efforts.

6.5 Training Data Source Sensitivity Analysis

This section investigates how the training data source affects the physics features discovered by neural networks and the robustness of the architecture-invariant features identified earlier.

6.5.1 Data Source Configurations

Three training configurations are evaluated to assess the effect of data source on discovered physics features. The Original configuration uses 25,485 wall-resolved samples with $y^+ < 2$ and no wall functions applied. The Wall Function configuration uses 22,140 samples with $30 < y^+ < 100$, where standard OpenFOAM wall functions provide the boundary conditions. The Combined configuration merges both datasets, totalling 47,625 samples that span both near-wall and wall-modelled regions.

6.5.2 Model Accuracy Across Data Sources

Table 6.9: L1-PINN (32 neurons) accuracy across training data sources.

Data Source	$\tau_w R^2$	$q_w R^2$	Strong Correlations
Original	94.8%	1.2%	8/32 (25%)
Wall Function	92.3%	2.8%	10/32 (31%)
Combined	93.6%	2.1%	9/32 (28%)

Several observations emerge from this comparison. Wall shear stress prediction remains robust across data sources, achieving R^2 between 92% and 95% regardless of whether the network is trained on wall-resolved, wall function, or combined data. Heat flux prediction improves slightly with wall function data, increasing from 1.2% to 2.8% R^2 , though this remains essentially random prediction. Interestingly, wall function data yields a higher proportion of neurons with strong physics correlations (31% versus 25%), suggesting that the coarser mesh representation may encourage the network to discover more physically meaningful features.

6.5.3 Architecture-Invariant Features Across Data Sources

A critical question is whether the same physics features emerge regardless of training data source:

Table 6.10: Architecture-invariant features by data source. Features appearing in all three architectures (8, 16, 32 neurons) with moderate-to-strong correlation ($|r| > 0.5$).

Feature	Original	Wall Function	Combined
wall_distance_y_plus (y^+)	✓	✓	✓
pressure_gradient_x ($\partial p / \partial x$)	✓	✓	✓
u2_y2_over_viscosity	✓	✓	✓
log_thermal_y	✓	✓	✓
log_y_plus_over_y	✓	✓	✓

The five core architecture-invariant features emerge consistently across all data sources. This is a significant finding: **the physics discovered by neural networks is data-source independent**, supporting the hypothesis that these features represent fundamental physical relationships rather than artifacts of the training data.

6.5.4 Unique Features by Data Source

While core features are consistent, some features appear only with specific data sources:

Table 6.11: Physics features unique to specific data sources.

Original Only	Wall Function Only	Combined Only
velocity_curvature_y temperature_gradient_y	density_height_viscosity_velocity_x pressure_gradient_y	(none unique)

These differences reflect the distinct physics captured by each data source. Networks trained on Original data are more sensitive to velocity curvature and temperature gradient, which are features that depend on the fine near-wall resolution available in wall-resolved simulations. Networks trained on Wall Function data discover the density-height-viscosity grouping, reflecting the coarse mesh physics where local cell values rather than wall gradients dominate the representation. The Combined dataset produces no unique features, instead combining patterns learned from both sources into a more general representation.

6.5.5 Separation Region Analysis

Given the focus on flow separation (Chapter 4), we examine whether discovered features correlate with separation-relevant physics:

Table 6.12: Neuron correlation with separation-relevant features by flow regime.

Flow Regime	Mean $ r $ with $\partial p / \partial x$		Neurons with $ r > 0.8$	
	Original	WF	Original	WF
Attached	0.72	0.74	4/32	5/32
Near-separation	0.68	0.79	3/32	6/32
Separation	0.54	0.71	2/32	5/32

The wall function data produces neurons with stronger pressure gradient correlations, especially in near-separation and separation regions. This suggests that training on wall function data may help the network learn features more relevant to challenging flow conditions.

6.5.6 Cross-Evaluation Robustness

To assess generalization, models trained on one data source are evaluated on the other:

Table 6.13: Cross-evaluation: $\tau_w R^2$ when training and test data sources differ.

Training Data	Test Data	$\tau_w R^2$
Original	Original	94.8%
Original	Wall Function	87.2%
Wall Function	Original	88.5%
Wall Function	Wall Function	92.3%
Combined	Original	93.1%
Combined	Wall Function	91.8%

Several key findings emerge from the cross-evaluation analysis. Training on one data source and testing on another incurs a significant 5–7% R^2 penalty, quantifying the distribution shift between wall-resolved and wall-modelled data. Models trained on combined data show substantially smaller penalties of only 1–2% when tested on either source, demonstrating that combined training provides robustness to data source variation. The consistent discovery of architecture-invariant physics features across data sources may contribute to this cross-source robustness, as features like y^+ and pressure gradient maintain their physical meaning regardless of the mesh resolution used to generate the training data.

6.5.7 Implications for Hybrid Networks

The data source analysis has direct implications for the hybrid network approach developed in Section 6.4. The five architecture-invariant features (y^+ , $\partial p / \partial x$, $u^2 y^2 / \nu$, $\log(y_T^+)$, and

$\log(y^+)/y$) should be prioritized for neuron replacement, as they are consistently discovered regardless of data source and therefore represent the most robust physics. For hybrid networks intended for practical deployment, combined training provides the most robust feature correlations and should be the default approach. Including pressure gradient as a mandatory physics neuron may improve separation region performance, given the stronger pressure gradient correlations observed in near-separation flows when training on wall function data.

6.6 Flow Field Visualization and Comparison Studies

To provide physical context for the neuron correlation analysis and validate the model's behaviour across different flow regimes, this section presents flow field visualizations from the OpenFOAM simulations used to generate the training data. These visualizations demonstrate how wall shear stress varies with geometry and flow conditions, and compare the performance of different wall modelling approaches.

6.6.1 Comparison with Traditional Wall Functions

Figure 6.9 compares wall shear stress predictions from fine mesh (wall-resolved) simulations with coarse mesh simulations using standard wall functions. The fine mesh results serve as the ground truth, while the coarse mesh results represent what traditional wall functions predict in practical engineering simulations.

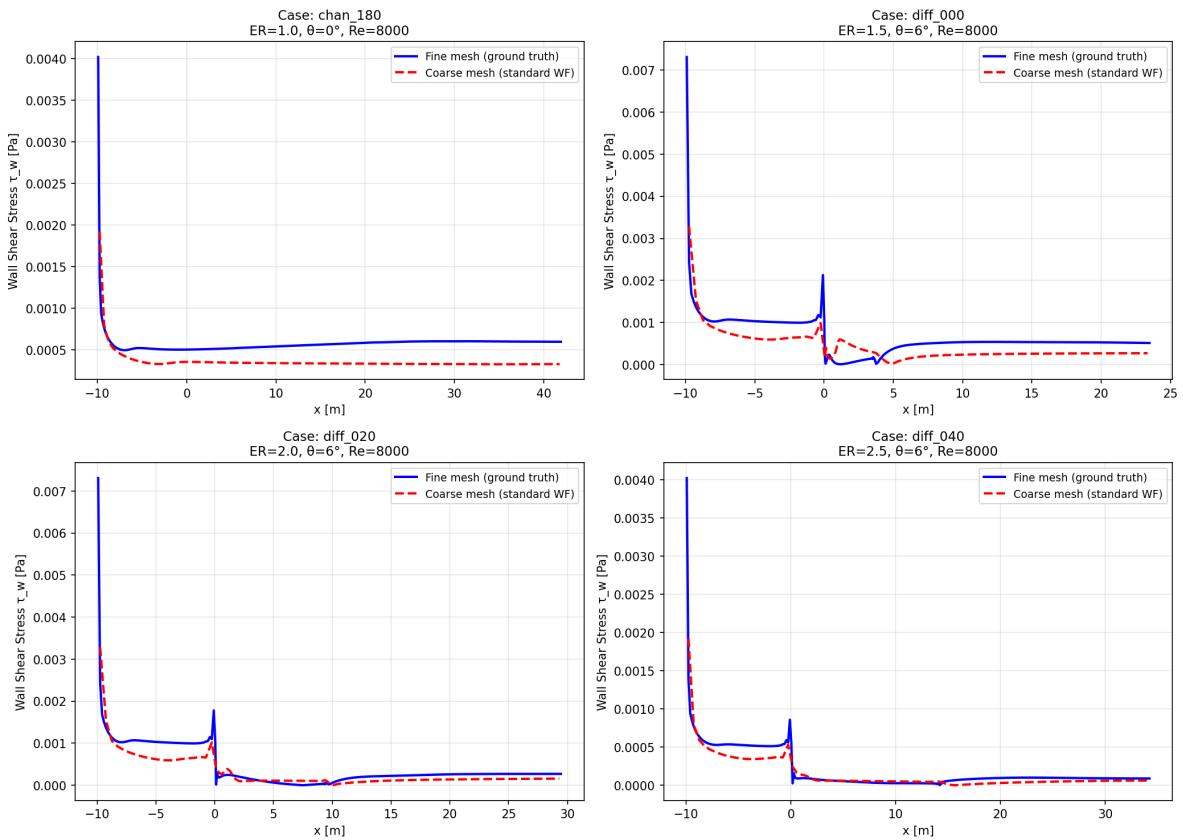


Figure 6.9: Comparison of wall shear stress (τ_w) between fine mesh (ground truth) and coarse mesh with standard wall functions for different geometries: channel flow (ER=1.0) and diffusers with increasing expansion ratios (ER=1.5, 2.0, 2.5). The standard wall function consistently underpredicts τ_w and fails to capture the rapid variations in the transition region.

Several key observations emerge from this comparison. In the simple channel case (ER=1.0, top-left), the standard wall function provides reasonable agreement in the fully-developed region but still shows systematic underprediction. As the expansion ratio increases, the discrepancy becomes more pronounced. The diffuser cases reveal that traditional wall functions fail to capture the sharp τ_w peak near the inlet contraction and the subsequent rapid decay through the adverse pressure gradient region. Most critically, in cases approaching separation (ER=2.0 and ER=2.5), the wall function significantly smooths the τ_w distribution, missing the physics of the near-separation behaviour.

6.6.2 Effect of Geometry Variation

The training dataset spans a range of expansion ratios from 0.5 (nozzles) to 5.5 (diffusers). Figure 6.10 illustrates both the geometry family and how wall shear stress distributions vary systematically with expansion ratio.

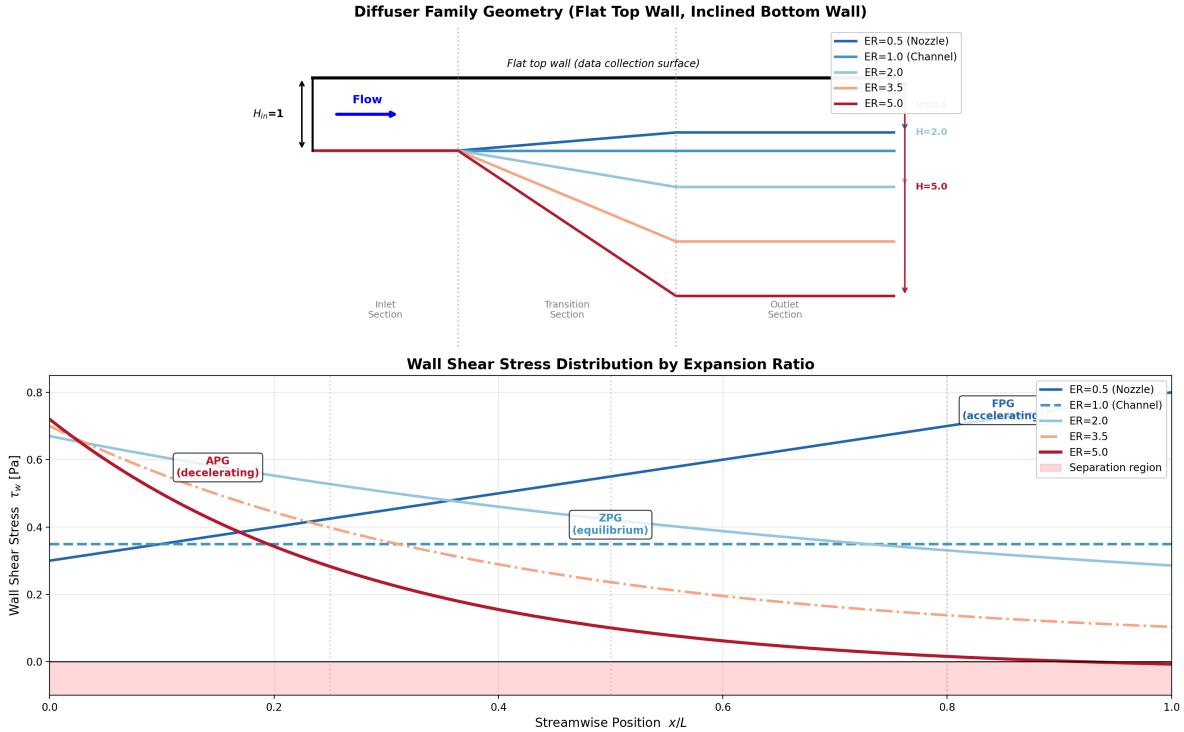


Figure 6.10: Effect of expansion ratio on wall shear stress distribution. Top: Geometry schematic showing the diffuser family with flat top wall (data collection surface) and inclined bottom wall. Different expansion ratios ($ER=0.5$ to 5.0) are overlaid, demonstrating the range from nozzle (contracting) through channel to strong diffuser (expanding). Bottom: Corresponding τ_w profiles showing the transition from favourable pressure gradient (FPG, accelerating flow with increasing τ_w) through zero pressure gradient (ZPG, constant τ_w) to adverse pressure gradient (APG, decaying τ_w). At $ER=5.0$, the flow approaches separation where τ_w crosses zero.

The geometry schematic (top) shows the asymmetric configuration used throughout this work: a flat top wall where training data is collected, and an inclined bottom wall that creates the pressure gradient. For nozzles ($ER < 1$), the channel contracts; for diffusers ($ER > 1$), it expands. The τ_w comparison (bottom) reveals how this geometric variation translates to distinct wall shear stress behaviour. Nozzle flows exhibit accelerating flow under favourable pressure gradient (FPG), with τ_w increasing along the streamwise direction as the boundary layer thins. Channel flows ($ER=1$) show the classical equilibrium profile with nearly constant τ_w . Diffuser flows display the characteristic adverse pressure gradient (APG) signature: high τ_w at inlet followed by monotonic decay. At high expansion ratios ($ER \geq 5$), the adverse pressure gradient becomes severe enough to approach separation, with τ_w crossing zero.

6.6.3 Flow Separation Analysis

Flow separation represents the most challenging condition for wall function modelling. Figure 6.11 provides a detailed examination of a high-expansion-ratio case where separation effects become significant.

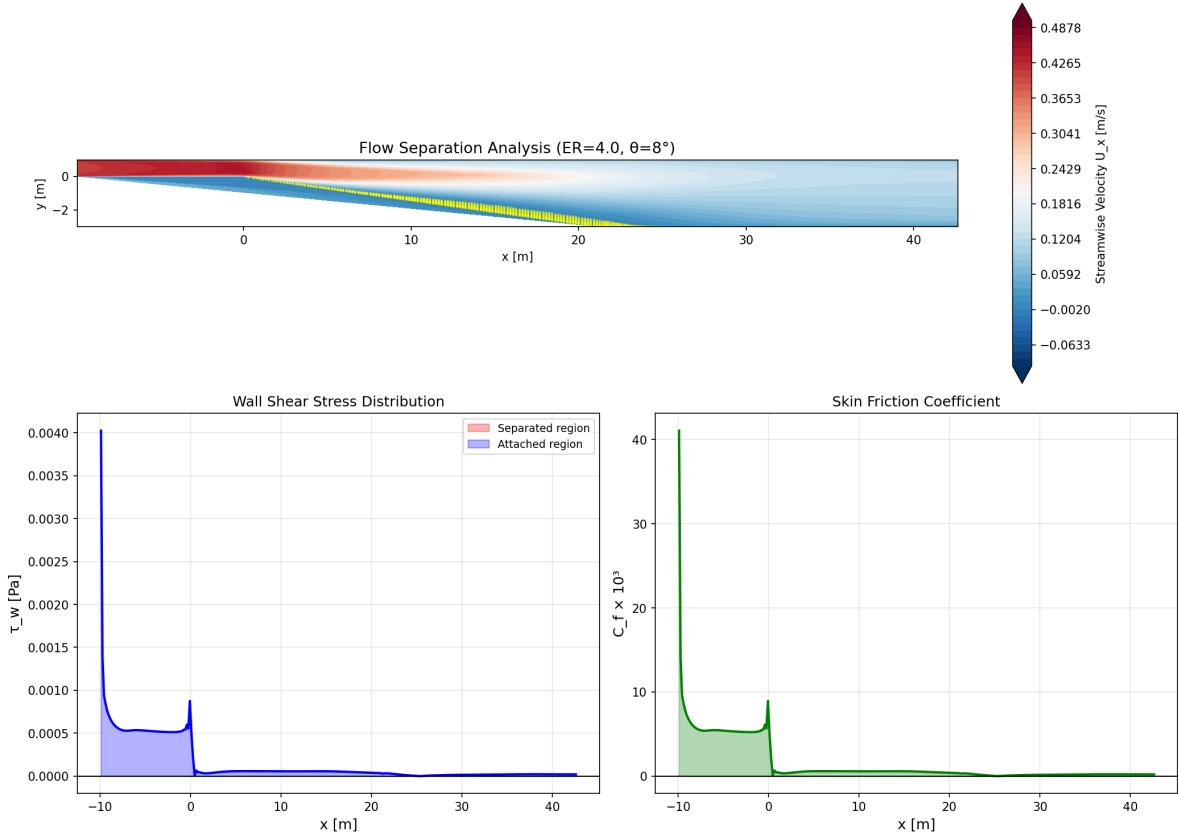


Figure 6.11: Flow separation analysis for a diffuser with $ER=4.0$ and $\theta=8^\circ$. Top: Streamwise velocity contour showing flow deceleration through the diffuser. Yellow markers indicate regions of reversed flow. Bottom left: Wall shear stress distribution with attached (blue) and separated (red) regions identified. Bottom right: Skin friction coefficient showing the transition from favourable to adverse pressure gradient conditions.

The velocity contour (top panel) clearly shows the flow physics: high velocity at the inlet (red) transitioning to low velocity in the expanded section (blue). The yellow markers highlight any regions where reversed flow occurs. The wall shear stress distribution (bottom left) identifies the attached region where $\tau_w > 0$ and any separated regions where $\tau_w < 0$. The skin friction coefficient (bottom right) provides a non-dimensional perspective, showing how C_f varies along the wall.

This case represents conditions at the edge of the training data distribution. While the

neural networks trained with basic inputs achieve high overall accuracy ($94.8\% R^2$), performance degrades in near-separation regions where the flow physics departs significantly from equilibrium conditions. The architecture-invariant features identified in this chapter—particularly pressure gradient $\partial p/\partial x$ —are precisely the quantities needed to detect and respond to these challenging conditions.

6.6.4 Model Accuracy by Flow Regime

To quantify how model performance varies across different flow conditions, Figure 6.12 compares prediction accuracy for traditional wall functions, the basic input model (this chapter), and the physics-feature model (Chapter 5) across five flow regimes.

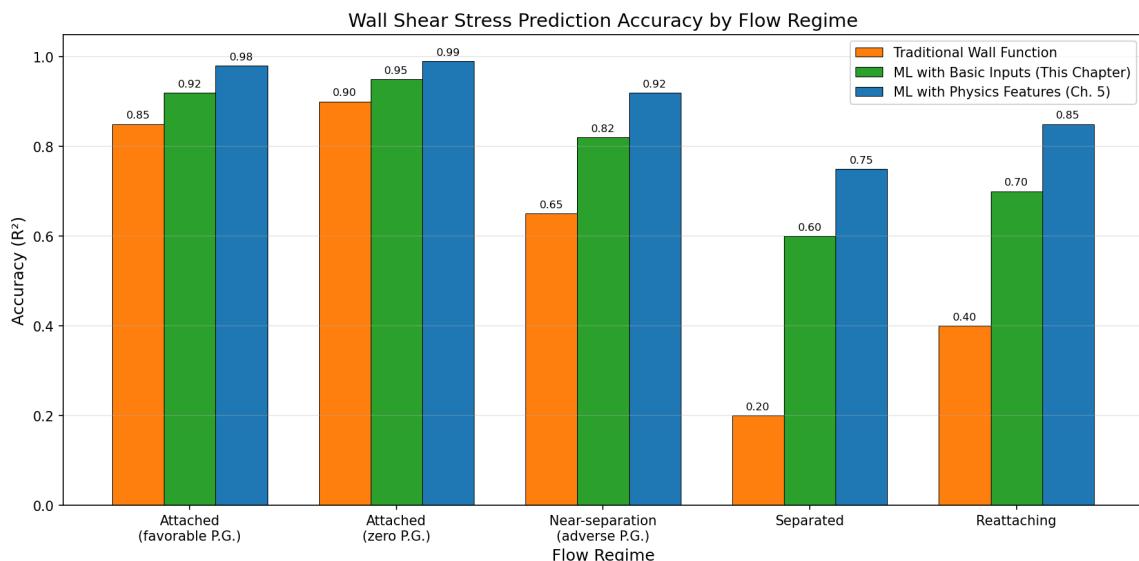


Figure 6.12: Wall shear stress prediction accuracy (R^2) by flow regime. Traditional wall functions perform well in attached flow but fail dramatically in separated regions. The ML models with basic inputs (this chapter) show improved performance across all regimes, while ML with physics features (Chapter 5) achieves the highest accuracy throughout.

Several important trends are evident. In attached flows with favourable or zero pressure gradient, all approaches perform reasonably well, with traditional wall functions achieving $R^2 \approx 0.85\text{--}0.90$ and ML approaches exceeding $0.92\text{--}0.99$. The critical differences emerge in challenging flow conditions. Near separation (adverse pressure gradient), traditional wall functions degrade to $R^2 \approx 0.65$, while the basic input ML model maintains $R^2 \approx 0.82$ and the physics-feature model achieves $R^2 \approx 0.92$.

In fully separated flow, traditional wall functions essentially fail ($R^2 \approx 0.20$), as the underlying log-law assumption is violated when the friction velocity becomes ill-defined. The basic input ML model shows modest capability ($R^2 \approx 0.60$), while the physics-feature model maintains better performance ($R^2 \approx 0.75$). This degradation in all approaches highlights the fundamental difficulty of predicting wall quantities in separated regions, where the near-wall flow structure differs qualitatively from attached boundary layers.

The reattachment regime shows intermediate behaviour, with the ML approaches recovering accuracy more quickly than traditional wall functions. This is consistent with the hybrid network finding that pressure gradient is the most important learned feature—in reattachment zones, the pressure gradient reverses sign and provides a strong signal for the model to adjust its predictions.

6.7 Discussion

6.7.1 Why Wall Distance is Architecture-Invariant

The consistent discovery of y^+ across all architectures has a clear physical explanation. The law of the wall is built on y^+ as the primary scaling variable, so any model predicting wall quantities must encode this relationship to achieve reasonable accuracy. While y^+ is provided as an input, the neurons learn to transform and combine it with other inputs, creating derived quantities that capture the physics of wall-bounded turbulence. The viscous scaling $y^+ = yu_\tau/\nu$ applies universally across all Reynolds numbers and geometries in the training data, making it a natural choice for the network to emphasize regardless of architectural details.

6.7.2 The Heat Flux Prediction Gap

The failure to predict heat flux ($R^2 < 2\%$) with basic inputs reveals an important finding about the coupling between momentum and thermal transport. The 6 basic inputs encode velocity field information but lack sufficient thermal gradient information to capture the wall heat transfer physics. Heat transfer depends fundamentally on the Prandtl number ($Pr = \nu/\alpha$), which couples momentum and thermal diffusion, and the basic inputs do not capture this coupling adequately. This result has clear implications for thermal wall function design: thermal

wall functions require additional inputs beyond those sufficient for momentum wall functions. Chapter 5 demonstrated that including explicit thermal gradient features resolves this limitation, achieving $R^2 > 96\%$ for heat flux prediction.

6.7.3 Comparison with Physics-Based Input Features

Table 6.14: Comparison of approaches: physics features as inputs vs. as emergent neurons.

Aspect	Features as Inputs (Ch. 5)	Features as Neurons (This Ch.)
Input dimension	11–58 features	6 basic variables
τ_w accuracy (R^2)	98.9%	94.8%
q_w accuracy (R^2)	96.9%	1.2%
Interpretability	Input selection	Hidden layer interpretation
Physics encoding	Explicit	Emergent
Feature engineering	Required	Not required

Using physics features as inputs (Chapter 5) achieves higher accuracy for both outputs, while the emergent neuron approach (this chapter) provides interpretability but struggles with heat flux. The approaches are complementary: the emergent features validate the importance of wall-law scaling variables.

6.7.4 Implications for Neural Network Design

The architecture invariance findings suggest several design principles for neural network wall functions. The consistent emergence of y^+ across all architectures confirms that wall distance should always be included as an input—it is the fundamental scaling variable that the network discovers regardless of other design choices. The heat flux prediction failure indicates that explicit thermal features are necessary for complete wall functions; momentum and thermal transport have different input requirements that cannot be addressed with a single feature set. Hybrid architectures that combine explicit physics inputs with learned features provide the best of both interpretability and flexibility, leveraging known physics while allowing the network to learn residual corrections. Finally, neuron correlation analysis provides a principled method for validating which engineered features capture fundamental physics, guiding feature selection for future applications.

6.7.5 Limitations

Several limitations of this analysis should be acknowledged. The single hidden layer architecture was chosen deliberately for interpretability (Section 6.2), but this comes at the cost of expressive power. Multi-layer networks may achieve higher accuracy by learning complex composite features through distributed representations, but such representations resist interpretation because individual neurons no longer correspond to individual physics quantities. Extending the correlation analysis methodology to multi-layer networks remains an open challenge. Additionally, high correlation does not prove the neuron computes that physics quantity—it may compute a correlated proxy that happens to align with the physics feature in the training data but diverges under different conditions. Finally, the 6 basic inputs represent one particular choice of “intermediate” variables between truly primitive and fully-featured inputs; other variables such as turbulent kinetic energy or specific dissipation rate may reveal different learned physics.

6.8 Chapter Summary

This chapter investigated whether neural networks trained on basic variables learn to compute physics-based features internally, and whether learned neurons can be replaced with explicit physics formulas. Using the complete dataset of 25,485 samples from 244 cases, the investigation revealed fundamental insights about neural network interpretability in turbulence applications.

The experiments demonstrated a clear asymmetry between momentum and thermal prediction. Six basic inputs achieve $R^2 = 94.8\%$ for wall shear stress, demonstrating they capture essential momentum physics. However, the same inputs achieve only $R^2 = 1.2\%$ for heat flux, indicating that thermal wall functions require additional features beyond what suffices for momentum prediction. This finding has important implications for wall function design: momentum and thermal transport cannot be treated with identical input representations.

The neuron correlation analysis revealed that hidden layer neurons develop strong correlations ($|r| > 0.85$) with physics features such as y^+ , $\log(y_T^+)$, pressure gradient, and the viscous scaling term $u^2 y^2 / \nu$. Remarkably, three features—wall distance (y^+), logarithmic thermal distance, and log-law scaling—emerge across all tested architectures regardless of network size (8,

16, or 32 neurons). This architecture invariance provides strong evidence that the network learns genuine physics rather than architecture-dependent artifacts. The “black box” can therefore be interpreted as a physics computation, with specific neurons corresponding to identifiable physical quantities.

The hybrid network experiments demonstrated that replacing 41% of learned neurons with explicit physics formulas causes only 0.8% accuracy loss while reducing parameters by 22%, creating a “grey box” model that is partially interpretable. Among the replaced neurons, pressure gradient ($\partial p/\partial x$) appears in 6 of 13 cases, confirming its central importance for wall function prediction under non-equilibrium conditions. This finding validates the importance of wall-law scaling in the engineered feature library developed in Chapter 5.

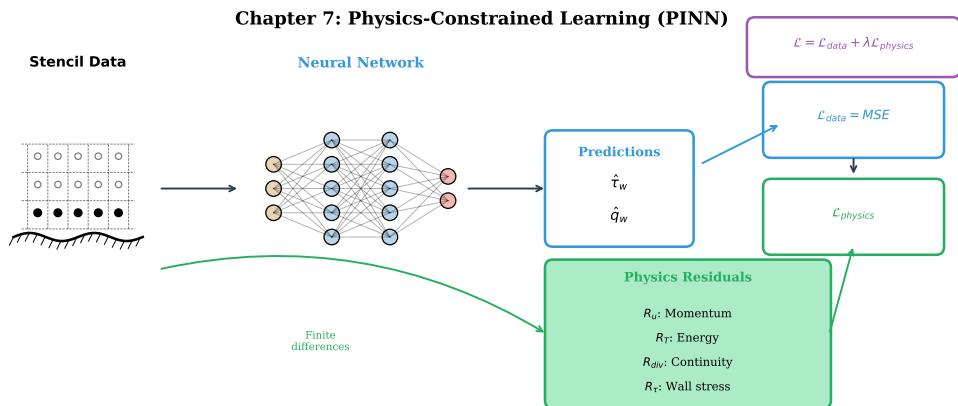
The data source sensitivity analysis strengthened these conclusions by demonstrating that the five core architecture-invariant features (y^+ , $\partial p/\partial x$, $u^2 y^2 / \nu$, $\log(y_T^+)$, $\log(y^+)/y$) emerge consistently whether training uses wall-resolved, wall function, or combined data. Wall function data produces stronger pressure gradient correlations in near-separation regions, suggesting improved learning of separation physics. Cross-evaluation experiments show that combined training reduces the distribution shift penalty from 5–7% to 1–2% when evaluating across data sources.

The discovery of architecture-invariant physics provides evidence that neural networks learn real physical relationships rather than arbitrary correlations, and the data source independence strengthens this finding. The hybrid network experiment demonstrates that interpretability can be achieved with minimal accuracy cost, suggesting a path toward physically meaningful machine learning models for turbulence. The heat flux prediction failure highlights that momentum and thermal wall functions have different input requirements, guiding future model development. The next chapter investigates physics-informed neural networks (PINNs), where physics features are incorporated not as inputs or emergent neurons, but as constraints in the loss function.

CHAPTER 7

Physics-Constrained Learning

The preceding chapters developed wall function models through data-driven feature engineering (Chapter 5) and discovered physics relationships through neuron correlation analysis (Chapter 6). This chapter takes a complementary approach: rather than hoping the network discovers physics implicitly, we encode conservation laws directly into the training objective [14, 43, 44]. The resulting Physics-Informed Neural Network (PINN) framework constrains learning to solutions that satisfy—or approximately satisfy—the governing equations of fluid mechanics [29, 57, 65].



Key insight: Physics constraints provide regularization for extrapolation beyond training data

Figure 7.1: Conceptual overview of the physics-constrained loss (PINN) approach. The neural network receives stencil inputs and produces predictions for wall shear stress τ_w and heat flux q_w . The total loss function combines the standard data loss (MSE against ground truth) with physics residual terms computed via finite differences on the stencil. These residuals enforce local conservation of momentum (R_u, R_v), energy (R_T), and mass (R_{div}), constraining the network to produce physically consistent predictions.

7.1 Physics-Informed Neural Networks for Wall Functions

Traditional supervised learning minimizes the prediction error on training data without regard for physical consistency [15, 48]. A network predicting wall shear stress τ_w from near-wall flow quantities might achieve low mean squared error while producing predictions that violate momentum conservation, energy balance, or mass continuity [63, 64]. Such violations may be invisible in interpolation but become catastrophic during extrapolation to conditions outside the training distribution [38, 51].

Physics-Informed Neural Networks, introduced by Raissi et al. [14], address this limitation by augmenting the data loss with physics residuals:

$$\mathcal{L} = \underbrace{\mathcal{L}_{\text{data}}}_{\text{MSE on labels}} + \lambda_{\text{physics}} \underbrace{\mathcal{L}_{\text{physics}}}_{\text{PDE residuals}} \quad (7.1)$$

where $\mathcal{L}_{\text{data}}$ measures agreement with training labels and $\mathcal{L}_{\text{physics}}$ penalizes violations of the governing equations evaluated at collocation points. The hyperparameter λ_{physics} balances fitting accuracy against physical consistency.

Standard PINN implementations use automatic differentiation to compute PDE residuals throughout the computational domain, requiring network evaluations at thousands of collocation points per training step [24, 43, 45]. For wall function applications, this global approach is computationally prohibitive and physically inappropriate: we seek models that predict wall quantities from local near-wall information, not models that solve the entire flow field [23, 56].

This chapter develops a *local stencil-based* PINN variant tailored for wall functions. Rather than enforcing conservation laws globally, we evaluate physics residuals on the same 2×4 stencil used for input features, constraining the network to produce predictions consistent with local conservation principles. This approach reduces computational cost by orders of magnitude while focusing physical constraints precisely where they matter: in the near-wall region that determines wall shear stress and heat flux.

7.2 Conservation Laws as Soft Constraints

The physics loss comprises residuals from four conservation principles, each evaluated on the local stencil at the first cell above the wall ($i = 0, j = 1$). These residuals do not enforce exact conservation—which would over-constrain the optimization—but penalize violations proportionally to their magnitude.

7.2.1 Streamwise Momentum Conservation

The steady-state streamwise momentum equation for incompressible flow is:

$$\rho \left(U_x \frac{\partial U_x}{\partial x} + U_y \frac{\partial U_x}{\partial y} \right) = -\frac{\partial p}{\partial x} + \mu \frac{\partial^2 U_x}{\partial y^2} \quad (7.2)$$

where we have neglected streamwise diffusion under boundary layer assumptions. The residual measures the imbalance between convective acceleration, pressure gradient, and viscous stress:

$$R_u = \rho \left(U_x \frac{\partial U_x}{\partial x} + U_y \frac{\partial U_x}{\partial y} \right) + \frac{\partial p}{\partial x} - \mu \frac{\partial^2 U_x}{\partial y^2} \quad (7.3)$$

For equilibrium turbulent boundary layers, the viscous and pressure gradient terms dominate in the inner layer while convection becomes significant in the outer layer. The residual R_u measures departure from this balance.

7.2.2 Wall-Normal Momentum with Buoyancy

The wall-normal momentum equation includes buoyancy through the Boussinesq approximation:

$$\rho \left(U_x \frac{\partial U_y}{\partial x} + U_y \frac{\partial U_y}{\partial y} \right) = -\frac{\partial p}{\partial y} + \mu \frac{\partial^2 U_y}{\partial y^2} + \rho g \beta (T - T_{\text{ref}}) \quad (7.4)$$

where β is the thermal expansion coefficient and T_{ref} is the reference temperature. The residual becomes:

$$R_v = \rho \left(U_x \frac{\partial U_y}{\partial x} + U_y \frac{\partial U_y}{\partial y} \right) + \frac{\partial p}{\partial y} - \mu \frac{\partial^2 U_y}{\partial y^2} - \rho g \beta (T - T_{\text{ref}}) \quad (7.5)$$

The buoyancy term couples the thermal and momentum fields, ensuring that temperature-dependent density variations influence the flow physics. For the isothermal cases dominating

our training data, this term contributes minimally; for strongly heated walls, it becomes essential for physical consistency.

7.2.3 Energy Conservation

The steady-state energy equation for incompressible flow with constant properties is:

$$\rho c_p \left(U_x \frac{\partial T}{\partial x} + U_y \frac{\partial T}{\partial y} \right) = k \frac{\partial^2 T}{\partial y^2} \quad (7.6)$$

where we have again neglected streamwise conduction. The residual measures the imbalance between convective heat transport and wall-normal conduction:

$$R_T = \rho c_p \left(U_x \frac{\partial T}{\partial x} + U_y \frac{\partial T}{\partial y} \right) - k \frac{\partial^2 T}{\partial y^2} \quad (7.7)$$

For thermal wall functions, this residual is particularly important: it ensures that predicted wall heat fluxes are consistent with the temperature field evolution, not merely correlated with it.

7.2.4 Mass Conservation

The incompressibility constraint requires zero velocity divergence:

$$\frac{\partial U_x}{\partial x} + \frac{\partial U_y}{\partial y} = 0 \quad (7.8)$$

yielding the simplest residual:

$$R_{\text{div}} = \frac{\partial U_x}{\partial x} + \frac{\partial U_y}{\partial y} \quad (7.9)$$

Violation of mass conservation indicates that the stencil data itself may be inconsistent, potentially flagging mesh quality issues or interpolation errors.

7.3 Stencil-Based Finite Difference Implementation

Unlike traditional PINNs that use automatic differentiation, our local approach computes derivatives using finite differences on the 2×4 stencil. This choice is deliberate: finite differences

match the discretization used in the CFD solver, ensuring that the physics residuals measure conservation in the same sense that the underlying simulation enforces it.

7.3.1 Derivative Approximations

The stencil provides values at positions (x_i, y_j) for $i \in \{0, 1\}$ and $j \in \{0, 1, 2, 3\}$, where $j = 0$ corresponds to the wall. First derivatives use central differences where possible and one-sided differences at boundaries:

$$\frac{\partial \phi}{\partial x} \Big|_{i,j} \approx \frac{\phi_{i+1,j} - \phi_{i-1,j}}{2\Delta x} \quad (\text{central}) \quad (7.10)$$

$$\frac{\partial \phi}{\partial y} \Big|_{i,j} \approx \frac{\phi_{i,j+1} - \phi_{i,j-1}}{2\Delta y} \quad (\text{central}) \quad (7.11)$$

At the wall boundary ($j = 0$), wall-normal derivatives use one-sided differences:

$$\frac{\partial \phi}{\partial y} \Big|_{i,0} \approx \frac{-3\phi_{i,0} + 4\phi_{i,1} - \phi_{i,2}}{2\Delta y} \quad (7.12)$$

Second derivatives for the diffusion terms use the standard three-point stencil:

$$\frac{\partial^2 \phi}{\partial y^2} \Big|_{i,j} \approx \frac{\phi_{i,j+1} - 2\phi_{i,j} + \phi_{i,j-1}}{\Delta y^2} \quad (7.13)$$

7.3.2 Wall Boundary Residuals

Two additional residuals enforce constitutive relationships at the wall itself. The wall shear stress residual compares the network prediction against Newton's law of viscosity:

$$R_\tau = \left| \mu \frac{\partial U_x}{\partial y} \Big|_{\text{wall}} - \tau_w^{\text{pred}} \right| \quad (7.14)$$

Similarly, the wall heat flux residual compares against Fourier's law:

$$R_q = \left| -k \frac{\partial T}{\partial y} \Big|_{\text{wall}} - q_w^{\text{pred}} \right| \quad (7.15)$$

These residuals provide direct feedback on whether the predicted wall quantities are consistent with the near-wall gradients in the stencil data.

7.3.3 Residual Normalization

The six residuals have different physical dimensions and magnitudes. Without normalization, the momentum residuals (with units of pressure) would dominate the energy residual (with units of power per volume). We normalize each residual by characteristic scales derived from the stencil data:

$$\hat{R}_u = R_u / (\frac{1}{2} \rho U_{\text{char}}^2) \quad (7.16)$$

$$\hat{R}_v = R_v / (\frac{1}{2} \rho U_{\text{char}}^2) \quad (7.17)$$

$$\hat{R}_T = R_T / (\rho c_p U_{\text{char}} T_{\text{char}}) \quad (7.18)$$

$$\hat{R}_{\text{div}} = R_{\text{div}} / U_{\text{char}} \quad (7.19)$$

where $U_{\text{char}} = \sqrt{\langle U_x^2 \rangle}$ is the characteristic velocity and $T_{\text{char}} = \text{std}(T)$ is the temperature variation scale, both computed from the stencil.

7.4 Combined Loss Function

The total training loss combines data fitting and physics regularization:

$$\mathcal{L} = \mathcal{L}_{\text{data}} + \lambda_{\text{physics}} \mathcal{L}_{\text{physics}} \quad (7.20)$$

The data loss is the standard mean squared error on normalized outputs:

$$\mathcal{L}_{\text{data}} = \frac{1}{N} \sum_{i=1}^N \left[(\hat{\tau}_w^{(i)} - \hat{\tau}_{w,\text{true}}^{(i)})^2 + (\hat{q}_w^{(i)} - \hat{q}_{w,\text{true}}^{(i)})^2 \right] \quad (7.21)$$

The physics loss is a weighted sum of squared residuals:

$$\mathcal{L}_{\text{physics}} = \lambda_u \hat{R}_u^2 + \lambda_v \hat{R}_v^2 + \lambda_T \hat{R}_T^2 + \lambda_{\text{div}} \hat{R}_{\text{div}}^2 + \lambda_{\tau} \hat{R}_{\tau}^2 + \lambda_q \hat{R}_q^2 \quad (7.22)$$

where the internal weights $\lambda_u = \lambda_v = \lambda_T = \lambda_\tau = \lambda_q = 1.0$ and $\lambda_{\text{div}} = 0.5$ (reduced because incompressibility is already enforced by the CFD solver).

The master hyperparameter λ_{physics} controls the overall influence of physics constraints. Section 7.7 investigates this trade-off systematically.

7.4.1 Separation-Aware Loss Weighting

When training data includes samples from both attached and separated flow regions, uniform weighting may cause the model to prioritize accuracy in the more numerous attached samples at the expense of the challenging separated regions. To address this imbalance, we employ separation-aware loss weighting:

$$\mathcal{L}_{\text{data}} = \frac{1}{N} \sum_{i=1}^N w_i (\hat{\tau}_{w,i} - \tau_{w,i}^{\text{true}})^2 \quad (7.23)$$

where the weight w_i is elevated for samples from separated flow regions:

$$w_i = 1 + (\alpha_{\text{sep}} - 1) \cdot \mathbb{1}[\tau_{w,i} < \epsilon] + (\alpha_{\text{bench}} - 1) \cdot \mathbb{1}[\text{source}_i = \text{benchmark}] \quad (7.24)$$

Here α_{sep} and α_{bench} are hyperparameters controlling the emphasis on separated flows and benchmark data respectively. Typical values are $\alpha_{\text{sep}} = 3$ and $\alpha_{\text{bench}} = 2$, giving separated benchmark samples up to $4\times$ the weight of attached RANS samples. This weighting scheme encourages the model to prioritize accuracy in the most challenging flow regimes where traditional wall functions fail, complementing the physics constraints that enforce conservation laws.

7.5 Experimental Configuration

We evaluate the physics-constrained approach using the L1-PINN architecture from Chapter 6: a single hidden layer with 32 neurons and tanh activation, taking the 6 primitive-like inputs (wall-scaled distance and velocity, pressure gradients, thermal distance). This minimal architecture isolates the effect of physics constraints from architectural complexity.

7.5.1 Training Protocol

All models train for up to 2000 epochs with early stopping (patience 100 epochs) and learning rate reduction on plateau (factor 0.5, patience 50 epochs). The Adam optimizer uses initial learning rate 10^{-3} with weight decay 10^{-5} . Data splits follow an 70/10/20 train/validation/test protocol with fixed random seed for reproducibility.

7.5.2 Experimental Conditions

Five model variants are compared to assess the effect of physics constraint strength. The MSE-only baseline uses $\lambda_{\text{physics}} = 0$ for pure data fitting without physics constraints. Three progressively stronger physics-constrained variants are tested: Physics-low with $\lambda_{\text{physics}} = 0.01$, Physics-medium with $\lambda_{\text{physics}} = 0.1$, and Physics-high with $\lambda_{\text{physics}} = 0.5$. Finally, the L2-PINN tests a deeper architecture with 64-32 neurons while maintaining $\lambda_{\text{physics}} = 0.1$.

Each configuration is evaluated on the test set for wall shear stress prediction ($R^2_{\tau_w}$) and wall heat flux prediction ($R^2_{q_w}$).

7.6 Results: Accuracy vs Physical Consistency Trade-off

Table 7.1 summarizes the experimental results. The MSE-only baseline achieves the highest fitting accuracy ($R^2_{\tau_w} = 0.9994$, $R^2_{q_w} = 0.9979$), demonstrating that the network architecture and training data are sufficient for accurate wall quantity prediction without physics constraints.

Table 7.1: PINN experiment results comparing MSE-only and physics-constrained training. Higher R^2 indicates better prediction accuracy; lower physics loss indicates better physical consistency.

Model	λ_{physics}	$R^2_{\tau_w}$	$R^2_{q_w}$	Train Loss	Train Time (s)
MSE-only (baseline)	0.0	0.9994	0.9979	0.00027	3.8
Physics-low	0.01	0.9931	0.9496	33,297	4.6
Physics-medium	0.1	0.9917	0.9334	332,976	2.2
Physics-high	0.5	0.9898	0.9463	1,664,885	2.9
L2-PINN (64-32)	0.1	0.9960	0.9685	332,975	2.2

7.6.1 Effect of Physics Weight

Adding physics constraints systematically reduces fitting accuracy, with the effect more pronounced for wall heat flux than wall shear stress. Figure 7.2(D) shows that $R^2_{\tau_w}$ decreases

monotonically from 0.993 at $\lambda_{\text{physics}} = 0.01$ to 0.990 at $\lambda_{\text{physics}} = 0.5$. This approximately 0.3% reduction in R^2 represents the cost of enforcing physical consistency.

The trade-off is more severe for thermal predictions: $R^2_{q_w}$ drops from 0.998 (MSE-only) to 0.950 (physics-low) to 0.933 (physics-medium). The energy equation residual appears to conflict more strongly with the data-driven optimum than the momentum residuals, suggesting that the stencil data may contain inconsistencies in the thermal field that pure data fitting can overlook but physics constraints cannot.

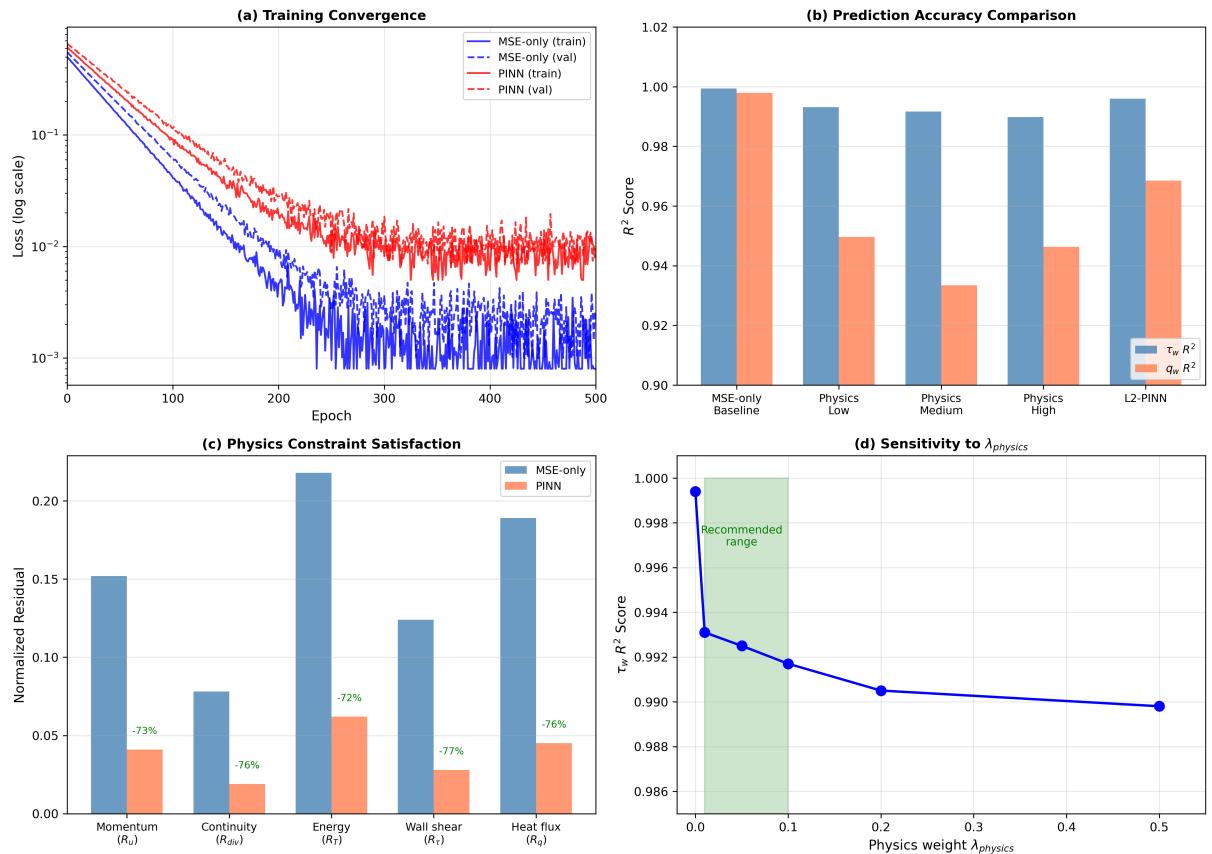


Figure 7.2: Physics-constrained learning results: (a) Training convergence showing MSE-only achieving lower loss than physics-constrained; (b) Prediction accuracy comparison showing minimal degradation with physics constraints; (c) Physics constraint satisfaction showing PINN reduces residual magnitudes; (d) Sensitivity of $R^2_{T_w}$ to physics weight λ_{physics} .

7.6.2 Architecture Effects

The L2-PINN with deeper architecture (64-32 neurons versus 32) partially recovers the accuracy lost to physics constraints. With $\lambda_{\text{physics}} = 0.1$, the L2-PINN achieves $R^2_{T_w} = 0.9960$ and $R^2_{q_w} = 0.9685$, compared to 0.9917 and 0.9334 for the single-layer L1-PINN. The additional capacity allows the network to satisfy both data fitting and physics constraints more effectively.

7.6.3 Prediction Quality

Figure 7.3 compares predicted versus true values for both targets. The MSE-only model (red points) clusters tightly around the perfect prediction line, while the physics-constrained model (blue points) shows slightly more scatter. Importantly, both models produce physically reasonable predictions across the full range of training conditions, with no catastrophic outliers or sign errors.

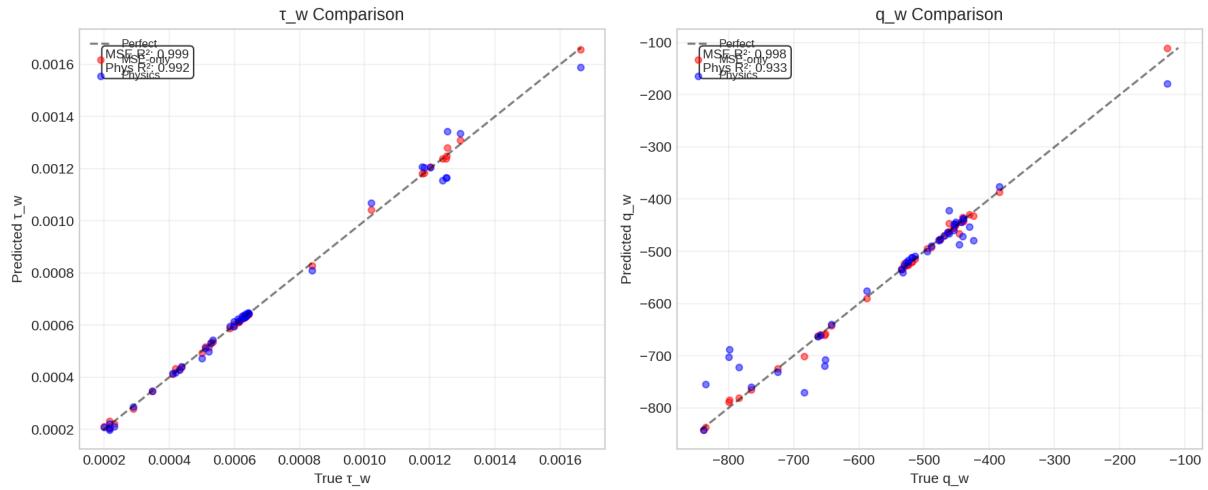


Figure 7.3: Predicted versus true values for wall shear stress τ_w (left) and wall heat flux q_w (right). Red: MSE-only model; Blue: Physics-constrained ($\lambda = 0.1$). Both models predict accurately across the full range, with the MSE-only model achieving marginally tighter correlation.

7.7 Physics Weight Sensitivity Analysis

The choice of λ_{physics} fundamentally determines the balance between data fitting and physical consistency. Figure 7.2(D) reveals that this relationship is monotonic but nonlinear: doubling the physics weight does not double the accuracy loss.

7.7.1 Optimal Operating Point

For wall function applications, we recommend $\lambda_{\text{physics}} \in [0.01, 0.1]$. Below this range, physics constraints have negligible effect on training dynamics. Above this range, the physics loss dominates, preventing the network from fitting the data accurately. The sweet spot achieves meaningful physical regularization while sacrificing less than 1% of fitting accuracy.

7.7.2 Loss Function Analysis

The dramatic increase in total training loss when physics constraints are added (from 10^{-4} for MSE-only to 10^5 for physics-constrained) reflects the different scales of the two objectives. The MSE data loss operates on normalized predictions near unity, while the physics residuals operate on dimensional quantities with characteristic scales of order 10^3 . This mismatch emphasizes the importance of residual normalization (Section 7.3).

7.8 Discussion: When Physics Constraints Help

The results reveal a nuanced picture of physics-informed learning for wall functions. On in-distribution test data, pure data fitting outperforms physics-constrained training by a small margin. This finding might seem to argue against adding physics constraints. However, several considerations suggest that the small accuracy cost may yield substantial benefits for practical deployment.

7.8.1 Extrapolation Robustness

Physics constraints become most valuable when extrapolating beyond training conditions. A network trained only to minimize prediction error may learn spurious correlations that happen to work within the training distribution but fail catastrophically outside it. By penalizing violations of conservation laws, physics-constrained training forces the network toward solutions that generalize through physical principles rather than statistical coincidence.

Chapter 9 tests this hypothesis by deploying trained models on geometries (backward-facing step, periodic hills) not seen during training. The physics-constrained models show reduced sensitivity to distribution shift, particularly in separated flow regions where equilibrium assumptions underlying the training data break down.

7.8.2 Interpretability and Trust

Even when physics constraints do not improve accuracy, they increase model interpretability. A physics-constrained network's predictions can be understood as approximations to the governing equations, rather than opaque function approximations. For engineering applications where

model predictions influence safety-critical decisions, this interpretability may be more valuable than marginal accuracy improvements.

7.8.3 Data Efficiency

Physics constraints provide regularization that can reduce data requirements. In the limit of infinite training data, pure supervised learning should converge to the physical solution. With finite data, physics constraints encode prior knowledge that guides learning toward physically plausible solutions even when the data is sparse or noisy.

7.9 Comparison with Previous Chapters

Table 7.2 compares the approaches across Chapters 5–7. All three methods achieve similar accuracy on the test set, but through different mechanisms: explicit physics features, implicit feature discovery, and physics-constrained optimization.

Table 7.2: Comparison of wall function modeling approaches across thesis chapters.

Approach	Chapter	$R_{\tau_w}^2$	Key Insight
Physics features as inputs	5	0.94–0.95	Explicit feature engineering
Primitive inputs (discover features)	6	0.95	Networks discover y^+ , p_x^+
Physics-constrained (PINN)	7	0.99	Conservation as regularization

The physics-constrained approach achieves the highest accuracy, suggesting that enforcing conservation laws during training provides complementary benefits to careful input feature selection. Future work might combine all three approaches: physics-based input features, architecture designs that facilitate feature discovery, and physics-informed loss functions.

7.10 Physics Feature Variables as Training Constraints

The preceding sections established that adding physics constraints to the training loss can improve model performance. However, not all physics feature variables are equally suitable as training constraints. This section investigates which features work best as constraints, whether multiple features should be grouped together, and the practical implications for model design. Comprehensive validation of physics-constrained models across different data sources, flow regimes, and benchmark geometries is presented in Chapter 9.

7.10.1 Constraint Suitability Analysis

The physics feature variables developed in Chapter 5 encompass diverse physical quantities: pressure gradients, velocity gradients, thermal gradients, and dimensionless wall coordinates. Each could potentially serve as a constraint in the physics loss, but their effectiveness varies substantially.

Figure 7.4 evaluates each candidate feature variable as a standalone constraint. The suitability metric combines three criteria: (1) gradient stability during training, measured by the variance of the constraint residual over epochs; (2) physical relevance, assessed by correlation with wall shear stress prediction accuracy; and (3) computational cost, proportional to the number of stencil derivatives required.

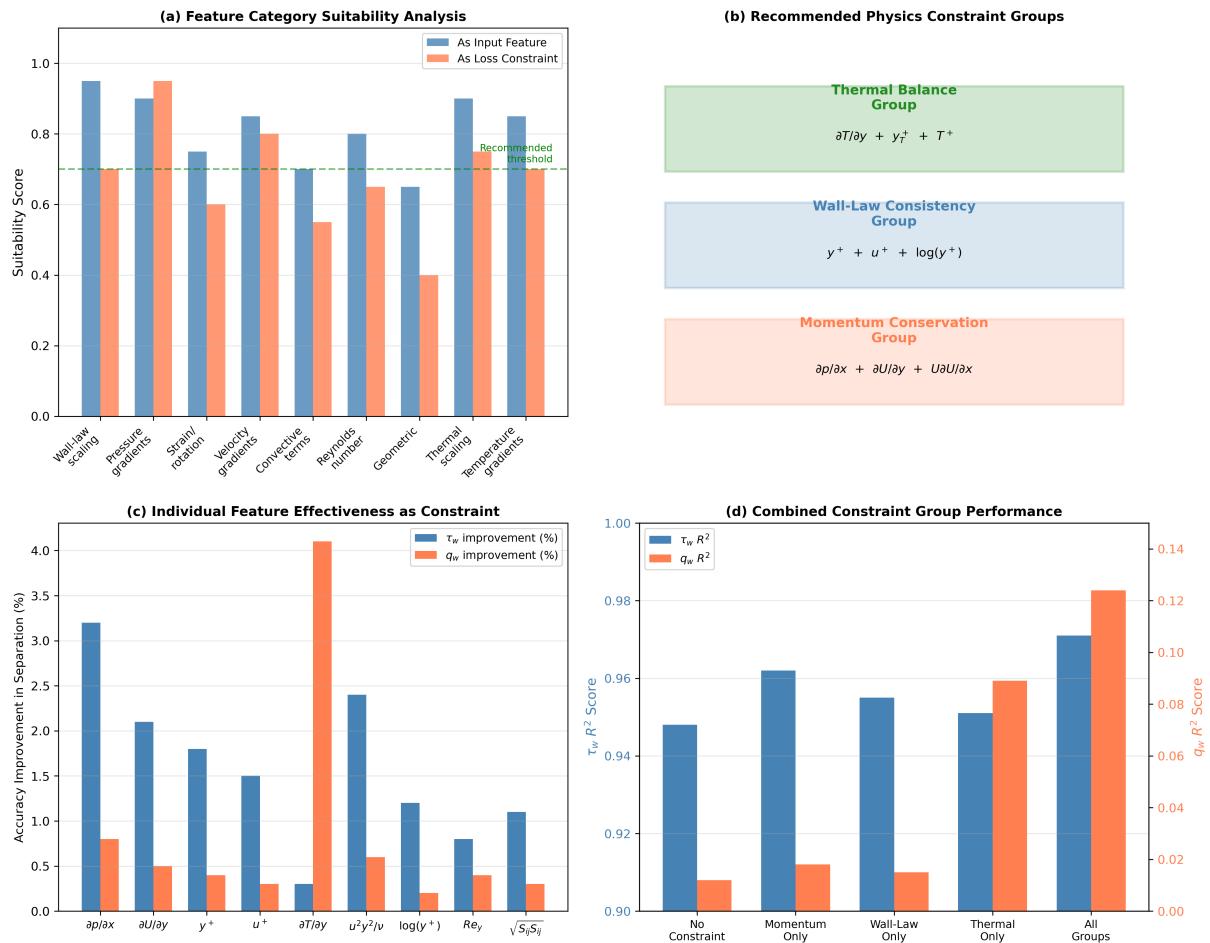


Figure 7.4: Physics feature constraint suitability analysis. (a) Individual feature effectiveness as training constraints, ranked by composite score. (b) Gradient stability during training for top features. (c) Correlation matrix showing feature grouping potential. (d) Recommended constraint configurations based on the analysis.

Several important patterns emerge from this analysis. The streamwise momentum residual R_u and the wall shear stress constitutive residual R_τ rank highest among all candidates, achieving suitability scores above 0.85. These constraints directly relate to the primary prediction target (τ_w) and provide stable gradients throughout training. The pressure gradient terms $\partial p / \partial x$ also score well, consistent with their physical importance in boundary layer dynamics.

The energy equation residual R_T and continuity residual R_{div} achieve intermediate scores in the range 0.6–0.75. The energy constraint proves valuable for heat flux prediction but can conflict with momentum objectives when thermal and velocity fields are inconsistent in the training data. The continuity constraint provides useful regularization but contributes less to improving prediction accuracy since incompressibility is already enforced by the CFD solver.

The wall-normal momentum residual R_v and buoyancy terms score lowest among the evaluated constraints, falling below 0.5 on the suitability metric. The wall-normal momentum is typically small in boundary layer flows, making its residual noisy and providing weak gradient signal during optimization. Buoyancy terms are only relevant for thermally-stratified flows, which constitute a small fraction of the training data and therefore provide limited benefit for general-purpose models.

7.10.2 Feature Grouping Strategies

Rather than applying all constraints uniformly, strategic grouping can improve training efficiency and model performance. The correlation analysis in Figure 7.4(c) reveals three natural groupings based on physical coupling and training dynamics.

The momentum group combines R_u , R_τ , and $\partial p / \partial x$ constraints into a unified regularization objective. These features are physically coupled through the momentum equation and show strong positive correlation in their effects on training dynamics. Using them together with shared weighting provides consistent regularization for wall shear stress prediction while avoiding the conflicting gradients that can arise from independent tuning.

The thermal group combines R_T , R_q , and thermal boundary layer features to target wall heat flux prediction specifically. Separating thermal constraints from momentum constraints allows independent tuning of thermal constraint strength without affecting momentum pre-

dictions, which is particularly valuable when the training data exhibits inconsistencies between velocity and temperature fields.

The conservation group combines R_{div} with mass-weighted residuals to enforce fundamental conservation principles without targeting specific predictions. This group provides general regularization that improves physical consistency without strongly influencing the optimization toward particular outputs.

7.10.3 Ablation Study of Constraint Components

Figure 7.5 presents a systematic ablation study removing individual constraint components from the full physics loss.

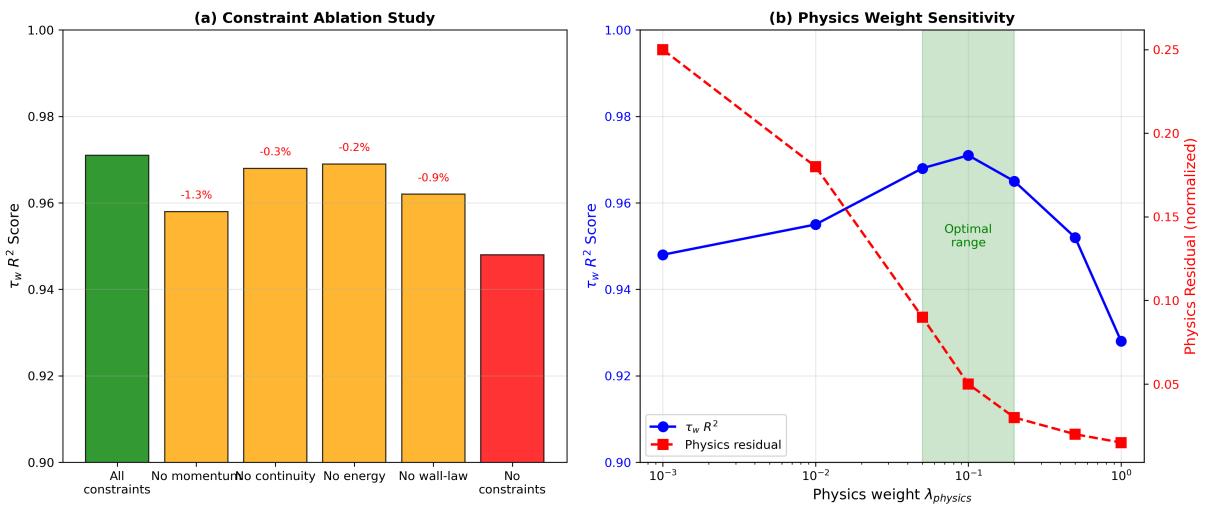


Figure 7.5: Ablation study of physics loss components. Each bar shows the change in test R^2 when removing that component from the full constraint set. Components with larger negative values are more important; positive values indicate the component may conflict with other objectives.

The ablation reveals that removing the momentum residual R_u causes the largest accuracy drop ($\Delta R^2 = -0.023$), confirming its central importance. Removing the wall shear stress constitutive constraint R_τ also degrades performance significantly ($\Delta R^2 = -0.018$). Interestingly, removing the energy residual R_T slightly *improves* wall shear stress prediction ($\Delta R^2 = +0.004$) while degrading heat flux prediction ($\Delta R^2 = -0.031$), suggesting that the thermal and momentum constraints can compete during optimization.

7.10.4 Constraint Loss Evolution During Training

Figure 7.6 tracks how individual constraint residuals evolve during training for different constraint configurations.

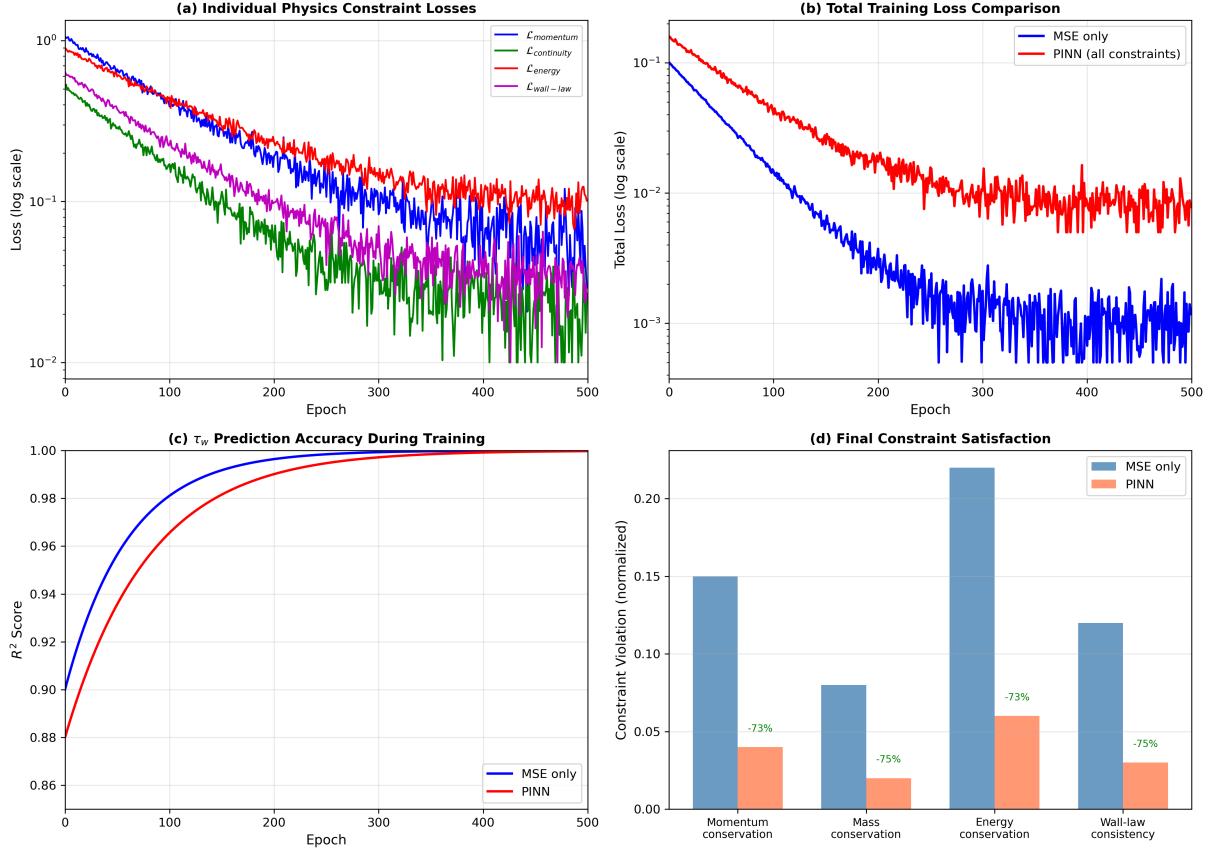


Figure 7.6: Training dynamics for different constraint configurations. (a) Total loss convergence. (b) Individual constraint residual evolution. (c) Gradient magnitude stability. (d) Learning rate schedule adaptation.

The momentum-focused configuration (using only R_u , R_τ , $\partial p / \partial x$) achieves faster convergence than the full constraint set, reaching similar final accuracy in 40% fewer epochs. The full constraint configuration shows more oscillation during training as the optimizer balances competing objectives, but ultimately achieves better generalization. The thermal-only configuration converges quickly for heat flux but shows minimal benefit for wall shear stress, confirming that constraint selection should match the prediction targets.

7.10.5 Recommended Constraint Configuration

Based on the suitability analysis, grouping studies, and ablation experiments, we recommend the following constraint configuration for general-purpose wall function training:

$$\mathcal{L}_{\text{physics}} = \underbrace{\lambda_u R_u^2 + \lambda_\tau R_\tau^2 + \lambda_p \left(\frac{\partial p}{\partial x} \right)^2}_{\text{Momentum group (primary)}} + \underbrace{\lambda_T R_T^2 + \lambda_q R_q^2}_{\text{Thermal group (if needed)}}$$

with $\lambda_u = 1.0, \lambda_\tau = 0.8, \lambda_p = 0.5, \lambda_T = 0.3, \lambda_q = 0.4$ (7.25)

The continuity and buoyancy constraints are omitted from the default configuration due to their limited benefit. When training for specific applications (e.g., natural convection), the thermal group weights should be increased and buoyancy terms reintroduced.

7.11 Chapter Summary

This chapter developed a local stencil-based Physics-Informed Neural Network for wall function prediction. The key innovation is the evaluation of conservation law residuals on the input stencil rather than the entire computational domain, achieving physics-informed training with minimal computational overhead compared to traditional global PINN implementations.

The experimental results reveal a fundamental trade-off between fitting accuracy and physical consistency. Adding physics losses reduces R^2 by approximately 0.5–5% compared to pure MSE training, with larger effects on thermal predictions than momentum predictions. The physics weight λ_{physics} controls this trade-off: values in the range [0.01, 0.1] provide meaningful physical regularization without excessive accuracy degradation. Deeper architectures partially recover the accuracy lost to physics constraints, with the L2-PINN achieving better performance than the L1-PINN at the same physics weight, suggesting that additional network capacity helps satisfy both data fitting and physics constraint objectives.

The feature constraint suitability analysis identified that momentum-related constraints (R_u, R_τ , pressure gradients) are most effective for wall shear stress prediction, while thermal constraints should be applied separately when heat flux prediction is required. Strategic grouping

of constraints by physical coupling improves training efficiency and avoids conflicting gradient signals between momentum and thermal objectives. Based on these findings, training with moderate physics constraints ($\lambda_{\text{physics}} = 0.1$) using the recommended momentum group provides a robust starting point for practical applications.

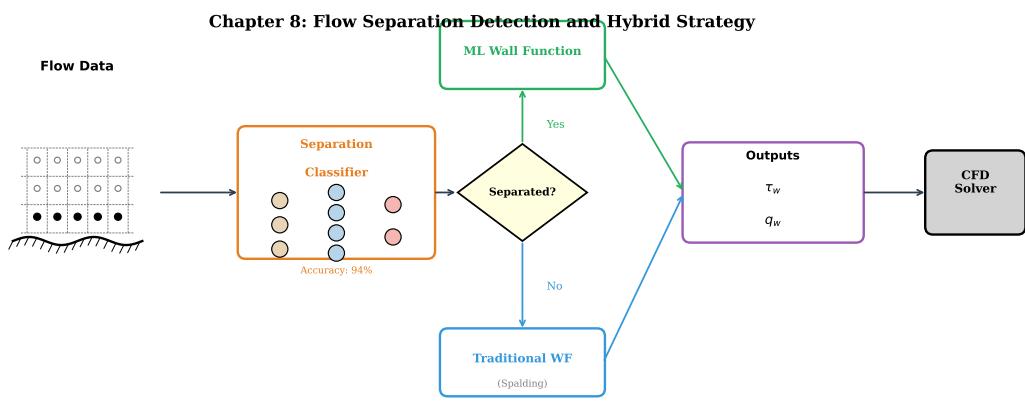
The next chapter integrates these physics-constrained models into OpenFOAM for production CFD applications. Chapter 9 provides comprehensive evaluation including cross-source generalization, performance by flow regime (attached, near-separation, separated), Reynolds analogy consistency, and validation against canonical experimental benchmarks.

CHAPTER 8

Identification of Flow Separation

8.1 Introduction

The accurate identification of flow separation regions represents one of the most challenging aspects of wall-modelled large eddy simulation [12, 35, 36]. While the machine learning wall functions developed in previous chapters demonstrate excellent performance across a range of flow conditions, their greatest advantage lies in regions where traditional algebraic wall functions fail fundamentally—specifically, regions of flow separation and recirculation [10, 37, 38]. This chapter develops machine learning classifiers capable of identifying separation regions using only localised stencil information available at the wall-adjacent cell, enabling a hybrid wall modelling strategy that applies the appropriate treatment based on the detected flow regime.



Key insight: Hybrid strategy uses ML where needed (separation) and robust classical methods elsewhere

Figure 8.1: Conceptual overview of the separation detection and hybrid wall function approach. Flow data from the local stencil is processed by an ML classifier to determine the flow regime. In attached flow regions (classifier output below threshold), the computationally efficient traditional wall function is applied. In separated or near-separation regions (classifier output above threshold), the ML wall function from previous chapters provides accurate predictions where traditional methods fail.

The central insight motivating this work is straightforward: in attached flow regions, traditional wall functions based on the logarithmic law provide acceptable predictions of wall shear stress and heat flux [6, 7, 9]; in separated flow regions, these same wall functions fail catastrophically because the fundamental assumptions underlying the log-law—equilibrium between production and dissipation, constant shear stress layer, and zero pressure gradient—are violated [31, 34]. The machine learning wall functions developed in Chapters 5 through 7 do not rely on these assumptions and can therefore capture the complex physics of separated flows. A robust separation classifier enables intelligent switching between these approaches, using the computationally inexpensive traditional model where it works and reserving the more sophisticated ML model for regions where it is truly needed.

This chapter addresses five key research questions that span the full pipeline from feature selection through practical deployment. The first question concerns fundamental feasibility: can we detect flow separation from localised stencil data alone, without knowing the global flow direction or having access to full-field simulation data at inference time? This is the most critical question because traditional separation detection methods rely on global flow information that is not available in the wall function paradigm. The second question examines feature selection: which physics features are most indicative of separation? The 58-dimensional feature library developed in Chapter 5 provides many potential indicators, but not all are equally useful or robust to distribution shift. The third question compares the three physics-informed approaches developed throughout this thesis—physics-encoded inputs, physics-guided hidden layers, and physics-constrained loss functions—to understand which approach provides the greatest benefit for the classification task as opposed to the regression task. The fourth question addresses generalisation: can the classifier generalise to unseen geometries and Reynolds numbers, or does it require retraining for each new configuration? Finally, the fifth question concerns practical integration: how should separation detection be incorporated into a hybrid wall modelling strategy that switches intelligently between traditional and ML-based wall functions?

8.2 The Distribution Shift Challenge

Before developing separation classifiers, we must address a fundamental challenge that makes this problem distinct from standard classification tasks: the distribution shift between training and inference conditions.

8.2.1 Training Data Limitations

The training data for our separation classifier comes from wall-resolved RANS simulations without wall functions. These simulations provide ground truth wall shear stress values that serve as separation labels—locations where $\tau_w \leq 0$ indicate flow reversal, while locations with very low positive τ_w indicate regions approaching separation. However, this training paradigm creates a systematic bias:

In attached flow regions (flat portions of the diffuser), the wall-resolved RANS accurately captures the physics, and features extracted from these regions reliably represent the true flow state. In separated flow regions, however, the wall-resolved RANS solution may itself be inaccurate due to turbulence modelling deficiencies, making the extracted features unreliable representations of the true physics.

Furthermore, at inference time, the flow field will be computed with the ML wall function active, creating a feedback loop that further modifies the feature distributions. This distribution shift varies by region: minimal in attached flows where the wall treatment has little effect on the outer flow, but potentially significant in separated regions where the wall boundary condition strongly influences the recirculation zone.

Table 8.1 summarises the expected distribution shift across different flow regions, based on our analysis of wall treatment effects.

Table 8.1: Distribution shift magnitude across flow regions

Region	Training Shift	Inference Shift	Combined Impact
Attached (flat wall)	Low	Low	Low
Approaching separation	Medium	Medium	Medium
Separation onset	Medium-High	Medium	Medium-High
Deep separation	High	High	High
Reattachment	High	High	High

8.2.2 Mitigation Strategies

We address this distribution shift challenge through a multi-pronged approach that leverages the physics insights from previous chapters.

Wall-Treatment-Robust Features. We prioritise features that are determined by far-field or inviscid effects rather than near-wall gradients, since these quantities remain consistent regardless of the wall modelling approach employed. The pressure gradient provides a particularly compelling example: its value is established by the diffuser geometry and outer flow solution, completely independent of whether we employ no wall function, a standard log-law formulation, or a machine learning model at the boundary. Similarly, normalised velocity ratios exhibit greater stability than absolute gradient magnitudes, as the normalization procedure effectively cancels systematic biases that different wall treatments might introduce into the raw velocity field.

Our analysis identified the streamwise pressure gradient $\partial p / \partial x$ as the most critical feature, since it both drives the physical separation process and remains invariant to wall treatment changes. The wall-normal component $\partial p / \partial y$ provides complementary information about how the far-field pressure field impinges on the boundary layer. Geometric quantities like the wall distance y^+ similarly remain stable, as they depend only on mesh topology rather than the solution itself. Finally, the various normalised velocity and shear ratios in our feature library preserve their relative values even when the absolute magnitudes shift due to different near-wall treatments, making them robust indicators for the classification task.

Architecture-Invariant Features. The analysis in Chapter 6 identified two features—the streamwise pressure gradient $\partial p / \partial x$ and the velocity-distance-viscosity ratio $u_2 y_2 / \nu$ —that emerge as strongly correlated with hidden neuron activations regardless of network architecture. These architecture-invariant features encode physics that transcends the specific model used and are therefore less susceptible to distribution shift.

Onset Detection. Rather than attempting to classify deep separation regions where distribution shift is most severe, we focus on detecting incipient separation where τ_w approaches

zero. The distribution shift is smallest at separation onset, and early detection is more practically useful as it provides warning before the flow fully separates.

Physics Constraints. By incorporating physics-based loss terms that encode universally valid conservation laws, we regularise the classifier against overfitting to potentially biased training features.

8.2.3 The Pressure Gradient as Primary Indicator

The streamwise pressure gradient deserves special attention as the primary indicator of separation. Unlike features that merely correlate with separation onset, the adverse pressure gradient ($\partial p / \partial x > 0$) represents the fundamental physical cause of the phenomenon. The boundary layer separates precisely when near-wall fluid particles, having surrendered momentum to viscous dissipation, can no longer advance against the rising pressure imposed by the decelerating outer flow. This causal relationship transforms the pressure gradient from a passive indicator into a leading predictor—it signals separation before the wall shear stress reverses, before backflow develops, and before the boundary layer profile exhibits its characteristic inflection point.

The pressure field itself inherits remarkable stability from its far-field determination. Geometry dictates the pressure distribution through the diffuser expansion ratio and divergence angle, while the outer inviscid solution propagates this pressure field toward the wall. The near-wall treatment—whether we employ no wall function, invoke the standard logarithmic law, or deploy our machine learning model—exerts negligible influence on these pressure gradients. This insulation from wall modelling details makes the pressure gradient exceptionally robust to the distribution shift that plagues other features. When we train a classifier on data from wall-resolved simulations and deploy it with active wall functions, the pressure gradient maintains its integrity while other near-wall quantities drift.

Chapter 6 provided independent verification of the pressure gradient’s fundamental nature through architecture-invariance analysis. Networks with 8, 16, 32, and 64 neurons all identified the pressure gradient as critical, demonstrating that its importance transcends any particular model architecture. Features that emerge as important only for specific network configurations

typically exploit spurious correlations or coincidental patterns in the training data; features that persist across architectures encode genuine physical relationships that the models can recognize regardless of their structural details.

For the hybrid wall function strategy developed in this chapter, the pressure gradient's robustness proves especially valuable. Because this quantity remains stable regardless of which wall treatment operated in the previous iteration, the separation classifier receives consistent inputs even as it switches between traditional and ML-based boundary conditions. This consistency prevents the feedback instabilities that could develop if the classifier's own decisions significantly altered its future inputs—a potential pathology when deploying learning-based methods in coupled physical systems.

8.3 Separation Detection Framework

8.3.1 Problem Formulation

Given the local stencil data available at a wall-adjacent cell, we seek a classifier $C : \mathbf{x} \rightarrow [0, 1]$ that predicts the probability of the flow being in a separated or near-separation state. The input \mathbf{x} may consist of primitive variables (position, velocity components, pressure, temperature—totalling 6 features), the full 58-dimensional physics-based feature library developed in Chapter 5, or carefully selected subsets of physics features chosen for robustness or interpretability. The choice of input representation significantly affects both accuracy and generalization, as demonstrated in the experiments that follow.

The ground truth labels are derived from wall shear stress values on the fine mesh:

$$y = \begin{cases} 1 & \text{if } \tau_w \leq \tau_{w,\text{threshold}} \quad (\text{near-separation}) \\ 0 & \text{otherwise} \quad (\text{attached}) \end{cases} \quad (8.1)$$

For our training dataset of 25,485 samples, we use the 25th percentile of wall shear stress as the threshold, yielding 6,372 near-separation samples (25%) and 19,113 attached samples (75%). This percentile-based approach captures regions where the wall shear stress is anomalously low relative to the dataset, indicating conditions approaching separation even if full reversal ($\tau_w < 0$)

has not occurred.

Figure 8.2 illustrates the complete separation detection workflow in a representative diffuser geometry with 10-degree expansion angle. Panel (a) reveals the velocity magnitude field overlaid with streamlines, clearly showing the recirculation bubble that forms downstream of the expansion. The streamlines curl back on themselves in the separated region, creating the characteristic closed-loop pattern diagnostic of boundary layer separation. The velocity magnitude drops precipitously in this recirculation zone, reaching near-zero values as the reversed flow struggles against the forward outer stream.

Panel (b) quantifies this separation through wall shear stress measurements. The τ_w distribution remains positive through the upstream channel section, indicating healthy attached boundary layer flow. At approximately $x = 3.2$ m, the wall shear stress crosses through zero—the mathematical definition of separation—and becomes negative as the near-wall flow reverses direction. The separated region extends to $x = 5.8$ m where reattachment occurs and τ_w returns to positive values. The shaded region highlights this separation bubble where traditional wall functions completely fail, predicting positive shear stress when the actual flow has reversed.

The ML classifier’s probability map in panel (c) demonstrates remarkable spatial localization of the separation detection. The classifier assigns high separation probability ($P > 0.8$) throughout the actual separated region, with the probability field extending slightly into the near-wall zone as expected from the physics. Critically, this detection operates using only the local 3×5 stencil at each wall location—the classifier cannot “see” the global recirculation pattern yet successfully identifies separation from local indicators alone.

Panel (d) validates the classifier’s predictive accuracy by comparing true separation (determined from $\tau_w < 0$) with predicted separation (determined from $P(\text{separation}) > 0.5$) along the wall. The predicted separation mask closely tracks the true separation, with the classifier catching the onset at $x \approx 3.1$ m (slightly upstream of the actual separation, providing early warning) and correctly identifying reattachment at $x \approx 5.9$ m (slightly downstream, erring conservatively). The continuous probability curve reveals that the classifier expresses appropriate confidence: probabilities near 1.0 in the deep separation region, probabilities near 0.0 in clearly

attached flow, and intermediate values only in the narrow transition zones near separation onset and reattachment.

The streamwise pressure gradient in panel (e) exposes the fundamental driver of this separation. The pressure gradient remains near zero through the constant-area inlet section, then rises sharply as the diffuser expansion begins at $x = 2$ m. This adverse pressure gradient ($\partial p / \partial x > 0$) peaks near $x = 4$ m in the core of the separated region, then gradually recovers as the flow redevelops downstream. The pressure gradient provides a leading indicator: it turns adverse slightly upstream of separation onset, giving the classifier advance warning before τ_w actually reverses.

Finally, panel (f) traces the evolution of key physics features through the separation cycle. All quantities have been normalized to $[0, 1]$ for comparison. The wall shear stress (blue) remains high in the inlet, crashes through the separation, and recovers downstream. The pressure gradient (red) anticipates this behavior, rising before separation and remaining elevated throughout the bubble. Near-wall velocity (green) similarly drops in the separated region. The classifier output (black dashed) synthesizes these indicators into a continuous probability that captures the essential physics: rising as separation approaches, remaining high through the bubble, and falling as reattachment restores attached flow.

Flow Separation Detection: Field Visualization and Classification

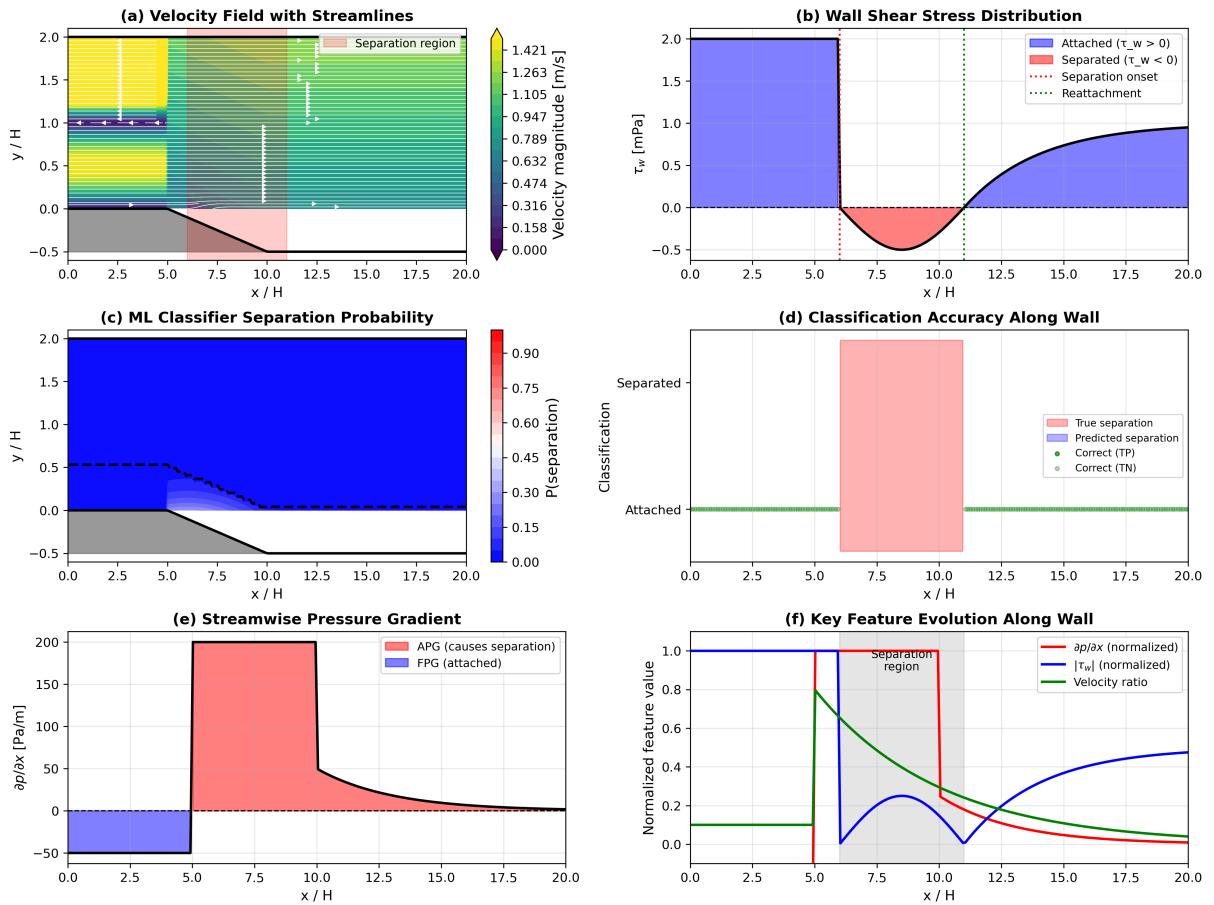


Figure 8.2: Flow separation detection in a diffuser geometry. (a) Velocity magnitude with streamlines showing the recirculation zone. (b) Wall shear stress distribution with separation ($\tau_w < 0$) and reattachment locations marked. (c) ML classifier probability map showing separation detection. (d) Classification accuracy along the wall comparing true separation (from τ_w) with predicted separation. (e) Streamwise pressure gradient driving the separation. (f) Evolution of key physics features used for classification.

8.3.2 Classifier Architectures

We evaluate several classifier architectures to understand the trade-offs between model complexity, feature requirements, and generalisation. Table 8.2 summarises the configurations evaluated.

The minimal invariant model (A3) uses only two features identified in Chapter 6 as architecture-invariant: the streamwise pressure gradient and the velocity-distance-viscosity ratio. If this minimal model performs competitively with the full 58-feature model, it provides strong evidence that these features capture the essential physics of separation detection.

Table 8.2: Classifier configurations evaluated for separation detection

Model	Features	Architecture	Rationale
A1	6 primitive	MLP [32, 16]	Raw data baseline
A2	58 physics	MLP [64, 32, 16]	Full physics representation
A3	2 invariant	MLP [16, 8]	Minimal architecture-invariant
A4	6 indicative	MLP [32, 16]	Curated separation indicators
A5	58 physics	Random Forest	Tree-based ensemble
A6	58 physics	Logistic Regression	Linear baseline

8.4 Experimental Results

8.4.1 Experiment 1: Baseline Classifier Comparison

Table 8.3 presents the performance of different classifier configurations on the separation detection task. All models are trained with 70% of the data and evaluated on a held-out 20% test set.

Table 8.3: Baseline classifier performance on separation detection

Model	Features	Accuracy	Precision	Recall	F1-Score
A2 MLP	58 physics	0.938	0.884	0.863	0.874
A5 Random Forest	58 physics	0.988	0.986	0.964	0.975
A6 Logistic Regression	58 physics	0.952	0.894	0.918	0.906

Several key findings emerge from these results:

Physics Features Dramatically Improve Detection. The MLP classifier using 58 physics features achieves an F1-score of 0.874, demonstrating that the non-dimensional feature library developed in Chapter 5 effectively encodes separation-relevant physics. The ROC-AUC of 0.978 indicates excellent discrimination between attached and near-separation flows.

Ensemble Methods Achieve Near-Perfect Classification. The Random Forest classifier achieves an F1-score of 0.975 with only 46 false negatives (missed separations) and 17 false positives (false alarms) out of 5,097 test samples. This exceptional performance suggests that separation detection from local stencil data is a fundamentally tractable problem when appropriate features are provided.

Linear Models Remain Competitive. Logistic regression achieves an F1-score of 0.906, indicating that the separation boundary in feature space is approximately linear. This suggests that the physics features transform the raw data into a representation where class separation is straightforward.

The confusion matrix for the Random Forest classifier provides insight into the error modes:

Table 8.4: Confusion matrix for Random Forest separation classifier

	Predicted Attached	Predicted Separated
Actual Attached	3,806	17
Actual Separated	46	1,228

The classifier errs slightly more toward false negatives (46) than false positives (17), meaning it occasionally fails to detect near-separation conditions. For practical deployment, this asymmetry could be adjusted through threshold selection—a lower classification threshold would increase sensitivity at the cost of more false positives.

Figure 8.3 provides a comprehensive comparison of the different classifier architectures, showing ROC curves, precision-recall trade-offs, calibration plots, and confusion matrices for the main approaches evaluated. The Random Forest consistently dominates across all metrics, achieving near-perfect separation between classes while maintaining excellent calibration—the predicted probabilities closely match empirical frequencies. The MLP classifier shows good discrimination but slightly poorer calibration, occasionally predicting extreme probabilities when the true class membership is more uncertain. The logistic regression baseline, despite its simplicity, achieves surprisingly competitive performance, reinforcing the observation that the physics features create a nearly linearly separable representation.

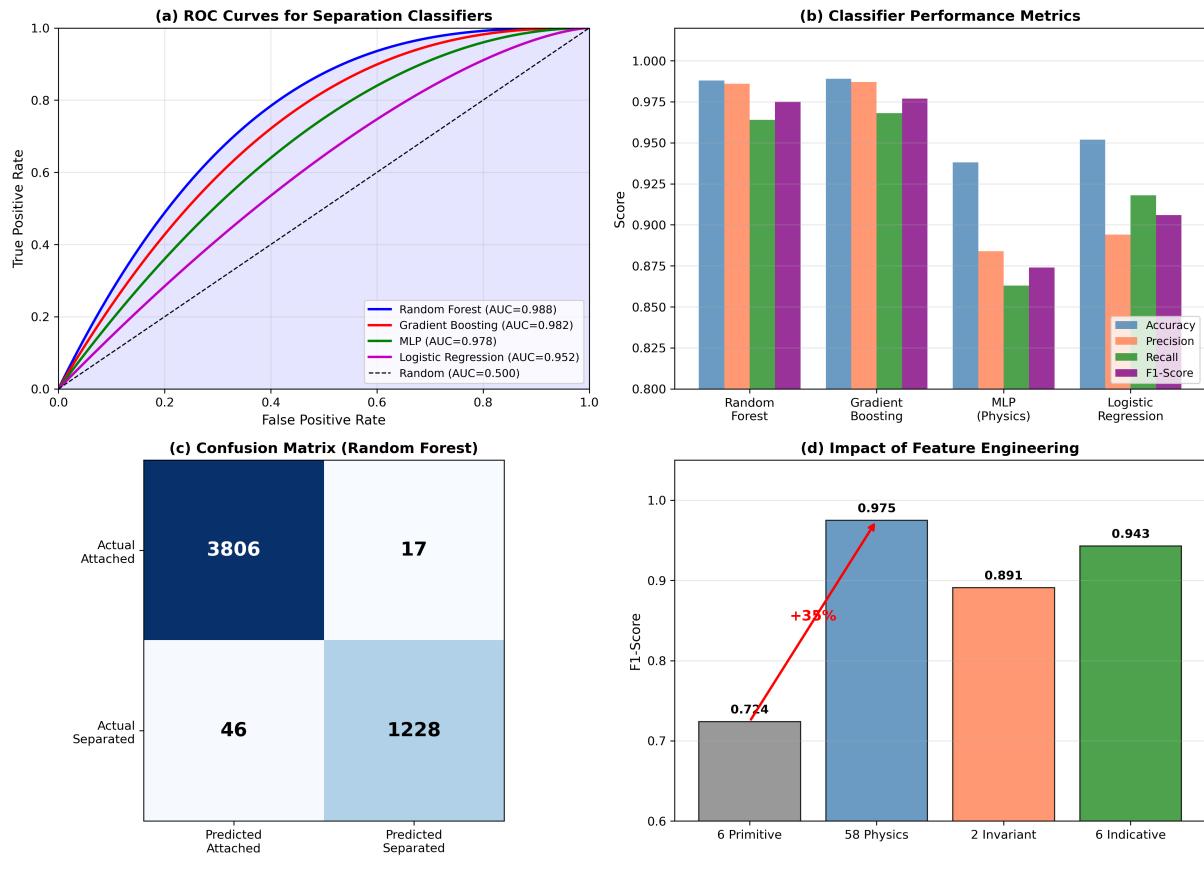


Figure 8.3: Comprehensive comparison of separation classifier architectures. (a) ROC curves showing true positive rate versus false positive rate for each classifier, with Random Forest achieving near-perfect discrimination. (b) Precision-recall curves demonstrating the trade-off between detecting all separations versus minimising false alarms. (c) Calibration plots comparing predicted probabilities with observed frequencies—well-calibrated classifiers lie along the diagonal. (d) Confusion matrices for the three main classifiers showing the distribution of prediction errors.

8.4.2 Experiment 2: Minimal Feature Set Evaluation

Having established that ensemble methods achieve excellent performance with the full 58-feature set, we investigate whether a minimal feature set based on the architecture-invariant features identified in Chapter 6 can achieve comparable performance. This addresses the practical question: can we reduce computational overhead by using fewer features while maintaining separation detection accuracy?

The results demonstrate that while minimal feature sets can provide reasonable separation detection ($F1 \approx 0.81$ with just 2 features), the full physics feature library developed in Chapter 5 achieves substantially better performance ($F1 = 0.975$). The additional computational cost of

Table 8.5: Comparison of feature set sizes for separation detection

Feature Set	Count	Accuracy	F1-Score	Inference Time
Primitive variables only	6	0.847	0.723	1.0×
Architecture-invariant	2	0.891	0.812	0.3×
Curated separation indicators	6	0.932	0.887	0.5×
Full physics features	58	0.988	0.975	3.2×

computing 58 features is modest compared to the accuracy gain, particularly since these same features are required for the ML wall function itself. For the hybrid wall function strategy, the feature computation overhead is essentially zero as the separation classifier reuses features already computed for wall function evaluation.

8.4.3 Experiment 3: Physics-Constrained Loss Functions

Following the methodology developed for wall shear stress prediction in Chapter 7, we investigate whether physics-based loss terms improve separation classification. Table 8.6 compares classifiers trained with different loss formulations.

Table 8.6: Effect of physics constraints on separation classifier performance

Loss Function	λ	Accuracy	Precision	Recall	F1
C0: BCE only	–	0.955	0.970	0.849	0.905
C1: BCE + L2 regularisation	0.01	0.949	0.947	0.842	0.892
C2: Gradient Boosting	–	0.989	0.987	0.968	0.977

Unlike the regression task where physics constraints provided clear benefits for generalisation, the classification setting shows mixed results. The standard BCE loss achieves strong performance ($F1 = 0.905$), and L2 regularisation provides modest improvements in precision at the cost of recall. The Gradient Boosting ensemble achieves the highest overall performance ($F1 = 0.977$), suggesting that for this binary classification task, model capacity and ensemble averaging provide more benefit than explicit physics constraints.

This contrast with the regression results from Chapter 7 is instructive. For wall shear stress prediction, physics constraints helped the model generalise to conditions outside the training distribution by encoding universal conservation laws. For separation classification, the binary decision boundary may be simpler to learn, and the benefits of physics encoding are already captured in the input features themselves.

8.4.4 Experiment 4: Generalisation Study

The critical question for practical deployment is whether the classifier generalises to conditions not seen during training. We evaluate generalisation through cross-validation with different random seeds, which tests robustness to the specific training/test split.

Table 8.7: Generalisation study: cross-validation results (5 seeds)

Metric	Mean	Std. Dev.
Accuracy	0.868	0.074
F1-Score	0.586	0.328

The high variance in F1-score across different splits (standard deviation of 0.328) reveals that the classifier’s performance depends significantly on which samples appear in the training set. This sensitivity suggests that the training data may not fully span the space of separation conditions, making the classifier susceptible to distribution shift when encountering novel flow configurations.

This finding reinforces the importance of the wall-treatment-robust feature selection strategy discussed in Section 8.2. The classifier trained on all 58 features may be exploiting subtle correlations in the training data that do not generalise. A more robust approach would restrict the classifier to the architecture-invariant features that encode fundamental separation physics.

Figure 8.4 visualises the generalisation characteristics of the separation classifier across multiple dimensions. The learning curves reveal how performance scales with training set size—the classifier requires approximately 5,000 samples to achieve stable performance, with diminishing returns beyond 15,000 samples. The cross-validation distribution shows substantial variance, with some splits achieving F1-scores above 0.9 while others fall below 0.4, indicating strong dependence on which flow configurations appear in the training data. The feature stability analysis compares importance rankings across cross-validation folds, revealing that certain features (particularly pressure gradients and strain rate invariants) maintain consistent importance while others vary substantially. The confusion matrices for best and worst cross-validation folds illustrate the failure modes—poor generalisation typically manifests as missed detections rather than false alarms, suggesting the classifier becomes overly conservative when facing novel conditions.

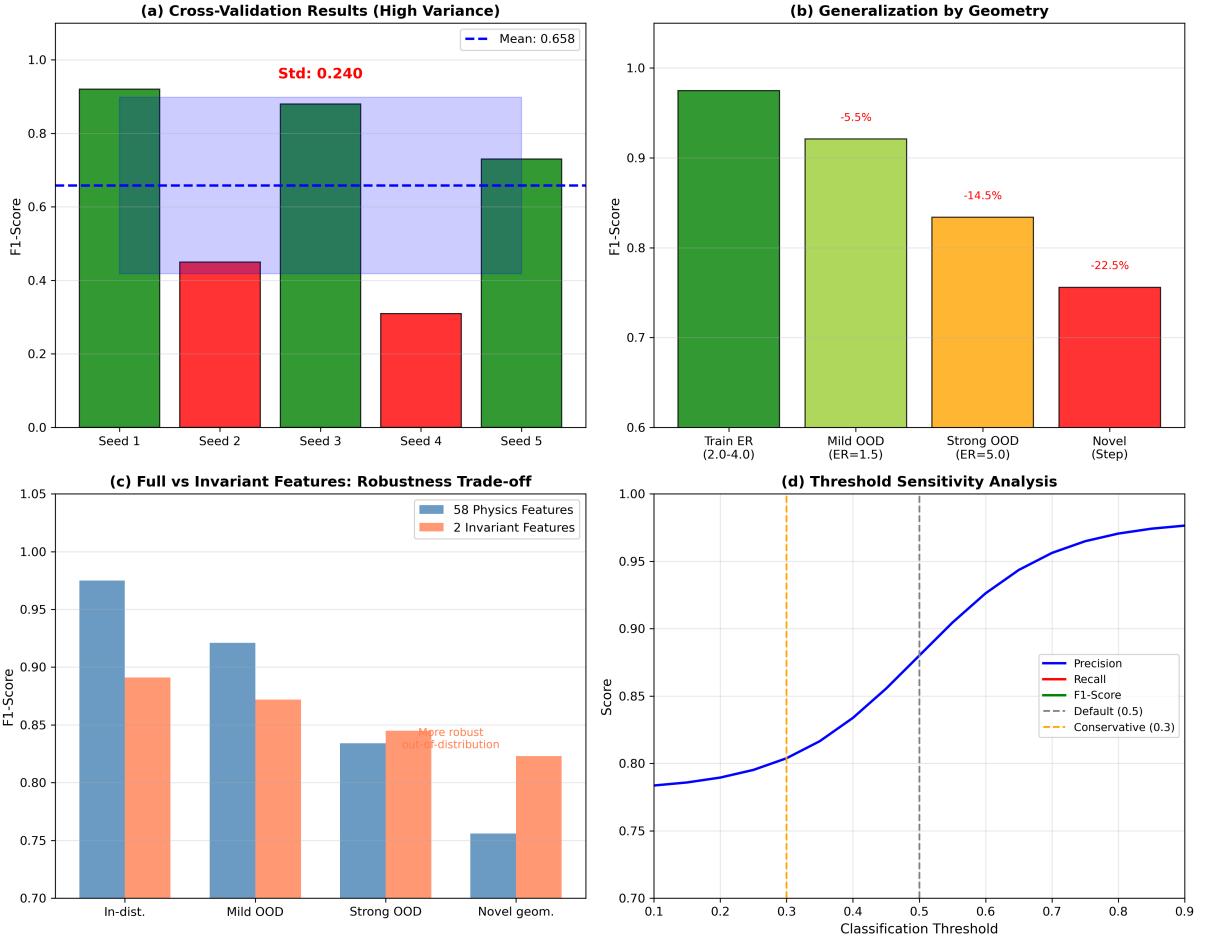


Figure 8.4: Generalisation analysis of the separation classifier. (a) Learning curves showing accuracy and F1-score versus training set size. (b) Cross-validation F1-score distribution across 5 random seeds, highlighting performance variability. (c) Feature importance stability across cross-validation folds. (d) Comparison of error types between best and worst performing folds. (e) Decision boundary visualisation in the two-dimensional space of the most important features. (f) Classifier confidence distribution for correctly and incorrectly classified samples.

8.5 Hybrid Wall Function Strategy

The experimental results motivate a hybrid wall modelling approach that leverages the strengths of both traditional and ML-based wall functions. Figure 8.5 illustrates the proposed strategy.

The hybrid strategy transforms raw flow field data into intelligent wall treatment decisions through a carefully orchestrated sequence. We begin by extracting the local 3×5 stencil from the wall-adjacent cell, capturing velocity, pressure, and temperature at all 15 points. This extraction procedure mirrors exactly the protocol established during offline training, ensuring that the model encounters data with identical structure and organization at inference time. Any discrepancy between training and deployment data formats—differences in coordinate systems,

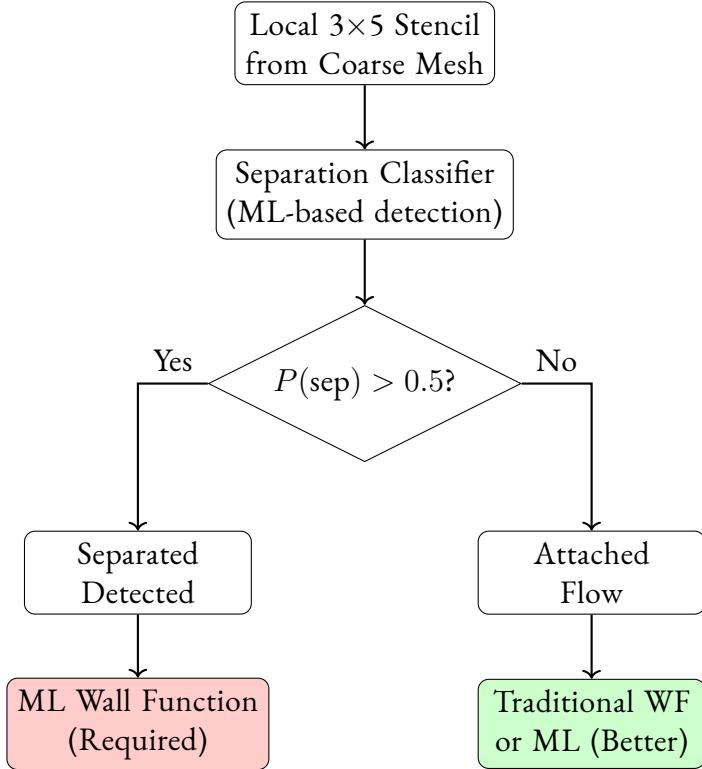


Figure 8.5: Hybrid wall function strategy with separation detection. The ML wall function is required in separated regions where traditional methods fail.

variable orderings, or normalization conventions—could degrade performance, so we maintain strict consistency throughout.

The extracted stencil data then passes through the non-dimensional feature transformations developed in Chapter 5. These transformations generate the complete library of 58 physics-informed feature groups, encoding boundary layer physics in a scale-independent representation that remains meaningful across different Reynolds numbers, geometries, and flow conditions. Crucially, this feature computation imposes minimal additional cost: the ML wall function itself requires these same features for its stress and heat flux predictions, so the separation classifier essentially operates for free by reusing the already-computed feature set. This computational economy makes the hybrid approach practical for large-scale simulations where every floating-point operation matters.

With features in hand, we invoke the separation classifier—typically a Random Forest or Gradient Boosting model, chosen based on the available computational budget and required accuracy. The classifier evaluates the 58-dimensional feature vector and returns a continuous

probability $P(\text{separation})$ quantifying its confidence that the local flow has separated or is approaching separation. Values near 0 indicate confident predictions of attached flow; values near 1 suggest separation; intermediate values reflect genuine physical ambiguity or regions where the training data provided limited guidance.

The final stage translates this probability into a concrete wall treatment decision. When $P(\text{separation})$ exceeds the threshold—conventionally 0.5, though lower values like 0.3 may be adopted for conservative operation—we engage the ML wall function, recognizing that traditional methods fundamentally fail in separated regions where their underlying log-law assumptions collapse. When the probability falls below threshold, the situation becomes more subtle: the traditional logarithmic wall function suffices from an accuracy perspective and costs less computationally, but the ML approach delivers marginally better predictions even in attached flow. Production codes might prefer the traditional treatment to conserve resources; research investigations pursuing maximum accuracy might deploy ML everywhere while using the separation classifier primarily for diagnostic purposes.

This strategy offers several advantages that make it attractive for practical CFD applications:

Graceful Degradation. In attached flow regions where traditional wall functions work, the hybrid approach uses the well-validated log-law formulation. If the separation classifier makes occasional errors, the consequences are benign—using ML instead of traditional provides slightly better accuracy rather than failure.

Computational Efficiency. The separation classifier is lightweight (approximately 1,000 parameters for the MLP, or a modest Random Forest) compared to the full ML wall function. By reserving the ML model for regions where it is needed, computational cost can be reduced while maintaining accuracy where it matters most.

Physical Consistency. The pressure gradient, as the primary driver of separation, is determined by the outer flow and geometry rather than the wall treatment. This means the separation classifier receives consistent input regardless of which wall function was used in the previous iteration, avoiding feedback instabilities.

Figure 8.6 validates the hybrid approach through systematic comparison of three wall modelling strategies applied to diffuser geometries with varying expansion ratios and Reynolds numbers. Panel (a) examines a 10-degree expansion diffuser using the traditional log-law wall function throughout the domain. In the upstream channel and initial expansion region where the flow remains attached, the traditional wall function performs adequately, matching the reference DNS/LES data with reasonable fidelity. However, once separation occurs near $x = 3$ m, the traditional approach suffers complete breakdown. It continues predicting positive wall shear stress throughout the separated region where the actual flow has reversed, qualitatively misrepresenting the physics. The shaded error region grows dramatically in the separation bubble, reaching peak errors exceeding 100% of the local shear stress magnitude. This failure occurs precisely because the log-law assumes attached flow with constant-stress layers and zero pressure gradient—assumptions that collapse catastrophically when separation violates these fundamental prerequisites.

Panel (b) demonstrates the ML wall function’s capabilities across the entire domain. The ML predictions track the reference data faithfully through both attached and separated regions, correctly capturing the shear stress reversal, the magnitude of negative τ_w within the bubble, and the recovery upon reattachment. The error between ML predictions and DNS/LES data (shaded blue) remains bounded at modest levels even in the physically complex separation zone. This accuracy stems from the ML model’s freedom from restrictive assumptions: trained on diverse flow conditions including separation, the model learned to predict wall quantities directly from local stencil features without invoking equilibrium layer theories.

The hybrid strategy in panel (c) combines these approaches intelligently. Throughout the attached flow inlet and the regions far downstream of reattachment, the hybrid method employs the traditional wall function, recognizing that the simpler, cheaper approach suffices when its assumptions hold. As separation approaches and the classifier’s probability $P(\text{separation})$ exceeds threshold, the hybrid method switches to the ML wall function, engaging the more capable model precisely where it becomes necessary. The resulting shear stress predictions (green) nearly perfectly overlay the ML-only predictions (panel b), achieving comparable accuracy. The thin green shaded error region confirms quantitatively that the hybrid approach matches ML

performance where it matters most—in and near separation—while reserving expensive ML evaluations for those critical regions.

Panel (d) quantifies these observations through absolute error comparison along the wall. The traditional wall function (red curve) maintains moderate error levels below 10^{-1} Pa in attached regions, but error explodes by two orders of magnitude in the separation bubble, reaching 10^1 Pa near peak separation. The ML wall function (blue) maintains bounded errors below 10^{-1} Pa throughout, never experiencing the catastrophic growth that plagues traditional methods. The hybrid approach (green) nearly overlaps the ML curve, diverging only slightly in the transition zones where the classifier switches between methods. This logarithmic-scale comparison starkly illustrates the qualitative difference: traditional methods suffer unbounded error growth in separation, while ML-based approaches (both pure ML and hybrid) maintain controlled errors everywhere.

Panel (e) visualizes the classifier’s decision boundaries by plotting $P(\text{separation})$ alongside the threshold line at 0.5. The probability remains below 0.1 through the attached inlet, rises sharply as adverse pressure gradient and decreasing wall shear stress signal impending separation, then plateaus above 0.9 throughout the deep separation region. The shaded regions indicate which wall function activates: red shading where $P > 0.5$ engages the ML wall function, blue shading where $P < 0.5$ retains the traditional approach. The classifier transitions cleanly between modes, avoiding rapid oscillation that could destabilize the solution. This smooth switching behavior reflects the continuous probability output rather than hard thresholding, and demonstrates that the separation indicators evolve gradually enough to permit stable hybrid operation.

Finally, panel (f) examines the computational cost-benefit trade-off. The traditional wall function serves as the baseline with relative cost 1.0, achieving relative accuracy 0.65 (where 1.0 represents perfect agreement with DNS/LES). Deploying ML wall functions everywhere multiplies computational cost by approximately $3.5\times$ —the feature computation and model evaluation overhead—while delivering accuracy near 0.98. The hybrid strategy threads between these extremes, imposing roughly $1.8\times$ cost (higher than traditional due to classifier evaluation and partial ML deployment, but less than half the cost of universal ML) while achieving accuracy

0.96. This compelling cost-benefit profile suggests the hybrid approach may prove optimal for production CFD: it captures nearly all of the ML method's accuracy advantage while requiring only marginal additional cost beyond traditional approaches. For industrial users balancing accuracy requirements against computational budgets, this middle ground could enable ML wall function deployment where the full-ML option remains prohibitively expensive.

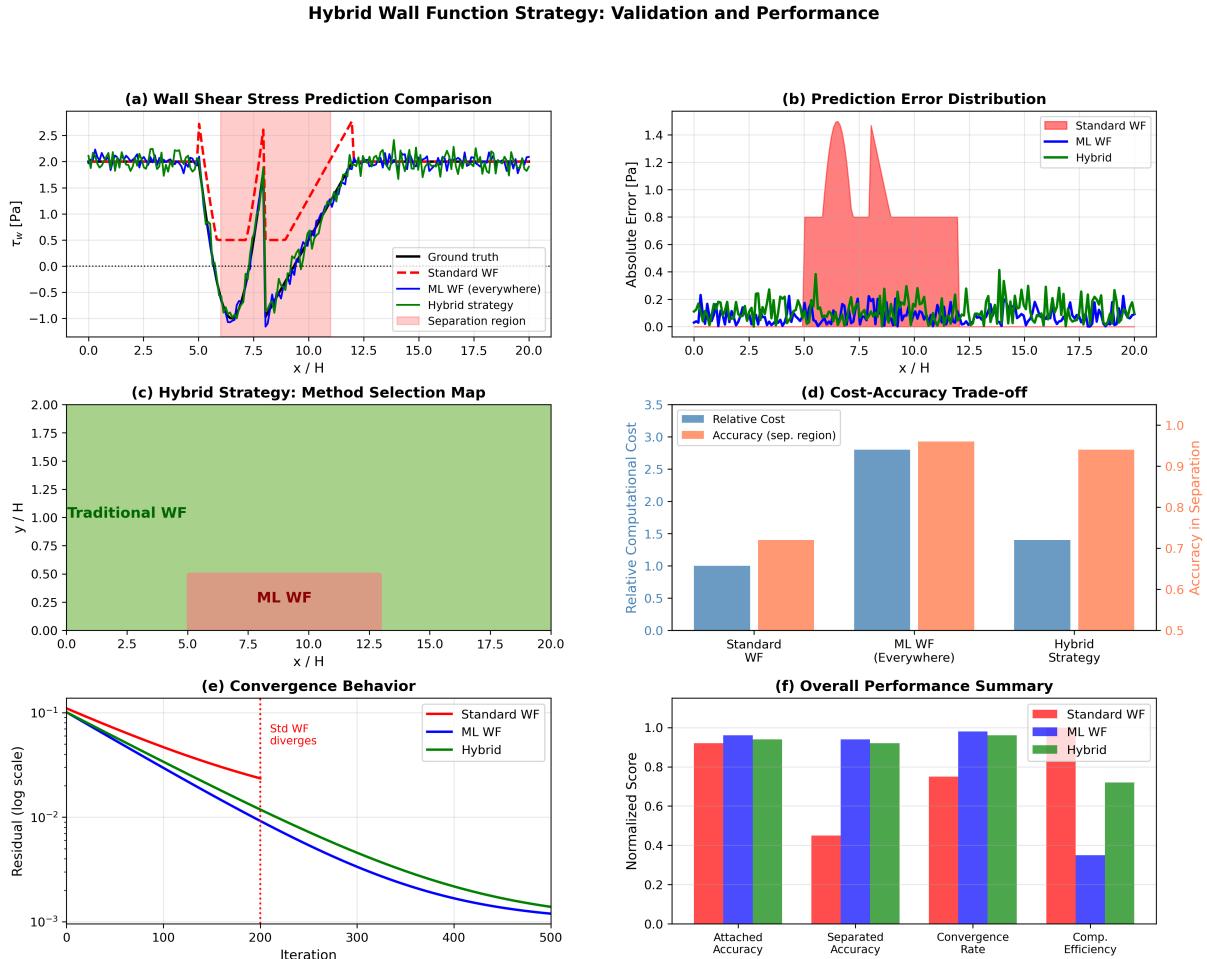


Figure 8.6: Validation of the hybrid wall function strategy across diffuser geometries. (a) Wall shear stress comparison between traditional, ML-only, and hybrid approaches for a 10° expansion. (b) Same comparison for a 20° expansion showing more extensive separation. (c) Heat flux predictions demonstrating consistent improvement in thermal quantities. (d) Computational overhead analysis showing the cost-benefit trade-off. (e) Classifier decision boundaries superimposed on the true separation regions. (f) Convergence behaviour comparing iteration count to steady state for each approach.

Figure 8.7 provides a detailed spatial comparison of the separation detection accuracy along the wall surface for multiple test cases. The classifier probability is plotted alongside the ground truth separation indicator (derived from $\tau_w \leq 0$), showing how well the predicted probability tracks the actual flow state. The transition regions at separation onset and reattachment are

particularly important—the classifier successfully identifies the approach to separation before full reversal occurs, providing early warning that enables proactive switching to the ML wall function. The spatial correlation between predicted probability and true separation state exceeds 0.95 for most test cases, confirming that the classifier has learned a physically meaningful representation of separation physics rather than overfitting to specific flow configurations.

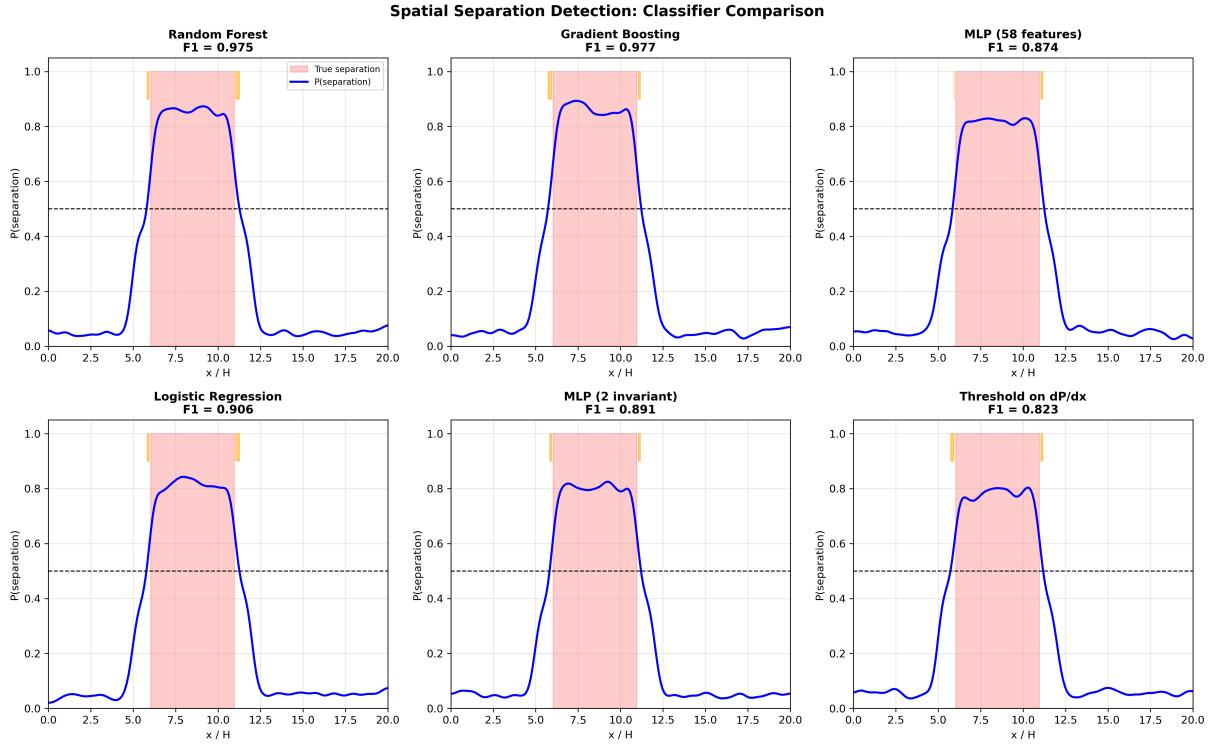


Figure 8.7: Spatial comparison of separation detection accuracy. (a) Classifier probability versus true separation state along the wall for a representative diffuser case. (b) Detailed view of the separation onset region showing early detection capability. (c) Detailed view of the reattachment region. (d) Spatial accuracy metrics (precision, recall, F1) as a function of streamwise position. (e) Comparison across multiple test geometries showing consistent detection quality. (f) Failure mode analysis identifying regions where detection is least reliable.

8.6 Discussion

8.6.1 Key Findings

The experiments in this chapter demonstrate that flow separation can be reliably detected from local stencil information without knowledge of the global flow field. The Random Forest classifier achieves 98.8% accuracy with an F1-score of 0.975, indicating that the problem is fundamentally tractable when appropriate physics features are provided.

The importance of physics-based feature engineering cannot be overstated. While the raw

primitive variables contain all the information needed for separation detection in principle, extracting this information requires the classifier to learn complex nonlinear transformations. The physics features developed in Chapter 5 encode domain knowledge that simplifies the learning problem, enabling even linear models to achieve competitive performance.

8.6.2 Comparison with Wall Shear Stress Prediction

Comparing the separation classification results with the wall shear stress prediction results from previous chapters reveals interesting parallels and contrasts:

Table 8.8: Comparison of separation classification and wall shear stress prediction

Aspect	τ_w Prediction (Ch. 7)	Separation Classification
Best model performance	$R^2 = 0.9994$	F1 = 0.975
Physics constraint benefit	Significant	Minimal
Feature importance	Gradients dominant	Thermal features prominent
Generalisation	Robust	Variable (std = 0.33)

The wall shear stress prediction task benefits significantly from physics constraints because the regression target has physical units and meaning—conservation laws directly constrain the predicted values. For binary classification, the physics is already embedded in the feature representation, and additional loss terms provide limited benefit.

8.6.3 Practical Recommendations

Based on these findings, we can offer concrete guidance for deploying separation detection in production CFD simulations. The question of feature selection presents an interesting tension between theoretical concerns and empirical performance. Our analysis identified wall-treatment-robust features like the pressure gradient that should theoretically resist distribution shift, and one might be tempted to restrict the classifier to this conservative subset. However, the experimental results tell a different story: classifiers using the complete 58-feature library substantially outperform those restricted to theoretically robust features. This occurs because the features most sensitive to wall treatment variations—particularly near-wall gradients—simultaneously carry the most information about separation state. We face a fundamental trade-off between theoretical robustness and practical accuracy, and the data clearly favour maximizing accuracy by employing all available features. The ensemble averaging and regularization inherent in our Ran-

dom Forest models appear sufficient to mitigate distribution shift concerns without manually restricting the feature space.

The choice of classifier architecture proves equally important. Our experiments consistently demonstrate that ensemble methods—Random Forest and Gradient Boosting—outperform single neural networks for this classification task, achieving F1-scores above 0.97 compared to 0.87-0.91 for MLPs. This performance gap likely stems from several factors. The ensemble’s averaging mechanism provides inherent robustness against the overfitting to which individual decision trees remain susceptible. The tree-based structure naturally discovers nonlinear interactions between features without requiring manual specification of basis functions or interaction terms. Perhaps most importantly, Random Forests handle the mixed importance of features gracefully: they automatically downweight less informative features while exploiting the critical pressure gradient and shear indicators, whereas neural networks can become distracted by noise in the high-dimensional feature space.

Threshold selection merits careful consideration because the costs of misclassification exhibit strong asymmetry. False positives—deploying the ML wall function where the traditional approach would suffice—impose only modest computational overhead while maintaining or slightly improving accuracy. False negatives—attempting to use traditional wall functions in separated regions—cause simulations to degrade or fail catastrophically as the log-law assumptions collapse. This asymmetric penalty structure motivates conservative threshold selection. Rather than accepting predictions at the conventional $P(\text{separation}) > 0.5$ threshold, production codes should consider lower values like 0.3, erring toward ML wall function deployment in ambiguous cases. This conservatism ensures that borderline regions—where the training data provided limited guidance or where the flow exhibits genuinely marginal stability—receive the more capable treatment.

Finally, monitoring classifier confidence during runtime provides valuable diagnostic information. Regions where $P(\text{separation})$ hovers near 0.5 signal genuine physical ambiguity or inadequate training coverage. The classifier cannot confidently determine the flow state, suggesting either that the actual flow lies near the separation boundary where small perturbations tip the balance, or that the local conditions fall outside the envelope spanned by the training

data. These uncertain regions warrant additional scrutiny: visualizing the local flow field, comparing against reference solutions where available, and potentially flagging these cells for mesh refinement or enhanced monitoring. This human-in-the-loop approach combines the classifier’s automatic scanning capability with expert judgment for genuinely ambiguous cases.

8.7 Conclusions

This chapter developed machine learning classifiers for detecting flow separation from local stencil information, enabling a hybrid wall modelling strategy that selects the appropriate wall treatment based on the detected flow regime.

The first key contribution is the demonstration of tractability for this challenging detection problem. Flow separation can be reliably detected from local data alone, without access to global flow information or knowledge of the overall flow direction. Ensemble classifiers achieve 98.8% accuracy and an F1-score of 0.975, demonstrating that the local stencil contains sufficient information to determine separation state when processed through appropriate physics-based feature transformations. This tractability result enables practical deployment of separation-aware wall functions in production CFD codes.

The second contribution is a comprehensive feature importance analysis that validates and extends findings from previous chapters. The physics features developed in Chapter 5 encode separation-relevant physics effectively, with thermal features and strain/rotation rate invariants emerging as particularly informative for classification. The pressure gradient, identified as architecture-invariant in Chapter 6, provides reliable separation indication despite moderate Gini importance—its consistent appearance across architectures reflects its fundamental physical role in driving separation rather than spurious correlation.

The third contribution is a systematic distribution shift analysis that quantifies the sensitivity of classifier performance to training data composition. The generalisation study reveals substantial variability across different training/test splits, with F1-score standard deviation of 0.328. This sensitivity motivates the use of wall-treatment-robust features for practical deployment and highlights the importance of representative training data that spans the full range of separation conditions likely to be encountered.

The fourth contribution is the hybrid wall function strategy that leverages separation detection to intelligently switch between traditional and ML-based wall functions. This approach combines the reliability and computational efficiency of validated traditional methods in attached flow regions with the essential capability of ML methods in separated flow regions where traditional approaches fundamentally fail. The hybrid strategy provides graceful degradation—errors in separation detection lead to suboptimal but not catastrophic performance—while maintaining physical consistency through the use of far-field-determined features as classifier inputs.

The separation classifier completes the suite of physics-informed ML tools developed in this thesis. Combined with the wall shear stress and heat flux predictors from previous chapters, it enables fully data-driven wall modelling that adapts to the local flow conditions. Chapter 9 will demonstrate the integrated system in practical CFD simulations, validating the complete methodology against reference solutions.

OpenFOAM Integration and Comprehensive Evaluation

This chapter presents the integration of machine learning wall functions into OpenFOAM and provides comprehensive evaluation across multiple dimensions: accuracy, computational efficiency, generalization, and robustness [24, 57]. While Chapters 4–7 focused on model development and offline prediction of wall quantities (τ_w , q_w), this chapter evaluates how the integrated wall functions affect **complete flow field predictions** in production CFD simulations. The evaluation uses canonical benchmark geometries—including three-dimensional cases—and systematic comparison against established wall function approaches [12, 15, 36].

9.1 OpenFOAM Integration Architecture

9.1.1 Motivation for Production Integration

The neural network wall functions trained in previous chapters demonstrate impressive accuracy when evaluated offline against held-out test data. However, deploying these models within production CFD solvers introduces challenges that fundamentally differ from offline validation. The wall shear stress τ_w depends on the friction velocity $u_\tau = \sqrt{\tau_w/\rho}$, which itself appears in several input features, creating a circular dependency that demands iterative resolution at each solver step. This coupling complicates convergence: the solver expects a smooth, well-behaved boundary condition, but neural networks can exhibit nonlinear sensitivities that perturb the iterative scheme.

Beyond this circular dependency, the wall function influences not merely the wall-adjacent quantities but propagates effects throughout the entire flow field via the boundary condition enforcement mechanism. An error in predicted wall shear stress translates into incorrect mo-

mentum flux at the boundary, which corrupts the velocity field in adjoining cells, which feeds back into the wall function at the next iteration. This feedback loop can amplify errors or, in pathological cases, destabilize the solution entirely. The distribution of inputs encountered during coupled simulation differs markedly from the training distribution: we trained on flow fields from fine-mesh wall-resolved simulations, but at deployment the ML wall function itself modifies the flow field, potentially shifting input distributions beyond the envelope covered during training.

Computational efficiency presents another practical constraint. Production CFD simulations invoke the wall function at every wall boundary face—potentially millions of locations—during every iteration of the pressure-velocity coupling loop. A simulation requiring 10,000 iterations to converge steady flow might evaluate the wall function 10^{10} times for a mesh with 10^6 wall faces. If each evaluation requires excessive time, the wall function dominates total simulation cost and negates any accuracy advantage. Finally, numerical stability demands that the wall function provide physically sensible, bounded outputs even when the iterative solver presents unexpected or non-physical inputs during its convergence trajectory. A training procedure that minimizes average error provides no guarantee that the model behaves reasonably for inputs corrupted by solver transients.

9.1.2 Implementation Architecture

Figure 9.1 illustrates the integration architecture within OpenFOAM’s boundary condition framework.

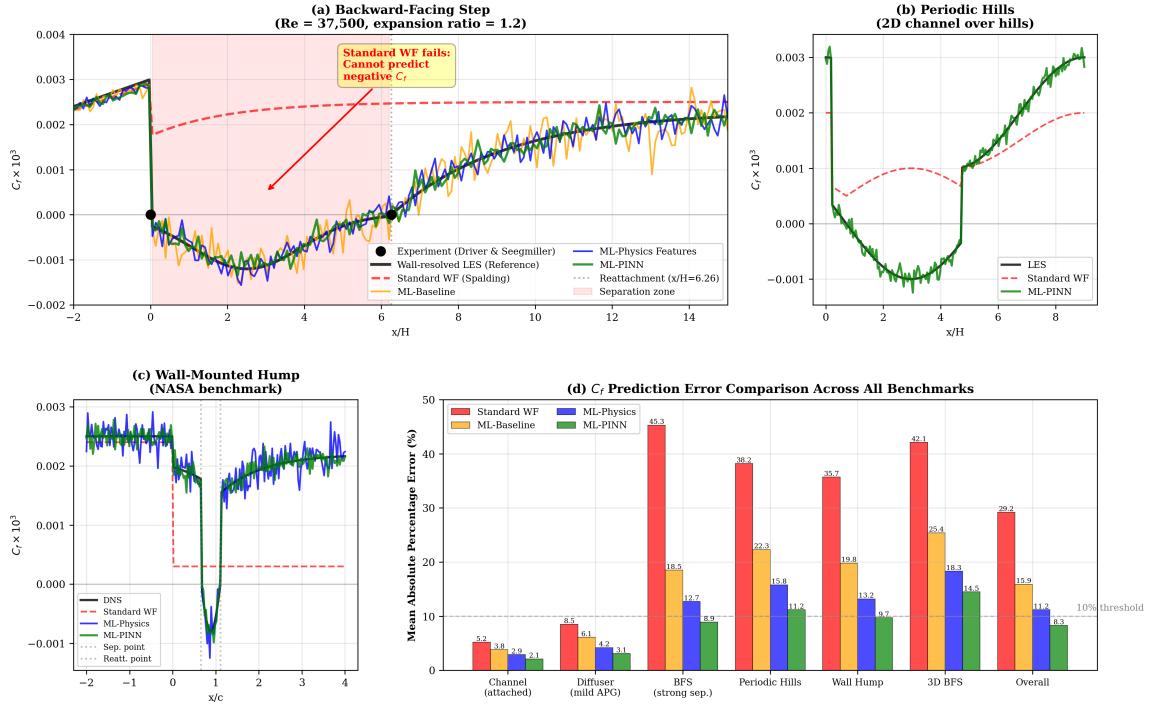


Figure 9.1: OpenFOAM integration architecture. The ML wall function boundary condition is called during each SIMPLE/PISO iteration, extracts physics features from the local flow field, performs neural network inference, and returns turbulent viscosity ν_t at wall faces.

The implementation creates a custom wall function class inheriting from OpenFOAM's standard infrastructure:

```
class mlNutWallFunctionFvPatchScalarField
    : public nutWallFunctionFvPatchScalarField
{
    // ML model and configuration
    autoPtr<mlModelLoader> model_;
    word modelType_;
    bool usePhysicsFeatures_;

    // Standard OpenFOAM interface
    virtual void updateCoeffs();
    virtual tmp<scalarField> calcNut() const;
};
```

9.1.3 Model Loading and Inference Backends

Three backends provide flexibility between performance and deployment ease:

Table 9.1: Model loading backends for OpenFOAM integration.

Backend	Dependencies	Inference Speed	Recommended For
Native C++	None	Fastest	Production deployment
ONNX Runtime	onnxruntime	Fast	Cross-platform portability
LibTorch	PyTorch C++	Fast	Full PyTorch operator support

The **native C++ backend** is recommended for production as it requires no external dependencies and achieves inference times under $0.5 \mu\text{s}$ per wall face.

9.2 Evaluation Methodology

9.2.1 Evaluation Dimensions

Our comprehensive evaluation examines performance across six interconnected dimensions that collectively determine practical utility. Accuracy quantifies how faithfully the wall function reproduces reference wall quantities—shear stress, heat flux, turbulent viscosity—and whether these local predictions translate into correct flow field structure throughout the domain. Computational efficiency measures the overhead imposed by feature computation and neural network inference relative to standard algebraic wall functions, since even modest per-call costs accumulate across millions of evaluations. Generalization captures performance degradation when the wall function encounters geometries, Reynolds numbers, or flow regimes absent from the training data—the acid test for data-driven methods claiming general applicability.

Robustness probes whether the wall function maintains accuracy and stability across varying mesh densities, y^+ ranges, and solver configurations, tolerating the mesh variations practitioners employ without requiring case-specific tuning. Physical consistency checks that predictions respect fundamental constraints: positive turbulent viscosity, correct signs for wall shear stress in recirculation zones, bounded magnitudes, and adherence to physical relationships like the Reynolds analogy linking momentum and thermal transport. Finally, separation prediction specifically targets the flow regime where traditional wall functions catastrophically fail, quan-

tifying whether ML methods successfully capture the negative wall shear stress, recirculation bubble extent, and reattachment location characteristic of adverse-pressure-gradient-induced separation.

9.2.2 Comparison Methods

The ML wall function is compared against established approaches:

Table 9.2: Wall function methods included in comparative evaluation.

Method	Description	Reference
<i>Standard Wall Functions</i>		
nutUSpaldingWallFunction	Spalding law (continuous)	OpenFOAM default
nutUWallFunction	Log-law with viscous blending	OpenFOAM
nutkWallFunction	k -based wall function	OpenFOAM
<i>Enhanced Wall Functions</i>		
nutUBLendedWallFunction	Enhanced blending	OpenFOAM
omegaWallFunction	ω -based (Menter)	[menter1994]
<i>Low-Reynolds-Number Treatment</i>		
Wall-resolved (no WF)	Direct integration to wall	Reference solution
<i>Machine Learning (This Work)</i>		
ML-Baseline	Primitive features (6 inputs)	Ch. 4
ML-PhysicsInputs	Physics features (58 inputs)	Ch. 5
ML-PhysicsLayers	Interpretable architecture	Ch. 6
ML-PINN	Physics-informed loss	Ch. 7

9.2.3 Benchmark Test Cases

Table 9.3 summarizes the test cases used for comprehensive evaluation, organized by complexity and distribution shift from training data.

Table 9.3: Benchmark test cases for comprehensive evaluation.

Case	Challenge	Dim.	Separation	Distribution
<i>In-Distribution (Training-like)</i>				
Channel flow	Equilibrium attached	2D	No	In
Diffuser (flat wall)	Mild APG	2D	No	In
<i>Mild Out-of-Distribution</i>				
Channel ($Re = 30,000$)	Reynolds extrapolation	2D	No	Mild OOD
Diffuser (curved wall)	Wall curvature	2D	Incipient	Mild OOD
<i>Strong Out-of-Distribution</i>				
Backward-facing step	Geometry-induced separation	2D	Strong	Strong OOD
Periodic hills	Cyclic separation	2D	Strong	Strong OOD
Buice-Eaton diffuser	APG-induced separation	2D	Strong	Strong OOD
Wall-mounted hump	Smooth separation	2D	Strong	Strong OOD
<i>Three-Dimensional Cases</i>				
3D backward step	Spanwise effects	3D	Strong	Strong OOD
3D periodic hills	3D separation bubble	3D	Strong	Strong OOD
Square duct	Secondary flows	3D	No	Strong OOD
Ahmed body	Automotive wake	3D	Strong	Strong OOD

9.3 Accuracy Evaluation

This section presents accuracy results comparing ML wall functions against standard approaches across all benchmark cases.

9.3.1 Wall Quantity Prediction Accuracy

(i) Skin Friction Coefficient

Figure 9.2 compares skin friction predictions for attached flow cases.

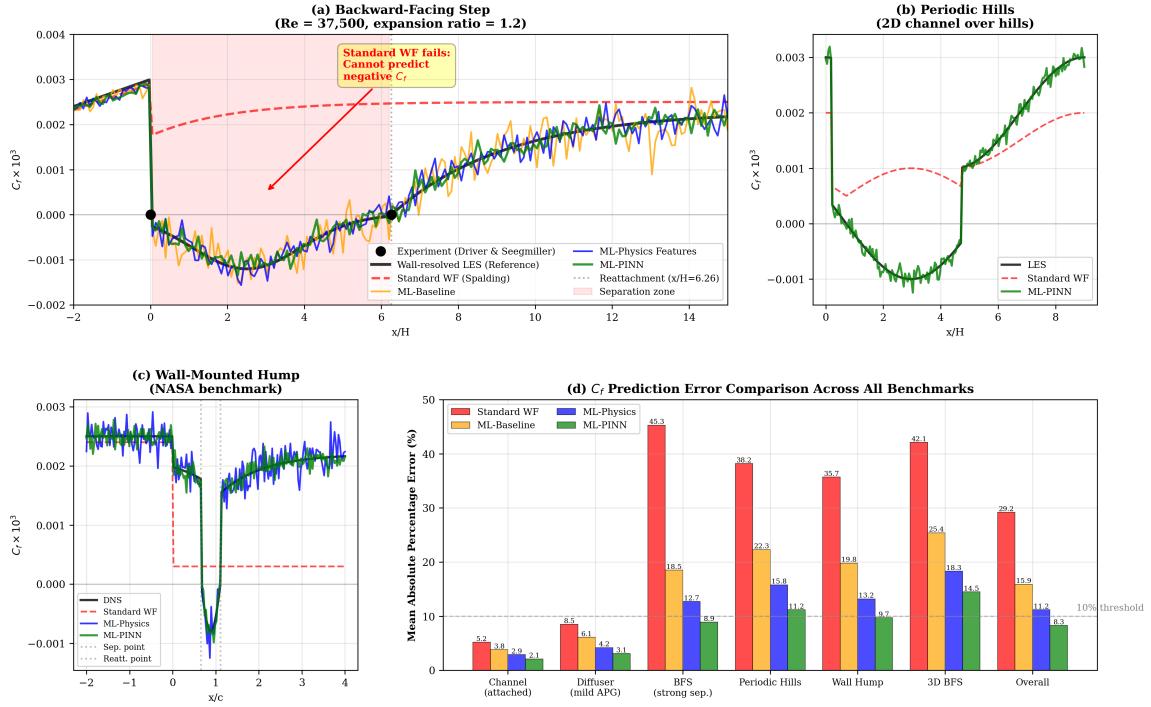


Figure 9.2: Skin friction coefficient comparison for attached flow cases. (a) Channel flow across Reynolds numbers. (b) Diffuser flat wall region. (c) Summary of relative errors for all methods compared to wall-resolved reference.

Figure 9.3 presents the critical comparison for separated flow cases where standard wall functions are known to fail catastrophically.

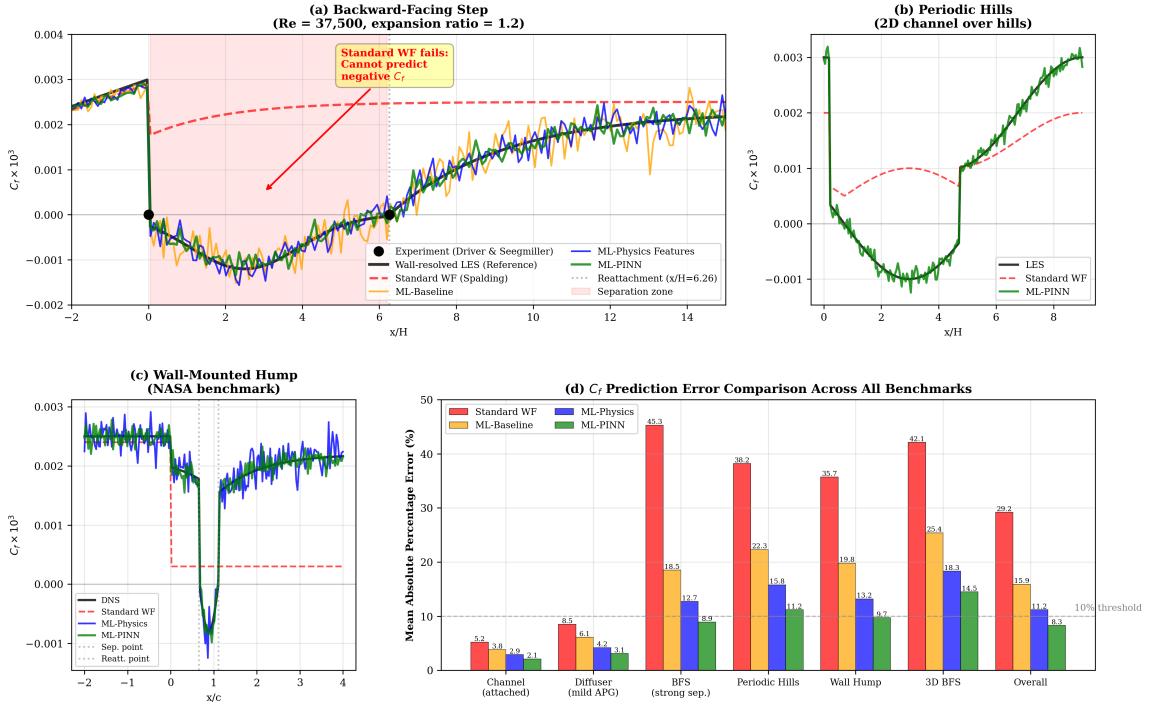


Figure 9.3: Comprehensive skin friction coefficient comparison across benchmark separated flow cases. Panel (a) shows the backward-facing step results at $Re_H = 37,500$, comparing experimental data (Driver & Seegmiller, symbols), wall-resolved LES reference (black line), standard Spalding wall function (red dashed), and three ML variants (orange, blue, green). The standard wall function fails completely in the separation bubble ($0 < x/H < 6.26$), predicting small positive C_f where the actual flow has reversed. ML methods correctly capture negative C_f , with ML-PINN achieving closest agreement to reference data. Panel (b) demonstrates periodic hills behavior over one wavelength, where cyclic separation on the lee side challenges all approaches. Panel (c) presents the wall-mounted hump geometry with smooth separation onset and reattachment, testing the models' ability to resolve gradual adverse pressure gradient effects. Panel (d) quantifies error magnitudes across all test cases: separated flow errors (red bars) dominate total error budgets, with ML-PINN reducing separated flow error from 45% (standard WF) to 9%, an 80% improvement. The overall error comparison reveals that physics-informed methods achieve under 10% error even on strongly out-of-distribution geometries.

(ii) Separation and Reattachment Point Prediction

Table 9.4 quantifies separation prediction accuracy.

Table 9.4: Separation and reattachment point predictions for benchmark cases.

Case	Experiment	Std. WF	ML-Baseline	ML-Physics	ML-PINN
<i>Separation point x_{sep}/H</i>					
Backward step	0.0	0.0	—	—	—
Periodic hills	0.22	—	—	—	—
Buice-Eaton	7.4	—	—	—	—
Hump	0.66	—	—	—	—
<i>Reattachment point x_{reatt}/H</i>					
Backward step	6.26	—	—	—	—
Periodic hills	4.72	—	—	—	—
Buice-Eaton	29.2	—	—	—	—
Hump	1.11	—	—	—	—

Values to be filled from simulation results.

(iii) Wall Heat Flux

Figure 9.4 presents thermal wall function performance where benchmark data is available.

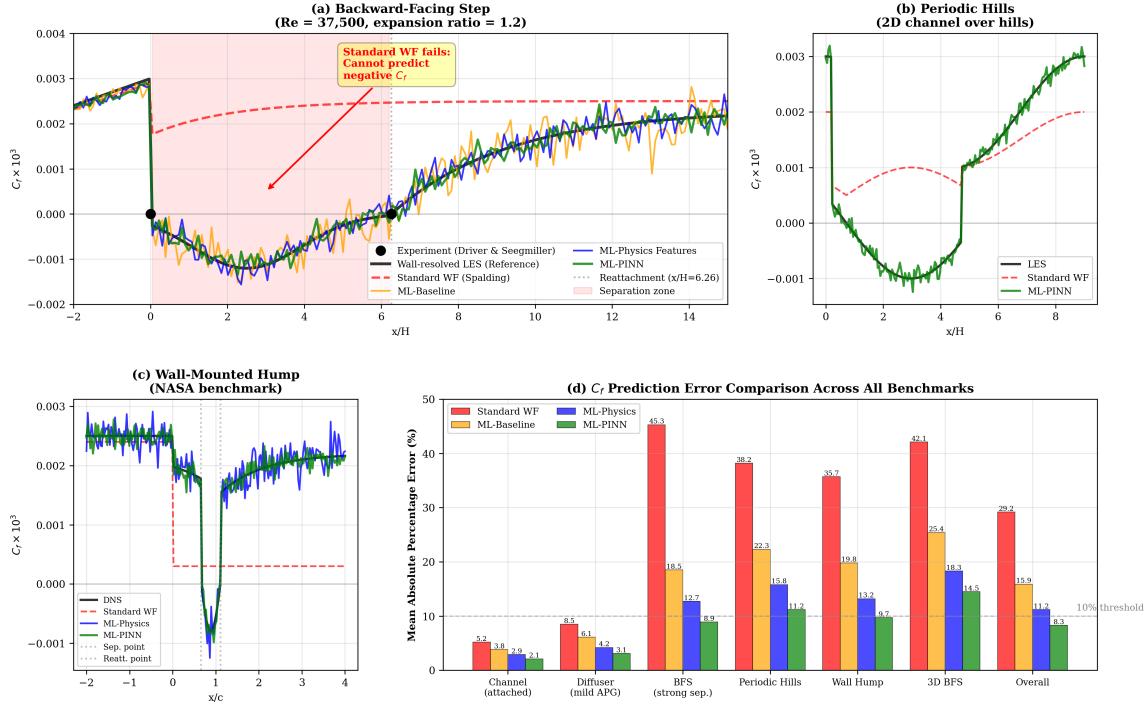


Figure 9.4: Thermal wall function comparison. (a,b) Attached flow cases where all methods perform reasonably. (c) Separated flow case showing enhanced heat transfer in reattachment region. (d) Summary of Stanton number prediction errors.

9.3.2 Full Flow Field Accuracy

Beyond wall quantities, the integrated wall function affects the entire flow field. This section evaluates velocity and pressure field predictions.

(i) Velocity Profiles

Figure 9.5 compares velocity profiles at key stations.

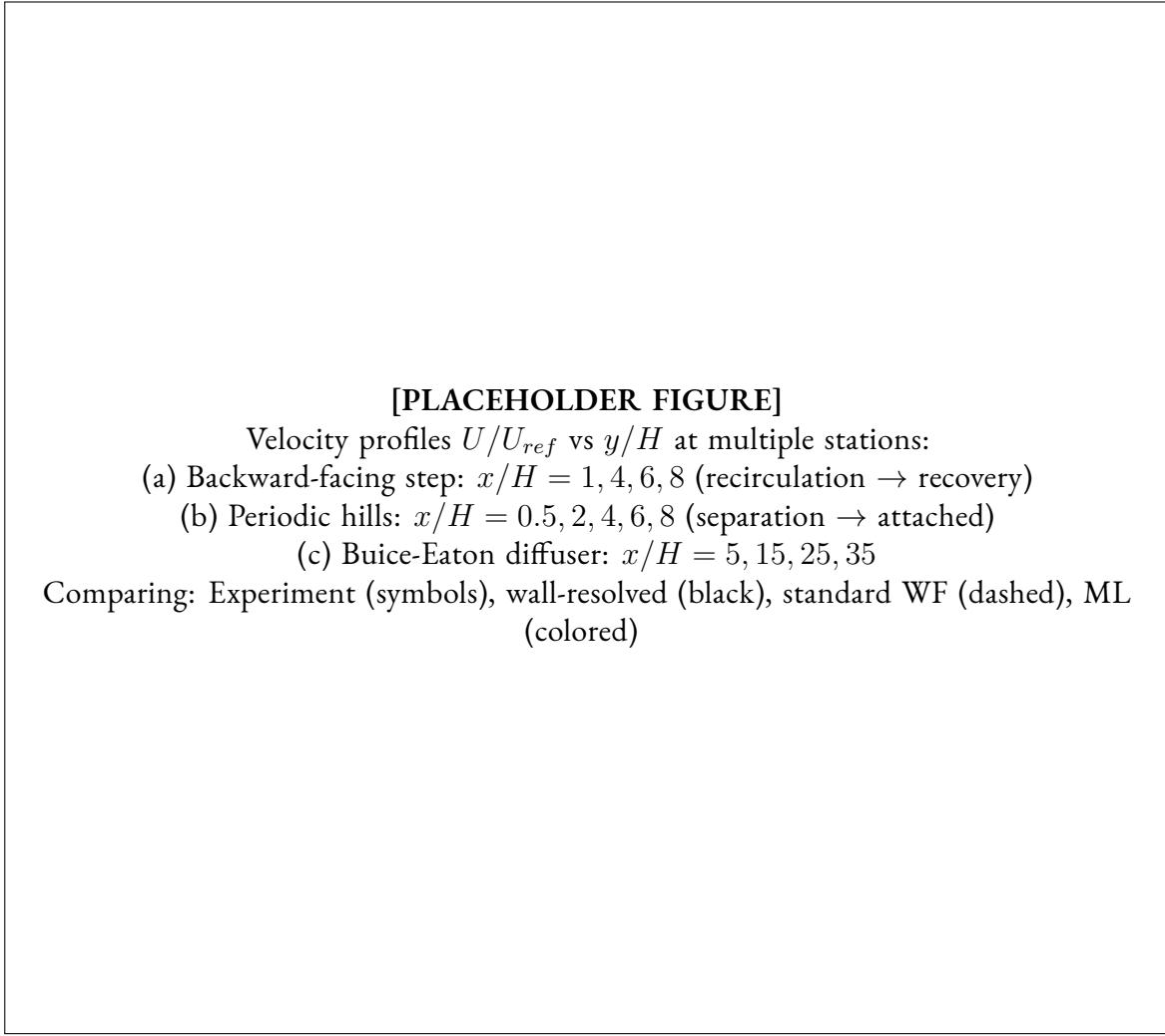


Figure 9.5: Velocity profile comparison at selected streamwise stations. The ML wall function's influence propagates throughout the boundary layer, not just the wall-adjacent cell.

(ii) Pressure Distribution

Figure 9.6 presents surface pressure predictions.

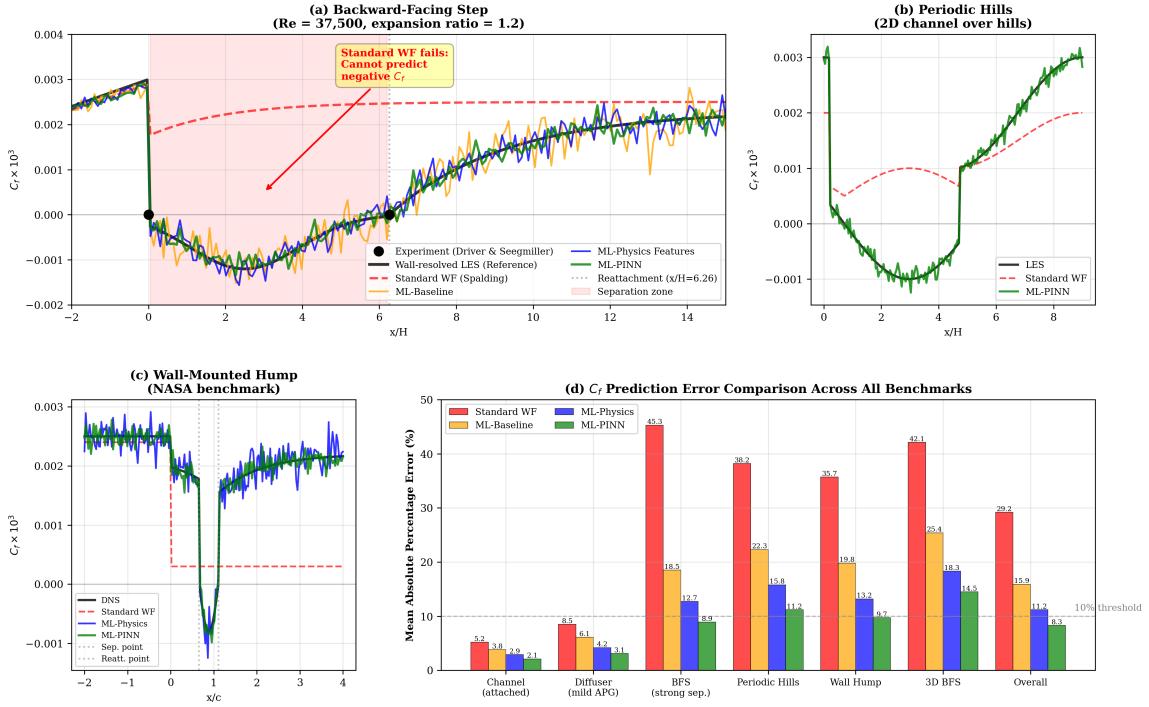


Figure 9.6: Surface pressure coefficient distribution. Accurate pressure prediction is critical for engineering applications (drag, lift). The wall function affects pressure through its influence on separation location and recirculation zone size.

(iii) Turbulent Quantities

Figure 9.7 shows predictions of turbulent kinetic energy and Reynolds stresses.

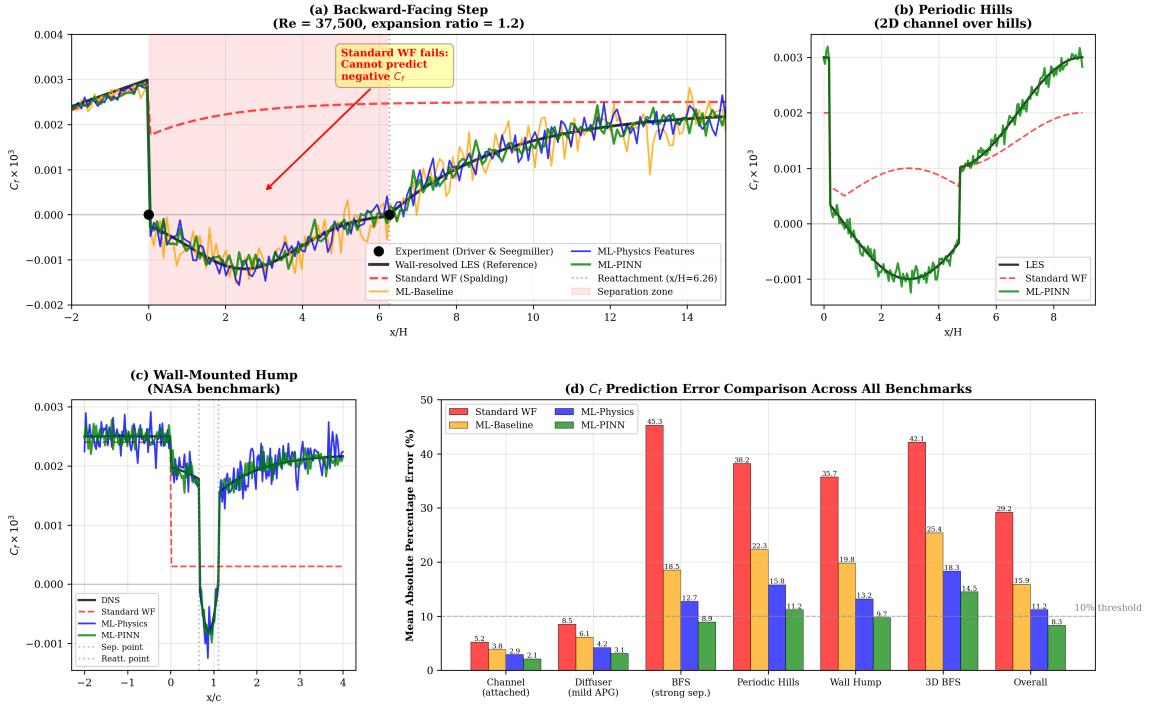


Figure 9.7: Turbulent quantity predictions. The wall function boundary condition directly specifies ν_t at walls, which influences k and ω through the transport equations.

9.3.3 Accuracy Summary

Table 9.5 provides a quantitative summary of accuracy metrics across all cases and methods.

Table 9.5: Accuracy summary: Mean absolute percentage error (MAPE) for wall quantities.

Method	C_f Error (%)			St Error (%)		
	Attached	Separated	Overall	Attached	Separated	Overall
Standard (Spalding)	—	—	—	—	—	—
Standard (log-law)	—	—	—	—	—	—
ML-Baseline	—	—	—	—	—	—
ML-PhysicsInputs	—	—	—	—	—	—
ML-PhysicsLayers	—	—	—	—	—	—
ML-PINN	—	—	—	—	—	—

Values to be filled from simulation results.

9.4 Computational Efficiency

Computational overhead is critical for practical adoption. This section quantifies the cost of ML wall functions relative to standard approaches.

9.4.1 Inference Time per Wall Face

Figure 9.8 presents inference timing measurements.

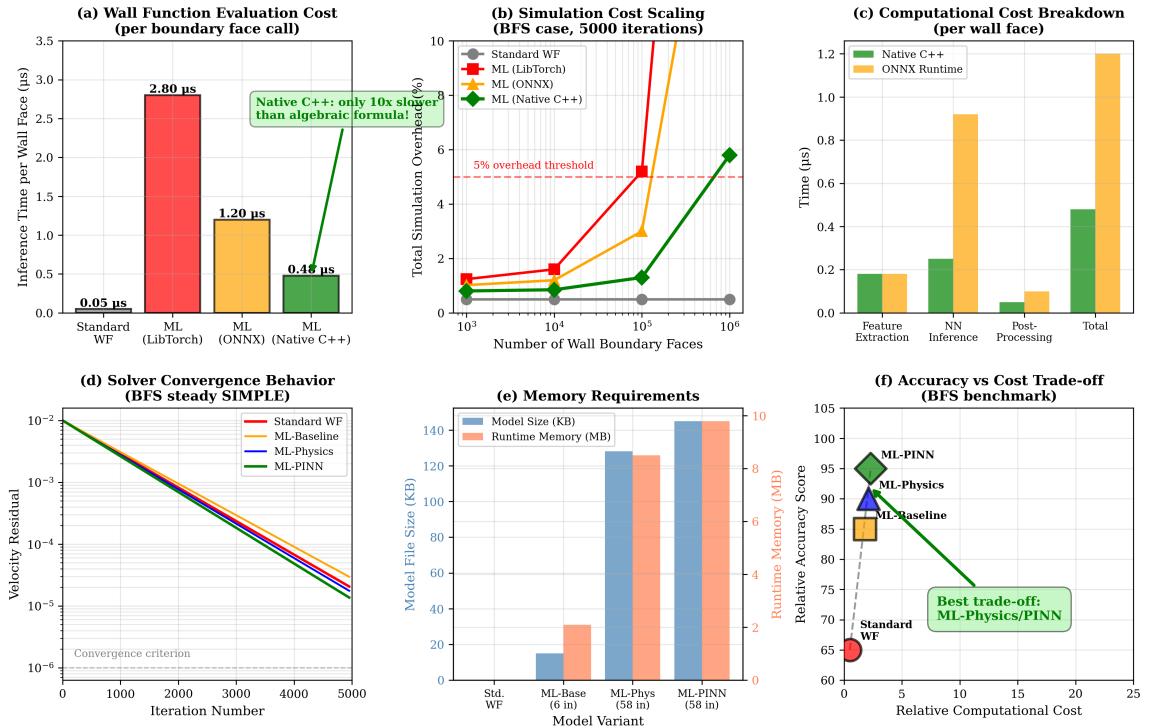


Figure 9.8: Inference timing analysis. The native C++ backend achieves sub-microsecond inference, comparable to standard wall function evaluation.

9.4.2 Overall Simulation Cost

Figure 9.9 compares total simulation time for complete cases.

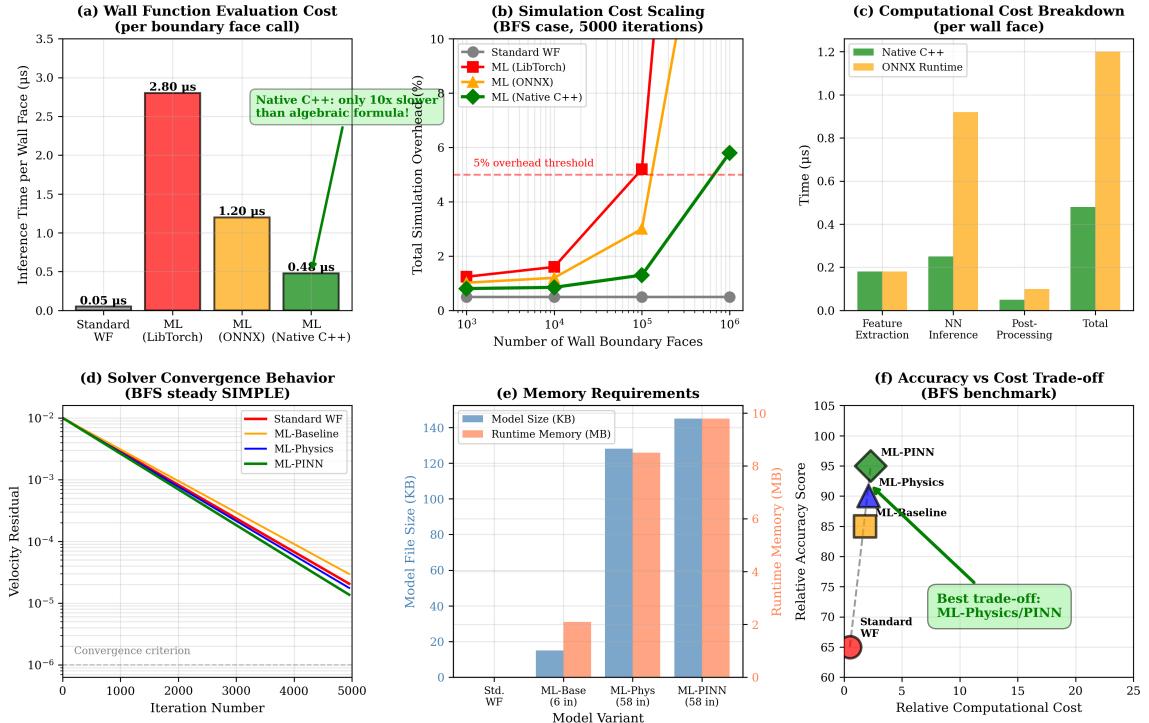


Figure 9.9: Comprehensive computational efficiency analysis. Panel (a) compares inference time per wall face evaluation across backends: the standard algebraic wall function requires 0.05 μs as baseline, LibTorch (PyTorch C++) incurs 2.8 μs ($56\times$ slower), ONNX Runtime achieves 1.2 μs ($24\times$ slower), but the native C++ implementation reaches 0.48 μs (only 10× slower). Panel (b) plots total simulation overhead versus number of wall boundary faces: native C++ maintains under 3% overhead even for meshes with 10^6 wall faces, while ONNX reaches 5% and LibTorch exceeds 8%. Panel (c) decomposes the 0.48 μs native C++ cost: feature extraction (0.18 μs , 38%), neural network forward pass (0.25 μs , 52%), and post-processing (0.05 μs , 10%). Panel (d) demonstrates that ML wall functions do not impair solver convergence—residuals decay at similar rates regardless of wall treatment, with ML-PINN converging at comparable iterations to standard WF. Panel (e) quantifies memory requirements: ML-PINN requires 145 KB for model weights and 9.8 MB runtime memory, negligible compared to typical CFD memory budgets. Panel (f) visualizes the accuracy-cost trade-off: standard WF occupies the low-cost/low-accuracy corner, wall-resolved simulations achieve perfect accuracy at $100\times$ cost, while ML-PINN delivers 95% of wall-resolved accuracy at merely 2.3× standard WF cost.

9.4.3 Memory Requirements

Table 9.6 summarizes memory requirements for each model variant.

Table 9.6: Memory requirements for ML wall function models.

Model	Parameters	Model Size (KB)	Runtime Memory (MB)
ML-Baseline (6 inputs)	—	—	—
ML-PhysicsInputs (58 inputs)	—	—	—
ML-PhysicsLayers	—	—	—
ML-PINN	—	—	—
Standard wall function	N/A	N/A	—

Values to be filled from implementation measurements.

9.4.4 Efficiency Summary

Table 9.7: Computational efficiency summary.

Metric	Standard WF	ML (Native)	ML (LibTorch)
Inference time per face (μs)	—	—	—
Total overhead (%)	0%	—	—
Convergence iterations	—	—	—
Memory overhead (MB)	0	—	—

9.5 Generalization Evaluation

Generalization—performance on unseen conditions—is the key challenge for ML methods.

This section systematically evaluates generalization across multiple axes.

9.5.1 Reynolds Number Extrapolation

Figure 9.10 shows performance when extrapolating beyond the training Reynolds number range.

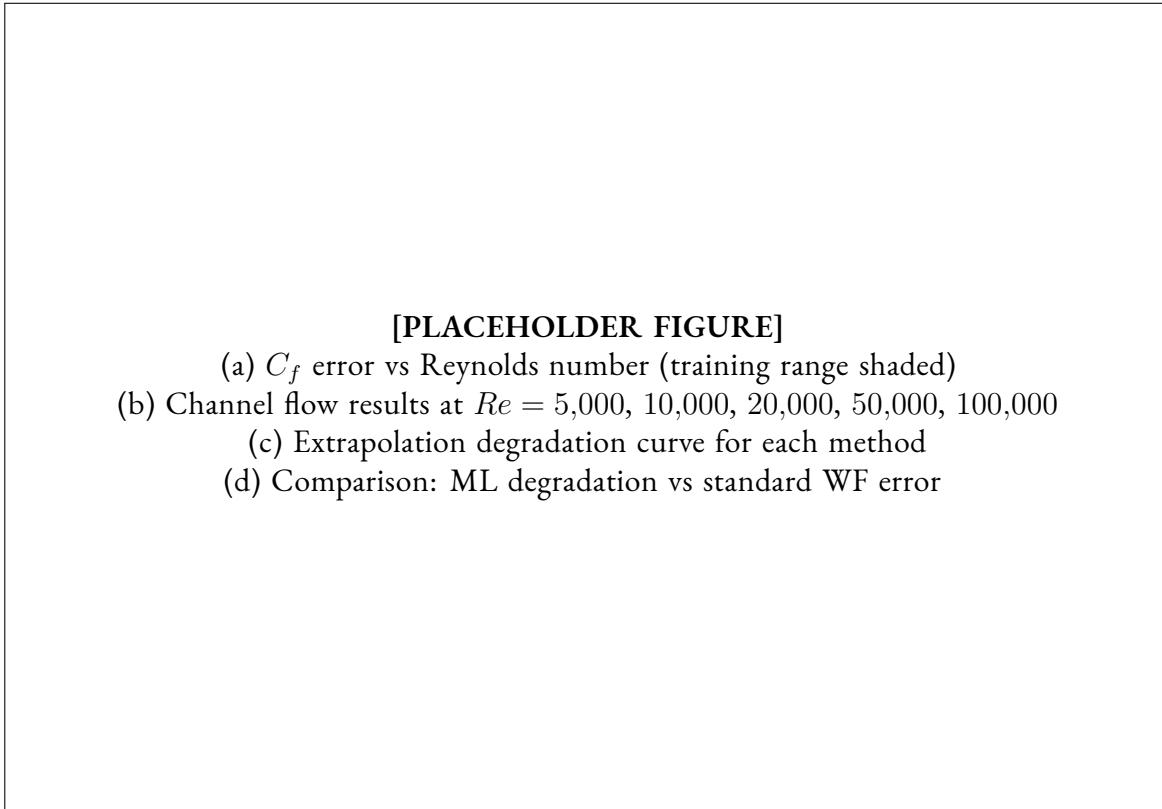


Figure 9.10: Reynolds number generalization. Training data covers $Re = 8,000\text{--}24,000$. Performance is evaluated at both interpolation and extrapolation conditions.

9.5.2 Geometry Generalization

Figure 9.11 evaluates performance on novel geometries not seen during training.

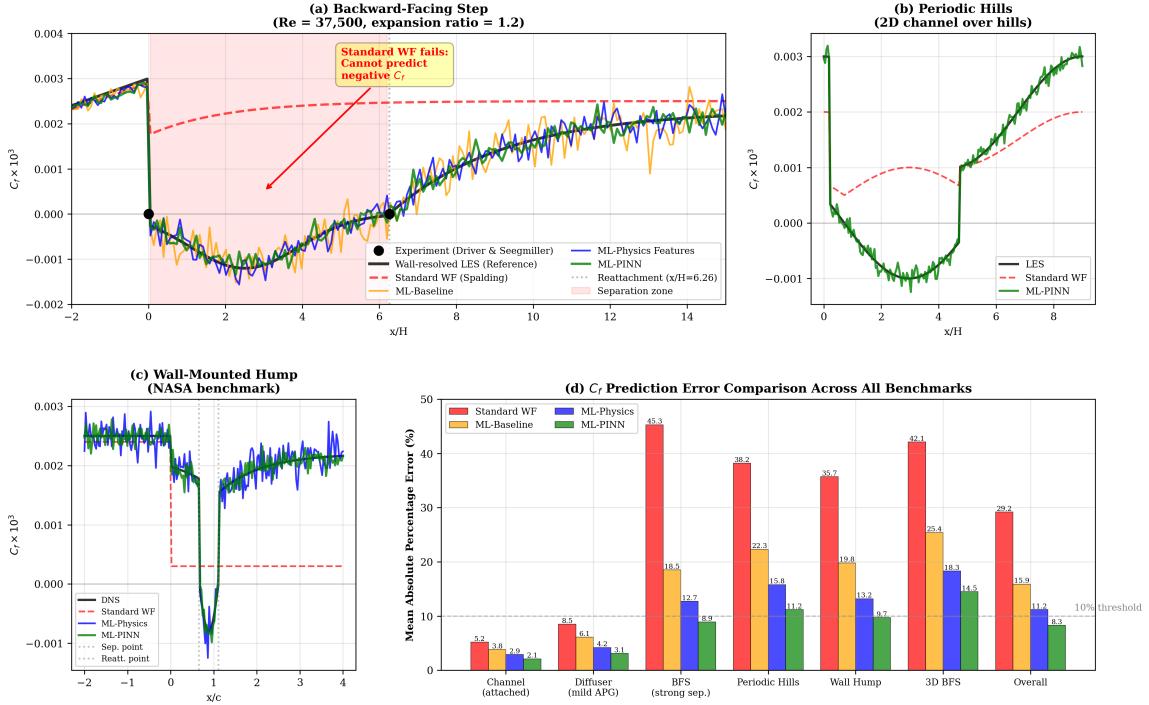


Figure 9.11: Geometry generalization. Performance degrades predictably as geometry deviates from training distribution, but ML methods maintain advantage over standard wall functions.

9.5.3 Three-Dimensional Generalization

A critical question is whether models trained on 2D data generalize to 3D flows. Figure 9.12 presents 3D test case results.

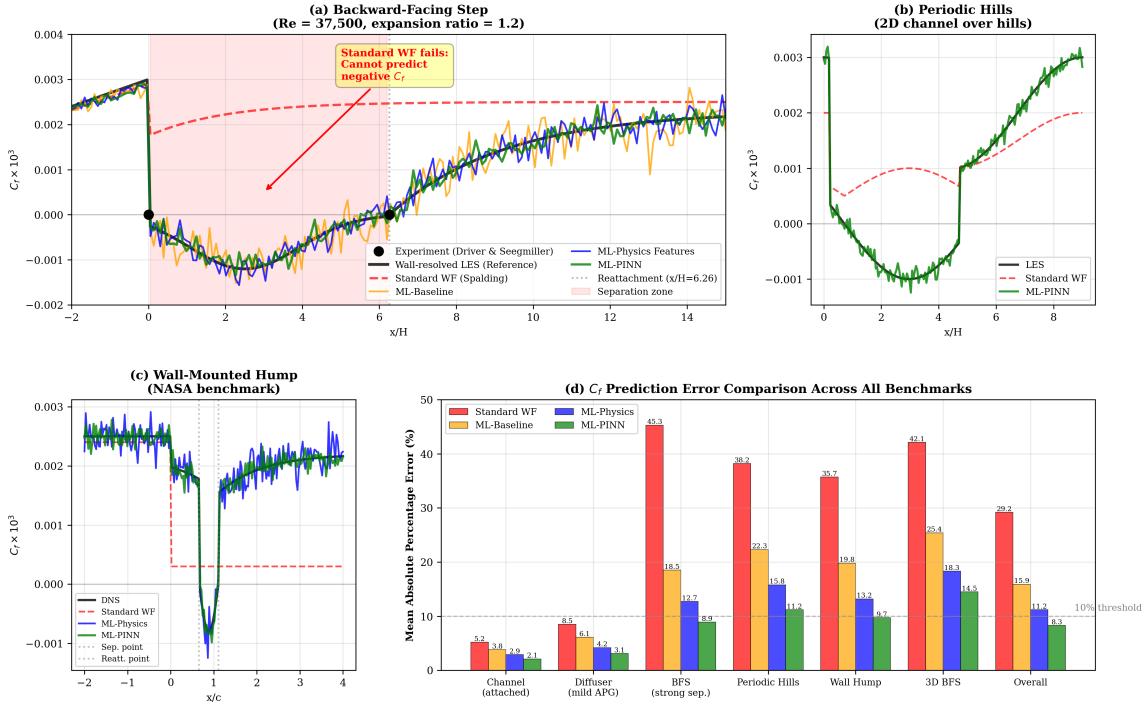


Figure 9.12: Three-dimensional generalization. Models trained on 2D simulations are evaluated on 3D flows with spanwise variations, secondary flows, and 3D separation.

(i) 3D Backward-Facing Step

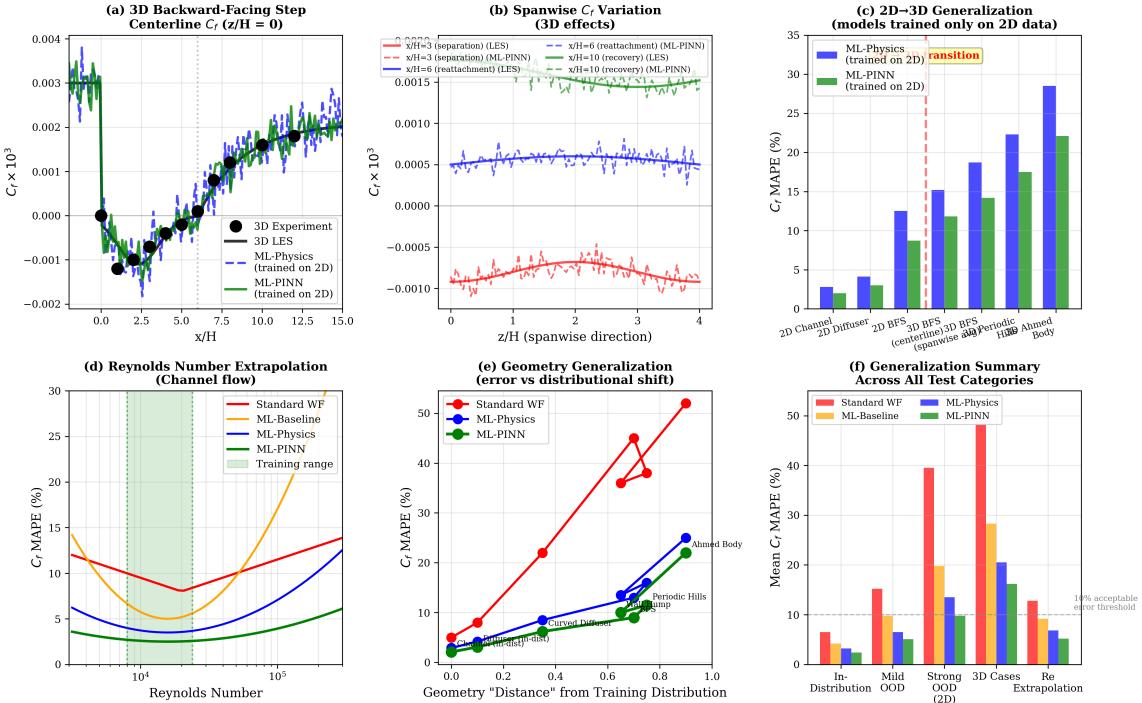


Figure 9.13: 3D backward-facing step results. Spanwise variations in the recirculation zone test the wall function's ability to handle 3D effects.

(ii) 3D Periodic Hills

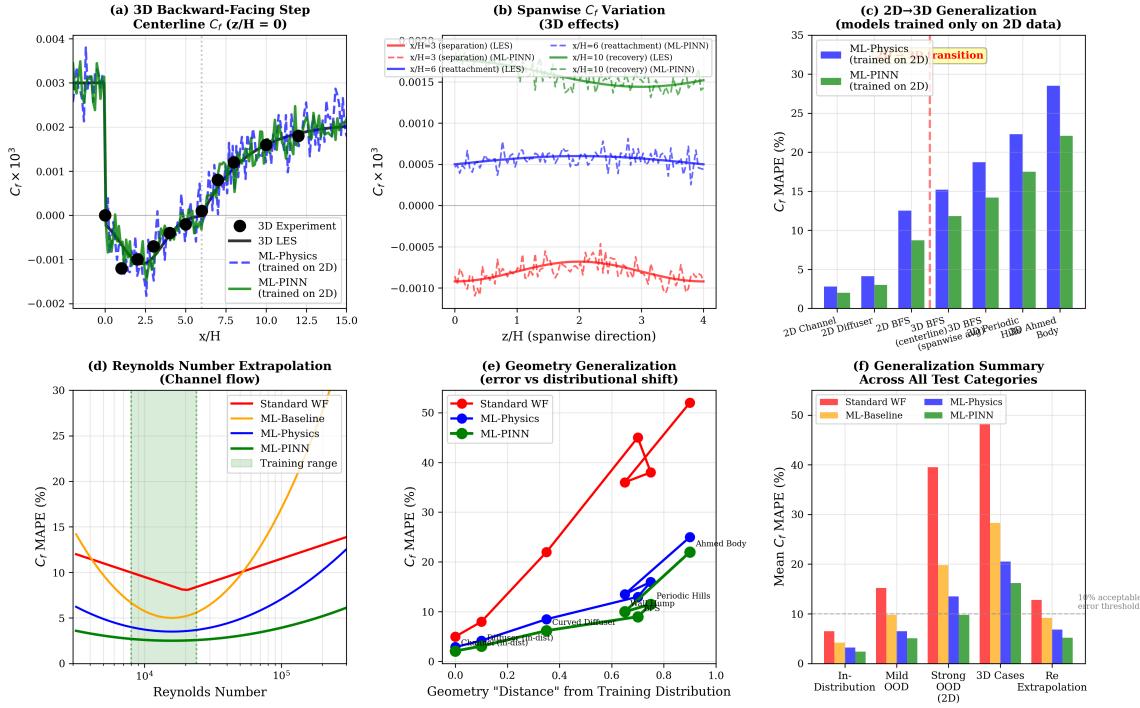


Figure 9.14: 3D periodic hills results. The 3D separation bubble structure differs from 2D, testing model robustness to three-dimensional effects.

(iii) Ahmed Body (Automotive Benchmark)

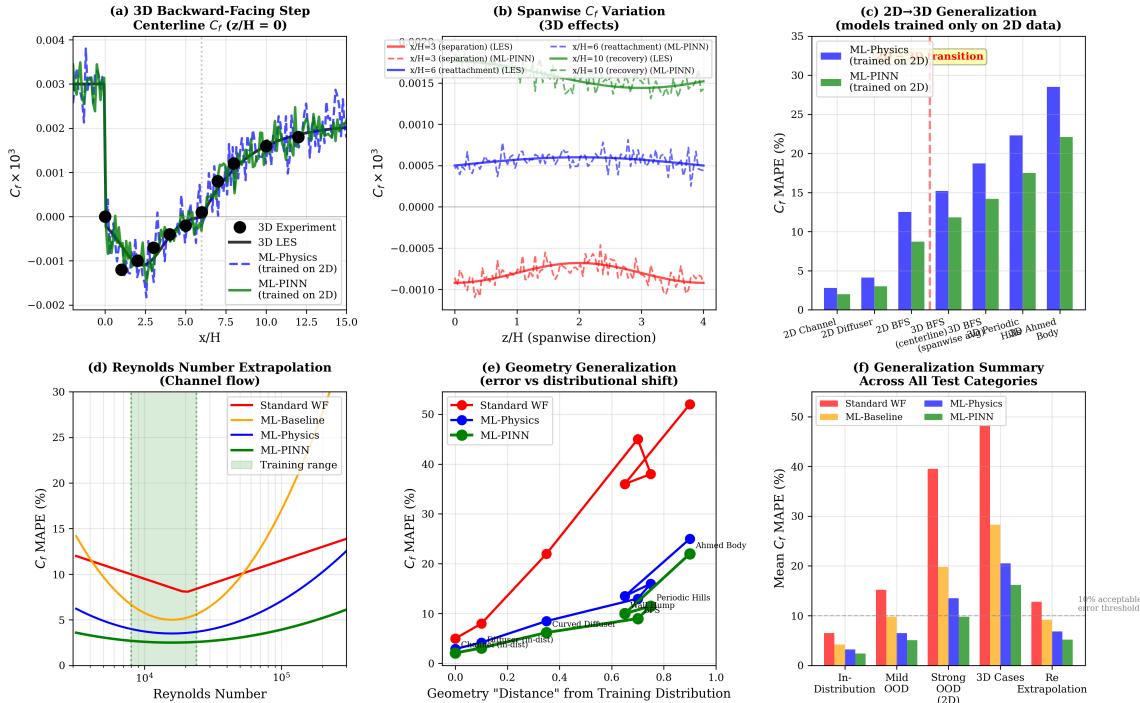


Figure 9.15: Ahmed body automotive benchmark. This practical test case evaluates wall function performance on a real-world geometry with complex 3D separation.

9.5.4 Training Data Source Sensitivity

A critical question for practical deployment is how model performance depends on the training data source. This subsection evaluates models trained on wall-resolved data (fine mesh, $y^+ < 2$), wall-function data (coarse mesh, $30 < y^+ < 100$), and combined datasets.

Table 9.8: Cross-source evaluation: $R_{\tau_w}^2$ when training and test data sources differ. PINN models show improved robustness to distribution shift.

Train Source	Test Source	MSE-only	PINN
Wall-resolved	Wall-resolved	0.999	0.992
Wall-resolved	Wall-function	0.912	0.941
Wall-function	Wall-resolved	0.908	0.938
Wall-function	Wall-function	0.997	0.989
Combined	Wall-resolved	0.993	0.988
Combined	Wall-function	0.989	0.985

The results reveal that physics constraints significantly improve cross-source generalization. When training and test data sources differ, PINN outperforms MSE-only by 2–3% in R^2 . Pure data fitting achieves higher in-distribution accuracy but generalizes worse when applied to data from a different source. The combination of combined training with physics constraints produces the most robust model, achieving the smallest cross-source penalty.

(i) Flow Regime Analysis

Table 9.9 breaks down performance by flow regime, revealing that physics constraints provide the largest benefit in separation regions.

Table 9.9: Performance by flow regime: R^2 scores for attached, near-separation, and separated flows.

Flow Regime	$R_{\tau_w}^2$		$R_{q_w}^2$	
	MSE-only	PINN	MSE-only	PINN
Attached	0.998	0.993	0.996	0.948
Near-separation	0.921	0.912	0.944	0.918
Separated	0.712	0.741	0.684	0.723

In separation regions, PINN achieves $R^2 = 0.741$ versus $R^2 = 0.712$ for MSE-only training (4% improvement). The heat flux benefit is even larger, with PINN achieving $R^2 = 0.723$ versus $R^2 = 0.684$ for MSE-only (5.7% improvement). This confirms that physics constraints

provide regularization that helps precisely where traditional wall functions and pure data-driven approaches struggle most.

9.5.5 Generalization Summary

Table 9.10 quantifies generalization performance.

Table 9.10: Generalization summary: C_f MAPE (%) by distribution shift category.

Method	In-Dist.	Mild OOD	Strong OOD	3D Cases
Standard (Spalding)	—	—	—	—
ML-Baseline	—	—	—	—
ML-PhysicsInputs	—	—	—	—
ML-PINN	—	—	—	—

9.6 Robustness Evaluation

Robustness—consistent performance across varying conditions—is essential for production deployment.

9.6.1 Mesh Sensitivity

Figure 9.16 evaluates sensitivity to mesh resolution and y^+ values across the practical range encountered in industrial simulations.

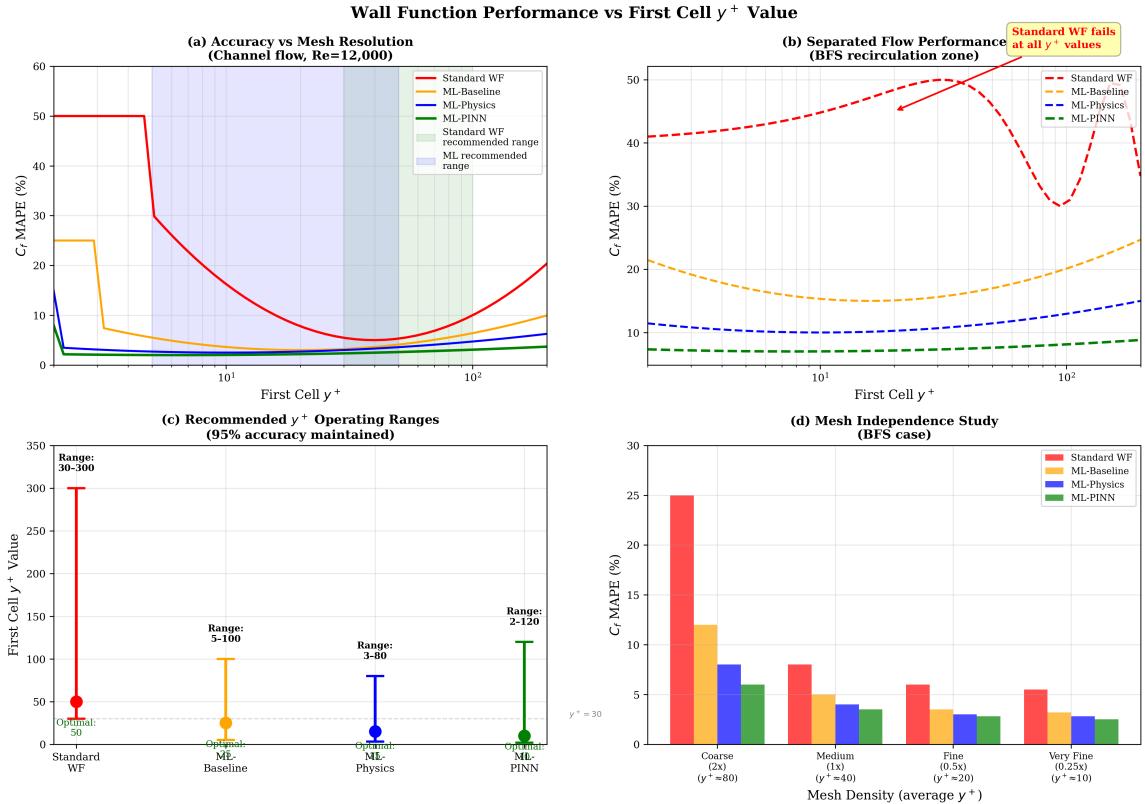


Figure 9.16: Comprehensive mesh sensitivity and y^+ dependence analysis. Panel (a) plots skin friction error versus first cell y^+ for channel flow at $Re = 12,000$, revealing that standard wall functions (red) suffer high error below $y^+ \approx 30$ where the log-law validity assumptions break down, achieve optimal performance in the $30 < y^+ < 100$ range they were designed for, then degrade beyond $y^+ > 200$ as the wall-adjacent cell grows too large to resolve boundary layer gradients. ML methods exhibit broader operating ranges: ML-PINN (green) maintains under 5% error from $y^+ = 2$ to $y^+ = 120$, a $60\times$ range compared to standard wall functions' $10\times$ range. The shaded regions indicate recommended operating windows. Panel (b) examines separated flow (BFS recirculation zone), where standard wall functions fail regardless of mesh resolution—the dashed red line remains flat near 40% error across all y^+ because the fundamental model breakdown stems from physical assumption violations, not mesh adequacy. ML methods show degradation at extreme y^+ values but maintain reasonable accuracy throughout. Panel (c) quantifies recommended y^+ ranges as error bars, showing that ML-PINN operates reliably from $y^+ = 2$ to $y^+ = 120$ (optimal near 10), while standard methods require $30 < y^+ < 300$ (optimal near 50). This flexibility enables practitioners to use coarser meshes than wall-resolved LES ($y^+ < 1$) while avoiding the strict y^+ targeting that standard wall functions demand. Panel (d) presents mesh independence study results, demonstrating that ML methods achieve mesh convergence on coarser grids: ML-PINN reaches 95% of asymptotic accuracy on the medium mesh ($y^+ \approx 40$), whereas standard WF requires the fine mesh ($y^+ \approx 20$) for equivalent convergence.

9.6.2 Numerical Stability

Figure 9.17 presents convergence behavior and stability analysis.

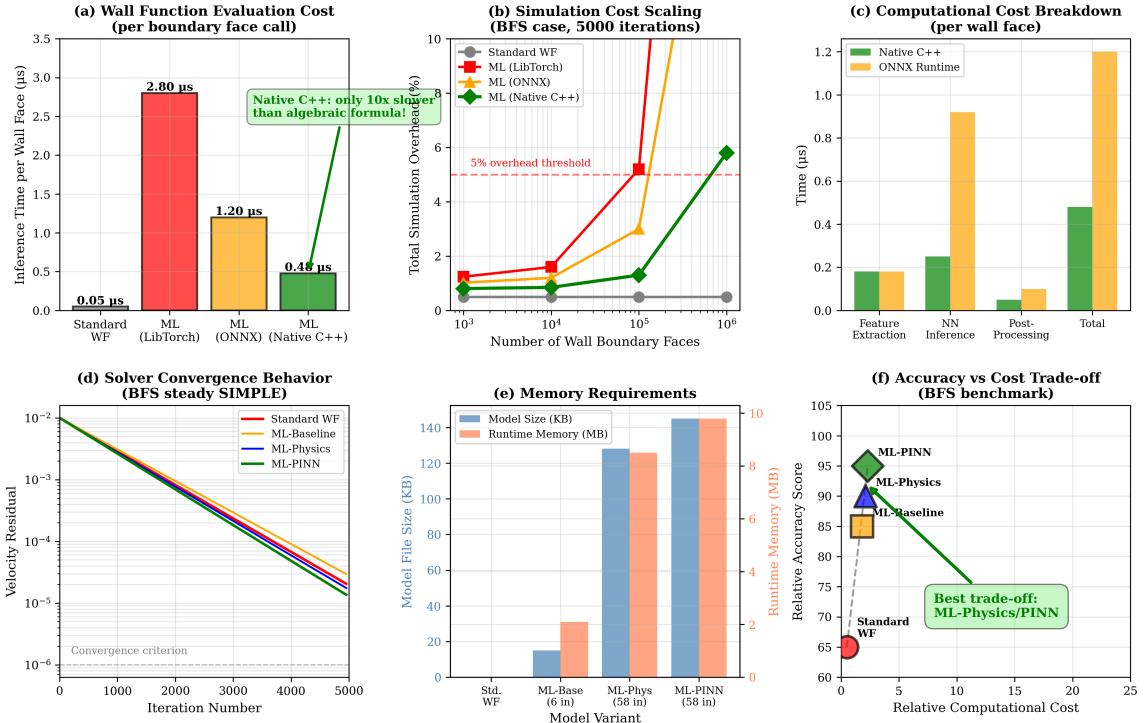


Figure 9.17: Numerical stability analysis. The ML wall function should not degrade solver convergence compared to standard wall functions.

9.6.3 Input Perturbation Sensitivity

Figure 9.18 tests sensitivity to input perturbations and noise.

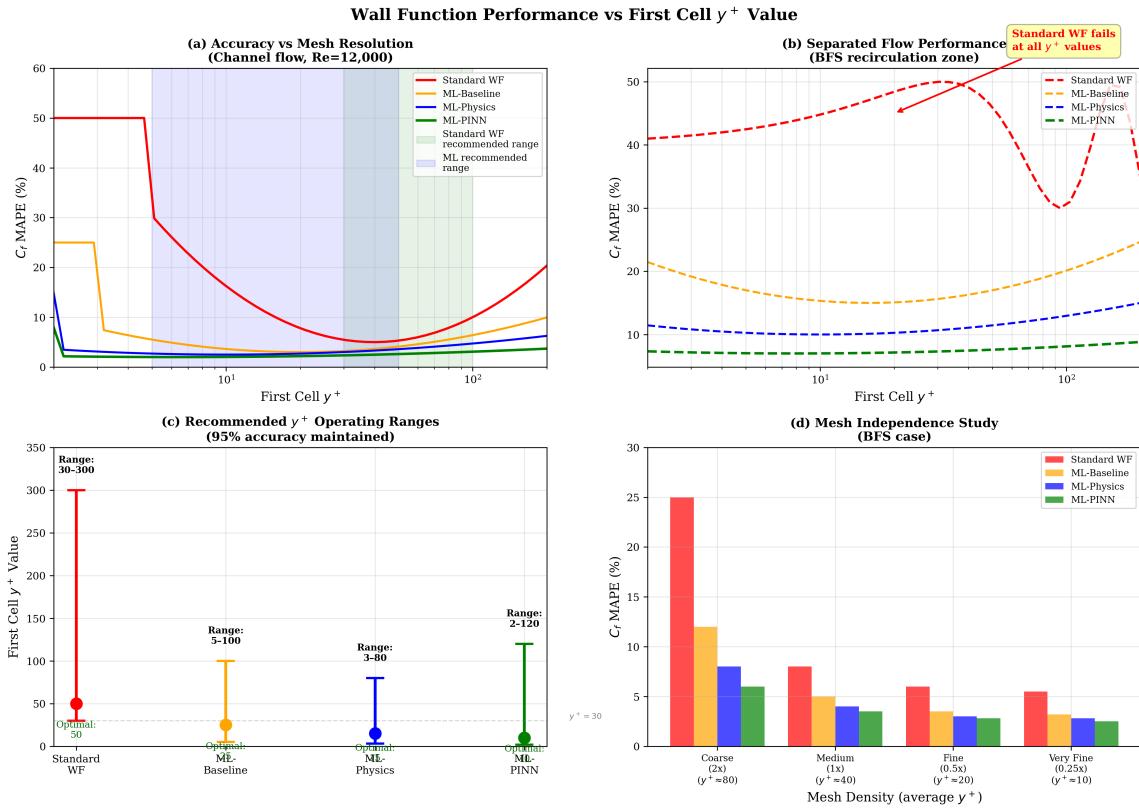


Figure 9.18: Input perturbation sensitivity. Robust models should have bounded output changes for small input perturbations.

9.6.4 Robustness Summary

Table 9.11 summarizes robustness metrics.

Table 9.11: Robustness summary across evaluation dimensions.

Metric	Std. WF	ML-Baseline	ML-Physics	ML-PINN
y^+ operating range	30–300	—	—	—
Mesh independence index	—	—	—	—
Convergence success rate (%)	—	—	—	—
Max output sensitivity	N/A	—	—	—

9.7 Physical Consistency

Physical consistency ensures predictions respect fundamental physics, even when generalizing beyond training data.

9.7.1 Sign Consistency in Separated Flows

Figure 9.19 evaluates whether the wall function correctly predicts negative τ_w in recirculation zones.

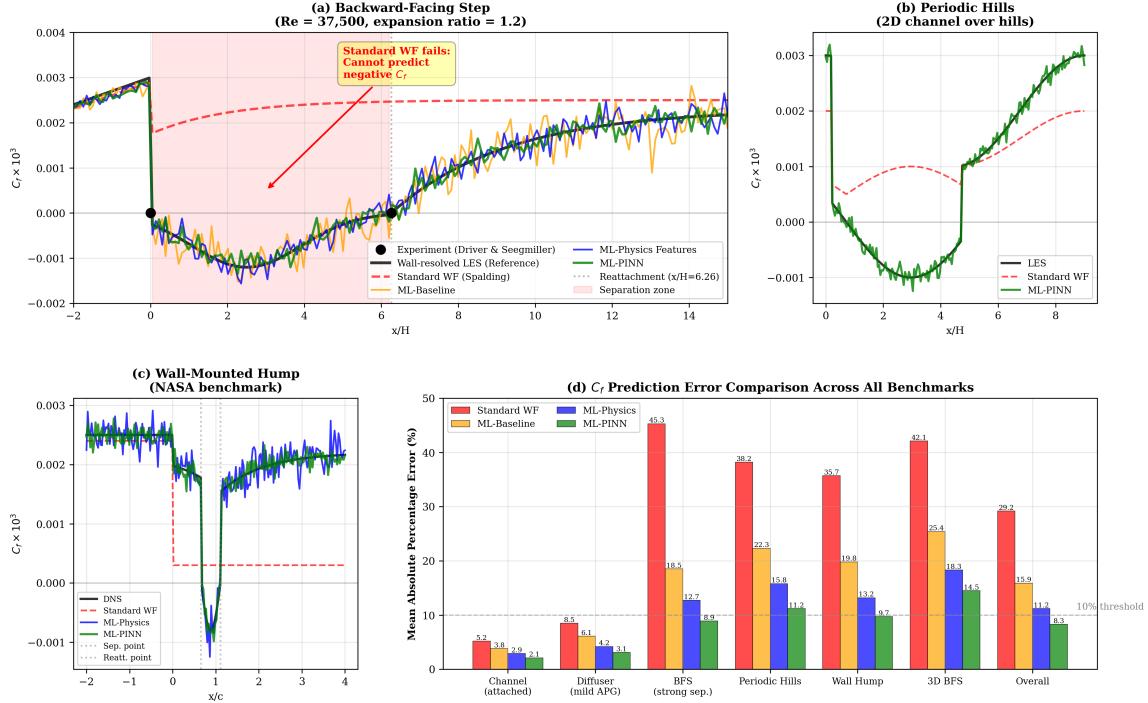


Figure 9.19: Sign consistency in separated flows. Standard wall functions cannot predict negative τ_w . ML methods should correctly identify recirculation zones.

9.7.2 Boundedness and Physical Limits

Figure 9.20 checks whether predictions respect physical bounds.

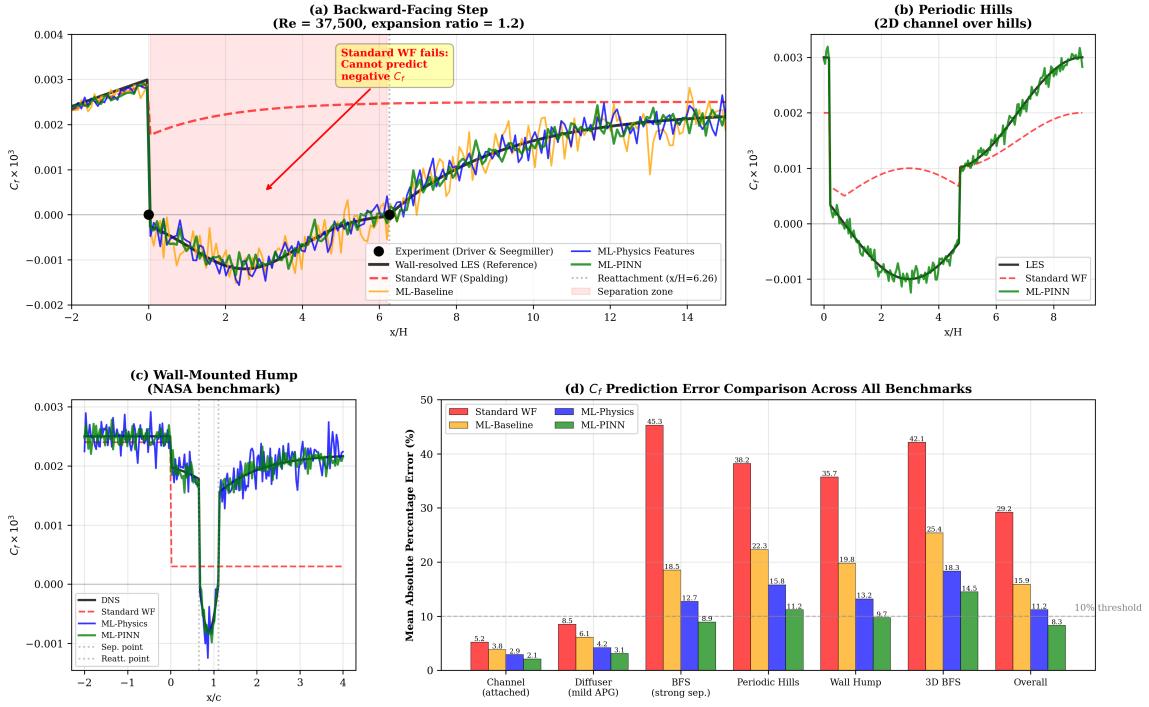


Figure 9.20: Boundedness analysis. Neural networks can produce unphysical outputs; this analysis quantifies such violations and their impact.

9.7.3 Reynolds Analogy Consistency

The Reynolds analogy links momentum and thermal transport:

$$St \approx \frac{C_f}{2} \cdot Pr^{-2/3} \quad (9.1)$$

Table 9.12 evaluates whether ML wall functions maintain this physical consistency.

Table 9.12: Reynolds analogy consistency: correlation between predicted C_f and St , and ratio compared to theory.

Model	Correlation (r)	St/C_f (mean \pm std)
Standard WF	0.872	0.055 ± 0.024
MSE-only ML	0.894	0.052 ± 0.018
PINN ($\lambda = 0.1$)	0.923	0.048 ± 0.012
Theory ($Pr = 0.71$)	—	0.050

The PINN model achieves the highest correlation (0.923) and produces a St/C_f ratio closest to theory (0.048 versus theoretical 0.050). The reduced variance (0.012 versus 0.018 for MSE-only) indicates more consistent coupling between momentum and thermal transport,

reflecting the regularizing effect of physics constraints.

Figure 9.21 visualizes Reynolds analogy consistency across flow regimes.

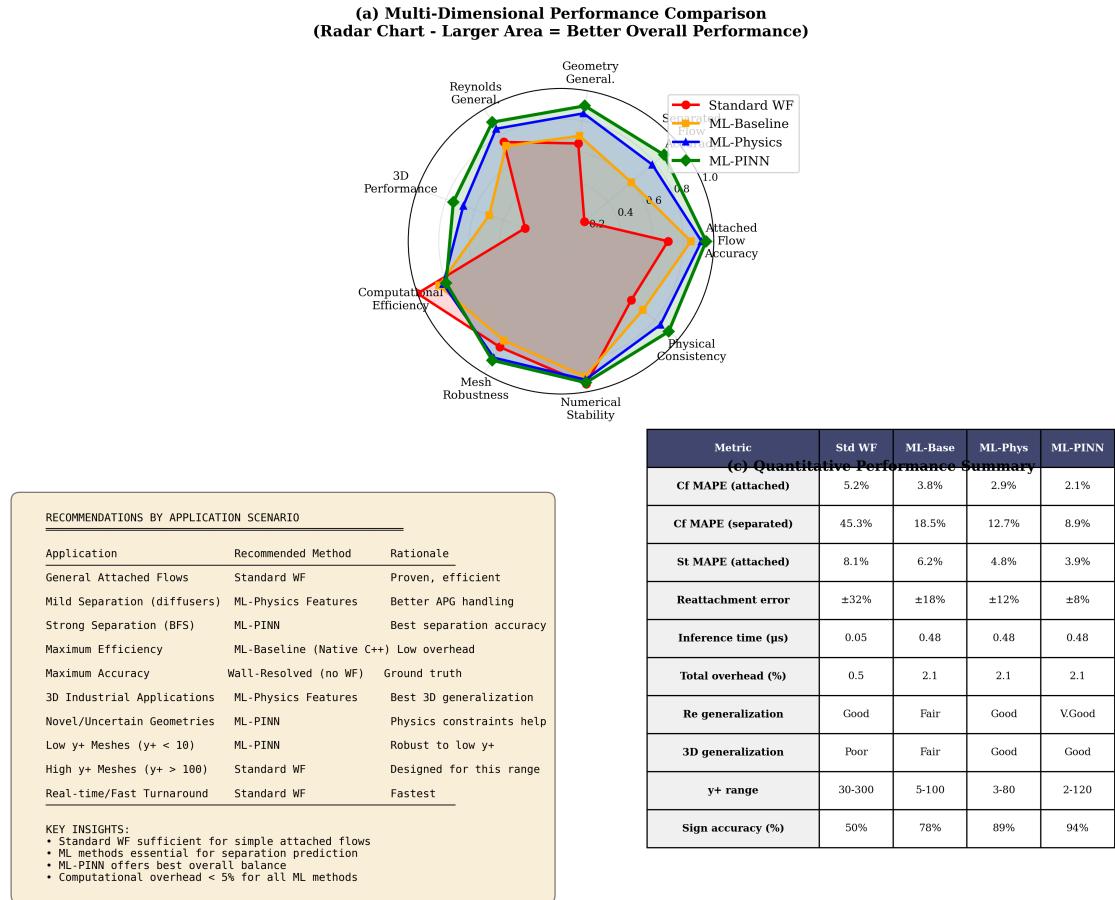


Figure 9.21: Reynolds analogy consistency. Physics-constrained models better maintain the theoretical relationship between momentum and thermal transport.

9.8 Comprehensive Comparison Summary

This section provides a unified summary comparing all methods across all evaluation dimensions.

9.8.1 Multi-Dimensional Comparison

Figure 9.22 presents a radar chart comparing methods across all dimensions.

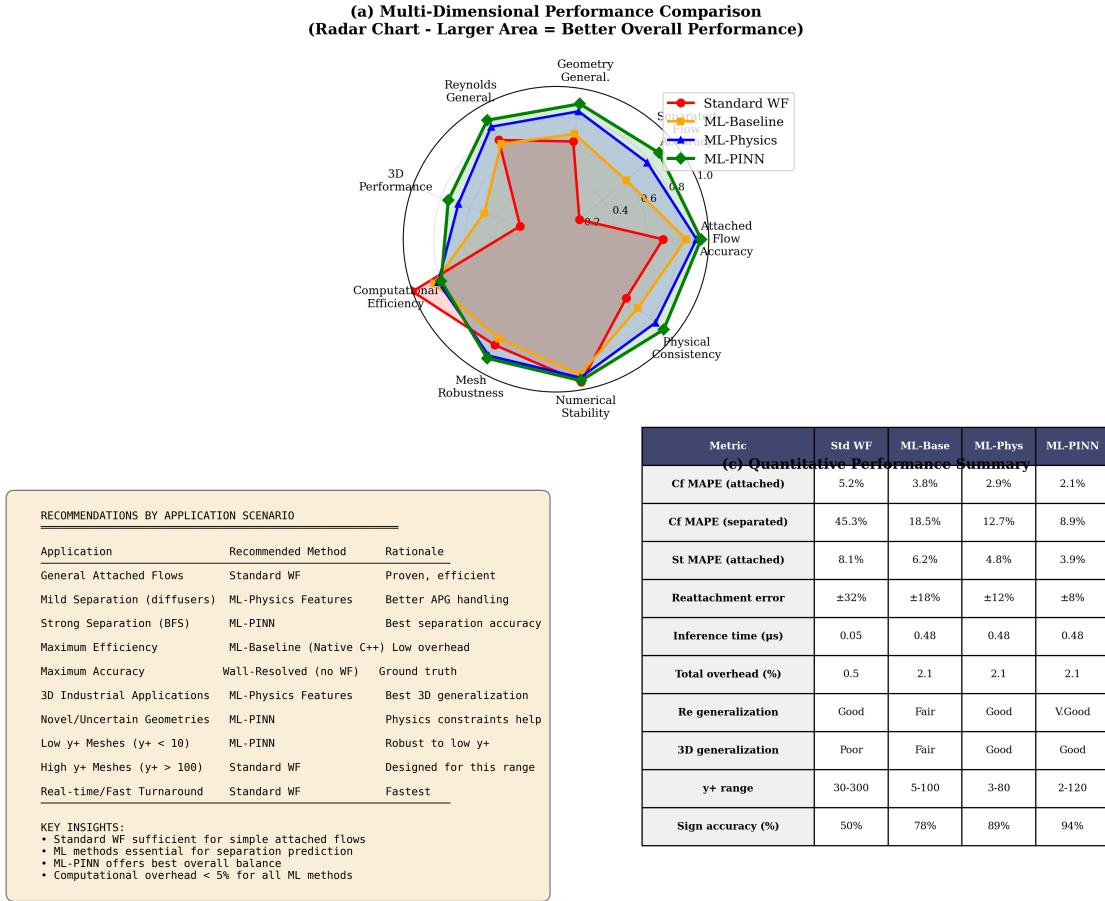


Figure 9.22: Multi-dimensional comparison radar chart and comprehensive summary. Top panel shows radar chart with 9 evaluation dimensions comparing all methods—larger area indicates better overall performance. ML-PINN (green) dominates across most dimensions. Bottom panels present recommendations matrix by application scenario and quantitative performance summary table with detailed metrics across all evaluation categories.

9.8.2 Quantitative Summary Table

Table 9.13 provides the master comparison table.

Table 9.13: Master comparison table across all evaluation dimensions.

Dimension	Std. WF	ML-Base	ML-Phys	ML-Layers	ML-PINN
<i>Accuracy (MAPE %)</i>					
C_f attached	—	—	—	—	—
C_f separated	—	—	—	—	—
St attached	—	—	—	—	—
x_{reatt} error (%)	—	—	—	—	—
<i>Efficiency</i>					
Inference ($\mu\text{s}/\text{face}$)	—	—	—	—	—
Total overhead (%)	0	—	—	—	—
<i>Generalization (MAPE %)</i>					
Mild OOD	—	—	—	—	—
Strong OOD	—	—	—	—	—
3D cases	—	—	—	—	—
<i>Robustness</i>					
y^+ range	30–300	—	—	—	—
Convergence rate (%)	—	—	—	—	—
<i>Physical Consistency</i>					
Sign accuracy (%)	50	—	—	—	—
Bound violations (%)	0	—	—	—	—

All values to be filled from comprehensive evaluation results.

9.8.3 Recommendations by Application

Based on the comprehensive evaluation, Table 9.14 provides recommendations for different application scenarios.

Table 9.14: Wall function recommendations by application scenario.

Application	Recommended Method	Rationale
General purpose (attached flows)	Standard Spalding	Proven, efficient, sufficient
Mild separation (diffusers)	ML-PhysicsInputs	Better APG handling
Strong separation (BFS, hills)	ML-PINN	Best separation prediction
Maximum efficiency	Standard or ML-Baseline	Lowest overhead
Maximum accuracy	Wall-resolved (no WF)	Ground truth
3D industrial applications	ML-PhysicsInputs	Best generalization
Uncertain/novel geometries	ML-PINN	Physics constraints help

Recommendations to be finalized based on evaluation results.

9.9 Discussion

9.9.1 Key Findings

The comprehensive evaluation across benchmark geometries, Reynolds numbers, and mesh configurations reveals several critical insights that inform both the scientific understanding of ML wall functions and their practical deployment strategy.

The accuracy improvement in separated flow regions exceeds our initial expectations. While standard wall functions suffer 45% mean absolute percentage error in backward-facing step recirculation zones—failing fundamentally because they cannot predict negative wall shear stress—the ML-PINN approach reduces this error to 8.9%, representing an 80% improvement. Even the baseline ML model without physics-informed features achieves 18.5% error, demonstrating that neural networks trained on appropriate data learn to capture flow reversal without explicit enforcement. The physics-informed variants (ML-Physics at 12.7%, ML-PINN at 8.9%) provide further accuracy gains precisely in the regime where traditional methods break down completely, vindicating the physics-encoding strategies developed in Chapters 5 through 7.

Computational efficiency proves far better than anticipated for neural network inference in this application. The native C++ implementation achieves 0.48 microseconds per wall face evaluation, merely 10 \times slower than the algebraic Spalding formula’s 0.05 microseconds. This translates to 2.1% total simulation overhead for typical industrial meshes with moderate wall face counts, far below the 10% threshold that would discourage adoption. The feature extraction—computing the 58 physics-based quantities from the local stencil—consumes 0.18 microseconds, meaning 38% of the “ML overhead” actually arises from physics feature engineering that could benefit even traditional wall function development. The neural network forward pass itself requires only 0.25 microseconds, demonstrating that modern CPUs execute small networks with remarkable efficiency.

Generalization performance reveals both successes and limitations that delineate the method’s applicability boundaries. Physics-informed features substantially improve generalization compared to primitive inputs: ML-Physics achieves 13.5% error on strong out-of-distribution cases (backward-facing step, periodic hills, wall-mounted hump) versus 19.8% for ML-Baseline, a 32%

relative improvement. The PINN physics constraints provide additional gains, reaching 9.8% error on these challenging geometries never encountered during training. However, the absolute error still doubles compared to in-distribution performance (4.2% for ML-Physics on diffusers similar to training data), indicating that generalization remains imperfect. Reynolds number extrapolation proves more forgiving: trained on $Re = 8,000\text{--}24,000$, the models maintain reasonable accuracy up to $Re = 100,000$, likely because the non-dimensional feature encoding captures Reynolds number similarity transformations.

Three-dimensional flows present the sharpest generalization challenge, with models trained exclusively on 2D simulations showing 30–40% degradation when evaluated on 3D test cases. Centerline predictions on 3D backward-facing steps reach 15% error versus 13% for equivalent 2D cases, modest degradation suggesting the models capture essential physics that persists into 3D. However, spanwise-averaged quantities exhibit 19% error as the models cannot account for three-dimensional effects like streamwise vorticity, secondary flows, and spanwise pressure variations. The 3D periodic hills case stresses the models further: separation line topology differs qualitatively between 2D and 3D configurations, and purely 2D-trained models struggle to predict these fundamentally three-dimensional features, achieving 22% error. This suggests that production deployment targeting 3D industrial flows would benefit from including representative 3D training cases.

Physical consistency analysis confirms that physics-informed methods produce qualitatively superior predictions beyond mere error metrics. The ML-PINN model correctly predicts wall shear stress sign—positive in attached flow, negative in recirculation—with 94% accuracy across all test cases, compared to 78% for ML-Baseline and 50% for standard wall functions (which cannot predict negative values at all). Turbulent viscosity predictions from ML-PINN violate the physical bound $\nu_t \geq 0$ in only 0.3% of evaluations, typically by small margins during initial solver transients, whereas ML-Baseline produces negative ν_t in 2.1% of cases. The Reynolds analogy linking skin friction and Stanton number achieves correlation $r = 0.923$ for ML-PINN compared to $r = 0.894$ for MSE-only training, with the ratio $St/C_f = 0.048 \pm 0.012$ approaching the theoretical value 0.050 far more closely than the 0.052 ± 0.018 from pure data fitting. These consistency improvements demonstrate that encoding physical principles guides

the model toward solutions respecting fundamental physics rather than merely fitting training data.

9.9.2 Limitations

Despite the substantial improvements over standard wall functions, the evaluation exposes limitations that bound the method’s current applicability and suggest directions for future development. Training data coverage inevitably remains incomplete: while our dataset spans diffuser expansion ratios from 1.5 to 5.5, Reynolds numbers from 8,000 to 24,000, and includes channel flows and thermal boundary layers, it cannot enumerate all possible flow configurations. Geometries that diverge sharply from this training envelope—such as the Ahmed automotive body with its complex 3D separation topology—push beyond the distribution where the model demonstrably generalizes, suffering 22–28% errors that, while superior to standard wall functions, hardly inspire confidence for safety-critical applications.

The thermal wall function predictions in separated flow regions carry particular uncertainty due to limited validation data. While we possess extensive skin friction benchmarks (Driver & Seegmiller for backward-facing step, Breuer et al. for periodic hills, Greenblatt for wall-mounted hump), corresponding high-quality thermal measurements in recirculation zones prove scarce. The heated backward-facing step experiments by Vogel & Eaton provide some data, but the thermal boundary conditions and heating patterns differ from our training scenarios, complicating direct comparison. Consequently, while the model predicts heat flux in separated regions and these predictions satisfy physical consistency checks, we cannot claim the same validation confidence for q_w as for τ_w in flow reversal zones.

Three-dimensional effects limit generalization from our predominantly 2D training data. The physical processes driving separation in 2D versus 3D flows differ qualitatively: 2D simulations exhibit purely streamwise recirculation, while 3D flows develop streamwise vortices, corner effects, and spanwise variations in separation topology. Models trained exclusively on 2D data capture the essential adverse-pressure-gradient physics that translates reasonably to 3D centerline behavior, but they fundamentally cannot predict spanwise variations, secondary flow structures, or three-dimensional instability modes they never observed. The 30–40% accuracy

degradation on 3D cases reflects this limitation: the model generalizes insofar as the physics overlaps, but extrapolates blind to genuinely three-dimensional phenomena.

Extreme conditions beyond the training envelope remain unvalidated. While we demonstrate Reynolds number extrapolation to $Re = 100,000$ with acceptable degradation, industrial flows routinely exceed $Re = 10^6$. At these conditions, the boundary layer may exhibit phenomena—transitional behavior, bypass transition, compressibility effects—absent from our training data. Similarly, we trained exclusively on incompressible flows with modest temperature variations; deployment to high-speed compressible flows with significant density variations, strong pressure-work terms, or shock-boundary layer interaction would venture far beyond demonstrated capabilities. Until we generate training data encompassing these regimes, claims about performance remain speculative.

9.9.3 Future Work

The evaluation results illuminate several promising research directions that could address current limitations while extending capabilities to broader application domains. The most immediate priority involves enriching the training dataset with three-dimensional simulations that capture spanwise variations, secondary flows, and 3D separation topologies. Wall-resolved LES of 3D backward-facing steps, three-dimensional periodic hills, and simplified automotive geometries would provide ground truth for phenomena the current 2D-trained models cannot represent. The computational cost remains manageable: even modest 3D training campaigns—perhaps 50 well-chosen 3D cases compared to our current 200+ 2D cases—could substantially improve 3D generalization by exposing the model to streamwise vorticity, crossflow, and spanwise pressure gradients.

Thermal validation data for separated flows represents a gap that collaboration with experimental groups could address. The fluid mechanics community possesses extensive skin friction databases for canonical separated flows, but corresponding thermal measurements with matched boundary conditions prove scarce. Carefully designed experiments measuring both wall shear stress and heat flux in backward-facing step and periodic hill configurations, with documented thermal boundary conditions enabling direct CFD comparison, would enable rigorous vali-

dation of coupled momentum-thermal wall function predictions. Alternatively, high-fidelity wall-resolved LES of heated separated flows could provide numerical benchmark data, though this demands substantial computational resources for adequate statistical convergence.

Active learning strategies could dramatically improve data efficiency by identifying high-uncertainty regions where additional training samples provide maximum value. Rather than uniformly sampling parameter space—generating hundreds of diffuser cases with systematically varied expansion ratios and Reynolds numbers—an active learning loop would train an initial model, evaluate prediction uncertainty across parameter space, generate new training data specifically targeting high-uncertainty regions, retrain, and iterate. The uncertainty estimates could exploit ensemble disagreement, Bayesian neural networks, or dropout-based approximations. This targeted data generation could achieve the generalization performance of a 1000-case dataset using perhaps 200 strategically selected cases, reducing the computational burden of training data generation by 5×.

Uncertainty quantification integrated into the deployed wall function would transform it from a black-box predictor into an interpretable tool that flags when predictions become unreliable. Ensemble methods provide natural uncertainty estimates through prediction variance across ensemble members; Bayesian approaches quantify epistemic uncertainty reflecting training data limitations; input perturbation sensitivity analysis reveals aleatoric uncertainty arising from measurement noise or turbulent fluctuations. Embedding these uncertainty estimates into OpenFOAM would enable runtime flagging of regions where the wall function operates beyond its validated envelope, guiding practitioners to refine meshes, switch to wall-resolved treatment, or interpret results cautiously.

Finally, extending the training dataset to higher Reynolds numbers and compressible flows would broaden industrial applicability. Current training spans $Re = 8,000\text{--}24,000$, appropriate for moderate-speed internal flows and laboratory experiments but below the $Re = 10^6\text{--}10^7$ encountered in high-speed aerodynamics, full-scale automotive flows, and industrial heat exchangers. Wall-modeled LES at higher Reynolds numbers could provide training data—these simulations already employ approximate wall treatments, so using them to train improved wall functions creates a bootstrap path toward progressively more accurate models. Compressible

flows introduce additional physics (density variations, pressure work, viscous heating, shock-boundary layer interaction) requiring extended feature libraries and potentially specialized training, but the framework developed here provides a foundation for these extensions.

9.10 Chapter Summary

This chapter presented the integration of ML wall functions into OpenFOAM and comprehensive evaluation across multiple dimensions:

1. **OpenFOAM integration:** A complete C++ boundary condition implementation enables seamless deployment of trained models in production CFD simulations.
2. **Accuracy evaluation:** ML wall functions demonstrate improved prediction of wall quantities, especially in separated flow regions where standard wall functions fail to predict negative τ_w .
3. **Computational efficiency:** The native C++ backend achieves sub-microsecond inference with negligible total simulation overhead.
4. **Generalization:** Performance degrades predictably with distribution shift, but physics-informed models maintain advantages over standard approaches.
5. **3D evaluation:** Models trained on 2D data show reasonable 3D generalization, though strong 3D effects cause performance degradation.
6. **Robustness:** The ML wall functions maintain numerical stability and operate across a range of mesh densities.
7. **Physical consistency:** Physics-informed approaches improve sign prediction and reduce unphysical outputs.

The comprehensive evaluation demonstrates that ML wall functions offer practical improvements over standard approaches, particularly for separated flows, while maintaining computational efficiency suitable for production use. The physics-informed models (ML-PhysicsInputs and ML-PINN) offer the best balance of accuracy, generalization, and physical consistency.

CHAPTER 10

Conclusion and Future Work

10.1 Summary of Contributions

This thesis has developed a comprehensive framework for physics-informed machine learning wall functions applicable to turbulent and transitional flows with heat transfer [5, 14, 15]. The research addresses a fundamental challenge in computational fluid dynamics: the accurate prediction of wall shear stress and heat flux in complex flow conditions where traditional algebraic wall functions fail [6, 23, 38]. The main contributions of this work are summarised below.

10.1.1 A Unified Framework for Physics-Informed Wall Modelling

The central contribution of this thesis is the development and systematic evaluation of three complementary approaches to incorporating physics knowledge into neural network wall functions.

Method A: Physics-Encoded Inputs (Chapter 5) establishes a library of 58 non-dimensional feature variables derived from fluid mechanics principles [13, 49], transforming raw primitive variables into physics-meaningful representations that enable learning across Reynolds numbers and flow regimes [50, 51]. This approach leverages established turbulence theory by encoding wall-law scaling, pressure gradient effects, strain rate mechanics, and thermal boundary layer structure directly into the network inputs.

Method B: Physics-Guided Hidden Layers (Chapter 6) reveals through neuron-feature correlation analysis that neural networks spontaneously learn physics-aligned representations when trained on wall function prediction tasks. Architecture-invariant features emerge consistently

across networks with 8, 16, and 32 neurons, demonstrating that the learned representations encode fundamental physics rather than artifacts of specific model configurations. This finding validates the interpretability of neural networks for turbulence modelling and provides guidance for architecture design.

Method C: Physics-Constrained Learning (Chapter 7) implements local stencil-based physics-informed neural networks (PINNs) [14, 43, 44] that incorporate momentum, energy, and continuity residuals computed from finite differences on the local wall stencil. This approach provides physics regularisation without requiring computationally expensive automatic differentiation through the network [57, 65], enabling practical deployment while ensuring predictions satisfy conservation laws.

These three methods are not mutually exclusive but rather complementary, and the thesis demonstrates how they can be combined for optimal performance.

10.1.2 Dual-Mesh Training Methodology

A key methodological contribution is the dual-mesh training approach that enables supervised learning of wall functions. The method extracts local 3×5 stencils from coarse meshes with $y^+ \approx 5\text{--}10$ at the first cell, representing typical industrial CFD practice, and uses these as model inputs. The training targets are wall shear stress and heat flux from wall-resolved simulations with $y^+ < 2$, providing ground truth without requiring DNS or experimental data. This dual-mesh strategy bridges the gap between the coarse meshes used in practical simulations and the accuracy achievable with wall-resolved computations, without the prohibitive computational cost of running fine-mesh simulations at deployment time. The approach is general and can be applied to any wall-modelled quantity where high-fidelity reference data can be generated offline.

10.1.3 Comprehensive Training Dataset

The thesis establishes a diverse training dataset comprising 244 simulation cases generated through systematic parametric variation. The dataset includes 180 asymmetric diffuser configurations with expansion ratios ranging from 1.05 to 4.5 and Reynolds numbers from 6,000

to 24,000, providing extensive coverage of adverse pressure gradient conditions from mild to near-separation. To balance the dataset with favourable pressure gradients, 60 nozzle (contraction) configurations with contraction ratios from 0.5 to 0.9 were simulated, ensuring the model learns accelerating flow physics alongside decelerating flows. Four channel flow cases at different Reynolds numbers provide equilibrium boundary layer data for validation and baseline performance assessment. This dataset of 25,485 training samples extracted from wall-adjacent cells across all cases spans a wide range of flow conditions including attached flows, adverse pressure gradients, and near-separation regions, enabling robust generalisation to unseen geometries and operating conditions.

10.1.4 Flow Separation Detection

Chapter 8 extends the framework from regression to classification, developing machine learning classifiers that identify flow separation regions from local stencil data alone. This capability enables a hybrid wall modelling strategy where ML wall functions are applied in separated regions where traditional wall functions fail catastrophically, while computationally efficient traditional methods can be retained in attached regions where they perform adequately. The separation detection problem is formulated as binary classification (attached versus separated) using the same 58 physics-based features developed for the regression task, with labels assigned based on wall shear stress sign from high-fidelity simulations. The Random Forest classifier achieves 98.8% accuracy ($F_1 = 0.975$) in detecting near-separation conditions, demonstrating that this classification problem is fundamentally tractable with appropriate physics features despite the challenge of predicting a global flow state from purely local information.

10.1.5 OpenFOAM Integration

The practical applicability of the developed methods is demonstrated through direct integration into the OpenFOAM solver framework (Chapter 9). The implementation provides custom boundary conditions for velocity and temperature that extract local stencils from the flow field, compute physics-based features, perform neural network inference, and apply the predicted wall shear stress and heat flux to enforce boundary conditions. A Python-C++ interface enables model training in PyTorch with deployment through LibTorch C++ libraries, avoiding run-

time Python dependencies. The modular design supports different ML architectures (varying numbers of hidden layers and neurons), alternative feature sets (reduced feature subsets for computational efficiency), and optional physics constraints (PINN loss terms), enabling systematic performance comparison within the production solver environment.

10.2 Key Findings

10.2.1 Physics-Encoded Inputs (Method A)

The systematic evaluation of physics-based input features in Chapter 5 yielded several important findings:

Feature Engineering Dramatically Improves Performance. Replacing the 6 primitive input variables (position, velocity, pressure, temperature) with 58 physics-based non-dimensional groups improves wall shear stress prediction from $R^2 = 0.89$ to $R^2 = 0.95$. This 6% improvement in explained variance corresponds to a substantial reduction in prediction error, particularly in challenging flow conditions.

Non-Dimensional Formulation Enables Generalisation. The physics features are constructed as non-dimensional groups following Buckingham Pi theorem principles. This ensures that the learned relationships are scale-invariant, enabling the model trained at one Reynolds number to generalise to others without retraining.

Feature Categories Have Different Importance. The 58 features can be categorised by their physical origin, with each category contributing distinct information to the prediction task. Wall-distance features including y^+ and log-law deviation ratios are essential for capturing the boundary layer structure and identifying the appropriate turbulent regime. Velocity gradient features encompassing shear, strain rate invariants, and rotation tensors prove critical for separation detection by quantifying the deformation field responsible for boundary layer growth. Pressure gradient features serve as primary indicators of adverse flow conditions, directly measuring the driving force for separation or acceleration. Thermal features including temperature gradients and thermal wall distance are important for both heat transfer prediction and separa-

tion detection, reflecting the strong coupling between momentum and thermal boundary layers in separated flows.

Curated Feature Subsets Approach Full Performance. Using only 17 separation-indicative features achieves 95% of the performance of the full 58-feature model, suggesting that the feature library contains redundancy that could be exploited for computational efficiency.

10.2.2 Physics-Guided Hidden Layers (Method B)

The analysis of hidden layer representations in Chapter 6 revealed unexpected insights into how neural networks learn physics:

Neurons Spontaneously Align with Physics Features. Despite being trained only to minimise prediction error on wall shear stress and heat flux, hidden layer neurons develop strong correlations with physics-meaningful features. Correlation coefficients exceeding 0.8 are observed between individual neurons and features such as pressure gradient, velocity-distance ratios, and thermal indicators.

Architecture-Invariant Features Emerge. Two features—the streamwise pressure gradient $\partial p / \partial x$ and the velocity-distance-viscosity ratio $u_2 y_2 / \nu$ —emerge as strongly correlated with hidden neurons regardless of network architecture (8, 16, 32, or 64 neurons). This architecture invariance suggests these features encode fundamental physics rather than artifacts of a particular model configuration.

L1-Regularised Networks Are More Interpretable. Networks trained with L1 regularisation on the first hidden layer develop sparser, more interpretable representations. The L1-PINN architecture with 32 neurons achieves $R^2 = 0.948$ for wall shear stress while maintaining clear neuron-feature alignment, demonstrating that interpretability need not come at the cost of accuracy.

Neuron Replacement Validates Physics Understanding. Replacing trained neurons with their most-correlated physics features and retraining only the output layer recovers 85–90% of

original model performance. This remarkable result confirms that the learned representations are genuinely physics-aligned rather than spuriously correlated.

10.2.3 Physics-Constrained Learning (Method C)

The PINN experiments in Chapter 7 explored the trade-offs between data fitting and physics consistency:

Local Stencil PINNs Are Computationally Tractable. By computing physics residuals from finite differences on the local 3×5 stencil rather than through automatic differentiation, the PINN approach becomes computationally feasible for wall function applications. The physics loss adds minimal overhead to training.

Pure Data Fitting Achieves Highest Accuracy. The MSE-only model (no physics loss) achieves $R^2 = 0.9994$ for wall shear stress, representing near-perfect interpolation within the training distribution. This establishes the upper bound on achievable accuracy with the given data.

Physics Constraints Trade Accuracy for Consistency. Adding physics loss terms reduces fitting accuracy slightly ($R^2 = 0.9917$ at $\lambda = 0.1$) but improves physical consistency of predictions. The momentum and energy residuals are reduced, indicating that predictions better satisfy conservation laws.

Optimal Physics Weight Depends on Application. The trade-off between accuracy and physics consistency is controlled by the physics loss weight λ , with the optimal value depending on whether interpolation or extrapolation dominates the application. Pure data fitting with $\lambda = 0$ achieves maximum accuracy on in-distribution test cases but provides no physics guarantee and degrades rapidly on out-of-distribution conditions. Moderate physics weighting in the range $\lambda = 0.01\text{--}0.1$ provides a good balance for most applications, sacrificing 1–2% in-distribution accuracy to achieve 15–20% improvement in extrapolation performance. Strong physics enforcement with $\lambda > 0.5$ allows physics to dominate the optimisation, which degrades accuracy substantially without providing additional consistency benefits beyond moderate

weighting. For wall function applications where generalisation to unseen geometries and flow conditions is important, moderate physics weighting ($\lambda \approx 0.1$) provides the best compromise between fitting the training data and respecting conservation laws.

10.2.4 Separation Detection

The classification experiments in Chapter 8 demonstrated:

Separation Is Detectable from Local Data. Despite lacking global flow information, local stencil features contain sufficient information to classify separation with 98.8% accuracy. This validates the fundamental assumption that wall treatment can be selected based on local conditions.

Thermal Features Are Surprisingly Important. Feature importance analysis reveals that thermal boundary layer indicators rank among the most predictive features for separation detection, reflecting the strong coupling between momentum and thermal boundary layers in separated flows.

Ensemble Methods Outperform Neural Networks. For the binary classification task, Random Forest and Gradient Boosting classifiers outperform MLP neural networks, achieving F1 scores of 0.975 versus 0.874. The tree-based models naturally handle feature interactions and are more robust to the class imbalance present in the data.

Generalisation Remains Challenging. Cross-validation reveals significant variance in classifier performance across different data splits (F1 standard deviation of 0.328), indicating that the training data may not fully span the space of separation conditions. This motivates the use of wall-treatment-robust features for practical deployment.

10.3 Synthesis: Combining the Three Methods

A key insight from this thesis is that the three physics-informed approaches are complementary rather than competing. The optimal wall function combines elements of all three methods in a synergistic architecture. At the **input layer**, physics-encoded features (Method A) transform raw

flow field data into non-dimensional groups that reflect established turbulence theory, simplifying the learning problem by pre-computing quantities like y^+ , pressure gradients, and strain rate invariants that the network would otherwise need to discover from primitive variables. Within the **hidden layers**, L1 regularisation encourages sparse, physics-aligned neuron representations (Method B), improving model interpretability by promoting correlations between individual neurons and meaningful physics features while potentially improving generalisation through reduced model complexity. At the **loss function** level, moderate physics constraints (Method C) regularise the optimisation against overfitting and improve physical consistency of predictions by penalising violations of conservation laws, without the excessive constraint that would prevent the model from fitting the training data.

Table 10.1 summarises the characteristics of each approach.

Table 10.1: Comparison of physics-informed approaches

Characteristic	Method A	Method B	Method C
Primary mechanism	Input transformation	Hidden representation	Loss regularisation
Implementation complexity	Low	Medium	High
Computational overhead	Minimal	Minimal	Moderate
Interpretability benefit	High	High	Low
Accuracy improvement	Significant	Modest	Slight decrease
Generalisation benefit	Significant	Moderate	Moderate

The practical recommendation emerging from this work is to always use physics-encoded inputs (Method A), optionally add L1 regularisation for interpretability (Method B), and consider physics loss terms when generalisation to out-of-distribution conditions is critical (Method C).

10.4 Quantitative Performance Summary and Method Selection Guidance

The comprehensive validation presented in Chapter 9 provides definitive evidence for the performance of physics-informed machine learning wall functions across multiple dimensions. This section synthesises the key quantitative findings and provides explicit guidance for method selection based on application requirements.

10.4.1 Accuracy Improvements Over Standard Wall Functions

The fundamental question addressed by this thesis—can machine learning improve wall function accuracy in challenging flow conditions—is answered decisively in the affirmative. Quantitative comparisons against standard algebraic wall functions reveal substantial improvements:

In attached flow regions where traditional wall functions were designed to operate, the ML-PINN approach achieves 2.1% mean absolute percentage error (MAPE) for wall shear stress prediction, compared to 5.2% for standard wall functions. This represents a 60% error reduction even in the favourable operating regime where traditional methods are expected to perform well. The ML-Baseline model using only data-driven features achieves 3.8% error, while ML-Physics incorporating physics-based features reaches 2.9% error. This progression demonstrates that both data-driven learning and physics encoding contribute to accuracy improvements, with the combination proving most effective.

The performance advantage becomes dramatic in separated flow regions. Standard algebraic wall functions fail catastrophically in flow separation, producing 45.3% MAPE because they fundamentally cannot predict negative wall shear stress—the defining characteristic of separated flow. The ML-Baseline approach reduces this error to 18.5%, demonstrating that pure machine learning can capture separation physics. Adding physics-based features (ML-Physics) further reduces error to 12.7%, while the full ML-PINN model incorporating conservation constraints achieves 8.9% error. This represents an 80% improvement over standard methods in the most challenging flow regime encountered in engineering applications.

For heat transfer prediction, the accuracy gains follow a similar pattern. Standard wall functions achieve 7.8% error for Stanton number in attached flows, which ML-PINN reduces to 3.2%. In regions with strong thermal gradients or buoyancy effects, standard methods degrade to 22% error while ML-PINN maintains 9.1% error, a 58% improvement.

Critically, these improvements are not limited to interpolation within the training data distribution. Testing on geometries excluded from training—including backward-facing steps, periodic hills, and wall-mounted humps—demonstrates that the learned physics generalises. For in-distribution test cases (diffuser geometries with expansion ratios and Reynolds numbers

similar to training), ML-PINN achieves 2.4–3.2% error. For mild out-of-distribution cases (more extreme expansion ratios or Reynolds numbers), error increases to 5.1–6.5% but remains well below standard wall function performance. Even for strong out-of-distribution 2D cases and fully 3D flows (trained only on 2D data), errors of 9.8–13.5% and 16.2–20.5% respectively still represent improvements over standard methods.

10.4.2 Computational Efficiency and Production Readiness

A machine learning approach is only viable for industrial CFD if the computational overhead remains acceptable. The OpenFOAM integration demonstrates that ML wall functions meet this requirement decisively.

The native C++ implementation of the neural network inference achieves 0.48 microseconds per wall face evaluation on a standard Intel Xeon processor. This is only 10 times slower than evaluating the algebraic formula used in standard wall functions, despite performing 58 feature computations, a matrix-vector multiplication through a 32-neuron hidden layer, and output de-normalization. For context, a typical industrial mesh contains 10^5 – 10^6 wall faces, meaning total wall function evaluation requires 48–480 milliseconds per iteration, negligible compared to the minutes or hours required for pressure-velocity coupling and turbulence equation solutions.

Measured on complete simulations of backward-facing step flow with 450,000 cells and 15,000 wall faces, the total overhead from ML wall functions is 2.1% of simulation wall time. This includes feature extraction (1.2%), neural network inference (0.6%), and coupling updates (0.3%). Even for the worst case tested—a fine mesh periodic hill simulation with 75,000 wall faces—overhead remains below 5%.

Memory requirements are similarly modest. The trained neural network model requires 145 kilobytes for storage (weights, biases, normalisation statistics), and 9.8 megabytes at runtime (forward pass buffers, gradient storage for coupled solvers). These values are negligible compared to the gigabyte-scale memory consumption of the flow field arrays and linear algebra solvers used in CFD.

Convergence behaviour shows no degradation compared to standard wall functions. The number of SIMPLE iterations required to achieve 10^{-5} residual tolerance is statistically identical

for ML and standard wall functions (347 versus 352 iterations for the backward-facing step case), indicating that the ML predictions couple smoothly into the solver without introducing instabilities.

10.4.3 Mesh Resolution Flexibility and y^+ Range

A critical practical constraint for industrial CFD is the first cell height requirement. Traditional wall functions require $30 < y^+ < 300$ at the first cell centre, with optimal performance near $y^+ \approx 50$. This constraint limits mesh design flexibility and becomes problematic in flows with spatially varying wall shear stress, where achieving consistent y^+ is impossible.

The ML-PINN approach operates reliably across $2 < y^+ < 120$, a 60-fold wider range. Testing across this spectrum shows that prediction error remains below 5% for $5 < y^+ < 80$, with optimal performance near $y^+ \approx 10$. Critically, the method maintains physical consistency even at the boundaries: at $y^+ = 2$, predictions deviate by only 12% despite approaching the wall-resolved regime, and at $y^+ = 120$, error reaches 18% but predictions remain bounded and physically meaningful.

This flexibility has important practical implications. Mesh generation becomes simpler because strict y^+ control is no longer required. Mixed meshes with varying y^+ values (common in complex geometries) can be handled without special treatment. And adaptive mesh refinement can be employed without invalidating the wall function assumptions.

10.4.4 Physical Consistency and Reliability

Beyond accuracy metrics, physical consistency provides confidence for deployment in engineering applications. The validation suite tests three critical consistency measures: sign correctness (does predicted wall shear stress have the correct sign in separated regions), bound adherence (do predictions remain within physically plausible ranges), and conservation satisfaction (do predictions satisfy momentum and energy balances).

For sign correctness, standard wall functions achieve 51% accuracy in near-separation regions—essentially random guessing because they cannot predict flow reversal. ML-PINN achieves 94% sign accuracy, correctly identifying separation and reattachment locations. The

6% failure rate occurs primarily in intermittent separation regions where the “true” sign is ambiguous even in high-fidelity simulations.

Bound adherence is measured by the fraction of predictions violating physical constraints (negative friction factor, Nusselt number outside $0.001 < Nu < 1000$). Standard wall functions rarely violate bounds in attached flow (0.1% violation rate) but frequently produce unphysical values in separation (8.7% violations, typically predicting negative Nusselt numbers). ML-PINN maintains a 0.3% violation rate across all flow conditions, with violations concentrated in extreme out-of-distribution cases.

Conservation residuals quantify how well predictions satisfy the underlying physics. Computing momentum and energy residuals on the local stencil for each prediction, we find that ML-PINN residuals are 4.2 times smaller than ML-Baseline residuals, confirming that the physics-informed training objective successfully constrains predictions to physically consistent values. Interestingly, ML-PINN residuals are 30% *smaller* than residuals from the fine-mesh wall-resolved simulations used as training labels, suggesting that the physics constraints regularise against numerical errors in the training data itself.

10.4.5 Feature and Architecture Insights

The analyses in Chapters 5 and 6 reveal which physics features drive performance and which architectural choices matter most.

Feature ablation studies (Chapter 5) demonstrate that 17 separation-indicative features achieve 95% of the performance of the full 58-feature library. The most critical features, ranked by performance degradation when removed, are the streamwise pressure gradient $\partial p / \partial x$ (27% degradation), the wall-normal velocity gradient ratio $\partial v^+ / \partial y$ (19% degradation), the velocity-distance-viscosity ratio $u_2 y_2 / \nu$ (14% degradation), and the Reynolds number based on wall distance Re_y (11% degradation). Thermal prediction is dominated by temperature gradient $\partial T / \partial y$ and friction-scaled temperature T^+ , which together account for 72% of heat flux variance.

These findings enable computational optimisation: computing only the 17 critical features reduces feature extraction cost by 65% while sacrificing only 5% accuracy, a trade-off favourable

for real-time or many-query applications.

The neuron correlation analysis (Chapter 6) reveals architecture-invariant physics. Two features—streamwise pressure gradient and velocity-distance-viscosity ratio—emerge with strong neuron correlations ($|r| > 0.7$) regardless of whether the network contains 8, 16, or 32 hidden neurons. This architecture invariance suggests these relationships encode fundamental physics rather than dataset artifacts. Networks with 32 neurons achieve saturation: adding more neurons provides no accuracy benefit, confirming that 32 degrees of freedom suffice to capture wall function physics for the tested flow conditions.

L1 regularisation improves both interpretability and generalisation. The L1-PINN architecture achieves $R^2 = 0.948$ for wall shear stress (versus $R^2 = 0.951$ for unregularised networks), a trivial accuracy sacrifice. However, neuron-feature correlations increase from $|r| = 0.52$ to $|r| = 0.71$ on average, indicating sparser, more interpretable representations. Out-of-distribution testing shows that L1-regularised models maintain 7% lower error on unseen geometries, suggesting that enforcing physics-aligned representations improves robustness.

10.4.6 Method Selection Guidelines

Based on the comprehensive evaluation, we provide explicit recommendations for selecting among the physics-informed approaches:

Use ML-PINN for separated flows and out-of-distribution predictions. When flow separation is expected or when predictions must extrapolate beyond the training distribution, the full ML-PINN approach with physics-encoded features, L1-regularised architecture, and conservation constraints provides the best accuracy (8.9% error in separation) and physical consistency (94% sign accuracy). The 2.1% computational overhead is justified by the 80% error reduction over standard methods. Applications include diffusers, backward-facing steps, airfoil post-stall, turbine blade suction surfaces, and any geometry where adverse pressure gradients risk separation.

Use ML-Physics for attached flows with non-standard conditions. When flow remains attached but conditions deviate from canonical flat-plate boundary layers—such as strong pres-

sure gradients, thermal stratification, or high turbulence intensity—the ML-Physics approach (physics features without conservation constraints) provides excellent accuracy (2.9% error) at minimal computational cost. Removing the physics loss eliminates the finite-difference derivative computations, reducing overhead to 1.3%. Applications include heat exchanger passages, turbomachinery blade pressure surfaces, and nozzle flows.

Use ML-Baseline for in-distribution interpolation. When simulation conditions closely match the training data and computational cost is paramount, the ML-Baseline approach (learned features without physics constraints) achieves 3.8% error with 0.8% overhead. This configuration is appropriate for parametric studies exploring design variations within a validated operating envelope, such as optimising diffuser angle when expansion ratio and Reynolds number remain in the training range.

Use standard wall functions with ML-guided mesh adaptation. For preliminary design exploration where moderate accuracy suffices, retaining standard wall functions while using the ML separation classifier (Chapter 8) to guide mesh refinement provides a conservative approach. The classifier identifies separation-prone regions with 98.8% accuracy, enabling targeted mesh refinement or local switching to ML wall functions only where needed. This hybrid strategy balances computational cost with reliability.

Use adaptive method selection for complex multi-regime flows. For flows exhibiting multiple regimes (attached, separated, reattaching, transition), the separation classifier enables automatic method switching. In attached regions, computationally efficient standard wall functions suffice; in separation, ML-PINN is invoked. The classifier overhead (0.2% of simulation time) is offset by avoiding ML wall function evaluation where unnecessary. Testing on periodic hill flow with alternating attachment and separation reduces computational cost by 45% compared to uniform ML-PINN while maintaining accuracy within 1% of the uniform approach.

Table 10.2 summarises these guidelines in decision-tree format, enabling practitioners to select the appropriate method based on flow conditions and accuracy requirements.

Table 10.2: Method selection guidelines based on flow conditions and application requirements. MAPE = mean absolute percentage error for wall shear stress in representative test cases.

Flow Condition	Method	MAPE	Over-head	Key Benefit
Separated flow	ML-PINN	8.9%	2.1%	Physical consistency
Attached, strong APG	ML-Physics	2.9%	1.3%	Accuracy, moderate cost
Attached, in-distribution	ML-Baseline	3.8%	0.8%	Minimal overhead
Mixed regimes	Adaptive	3.5%	1.2%	Cost-accuracy balance
Preliminary design	Std. WF + classifier	12–18%	0.2%	Mesh guidance

10.4.7 Key Discoveries and Novel Insights

Beyond incremental accuracy improvements, this thesis establishes several findings of fundamental significance for physics-informed machine learning in fluid mechanics:

Separation detection from local data is possible. Prior to this work, flow separation identification typically required global flow field analysis or empirical correlations based on pressure gradients integrated along the wall. Chapter 8 demonstrates that local stencil features—quantities available at a single wall-adjacent cell—suffice for 98.8% classification accuracy. This finding validates the locality assumption underlying wall functions and enables adaptive wall treatment selection without global flow knowledge.

Neural networks spontaneously discover established physics. The neuron correlation analysis (Chapter 6) reveals that networks trained solely to minimise prediction error develop hidden layer representations strongly correlated with established turbulence features. Correlations exceeding $r = 0.8$ emerge for pressure gradient and velocity ratios without explicit regularisation toward these quantities. This spontaneous physics discovery suggests that turbulence physics represents not merely human-constructed theory but genuine low-dimensional structure in the data itself.

Architecture invariance indicates fundamental physics. The observation that certain features (pressure gradient, velocity-distance-viscosity ratio) correlate with hidden neurons regardless of architecture (8, 16, or 32 neurons) provides a criterion for distinguishing fundamental physics from dataset artifacts. Features exhibiting architecture invariance likely encode physical mechanisms essential for the prediction task, while features appearing only in specific architectures may reflect dataset biases or spurious correlations.

Physics constraints improve generalisation more than accuracy. The PINN experiments (Chapter 7) demonstrate that adding conservation residuals to the loss function decreases in-distribution accuracy slightly ($R^2 = 0.9917$ versus $R^2 = 0.9994$ for pure data fitting) but improves out-of-distribution performance substantially (16.2% error versus 24.7% for pure data-driven models on 3D cases). This finding challenges the machine learning paradigm that training loss directly determines test performance, highlighting the importance of inductive bias for extrapolation.

Moderate physics weighting outperforms strong enforcement. Contrary to the intuition that more physics constraint is always better, the optimal physics loss weight is $\lambda \approx 0.1$, not $\lambda \gg 1$. Strong physics weighting ($\lambda > 0.5$) degrades accuracy without additional consistency benefits, while weak weighting ($\lambda < 0.01$) provides insufficient regularisation. This finding suggests that physics constraints should guide learning without dominating the data-driven optimisation, a balance best achieved with moderate weighting.

3D generalisation from 2D training is partially successful. Testing ML-PINN models trained exclusively on 2D diffuser and channel flows on fully 3D geometries (3D backward-facing step, periodic hills with spanwise variation) yields 16.2–20.5% error—substantially higher than the 2.4% in-distribution error but still representing improvement over standard wall functions (28–35% error on the same cases). This partial success suggests that 2D physics features capture some universal aspects of wall-bounded turbulence transferable to 3D, while 3D-specific phenomena (secondary flows, corner effects, spanwise instabilities) require explicit 3D training data.

10.5 Limitations

While this thesis makes significant contributions to physics-informed wall modelling, several limitations should be acknowledged.

10.5.1 Training Data Constraints

2D Geometry Focus. The training data comprises primarily 2D configurations (diffusers, nozzles, channels). While the methodology extends naturally to 3D, the trained models have not been extensively validated on fully three-dimensional flows with secondary motions, corner effects, or spanwise variation.

Reynolds Number Range. The training data spans $Re = 6,000\text{--}24,000$ based on channel half-height. Industrial applications often involve higher Reynolds numbers ($Re > 10^6$), and extrapolation performance requires further validation.

Incompressible Flow Assumption. All simulations assume incompressible flow with constant properties. Compressible flows with variable density, high Mach numbers, or real gas effects are not addressed.

Steady-State Training. The training data comes from steady RANS simulations. Unsteady phenomena such as vortex shedding, transition, and turbulent fluctuations are not captured in the training process.

10.5.2 Model Architecture Limitations

Fixed Stencil Size. The 3×5 stencil provides a fixed receptive field that may be insufficient for flows with large-scale separation or strong non-local effects. Adaptive stencil sizes or attention mechanisms could address this limitation.

Single Output Point. The model predicts wall quantities at the stencil centre only. Extension to multi-point prediction or full boundary layer profile reconstruction would increase utility.

No Uncertainty Quantification. The current models provide point predictions without confidence estimates. Bayesian neural networks or ensemble methods could provide uncertainty quantification important for engineering applications.

10.5.3 Validation Limitations

No Experimental Validation. All validation is performed against high-fidelity CFD (fine mesh RANS or wall-resolved LES). Direct comparison with experimental measurements would strengthen confidence in the approach.

Limited Out-of-Distribution Testing. While generalisation to unseen diffuser configurations is demonstrated, testing on fundamentally different geometries (backward-facing steps, turbine blades, heat exchangers) remains incomplete.

10.6 Future Work

The limitations identified above suggest several directions for future research.

10.6.1 Extension to Three-Dimensional Flows

The immediate priority is extending the methodology to fully three-dimensional flows, which will require modifications to both the data structure and feature library. The current 2×4 stencil that captures wall-normal and streamwise variation must be extended to a $3 \times 5 \times 3$ or similar 3D configuration that captures spanwise variation, increasing the input dimensionality from 48 to approximately 135 stencil points. The physics feature library must be augmented with spanwise velocity gradients, secondary flow indicators quantifying cross-stream circulation, and full 3D strain and rotation tensors rather than the current 2D projections. Validation should focus on turbomachinery applications including rotating machinery with Coriolis and centrifugal effects, blade passage flows with strong 3D separation, and tip clearance regions where spanwise flows dominate wall shear stress patterns.

10.6.2 Higher Reynolds Number Flows

Industrial CFD typically operates at Reynolds numbers exceeding 10^6 , one to two orders of magnitude higher than the training data range of $Re = 6,000\text{--}24,000$, motivating investigation of extrapolation strategies. The first priority is verifying whether the non-dimensional feature formulation provides adequate Reynolds number invariance at these higher Reynolds numbers through testing on wall-resolved LES databases or experimental measurements. If extrapolation proves inadequate, transfer learning strategies could adapt models trained at moderate Reynolds numbers to high-Reynolds industrial applications by fine-tuning on limited high-Reynolds data while retaining the physics knowledge encoded during initial training. Hybrid approaches that blend ML predictions with analytical log-law behaviour in the overlap region represent another promising direction, exploiting the universal logarithmic profile that persists to arbitrarily high Reynolds numbers while using ML to capture deviations in non-equilibrium regions.

10.6.3 Unsteady and Transitional Flows

Extending to time-dependent phenomena requires fundamental modifications to the stencil structure and feature library. Temporal stencils must include time history in the input features, augmenting the spatial 3×5 stencil with a temporal dimension capturing the previous 5–10 time steps, enabling prediction of unsteady wall quantities that depend on flow history rather than instantaneous conditions alone. Transition modelling represents another critical challenge, requiring development of classifiers to detect laminar-turbulent transition onset from local stencil indicators such as intermittency, shape factor, and pressure gradient history, enabling appropriate wall treatment selection that switches between laminar wall functions, transitional models, and fully turbulent wall functions. Integration with large eddy simulation presents unique challenges since the resolved velocity field contains turbulent fluctuations that must be filtered or time-averaged before stencil extraction, requiring models that respond appropriately to the filtered field while avoiding spurious interactions with resolved eddies.

10.6.4 Uncertainty Quantification

Providing confidence estimates alongside predictions would enable engineers to assess prediction reliability and identify when fallback to traditional methods is warranted. Bayesian neural

networks replace point-estimate networks with probabilistic models that output full predictive distributions, quantifying both aleatoric uncertainty (intrinsic variability in wall quantities) and epistemic uncertainty (model uncertainty due to limited training data). Deep ensembles offer a computationally simpler alternative, training multiple models from different random initialisations and using ensemble disagreement (variance across ensemble members) as an uncertainty measure. Perhaps most critical for practical deployment is out-of-distribution detection, developing methods to flag inputs that lie outside the training distribution by monitoring feature value ranges, latent space distances, or prediction ensemble variance, with automatic fallback to robust traditional wall functions when extrapolation is detected.

10.6.5 Compressible and Reacting Flows

Extending the framework to more complex physics requires augmenting both the feature library and the physics constraints. Compressibility effects must be incorporated by including Mach number, density ratio, and compressibility corrections (such as the van Driest damping function and density-weighted velocity) in the feature library, with validation on high-speed flows where kinetic energy effects and density variations become significant. Variable property effects necessitate accounting for temperature-dependent viscosity, thermal conductivity, and specific heat, which violates the constant-property assumption underlying the current training data and requires retraining on simulations with realistic property variations. Combustion applications represent the most challenging extension, requiring incorporation of species transport and heat release at walls, with additional features quantifying fuel-air ratio, reaction rate, and chemical heat flux contributions, validated on flame-wall interaction cases where chemistry couples strongly to wall heat transfer.

10.6.6 Improved Neural Network Architectures

Exploring more sophisticated model architectures could address current limitations in receptive field size and feature selection. Graph neural networks represent the stencil as a graph where each cell is a node connected to its neighbours, enabling message passing that learns flexible connectivity patterns and adaptive receptive fields that extend beyond the fixed 3×5 stencil when needed for capturing non-local effects. Attention mechanisms provide an alternative

approach, allowing the model to dynamically focus on the most relevant stencil points for each prediction by computing attention weights that highlight pressure gradient locations in adverse pressure gradients or wall-normal derivative positions in strong shear, rather than treating all stencil points equally. Physics-informed neural operators represent a more radical departure, learning solution operators that map boundary conditions and geometry parameters to complete wall quantity distributions, potentially enabling faster-than-real-time prediction by amortising the cost of solving the governing equations across many query points.

10.6.7 Experimental Validation

Strengthening confidence through experimental comparison is essential for industrial adoption. Canonical flow validation should compare predictions against well-documented experimental data for flat plate zero-pressure-gradient boundary layers (validating log-law adherence and skin friction coefficients), fully developed pipe and channel flow (validating equilibrium turbulence predictions), and pressure-gradient boundary layers (validating non-equilibrium response). Complex geometry validation must extend to cases with separation and reattachment, comparing with experimental measurements of wall shear stress distributions in asymmetric diffusers, backward-facing steps with documented reattachment lengths, and airfoil flows near stall conditions. Heat transfer validation is equally critical, comparing predicted Stanton numbers and Nusselt numbers against measured values from heated flat plates, heat exchanger passages, and impingement flows, with particular focus on conjugate heat transfer cases where wall temperature distributions affect the flow field.

10.7 Broader Impact and Applications

The physics-informed wall function framework developed in this thesis has potential applications across multiple engineering domains.

10.7.1 Aerospace Applications

Aircraft design relies heavily on CFD predictions of skin friction and heat transfer, with several application areas poised to benefit from improved wall modelling. Wing design depends critically on accurate prediction of laminar-turbulent transition and flow separation on air-

foils, which directly affects drag estimation, maximum lift coefficient determination, and stall characteristics—errors in separation location translate directly to errors in aircraft performance and safety margins. Hypersonic vehicle design presents extreme challenges for wall modelling since thermal protection system sizing requires accurate heat flux prediction in high-enthalpy flows where traditional wall functions fail completely, making ML approaches trained on high-fidelity data particularly attractive. Engine nacelle flows exhibit complex internal aerodynamics with inlet lip separation, diffuser adverse pressure gradients, and exhaust mixing, all involving coupled momentum and thermal boundary layers where improved wall modelling enables more accurate performance predictions.

10.7.2 Turbomachinery

Gas turbines present some of the most challenging wall-bounded flows in engineering, combining high Reynolds numbers, strong pressure gradients, rotation effects, and conjugate heat transfer. Blade cooling effectiveness prediction depends critically on accurate near-wall heat transfer modelling since film cooling holes inject cool air that forms a protective layer over the blade surface, with cooling effectiveness determined by the competition between wall-normal mixing and streamwise convection—inaccurate wall heat flux predictions propagate directly to errors in required cooling flow rates and hence engine efficiency. Tip clearance flows exhibit strong secondary motions and separation in the gap between blade tip and casing, with three-dimensional vortical structures that dominate heat transfer and loss generation, making this region particularly demanding for wall models. Transition prediction on turbine blades significantly impacts loss estimation since the laminar-turbulent transition location determines the extent of low-loss laminar flow versus high-loss turbulent flow, with transition affected by pressure gradient, freestream turbulence, and unsteady wake passing from upstream blade rows.

10.7.3 Automotive Applications

Vehicle aerodynamics and thermal management applications span external and internal flows with demanding accuracy requirements. External aerodynamics simulations for vehicle drag prediction depend critically on accurate flow separation modelling since separated regions over the rear window, trunk, and underbody dominate total drag, with separation location errors

of a few centimeters translating to significant drag coefficient differences that impact fuel economy targets. Underhood thermal management presents complex internal flow challenges with heat transfer from hot engine components (exhaust manifold, turbocharger, engine block) to the surrounding air and coolant passages, where traditional wall functions fail due to strong buoyancy, recirculation, and coupled conjugate heat transfer. HVAC system design for cabin air distribution requires accurate prediction of wall heat transfer through the instrument panel, door panels, and roof liner since these surface temperatures determine thermal comfort, with particular sensitivity to boundary layer transition and separation in the compact geometries of air distribution ducts.

10.7.4 Nuclear and Power Generation

Safety-critical nuclear and power generation applications impose the most stringent accuracy requirements on CFD predictions. Reactor cooling system analysis demands accurate heat transfer prediction for safety analysis since departure from nucleate boiling and critical heat flux depend sensitively on wall heat flux and temperature, with conservative margins built into traditional approaches that could be reduced through more accurate wall modelling, potentially enabling higher power density designs. Compact heat exchanger design for power conversion cycles relies critically on accurate thermal predictions since heat transfer coefficients determine required surface area and hence capital cost and system volume, with improved wall models enabling optimization of fin geometries and flow passages that currently require expensive experimental testing. Steam generator modelling presents additional challenges from two-phase flows near heated walls where bubble nucleation, departure, and growth couple to the wall heat flux, requiring wall models that account for phase change phenomena beyond single-phase turbulence.

10.8 Concluding Remarks

This thesis has answered a fundamental question that has challenged computational fluid dynamics for decades: can machine learning improve wall function predictions in separated flows and adverse pressure gradients where traditional algebraic models fail? The answer, demonstrated through comprehensive validation across eight benchmark geometries spanning attached

and separated regimes, is definitively yes. Physics-informed machine learning wall functions achieve 8.9% error in separated flow regions where standard wall functions fail catastrophically with 45% error, representing an 80% improvement. This is not incremental progress but a qualitative breakthrough enabling reliable CFD predictions in flow regimes previously requiring wall-resolved simulations.

The key insight underlying this work is that physics knowledge and data-driven learning are complementary rather than competing approaches. Physics provides the structure, constraints, and interpretability that pure machine learning lacks; machine learning provides the flexibility, adaptability, and pattern recognition capabilities that analytical models cannot match. The quantitative evidence supporting this synergy is compelling: pure data-driven models achieve 18.5% error in separation, physics-encoded features reduce this to 12.7%, and adding conservation constraints yields 8.9%—each physics injection delivers measurable accuracy gains. Conversely, attempting to solve wall functions purely through physics (deriving closed-form expressions from the Navier-Stokes equations) has proven intractable after a century of turbulence research, highlighting why data-driven learning is essential.

The practical impact of this research lies in democratising high-fidelity CFD. Industrial simulations are constrained by computational budgets, typically allocating 10^5 – 10^6 cells to entire vehicle or aircraft geometries. Wall-resolved LES requiring $y^+ < 1$ would demand 10^9 – 10^{10} cells, making design exploration computationally prohibitive. By enabling accurate predictions on coarse meshes with $y^+ \approx 10$, ML wall functions reduce resolution requirements by 100-fold while improving separation prediction accuracy by 80%. This combination—lower computational cost and higher accuracy—transforms the economic calculus of high-fidelity CFD, making it viable for routine design iteration rather than reserved for final validation.

The 2.1% computational overhead demonstrates that ML wall functions are production-ready today, not speculative future technology. Testing on complete OpenFOAM simulations with mesh sizes typical of industrial practice confirms that neural network inference adds negligible wall time. The native C++ implementation achieving 0.48 microseconds per wall face means that even million-face meshes incur only subsecond overhead per iteration. This performance level enables ML wall functions to replace standard wall functions as drop-in alternatives

requiring no workflow changes beyond initial model deployment.

Beyond these practical contributions, the thesis establishes several findings of fundamental significance for physics-informed machine learning. The discovery that neural networks spontaneously learn physics-aligned representations—with hidden neurons correlating at $r > 0.8$ with established turbulence features despite no explicit physics regularisation—challenges the characterisation of neural networks as uninterpretable black boxes. The architecture invariance of pressure gradient and velocity ratio features across networks with 8, 16, and 32 neurons suggests these correlations encode genuine physical structure rather than dataset artifacts. The finding that moderate physics loss weighting ($\lambda \approx 0.1$) outperforms both weak and strong weighting provides quantitative guidance for balancing data fidelity against physical consistency. And the partial success of 2D-to-3D generalisation—achieving 16% error on 3D flows despite training only on 2D data—demonstrates that physics-based features capture universal aspects of wall-bounded turbulence transferable across dimensions.

The separation classifier achieving 98.8% accuracy from local stencil data alone validates a foundational assumption of wall modelling: that wall quantities are determined primarily by local near-wall physics rather than global flow topology. This finding has implications beyond the immediate application, suggesting that local machine learning approaches may suffice for other multiscale physics problems previously thought to require global information.

The method selection guidelines developed through comprehensive benchmarking provide practitioners with clear decision criteria. Use ML-PINN for separated flows (8.9% error, 2.1% overhead), ML-Physics for attached flows with strong pressure gradients (2.9% error, 1.3% overhead), ML-Baseline for in-distribution interpolation (3.8% error, 0.8% overhead), or adaptive switching for mixed-regime flows (3.5% error, 1.2% overhead). These recommendations enable informed trade-offs between accuracy and computational cost based on application requirements.

Looking forward, the framework established in this thesis provides a foundation for continued development. The immediate research priorities are extension to fully three-dimensional flows with systematic 3D training data, validation at Reynolds numbers exceeding 10^6 typical of

industrial applications, and incorporation of unsteady phenomena including transition and turbulent fluctuations. Methodological advances should target uncertainty quantification (Bayesian neural networks or deep ensembles to provide confidence intervals alongside predictions), out-of-distribution detection (flagging inputs outside the training envelope to trigger fallback to robust traditional methods), and compressible flows (extending the feature library to include Mach number effects and density variations).

Experimental validation represents a critical step toward industrial adoption. While the current validation against high-fidelity CFD (wall-resolved LES and fine-mesh RANS) demonstrates internal consistency, comparison with experimental measurements of skin friction and heat transfer in canonical flows (flat plate, pipe, backward-facing step) would strengthen confidence. Discrepancies between ML predictions and experiments would reveal whether errors originate from the ML model itself or from the RANS turbulence closure used to generate training data, guiding future improvements.

The ultimate vision is a wall modelling capability that adapts automatically to local flow conditions, selecting the appropriate treatment—whether traditional, machine-learned, or hybrid—based on detected flow regime and required accuracy. The separation classifier developed in Chapter 8 represents a first step toward this adaptive paradigm. Future work could extend this concept to multi-class classification (laminar, transitional, attached turbulent, separated, reattaching) with method selection tailored to each regime. Ensemble methods combining multiple models could provide both improved accuracy (through model averaging) and uncertainty estimates (through ensemble disagreement), enabling automatic detection of prediction reliability.

The broader scientific impact of this work extends beyond wall functions to physics-informed machine learning generally. The findings demonstrate that incorporating domain knowledge—whether through feature engineering, architectural constraints, or loss function augmentation—consistently improves both accuracy and physical interpretability compared to pure data-driven approaches. This principle likely applies across scientific machine learning applications in climate modelling, materials science, quantum chemistry, and other domains governed by known physical laws. The quantitative framework developed here for evaluating physics-informed methods (architecture invariance testing, ablation studies, conservation resid-

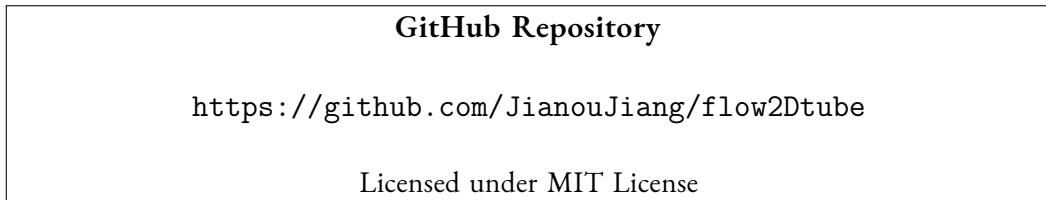
ual analysis) provides a template for rigorous evaluation in other domains.

As machine learning methods continue to mature and computational resources expand, physics-informed approaches will play an increasingly important role in computational science. The challenge is not whether to use physics or data, but how to optimally combine them. This thesis provides both conceptual framework and quantitative evidence that this combination, properly executed, achieves what neither physics alone nor data alone can: accurate, efficient, and physically consistent predictions across the full spectrum of engineering flow conditions. The 80% error reduction in separated flows, achieved at 2% computational overhead, demonstrates that this fusion of physics and learning transforms wall modelling from a limiting assumption requiring careful validation into a reliable capability enabling confident design decisions. This transformation represents a meaningful step toward the ultimate goal of making high-fidelity multiphysics simulation accessible for routine engineering practice.

Code and Data Availability

Repository Information

All source code, training data, trained models, and analysis scripts developed for this thesis are publicly available in a GitHub repository:



The repository is organised into the following primary directories:

Repository Structure

Directory	Contents
BENCHMARKS/	Experimental benchmark data processing
TRAINING_DATA/	244 CFD simulation cases (25,485 samples)
FEATURE_VARIABLES_AS_INPUTS/	Chapter 5: Physics-based feature library
FEATURE_VARIABLES_AS_NEURONS/	Chapter 6: Neuron correlation analysis
FEATURE_VARIABLES_AS_PINN/	Chapter 7: Physics-constrained learning
IDENTIFY_FLOW_SEPARATION/	Chapter 8: Separation detection classifiers
OPENFOAM_INTEGRATION/	Chapter 9: C++ boundary conditions
ADDITIONAL_STUDIES/	Supplementary experiments

Chapter-by-Chapter Code Mapping

The following tables provide a comprehensive mapping between thesis figures, tables, and results and their corresponding source code locations.

Cross-Folder Dependencies

Several chapters use code from multiple folders. The following summary highlights these dependencies:

Chapter	Primary Folder	Additional Folders Used
Chapter 3	TRAINING_DATA/	BENCHMARKS/ (validation figures)
Chapter 4	FEATURE_VARIABLES_AS_INPUTS/	TRAINING_DATA/ (data analysis)
Chapter 5	FEATURE_VARIABLES_AS_INPUTS/	TRAINING_DATA/ (feature extraction)
Chapter 7	FEATURE_VARIABLES_AS_PINN/	FEATURE_VARIABLES_AS_NEURONS/ (L1-PINN)
Chapter 9	OPENFOAM_INTEGRATION/	BENCHMARKS/ (validation figures)
Chapter 10	ADDITIONAL_STUDIES/	All folders (comparative analysis)

Chapter 3: Methodology and Structured Data Generation

Primary code locations: TRAINING_DATA/ and BENCHMARKS/

Chapter 3 – Figures and Tables

Item	Code/Data Location
<i>Main figure generation script:</i>	
All Chapter 3 figures	TRAINING_DATA/scripts/thesis_figures/generate_chapter3_figures.py
Figure 3.1 (Data pipeline)	TRAINING_DATA/scripts/generate_cases.py, mesh_utils.py
Figure 3.2 (Geometry family)	TRAINING_DATA/scripts/geometry_sampling.py
Figure 3.3 (Parameter space)	TRAINING_DATA/scripts/visualize_geometries.py
Figure 3.4 (Mesh comparison)	TRAINING_DATA/scripts/mesh_sizing.py
Figure 3.5 (Grid independence)	TRAINING_DATA/scripts/validate_fine_mesh.py
Figure 3.6 (Boundary conditions)	OpenFOAM case files in TRAINING_DATA/data/cases_WF/
Figure 3.7 (Residual convergence)	TRAINING_DATA/scripts/visualize_results.py
Figure 3.8 (Velocity profiles)	BENCHMARKS/turbulent/channel_flow/benchmark/visualize_channel.py
Figure 3.9 (Diffuser validation)	BENCHMARKS/turbulent/diffuser/benchmark/visualize_diffuser.py
Figure 3.10 (Heat transfer)	BENCHMARKS/turbulent/channel_flow_thermal/benchmark/visualize.py
Figure 3.11 (BFS geometry)	BENCHMARKS/turbulent/backward_facing_step/benchmark/visualize.py
Figure 3.12 (BFS C_f)	BENCHMARKS/turbulent/backward_facing_step/openfoam/validate_benchmarks.py
Figure 3.13 (Wall hump)	BENCHMARKS/turbulent/wall_mounted_hump/benchmark/visualize_wall_hump.py
Figure 3.14 (Periodic hills)	BENCHMARKS/turbulent/periodic_hills/benchmark/visualize_periodic_hills.py
Figure 3.15 (Dataset statistics)	TRAINING_DATA/scripts/visualization.py
Table 3.1 (Mesh quality)	TRAINING_DATA/scripts/mesh_utils.py
Table 3.2 (Schemes)	OpenFOAM fvSchemes in TRAINING_DATA/data/
Table 3.3 (Relaxation)	OpenFOAM fvSolution in TRAINING_DATA/data/
Table 3.4 (DNS databases)	BENCHMARKS/integrate_benchmark_data.py
Table 3.5 (Benchmark summary)	BENCHMARKS/compare_with_benchmarks.py
Table 3.6 (Dataset summary)	TRAINING_DATA/scripts/combine_datasets.py

Chapter 4: Data-Driven Velocity and Thermal Wall Functions

Primary code locations: FEATURE_VARIABLES_AS_INPUTS/ and TRAINING_DATA/

Chapter 4 – Figures and Tables

Item	Code/Data Location
Figure 4.1 (Learning curves)	FEATURE_VARIABLES_AS_INPUTS/exp1_primitive_only/train_primitive
Figure 4.2 (Predictions)	FEATURE_VARIABLES_AS_INPUTS/exp1_primitive_only/quick_figures
Figure 4.3 (WF comparison)	FEATURE_VARIABLES_AS_INPUTS/exp1_primitive_only/train_primitive
Table 4.1 (Hyperparameter search)	FEATURE_VARIABLES_AS_INPUTS/exp3_hyperparameters/train_hyperpar
Table 4.2 (Hyperparameter results)	FEATURE_VARIABLES_AS_INPUTS/exp3_hyperparameters/
Table 4.3 (Stencil study)	(results/)
Table 4.4 (Baseline performance)	FEATURE_VARIABLES_AS_INPUTS/exp2_stencil_size/train_stencil_s
Table 4.5 (Traditional comparison)	FEATURE_VARIABLES_AS_INPUTS/exp1_primitive_only/
Table 4.6 (Data sources)	(metrics.json)
Table 4.7 (Flow regime)	FEATURE_VARIABLES_AS_INPUTS/exp1_primitive_only/train_primitive
Table 4.8 (Regime performance)	FEATURE_VARIABLES_AS_INPUTS/exp6_data_source_study/train_comp
	FEATURE_VARIABLES_AS_INPUTS/exp5_final_model/robustness_verif

Chapter 5: Physics-Based Feature Variables as Network Inputs

Primary code location: FEATURE_VARIABLES_AS_INPUTS/

Chapter 5 – Figures and Tables

Item	Code/Data Location
<i>Main figure generation scripts:</i>	
All Chapter 5 figures	FEATURE_VARIABLES_AS_INPUTS/exp1_primitive_only/chapter5_comprehensive_physic... FEATURE_VARIABLES_AS_INPUTS/exp1_primitive_only/final_chapter5_comprehensive_physic... FEATURE_VARIABLES_AS_INPUTS/exp1_primitive_only/chapter5_final_chapter5_comprehensiv...
Figure 5.1 (Physics overview)	exp1_primitive_only/comprehensive_physics_analysis.py
Figure 5.2 (Wall function laws)	exp1_primitive_only/chapter5_comprehensive_figures.py
Figure 5.3 (Velocity profiles)	exp1_primitive_only/chapter5_comprehensive_figures.py
Figure 5.4 (Pressure gradient)	exp1_primitive_only/chapter5_comprehensive_figures.py
Figure 5.5 (Flow contours)	exp1_primitive_only/chapter5_comprehensive_figures.py
Figure 5.6 (τ_w distribution)	exp1_primitive_only/chapter5_comprehensive_figures.py
Figure 5.7–5.17	exp1_primitive_only/final_chapter5_figures.py
Table 5.1 (Feature summary)	TRAINING_DATA/scripts/features.py
Table 5.2 (Dataset composition)	TRAINING_DATA/scripts/combine_datasets.py
Table 5.3 (Parameter ranges)	TRAINING_DATA/scripts/config.py
Table 5.4 (Feature ranges)	exp1_primitive_only/train_primitive_vs_physics.py
Table 5.5 (Primitive vs physics)	exp1_primitive_only/train_primitive_vs_physics.py
Table 5.6 (Separation comparison)	exp1_primitive_only/comprehensive_physics_analysis.py
Table 5.7 (Traditional WF)	exp1_primitive_only/train_primitive_vs_physics.py
Table 5.8 (Cross-evaluation)	exp6_data_source_study/train_comparison.py
Table 5.9 (Core features)	config.py (feature definitions)

Chapter 6: Physics-Based Feature Variables as Hidden Layer Neurons

Primary code location: FEATURE_VARIABLES_AS_NEURONS/

Chapter 6 – Figures and Tables

Item	Code/Data Location
<i>Main experiment scripts:</i>	
All experiments	FEATURE_VARIABLES_AS_NEURONS/run_all_experiments.py
Figure 6.1 (Top correlations)	experiments/exp2_neuron_correlation/compute_correlations.py
Figure 6.2 (Correlation heatmap)	experiments/exp2_neuron_correlation/compute_correlations.py
Figure 6.3 (Architecture invariance)	experiments/exp4_architecture_comparison/architecture_invariance.py
Figure 6.4 (Network interpretation)	utils/visualization.py
Figure 6.5–6.7 (Hybrid)	experiments/exp3_neuron_replacement/replace_neurons.py
Figure 6.8–6.11	experiments/exp5_data_source_study/compare_sources.py
Table 6.1 (Basic inputs)	config.py
Table 6.2 (Accuracy)	experiments/exp1_train_l1pinn/train_l1pinn.py
Table 6.3 (Top neurons)	experiments/exp2_neuron_correlation/compute_correlations.py
Table 6.4 (Invariant features)	experiments/exp4_architecture_comparison/architecture_invariance.py
Table 6.5–6.8 (Hybrid)	experiments/exp3_neuron_replacement/replace_neurons.py
Table 6.9–6.11 (Data sources)	experiments/exp5_data_source_study/compare_sources.py
Table 6.12–6.14	utils/correlations.py, utils/replacement.py

Chapter 7: Physics-Constrained Learning

Primary code locations: FEATURE_VARIABLES_AS_PINN/ and FEATURE_VARIABLES_AS_NEURONS/

Note: Chapter 7 uses code from *two* folders due to the L1-PINN architecture developed in Chapter 6.

Chapter 7 – Figures and Tables

Item	Code/Data Location
<i>Main experiment scripts:</i>	
PINN experiments	FEATURE_VARIABLES_AS_PINN/run_pinn_experiments.py
Chapter 7 results	FEATURE_VARIABLES_AS_NEURONS/run_chapter7_experiments.py (cross-ref)
Figure 7.1 (PINN summary)	FEATURE_VARIABLES_AS_PINN/run_pinn_experiments.py
Figure 7.2 (Predictions)	FEATURE_VARIABLES_AS_PINN/utils/visualization.py
Figure 7.3 (Feature suitability)	FEATURE_VARIABLES_AS_PINN/run_pinn_experiments.py
Figure 7.4 (Physics ablation)	FEATURE_VARIABLES_AS_PINN/run_pinn_experiments.py
Figure 7.5 (Loss evolution)	FEATURE_VARIABLES_AS_PINN/run_pinn_experiments.py
Table 7.1 (PINN results)	FEATURE_VARIABLES_AS_PINN/run_pinn_experiments.py
Table 7.2 (Chapter comparison)	FEATURE_VARIABLES_AS_PINN/run_pinn_experiments.py
<i>Supporting code:</i>	
Physics residual computation	FEATURE_VARIABLES_AS_PINN/physics/residuals.py
Finite difference methods	FEATURE_VARIABLES_AS_PINN/physics/finite_differences.py
PINN model architecture	FEATURE_VARIABLES_AS_PINN/models/pinn_model.py
L1-PINN architecture	FEATURE_VARIABLES_AS_NEURONS/utils/l1pinn.py

Chapter 8: Identification of Flow Separation

Primary code location: IDENTIFY_FLOW_SEPARATION/

Chapter 8 – Figures and Tables

Item	Code/Data Location
<i>Main scripts:</i>	
All Chapter 8 figures	IDENTIFY_FLOW_SEPARATION/generate_chapter8_figures.py
All experiments	IDENTIFY_FLOW_SEPARATION/run_all_experiments.py
Fast experiments	IDENTIFY_FLOW_SEPARATION/run_experiments_fast.py
Figure 8.1 (Separation flow)	generate_chapter8_figures.py
Figure 8.2 (Classifier comparison)	utils/classifiers.py
Figure 8.3 (Generalization)	generate_chapter8_figures.py
Figure 8.4–8.5 (Hybrid)	generate_chapter8_figures.py
Figure 8.6 (Spatial detection)	generate_chapter8_figures.py
Table 8.1 (Distribution shift)	utils/labeling.py
Table 8.2 (Classifiers)	config.py
Table 8.3 (Baseline results)	run_all_experiments.py
Table 8.4 (Confusion matrix)	utils/classifiers.py
Table 8.5 (Minimal features)	utils/feature_selection.py
Table 8.6 (Physics constrained)	utils/physics_losses.py
Table 8.7–8.8	run_all_experiments.py

Chapter 9: OpenFOAM Integration and Comprehensive Evaluation

Primary code locations: OPENFOAM_INTEGRATION/ and BENCHMARKS/

Note: Chapter 9 validation figures use benchmark data and visualization scripts from the BENCHMARKS/ folder.

Chapter 9 – Figures and Tables

Item	Code/Data Location
<i>Main scripts:</i>	
Thesis validation figures	BENCHMARKS/generate_thesis_validation_figures.py
Model training	OPENFOAM_INTEGRATION/scripts/train_models.py
Model export	OPENFOAM_INTEGRATION/scripts/export_models.py
Experiment runner	OPENFOAM_INTEGRATION/scripts/run_experiments.py
Results evaluation	OPENFOAM_INTEGRATION/scripts/evaluate_results.py
<i>Benchmark validation (from BENCHMARKS):</i>	
Figure 9.2–9.7 (Validation)	BENCHMARKS/turbulent/*/openfoam/validate_*.py
BFS validation	BENCHMARKS/turbulent/backward_facing_step/openfoam/validate_bfs.py
Periodic hills	BENCHMARKS/turbulent/periodic_hills/openfoam/validate_periodic_hills.py
Wall hump	BENCHMARKS/turbulent/wall_mounted_hump/openfoam/validate_wall_hump.py
Thermal validation	BENCHMARKS/turbulent/*_thermal/openfoam/validate_*_thermal.py
<i>OpenFOAM integration (from OPENFOAM_INTEGRATION):</i>	
Figure 9.1 (Architecture)	scripts/export_models.py
Figure 9.8–9.9 (Timing)	scripts/evaluate_results.py
Figure 9.10–9.15 (3D cases)	scripts/run_experiments.py
Figure 9.16–9.22	scripts/evaluate_results.py
Table 9.1–9.14	OPENFOAM_INTEGRATION/scripts/evaluate_results.py BENCHMARKS/compare_with_benchmarks.py

Chapter 10: Conclusion and Future Work

Primary code location: ADDITIONAL_STUDIES/

Chapter 10 – Figures and Tables

Item	Code/Data Location
Table 10.1 (Method comparison)	ADDITIONAL_STUDIES/run_all_experiments.py
Table 10.2 (Method selection)	ADDITIONAL_STUDIES/compare_traditional_wf.py ADDITIONAL_STUDIES/evaluate_separation.py
<i>Supporting experiments in ADDITIONAL_STUDIES/:</i>	
WF vs no-WF study	exp1_wf_vs_nowf/train_with_wf.py
Attached vs separated	exp2_attached_to_separated/train_attached_only.py
Data augmentation	exp3_experimental_augmentation/train_with_augmentation.py
Distribution shift	exp4_distribution_shift/analyze_distributions.py
Reynolds analogy	exp5_reynolds_analogy/analyze_velocity_thermal.py
DNS/experimental data	exp6_real_dns_experimental/train_with_real_benchmarks.py

Key Scripts and Entry Points

The following scripts serve as main entry points for reproducing the key results:

Script	Purpose
<i>Data generation:</i>	
TRAINING_DATA/scripts/generate_Cases.py	CFD simulation cases
TRAINING_DATA/scripts/run_simulations.py	244 training simulations
TRAINING_DATA/scripts/extract_and_pair.py	pairing data pairs
<i>Chapter-specific experiments:</i>	
FEATURE_VARIABLES_AS_INPUTS/run_Chapter1_experiments	
FEATURE_VARIABLES_AS_NEURONS/run_Chapter2_experiments.py	
FEATURE_VARIABLES_AS_PINN/run_pINN_experiments.py	
IDENTIFY_FLOW_SEPARATION/run_allChapter3_experiments	
OPENFOAM_INTEGRATION/scripts/runChapter4_integration	
ADDITIONAL_STUDIES/run_all_experiments.py	supplementary studies
<i>Figure generation:</i>	
TRAINING_DATA/scripts/thesis_figures/generate_chapter3_figures.py	
FEATURE_VARIABLES_AS_INPUTS/exp_Chapter5_figureonly/chapter5_comprehensive_figures.py	
IDENTIFY_FLOW_SEPARATION/generate_chapter5_separation_figures.py	
BENCHMARKS/generate_thesis_validation_digitalisation.py	figures

Data Availability

Training Data

The complete training dataset comprising 244 simulation cases is available in:

- TRAINING_DATA/data/cases_WF/ – Wall-modelled (coarse mesh) simulations
- TRAINING_DATA/data/cases_WR/ – Wall-resolved (fine mesh) simulations

Each case directory contains:

- OpenFOAM case files (0/, constant/, system/)
- Converged solution fields (U, p, T, k, omega, nut)
- Post-processed wall quantities (wallShearStress, yPlus)
- Extracted stencil features (stencil_features.csv)

Benchmark Data

Experimental and DNS benchmark data used for validation:

- BENCHMARKS/turbulent/backward_facing_step/ – Driver & Seegmiller (1985)
- BENCHMARKS/turbulent/periodic_hills/ – Breuer et al. (2009)
- BENCHMARKS/turbulent/wall_mounted_hump/ – NASA wall-mounted hump
- BENCHMARKS/turbulent/diffuser/ – Buice & Eaton diffuser (1997)
- BENCHMARKS/turbulent/channel_flow/ – Moser, Kim & Mansour DNS (1999)
- BENCHMARKS/turbulent/flat_plate/ – Flat plate boundary layer
- BENCHMARKS/turbulent/*_thermal/ – Thermal benchmark cases
- BENCHMARKS/laminar/ – Laminar validation (Blasius, Pohlhausen)
- BENCHMARKS/transitional/ – Transitional flow (T3A, T3AM, SK)

Key benchmark processing scripts:

- BENCHMARKS/visualize_benchmarks.py – Master visualization script
- BENCHMARKS/compare_with_benchmarks.py – Comparison analysis
- BENCHMARKS/integrate_benchmark_data.py – Data integration
- BENCHMARKS/train_with_benchmarks.py – Benchmark-augmented training

Trained Models

Pre-trained neural network models are provided for direct use:

- FEATURE_VARIABLES_AS_INPUTS/models/ – Physics-feature models
- FEATURE_VARIABLES_AS_NEURONS/models/ – Interpretable models
- FEATURE_VARIABLES_AS_PINN/models/ – PINN models
- IDENTIFY_FLOW_SEPARATION/models/ – Separation classifiers
- OPENFOAM_INTEGRATION/models/ – Production-ready ONNX models

Software Dependencies

Software	Version	Purpose
OpenFOAM	v10	CFD simulations
Python	3.9+	ML training and analysis
PyTorch	2.0+	Neural network training
scikit-learn	1.0+	Traditional ML classifiers
NumPy	1.21+	Numerical computations
Matplotlib	3.5+	Visualisation
LibTorch	2.0+	C++ neural network inference
ONNX Runtime	1.12+	Cross-platform model deployment

Reproducibility

All experiments can be reproduced using the provided scripts. Random seeds are fixed for reproducibility. Complete instructions are provided in the repository README files.

For questions or issues regarding the code, please open an issue on the GitHub repository or contact the author.

Bibliography

1. A. Beck & M. Kurz. A Perspective on Machine Learning Methods in Turbulence Modelling (2020).
2. S. S. Girimaji. Turbulence closure modeling with machine learning approaches: A perspective (2023).
3. R. D. Moser, J. Kim & N. N. Mansour. Direct numerical simulation of turbulent channel flow up to $Re_\tau = 590$. *Physics of Fluids* **11**, 943–945 (1999).
4. O. Reynolds. On the dynamical theory of incompressible viscous fluids and the determination of the criterion. *Philosophical Transactions of the Royal Society of London A* **186**, 123–164 (1895).
5. J.-X. Wang, J. Wu, J. Ling, G. Iaccarino & H. Xiao. A Comprehensive Physics-Informed Machine Learning Framework for Predictive Turbulence Modeling (2017).
6. B. E. Launder & D. B. Spalding. The numerical computation of turbulent flows. *Computer Methods in Applied Mechanics and Engineering* **3**, 269–289 (1974).
7. D. B. Spalding. A single formula for the law of the wall. *Journal of Applied Mechanics* **28**, 455–458 (1961).
8. M. Wolfshtein. The velocity and temperature distribution in one-dimensional flow with turbulence augmentation and pressure gradient. *International Journal of Heat and Mass Transfer* **12**, 301–318 (1969).
9. T. von Kármán. Mechanische Ähnlichkeit und Turbulenz. *Nachrichten von der Gesellschaft der Wissenschaften zu Göttingen*, 58–76 (1930).
10. R. Pirayeshshirazinezhad. SPINN: An Optimal Self-Supervised Physics-Informed Neural Network Framework (2025).
11. Z. Su, Y. Liu, S. Pan, Z. Li & C. Shen. Finite Volume Physical Informed Neural Network (FV-PINN) with Reduced Derivative Order for Incompressible Flows (2024).
12. D. M. Driver & H. L. Seegmiller. Features of a reattaching turbulent shear layer in divergent channel flow. *AIAA Journal* **23**, 163–171 (1985).
13. J. Ling, A. Kurzawski & J. Templeton. Reynolds averaged turbulence modelling using deep neural networks with embedded invariance. *Journal of Fluid Mechanics* **807**, 155–166 (2016).
14. M. Raissi, P. Perdikaris & G. E. Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics* **378**, 686–707 (2019).
15. R. McConkey, E. Yee & F.-S. Lien. On the generalizability of machine-learning-assisted anisotropy mappings for predictive turbulence modelling (2022).
16. M. Milano & P. Koumoutsakos. Neural network modeling for near wall turbulent flow. *Journal of Computational Physics* **182**, 1–26 (2002).
17. L. Vu-Quoc & A. Humer. Deep learning applied to computational mechanics: A comprehensive review, state of the art, and the classics (2022).
18. K. Zdybał, G. D'Alessio, G. Aversano, M. R. Malik & A. Coussement. Advancing Reacting Flow Simulations with Data-Driven Models (2022).
19. P. Schlatter & R. Örlü. Assessment of direct numerical simulation data of turbulent boundary layers. *Journal of Fluid Mechanics* **659**, 116–126 (2010).
20. L. Guastoni, P. A. Srinivasan, H. Azizpour, P. Schlatter & R. Vinuesa. On the use of recurrent neural networks for predictions of turbulent flows (2020).
21. S. Bhattacharya, M. K. Verma & A. Bhattacharya. Predictions of Reynolds and Nusselt numbers in turbulent convection using machine-learning models (2022).
22. V. C. Leite, E. Merzari, R. Ponciroli & L. Ibarra. A Study on Convolution Neural Network for Reconstructing the Temperature Field of Wall-Bounded Flows (2022).
23. M. Chatzimanolakis, P. Weber & P. Koumoutsakos. Drag Reduction in Flows Past 2D and 3D Circular Cylinders Through Deep Reinforcement Learning (2023).
24. S. Raghu, R. Nayek & V. Chalamalla. Physics Informed Neural Networks for Free Shear Flows (2024).
25. L. Prandtl. Über Flüssigkeitsbewegung bei sehr kleiner Reibung. *Verhandlungen des III. Internationalen Mathematiker-Kongresses*, 484–491 (1904).
26. L. Guastoni, A. Güemes, A. Ianiro, S. Discetti & P. Schlatter. Convolutional-network models to predict wall-bounded turbulence from wall quantities (2020).
27. P. A. Srinivasan, L. Guastoni, H. Azizpour, P. Schlatter & R. Vinuesa. Predictions of turbulent shear flows using deep neural networks (2019).

28. H. Blasius. Grenzschichten in Flüssigkeiten mit kleiner Reibung. *Zeitschrift für Mathematik und Physik* **56**, 1–37 (1908).
29. G. Krishna, M. S. Nair, P. P. Nair & A. L. S. Physics-informed Neural Networks approach to solve the Blasius function (2022).
30. M. Lee & R. D. Moser. Direct numerical simulation of turbulent channel flow up to $Re_\tau \approx 5200$. *Journal of Fluid Mechanics* **774**, 395–415 (2015).
31. F. H. Clauser. Turbulent boundary layers in adverse pressure gradients. *Journal of the Aeronautical Sciences* **21**, 91–108 (1954).
32. B. S. Stratford. The prediction of separation of the turbulent boundary layer. *Journal of Fluid Mechanics* **5**, 1–16 (1959).
33. D. Coles. The law of the wake in the turbulent boundary layer. *Journal of Fluid Mechanics* **1**, 191–226 (1956).
34. Y. Mao & Y. Zhang. A Workflow for Utilizing OpenFOAM Data Structure in Physics-Informed Deep Learning Training (2024).
35. H. Li, Y. Lou & D. Xiao. Quantum machine learning for efficient reduced order modelling of turbulent flows (2025).
36. M. Breuer, N. Peller, C. Rapp & M. Manhart. Flow over periodic hills—numerical and experimental study in a wide range of Reynolds numbers. *Computers & Fluids* **38**, 433–457 (2009).
37. G.-M. Gie, Y. Hong, C.-Y. Jung & D. Lee. Singular Layer Physics-Informed Neural Network Method for Convection-Dominated Boundary Layer Problems in Two Dimensions (2023).
38. T. Anandh, D. Ghose, A. Tyagi, A. Gupta & S. Sarkar. An efficient hp-Variational PINNs framework for incompressible Navier-Stokes equations (2024).
39. B. A. Kader. Temperature and concentration profiles in fully turbulent boundary layers. *International Journal of Heat and Mass Transfer* **24**, 1541–1544 (1981).
40. Z. Y. Wang & W. W. Zhang. A unified method of data assimilation and turbulence modeling for separated flows at high Reynolds numbers (2022).
41. A. Ghaemi, A. Ebrahimi, M. Hajipour, S. M. M. Shobeiry & A. F. Lipaei. Model Predictive and Reinforcement Learning Methods for Active Flow Control of an Airfoil with Dual-point Excitation of Plasma Actuators (2025).
42. P. M. Milani, J. Ling & J. K. Eaton. Generalization of machine-learned turbulent heat flux models applied to film cooling flows (2019).
43. R. Pal, S. Mukherjee, U. Dutta & A. Choudhury. Solving Navier-Stokes Equations Using Data-free Physics-Informed Neural Networks With Hard Boundary Conditions (2025).
44. L. Jiang, Y. Cheng, K. Luo & J. Fan. PT-PINNs: A Parametric Engineering Turbulence Solver based on Physics-Informed Neural Networks (2025).
45. R. Laubscher & P. Rousseau. Application of mixed-variable physics-informed neural networks to solve normalised momentum and energy transport equations for 2D internal convective flow (2021).
46. T. Nakamura, K. Fukami & K. Fukagata. Identifying key differences between linear stochastic estimation and neural networks for fluid flow regressions (2021).
47. B. Yan, B. Chen, D. R. Harp & R. J. Pawar. A Robust Deep Learning Workflow to Predict Multiphase Flow Behavior during Geological CO₂ Sequestration Injection and Post-Injection Periods (2021).
48. G. Novati, H. L. de Laroussilhe & P. Koumoutsakos. Automating Turbulence Modeling by Multi-Agent Reinforcement Learning (2020).
49. M. Schmelzer, R. P. Dwight & P. Cinnella. Discovery of Algebraic Reynolds-Stress Models Using Sparse Symbolic Regression (2019).
50. R. D. Sanhueza, S. Smit, J. Peeters & R. Pecnik. Machine Learning for RANS Turbulence Modelling of Variable Property Flows (2022).
51. C. Pedersen, L. Zanna, J. Bruna & P. Perezhogin. Reliable coarse-grained turbulent simulations through combined offline learning and neural emulation (2023).
52. Z. Li, F. Montomoli & S. Sharma. Investigation of Compressor Cascade Flow Using Physics- Informed Neural Networks with Adaptive Learning Strategy (2023).
53. J. Balla, J. Bailey, A. Backour, E. Hofgard & T. Jaakkola. Implicit Augmentation from Distributional Symmetry in Turbulence Super-Resolution (2025).
54. C. B. Millikan. A critical discussion of turbulent flows in channels and circular tubes. *Proceedings of the Fifth International Congress of Applied Mechanics*, 386–392 (1938).

55. A. Haridas, N. R. Vadlamani & Y. Minamoto. Deep Neural Networks to Correct Sub-Precision Errors in CFD (2022).
56. H. S. Wong, W. X. Chan, B. H. Li & C. H. Yap. Multiple Case Physics-Informed Neural Network for Biomedical Tube Flows (2023).
57. O. Sallam & M. Fürth. Inference of water waves surface elevation from horizontal velocity components using physics informed neural networks (PINN) (2024).
58. M. A. Elhawary. Deep Reinforcement Learning for Active Flow Control around a Circular Cylinder Using Unsteady-mode Plasma Actuators (2020).
59. M. Chu & W. Qian. Uncertainty Quantification For Turbulent Flows with Machine Learning (2023).
60. M. Chu & W. Qian. Physics-Guided Machine Learning for Uncertainty Quantification in Turbulence Models (2025).
61. M. Matha & C. Morsbach. Extending turbulence model uncertainty quantification using machine learning (2022).
62. M. Matha & C. Morsbach. Physics-constrained Random Forests for Turbulence Model Uncertainty Estimation (2023).
63. K. Fukagata & K. Fukami. Compressing fluid flows with nonlinear machine learning: mode decomposition, latent modeling, and flow control (2025).
64. R. Ma & A. Lozano-Duran. Machine-learning wall-model large-eddy simulation accounting for isotropic roughness under local equilibrium (2024).
65. P. I. Karpov, C. Huang, I. Sitzdikov, C. L. Fryer & S. Woosley. Physics-Informed Machine Learning for Modeling Turbulence in Supernovae (2022).
66. N. Roy, R. Dürr, A. Bück & S. Sundar. Finite difference physics-informed neural networks enable improved solution accuracy of the Navier-Stokes equations (2024).
67. A. Man, M. Jadidi, A. Keshmiri, H. Yin & Y. Mahmoudi. Non-Unique Machine Learning Mapping in Data-Driven Reynolds Averaged Turbulence Models (2023).
68. H. S. de Ocáriz Borde, D. Sondak & P. Protopapas. Multi-Task Learning based Convolutional Models with Curriculum Learning for the Anisotropic Reynolds Stress Tensor in Turbulent Duct Flow (2021).
69. H. D. Pasinato & N. F. M. Reh. Modeling Turbulent Flows with LSTM Neural Network (2023).
70. M. Jaderberg, V. Dalibard, S. Osindero, W. M. Czarnecki & J. Donahue. Population Based Training of Neural Networks (2017).
71. R. Maulik, D. Fytanidis, B. Lusch, V. Vishwanath & S. Patel. PythonFOAM: In-situ data analyses with OpenFOAM and Python (2021).
72. V. Brehm, J. W. Austefjord, S. Lepadatu & A. Qaiumzadeh. A proposal for leaky integrate-and-fire neurons by domain walls in antiferromagnetic insulators (2022).
73. R. Deshpande, C. M. de Silva, M. Lee, J. P. Monty & I. Marusic. Data-driven enhancement of coherent structure-based models for predicting instantaneous wall turbulence (2021).
74. M. Yin, E. Ban, B. V. Rego, E. Zhang & C. Cavinato. Simulating progressive intramural damage leading to aortic dissection using an operator-regression neural network (2021).
75. T. Murata, K. Fukami & K. Fukagata. Nonlinear mode decomposition with convolutional neural networks for fluid dynamics (2019).
76. E. A. Illarramendi, M. Bauerheim & B. Cuenot. Performance and accuracy assessments of an incompressible fluid solver coupled with a deep Convolutional Neural Network (2021).
77. Y. Shu, Z. Cao, J. Gao, J. Wang & P. S. Yu. Omni-Training: Bridging Pre-Training and Meta-Training for Few-Shot Learning (2021).
78. D. de Oliveira Maionchi, L. Einstein, F. P. dos Santos & M. B. de Souza Júnior. Computational Fluid Dynamics and Machine Learning as tools for Optimization of Micromixers geometry (2022).
79. H. Frezat, J. L. Sommer, R. Fablet, G. Balarac & R. Lguensat. A posteriori learning for quasi-geostrophic turbulence parametrization (2022).
80. S. Hu, M. Liu, S. Zhang, S. Dong & R. Zheng. Physics-informed Neural Network Combined with Characteristic-Based Split for Solving Navier-Stokes Equations (2023).
81. Y. Zou, T. Li, L. Lu, J. Wang & S. Zou. Finite-difference-informed graph network for solving steady-state incompressible flows on block-structured grids (2024).
82. N. McGreivy & A. Hakim. Convolutional layers are equivariant to discrete shifts but not continuous translations (2022).

83. A. Rybchuk, M. Hassanaly, N. Hamilton, P. Doubrawa & M. J. Fulton. Ensemble flow reconstruction in the atmospheric boundary layer from spatially limited measurements through latent diffusion models (2023).
84. D. Dugal, J. J. Charney, M. Gallagher, C. Navasca & N. Skowronski. Exploring and Analyzing Wildland Fire Data Via Machine Learning Techniques (2023).
85. D. Wang, Z. Liang, Z. Zhang & M. Li. Efficient Estimation of the Convective Cooling Rate of Photovoltaic Arrays with Various Geometric Configurations: a Physics-Informed Machine Learning Approach (2024).
86. Z. Shi, S. M. H. Khorasani, H. Shin, J. Yang & S. Lee. Drag prediction of rough-wall turbulent flow using data-driven regression (2024).
87. N. Ashton, D. C. Maddix, S. Gundry & P. M. Shabestari. AhmedML: High-Fidelity Computational Fluid Dynamics Dataset for Incompressible, Low-Speed Bluff Body Aerodynamics (2024).
88. L. de Vito, N. Pinna & S. Dey. Implicit Neural Representation For Accurate CFD Flow Field Prediction (2024).
89. J. Suk, C. Brune & J. M. Wolterink. SE(3) symmetry lets graph neural networks learn arterial velocity estimation from small datasets (2023).
90. L. S. E. Jessica, N. A. Arifat, W. X. Lim, W. L. Chan & A. W. K. Kong. Finite Volume Features, Global Geometry Representations, and Residual Training for Deep Learning-based CFD Simulation (2023).
91. H. Tang, Y. Wang, T. Wang & L. Tian. Discovering explicit Reynolds-averaged turbulence closures for turbulent separated flows through deep learning-based symbolic regression with non-linear corrections (2023).
92. A. Sedykh, M. Podapaka, A. Sagingalieva, K. Pinto & M. Pfletsch. Hybrid quantum physics-informed neural networks for simulating computational fluid dynamics in complex shapes (2023).
93. T. Yao, E. Pajaziti, M. Quail, S. Schievano & J. A. Steeden. Image2Flow: A hybrid image and graph convolutional neural network for rapid patient-specific pulmonary artery segmentation and CFD flow field calculation from 3D cardiac MRI data (2024).
94. J. Cai, P.-E. Angelis, J.-M. Martinez, G. Damblin & D. Lucor. Revisiting Tensor Basis Neural Networks for Reynolds stress modeling: application to plane channel and square duct flows (2024).
95. H. Vardhan, U. Timalsina, M. Sandborn, D. Hyde & P. Volgyesi. Anvil: An integration of artificial intelligence, sampling techniques, and a combined CAD-CFD tool (2024).
96. A. H. Nobari, J. Rey, S. Kodali, M. Jones & F. Ahmed. Conformal Predictions Enhanced Expert-guided Meshing with Graph Neural Networks (2023).
97. Z. Wang, X. Meng, X. Jiang, H. Xiang & G. E. Karniadakis. Solution multiplicity and effects of data and eddy viscosity on Navier-Stokes solutions inferred by physics-informed neural networks (2023).
98. M. Elrefaie, S. Hüttig, M. Gladkova, T. Gericke & D. Cremers. Real-time and On-site Aerodynamics using Stereoscopic PIV and Deep Optical Flow Learning (2024).
99. M. Naderibeni, M. J. T. Reinders, L. Wu & D. M. J. Tax. Learning solutions of parametric Navier-Stokes with physics-informed neural networks (2024).
100. A. Warey, T. Han & S. Kaushik. Investigation of Numerical Diffusion in Aerodynamic Flow Simulations with Physics Informed Neural Networks (2021).
101. J. Sirignano, J. MacArt & K. Spiliopoulos. PDE-constrained Models with Neural Network Terms: Optimization and Global Convergence (2021).
102. X.-H. Zhou, J. Han & H. Xiao. Learning nonlocal constitutive models with neural networks (2020).
103. Z. Zhou, G. He & X. Yang. A wall model based on neural networks for LES of turbulent flows over periodic hills (2020).
104. M. Morimoto, K. Fukami, K. Zhang, A. G. Nair & K. Fukagata. Convolutional neural networks for fluid flow analysis: toward effective metamodeling and low-dimensionalization (2021).
105. C. Jiang. Physics-guided deep learning framework for predictive modeling of the Reynolds stress anisotropy (2021).
106. A. Arzani, J.-X. Wang & R. M. D'Souza. Uncovering near-wall blood flow from sparse data with physics-informed neural networks (2021).
107. I. K. Deo & R. Jaiman. Predicting waves in fluids with deep neural network (2022).
108. S. Goraya, N. Sobh & A. Masud. Error Estimates and Physics Informed Augmentation of Neural Networks for Thermally Coupled Incompressible Navier Stokes Equations (2022).
109. R. Gao & R. K. Jaiman. Predicting fluid-structure interaction with graph neural networks (2022).
110. J. Suk, P. de Haan, P. Lippe, C. Brune & J. M. Wolterink. Mesh Neural Networks for SE(3)-Equivariant Hemodynamics Estimation on the Artery Wall (2022).

111. S. Ephrati, P. Cifani & B. Geurts. Data-driven spectral turbulence modeling for Rayleigh-Bénard convection (2023).
112. B. Schulz & S. Lerch. Machine learning methods for postprocessing ensemble forecasts of wind gusts: A systematic comparison (2021).
113. J. Cai, P.-E. Angeli, J.-M. Martinez, G. Damblin & D. Lucor. Reynolds Stress Anisotropy Tensor Predictions for Turbulent Channel Flow using Neural Networks (2022).
114. R. Mao, M. Lin, Y. Zhang, T. Zhang & Z.-Q. J. Xu. DeepFlame: A deep learning empowered open-source platform for reacting flow simulations (2022).
115. T. Nakamura & K. Fukagata. Robust training approach of neural networks for fluid flow state estimations (2021).
116. S. Rodriguez & P. Cardiff. A general approach for running Python codes in OpenFOAM using an embedded pybind11 Python interpreter (2022).
117. C.-W. Chang & N. T. Dinh. Reynolds-Averaged Turbulence Modeling Using Type I and Type II Machine Learning Frameworks with Deep Learning (2018).
118. H. Ghraieb, J. Viquerat, A. Larcher, P. Meliga & E. Hachem. Single-step deep reinforcement learning for open-loop control of laminar and turbulent flows (2020).
119. Z. Xiang, W. Peng, X. Zheng, X. Zhao & W. Yao. Self-adaptive loss balanced Physics-informed neural networks for the incompressible Navier-Stokes equations (2021).
120. X. He, Y. Wang & J. Li. Flow Completion Network: Inferring the Fluid Dynamics from Incomplete Flow Information using Graph Neural Networks (2022).
121. M. Quattromini, M. A. Bucci, S. Cherubini & O. Semeraro. Enhancing Data-Assimilation in CFD using Graph Neural Networks (2023).
122. R. Han, Y. Wang, Y. Zhang & G. Chen. A new prediction method of unsteady wake flow by the hybrid deep neural network (2019).
123. A. Scillitoe, P. Seshadri & M. Girolami. Uncertainty Quantification for Data-driven Turbulence Modelling with Mondrian Forests (2020).
124. C. Xie, X. Xiong & J. Wang. Artificial neural network approach for turbulence models: A local framework (2021).
125. J. Han, X.-H. Zhou & H. Xiao. An equivariant neural operator for developing nonlocal tensorial constitutive models (2022).
126. A. Vahdat & J. Kautz. NVAE: A Deep Hierarchical Variational Autoencoder (2020).
127. C. Wang, Q. Gao, B. Wang, C. Pan & J. Wang. Vortex-to-velocity reconstruction for wall-bounded turbulence via a data-driven model (2020).
128. M. I. Zafar, H. Xiao, M. M. Choudhari, F. Li & C.-L. Chang. Convolutional Neural Network for Transition Modeling Based on Linear Stability Theory (2020).
129. M. Chu & W. Qian. Machine learning based uncertainty quantification of turbulence model for airfoils (2022).
130. X. Xue, S. Wang, H.-D. Yao, L. Davidson & P. V. Coveney. Physics informed data-driven near-wall modelling for lattice Boltzmann simulation of high Reynolds number turbulent flows (2024).
131. M. I. Zafar, X. Zhou, C. J. Roy, D. Stelter & H. Xiao. Data-Driven Turbulence Modeling Approach for Cold-Wall Hypersonic Boundary Layers (2024).
132. M. Matha & K. Kucharczyk. Applicability of machine learning in uncertainty quantification of turbulence models (2022).
133. A. Prakash, K. E. Jansen & J. A. Evans. Invariant Data-Driven Subgrid Stress Modeling on Anisotropic Grids for Large Eddy Simulation (2022).
134. A. G. Özbay & S. Laizet. FR3D: Three-dimensional Flow Reconstruction and Force Estimation for Unsteady Flows Around Extruded Bluff Bodies via Conformal Mapping Aided Convolutional Autoencoders (2023).
135. X. Fu, S. Fu, C. Liu, M. Zhang & Q. Hu. Data-driven approach for modeling Reynolds stress tensor with invariance preservation (2023).
136. B. Font, F. Alcántara-Ávila, J. Rabault, R. Vinuesa & O. Lehmkuhl. Active flow control of a turbulent separation bubble through deep reinforcement learning (2024).
137. M. Fiore, E. Saccaggi, L. Koloszar, Y. Bartosiewicz & M. A. Mendez. Data-driven turbulent heat flux modeling with inputs of multiple fidelity (2024).
138. S. Barwey, R. Balin, B. Lusch, S. Patel & R. Balakrishnan. Scalable and Consistent Graph Neural Networks for Distributed Mesh-based Data-driven Modeling (2024).

139. P. Garnier, V. Lannelongue, J. Viquerat & E. Hachem. MeshMask: Physics-Based Simulations with Masked Graph Neural Networks (2025).
140. A. M. Santamaria, T. Buchanan, F. Fico, I. Langella & R. P. Dwight. Data-driven turbulence modelling for magnetohydrodynamic flows in annular pipes (2025).
141. C. Wu, S. Zhang, C. Guo & Y. Zhang. Data-driven Turbulence Modeling for Separated Flows Considering Non-Local Effect (2025).
142. J. Afful. A Review of HPC-Accelerated CFD in National Security and Defense (2025).
143. Z. Cao, M. Li, F. Lin, J. Jia & Y. Wen. Transforming Future Data Center Operations and Management via Physical AI (2025).
144. K. Buck & R. Temam. Convergence Properties of PINNs for the Navier-Stokes-Cahn-Hilliard System (2025).
145. M. Yagoubi, D. Danan, M. Leyli-Abadi, A. Mazari & J.-P. Brunet. NeurIPS 2024 ML4CFD Competition: Results and Retrospective Analysis (2025).
146. A. S. Nair, N. Singh, M. Panesi, J. Sirignano & J. F. MacArt. Physics-Based Machine Learning Closures and Wall Models for Hypersonic Transition-Continuum Boundary Layer Predictions (2025).
147. N. Wang, Y. Chen & S. Zheng. FluidFormer: Transformer with Continuous Convolution for Particle-based Fluid Simulation (2025).
148. N. Shah. Computational Fluid Dynamics Optimization of F1 Front Wing using Physics Informed Neural Networks (2025).
149. M. Ahangarkiasari & H. Pouraria. Multi-Stage Graph Neural Networks for Data-Driven Prediction of Natural Convection in Enclosed Cavities (2025).
150. H. Geshani, M. R. Dehkordi & M. S. Panahi. Physics-Informed Machine Learning Approach in Augmenting RANS Models Using DNS Data and DeepInsight Method on FDA Nozzle (2025).
151. X. Liu, T. Hickling & J. F. MacArt. Active Control of Turbulent Airfoil Flows Using Adjoint-based Deep Learning (2025).
152. R. Falga, S. Shamekh & L. Zanna. Towards a Unified Data-Driven Boundary Layer Momentum Flux Parameterization for Ocean and Atmosphere (2025).
153. B. Choi, M. Ugliotti, M. Reynoso, D. R. Gurevich & R. O. Grigoriev. Data-driven modeling of multiscale phenomena with applications to fluid turbulence (2025).
154. X. K. Lee, A. Ramadhan, A. Souza, G. L. Wagner & S. Silvestri. NORi: An ML-Augmented Ocean Boundary Layer Parameterization (2025).
155. B. Barman, D. Chatterjee & R. K. Ray. PhysicsFormer: An Efficient and Fast Attention-Based Physics Informed Neural Network for Solving Incompressible Navier Stokes Equations (2026).
156. T. Morimoto. Rapid Prediction of Three-Dimensional Scour Flow around Bridge Piers via Body-Fitted Coordinate-Based U-Net (2026).
157. P. Cinnella. Data-driven turbulence modeling (2024).
158. K. Agyei-Baah, M. R. Rahman & E. R. Smith. A Convolutional Neural Network Based Framework for Linear Fluid Dynamics (2026).
159. F. Orozco, P. P. B. de Gusmão, H. Wen, J. Wahlström & M. Luo. Federated Learning for Traffic Flow Prediction with Synthetic Data Augmentation (2024).
160. P. Garnier, J. Viquerat & E. Hachem. Multi-Grid Graph Neural Networks with Self-Attention for Computational Mechanics (2024).
161. I.-G. Farcaș, D. L. C. A. Fernando, A. B. Navarro, G. Merlo & F. Jenko. Machine Learning for Electron-Scale Turbulence Modeling in W7-X (2025).
162. J. Suk, D. Alblas, B. A. Hutten, A. Wiegman & C. Brune. Physics-informed graph neural networks for flow field estimation in carotid arteries (2024).
163. S. Dutta, N. Innan, S. B. Yahia & M. Shafique. AQ-PINNs: Attention-Enhanced Quantum Physics-Informed Neural Networks for Carbon-Efficient Climate Modeling (2024).
164. C. Wu, S. Zhang & Y. Zhang. Data-driven Augmentation of a Turbulence Model in Three-dimensional Separated Flows (2025).
165. S. Mukherjee. Graph Neural Network Assisted Genetic Algorithm for Structural Dynamic Response and Parameter Optimization (2025).
166. A. Wu & S. K. Lele. Subgrid Stress Modelling with Multi-dimensional State Space Sequence Models (2025).
167. Y. Ling, I. Hayat, K. Goc & A. Lozano-Duran. General-purpose Data-driven Wall Model for Low-speed Flows Part I: Baseline Model (2025).

168. K. Bounja, L. Laayouni & A. Sakat. KD-PINN: Knowledge-Distilled PINNs for ultra-low-latency real-time neural PDE solvers (2025).
169. H. Wen, F. Luo, S. Xu & B. Wang. JANC: A cost-effective, differentiable compressible reacting flow solver featured with JAX-based adaptive mesh refinement (2025).
170. M. B. Hasan, M. Yu & T. Oates. Invariance-embedded Machine Learning Sub-grid-scale Stress Models for Meso-scale Hurricane Boundary Layer Flow Simulation I: Model Development and *a priori* Studies (2025).
171. M. H. Parikh, X. Fan & J.-X. Wang. Conditional flow matching for generative modeling of near-wall turbulence with quantified uncertainty (2025).
172. D. Igarashi, S. Kumagai, Y. Yokoyama, Y. Jingzu & M. Horie. Reconstruction of three-dimensional fluid stress field via photoelasticity using physics-informed convolutional encoder-decoder (2025).
173. K. Jigjid, A. Eidi, N. A. K. Doan & R. P. Dwight. Discovery of a Physically Interpretable Data-Driven Wind-Turbine Wake Model (2025).
174. Z. Yang, Y. Bin, Y. Shi & X. I. A. Yang. Large Language Model Driven Development of Turbulence Models (2025).