공학박사학위논문

Efficient Aerodynamic Design via Data-driven Approaches

데이터 기반 기법을 통한 공력 설계 효율화

2023년 8월

서울대학교 대학원

항공우주공학과

양 선 웅

Efficient Aerodynamic Design via Data-driven Approaches

데이터 기반 기법을 통한 공력 설계 효율화

지도교수 이 관 중

이 논문을 공학박사 학위논문으로 제출함

2023년 5월

서울대학교 대학원

항공우주공학과

양 선 웅

양선웅의 공학박사 학위논문을 인준함

2023년 6월

위 원 장	김 현 진
부위원장	이 관 중
위 원	강 남 우
위 원	정 신 규
위 원	이 상 아

Abstract

Efficient Aerodynamic Design via Data-driven Approaches

Sunwoong Yang
Department of Aerospace Engineering
The Graduate School
Seoul National University

The development of high-performance computing (HPC) technologies has led to increased interest in accelerating the aerodynamic design process, which has been hindered by its demanding computational cost. Despite advances in computational methods and simulation techniques, the aerodynamic design process is still burdensome and remains a significant bottleneck. The iterative nature of the process, coupled with the need for high-fidelity computational fluid dynamics solvers makes it challenging to achieve short turnaround times and creative design exploration. To address these challenges, regression models trained on existing datasets have gained popularity as a way to replace expensive flow simulations or experiments. They help mitigate the overuse of HPC by improving the efficiency of the design process, from simple prediction of quantities of interest to design optimization and knowledge extraction.

However, there are several bottlenecks that limit the use of regression models in the aerodynamic design process. First, aerodynamic design often requires a high-dimensional input space and therefore, traditional regression models suffer from the curse of dimensionality. As the number of input variables increases, the number of possible configurations within the input space grows exponen-

tially. This leads to a sparsity problem, where the number of available data points is insufficient to cover the high-dimensional space. Second, most popular regression models are designed to predict a single output, which limits their application in high-dimensional output spaces. In multi-output regression tasks, they are trained independently for each output, so that the required training time increases linearly with the output dimension, and the correlations within the outputs are completely ignored. Finally, reliable and efficient uncertainty quantification (UQ) should be coupled with regression models to account for the inherent risk associated with the aerodynamic design process. In the context of regression models, since there is always a chance that the predicted values will not match the actual values, UQ is particularly crucial to provide information about the reliability or trustworthiness of their predictions. There are popular regression models for UQ, such as Gaussian process regression and Bayesian neural networks, but their computational complexity and inefficiency prevent them from being viable options for engineers who prioritize practicality, highlighting the need for a reliable and efficient regression model for UQ.

In this regard, the contributions of the dissertation can be summarized as follows:

1. The high-dimensional input space in aerodynamic design is reduced to the low-dimensional latent space using dimensionality reduction (DR) techniques. By alleviating the curse of dimensionality with DR techniques, the feasibility of applying gradient-free optimizers to the reduced input space is also investigated. For this purpose, inverse design optimization of the wind turbine airfoil is adopted as a case study. Finally, it is proved that the original high-dimensional input space can be successfully reduced using a deep learning-based DR technique, and also the genetic algorithm is applied within its framework to validate its feasibility to be coupled

with gradient-free optimizers.

2. The prediction of high-dimensional data is performed using reduced-order modeling (ROM) techniques. The DR process is required in its procedure to construct the latent space and this study focuses on the fact that since it acts as an intermediary in ROM for predicting high-dimensional data, the latent space inevitably affects ROM performance. In this regard, the prediction of flow fields around a transonic airfoil is adopted as a case study, and the physical interpretability of the latent space constructed by various machine learning-based DR techniques is investigated. Furthermore, the impact of its interpretability on ROM performance is analyzed, and finally, its significance for the accuracy and efficiency of predicting high-dimensional output space via ROM is validated.

3. Regression model capable of reliable and efficient UQ is investigated. To this end, the deep ensembles (DE) approach is analyzed in the task of predicting missile aerodynamic performance. This simple and scalable DE method is validated for multi-output regression tasks. Also, since the poor reliability of the uncertainty quantified by DE is observed, a simple post-hoc calibration method is applied to DE models to correct the unsatisfactory uncertainty quality. The results show that the DE technique can perform more reliable and efficient UQ compared to GPR, which is most commonly used in engineering fields. Finally, the impact of the proposed calibration method on DE-based Bayesian optimization is investigated.

Keywords: Aerodynamic Design, Data-driven Approaches, Regression Modeling, Generative Modeling, Reduced-order Modeling, Uncertainty Quantification
Student Number: 2020-34020

Contents

Abstract **1**

1 Introduction **15**

1.1 Motivation and objectives 15

1.2 Contributions of the dissertation 22

1.3 Overview of the dissertation 24

2 High-dimensional input space **25**

2.1 Introduction 25

2.2 Methodologies 29

2.2.1 Multi-layer perceptron (MLP) 29

2.2.2 Autoencoder (AE) 30

2.2.3 Variational autoencoder (VAE) 31

2.3 Inverse design optimization framework 34

2.3.1 Two-step deep learning approach 34

2.3.2 Target distribution optimization 34

2.3.3 Active learning and transfer learning 35

2.4 Framework validation: optimization of the airfoil for wind turbine blades 38

2.4.1 Optimization of the airfoil in wind turbine blades 38
2.4.2 Architectures of the two-step deep learning models 43
2.4.3 Single-objective optimization results and discussion 46
2.4.4 Multi-objective optimization results and discussion 51
2.5 Summary 59
2.6 Additional results 61

3 High-dimensional output space 62
3.1 Introduction 62
3.2 β-variational autoencoder (β-VAE) 68
3.3 Physics-aware reduced-order modeling 70
3.4 Numerical experiments 74
3.4.1 Data preparation 74
3.4.2 Training details 75
3.5 Results and discussion 79
3.5.1 Training results 79
3.5.2 Independence of LVs 81
3.5.3 Information intensity of LVs 82
3.5.4 Physics-awareness of LVs 87
3.5.5 Physics-aware ROM 93
3.6 Summary 99
3.7 Additional results 101
3.7.1 POD results 101
3.7.2 Scalability of extracting physics-aware LVs in practical problem 103

4 Reliable and efficient uncertainty quantification 106
4.1 Introduction 106

4.2 Implementation and evaluation of DE . . . 112
4.2.1 Deep ensembles (DE) . . . 112
4.2.2 Uncertainty quality evaluation . . . 116
4.2.3 Uncertainty calibration: STD scaling . . . 121
4.3 Application of DE to aerodynamic performance regression task . 125
4.3.1 Data preparation and training details . . . 125
4.3.2 Evaluation of regression performance . . . 127
4.3.3 Evaluation of UQ performance . . . 129
4.3.4 Theoretical derivation: underconfidence of DE in regression tasks . . . 131
4.4 DE models with STD calibration . . . 135
4.4.1 STD calibration of DE models . . . 135
4.4.2 Effects of STD calibration on Exploratory Behavior in Bayesian optimization . . . 139
4.5 Summary . . . 143
4.6 Additional results . . . 146
4.6.1 Controversial issues on MC-dropout . . . 146
4.6.2 Hyperparameter tuning results in Sec. 4.3.1 . . . 146
4.6.3 Additional results in Sec. 4.3.3 . . . 149
4.6.4 Additional results in Sec. 4.4.1 . . . 149

5 Concluding remarks **152**
5.1 Summary of the dissertation . . . 152
5.2 Limitations of the dissertation . . . 156
5.3 Embarking on a journey towards acceleration of 3D aerodynamic simulations . . . 159

6 References **160**

국문 초록 **187**

List of Figures

2.1 Flowchart of the two-step deep learning approach. 34

2.2 Flowchart of the inverse design optimization framework. 37

2.3 Comparison of pressure distributions from Xfoil and experimental results in Ref. [1] (adopted airfoil configuration is also visualized). 39

2.4 Shape parameters for airfoil representation: six PARSEC parameters are used. 40

2.5 Architecture of the VAE. 46

2.6 Convergence history of single-objective optimization with active learning. 47

2.7 Loss history of (a) MLP, and (b) VAE. For both models, the history of the first iteration and last (24th) iteration of active learning is represented. 47

2.8 Comparison of the baseline and optimum airfoil shape of single-objective optimization. 48

2.9 Comparison of the generated and calculated pressure distribution of the optimum airfoil (baseline pressure distribution is also included). 49

2.10 Comparison of 50 randomly selected C_p training data (black lines, a) and 50 generated C_p distributions by the VAE (red lines, b). The black dashed box near the leading-edge of the lower curve indicates clear distinctions between generated distributions. . . . 50

2.11 Loss history of (a) MLP, and (b) VAE. For both models, the history of the first iteration and last (59th) iteration of active learning is represented. 52

2.12 Pareto solutions of multi-objective optimization. The discontinuity in the Pareto solutions is due to C_d constraint violation. . . . 53

2.13 Airfoil shape comparison of six selected Pareto solutions. 53

2.14 Comparison of generated and calculated C_p distributions of six selected Pareto solutions. 55

2.15 Heatmaps of two objective functions within the latent space: (a) L/D and (b) area. Twelve points are selected to investigate the rapid change at $z_2 \approx 0.55$ (top), and the latent space of six selected Pareto solutions is shown in the heatmap of area (b). . . 56

2.16 C_p distributions of 12 points selected in Fig. 2.15. 58

2.17 Trends in the leading-edge radius ($R_{L.E.}$) of 12 selected points in Fig. 2.15. 58

2.18 Pareto solutions of multi-objective optimization without C_d constraint. For comparison with Fig. 2.12, six designs previously selected from the Pareto solutions with C_d constraint are also shown. 61

3.1 Overall structure of physics-aware reduced-order modeling. . . . 70

3.2 Illustrative schematic showing the process of extracting physics-aware LVs by β-VAE: the ideal case is to extract the actual physical parameters (Ma and AoA) from the given dataset. . . . 72
3.3 Computational grid used for the flow analysis; structured O-grid with a size of 512×256. 75
3.4 Structures of the AE and VAE/β-VAE. 75
3.5 Loss history of the trained AE/VAE/β-VAE models. 80
3.6 MSE and KL-divergence of the trained VAE/β-VAE models. . . 80
3.7 Reconstructed pressure fields of the trained models. 81
3.8 Absolute values of the components in the Pearson correlation matrix for LVs. 83
3.9 Determinants of Pearson correlation matrices for combinations of 2 to 7 LVs. 83
3.10 KL-divergence and Sobol results with respect to LVs from the training dataset. 85
3.11 Standard deviations of LVs from the training dataset. 86
3.12 Latent traversal plots of pressure flow fields for two extreme LVs: first (most dominant) and last (most trivial) LVs ranked by KL-divergence. 87
3.13 Investigation of physical features contained in the top two LVs: (a) distributions of training dataset and boundary data with respect to Ma and AoA, and (b) distributions of training dataset for 1st and 2nd LVs (the left figures are colored by Ma, and the rights by AoA). 90
3.14 The results of the single variable LR: (a) Ma=$f(LV_{Ma})$, and (b) AoA=$f(LV_{AoA})$. 92

3.15 Latent traversal plots of airfoil surface pressure distributions in 1000-VAE: (a) traversal of LV_{Ma}, and (b) traversal of LV_{AoA}. . . 93

3.16 MSE of the regression models in ROM. 94

3.17 Comparison of the response surface of two LVs in the 1000-VAE: (a) physics-aware LV, (b) physics-unaware LV. 95

3.18 MSE of ROM prediction with the exclusion of k^{th} LV. 96

3.19 Comparison of prediction MSE between physics-aware ROM and physics-unaware ROM. 97

3.20 Pressure contour predicted from AE/β-VAE-based ROMs: (a) prediction, (b) absolute error. 98

3.21 Pressure contour predicted from POD-based ROM: (a) prediction, (b) absolute error. 102

3.22 Latent traversal plots of airfoil surface pressure distributions in POD: (a) traversal of 1^{st} LV, and (b) traversal of 2^{nd} LV. 102

3.23 Preprocessed training dataset consisting of (a) 32 surface pressure values, (b) C_l, (c) C_d, and (d) C_m. 104

3.24 Investigation of physical features contained in the top two LVs for sparse and noisy datasets. 105

4.1 Flowchart of Bayesian optimization. 108

4.2 Flowchart of DE approach. 116

4.3 (a) Illustration of well-calibrated/miscalibrated models: 60% CI of the well-calibrated model contains 60% of the test data, whereas that of the underconfident and overconfident model contains 80% and 40% of the data, respectively. (b) Illustration of CI-based reliability plot. 118

4.4 Illustration of error-based reliability plot. Underconfident model overestimates RMV relative to RMSE, while overconfident model underestimates RMV. The ideal model estimates the equivalent RMV and RMSE as the $y = x$ black dashed line. 121
4.5 Loss history of all trained models. NLL calculated by the test dataset is adopted the results of the hyperparameter tuning (Table 4.3 in Sec. 4.6.2). 126
4.6 Comparison of regression accuracy between GPR and DE-2: kernel density estimation (KDE) of test dataset with respect to NLL and RMSE (averaged values of all six QoIs). The stars and circles represent the maximum and median points of each model, respectively. 128
4.7 Comparison of regression accuracy between GPR and all DE models: comprehensive results in terms of all aerodynamic QoIs. (a) NLL, (b) RMSE. 129
4.8 Reliability plots of GPR: (a) CI-based reliability plot, (b) Error-based reliability plot. 130
4.9 Reliability plots of DE: for simplicity, only the C_{SF} results of different DE models are shown. (a) CI-based reliability plot, (b) Error-based reliability plot. In (b), to clearly show the decreasing tendency of UQ quality with increasing M, the linear regression model of the scatter points of each DE model is shown as a dashed line with the corresponding color. 132
4.10 Reliability plots of DE after STD calibration: (a) CI-based reliability plot, (b) Error-based reliability plot. The noticeable effects of STD calibration can be found when compared with the corresponding figure before STD calibration, Fig. 4.9. 137

4.11 AUCE and ENCE of DE models before and after STD calibration. Those of GPR are also shown for comparison. 138
4.12 CIs of 68% confidence level predicted by DE-16: comparison between before and after STD calibration. 140
4.13 Effects of STD calibration for DE models on Bayesian optimization results. 142
4.14 Reliability plots of vanilla DE models: (left) CI-based reliability plots, (right) error-based reliability plots. 150
4.15 Reliability plots of DE models after STD calibration: (left) CI-based reliability plots, (right) error-based reliability plots. 151

List of Tables

2.1 Design space of the six airfoil shape parameters: the baseline airfoil is selected as the median value of each range 41

2.2 Flight conditions, objective functions, and constraints for single-objective and multi-objective optimizations 42

2.3 Summary of the QoIs of the optimum solution 49

2.4 Summary of the QoI of six selected Pareto solutions 54

2.5 Nomenclatures of twelve points extracted to investigate the sharp changes in the QoI heatmaps 56

3.1 Details of the blocks and layers of VAE/β-VAE used in this study. 76

3.2 Network structure of the VAE/β-VAE used in this study. 76

4.1 Optimized scaling factors for STD calibration 136

4.2 Comprehensive comparison between GPR and DE-2 144

4.3 Results of hyperparameter tuning: several structures of probabilistic NN used in the DE model are tested. 147

4.4 Results of hyperparameter tuning: several GPR models are tested. 148

Chapter 1

Introduction

1.1 Motivation and objectives

The advent of high-performance computing (HPC) technologies has had a significant impact on the fields of aerospace design by dramatically reducing the time required to acquire vast amounts of data. With the ability to handle large-scale scientific computations with ease, these HPC systems have greatly advanced the capabilities of computational fluid dynamics (CFD). In fact, NASA has recognized the pivotal role of HPC in enhancing the impact of CFD on the aerospace design process by providing improved understanding and insight into critical physical phenomena [2]. Progress in data storage technologies has also made it possible to store large datasets, including simulation results, historical data, and auxiliary data augmented from existing data, which can be utilized to enhance the reliability and efficiency of the traditional aerospace design process. In essence, the combination of HPC technologies and the resulting big datasets has revolutionized the aerospace design process by

enabling engineers to explore creative and superior design candidates, thereby improving overall performance.

Despite the development of advanced computational methods and simulation techniques, the computational cost of the aerodynamic design process still remains a significant bottleneck. The iterative nature of its process makes it challenging to achieve short turnaround times and creative exploration within the design space. It becomes even worse when coupled with the repetitive execution of expensive flow simulations such as Reynolds-averaged Navier-Stokes (RANS) or large eddy simulation (LES), which can significantly increase the computational time and resources required for the design process. It should be noted that even RANS has limitations in reliably and accurately capturing turbulent flows with significant regions of separation [2], highlighting that there are still limitations to using HPC naively in aerodynamic design. Nevertheless, the benefits of well-designed objects with enhanced aerodynamic efficiency make it imperative to invest resources in conducting an accurate but demanding aerodynamic analysis. As a result, aerospace engineers are constantly looking for new ways to reduce the computational cost of aerodynamic analysis while leveraging high-fidelity CFD results to improve the accuracy of the aerodynamic design process.

To overcome this bottleneck, researchers have investigated various techniques to alleviate the computational cost of aerodynamic design. To name a few, they have attempted to utilize multi-fidelity flow solvers [3, 4, 5, 6], parallel computing [7, 8, 9], and advanced optimization algorithms [10, 11, 12, 13, 14, 15] to speed up the flow field calculations or improve the design process efficiency. Last but not least, one of the most popular approaches is to simply replace the implementation of flow simulation or experiment by training a regression model (also known as a surrogate model) on a given dataset and then using it

to predict quantities of interest (QoIs) [16, 17, 18]. The regression model aims to mitigate the naive overuse of HPC in the aerodynamic design process, where computational cost is still limited and burdensome, by exploiting already obtained datasets. This data-driven approach can be extensively extended: from design space exploration during the design optimization process [19] to the prediction of high-dimensional fields by reduced-order modeling (ROM) [20]. Even when there are multiple data sources due to different fidelities of data collection processes, regression models can also be utilized to fuse them [21]. There are a variety of regression models that engineers can use; to name a few, radial basis function (RBF) [22], support vector regression (SVR) [23], Gaussian process regression (GPR) [24], and multilayer perceptron (MLP) [25]. From the perspective that the regression model can facilitate the realization of digital twins by replacing the demanding simulations required within their process [26], its potential seems boundless.

There are numerous studies that leveraged regression models to accelerate the aerodynamic design procedure, and their use can be categorized into three groups: 1) prediction of QoIs, 2) design optimization, and 3) knowledge extraction. First, the prediction of QoIs based on regression models can be considered as the simplest application. Espinosa Barcenas et al. [27] used MLP to predict aerodynamic coefficients of various airfoil and wing configurations. Balla et al. [28] also used MLP to predict not just the aerodynamic coefficients, but also the pressure distribution on the surface of airfoils and wings. Sun et al. [29] conducted the inverse design of airfoil and wing configurations based on MLP, which directly provides airfoil shapes that fit the required aerodynamic characteristics.

Second, regression models can be used to speed up the aerodynamic optimization process by serving as cheap-to-evaluate functions. Lee et al. [30] ap-

plied cluster-based GPR to mitigate prediction errors in composite rotor blade optimization problems with highly nonlinear QoIs. Chae et al. [31] leveraged GPR for helicopter rotor shape optimization to improve hover aeroacoustic performance. Song and Keane [32] built GPR model and then conducted a multi-objective genetic algorithm considering the aerodynamic performance and noise effects of a three-dimensional subsonic engine nacelle.

Finally, regression models can be leveraged for knowledge extraction from the given dataset, as the data can be massively augmented by using previously trained models. This means that engineers can gain insights from large datasets that are augmented by previously trained models, rather than by collecting new data. In this regard, Martins and Ning [21] noted that regression models are helpful when we want to understand the design space, that is, how outputs vary with respect to inputs. In fact, there are a number of studies that have exploited regression models to extract insights from the datasets they are interested in. Obayashi et al. [33] performed data mining techniques to efficiently explore the design space of a fly-back booster and regional jet wing. Obayashi et al. [34] demonstrated that design knowledge discovered from visual data mining techniques can lead to an improved regional jet wing design even after a brief design exploration. Also, Kanazaki et al. [35] derived the design rules in a three-element airfoil that the gap and the deflection of the flap have a notable effect on the lift coefficient during landing and near-stall conditions.

However, there are several bottlenecks that hinder the application of regression models to the aerodynamic design process. First, aerodynamic design often demands high-dimensional input (design) space due to high geometric deformation freedom and various flight conditions. For example, Lyu et al. [36] noted that at least 200 design variables are required to take full advantage of aerodynamic shape optimization in transonic wing design. With this high-dimensional

input space, traditional regression models encounter the curse of dimensionality. As the number of input variables increases, the number of possible configurations within the input space grows exponentially, making it difficult to explore the design space thoroughly. This leads to a sparsity problem, where the number of available data points is insufficient to cover the high-dimensional space, preventing accurate capture of the complex relationships between input and output variables. Furthermore, also in terms of optimization, high-dimensional design space makes it infeasible to apply gradient-free optimizers, which typically require a massive number of evaluations to find reasonable solutions [37, 38]. Considering the fact that gradient-free optimizers are adopted by numerous engineers due to their ability to obtain global optimal solutions in discontinuous and multimodal problems [16], the infeasibility of gradient-free optimizers can be considered as an obstacle that we should overcome in order to fully exploit the advantages of regression models. Finally, in terms of knowledge extraction, high-dimensional input spaces reduce the ease of physical interpretation due to the complex relationships between a large number of design variables.

Second, most of the popular regression models, such as RBF, SVR, and GPR, are devised for predicting a single output, limiting their use in high-dimensional output spaces. More specifically, in multi-output regression tasks, these models are trained independently for each output, so that the following two drawbacks exist. One is that the required training time increases linearly with the output dimension, which indicates severely degraded efficiency especially in high-dimensional output tasks. The other is that the correlations within the outputs are completely ignored [39], so the complex and intricate physical systems in the real world cannot be accurately modeled. In this context, it has been demonstrated that multi-output regression methods generally yield better predictive performance when compared to single-output methods [40, 41, 42].

Considering that the trend of regression tasks in aerodynamics is moving from the simple multi-output predictions (e.g., prediction of C_l, C_d, and C_m) to the prediction of high-dimensional output data such as pressure and velocity fields around the objects [43, 44, 45], there should be a way to break through this barrier to the prediction of high-dimensional output spaces.

Finally, reliable uncertainty quantification (UQ) should be coupled with regression models to account for the inherent risk associated with the aerodynamic design process. Aerodynamic systems are inherently complex and uncertainties arise from various sources such as modeling assumptions, numerical approximations, and experimental measurements. Therefore, neglecting these uncertainties can lead to significant deviations between the predicted and actual performance of an aircraft, resulting in suboptimal designs and unexpected failures in the final system. In the context of regression models, since there is always a chance that the predicted values will not match the actual values, UQ is particularly crucial to provide information about the reliability or trustworthiness of their predictions. By coupling the regression models with UQ techniques such as Bayesian inference or Monte Carlo simulation, engineers can obtain not only the predictive values but also the variance or confidence intervals of those predicted outputs. This information can help engineers evaluate the reliability of the prediction and make informed decisions about design parameters. Although there are two popular regression models for UQ, namely GPR and Bayesian neural networks (BNNs), they cannot be considered efficient for the purpose of UQ. GPR is notorious for its time complexity of $O(n^3)$ and memory complexity of $O(n^2)$, where n denotes the dataset size [46, 47]. Even in multi-output regression tasks, as mentioned above, the GPR should be trained for each output independently, which requires linearly increasing training time with respect to the output dimension. BNNs can perform the multi-output pre-

diction only with a single regression model, but they require cumbersome and complicated training algorithms due to significant modifications to the conventional framework of NNs [48, 49, 50], and their additional model parameters lead to slower convergence [51]. Such computational complexity and inefficiency prevent GPR and BNNs from being viable options for engineers who prioritize practicality, highlighting the need for a reliable and efficient regression model for UQ.

In summary, overcoming these bottlenecks is a prerequisite for applying regression-based data-driven approaches to real-world engineering problems, which is directly related to the contributions of this dissertation in the next section.

1.2 Contributions of the dissertation

This section summarizes the contributions of the dissertation in terms of alleviating the three previously mentioned bottlenecks of regression models in aerodynamic design: 1) high-dimensional input space, 2) high-dimensional output space, and 3) reliable and efficient UQ.

1. The high-dimensional input space in aerodynamic design is reduced to the low-dimensional latent space using dimensionality reduction (DR) techniques. Also known as representation learning or manifold learning, these techniques find the low-dimensional latent representation of high-dimensional original data. They can significantly reduce the dimensionality of the input space to a level suitable for effectively training the regression models. By alleviating the curse of dimensionality with DR techniques, the feasibility of applying gradient-free optimizers to the reduced input space is also investigated. For this purpose, inverse design optimization of the wind turbine airfoil is adopted as a case study. This case study finally proves that the original high-dimensional input space can be successfully reduced using a deep learning-based DR technique, and also the genetic algorithm (GA) is applied to its framework to validate its feasibility to be coupled with gradient-free optimizers.

2. Prediction of high-dimensional data is performed using reduced-order modeling (ROM) techniques. ROM specializes in predicting high-dimensional QoIs by treating a high-fidelity CFD simulation as a black-box function and learning simplified models in a data-driven manner. As in the case of the high-dimensional input space, the DR process is again required in its procedure to construct the latent space. And this study focuses on the fact that since this latent space acts as an intermediary in ROM, it

inevitably affects the performance of ROM. In this regard, the prediction of flow fields around a transonic airfoil is adopted as a case study, and the physical interpretability of the latent space constructed by various machine learning-based DR techniques is investigated. Furthermore, the impact of its interpretability on ROM performance is analyzed, and finally, its significance is validated in terms of accuracy and efficiency for predicting high-dimensional output space via ROM.

3. Regression model capable of reliable and efficient UQ is investigated. To this end, the deep ensembles (DE) approach is analyzed in the task of predicting missile aerodynamic performance. DE is based on neural networks so all of the following are viable: universal approximation capability, scalability to large datasets, and multi-output regression. Last but not least, it is able to quantify the predictive uncertainty by a simple modification of the MLP structure. This dissertation aims to validate this simple and scalable DE method for multi-output regression tasks, which are the most common problems in practical engineering disciplines. The most popular regression model capable of UQ in engineering fields but not scalable to large datasets, GPR, is also compared with DE. Not only the validation is performed, but since the poor reliability of quantified uncertainty by DE is observed, a simple post-hoc calibration method is applied to DE models to correct the unsatisfactory uncertainty quality. The results show that the DE technique with calibration can perform more reliable and efficient UQ compared to GPR. Finally, the impact of the calibration method on Bayesian optimization is examined, verifying the fact that whether the DE is calibrated or not can result in completely different exploration characteristics during Bayesian optimization.

1.3 Overview of the dissertation

The remaining chapters of this dissertation are structured as follows:

- **Chapter 2** explores effective training approaches for regression models in the context of high-dimensional input spaces. The case study focuses on the inverse design optimization of wind turbine blade airfoils.
- **Chapter 3** then analyzes how the high-dimensional output space can be efficiently predicted by ROM. In this case, the flow field prediction of the transonic airfoil is conducted.
- **Chapter 4** examines the feasibility of the DE regression model to quantify reliable predictive uncertainty in the multi-output regression task. To this end, the case study of predicting multiple aerodynamic coefficients of the missile configuration is adopted.
- **Chapter 5** presents the concluding remarks and outlines future directions of this dissertation.

Chapter 2

High-dimensional input space

The work in this chapter was published in the Engineering with Computers [18].

2.1 Introduction

This chapter aims to address the high-dimensional input space that hinders the efficient application of regression models to aerodynamic design by reducing it to the low-dimensional latent space using DR techniques. By alleviating the curse of dimensionality with DR techniques, the feasibility of applying gradient-free optimizers to the reduced input space is also investigated. For this purpose, an inverse design optimization of the wind turbine airfoil is adopted as a case study, since the inverse design often requires high-dimensional input consisting of physical characteristics. The rest of this section highlights why the selected case study—inverse design optimization for the airfoil—is crucial in the aerodynamic design discipline, and how this dissertation successfully mitigates the

problems arising from the high-dimensional design space.

Recent advances in high-performance computing have enabled aerodynamic engineers to use high-fidelity analyses, offering a wide range of options in the aerodynamic shape design process. Accordingly, numerous novel design methodologies have been developed, most of which are based on two conventional aerodynamic design methods: inverse and direct design approaches [52, 53, 54]. In particular, inverse design is computationally efficient in that the desired target performance distribution is explicitly defined and the corresponding design shape can be calculated with a few iterations coupled with a flow solver [55, 56, 57].

However, the inverse method also has a critical disadvantage: whenever the target distribution changes, an iterative process to find the design shape matching the target distribution should be repeated. Considering that most design stages require significant trial and error, this process undermines the efficiency of the inverse design. Therefore, several researchers have used a surrogate model in inverse design to avoid this iterative process. In particular, multi-layer perceptron (MLP) surrogate models have been widely used owing to their universal approximation property [58]. Kharal and Saleem [59] and Sun et al. [29] used aerodynamic QoIs as the inputs of a MLP model to obtain airfoil shape parameters as the output during the inverse design procedure. Wang et al. [60] also applied MLP for the same purpose, but additionally performed dimensional reduction of input data to reduce the database size required for model training.

Though these studies do not require iterative procedures coupled with the flow solver, they still require the predefined performance distribution. For an efficient inverse design, an appropriate aerodynamic performance should be defined, which is highly dependent on the designer's engineering knowledge and experience. This ambiguity in specifying the target distribution has inspired

researchers to optimize it. Obayashi and Takanashi [61] and Kim and Rho [62] used control points-based techniques to parameterize the pressure distributions, and then optimized the distribution using GA. In these inverse design optimization processes, the aerodynamic QoIs of the distribution were obtained through theoretical/empirical predictions, and a number of constraints were imposed to ensure the reality of the distribution. For instance, Obayashi and Takanashi [61] estimated the viscous drag using the Squire–Young empirical formula and imposed six constraints for realistic pressure coefficient (C_p) distributions. Though these studies attempted to deal with the fundamental limitation of the inverse design method, the following problems still exist: 1) loss of the diverse representation capacity of the C_p distribution due to parameterization; 2) excessive constraints to ensure a realistic C_p distribution; 3) discrepancies between the QoI predicted theoretically/empirically and those calculated using a flow solver; and 4) impossibility of explicitly imposing geometric constraints on the design shape.

To address the limitations of the representation capacity, QoI discrepancies, and excessive constraints, Zhu et al. [63] reduced the dimension of the C_p distribution data via proper orthogonal decomposition (POD) and used the SVR model to predict the aerodynamic performance of the airfoil from the reduced C_p data. Then, a GA was implemented to optimize the C_p distribution coupled with POD and SVR. Finally, the airfoil shape corresponding to the optimized pressure distribution was obtained in an iterative manner coupled with the flow solver. However, the limitation of geometric constraints has not been still addressed since the prediction of the design shape is separated from the pressure optimization process. Therefore, when the shape predicted by the inverse design violates geometric constraints, the design process should be traced back to the optimization process. Additionally, at the optimum solution, the discrepancy

between the pressure distributions from prediction and calculation is noticeable, which indicates the low accuracy of its framework.

The drawbacks of the previous inverse design studies presented so far should be overcome by applying novel techniques in that computational efficiency, an essential advantage of inverse design, cannot be fully exploited. Therefore, this chapter proposes an inverse design optimization framework with a two-step deep learning approach. This approach refers to the sequential coupling of two deep learning models: variational autoencoder (VAE) [64] and MLP [25]. The VAE and MLP were used to generate a realistic target distribution and to predict the QoI and shape parameters from the generated distribution, respectively. Then, target distribution optimization was performed as the inverse design optimization based on this approach. Active learning and transfer learning strategies were applied to improve the accuracy of the two-step approach-based optimization with reasonable computational cost. The proposed inverse design optimization framework via a two-step deep learning approach was validated through aerodynamic shape optimization problems of the airfoil in wind turbine blades, where inverse design is actively being applied.

This chapter is organized as follows. Sec. 2.2 describes the mathematical background of two deep learning models used in the inverse design optimization framework. Sec. 2.3 presents the scheme of the proposed framework. In Sec. 2.4, validation of the framework with application to a wind turbine airfoil is performed and the results are discussed. Finally, Sec. 2.5 concludes the study, emphasizing the flexibility of the presented framework.

2.2 Methodologies

2.2.1 Multi-layer perceptron (MLP)

Engineers from various disciplines have been drawn to MLP due to their universal approximation capability [58, 65], scalability to large datasets through mini-batch training [66], and capability of predicting multi-output using a single regression model. This section provides a brief theoretical overview of the MLP, which is often referred to as neural networks (NNs).

The feed-forward mechanism propagates the data obtained from the input layer of MLP to the output layer. In this procedure, information moves via an affine transformation from the input layer to the output layer as follows:

$$y = Wx + b, \tag{2.1}$$

where x is a vector of nodes in the input layer and y is that in the output layer. W and b are the weight matrix and bias vector between the input and output layers, respectively. Regardless of the number of hidden layers between the input and output layers, nonlinearity between x and y cannot be captured since they are linearly correlated in Eq. 2.1. In this context, the concept of an activation function that modifies the output of MLP is introduced. By incorporating nonlinear activation functions at each layer, MLP becomes able to perform nonlinear modeling. A variety of nonlinear activation functions are available, including the LeakyReLU function [67], which is as follows:

$$f(x) = \begin{cases} x, & \text{if } x \geq 0 \\ ax, & \text{otherwise} \end{cases} \tag{2.2}$$

where a stands for a non-zero small gradient (0.01 for this study). In order to simulate nonlinear behavior, the MLP model then applies an activation function

(Eq. 2.2) to the output of the previous layer. The correspondingly transformed output is then utilized as the input for the subsequent layer. This process, known as feed-forward, is repeated through the hidden layers.

However, the feed-forward itself cannot achieve the expected accuracy because it lacks an algorithm for adaptively training the parameters of MLP, namely weights (W) and biases (b). To address this issue, the backpropagation training algorithm was introduced, which minimizes the loss function by adjusting the parameters to make the predicted values of the MLP similar to the desired target values as the training progresses [68]. To achieve this, gradient descent optimization techniques, such as Adagrad [69], RMSprop [70], and Adam [71], are utilized to minimize the loss function. In particular, Adam has become increasingly popular due to its strengths in dealing with sparse gradients and non-stationary objectives, combining Adagrad and RMSprop [71]. As the feed-forward process and backpropagation with gradient descent are repeated iteratively, the loss function will decrease to the desired level so that the training can be stopped. The converged weights and biases of the MLP model can then be used to perform almost real-time predictions using the feed-forward operation. Only the essential aspects of MLP are presented here, as many studies have already described them. More information on MLP can be found in Goodfellow et al. [25].

2.2.2 Autoencoder (AE)

The AE model is a widely used deep learning-based DR technique. The main objective of the AE is to output exactly what is inputted. Its structure consists of two parts: an encoder and a decoder. The input of the AE, $\mathbf{x}$, is entered into the encoder for the compression and exits as latent variables (LVs), $\mathbf{z}$. Then, $\mathbf{z}$ enters the decoder and exits as reconstructed data $\tilde{\mathbf{x}}$. The encoder and decoder

consist of MLP, which makes it possible to model nonlinearity in the reduction and reconstruction processes. Because the objective of training the AE model is to reconstruct $\tilde{\mathbf{x}}$ similar to the original data $\mathbf{x}$, the loss function is defined by Eq. 2.3, where N denotes the number of data samples. Although its loss function is defined by the mean square error (MSE) in this study, any other error metrics, such as the binary cross entropy between $\mathbf{x}$ and $\tilde{\mathbf{x}}$, can be used depending on the properties of the data. Note that the adopted MSE is the summation in both sample-wise and element-wise directions.

$$\mathcal{L}_{AE} = \frac{1}{N}\sum_{i=1}^{N}(\mathbf{x}_i - \tilde{\mathbf{x}}_i)^2, \tag{2.3}$$

However, this AE model has an obvious limitation: there is no training algorithm for guaranteeing a regularized latent space. Herein, the expression "regularized latent space" means that the latent space is trained to be smooth and continuous; thus, when inputs $\mathbf{x}_1$ and $\mathbf{x}_2$ are similar (or close), their corresponding mapped latent values, $\mathbf{z}_1$ and $\mathbf{z}_2$, should also be similar [72]. In AE model, as the original input $\mathbf{x}$ becomes condensed layer-by-layer through the encoder, its representation becomes increasingly abstract [73]. Therefore, the output of the encoder $\mathbf{z}$ is not guaranteed to be regularized in the training procedure of the AE. This explains why the decoder part of the AE model cannot be used as a generative model: the trained latent space is irregular and therefore its correlation with the reconstructed data is abstract.

2.2.3 Variational autoencoder (VAE)

A number of studies have focused on the limitations of this unregularized latent space trained by an AE and have alternatively applied VAE in their framework. The structure of the VAE is similar to that of the AE. One major difference is that VAE stochastically extracts the latent space $\mathbf{z}$ via random

sampling, whereas the AE model obtains its latent space deterministically.

The mathematical formulae for the VAE model are presented below (for further details, please refer to these references [64, 18]). Let's consider reducing the original dataset $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^{N}$ (assumed to be independently and identically distributed) to the latent space $\mathbf{z}$ using the VAE model. In the inference of $\mathbf{z}$ from $\mathbf{X}$, variational inference is adopted due to the intractability of the posterior $p_\theta(\mathbf{z}|\mathbf{x})$, where θ is a parameter of the VAE model (to be more specific, it is intractable owing to the likelihood of $p_\theta(\mathbf{z})$). Therefore, instead of the intractable posterior $p_\theta(\mathbf{z}|\mathbf{x})$, the VAE is trained to obtain its substitute, $q_\phi(\mathbf{z}|\mathbf{x})$ (ϕ is a variational parameter). Then, the log-likelihood of $p_\theta(\mathbf{x})$ can be expressed as:

$$\log(p_\theta(\mathbf{x})) = \int \log\left(\frac{p_\theta(\mathbf{x},\mathbf{z})}{q_\phi(\mathbf{z}|\mathbf{x})}\right) q_\phi(\mathbf{z}|\mathbf{x}) d\mathbf{z} + \int \log\left(\frac{q_\phi(\mathbf{z}|\mathbf{x})}{p_\theta(\mathbf{z}|\mathbf{x})}\right) q_\phi(\mathbf{z}|\mathbf{x}) d\mathbf{z}, \tag{2.4}$$

where the second term on the right-hand side (RHS) is the KL-divergence of $q_\phi(\mathbf{z}|\mathbf{x})$ from $p_\theta(\mathbf{z}|\mathbf{x})$, $KL(q_\phi(\mathbf{z}|\mathbf{x})||p_\theta(\mathbf{z}|\mathbf{x}))$, which is always non-negative according to its definition (KL-divergence is a measurement of the statistical distance between two probability distributions). Therefore, the first term on the RHS becomes the lower bound of the log-likelihood and the problem of maximizing the log-likelihood becomes the problem of maximizing the lower bound. This lower bound can be expressed as:

$$\int \log\left(\frac{p_\theta(\mathbf{x},\mathbf{z})}{q_\phi(\mathbf{z}|\mathbf{x})}\right) q_\phi(\mathbf{z}|\mathbf{x}) d\mathbf{z} = \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{z})] - \int \log\left(\frac{q_\phi(\mathbf{z}|\mathbf{x})}{p_\theta(\mathbf{z})}\right) q_\phi(\mathbf{z}|\mathbf{x}) dz, \tag{2.5}$$

where the first and second terms on the RHS are the reconstruction error and KL-divergence of $q_\phi(\mathbf{z}|\mathbf{x})$ from $p_\theta(\mathbf{z})$, respectively. However, owing to the existence of $q_\phi(\mathbf{z}|\mathbf{x})$ in the reconstruction error, the back-propagation process cannot be performed, and calculating the gradient of the reconstruction error with respect to ϕ is problematic due to the posterior $q_\phi(\mathbf{z}|\mathbf{x})$. Therefore, a "reparameterization trick" is adopted, which allows for back-propagation during the

sampling process. The concept behind this trick is to sample $\mathbf{z}$ from the auxiliary noise variable $\epsilon \sim N(0, 1^2)$: this random sampling makes the latent space in the VAE stochastically determined. To be more specific, the k$^{\text{th}}$ LV (z_k) is assumed to follow the distribution below:

$$z_k = \mu_k + \sigma_k \odot \epsilon. \tag{2.6}$$

where $\odot$ denotes Hadamard product (element-wise product). Accordingly, the KL-divergence, the second term on the RHS in Eq. 2.5 can be rewritten as below when the posterior $q_\phi(\mathbf{z}|\mathbf{x})$ and prior $p_\theta(\mathbf{z})$ are assumed to follow the Gaussian distribution $N(\mu, \sigma^2)$ and $N(0, \mathbf{I}^2)$, respectively.

$$KL(q_\phi(\mathbf{z}|\mathbf{x})||p_\theta(\mathbf{z})) = \frac{1}{2}\sum_{k=1}^{d}(\sigma_k^2 + \mu_k^2 - (\log(\sigma_k^2) + 1)), \tag{2.7}$$

where μ_k and σ_k represent the mean and standard deviation used during the reparameterization of z_k, and d denotes the dimension of the latent space. Herein, as the posterior approximation $q_\phi(\mathbf{z}|\mathbf{x})$ becomes similar to the prior $p_\theta(\mathbf{z})$, the KL-divergence decreases. Finally, the loss function of VAE model can be formulated as in Eq. 2.8, which consists of the reconstruction error (MSE term) and regularizer (KL-divergence term). Since the KL-divergence term serves to regularize the latent space to be trained, it is also called regularization loss. It induces a sparser latent space [64, 74, 75, 76] just as the L1 regularization term makes the model sparse in the Lasso regression [77].

$$\begin{aligned}\mathcal{L}_{VAE} &= \mathcal{L}_{AE} + KL(q_\phi(\mathbf{z}|\mathbf{x})||p_\theta(\mathbf{z})) \\ &= \underbrace{\frac{1}{N}\sum_{i=1}^{N}(\mathbf{x}_i - \tilde{\mathbf{x}}_i)^2}_{\substack{\text{reconstruction} \\ \text{error}}} + \underbrace{\frac{1}{2}\sum_{k=1}^{d}(\sigma_k^2 + \mu_k^2 - (\log(\sigma_k^2) + 1))}_{\substack{\text{regularization} \\ \text{error}}}.\end{aligned} \tag{2.8}$$

2.3 Inverse design optimization framework

2.3.1 Two-step deep learning approach

This section demonstrates the two-step deep learning approach, which is the combination of the VAE and MLP models. First, VAE is trained with the target performance distributions of the training data. When the training is completed, only the decoder part of the VAE model is used; it operates as a data generator that receives a low-dimensional latent variable and outputs a realistic high-dimensional target distribution. Then, the MLP is trained to predict QoIs and shape parameters from the target distribution. This regression process is more efficient and accurate than previous inverse design studies by eliminating the need for iterations and theoretical/empirical assumptions (prediction can be performed almost in real-time). In this study, these two deep learning models, the decoder of the VAE and MLP, are utilized sequentially; the decoder generates the distribution once it receives the latent variable, and the MLP outputs QoIs and shape parameters from this generated distribution. This structure refers to a two-step deep learning approach and its flowchart is shown in Fig. 2.1.

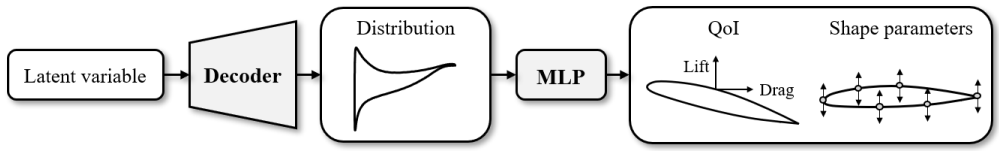


Figure 2.1: Flowchart of the two-step deep learning approach.

2.3.2 Target distribution optimization

The two-step deep learning approach allows mapping from the latent space of the target performance distribution to QoIs and shape parameters. Therefore,

the optimization of the target distribution can be performed based on this approach, where the inputs of the corresponding approach (LVs) are used as the optimization variable, and the outputs (QoIs and shape parameters) as the objective functions and constraints. In this study, since the shape parameters are incorporated into the target distribution optimization procedure, geometric constraints can be imposed explicitly. At the end of the optimization, numerical validation is performed regarding the optimum solutions by comparing the QoIs predicted using the two-step approach and QoIs calculated using the numerical flow solver. The inverse design optimization framework terminates when the differences in these values satisfy the error criterion. If not, the process described in Sec. 2.3.3 is repeated until it is satisfied.

2.3.3 Active learning and transfer learning

Since VAE and MLP models are trained with initial training data, the optimization based on them is unlikely to meet the desired accuracy at once. Therefore, surrogate-based optimization studies usually add training data repeatedly to increase the accuracy of the surrogate model. This technique is called the pool-based active learning strategy (or adaptive sampling) and is adopted in this study for an accurate inverse design optimization framework [78, 79]. When the error criterion at optimum solutions is not satisfied (as presented at the end of Sec 2.3.2), these solutions were added to the previous dataset and the deep learning models were trained again. In the training procedure based on data splitting, which splits the full dataset into training and test data, the designs newly added at every iteration in the active learning process should be incorporated into the training data to maximize the efficiency of its process. This is because to reflect the newly added designs directly in the model training, they should be included in the training dataset, not the test dataset (test data is not

used directly in model training). Then, optimization and numerical validation were performed based on these retrained models. This active learning strategy continues until the error between the QoI from the two-step deep learning approach and those from the numerical solver decreases so that they satisfy a predefined error criterion.

Although the active learning process was applied to effectively improve the accuracy of the framework, restarting model training with randomly initialized weights and biases at every iteration can severely degrade the computational efficiency. Moreover, in general, when adding new data via active learning, the model does not change significantly compared with that of the previous iteration, as only a small amount of data is added to the existing data. Therefore, this study used a parameter-based transfer learning strategy [80]. This technique ensures that the weights and biases of the previously trained models are transferred to the models to be newly trained [81, 82, 83]. Combining these two strategies, iterative active learning for model accuracy can be efficiently performed through transfer learning. The flowchart of the inverse design optimization framework, which summarizes the contents of Sec. 2.3, is shown in Fig. 2.2.

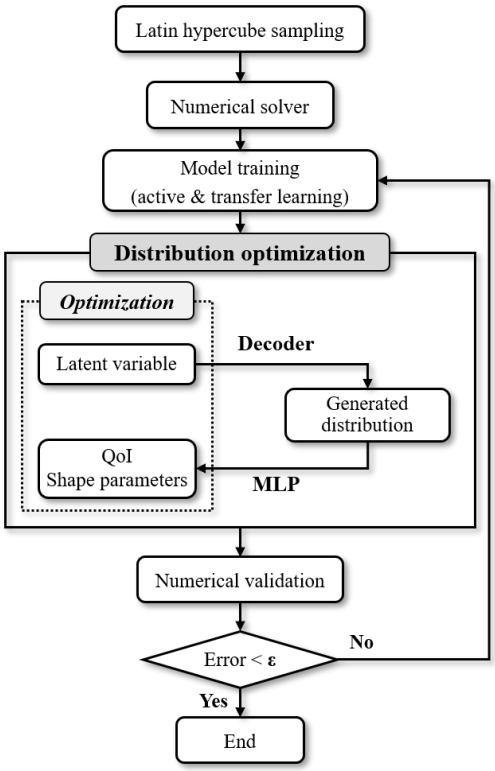


Figure 2.2: Flowchart of the inverse design optimization framework.

2.4 Framework validation: optimization of the airfoil for wind turbine blades

The proposed inverse design optimization framework can be applied to all inverse design problems in any engineering field. Specifically, in aerospace engineering, inverse design is actively being applied to wind turbine design [84, 85, 86]; therefore, to verify the accuracy, effectiveness, and robustness of this framework, airfoil optimization of a megawatt-class wind turbine is chosen as the application. For its airfoil design, structural and aerodynamic performances are mainly considered. In particular, the airfoil at the blade tip is known to be critical for the aerodynamic performance of the blade. This study aims to optimize the airfoil at the tip of the blade, mostly taking into account its aerodynamic properties (structural performance is indirectly considered by the airfoil area). Single-objective and multi-objective optimizations are performed to demonstrate the versatility of the proposed framework in different optimization problems. The following sections describe the optimization problems (Sec. 2.4.1), architectures of the two-step deep learning models used in the inverse design optimization (Sec. 2.4.2), and the results and discussion of the single-objective and multi-objective optimizations (Sec. 2.4.3 and 2.4.4).

2.4.1 Optimization of the airfoil in wind turbine blades

This section presents the optimization problems of the airfoil of a wind turbine blade tip region. Sec. 2.4.1.1 presents the validation of the numerical flow solver used in this study, and Sec. 2.4.1.2 presents the problem definitions for the optimization.

2.4.1.1 Flow solver

Xfoil is a two-dimensional panel code capable of viscous/inviscid analysis, and it derives accurate results in a very short time when used appropriately [87]. Because a wind turbine operates at relatively low Reynolds numbers, numerous wind turbine airfoil design studies have used this solver [88, 89, 90]. In this study, Xfoil is adopted to calculate the aerodynamic QoIs with reasonable computational cost. It should be noted that the proposed inverse design optimization framework can be coupled with any numerical solver with arbitrary QoIs.

Although Xfoil is a well-known and widely used solver, solver validation using experimental results is performed [1]. The experimental data are based on the GA(W)-1 airfoil with Reynolds of 6.310^6, Mach of 0.15, and angle of attack of 8.02° (the flow conditions of the validation are intended to be similar to those of subsequent airfoil optimizations). Accordingly, it is confirmed that this flow solver is appropriate for the wind turbine airfoil optimization performed in this study since it predicts a pressure distribution almost identical to that from experiments (Fig. 2.3).

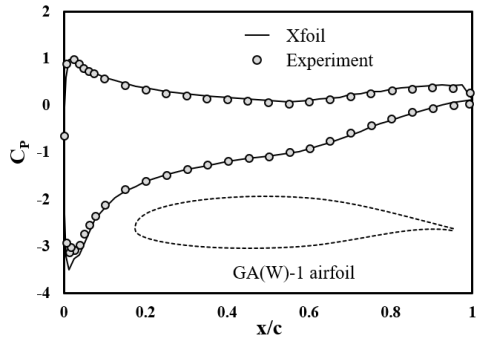


Figure 2.3: Comparison of pressure distributions from Xfoil and experimental results in Ref. [1] (adopted airfoil configuration is also visualized).

2.4.1.2 Optimization problem definitions

Before performing the optimization, the optimization problems are defined first. There are numerous parameterization methods for representing the airfoil shape, such as PARSEC, B-spline, and class-shape transformation (CST) [91, 92, 93]. In this study, PARSEC parameters are adopted as they were originally introduced due to their close relationship with the aerodynamic characteristics [19]. There are 11 PARSEC variables and among them, six are used as shown in Fig. 2.4: namely $R_{L.E.}$ (leading-edge radius), X_{up} (x-coordinate of the upper crest), Z_{up} (z-coordinate of the upper crest), X_{low} (x-coordinate of the lower crest), Z_{low} (z-coordinate of the lower crest), and $Z_{T.E.}$ (z-coordinate of the trailing-edge). These variables are selected from the prior sensitivity test, which demonstrated that they have a significant impact on the flow characteristics, whereas the other five PARSEC variables have little impact. The corresponding multi-dimensional design space to be explored in the optimization process is summarized in Table. 2.1 (the baseline airfoil shape is selected as the median value of each variable's range).

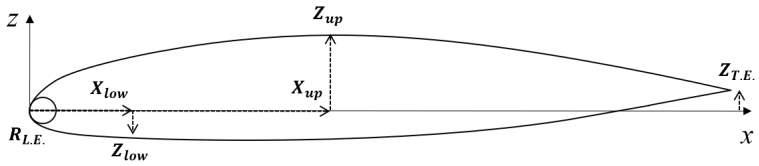


Figure 2.4: Shape parameters for airfoil representation: six PARSEC parameters are used.

When the airfoil shape is determined using PARSEC parameters, flow analysis proceeds. Xfoil is executed under predefined settings including the flight conditions, and the aerodynamic QoIs that will be used as objective functions or constraints in the optimization are calculated. The corresponding flight

Table 2.1: Design space of the six airfoil shape parameters: the baseline airfoil is selected as the median value of each range

Design variables	Lower bound	Baseline	Upper bound
$R_{L.E.}$	0.015	0.0275	0.04
X_{up}	0.3	0.375	0.45
Z_{up}	0.09	0.12	0.15
X_{low}	0.3	0.375	0.45
Z_{low}	0.09	0.12	0.15
$Z_{T.E.}$	0.09	0.12	0.15

conditions, objective functions, and constraints for single-objective and multi-objective airfoil optimizations are summarized in Table 2.2. In single-objective optimization, the objective is to maximize the lift-to-drag ratio (L/D), which is the most crucial factor for aerodynamic efficiency. Furthermore, some constraints are considered to exclude undesirable performance [94]: the drag should be less than the baseline value, the pitching moment coefficient at $c/4$ (c is the chord length) should be greater than the specific value to limit blade torsion, and the airfoil area should be at least 90% of the baseline area to prevent serious degradation of the structural performance. Note that the geometric constraint, airfoil area in this study, can be directly imposed in this framework as QoI in the target distribution optimization process (PARSEC parameters can also be set as geometric constraints, such as $Z_{up} < 0.15$, but these were realized by limiting the design space herein). In multi-objective optimization, the two objectives are to maximize the L/D ratio and airfoil area, considering the trade-off between aerodynamic and structural performances. Other constraints are the same as those in the single-objective optimization.

Table 2.2: Flight conditions, objective functions, and constraints for single-objective and multi-objective optimizations

Flight conditions	Reynolds number	6×10^6
	Mach number	0.25
	AoA	7°
Single-objective optimization	Objective function	Maximize L/D
	Constraints	$C_d <$ Baseline C_d
		$C_m > -0.08$
		Area $> 0.9 *$ Baseline Area
Multi-objective optimization	Objective functions	Maximize L/D
		Maximize Area
	Constraints	$C_d <$ Baseline C_d
		$C_m > -0.08$

Then, DoE is performed to train the MLP and VAE models; the sampled designs are used as the initial training data. Latin hypercube sampling is selected considering its uniformity in the design space [95]. A total of 500 initial designs are sampled and two deep learning models are trained based on them. Finally, the optimization proceeds regarding the trained LVs of the VAE as the optimization variables, and the QoIs and shape parameters as the objective functions and constraints, as shown in Fig. 2.2. For single-objective optimization, GA is adopted owing to its efficient global exploration in discontinuous and multimodal problems [96]. For multi-objective optimization, the non-dominated sorting genetic algorithm-II (NSGA-II) is adopted to obtain the diversified Pareto solutions of both objective functions [97]. Both optimization algorithms are implemented using Python package pymoo [98].

When the first optimization ends with two-step deep learning models trained using 500 DoE designs, active learning with a transfer learning strategy is conducted iteratively. Because there is only one optimal solution obtained in the single objective optimization, the single optimal solution is selected as the design to be infilled. On the other hand, there are several optimal solutions for multi-objective optimization (Pareto solutions). In this case, the leftmost, middle, and rightmost designs in the Pareto solutions are selected to be infilled in order to increase the overall accuracy of the Pareto solutions. These criteria for infilling can be determined arbitrarily by the engineer (the number of points to be added for each iteration and their distribution in the Pareto frontier can be determined as appropriate). Through these iterative procedures, the framework consisting of the two deep learning models will be able to satisfy the given error criterion. Again, note that the proposed framework can be applied to any inverse design problem (any design configuration, corresponding shape parameters, QoIs, numerical solver, and optimizer can be selected arbitrarily).

2.4.2 Architectures of the two-step deep learning models

In the two-step approach, the MLP model is trained to take the C_p distribution as input and output QoIs and shape parameters. First, all airfoils are discretized to share 199 identical x-coordinates: they are extracted using a two-sided hyperbolic tangent distribution function [99] from NACA 0012 airfoil (where the spacing at the leading-edge and trailing-edge is constrained as 0.001c and 0.005c, respectively). Then, the pressure coefficients of the corresponding points are used as the input of the MLP. And six shape parameters ($R_{L.E.}$, X_{up}, Z_{up}, X_{low}, Z_{low}, and $Z_{T.E.}$) and four QoIs (L/D, C_d, C_m, and area) are concatenated to form the 10 output nodes. Finally, the MLP has 199 input nodes, 10 output nodes (all the inputs and outputs are normalized), and two

hidden layers with 100 nodes: the MLP with the corresponding hidden layers was found to have sufficient accuracy for the regression in this problem. Then, LeakyReLU activation functions are applied to all the layers for nonlinearity. Adam is used as the optimizer with the MSE loss function to train this MLP architecture, and the initial learning rate starts at 0.001. For the first iteration in active learning, 500 initial samples are split into training and test data in the ratio of 8:2 (the same ratio was also used to train the VAE). A total of 30000 epochs with a mini-batch size of 100 are performed using a scheduler that multiplies the learning rate by 0.8 for every 3000 epochs. Because there are no weight or bias values to be used as a reference in the first iteration, they are initialized using He initialization [100]. Then, active and transfer learning are performed. Interestingly, during the MLP training, it is observed that if the parameters of all layers from the previous model are passed, the training terminates without any meaningful change from the previous model. This is because the number of newly added designs is small compared with that of existing training data, and their effect on the loss function becomes negligible. Therefore, active learning, which is intended to increase accuracy by appending previous optimum solutions to the current training data, becomes meaningless. In this regard, only the parameters of other layers are transferred from the previous model, and those of the last layer of the MLP are initialized using He initialization (in other words, transfer learning is applied except for the last layer). For subsequent iterations, the total epochs are set to 10000 with the same initial learning rate and scheduler as the first iteration. As a result, training the MLP in the first iteration using a personal computer (Intel Core i7-8700 3.2 GHz with 16 GB 2400 MHz DDR4 RAM) took 508 s, and subsequent iterations took an average of 186 s per iteration using Python package PyTorch [101]. The fact that subsequent iterations during active learning require a much shorter training time

than the first iteration emphasizes the effect of applying the transfer learning technique in this study: with transfer learning, active learning can be efficiently performed.

For the VAE model, the 199 C_p data previously inputted into the MLP are used as inputs and outputs (VAE has the same input and output). The 199-dimensional input data is reduced to four-dimensions using an encoder with hidden layers of 120, 60, and 30 nodes. Herein, the four-dimensions represent distribution parameters for random sampling in the latent space: two for the mean and the other two for the standard deviation of the Gaussian distribution. From these parameters, random sampling is performed, and the dimension is finally reduced to a two-dimensional latent space. These two dimensions are again reconstructed to 199 dimensions using a decoder with hidden layers of 30, 60, and 120 nodes. The corresponding architecture of the VAE is shown visually in Fig. 2.5. As in the MLP, the LeakyReLU activation function and Adam optimizer with MSE loss function are used. The initial learning rate starts at 0.001, and a total of 30000 epochs are performed with a mini-batch size of 100 and a scheduler multiplying the learning rate by 0.5 for every 5000 epochs. In contrast to the MLP, the situation in which the learning process terminates without reflecting information on a newly added design point is hardly observed in the VAE. Therefore, the parameters of all the layers from the previous iteration are transferred to the next iteration (in other words, transfer learning is applied to all layers). For subsequent iterations, a total of 10000 epochs are performed with the same initial learning rate and scheduler as the first iteration. As a result, training the VAE in the first iteration took 627 s, and subsequent iterations took an average of 226 s per iteration (again, the efficiency of transfer learning can be verified). Note all the hyperparameters of the MLP and VAE mentioned in this section are applied identically in the

single-objective and multi-objective optimizations.

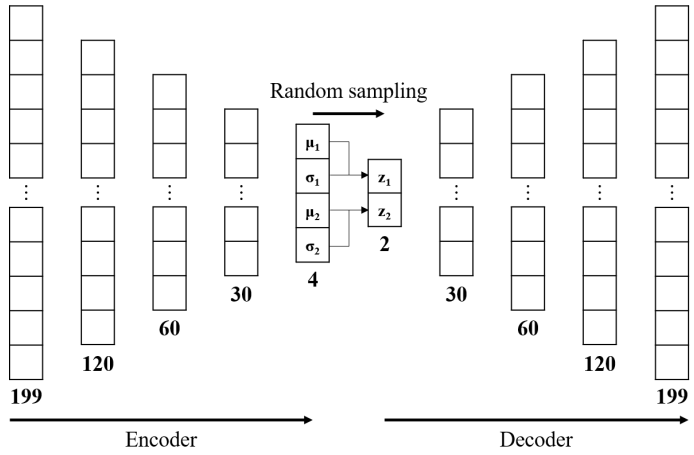


Figure 2.5: Architecture of the VAE.

2.4.3 Single-objective optimization results and discussion

Active learning of the single-objective optimization satisfies the error criterion (the error of the objective function at optimum solutions should be lower than 1%) after the 24 infilling iterations. The convergence history of optimization with active learning can be observed in Fig. 2.6 and the loss function histories of MLP and VAE are shown in Fig. 2.7. The objective function starts at approximately 65 and increases gradually, reaching a value of approximately 72 after 24 iterations. After that, no better optimal point was found. Subsequent analysis of the single-objective optimization results is based on the trained VAE and MLP at iteration 24. The total learning time for 24 iterations is (508+627) + (186+226) * 24 = 11,023 s.

The final optimal airfoil shape is shown in Fig. 2.8, and its QoIs (objective function and constraints) are summarized in Table 2.3. Its objective function (L/D) has a value of 72.22, which increased by 39% compared with the baseline.

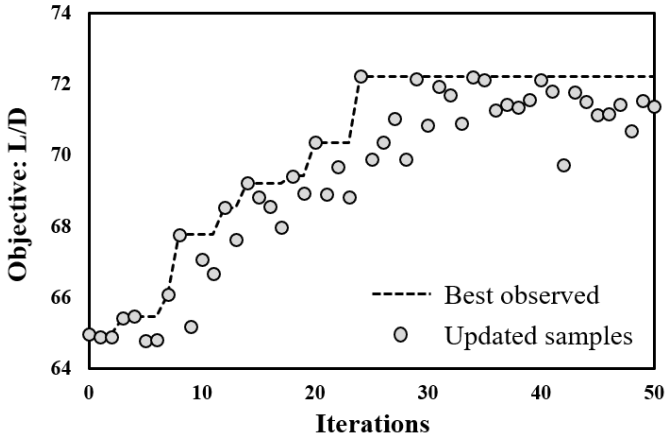
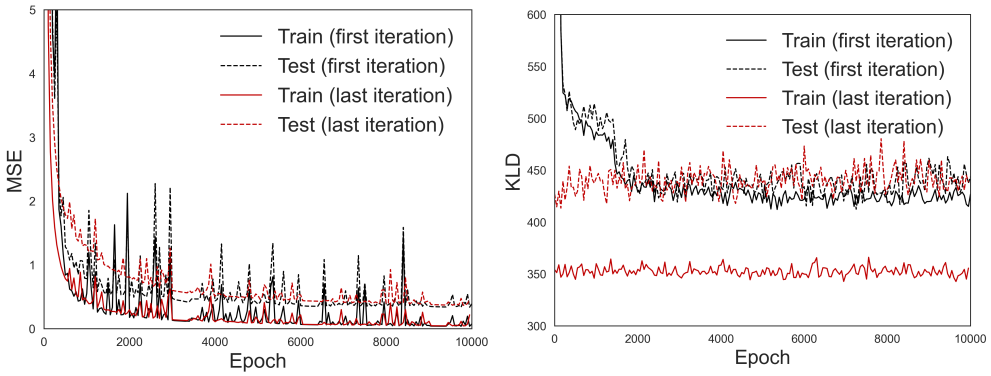


Figure 2.6: Convergence history of single-objective optimization with active learning.



(a) (b)

Figure 2.7: Loss history of (a) MLP, and (b) VAE. For both models, the history of the first iteration and last (24th) iteration of active learning is represented.

However, this value is just a prediction from the MLP model, and it is not certain whether the L/D calculated by Xfoil will have this value. Therefore, the

QoIs from Xfoil are compared with the predicted values from the MLP model. It is confirmed that the four QoI values have an error (between predicted and calculated) of less than 1%, and all the imposed constraints are satisfied. Note that the airfoil area satisfies the constraint imposed with little margin (0.6%), whereas other constraints (drag and pitching moment) are satisfied with some margin. Additionally, numerical validation is performed to verify whether the optimal airfoil actually shows the C_p distribution generated by the VAE model. Fig. 2.9 demonstrates that the C_p generated by the VAE and that calculated using Xfoil are almost indistinguishable. These results validate the accuracy of the MLP model in the two-step approach for single-objective optimization.

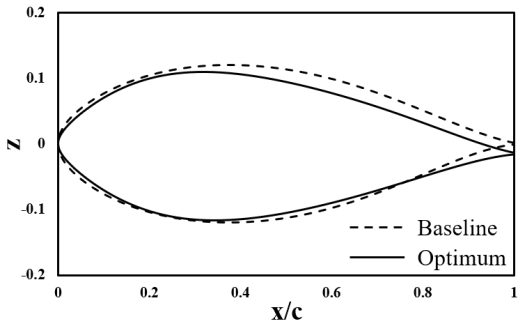


Figure 2.8: Comparison of the baseline and optimum airfoil shape of single-objective optimization.

Then, validation of the VAE model is performed. The VAE model reduces 199-dimensional C_p distribution data to a two-dimensional latent space and reconstructs it back to 199-dimensional data. The trained decoder in this study is used as a data generator that receives a two LVs and creates 199-dimensional C_p distribution data from them. However, if the generated distribution cannot represent the overall training data or has completely different characteristics from them, the optimization results from this generator become inaccurate and inef-

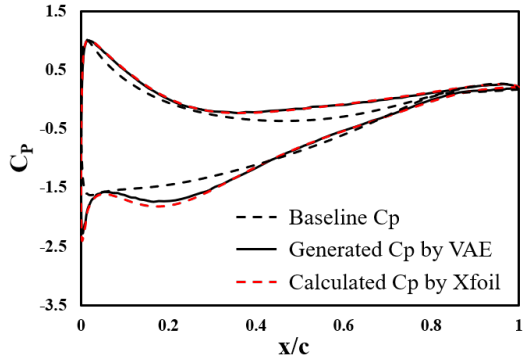


Figure 2.9: Comparison of the generated and calculated pressure distribution of the optimum airfoil (baseline pressure distribution is also included).

Table 2.3: Summary of the QoIs of the optimum solution

	L/D	C_d	C_m	Area $[m^2]$
Baseline	51.93	0.01436	-0.0040	0.1574
Optimum predicted (MLP)	72.22	0.01301	-0.0173	0.1420
Optimum calculated (Xfoil)	71.52	0.01305	-0.0172	0.1426
Error (Xfoil vs MLP) [%]	0.98	-0.28	0.72	-0.46
Comparison with baseline [%]	39.08	-9.40	332.50	-9.82

ficient. Therefore, the generated C_p distributions by the trained VAE decoder are analyzed. Fig. 2.10 shows 50 randomly selected C_p distributions from the 500 initial training C_p data (black lines, Fig. 2.10a) and 50 randomly generated C_p distributions by the decoder (red lines, Fig. 2.10b). Herein, the following points are confirmed. First, the data generated using the decoder covers the range of the training data. Moreover, although no other technique is applied to smoothen the C_p distribution during the VAE training, the decoder successfully generated smooth distributions indistinguishable from the training data.

This supports the reason for adopting a VAE in this framework instead of a GAN; the VAE generates sufficiently realistic data (continuous C_p in this case) without adopting auxiliary layers or filters to ensure the continuity of the data. Second, from the generated C_p distributions, we can also identify their dominant features. As the black dashed box indicates, significant shape differences are observed near the suction peak (near the leading-edge of the airfoil's upper surface), whereas most differences in the other regions are just slight shifts in the C_p values. From the fact that these dominant features near the suction peak are also observed in the training data (Fig. 2.10a), it can be concluded that the VAE model successfully learned the dominant characteristics of the training data. In summary, since we have confirmed that the data generated by the VAE can be well representative of the training data, the trained VAE model will perform successfully as a data generator in this framework.

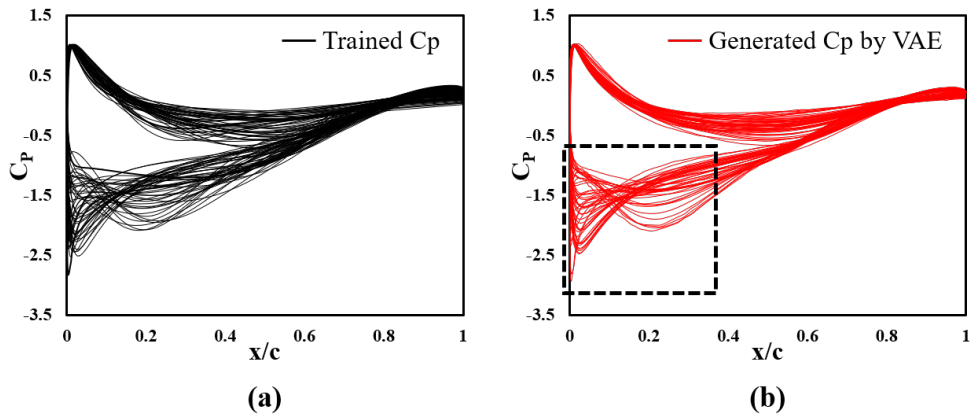


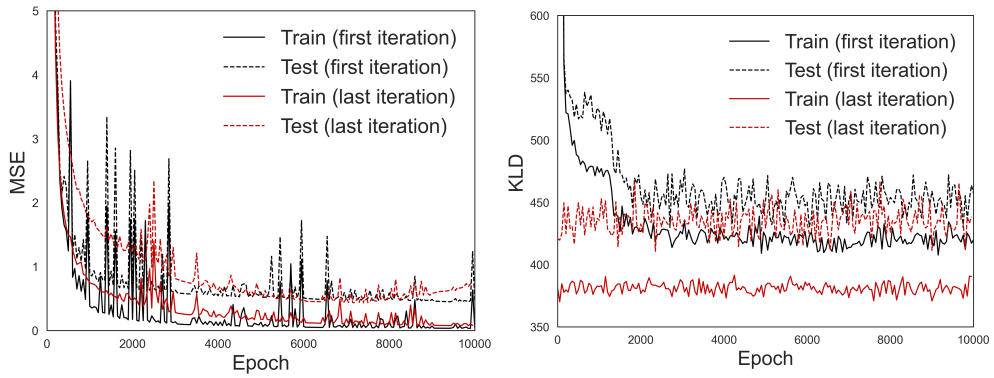
Figure 2.10: Comparison of 50 randomly selected C_p training data (black lines, a) and 50 generated C_p distributions by the VAE (red lines, b). The black dashed box near the leading-edge of the lower curve indicates clear distinctions between generated distributions.

2.4.4 Multi-objective optimization results and discussion

In Sec. 2.4.3, it was confirmed that the proposed inverse design optimization framework yields outstanding results when single-objective optimization is performed. In this section, results of multi-objective optimization are presented to ensure the universality of the framework in various optimization problems. In the previous single objective problem, which uses L/D as an objective function and airfoil area as a constraint, the area of the optimum solution barely satisfied the imposed constraint with little margin, 0.6%. This indicates that a potential increase in the objective function L/D is suppressed by the area constraint. Therefore, in the multi-objective optimization, these two QoIs are set as objective functions to consider their trade-off relationships between aerodynamic and structural performance. Refer to Table 2.2 for the problem definition of multi-objective optimization.

The active learning process of the multi-objective optimization converged after 59 iterations and the loss function histories of MLP and VAE are shown in Fig. 2.11. Accordingly, 677 data consisting of initial 500 designs and 177 infilled designs are calculated (the total learning time for 59 iterations is (508+627) + (186+226) * 59 = 25,443 s). The resultant Pareto frontier of the two objective functions is shown in Fig. 2.12. We observed a discontinuity in the middle of the calculated Pareto solutions, and it is concluded that this is due to C_d constraint violation (the Pareto frontier from the optimization without the C_d constraint is smoothly connected: this result is shown in Sec. 2.6). Among the Pareto solutions, six designs (A1-A3 and B1-B3) are selected for further analysis, and their airfoil shapes are shown in Fig. 2.13. Herein, it can be seen that along with the Pareto frontier, airfoil shapes of selected six designs change sequentially: as the performance criterion changes from area to L/D (from A1 to B3), the airfoil

thickness gradually decreases. Xfoil is performed on these six designs to estimate the error between the QoIs from framework prediction and solver calculation, as in single-objective optimization. The corresponding results are summarized in Table 2.4: the errors between the QoI predicted using the framework and those obtained using Xfoil are within a reasonable range. In particular, A2 and B1 have quite large errors because there are few points added nearby in the active learning process (the percentage errors of C_m are also large, but this is due to their scale). The accuracy near these points can be increased by adding points close to them (it is up to the engineer where to infill points in the Pareto solutions). The accuracy of the trained MLP model is evaluated in Fig. 2.14 by verifying that the six selected designs actually have C_p distributions generated by the VAE: it shows that the MLP is accurate enough in that the C_p calculated using Xfoil and that generated using the VAE are almost indistinguishable, as in single-objective optimization.



(a) (b)

Figure 2.11: Loss history of (a) MLP, and (b) VAE. For both models, the history of the first iteration and last (59th) iteration of active learning is represented.

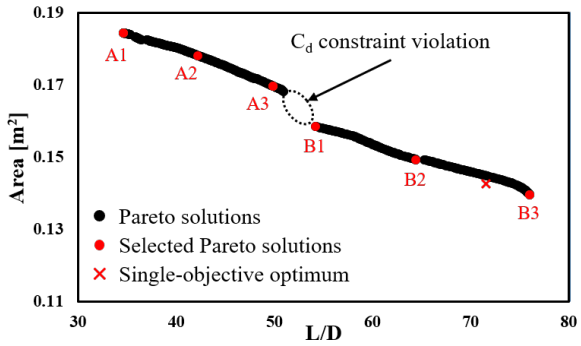


Figure 2.12: Pareto solutions of multi-objective optimization. The discontinuity in the Pareto solutions is due to C_d constraint violation.

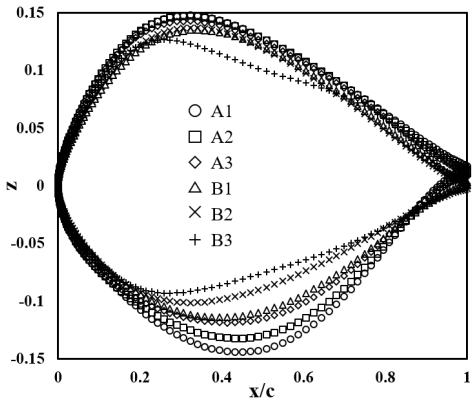


Figure 2.13: Airfoil shape comparison of six selected Pareto solutions.

In this framework, the LVs go through the VAE decoder and MLP sequentially to predict the QoI values. Based on this two-step approach, an optimization technique is applied to obtain LVs that maximize/minimize QoIs. Therefore, for efficient optimization through this two-step deep learning approach, two-step deep learning models should learn the precise correlation between the two spaces (the latent space and QoI space). However, in real engineering problems, this can be difficult due to abrupt changes in physical conditions such

Table 2.4: Summary of the QoI of six selected Pareto solutions

		L/D	Area $[m^2]$	C_d	C_m
Baseline		51.93	0.1574	0.01436	-0.0040
	Predicted	34.56	0.1844	0.01436	0.0513
A1	Calculated	34.26	0.1840	0.01445	0.0514
	Error [%]	-0.85	-0.21	0.63	0.17
	Predicted	42.16	0.1781	0.01434	0.0318
A2	Calculated	40.63	0.1789	0.01461	0.0343
	Error [%]	-3.64	0.44	1.91	7.71
	Predicted	49.86	0.1697	0.01423	0.0124
A3	Calculated	49.52	0.1697	0.01428	0.0124
	Error [%]	-0.68	0.03	0.35	0.19
	Predicted	54.13	0.1585	0.01436	-0.0027
B1	Calculated	55.22	0.1579	0.01442	-0.0075
	Error [%]	2.01	-0.38	0.43	179.92
	Predicted	64.42	0.1493	0.01433	-0.0256
B2	Calculated	64.99	0.1488	0.01445	-0.0333
	Error [%]	0.89	-0.36	0.86	30.05
	Predicted	75.99	0.1396	0.01307	-0.0387
B3	Calculated	76.32	0.1396	0.01299	-0.0383
	Error [%]	0.43	-0.05	-0.63	-1.12

as shock waves. When the mapping is inaccurate, the efficiency of the optimization technique based on their mapping will be significantly undermined. In this context, this study investigates the mapping between the latent space and QoI through heatmaps to verify how they are correlated, as shown in Fig. 2.15:

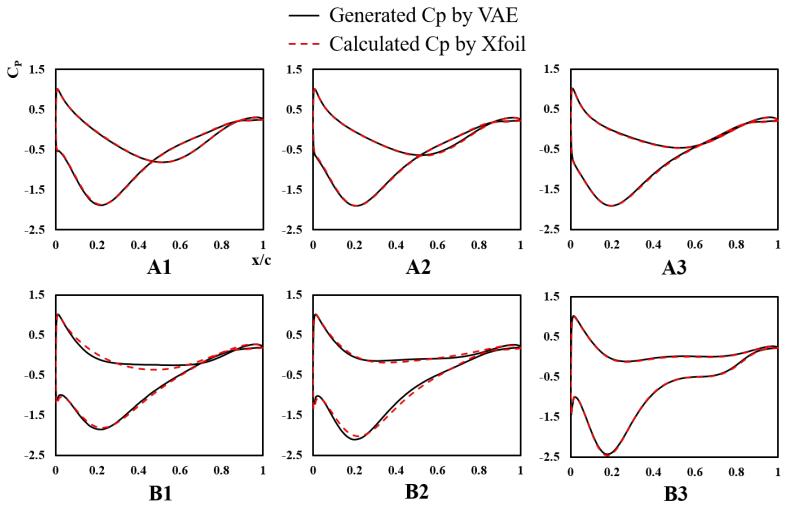


Figure 2.14: Comparison of generated and calculated C_p distributions of six selected Pareto solutions.

heatmap of L/D and area is shown in Fig. 2.15a and Fig. 2.15b, respectively. In this figure, it can be observed that the latent space is mapped continuously to both objective functions, indicating how optimization in this framework could be performed successfully. On the other hand, sharp changes in both objectives are detected when $z2$ has a value of approximately 0.55 (as indicated by the yellow squares). To investigate the changes occurring in the flowfield across this boundary, a total of 12 points are extracted nearby, as shown at the top of Fig. 2.15. The nomenclatures of these 12 points are summarized in Table 2.5.

To find the reason for the sharp change in QoIs at $z_2 \approx 0.55$, the C_p distributions generated through VAE from 12 selected LVs are shown in Fig. 2.16 (the horizontal and vertical axes of the subplots represent x/c and C_p, respectively, and they are scaled to the same range for the comparison). In this figure, the local leading-edge suction peaks on the lower curve are not observed in the C_p plots of $z_2 > 0.55$, whereas they are observed in the plots of $z_2 < 0.55$. There-

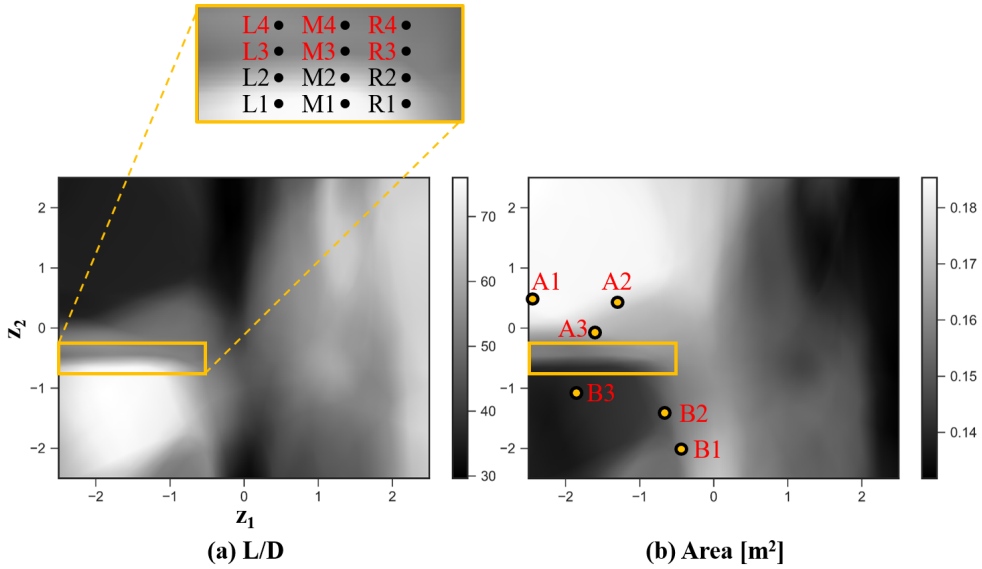


Figure 2.15: Heatmaps of two objective functions within the latent space: (a) L/D and (b) area. Twelve points are selected to investigate the rapid change at $z_2 \approx 0.55$ (top), and the latent space of six selected Pareto solutions is shown in the heatmap of area (b).

Table 2.5: Nomenclatures of twelve points extracted to investigate the sharp changes in the QoI heatmaps

		z_1		
		-2	-1.5	-1
z_2	-0.4	L4	M4	R4
	-0.5	L3	M3	R3
	-0.55	*Boundary of rapid change*		
	-0.6	L2	M2	R2
	-0.7	L1	M1	R1

fore, it can be inferred that $z_2 \approx 0.55$ is the boundary between the presence and absence of the leading-edge suction peak. In this regard, it is known that the sharper the leading-edge of the airfoil, the larger the suction peak behind the leading-edge (as the leading edge radius decreases, the angle at which the flow should bend increases, thereby increasing the suction in order to attach the flow to the airfoil surface [102]). Therefore, the trends in the leading-edge radius ($R_{L.E.}$) are depicted in Fig. 2.17 to verify whether the rapid changes in C_p are accompanied by the changes in $R_{L.E.}$. Indeed, as z_2 decreases (as the number corresponding to the second character of the nomenclature decreases), the leading-edge radius decreases, which is consistent with the prior knowledge. Moreover, in all cases of L, M, and R, there are noticeable gaps between 2 and 3. Considering that nomenclatures 1, 2, 3, and 4 are equally spaced in the z_2-direction, it can be concluded that a rapid change in the leading-edge radius at the boundary between 2 and 3 ($z_2 \approx 0.55$) leads to a sudden change in the trend of the leading-edge suction. Additionally, the six points selected from the Pareto solutions in Fig. 2.12 are scrutinized in a similar way. In Fig. 2.14, the leading-edge suction peaks are not found in the C_p distributions of A1-A3, but are found in B1-B3. In fact, when the latent variables of these designs are visualized as shown in Fig. 2.15b, the two groups (A1-A3 and B1-B3) are separated by a boundary of $z_2 \approx 0.55$. In summary, by analyzing the heatmaps, mapping between the latent space and QoIs using the two-step approach is verified to be generally continuous. Additionally, this mapping is verified to accurately reflect the rapid changes in QoIs, which occur frequently in real-world engineering applications. This flexibility of the two-step deep learning approach enables the optimization to be performed efficiently owing to the continuous and accurate mapping between the optimization inputs and outputs, highlighting the capability of the proposed inverse design optimization framework.

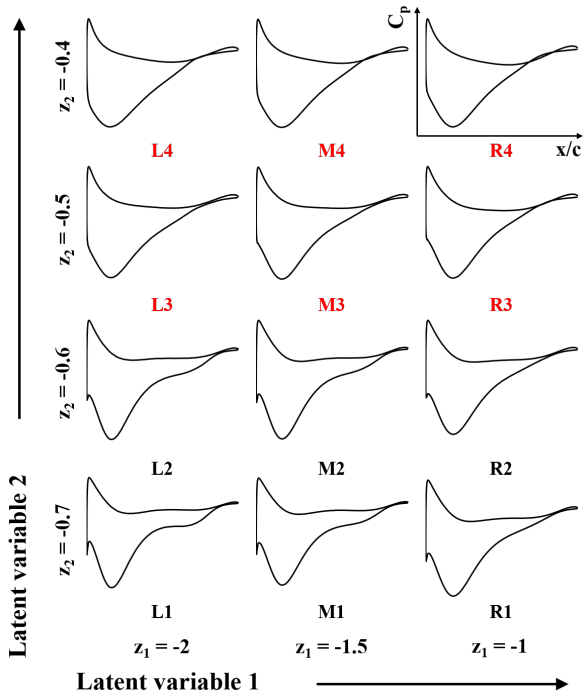


Figure 2.16: C_p distributions of 12 points selected in Fig. 2.15.

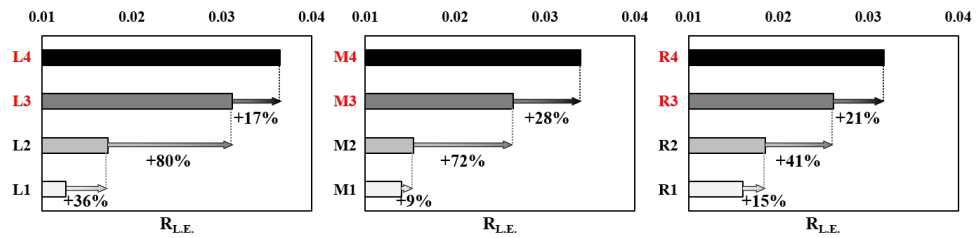


Figure 2.17: Trends in the leading-edge radius ($R_{L.E.}$) of 12 selected points in Fig. 2.15.

2.5 Summary

This study proposed a novel inverse design optimization framework with a two-step deep learning approach, which refers to consecutive coupling of VAE and MLP. Herein, the VAE generates a realistic target distribution and MLP predicts QoIs and shape parameters from the generated distribution. Then, the target distribution was optimized based on this two-step approach. To increase the accuracy, we used active learning to retrain the models with newly added designs. Herein, transfer learning was coupled to reduce the computational cost required for retraining. These techniques increase the accuracy of the framework with efficient computational resources. Finally, the limitations of previous inverse design studies can be eliminated through the proposed framework as follows:

1. The conventional inverse design process is substituted with the MLP surrogate model so that the iterations coupled with the flow solver are not required.

2. From the target distribution, the MLP surrogate model not only predicts the design shape, but also QoIs. Therefore, theoretical/empirical assumptions are not required for predicting QoIs from the target distribution, and the geometric constraints can be imposed explicitly in the target distribution optimization process.

3. For the inverse design optimization, realistic target performance distributions are generated using a VAE deep generative model so that the loss of the diverse representation capacity due to the parameterization of the distribution is mitigated, and excessive constraints for the realistic distribution are not required.

The proposed framework was validated using two constrained optimization problems: single-objective and multi-objective airfoil optimizations of the tip region of a megawatt-class wind turbine blade. In the single objective optimization, the prediction accuracy of the trained MLP model and the validity of the trained VAE model for generating realistic data were verified. In the multi-objective results, continuous mapping between the inputs and outputs of the framework was verified, which enabled successful optimization through the two-step approach. Furthermore, this mapping was confirmed to accurately reflect the rapid changes in QoIs, which occur frequently in real-world engineering applications. In summary, the results of the optimizations show that the proposed inverse design optimization framework via a two-step deep learning approach is accurate, efficient, and flexible enough to be applied to any other inverse design problem.

Considering that this novel framework can be coupled with any numerical solver with arbitrary design shape and QoIs, it can be easily applied and extended to various engineering fields. Moreover, the deep learning models in the two-step approach can be replaced by other suitable alternatives: the VAE by any data generator model and the MLP by any other regression model. For instance, the MLP, a widely used but quite simple model, was used in this study since the problem we investigated is not much complex to apply other advanced deep learning models. However, when the problem is complicated, other models such as convolutional neural networks or recurrent neural networks can be used instead of a simple MLP model. This flexibility contributes to the versatility of the framework by allowing it to be utilized with any model suitable for the engineering problem to which it applies.

2.6 Additional results

The Pareto frontier from the optimization without the C_d constraint is shown in Fig. 2.18: from the continuous Pareto solutions, it can be inferred that the discontinuity in the Pareto solutions in Fig. 2.12 was due to C_d constraint violation. Also, as the constraint is eliminated, Pareto solutions without C_d constraint have better performance near B1 design than Pareto solutions with C_d constraint.

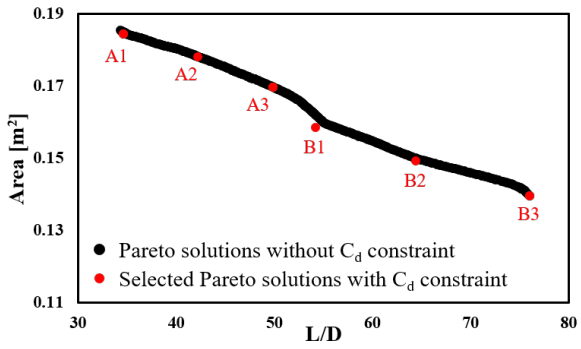


Figure 2.18: Pareto solutions of multi-objective optimization without C_d constraint. For comparison with Fig. 2.12, six designs previously selected from the Pareto solutions with C_d constraint are also shown.

Chapter 3

High-dimensional output space

The work in this chapter was published in the Physics of Fluids [20].

3.1 Introduction

This chapter aims to address the high-dimensional output space that hinders the efficient application of regression models to aerodynamic design by adopting ROM techniques. Specifically, this chapter focuses on the fact that since latent space acts as an intermediary in ROM for predicting high-dimensional data, it inevitably affects ROM performance. To this end, the prediction of high-dimensional flow fields around a transonic airfoil is adopted as a case study, and the physical interpretability of the latent space among various machine learning-based DR techniques is investigated. Furthermore, the impact of its interpretability on the ROM performance is analyzed, and finally, its significance on the accuracy and efficiency in predicting high-dimensional QoIs is verified. The remainder of this section highlights why the selected case study

(flow field prediction of transonic airfoil) is important in the aerodynamic design discipline, and how this chapter successfully leverages physics-aware latent space in the ROM procedure, alleviating the impediment arising from the high-dimensional output space.

CFD has been widely applied in numerous engineering disciplines. However, high-fidelity CFD simulations are often computationally intensive due to the fine discretization of space and time domains. A practical approach to reducing its computational burden is to use a regression model as a cost-efficient alternative to time-consuming flow analysis [19]. Such regression models have been developed extensively, to name a few, cubic spline interpolation [103], RBF [104], and GPR [105, 106]; however, these models are only suitable for estimating scalar quantities such as lift coefficients of the airfoil. Consequently, when the QoIs are high-dimensional vectors, such as pressure or the velocity field, the computational complexity becomes prohibitive due to "the curse of dimensionality."

Data-driven ROM has recently attracted attention for its potential to deal with this problem. It treats a high-fidelity computer simulation as a "black-box function" and generates simplified models in a data-driven manner without any modification of the governing equation. The main purpose of this approach is to reduce the degrees of freedom of the data using the DR technique (also known as representation learning or manifold learning), which finds the low-dimensional latent representation of high-dimensional original data. Through this technique, the dimensionality of the data can be significantly reduced to a level that is suitable for training the regression models effectively.

One of the most widely used DR techniques is POD [107], which is also referred to as principal component analysis or the Karhunen-Loève theorem. Given the training data, POD extracts a set of orthogonal bases (also referred to

as modes) that maximizes the variance of the projected data. Extracted modes are ranked by their energy content so that the dimensionality can be reduced by truncating non-dominant modes. Specifically, the original high-dimensional data can be represented in a low-dimensional subspace using linear combinations of the dominant modes. However, its linearity makes POD-based ROM suffer from performance degradation in nonlinear problems [108, 109, 110, 111], requiring an excessive number of modes compared to nonlinear DR methods for the same reconstruction accuracy.

In this regard, deep neural networks (DNNs) have become an alternative to POD for their performance on highly nonlinear problems. The most commonly used DNN-based DR technique is the AE, which learns the low-dimensional latent space of the original data in an unsupervised manner [112]. The nonlinear functions between its multiple hidden layers enable it to model nonlinearity, and in this context Hinton and Salakhutdinov [113] referred to AE as a nonlinear generalization of POD. Owing to this property, it was confirmed that the AE-based ROM shows higher reconstruction accuracy than the POD-based ROM in various nonlinear problems [114, 115, 116]. Moreover, the flow field reconstruction using AE can be further improved by applying convolutional neural networks (CNNs), which were introduced to train grid-pattern data efficiently [117, 118, 119, 120, 121, 122, 123].

However, there are several issues to be addressed in AE-based ROM. First, AE does not guarantee disentangled latent representation since its training algorithm only aims to reduce the reconstruction error without considering the regularity in latent space. Its irregularity allows multiple features to be entangled in a single latent variable (LV) [75], making it difficult to interpret the physical meanings of LVs. Second, since the model architecture of AE should be predetermined before training, its latent dimension needs to be blindly selected

large enough to contain the entire information (otherwise, a trial-and-error process should be performed). In POD, it is possible to truncate trivial LVs since it provides the energy contents of LVs (or modes). However, in AE, its algorithm does not treat LVs hierarchically according to their information intensity so that theoretically, the entire information is evenly distributed to each LV. Therefore, all dimensions blindly selected should be used, as truncation of even only one LV can severely compromise its reconstruction accuracy. As a result, the resultant redundancy of the latent dimension will degrade both the physical interpretability of LVs (too many LVs are entangled) and the efficiency of AE-based ROM (too many regression models should be trained).

The introduction of a VAE and its improved understanding has been providing a clue to these problems. While AE only minimizes reconstruction error, VAE also regularizes latent space by minimizing KL-divergence term [64]. The balance between these two terms in VAE can be adjusted by the hyperparameter β, which is referred to as β-VAE [75]. It is known that by weighting the KL-divergence term in β-VAE, two notable effects can be achieved. First, it learns disentangled latent representations: a single LV encodes only one representation feature of the original dataset and therefore becomes interpretable. This ensures independence between each LV as the POD extracts orthogonal bases. Second, the regularization loss (KL-divergence) encourages learning the most efficient latent representation [75, 124, 125]. Therefore, regardless of the predetermined latent dimension, only necessary LVs are automatically activated to contain the essential information [126]. In summary, unlike AE model, β-VAE enables its latent space to be information-intensive without being entangled, indicating its potential to further improve the performance of AE-based ROM.

However, there are few studies related to β-VAE in the field of fluid dynamics. Eivazi et al. [126] adopted β-VAE to extract interpretable nonlinear

modes for time-dependent turbulence flows and proposed a method to rank LVs by measuring their energies. However, their ranking method requires separate post-processing of the reconstructed data after model training, and the energies of LVs are indirectly calculated in the output space rather than in the latent space. Accordingly, the approach is unrelated to KL-divergence which is the main cause of information discrepancies between LVs. Though they finally concluded that their framework successfully extracted interpretable features of turbulent flows, it lacks objectivity since it was based on the visual analysis of the flow fields. Last but not least, their research ended up with nonlinear mode decomposition without showing how interpretable LVs can be efficiently utilized for the ROM process. Indeed, Wang et al. [118] already have applied β-VAE to ROM for transonic flow problems and verified its superior performance over POD. Though they extended β-VAE to practical application, they only focused on the implementation of β-VAE for ROM so that the interpretability of extracted features (main purpose of β-VAE) and their effects on ROM performance were not addressed and referred to as their future work.

This study aims to utilize physically interpretable and information-intensive LVs obtained by β-VAE, which are referred to as "physics-aware LVs", for the efficient ROM process. For this purpose, a two-dimensional (2D) transonic flow problem is adopted as benchmark case, and the independence and information intensity of LVs are investigated first to confirm whether they are physics-aware. Herein, we suggest applying the KL-divergence to rank LVs by their amount of information, which is the direct cause of their discrepancies and does not require the reconstruction process. Then, "physics-aware ROM", ROM which utilizes only physics-aware LVs is proposed for its efficiency in that the number of required regression models can be reduced significantly. The presented physics-aware ROM is compared with conventional ROMs, and finally, we successfully

verify its validity and efficiency.

The rest of this chapter is organized as follows. In Sec. 3.2, the β-VAE DR technique is described in detail, and in Sec. 3.3, the physics-aware ROM is newly proposed. In Sec. 3.4, the setup of the numerical experiment is presented. In Sec. 3.5, the process of extracting physics-aware LVs and the discovery of their actual physical meanings are described, and the effectiveness of physics-aware ROM based on them is investigated. And finally, in Sec. 3.6, the summary of this chapter and future work are presented.

3.2 β-variational autoencoder (β-VAE)

Higgins et al. [75] focused on the trade-off relationship between the reconstruction error and KL-divergence in the loss function of the VAE (Sec. 2.2.3). As the reconstruction accuracy increases, the degree of regularization in the latent space decreases. To tune the balance between these two performances, Higgins et al. [75] proposed the β-VAE, which can control the relative importance of the KL-divergence term using the adjustable hyperparameter β. It is a simple modification of the VAE: they have exactly the same structures but slightly different loss functions. The loss function of the β-VAE is as follows:

$$\mathcal{L}_{\beta\text{-}VAE} = \frac{1}{N}\sum_{i=1}^{N}(\mathbf{x}_i - \tilde{\mathbf{x}}_i)^2 + \beta \cdot KL(q_\phi(\mathbf{z}|\mathbf{x})||p_\theta(\mathbf{z})). \tag{3.1}$$

The only difference in the loss functions between the VAE and β-VAE is whether the KL-divergence term is weighted by hyperparameter β. By introducing this hyperparameter, Higgins et al. [75] achieved quantitative and qualitative improvements in the disentanglement within the latent representations over the traditional VAE model. Finally, they concluded that owing to the disentangling performance of β-VAE, the interpretable representations of the independent generating factors of the given dataset can be discovered automatically. In this chapter, the terminologies "disentanglement", "interpretability", "independence", and "orthogonality" are used interchangeably to refer to the following property of LVs: single LV stands for a single representation feature without the intervention of other LVs. However, what makes β-VAE special is not only its disentangling performance. The regularization loss (KL-divergence) is known to have sparsification effect to encourage the most efficient representation learning [75]. Burda et al. [124] and Sønderby et al. [127] practically confirmed this effect in that some LVs become inactive during the training process of VAE. In par-

ticular, Burda et al. [124] confirmed that inactive LVs have a negligible effect on the reconstruction. This finding indicates that VAE trains its LVs in an efficient manner by selectively activating LVs to contain the only necessary information. Since the β-VAE model has a regularization term in the loss function weighted by the hyperparameter β, it can be easily inferred that the sparsification effect in the β-VAE model will become more dominant as β increases. Taking these two outstanding advantages of β-VAE, disentangled and information-intensive latent space, a novel ROM framework is proposed in the next section.

3.3 Physics-aware reduced-order modeling

The main goal of ROM is to predict high-dimensional data with low computational cost. As described in previous sections, the high-dimensional data can be effectively reduced to the low-dimensional LVs, and reconstructed back to high-dimensions through DR techniques. In this context, ROM aims to predict high-dimensional data efficiently by predicting these LVs from input parameters using regression models. The overall structure of ROM is illustrated in Fig. 3.1. In the reconstruction process (solid arrows), the high-dimensional data is encoded into LVs and reconstructed back into high-dimensional data by the decoder. In the prediction process (dashed arrows), LVs are first predicted from input parameters through regression models (GPR [105, 106] is adopted for regression in this study), and then predicted LVs are decoded into high-dimensional data. Since the high-dimensional data can be repeatedly predicted from input parameters at a very low computational cost, this prediction process is often referred to as the online phase.

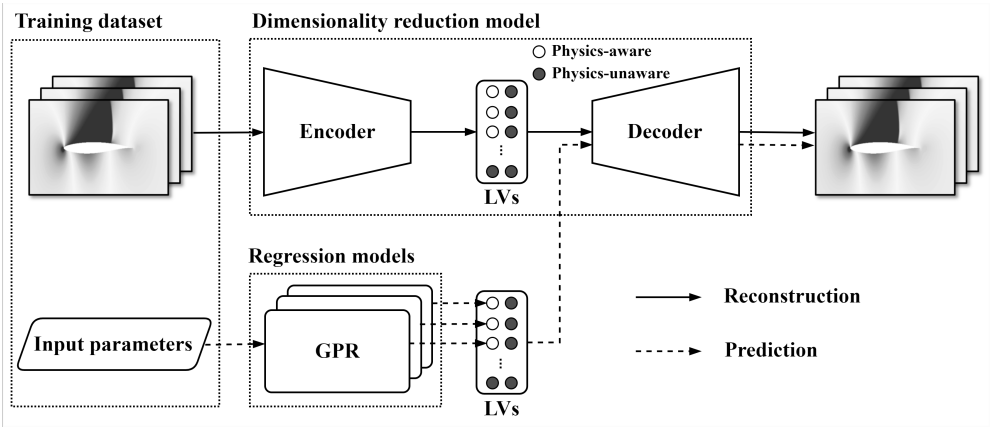


Figure 3.1: Overall structure of physics-aware reduced-order modeling.

Since LVs act as intermediaries in ROM for predicting high-dimensional

data, they inevitably affect ROM performance. In this regard, AE model has two critical drawbacks to be applied to ROM. First, it trains the entangled and therefore uninterpretable latent space. Second, its latent dimension need to be blindly selected large enough since the model architecture should be predetermined before the training. Therefore, the AE-based ROM has the disadvantages of utilizing physically uninterpretable LVs due to entangled LVs and the degraded efficiency due to excessive regression models. This study newly proposes physics-aware ROM via β-VAE to deal with the above issues, considering the following characteristics of β-VAE. As stated in Sec. 3.2, the latent space in β-VAE is trained to be disentangled and information-intensive. When LVs satisfy these two properties simultaneously, they can be regarded as the latent representations which contain both interpretable and necessary information. In that this study uses physical dataset, LVs will correspondingly contain physical information, and accordingly, we refer to those LVs as "physics-aware LVs." To ease the understanding of physics-aware LVs, the ideal schematic of their extraction with β-VAE is shown in Fig. 3.2: when the dataset is generated through Mach number (Ma) and angle of attack (AoA), the ideally extracted physics-aware LVs will be Ma and AoA. Finally, the ROM only utilizing these physics-aware LVs (which refers to "physics-aware ROM") is proposed for its efficiency that the number of regression models required in the prediction process can be significantly reduced.

In order to extract physics-aware LVs, we first estimate the independence of LVs. In this regard, Eivazi et al. [126] have already measured it using a Pearson correlation matrix and the same approach is applied herein. Then, the information intensity of LVs is quantified. Similar attempt has been made by Eivazi et al. [126], who proposed a strategy to rank LVs by energy percentage, which quantifies the contribution of each LV to the reconstruction quality. How-

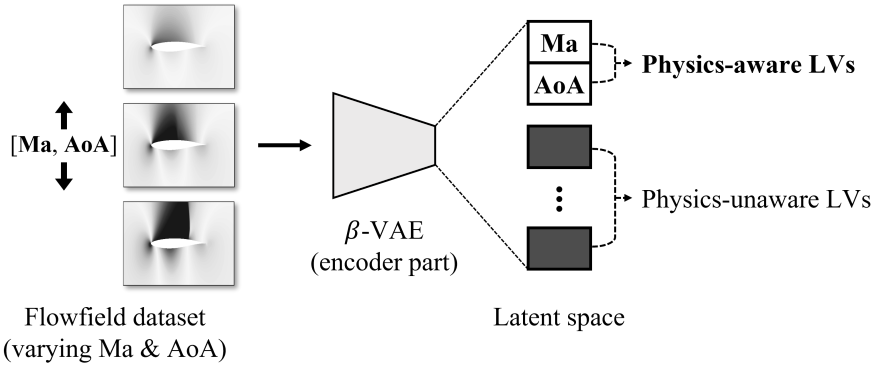


Figure 3.2: Illustrative schematic showing the process of extracting physics-aware LVs by β-VAE: the ideal case is to extract the actual physical parameters (Ma and AoA) from the given dataset.

ever, they ranked LVs not in the latent space which is directly related to them, but in the output space so that there exist two problems. First, its low practicality due to cumbersome post-processing: flow fields should be reconstructed using a forward selection of LVs and then energy percentage is calculated from them. Second, since this method is irrelevant to KL-divergence, which is the main cause of the inactiveness of LVs [124, 127], it cannot be regarded as a fundamental approach for ranking LVs based on their amount of information. Therefore, another ranking criterion should be proposed that estimates the information intensity of each LV without such burdensome post-processing. For this purpose, we propose to apply the KL-divergence (Eq. 2.7), the immediate cause for the sparser latent space due to its regularization effect. Since the calculation of KL-divergence is performed directly in the latent space, the decoder part of β-VAE does not even need to be utilized (no reconstruction required), solving all the limitations of the previous ranking approach.

In physics-aware ROM framework, only physics-aware LVs are utilized so

that the number of regression models required in the prediction process can be significantly reduced. For example, suppose that AE-based ROM and β-VAE-based ROM are performed where both AE and β-VAE have the latent dimension of 16. In the case of AE-based ROM, 16 regression models should be trained because exclusion of just one LV can degrade ROM performance significantly in that all 16 LVs contain information in an entangled manner. However, in β-VAE-based ROM, since physics-unaware LVs are judged not to contain any meaningful information, it is sufficient to utilize only the regression models of physics-aware LVs. The overall procedure of physics-aware ROM is summarized in Algorithm 1.

Algorithm 1 Physics-aware ROM via β-VAE

(1) Preparation of training dataset generated by physical parameters.

(2) Train β-VAE with dataset in (1).

(3) Extract activated LVs through estimating their independence by correlation coefficient and information intensity by KL-divergence.

(4) Train regression models to predict activated LVs from physical parameters in (1).

(5) Prediction of high-dimensional dataset from physical parameters based on regression models in (4) and decoder part of β-VAE trained in (2). During this process, deactivated LVs are fixed to their estimated mean values.

3.4 Numerical experiments

3.4.1 Data preparation

A 2D transonic flow problem, the benchmark case that is widely used in previous ROM studies [110, 111, 109, 118], is adopted for validation of the physics-aware ROM. The transonic flow field is generated by the KFLOW finite-volume-based CFD solver [128, 129]. Specifically, the Reynolds-averaged Navier-Stokes equation is solved coupled with the Spalart-Allmaras turbulence model. RAE 2822 airfoil is selected as the baseline geometry, and a structured O-grid with a shape of 512×256 (wall-tangential direction $\times$ wall-normal direction) is generated; the corresponding grid is shown in Fig. 3.3. The Reynolds number is fixed at 6.5×10^6, and two physical parameters are chosen: Ma and AoA. The design space of each variable is set to [0.5, 0.8] and [0°, 3°], respectively. Latin hypercube sampling is used to generate 500 sample points in 2D parameter space. A flow analysis of these samples is then conducted and the resultant velocity and pressure fields are normalized by their far-field conditions. As the variations of flow properties far from the airfoil are negligible, only the inner half of the entire grid with respect to the normal direction from the airfoil is used so that the resultant grid becomes 512×128. Consequently, a total dataset with a shape of $500 \times 3 \times 512 \times 128$ (dataset size $\times$ flow field components $\times$ wall-tangential direction grids $\times$ wall-normal direction grids) is obtained, where the flow-field components are the x-velocity, y-velocity, and pressure. Finally, the dataset size of 500 is split into a ratio of 9:1 so that the size of the training dataset is 450, and that of the test dataset is 50.

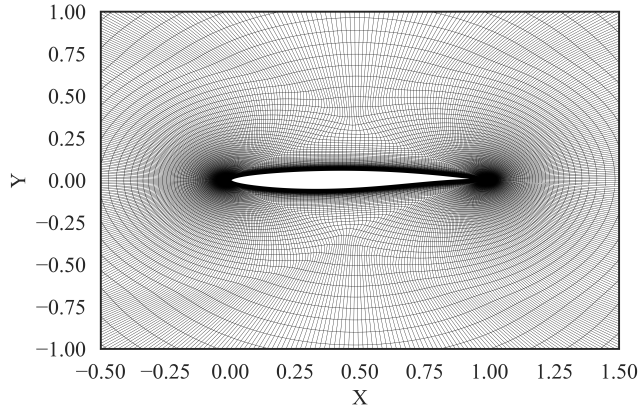


Figure 3.3: Computational grid used for the flow analysis; structured O-grid with a size of 512×256.

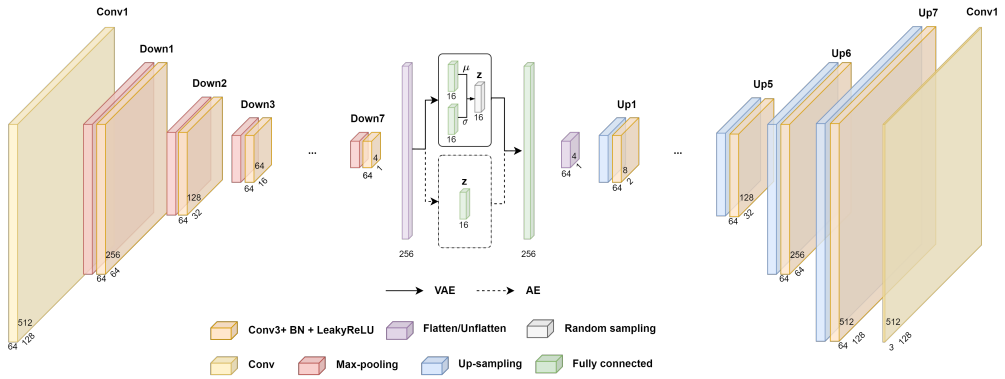


Figure 3.4: Structures of the AE and VAE/β-VAE.

3.4.2 Training details

In this study, AE, VAE, and β-VAE, are trained to investigate their differences in terms of DR for transonic flow. In particular, several β-VAE models are trained ($\beta \in [10, 20, 30, 40, 50, 100, 150, 200, 500, 750, 1000, 2000, 3000, 4000]$) to investigate the effects of the β value (technically speaking, β-VAE can be regarded as the VAE when β has a value of 1). For all models, the

Table 3.1: Details of the blocks and layers of VAE/β-VAE used in this study.

Name	Layer type	Filter	Kernel	Stride	Activation	Batch Norm.
Up	Max-polling	–	2×2	–	–	–
	Convolution	64	3×3	1	LeakyReLU	o
Down	Upsampling	–	2×2	–	–	–
	Convolution	64	3×3	1	LeakyReLU	o
Conv1_in	Convolution	64	1×1	1	LeakyReLU	o
Conv1_out	Convolution	3	1×1	1	–	–
FC	Fully Connected	–	–	–	–	–

Table 3.2: Network structure of the VAE/β-VAE used in this study.

Encoder		Bottleneck		Decoder	
Layer	**Output size**	**Layer**	**Output size**	**Layer**	**Output size**
Input	$3 \times 512 \times 128$	Flatten	256	Up1	$64 \times 8 \times 2$
Conv1_in	$64 \times 512 \times 128$	FC1: μ	16	Up2	$64 \times 16 \times 4$
Down1	$64 \times 256 \times 64$	FC2: σ	16	Up3	$64 \times 32 \times 8$
Down2	$64 \times 128 \times 32$	Resampling	16	Up4	$64 \times 64 \times 16$
Down3	$64 \times 64 \times 16$	FC3	256	Up5	$64 \times 128 \times 32$
Down4	$64 \times 32 \times 8$	Unflatten	$64 \times 4 \times 1$	Up6	$64 \times 256 \times 64$
Down5	$64 \times 16 \times 4$	–	–	Up7	$64 \times 512 \times 128$
Down6	$64 \times 8 \times 2$	–	–	Conv1_out	$3 \times 512 \times 128$
Down7	$64 \times 4 \times 1$	–	–	Output	$3 \times 512 \times 128$

dimension of the latent space should be determined blindly before the model training. The selected dimensions should be sufficient for encoding the training data generated from the 2D parameter domain (Ma and AoA). In this regard,

we adopted the approach suggested by Wang et al. [118] to infer the latent dimension of β-VAE considering the accuracy of the POD with corresponding dimension (their assumption was that POD requires much more dimensions than β-VAE for the equivalent reconstruction accuracy, which was also proved in their work). Finally, the dimension of the latent space in this study is determined to be 16 since it conserves 99.18% in terms of energy contents of POD, which is judged to be sufficient.

An appropriate encoder/decoder structure should be selected to effectively compress/reconstruct the data, from dimensions of $3\times512\times128$ to 16 and vice versa. To determine suitable structures, hyperparameter tuning based on a grid search was conducted. Finally, it was confirmed that the MSE reconstruction error, which represents the overall accuracy of the trained model, does not strongly depend on whether batch normalization, max-pooling, and up-sampling are applied. However, their application significantly affects the smoothness of the reconstructed flow field: it can be inferred that this is due to the effects of batch normalization and max-pooling that prevent overfitting, and the interpolation effect of up-sampling. If an artificial discontinuity (rather than a discontinuity that reflects a physical phenomenon, such as a shock wave) is observed in the reconstructed flow field, the flow field cannot be considered realistic. In this context, these points are important for predicting realistic flow fields using CNN-based deep learning models. Finally, the architectures of the selected models, which are considered to be sufficient in terms of MSE error and the smoothness of the reconstructed flow field, are shown in Tables 3.1 and 3.2 and Fig. 3.4. The only difference between the structures of the AE and VAE/β-VAE is the bottleneck, and the structures of the VAE and β-VAE are exactly the same.

The Adam optimizer is adopted to train selected models. The initial learning

rate is set to 10^{-3}, and it decays at a rate of 0.1 every 1000 epochs. The maximum epoch should be selected carefully because a model that does not fully converge can make significantly different predictions than a converged model [130]. The maximum number of epochs is set to 3000 since it is verified to guarantee sufficient convergence in terms of the loss function. A mini-batch size of 50 is selected so that nine iterations are performed per epoch (as the size of the training dataset is 450). Using a Tesla P100-PCIE-16GB GPU with Pytorch deep learning library [101], the average training time for each model is calculated to be approximately 3 h.

3.5 Results and discussion

In this section, the results of physics-aware ROM are presented in the following order. First, training results of β-VAE are demonstrated to investigate the effect of β on flow reconstruction. Next, the methods to measure the independence and information intensity of LVs are orderly presented to determine whether the LVs obtained by β-VAE are physics-aware LVs. At the same time, the validity of the KL-divergence ranking method to measure the information intensity of LVs is confirmed both quantitatively and qualitatively. Then, it is thoroughly investigated whether the physics-aware LVs actually have interpretable physical features through various techniques. In the end, it is verified that the physics-aware ROM based on these physics-aware LVs has equivalent prediction accuracy much more efficiently than physics-unaware ROM.

3.5.1 Training results

First, the loss function history of the trained models is shown in Fig. 3.5, and the decomposition of the resultant loss functions is shown in Fig. 3.6. More specifically, the loss function is decomposed into the MSE and KL-divergence (as in Eq. 3.1) to confirm their trade-off relationship. The MSE and KL-divergence of the VAE/β-VAE models are indicated by blue and red symbols, respectively.

The β and KL-divergence values cannot be defined in the AE model; therefore, only its MSE loss is indicated separately by a dashed line. Herein, a clear trade-off relationship between MSE and KL-divergence can be confirmed, as mentioned by Higgins et al. [75]. As β increases, KL-divergence decreases, while MSE increases. This is consistent with the concept of the β-VAE, which suppresses KL-divergence by increasing β. In particular, when $\beta > 1000$, the MSE increases significantly so that an accurate reconstruction of the flow field is no

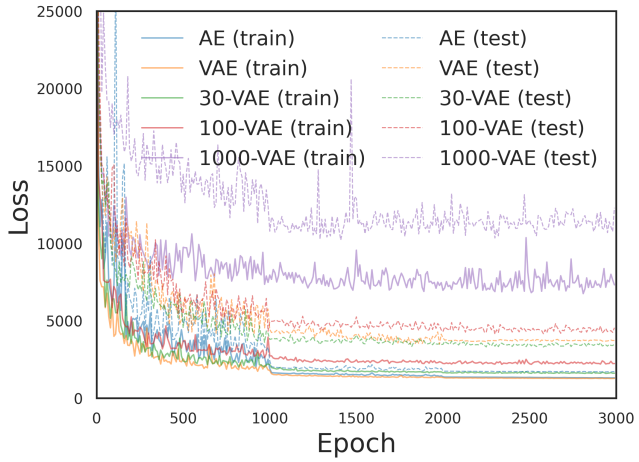


Figure 3.5: Loss history of the trained AE/VAE/β-VAE models.

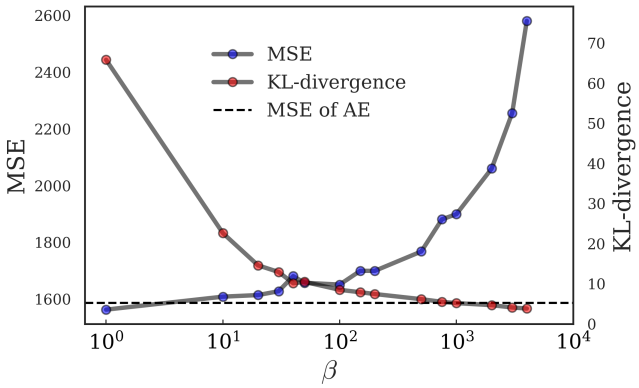


Figure 3.6: MSE and KL-divergence of the trained VAE/β-VAE models.

longer possible.

Second, the reconstructed pressure flow fields of the three test cases (which are not used during the training process) are shown in Fig. 3.7 to compare the trained models more intuitively and visually. The selected test cases are as follows: test case 1 is in the absence of a shock wave ($Ma = 0.61$ and $AoA = 2.12°$), test case 2 is in the presence of a weak shock wave ($Ma = 0.72$ and

$AoA = 1.68°$), and test case 3 is in the presence of a strong shock wave ($Ma = 0.78$ and $AoA = 1.57°$). Five models are compared, including the AE, VAE, 30-VAE (β-VAE with $\beta = 30$), 100-VAE ($\beta = 100$), and 1000-VAE ($\beta = 1000$). In Fig. 3.7, it can be confirmed that the reconstructed pressure contours for all three cases and all models do not exhibit any significant difference, indicating all models have been trained to reconstruct the accurate flow fields.

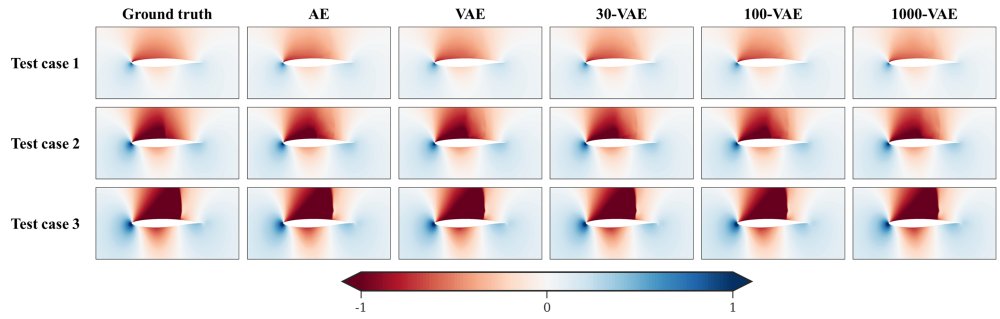


Figure 3.7: Reconstructed pressure fields of the trained models.

3.5.2 Independence of LVs

This section is to confirm whether LVs of β-VAE are actually trained to be disentangled from each other so that they are interpretable. In Fig. 3.8, the absolute values of the components in the Pearson correlation matrix are shown from AE to 1000-VAE. Since there is no algorithm in AE model to promote the independence of LVs, it has the largest values. However, in β-VAEs, their values decrease as β increases, which indicates that the LVs gradually become independent of each other. These results practically prove that the KL-divergence actually encourages the independence of each LV. Eivazi et al. [126] also computed the determinant of the correlation matrix to measure the degree of correlation within the entire set of LVs (when the determinant is

0/1, it indicates that these variables are completely correlated/uncorrelated). However, given the fact that the KL-divergence term leads to a sparser latent space, which will be discussed in detail in Sec. 3.5.3, examining the determinant of the whole matrix size of 16×16 is contradictory.

Therefore, the independence of various LV combinations is further analyzed. First, all possible combinations of two to seven LVs are obtained. Then, the determinants of these combinations are calculated and their statistics are summarized in Fig. 3.9. For any number of LVs used in a combination, the β-VAE models have significantly higher determinants than the AE and VAE models. In summary, the results presented in Fig. 3.8 and 3.9 consistently show that the β-VAE successfully learns uncorrelated LVs compared to AE and VAE models, owing to the KL-divergence, which forces the LVs to be independent. These independent (or uncorrelated) LVs are expected to have interpretable (or disentangled) physical features, which will be investigated in Sec. 3.5.4.

3.5.3 Information intensity of LVs

It was verified in Sec. 3.5.2 that the LVs in β-VAE are trained to contain disentangled and therefore interpretable information. In this section, we tried to rank such disentangled LVs according to their information intensity using KL-divergence. Fig. 3.10 shows the KL-divergence of each LV in AE/VAE/β-VAE. It can be confirmed that both the AE and VAE models do not have any inactive LVs. On the contrary, notable trends are observed in 30-VAE, 100-VAE, and 1000-VAE models: each model has only four (LV index 6, 8, 11, and 15), three (index 8, 11, and 15), and two (index 8 and 11) activated LVs, respectively. These results are consistent with those of Eivazi et al. [126] in that the number of activated LVs decreases as beta increases in the β-VAE model. In fact, a similar approach was applied by Sønderby et al. [127], who judged

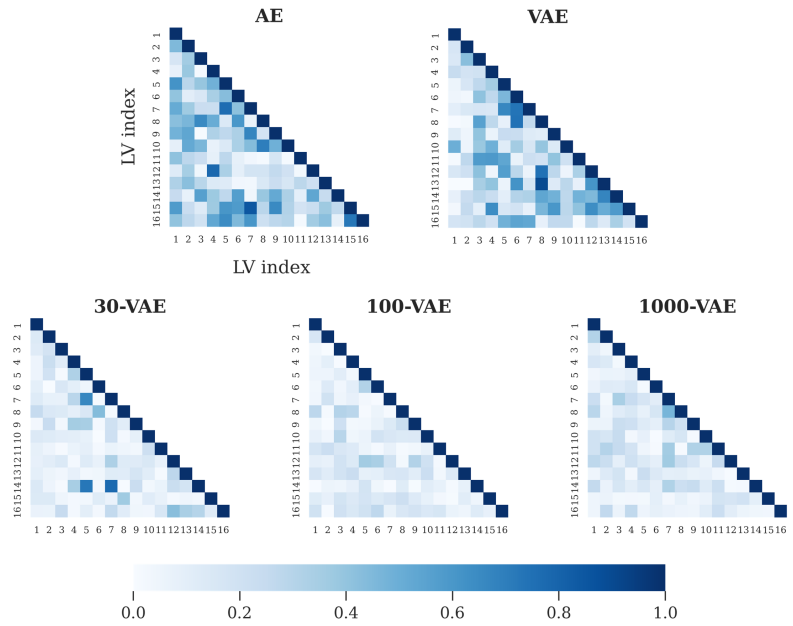


Figure 3.8: Absolute values of the components in the Pearson correlation matrix for LVs.

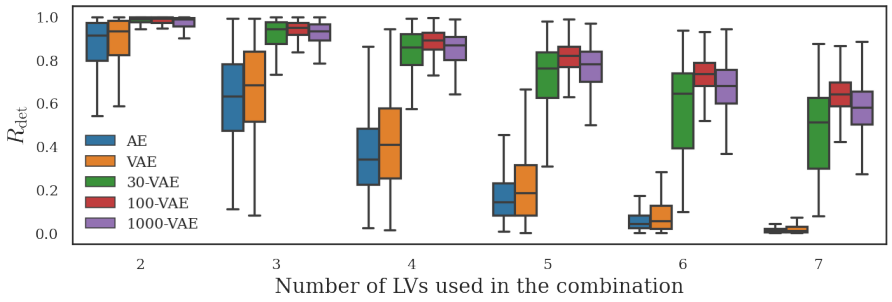


Figure 3.9: Determinants of Pearson correlation matrices for combinations of 2 to 7 LVs.

whether the LV is activated via the KL-divergence. However, the relationship between the activeness of the LV and KL-divergence was not described clearly. Therefore, we also attempt to clarify it: further investigations are conducted to

check whether the LV judged to be more active in terms of the KL-divergence actually has a greater effect on the reconstructed flow field. Sobol sensitivity analysis is utilized for this analysis [131]: a total 18432 latent vectors are sampled using the Saltelli sampler in the range of $[\hat{\mu}_k - 2\hat{\sigma}_k, \hat{\mu}_k + 2\hat{\sigma}_k]$, and their corresponding reconstructed flow fields are obtained through the decoder parts of the AE/VAE/β-VAE models. Since Sobol analysis should be conducted on the scalar output, a set of $3\times512\times128$ pixels of the flow fields is converted to a scalar value (the sum of all pixel components is used in this study, but any scalar value representing the main characteristics of the flow field can be used). Fig. 3.10 shows the calculated first-order Sobol indices. It can be confirmed that the higher the KL-divergence, the larger the value of the Sobol indices in the VAE/β-VAE models (because KL-divergence has nothing to do with the algorithm of the AE model, it does not exhibit a clear trend with the Sobol indices).

Additionally, since the standard deviation represents the dispersion of the variable, which can be regarded as its activeness, estimated standard deviations of the LVs ($\hat{\sigma}_k$) from the training dataset are also investigated. When $\hat{\sigma}_k$ is low for a specific LV, it can be understood that the corresponding LV remains inactive (or less dispersed) in the latent space during training: a situation where the value of the LV does not change even if the input data changes. In Fig. 3.11, it can be confirmed again that both AE and VAE models do not have any inactive LVs (all $\hat{\sigma}_k$ values are larger than 0.5). However, the most interesting point is that compared to Fig. 3.10, LVs with high KL-divergence also have high $\hat{\sigma}_k$ in 30-VAE, 100-VAE, and 1000-VAE models (the LV indices of the activated LVs in terms of KL-divergence exactly match those of $\hat{\sigma}_k$). To sum up, we suggest the use of KL-divergence as a criterion for ranking the activeness (information amount) of LVs for the two reasons. First, in that it is a regularization term

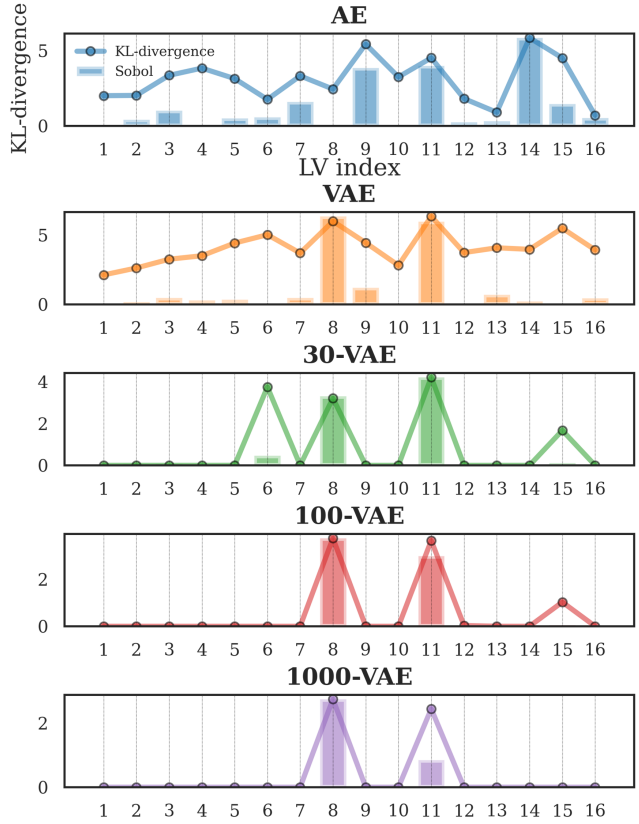


Figure 3.10: KL-divergence and Sobol results with respect to LVs from the training dataset.

that directly causes the inactiveness of the latent space and does not require any cumbersome post-processing. The justification of this decision-making process is successfully performed through intuitive but quantitative investigations by comparing its ranking with the Sobol indices (Fig. 3.10) and the $\hat{\sigma}_k$ from the training dataset (Fig. 3.11).

So far, using the proposed KL-divergence criterion, the inactiveness within the latent space has been investigated in a quantitative manner, but it does not provide straightforward information on what these activated/inactivated

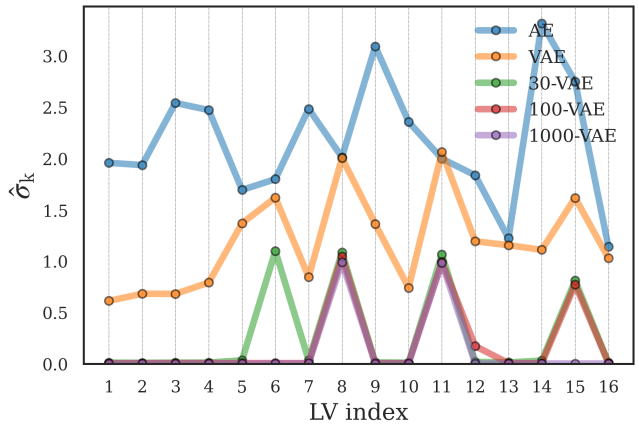


Figure 3.11: Standard deviations of LVs from the training dataset.

variables actually contain. Accordingly, each LV is visualized via a traversal of itself, which is the most widely adopted approach for this purpose (e.g., latent traversal plots of celebA, 3D chairs, and 2D shape dataset in the study by Higgins et al. [75] and those of pressure distribution over an airfoil in the study by Yang et al. [18]): it shows the traversal of a single LV while other LVs remain fixed so that one can visually understand the features of a specific LV without the intervention of other variables. In this study, a latent traversal plot is applied to identify the physical features of the flow field contained in the LV. Fig. 3.12 shows the pressure flow fields for two extreme LVs: one is the most dominant LV, which is ranked first by KL-divergence, and the other is the most trivial LV, which is ranked last. In AE, the pressure field changes abruptly as the most dominant LV changes; when the most trivial LV changes, the field changes gradually, but not as significantly as that of the most dominant LV. Since the sparsification effect does not occur in AE due to the absence of KL-divergence term in the loss function, all the variables are activated and therefore contain uninterpretable (or entangled) physical features. However, in VAE and

1000-VAE models, the most trivial LVs cause indistinguishable variations in the flow fields. This is because the KL-divergence forces inactiveness in the latent space, therefore, only necessary LVs become activated to contain meaningful physical features: it can be regarded as a virtue of β-VAE model. When β increases, information is packed into LVs more compactly [126], as observed in the traversal of the most dominant LV in the 1000-VAE model. This LV can be interpreted as containing information on the occurrence of the shock wave, and the variation it arouses is the largest compared to other models.

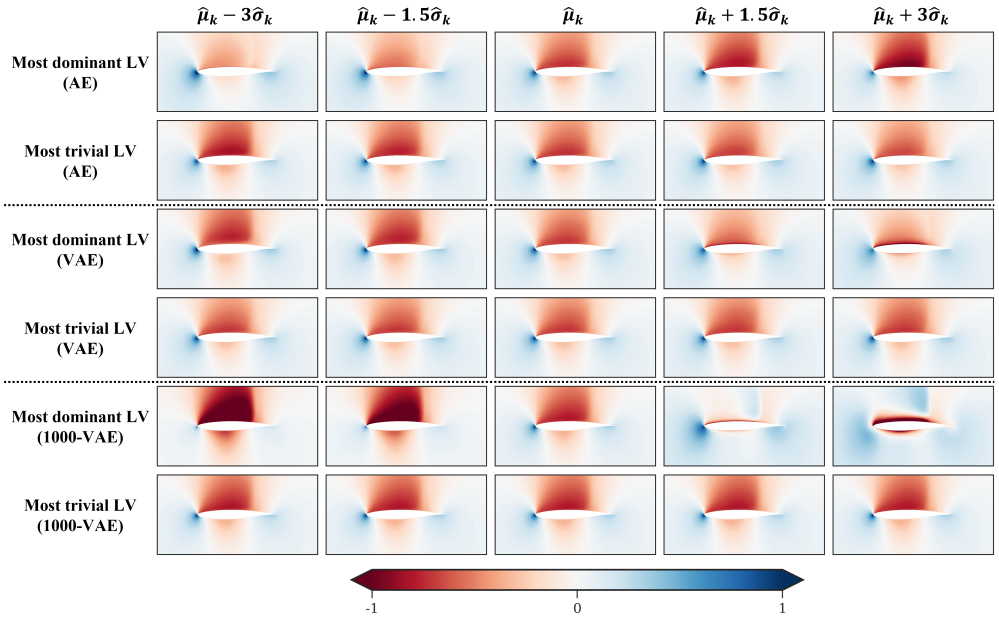


Figure 3.12: Latent traversal plots of pressure flow fields for two extreme LVs: first (most dominant) and last (most trivial) LVs ranked by KL-divergence.

3.5.4 Physics-awareness of LVs

The two requirements for physics-aware LVs are investigated so far: whether LVs are disentangled (Sec. 3.5.2) or information-intensive (Sec. 3.5.3). In this

section, physics-oriented investigations are performed to figure out the physical meanings each physics-aware LV contains.

First, the relationship between the two physical parameters used in this study (whose distribution is shown in Fig. 3.13a) and the top two ranked LVs by KL-divergence (1^{st} LV and 2^{nd} LV) is visually investigated as β varies. Accordingly, in Fig. 3.13b, the distributions of the training dataset with respect to those two LVs are shown. For the AE, VAE, 100-VAE, and 1000-VAE models, the plots of the first column are colored by Ma value, and the second column by AoA. Moreover, to confirm that how physical parameters can be represented by top two LVs, we also draw the trajectories of the boundary data in Fig. 3.13b (they are extracted from the boundary of the physical parameter space, as in Fig. 3.13a). In AE, neither Ma nor AoA have any noticeable trends. In contrast, in VAE, the plot in the first column exhibits a trend that is not clear, but still noticeable: as the $1^{\text{st}}/2^{\text{nd}}$ LV increases, AoA/Ma increases. However, the increase in Ma cannot be explained by the increase in the 2^{nd} LV alone (the same goes for 2^{nd} LV and AoA). These ambiguous correlations between the two LVs and two physical parameters become more clear in 100-VAE (for this case, as the $1^{\text{st}}/2^{\text{nd}}$ LV increases, Ma/AoA increases). Though 100-VAE shows a much more obvious relationship than AE, it can be confirmed that the physical parameters cannot be fully represented only by two dominant LVs in that its trajectory does not cover the entire training dataset. Considering the fact that there are three physics-aware LVs in 100-VAE, this may be an expected result. However, it should be noted here that the top two variables sorted by KL-divergence almost succeeded in representing the physical parameters, whereas when the plot is drawn with the 1st and 3rd dominant LVs, a very irregular pattern is observed. In this regard, the validity of ranking LVs through KL-divergence is confirmed once again. Finally, the 1000-VAE is investigated.

Since this model has two physics-aware LVs, one can expect that in the ideal case, they correspond to the two physical parameters (this ideal situation is depicted in Fig. 3.2). And these conjectures are actually happening in 1000-VAE: the correlations between two physical parameters and two LVs become clear, and the latent space of the training dataset is perfectly closed by its boundary trajectory. Here is the key finding of this study: though β-VAE has no information about the physical parameters used to generate the training data, it effectively extracts only two LVs out of total 16 LVs (especially when β=1000), which correspond to actual physical parameters Ma and AoA. It can be concluded that these marvelous results are owing to the orthogonality effect that makes LVs disentangled and the regularization effect that makes redundant LVs inactive. To make this clear, it should be noted that AE, one of the most widely used nonlinear DR techniques, fails to construct a physics-aware latent space in that it learns all 16 entangled LVs and therefore physically uninterpretable despite the same training dataset as β-VAE.

As the visual analysis in the previous paragraph is qualitative, a quantitative analysis is performed herein to verify whether LV_{Ma} or LV_{AoA} in 1000-VAE corresponds to Ma or AoA (for the sake of brevity, LV_{Ma} refers to the 1$^{\text{st}}$ LV, and LV_{AoA} refers to the 2$^{\text{nd}}$ LV, which implies that LV_{Ma} and LV_{AoA} are the LVs responsible for Ma and AoA, respectively). Single-variable linear regression (LR) models are trained for this purpose (since the relationships between physical parameters and LVs appear linear in Fig. 3.13, the LR model is considered to be sufficient). For example, LR model for Ma is trained with the input variable as LV_{Ma} and the output variable as Ma,which can be expressed as Ma=$f(LV_{Ma})$. The training dataset is the same as in Sec. 3.4.1 and physical parameters and LVs are standardized before training. If LV_{Ma} and Ma (or LV_{AoA} and AoA) have a linear relationship, the fitted LR models will perform well on the test

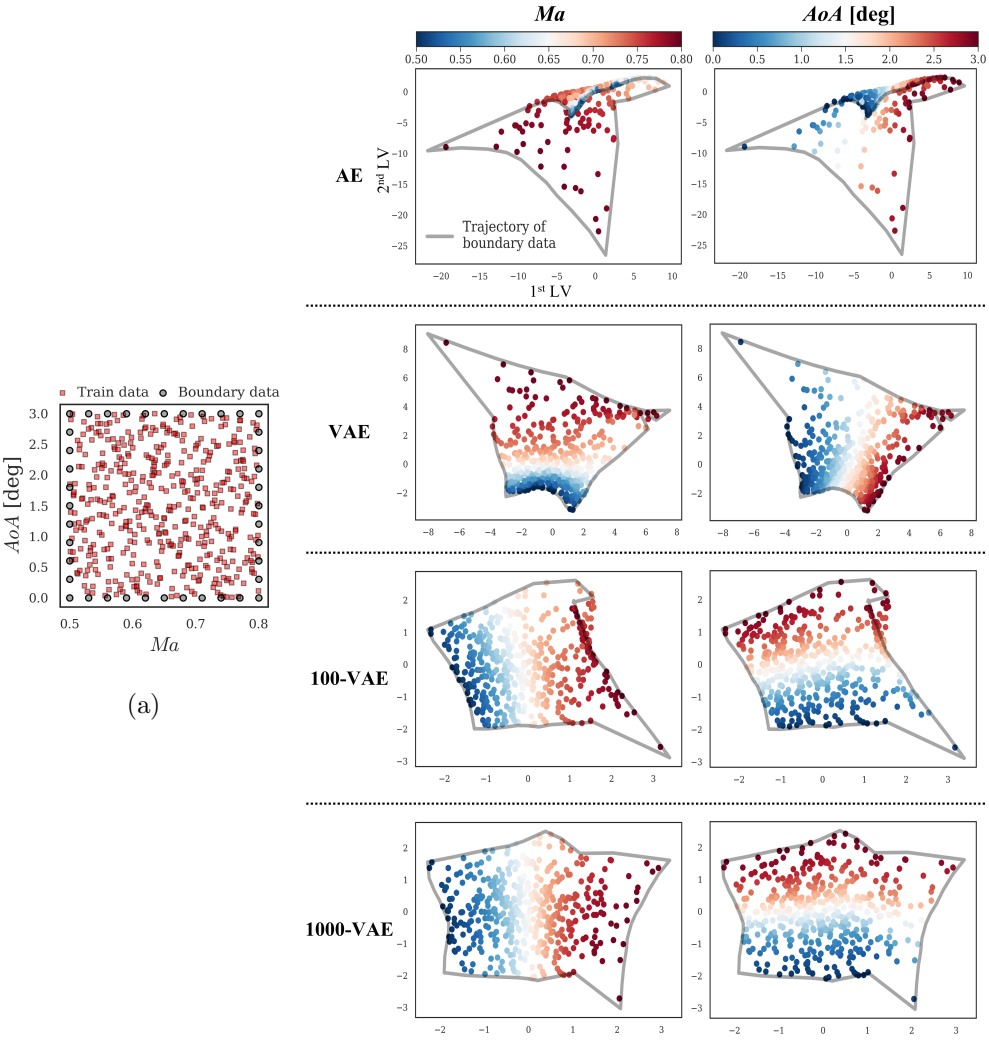


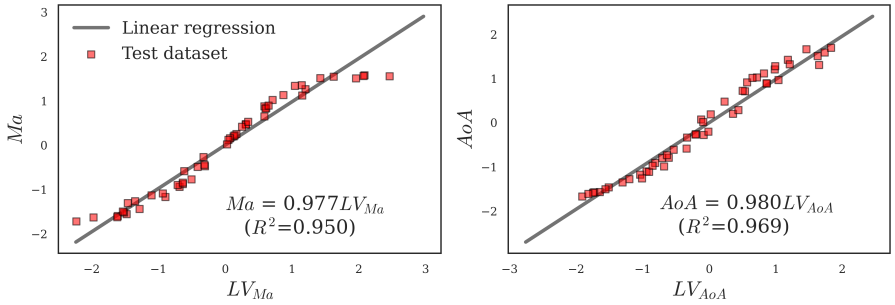
Figure 3.13: Investigation of physical features contained in the top two LVs: (a) distributions of training dataset and boundary data with respect to Ma and AoA, and (b) distributions of training dataset for 1^{st} and 2^{nd} LVs (the left figures are colored by Ma, and the rights by AoA).

dataset. The results are shown in Fig. 3.14. In each subplot, the equation of the trained LR model is also included. For both LR models $Ma{=}f(LV_{Ma})$ and $AoA{=}f(LV_{AoA})$, their equations clearly show that each physical parameter and LV are almost identical in that the coefficients are approximately 1 and the intercepts are 0. Also, the coefficients of determination (R^2) calculated based on the test dataset are 0.950 and 0.969, respectively, indicating that each physical parameter can be represented by the linear relationship of only one LV with sufficient accuracy. In addition, we train two additional LR models with both LVs as input features: one as $Ma{=}f(LV_{Ma}, LV_{AoA})$ and the other as $AoA{=}f(LV_{Ma}, LV_{AoA})$. If Ma needs both LVs for its expression, the R^2 value of the LR model $Ma{=}f(LV_{Ma}, LV_{AoA})$ will be significantly higher than that of $Ma{=}f(LV_{Ma})$. The trained LR models are described as follows:

$$\begin{bmatrix} Ma \\ AoA \end{bmatrix} = \begin{bmatrix} 0.978 & -0.011 \\ -0.017 & 0.981 \end{bmatrix} \begin{bmatrix} LV_{Ma} \\ LV_{AoA} \end{bmatrix}. \tag{3.2}$$

Herein, there are two notable points. First, the coefficient of LV_{AoA} is negligible (approximately 0) compared to that of LV_{Ma} (approximately 1) when modeling Ma, which indicates that LV_{AoA} is redundant variable for representing Ma (same principal applies when modeling AoA). The second interesting point is that the values of R^2 do not change compared to those of the single-variable LR models. The R^2 of LR model $Ma{=}f(LV_{Ma}, LV_{AoA})$ only increases 0.001 than $Ma{=}f(LV_{Ma})$, and $AoA{=}f(LV_{Ma}, LV_{AoA})$ model has the same R^2 as $AoA{=}f(LV_{AoA})$. These two points quantitatively suggest that the two active LVs represent Ma and AoA in a disentangled (or independent) manner.

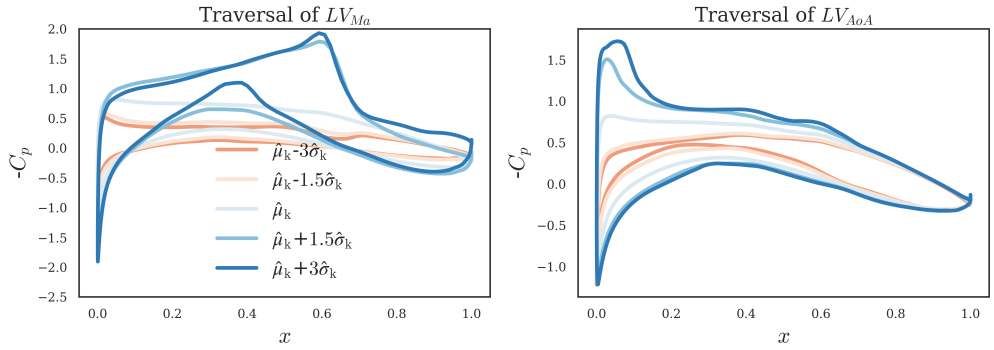
Since the physical meanings of LV_{Ma} and LV_{AoA} are verified both qualitatively and quantitatively, the latent traversal plots of airfoil surface pressure distributions are shown in Fig. 3.15 to check their influence on the flow field more intuitively. In the traversal of LV_{Ma} (Fig. 3.15a), when the LV is in the



(a) (b)

Figure 3.14: The results of the single variable LR: (a) $Ma{=}f(LV_{Ma})$, and (b) $AoA{=}f(LV_{AoA})$.

range of $\hat{\mu}_k - 3\hat{\sigma}_k$ to $\hat{\mu}_k$, there is no shock wave, but as it increases to $\hat{\mu}_k + 1.5\hat{\sigma}_k$ and $\hat{\mu}_k + 3\hat{\sigma}_k$, the occurrence of a shock wave is shown, indicating that LV_{Ma} corresponds to Ma. In the traversal of LV_{AoA} (Fig. 3.15b), a variation in the magnitude of the leading edge suction peak is observed, indicating LV_{AoA} corresponds to AoA. This visual analysis of the actual effects of LV_{Ma} and LV_{AoA} on the pressure distributions also leads to the consistent conclusion that they correspond to Ma and AoA, respectively. Recall that the previous Fig. 3.2 showed the ideal schematic for extracting physics-aware LVs. It is marvelous that 1000-VAE enables this ideal situation exactly where physics-aware LVs correspond to the physical generating factors of the training dataset. Although the ability of β-VAE to extract physics-aware LVs is first observed with simple physical parameters in this study, it has tremendous potential to be utilized to extract generating factors from any dataset in any discipline. The scalability of this framework to sparse and noisy dataset can be verified in Sec. 3.7.2.



(a) (b)

Figure 3.15: Latent traversal plots of airfoil surface pressure distributions in 1000-VAE: (a) traversal of LV_{Ma}, and (b) traversal of LV_{AoA}.

3.5.5 Physics-aware ROM

To summarize the results so far, physics-aware LVs are first extracted by estimating the independence and information intensity of each LVs. These physics-aware LVs are strongly correlated to physical parameters which are the generating factor of training dataset, significantly increasing the interpretablilty. Since LVs act as intermediaries in the prediction process of ROM, their impact on the ROM is bound to be enormous. In this regard, this section further investigates the effect of physics-awareness of LV on ROM.

Fig. 3.16 shows the MSE of the regression models (MSE_{reg}) in each ROM; It is calculated by the difference between true LV and predicted LV by regression model. The reason that the scale of MSE_{reg} is much smaller than that of MSE in Fig. 3.6 is because the dimension of LV is much smaller than that of flow fields. In that 16 regression models are required for 16 LVs, each point represents the MSE of each model, and the symbol o/x indicates whether each LV is physics-aware or physics-unaware. The results show that the MSE_{reg} values of

physics-unaware LVs are considerably higher than those of physics-aware LVs. It is because the correlations between physical parameters and physics-unaware LVs are too trivial to be trained by regression models. Fig. 3.17 supports this; the response surface of the physics-unaware LV (Fig. 3.17b) is much noisier than that of the physics-aware LV (Fig. 3.17a). Another interesting point in Fig. 3.16 is that $\mathrm{MSE}_{\mathrm{reg}}$ values of the physics-aware LVs decrease as β increases. This implies that the tight coupling/correlation between physics-aware LVs and physical parameters is advantageous in terms of regression performance in the ROM process.

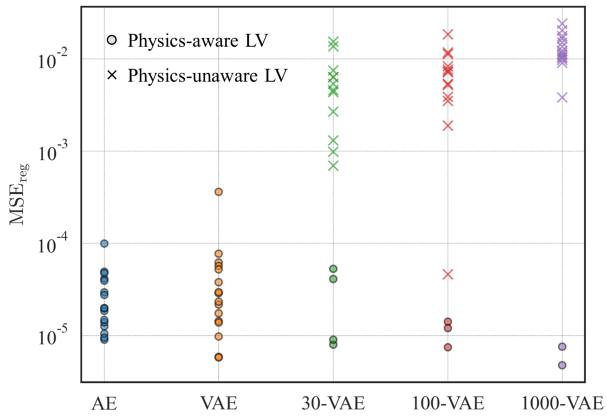
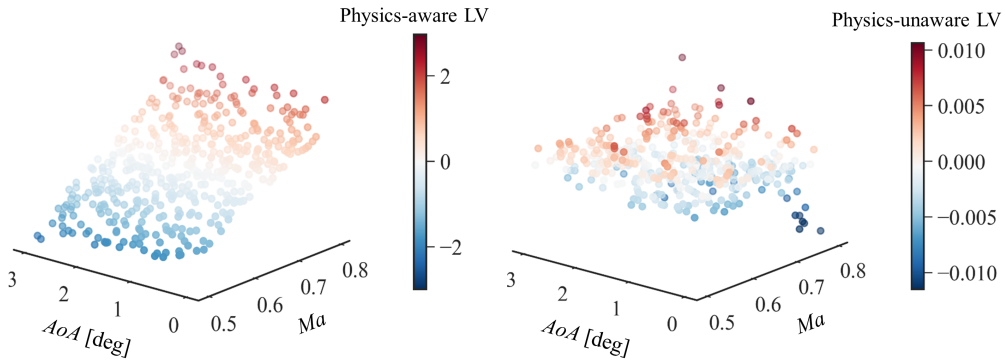


Figure 3.16: MSE of the regression models in ROM.

Then, the MSE between the ground truth flow fields and those predicted by ROM with the exclusion of k^{th} LV is calculated, which is denoted as $\mathrm{MSE}_{\mathrm{pred},-\mathrm{k}}$ where $-\mathrm{k}$ indicates the exclusion of the k^{th} LV; note that the prediction by ROM means obtaining the flow field from unknown input parameters, as already described in Fig. 3.1. Specifically, k^{th} LV is assumed to be constant as $\hat{\mu}_k$, whereas the other LVs are predicted from regression models so that the importance of



(a) (b)

Figure 3.17: Comparison of the response surface of two LVs in the 1000-VAE: (a) physics-aware LV, (b) physics-unaware LV.

k^{th} LV can be estimated. Fig. 3.18 demonstrates their results, where the x-axis indicates KL-divergence ranking of the LVs. An important observation is that the effect of the LV on $MSE_{pred,-k}$ decreases as the LV ranks down. From the fact that the LV ranked higher by KL-divergence has a greater effect on ROM accuracy, it can be concluded that the proposed ranking approach is also valid in terms of ROM performance. Moreover, the physics-unaware LVs (those ranked after 4^{th} in 30-VAE, after 3^{rd} in 100-VAE, and after 2^{nd} in 1000-VAE) have negligible effects on $MSE_{pred,-k}$. This implies that training regression models of physics-unaware LVs are meaningless with respect to the prediction accuracy of ROM; in other words, it is sufficient enough to train regression models of only physics-aware LVs. In this regard, to examine the necessity of each LV in ROM prediction, Fig. 3.19 visualizes the MSE_{pred} values of physics-aware ROM via β-VAE, where all physics-unaware LVs are excluded (e.g., only two LVs are used for ROM via 1000-VAE). For the comparison, those of physics-unaware

ROM via AE are also shown: herein, MSE_{pred} utilizing all 16 LVs or 15 LVs with one LV excluded is presented. Although a considerably small number of LVs are used in physics-aware ROM, the accuracy of them is comparable to that of AE-16LV, physics-unaware ROM. This can also be confirmed by the pressure contour presented in Fig. 3.20. Furthermore, even if only one LV is excluded in AE (AE-15LV), its ROM accuracy becomes equivalent to or even lower than that of 1000-VAE only with two LVs (1000-VAE-2LV). This result highlights the inefficiency of ROM through AE, in that it requires all 16 entangled LVs and therefore requires training 16 regression models. Using physics-aware ROM via 30-VAE, the equivalent accuracy can be achieved with only 4 regression models.

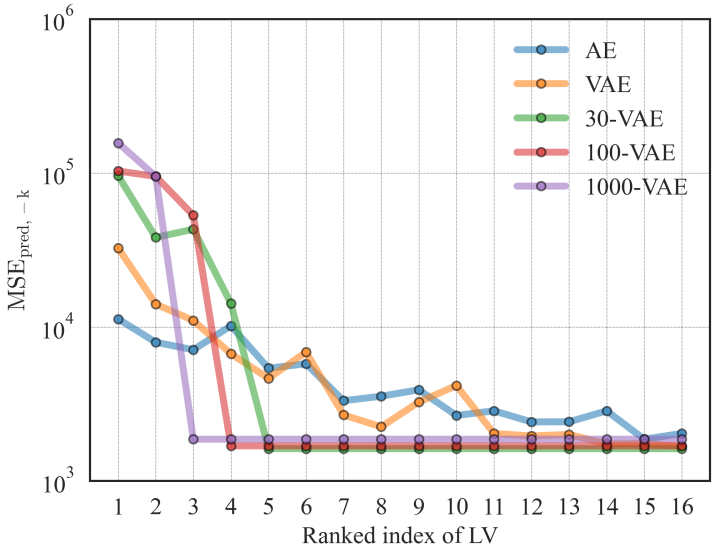


Figure 3.18: MSE of ROM prediction with the exclusion of k^{th} LV.

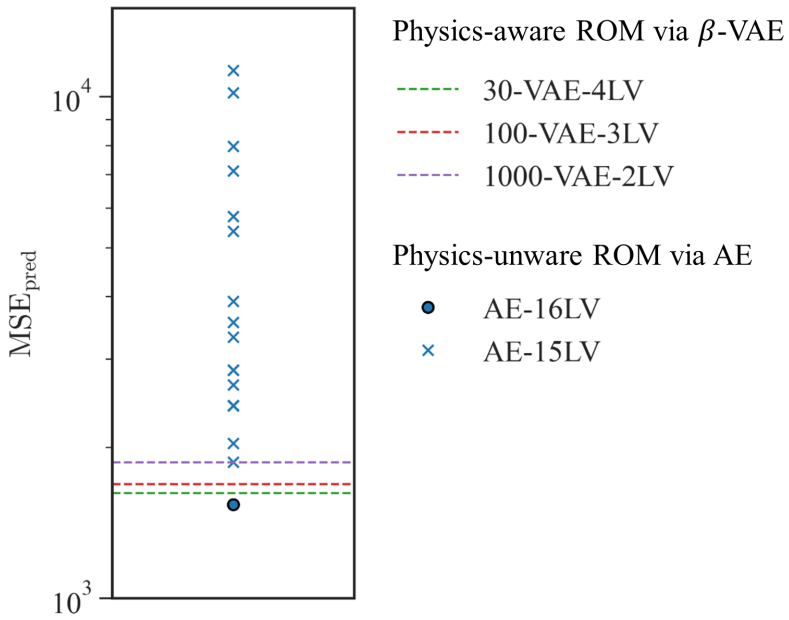
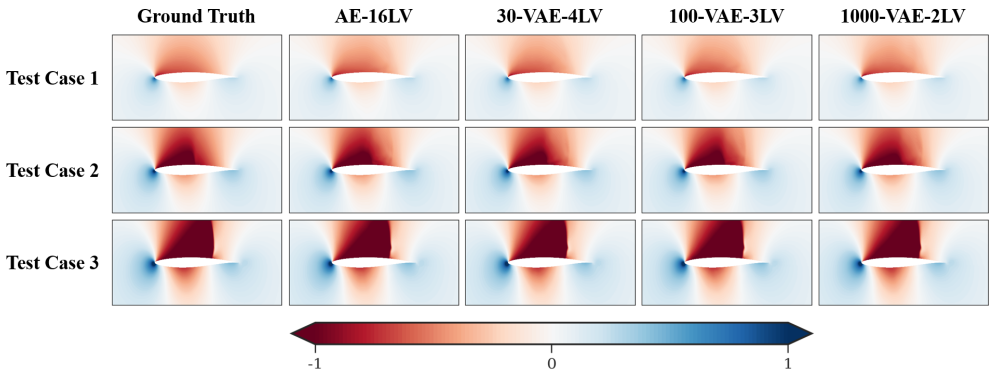
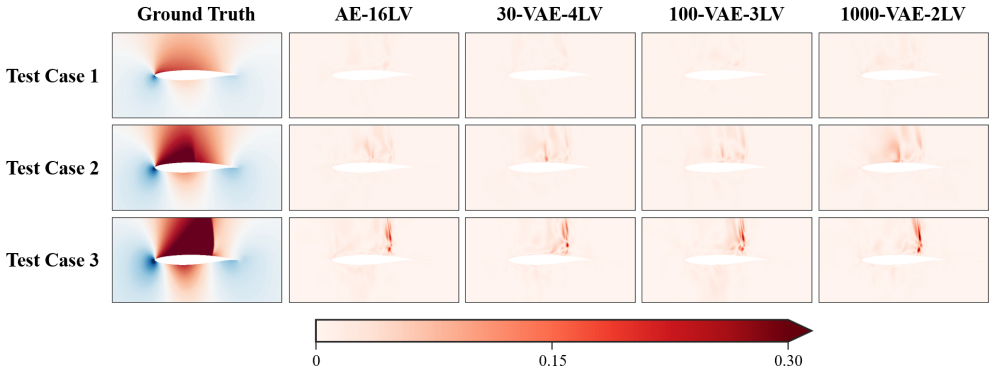


Figure 3.19: Comparison of prediction MSE between physics-aware ROM and physics-unaware ROM.



(a)



(b)

Figure 3.20: Pressure contour predicted from AE/β-VAE-based ROMs: (a) prediction, (b) absolute error.

3.6 Summary

This study proposed the physics-aware ROM based on physics-aware LVs, which are interpretable and information-intensive LVs extracted by β-VAE. The proposed framework is validated against the 2D transonic benchmark problem in the following order: first, the process of extracting physics-aware LVs is scrutinized by quantitatively estimating their independence and information intensity. Then, the actual physical meanings of these LVs are thoroughly investigated. Finally, the effectiveness of the proposed physics-aware ROM compared to conventional ROMs is verified. The key contributions of our study can be summarized as follows:

1. The impacts of hyperparameter β on the independence of LVs were scrutinized, and its effect on the independence of LVs was practically confirmed in that LVs become disentangled from each other as β increases.

2. KL-divergence ranking method was proposed to measure the information intensity of each LV. This approach has two following advantages over the previous ranking method: KL-divergence is the direct cause of the discrepancies in their information intensity and it does not require cumbersome post-processing of reconstructed data. The proposed criterion was confirmed to have a consistent trend with estimated standard deviations and Sobol indices, indicating their validity. Through this ranking method, the effect of β on the latent space regularization was practically confirmed in that LVs become information-intensive as β increases.

3. The physical meanings contained in physics-aware LVs were thoroughly investigated. The correlation between the physical generating factors of the training dataset and the information physics-aware LVs contain was

scrutinized as β varies. Finally, it was confirmed quantitatively and qualitatively that the extracted physics-aware LVs in 1000-VAE actually correspond to the generating factors, which were Ma and AoA in this study. To the best of the authors' knowledge, this is the first observation of physics-aware LVs in the fluid dynamics discipline.

4. The effects of physics-awareness of LVs on the accuracy of regression and prediction processes in ROM were analyzed and it was confirmed that only physics-aware LVs had a significant effect on their accuracy. Therefore, physics-aware ROM, which utilizes only physics-aware LVs is proposed for its efficiency in that the number of required regression models can be reduced significantly. Finally, compared to the conventional ROMs, its validity and efficiency were successfully verified.

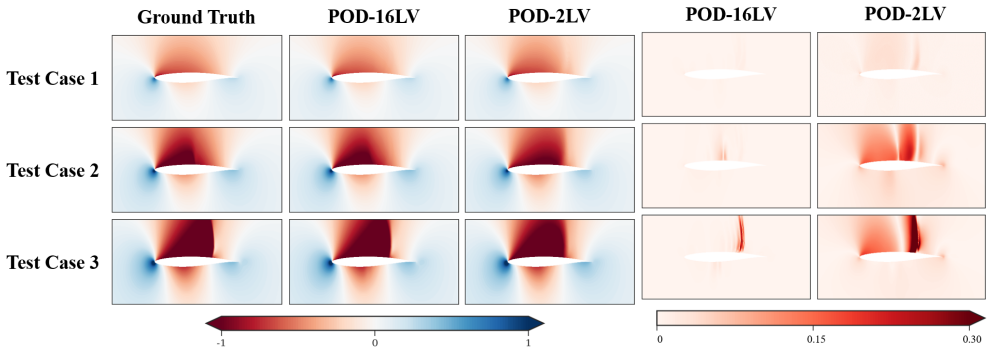
The presented data-driven physics-aware ROM has great potential in two engineering applications. First, the extraction process of physics-aware LVs can be applied to discover generating factors from the given dataset. For example, this application can be extended to identify fault-causing factors in sensor data from manufacturing processes. Second, physics-aware ROM can be an efficient alternative to conventional black-box ROMs in that it utilizes necessary regression models of only physically interpretable and information-intensive LVs, rather than redundant regression models of all uninterpretable LVs. Though the application of this framework was demonstrated via 2D transonic benchmark problem, it can be easily applied and extended to numerous engineering disciplines since no special assumptions have been made on this specific problem. For future work, a more comprehensive investigation on physics-aware LVs will be conducted, such as their scalability to the temporal dataset or their ability to discern redundant physical parameters.

3.7 Additional results

3.7.1 POD results

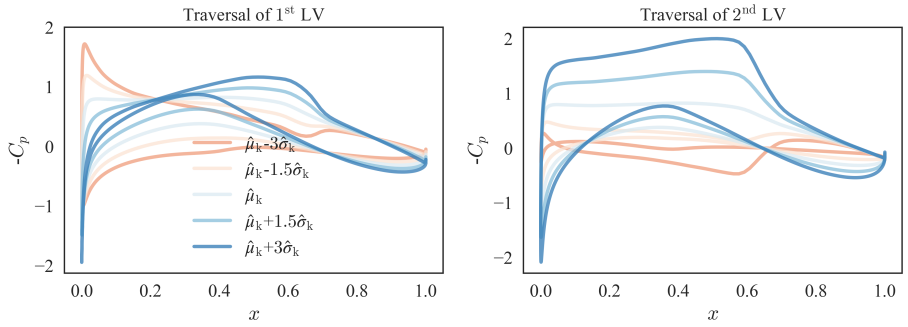
This study mainly focused on the nonlinear-based DR methods (AE/VAE/β-VAE). In this section, additional results by POD, which is the gold standard of the linear-based DR approach, are presented. Fig. 3.21 shows the pressure contour predicted by POD-based ROM and its absolute error contour. Herein, POD-16LV and POD-2LV each represent ROM with 16 and 2 LVs (or modes) of POD. When compared to Fig. 3.20, the error contour of POD-16LV is comparable to that of AE/VAE/β-VAE, whereas POD-2LV is much more worse. Indeed, $\mathrm{MSE}_{\mathrm{pred}}$ of ROM based on POD-2LV (4.94×10^4) is 26 times higher than that of 1000-VAE-2LV (1.87×10^3), while POD-16LV (2.07×10^3) is only 1.1 times higher than 1000-VAE-2LV. It is noteworthy that the ROM accuracy based on only 2 LVs (1000-VAE-2LV) is higher than that based on 16 LVs (POD-16LV).

Fig. 3.22 shows the same results as in Fig. 3.15 except it is based on POD. Due to the poor ROM accuracy of POD-2LV mentioned above, it is shown that the physical discontinuity (shock wave in this case) cannot be accurately captured. Furthermore, when compared to 1000-VAE in Fig. 3.15, the two dominant LVs of POD encode the physical characteristics (shock wave and leading edge suction peak) in an entangled manner; the traversals of them are not physically interpretable, unlike 1000-VAE. It indicates that despite its orthogonality, the physical interpretability of its latent space cannot be guaranteed. This section shows the obvious superiority of nonlinear-based DR methods over linear-based DR method (POD) in terms of reconstruction accuracy of ROM and physical interpretability of latent space, both of which are the main interests of this study.



(a) Predicted pressure contour (b) Absolute error contour

Figure 3.21: Pressure contour predicted from POD-based ROM: (a) prediction, (b) absolute error.



(a) (b)

Figure 3.22: Latent traversal plots of airfoil surface pressure distributions in POD: (a) traversal of 1st LV, and (b) traversal of 2nd LV.

3.7.2 Scalability of extracting physics-aware LVs in practical problem

In the main text, the framework for extracting physics-aware LVs using β-VAE is validated with regularized and well-organized CFD dataset. However, most of the data in real-world engineering is sparse and noisy. Accordingly, in this section, the practical scalability of the proposed method is verified by utilizing only a small portion of the training dataset with artificial noises. For this purpose, the training dataset in the main text (which is the tensor of $3 \times 512 \times 128$) is preprocessed to be a vector with only 35 elements: 32 surface pressure values, lift coefficient (C_l), drag coefficient (C_d), and pitching moment coefficient (C_m). Artificial Gaussian noises are added considering the scale of each element, and the final dataset is shown in Fig. 3.23. Then, AE and β-VAE models ($\beta \in [0.01, 0.1, 1]$) are trained and their top two ranked LVs are visualized in Fig. 3.24 (which corresponds to Fig. 3.13 in Sec. 3.5.4). Again, it can be seen that LVs of AE are highly entangled. On the other hand, LVs of β-VAE become more and more disentangled as β increases so that eventually in 1-VAE, each LV solely represents Ma and AoA, respectively. Accordingly, it can be concluded that the extraction of physics-aware LVs via β-VAE, which is first observed and reported in this study, has the potential to be applied to real-world engineering problems where training datasets are sparse and noisy.

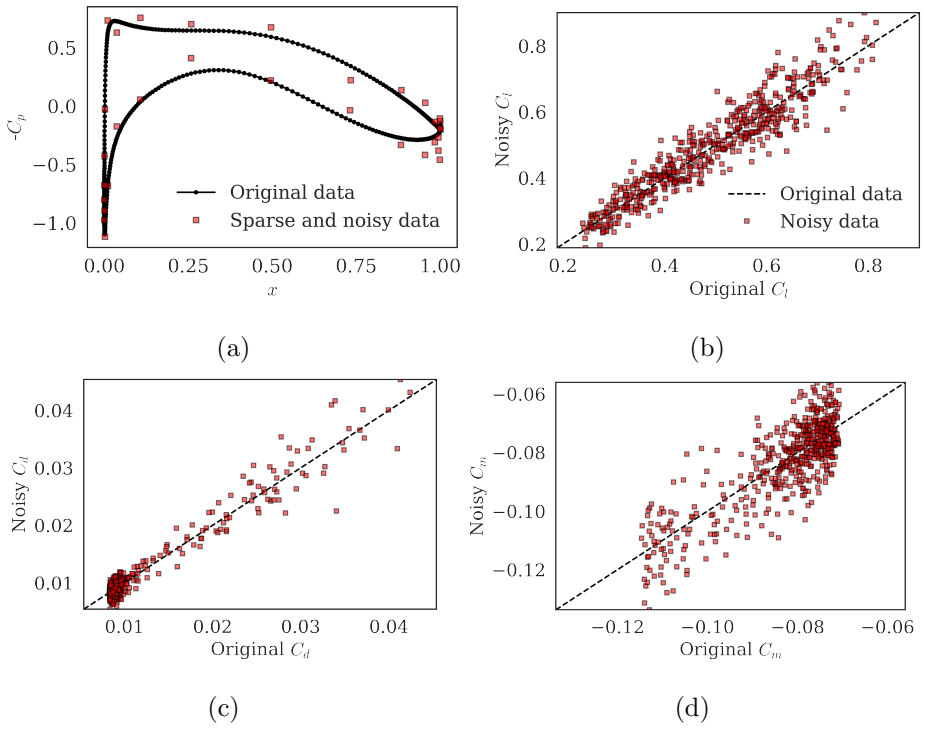


Figure 3.23: Preprocessed training dataset consisting of (a) 32 surface pressure values, (b) C_l, (c) C_d, and (d) C_m.

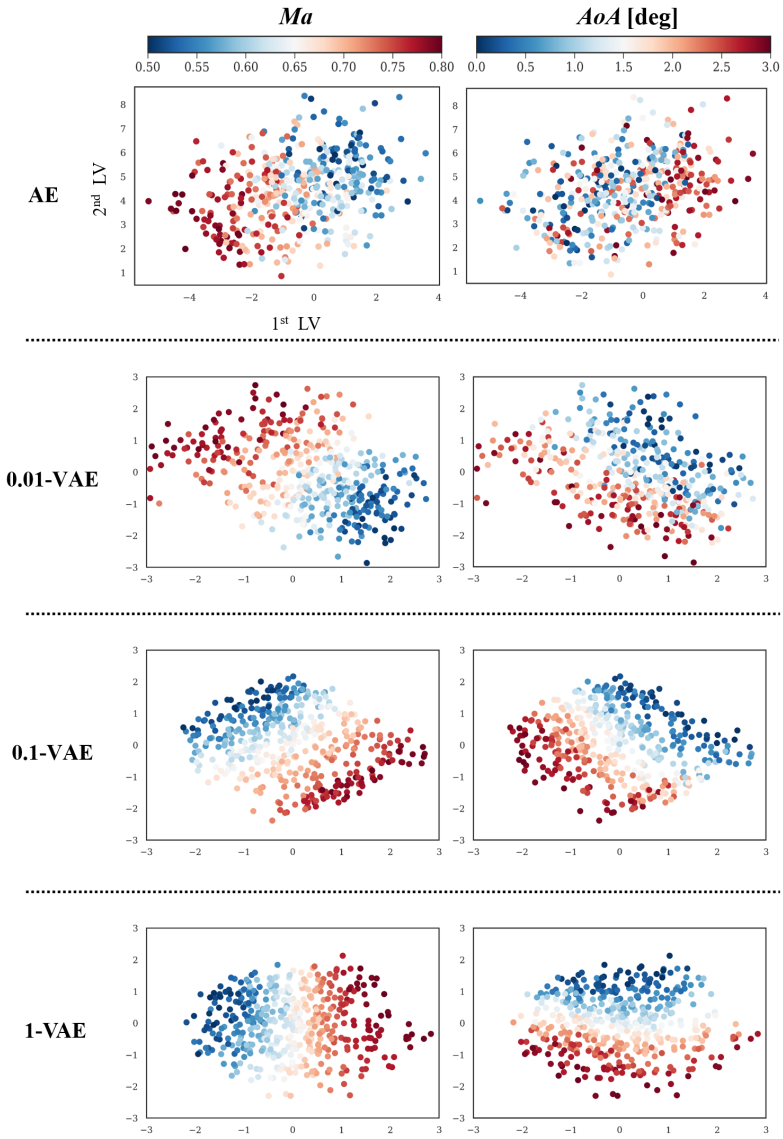


Figure 3.24: Investigation of physical features contained in the top two LVs for sparse and noisy datasets.

Chapter 4

Reliable and efficient uncertainty quantification

The work in this chapter is currently under review [132].

4.1 Introduction

This chapter aims to achieve a reliable and efficient UQ, which is crucial for the application of regression models to aerodynamic design, by using a DE approach that is capable of all of the following: universal approximation capability, scalability to large datasets, and multi-output regression. Specifically, this chapter aims to comprehensively investigate the DE model in the multi-output regression task, which is the most common problem in practical engineering disciplines, to predict the aerodynamic performance of a missile configuration. The most popular regression model capable of UQ in engineering fields but not scalable to large datasets, GPR, is also compared with DE. Not only the validation is performed, but since the poor reliability of the quantified uncertainty by DE is observed, a simple post-hoc calibration method is applied to

DE models to correct the unsatisfactory uncertainty quality. Finally, the impact of the proposed calibration method on Bayesian optimization is examined, verifying the fact that whether the DE is calibrated or not can result in unintended exploration characteristics when extended to Bayesian optimization. The remainder of this chapter highlights why the reliable UQ of the regression model is inevitable in the aerodynamic design process and how this dissertation extensively investigates the performance of the DE approach for this purpose. Finally, the main contributions of this chapter are summarized.

We are entering an era of high-performance computing technologies and they have enabled engineers to efficiently obtain vast amounts of data, so-called big data. Accordingly, numerous data-driven approaches have been studied to derive physical insights from the growing number of available datasets. The most popular but most fundamental one is to utilize a given dataset to train a regression model (also referred to as a surrogate model), which is used to predict quantities of interest (QoIs) [16, 17, 18]. This straightforward approach can be leveraged for a variety of applications, from exploration during the design optimization process [19] to the prediction of high-dimensional data via reduced-order modeling [20]. Furthermore, from the perspective that the regression model can accelerate the realization of digital twins by replacing the high-demand simulations required within its procedure [26], its potential seems boundless.

However, such impacts cannot be fully achieved by the regression model alone. In real-world engineering problems, *knowing what it does not know and therefore improving interpretability* is an indispensable issue. In the decision-making process based on the regression model, engineers should consider the predictive uncertainty derived from insufficient train data and imperfect regression model [133]. Otherwise, blind faith in regression models, especially during

risk assessment and management procedures, can lead to unexpected and therefore disastrous outcomes. The most common approach to deal with this issue is to perform Bayesian optimization, also known as efficient global optimization in engineering fields [134, 19, 31, 35]. Briefly, it aims to reduce model uncertainty by iteratively updating the model based on the acquisition function [135, 136, 137], which contains uncertainty information (Fig. 4.1). Since the Bayesian optimization process requires uncertainty quantification (UQ), whether the model quantifies the uncertainty over its prediction is the key consideration for engineers in determining which regression model to utilize.

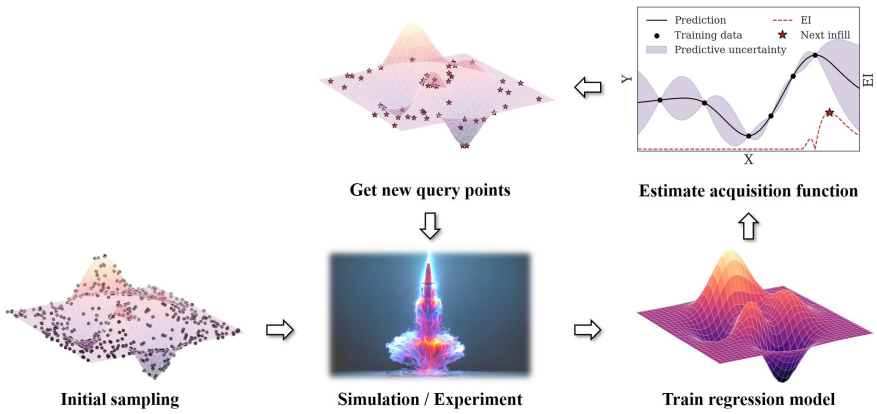


Figure 4.1: Flowchart of Bayesian optimization.

Gaussian process regression (GPR)—also known as Kriging—is one of the most widely used regression models capable of UQ in various engineering fields [138, 139, 140, 141, 142, 143, 144, 130, 145, 146, 147, 148, 149]. GPR allows engineers to identify which predictions are unreliable by providing predictive uncertainty, and it has become the most prevalent regression model for Bayesian optimization [16, 19, 150, 137]. However, GPR is notorious for its time com-

plexity of $O(n^3)$ and memory complexity of $O(n^2)$, where n denotes the dataset size [46, 47]. Even in multi-output regression tasks, since a GPR is trained for each output independently, the required training time increases linearly with respect to the output dimension, and the correlations within outputs become completely ignored [39].

In this regard, Bayesian neural networks (BNNs) [151, 152, 153] can be effective alternatives for the following reasons: 1) their universal approximation capability [58, 65]; 2) scalability to large datasets due to mini-batch training [66]; and 3) multi-output prediction only with a single regression model. Since BNNs aim to learn the probability distributions of the model parameters on the basis of Bayesian inference, they can estimate the uncertainty of their prediction, whereas traditional neural networks (NNs) only provide point estimates. However, their additional model parameters lead to slower convergence during the training [51] and require significant modifications to the conventional framework of NNs, leading to cumbersome and knotty training algorithms [48, 49, 50]. Such computational complexity and inefficiency prevent BNNs from being a viable option for engineers who prioritize practicality and are not familiar with Bayesian formalism.

Recently, easy-to-use but scalable approaches for approximating Bayesian inference have attracted the attention of engineers. Especially, deep ensembles (DE) [49] and MC-dropout [154, 155] require only a few modifications to standard (or vanilla) NNs, demonstrating their applicability to the fields of engineering. However, since MC-dropout has controversial issues about whether or not it is Bayesian inference [156, 157, 158, 159], it is out of our focus; see 4.6.1. DE, an approach to quantify the predictive uncertainty by leveraging ensembles of NNs, was first proposed by Lakshminarayanan et al. [49]. Their idea is so "simple and straightforward" that it only requires training multiple

NNs in parallel on the same training dataset. Despite its simplicity, several researchers have recognized that the DE provides not only accurate predictions, but also robust, reliable, and practically useful uncertainty on a wide variety of architectures and datasets, even on out-of-distribution (OOD) examples [160, 161, 162, 163]. Finally, it has come to be treated as the "gold standard for accurate and well-calibrated predictive distributions" [164].

However, most previous studies have focused on verifying whether DE accurately estimates the uncertainty in classification tasks [49, 165, 166, 161, 162, 163]. Its comprehensive validation has not been conducted in multi-output regression tasks, which are the most common problems in practical engineering disciplines. For example, de Becdelievre and Kroo [167] and Pawar et al. [168] utilized DE for tailless aircraft range optimization and boundary layer flow prediction tasks, respectively, without any validation of the estimated uncertainty in their problems. In this sense, our research focuses on a thorough validation of the DE approach in multi-output regression tasks, while comparing it with GPR, both in terms of regression accuracy and reliability of the estimated uncertainty. Especially, we seek to overcome the limitations of existing studies that blindly adopted the number of NNs used in DE without sufficient explanation of their effects [169, 162, 170, 167, 171, 165, 167, 172, 173]. Finally, a tendency of the quantified uncertainty to become underconfident with the number of NNs is observed and a practical calibration method is proposed to be applied. The corresponding effects are verified quantitatively with two uncertainty evaluation criteria, and their potential impact on Bayesian optimization is briefly investigated. The main contributions of this work can be summarized as follows:

1. First attempt to validate DE approach in the multi-output regression task.

2. The effect of the number of NNs used for DE is comprehensively investigated and two different criteria are utilized for rigorous validation of its uncertainty quality.

3. Accordingly, an increasing trend of underconfidence with the increasing number of NNs is first empirically observed in the regression task, and its analytical explanation is derived.

4. A simple post-hoc calibration method is applied to DE models for the correction of unsatisfactory uncertainty quality and its effectiveness is verified both qualitatively and quantitatively.

5. The potential impact of the proposed calibration method on Bayesian optimization is briefly examined: the possibility that different estimates of uncertainty could lead to different exploration behavior is examined.

6. Throughout the above procedures, GPR—the most well-known UQ model—is compared with DE, and the effectiveness of DE over GPR is confirmed.

The rest of this chapter is organized as follows. In Sec. 4.2, the background on how to implement DE and evaluate its uncertainty quality is described. In Sec. 4.3, the application of DE to a multi-output regression task in aerospace engineering is elaborated. It provides a thorough validation of DE models compared to GPR models, both in terms of prediction accuracy and uncertainty quality. In Sec. 4.4, a simple post-hoc calibration method is applied and its effects on uncertainty quality and Bayesian optimization are investigated. Finally, in Sec. 4.5, the conclusion and future work of this study are presented.

4.2 Implementation and evaluation of DE

DE was first proposed by Lakshminarayanan et al. [49] for the simple and scalable estimation of predictive uncertainty. Although its idea can be seen as a straightforward extension of NNs (making use of multiple NNs), DE has received little attention in the engineering disciplines, in contrast to its reputation in computer science. This is due to the lack of previous works explaining its algorithm friendly and comprehensively, and therefore the purpose of this section is to fill the academic gap by elaborating on the DE methodology and its validation. First, we introduce the background of how to implement DE in Sec. 4.2.1 and then how to evaluate its uncertainty quality is described in Sec. 4.2.2.

4.2.1 Deep ensembles (DE)

The NNs discussed in Sec. 2.2.1 are often considered "overconfident" because they do not provide any measure of uncertainty. For those who are interested in UQ, DE can be an alternative approach. DE is based on an ensemble of NNs, but there is a key distinction: unlike a standard NN, which only outputs QoIs as $\mu(x)$, the NN used for DE outputs them as a Gaussian distribution, $N\big(\mu(x), \sigma^2(x)\big)$. That is, it assumes that QoIs are sampled from $N\big(\mu(x), \sigma^2(x)\big)$ and aims to provide information about this distribution by outputting $\mu(x)$ and $\sigma^2(x)$. Here, $\mu(x)$ refers to the estimated/predicted value and $\sigma^2(x)$ refers to the estimated/predicted variance. It should be noted that the estimated variance $\sigma^2(x)$ indicates the aleatory uncertainty (uncertainty arising from noise inherent in the training data) regarding the estimated value $\mu(x)$ [174, 175]. With this specific NN architecture, the number of final nodes is doubled since it outputs not only the standard outputs, $\mu(x)$, but also the uncertainty about them, $\sigma^2(x)$. Due to the probabilistic distribution it provides, this type of NN is referred to

as a probabilistic NN.

The probabilistic NN architecture is adopted in the DE model since the vanilla NN structure cannot apply the proper scoring rule, which is the criterion for estimating the quality of predictive uncertainty [176]. Lakshminarayanan et al. [49] emphasized that with the vanilla NN architecture, which provides only the estimated value $\mu(x)$, the mean squared error (MSE) would be used as the loss function:

$$\text{MSE} = \big(y - \mu(x)\big)^2 \tag{4.1}$$

and therefore the information about the predictive uncertainty is entirely disregarded during the training. To address this issue, they proposed utilizing a probabilistic NN that can output both $\mu(x)$ and $\sigma^2(x)$. It allows the use of the proper scoring rule, negative log-likelihood (NLL), which is the standard metric for assessing the quality of probabilistic models [177]:

$$\text{NLL}(\mu(\text{x}), \sigma^2(\text{x}), \text{y}) = -\log\big(p_\theta(y|x)\big) = \frac{\log\sigma^2(x)}{2} + \frac{\big(y - \mu(x)\big)^2}{2\sigma^2(x)} + \frac{\log 2\pi}{2} \tag{4.2}$$

This NLL allows the intuitive interpretations as follows [178, 179]. 1) When some training points have high MSE, $\big(y - \mu(x)\big)^2$, the impact of the term $\frac{\big(y-\mu(x)\big)^2}{2\sigma^2(x)}$ is relatively significant compared to $\frac{\log\sigma^2(x)}{2}$. Therefore, the model is trained to output high denominator value, $\sigma^2(x)$, at the corresponding points to reduce the NLL. 2) At training points with low MSE, the term $\frac{\log\sigma^2(x)}{2}$ becomes relatively dominant and thus the model is encouraged to output low $\sigma^2(x)$ at those points. In summary, the NLL scoring rule-based training algorithm for the probabilistic NN facilitates the learning of reliable predictive uncertainty by estimating high uncertainty where prediction error is high and low uncertainty where prediction error is low. It should be noted that this cannot be accomplished in vanilla NN with MSE loss function.

However, using a single probabilistic NN is limited to estimating the aleatory uncertainty. To estimate the epistemic uncertainty arising from the model parameters due to insufficient training data, a further step is required. Lakshminarayanan et al. [49] suggested the use of multiple probabilistic NNs, called deep ensembles (DE), to quantify both aleatory and epistemic uncertainties. Specifically, they aimed to capture the epistemic uncertainty by using the multiple probabilistic NNs trained on the identical dataset (also identical architectures for NNs are used). The overall training procedure is summarized in Algorithm 2. There are two notable points herein: 1) the random initialization of the model parameters of the NNs in line 2; and 2) the random shuffling of the training dataset due to mini-batches in line 5. These two factors are regarded as the main causes of the individual NN with identical architecture in the ensemble being able to be trained with enough diversity [49]. See Fort et al. [161] for further information, which examined the effects of random initialization and random shuffling.

To see how the ensemble of probabilistic NNs trained in Algorithm 2 estimates two types of uncertainty, let $\mu_i(x)$ and $\sigma_i^2(x)$ be the predictive mean and predictive variance output by the ith individual NN. Herein, the predicted probabilities of y from the ith NN can be expressed as $N\big(\mu_i(x), \sigma_i^2(x)\big)$, indicating that there are multiple Gaussian distributions according to each NN in the ensemble. Lakshminarayanan et al. [49] suggested approximating the final probability of the output as a mixture of Gaussian probabilities as follows:

$$\hat{\mu} = \frac{1}{M} \sum_{i=1}^{M} \mu_i, \tag{4.3}$$

Algorithm 2 Training procedure of DE

Split the train dataset X (with input x and output y) into J mini-batches.
Randomly initializes model parameters of the M probabilistic NNs and set training epochs.
for $i = 1 : M$ **do** ▷ Loop for NN (parallelizable)
    for epochs **do** ▷ Loop for epoch
        for $j = 1 : J$ **do** ▷ Loop for mini-batch
            $\mu_{ij}, \sigma_{ij}^2 = \mathrm{NN}_i(x_{ij})$ ▷ Feed-forward with mini-batch x_j
            $\mathcal{L}_{ij} = \mathrm{NLL}(\mu_{ij}, \sigma_{ij}^2, y_{ij})$ ▷ Calculate NLL
            $\theta_i = \theta_i - learning\,rate * \delta\mathcal{L}_{ij}/\delta\theta$ ▷ Update model NN_i
        end for
    end for
end for

$$\begin{aligned} \underbrace{\hat{\sigma}^2}_{\substack{\text{predictive} \\ \text{uncertainty}}} &= \frac{1}{M}\sum_{i=1}^{M}\sigma_i^2 + (\frac{1}{M}\sum_{i=1}^{M}\mu_i^2 - \hat{\mu}^2) \\ &= \underbrace{E(\sigma_i)}_{\substack{\text{aleatory} \\ \text{uncertainty}}} + \underbrace{Var(\mu_i)}_{\substack{\text{epistemic} \\ \text{uncertainty}}} \end{aligned} \tag{4.4}$$

where M is the number of probabilistic NNs used for the ensemble. Accordingly, the final predictive value of DE is $\hat{\mu}$ and the final predictive uncertainty is $\hat{\sigma}^2$. As in Eq. 4.4, the predictive uncertainty can be decomposed into aleatory and epistemic uncertainty; see Scalia et al. [180] and Hu et al. [181] for more details. It should be noted that no additional training algorithm is required after the training of probabilistic NNs in Algorithm 2: only the mixture process of already trained NNs in Eq. 4.3 and Eq. 4.4 is required. The overall flowchart of DE from the training of probabilistic NNs to the final prediction is schematically shown

in Fig. 4.2.

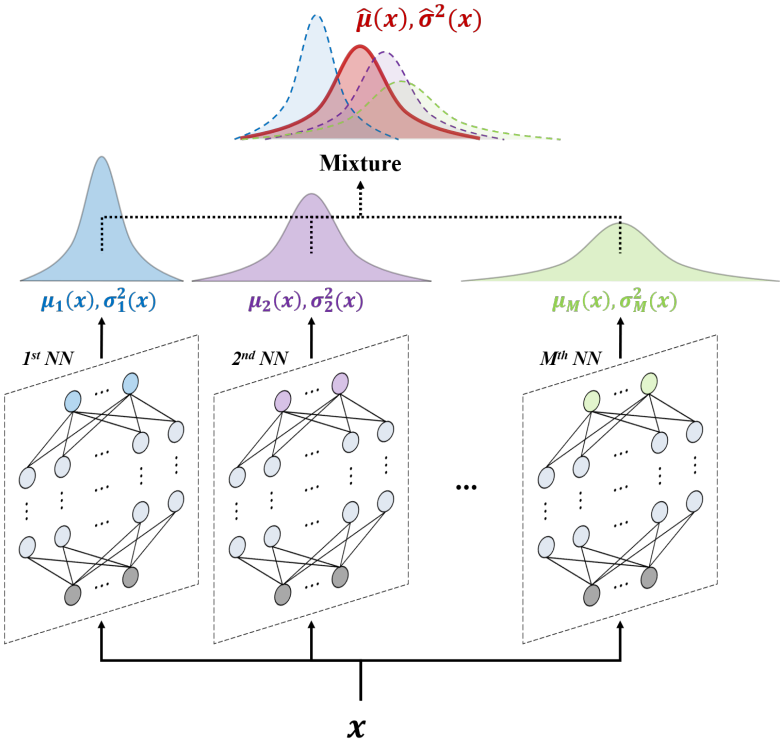


Figure 4.2: Flowchart of DE approach.

4.2.2 Uncertainty quality evaluation

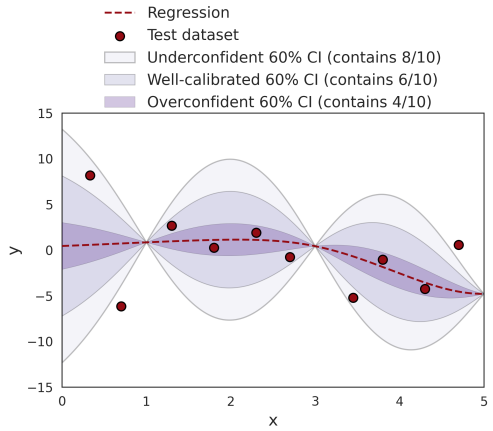
In the previous Sec. 4.2.1, we explored the ability of the DE technique to determine predictive uncertainty. However, engineers who are interested in predicting uncertainty require more than just the feasibility of UQ; they also require confidence in the reliability of the estimated uncertainty. Unfortunately, previous studies that employed GPR to evaluate predictive uncertainty in engineering fields have disregarded this point. Consequently, the purpose of this section is to address this gap by presenting two criteria for assessing the accu-

racy/reliability of estimated predictive uncertainty. These techniques are applicable to any regression model performing UQ, such as GPR and DE.

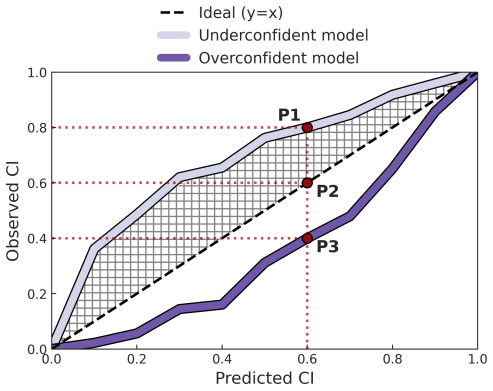
4.2.2.1 AUCE

The most widely used metric to evaluate the reliability of uncertainty is the area under the calibration error curve (AUCE) [182, 160]. The primary goal of this measure is to ensure that the confidence intervals (CI) estimated by the model are accurate in practice. The concept of AUCE is shown schematically in Fig. 4.3. In Fig. 4.3a, the CI labeled "Well-calibrated 60% CI" contains 60% of the test dataset (6 out of 10 points), where test dataset indicates the dataset used to verify the quality of the estimated uncertainty. Thus, a well-calibrated model would have a 60% CI that actually contains 60% of the test data. On the other hand, if the 60% CI contains more than 60% of the dataset (8 out of 10 points), the model is considered underconfident, which corresponds to the case of "Underconfident 60% CI." This means that the model is not confident enough about its prediction and overestimates its CI. Conversely, if the 60% CI contains less than 60% of the dataset (4 out of 10 points, "Overconfident 60% CI" case), the model is considered overconfident, meaning that it is too confident in its prediction and thus estimates a narrower CI than it actually should.

The difference between the CI estimated by the model and the actual data it contains can be assessed visually by the CI-based reliability plot shown in Fig. 4.3b. This plot compares the predicted CI from the model on the x-axis with the observed CI measured with the test dataset on the y-axis. To clarify, consider the situation depicted in Fig. 4.3a. In the underconfident case, which corresponds to point P1 (x=0.6, y=0.8), the predicted 60% CI actually corresponds to the observed 80% CI because 8 out of 10 points are included. Point P2 represents



(a)



(b)

Figure 4.3: (a) Illustration of well-calibrated/miscalibrated models: 60% CI of the well-calibrated model contains 60% of the test data, whereas that of the underconfident and overconfident model contains 80% and 40% of the data, respectively. (b) Illustration of CI-based reliability plot.

the well-calibrated case, where the predicted 60% CI by the model matches the actual 60% of data contained in the CI. In contrast, point P3 represents the overconfident case, where the model predicts a 60% CI that actually contains

only 40% of the data. In this context, the line $y = x$ represents an ideally well-calibrated model where the predicted CI perfectly matches the observed CI. The algorithm for the CI-based reliability plot is summarized in Algorithm 3.

Algorithm 3 Procedure for CI-based reliability plot

Prepare the test dataset X (with input x and output y).
Define candidates of CI to be investigated: $P = \{p_1, p_2, ..., p_K\}$.
$D = \varnothing$ ▷ Initialize dataset D to be plotted as y-axis
for $i = 1 : K$ **do** ▷ Loop for P
    $count = 0$ ▷ Initialize *count*
    Find $Q(\frac{p_i+1}{2}|\mu, \sigma^2)$, which is $\frac{p_i+1}{2}$ quantile of $N(\mu, \sigma^2)$.
    for $j = 1 : length(X)$ **do** ▷ Loop for X
        if $-Q\big(\frac{p_i+1}{2}|\mu(x_j), \sigma^2(x_j)\big) \leq y_j \leq Q\big(\frac{p_i+1}{2}|\mu(x_j), \sigma^2(x_j)\big)$ **then**
            $count+ = 1$
                ▷ Increase *count* if test data is within the estimated CI
        end if
    end for
    $\hat{p} = count/length(X)$ ▷ Calculate observed CI
    $D = D \cup \hat{p}$ ▷ Append $\hat{p}$ to D
end for
Plot CI-based reliability plot: x-axis with P and y-axis with D.

By utilizing this CI-based reliability plot, the AUCE, which is a metric that evaluates the quality of the estimated uncertainty, can be derived. In detail, it is calculated as the area between the ideal line $y = x$ and the reliability plot of the model. The hatched area in Fig. 4.3b corresponds to the AUCE of the underconfident model, and the mathematical expression for the AUCE is provided in the following equation [160]:

$$\text{AUCE} = \frac{1}{K}\sum_{i=1}^{K}|\hat{p} - p_i| \tag{4.5}$$

where K refers to the number of CI candidates as in Algorithm 3. By definition, a low AUCE value implies that the predictive uncertainty quantified by the model is reliable (or well-calibrated). Additional information on AUCE can be found in Naeini et al. [183], Gustafsson et al. [160], and Scalia et al. [180].

4.2.2.2 ENCE

Despite its reputation as a metric of uncertainty quality, AUCE has a critical shortcoming in that it only considers the average over the entire test dataset rather than individuals as mentioned by Levi et al. [184]. Moreover, they analytically and empirically elaborated that AUCE can be zero even when the predicted distribution is statistically independent from that of the ground truth. In this context, they proposed a novel approach to evaluate the quality of uncertainty, the expected normalized calibration error (ENCE). It was first proposed based on the intuitive assumption: for the well-calibrated model, the estimated uncertainty $\sigma^2(x)$ will be equal to $\big(y - \mu(x)\big)^2$, MSE. This condition can be expressed mathematically as follows, implying that a higher estimated variance should correspond to a higher expected MSE [185]:

$$\mathbb{E}_{x,y}[\big(y - \mu(x)\big)^2|\sigma^2(x)] = \sigma^2(x) \tag{4.6}$$

The above Eq. 4.6 indicates that the ideally (perfectly) well-calibrated model will have an expected error exactly equal to predictive uncertainty. In this sense, whether the model is well-calibrated can be visually inspected using the error-based reliability plot [180, 184]: x-axis as root mean squared error (RMSE), $y - \mu(x)$, and y-axis as root of the mean variance (RMV), $\sigma(x)$. Fig. 4.4 illustrates it, and by its definition in Eq. 4.6, $y = x$ line indicates the ideally

calibrated model. The procedure for its plotting is summarized in Algorithm 4.

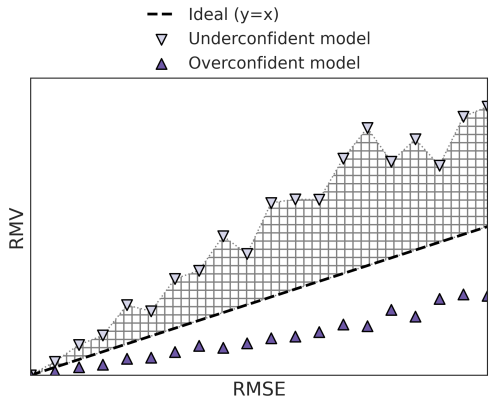


Figure 4.4: Illustration of error-based reliability plot. Underconfident model overestimates RMV relative to RMSE, while overconfident model underestimates RMV. The ideal model estimates the equivalent RMV and RMSE as the $y = x$ black dashed line.

Then, the area between the ideal $y = x$ line and the error-based reliability plot can be calculated. The normalized version of this value refers to ENCE, the second uncertainty quality metric, and is as follows:

$$\text{ENCE} = \frac{1}{B} \sum_{i=1}^{B} \frac{|\text{RMV}(i) - \text{RMSE}(i)|}{\text{RMV}(i)} \tag{4.7}$$

where B indicates the number of bins in Algorithm 4. Therefore, the ENCE of the underconfident model in Fig. 4.4 can be calculated as the hatched area divided by RMV. As with AUCE, the lower the ENCE value, the better the model is calibrated.

4.2.3 Uncertainty calibration: STD scaling

In situations where the estimated uncertainty from the model is imprecise in terms of AUCE (refer to Sec. 4.2.2.1) and ENCE (refer to Sec. 4.2.2.2), there

Algorithm 4 Procedure for error-based reliability plot

1: Prepare the test dataset X (with input x and output y).
2: Sort X according to y values.
3: Define the number of bins: B (assume B divides $length(X)$).
4: Divide sorted X into B bins, $\tilde{X} = \{\tilde{X}_1, \tilde{X}_2, ..., \tilde{X}_B\}$, such that each $\tilde{X}_i$ has the same size of $length(X)/B$.
5: $D_{RMSE} = \varnothing$ ▷ Initialize dataset D_{RMSE} to be plotted as x-axis
6: $D_{RMV} = \varnothing$ ▷ Initialize dataset D_{RMV} to be plotted as y-axis
7: **for** $i = 1 : B$ **do** ▷ Loop for $\tilde{X}$
8: $D_{RMSE} = D_{RMSE} \cup \sqrt{\frac{1}{|\tilde{X}_i|} \sum_{x \in \tilde{X}_i} \big(y(x) - \mu(x)\big)^2}$
9: ▷ Append RMSE to D_{RMSE}
10: $D_{RMV} = D_{RMV} \cup \sqrt{\frac{1}{|\tilde{X}_i|} \sum_{x \in \tilde{X}_i} \sigma^2(x)}$
11: ▷ Append RMV to D_{RMV}
12: **end for**
13: Plot error-based reliability plot: x-axis with D_{RMSE} and y-axis with D_{RMV}.

are various techniques for calibrating uncertainty. Some of these methods include histogram binning [186], isotonic regression [187], and temperature scaling [179]. The first two techniques are non-parametric, and therefore, the number of parameters utilized is dependent on the training dataset size. Conversely, temperature scaling is a parametric approach that needs a fixed number of parameters.

Given the practicality being a crucial consideration in applying UQ techniques to the engineering domain, this research adopts a straightforward approach: temperature scaling. More specifically, the study employs STD scaling, which is a regression task version of temperature scaling [184]. With STD scaling, it is only necessary to determine a scalar parameter, denoted as s, which is

used to multiply the standard deviation initially estimated by the DE model, $\hat{\sigma}$. The value of s used in the calibration process is selected to minimize the NLL, as shown below:

$$s = \underset{s}{\operatorname{argmin}}\left(\frac{\log\big(s\hat{\sigma}(x)\big)^2}{2} + \frac{\big(y-\hat{\mu}(x)\big)^2}{2\big(s\hat{\sigma}(x)\big)^2} + \frac{\log 2\pi}{2}\right), \tag{4.8}$$

Please note that this equation is the simple modification of Eq. 4.2, where $\sigma(x)$ is replaced by $s\hat{\sigma}(x)$. This calibration procedure is completely separate from the training procedure of DE; it is performed after the mixture step in Fig. 4.2, so it is called the post-hoc or post-process calibration method. It should be emphasized that the model parameters (weights and biases in the NN model) remain unchanged throughout the calibration process. The STD scaling method is intuitively explained as follows: if the estimated uncertainty from the trained model $\big(\hat{\sigma}(x)\big)$ is poorly calibrated, the calibrated version of the uncertainty $s\hat{\sigma}(x)$ is used in its place. It is important to note that this calibration process is intended solely to correct the estimated uncertainty, and therefore, only the output $\hat{\sigma}(x)$ of the DE changes, while the predictive value $\hat{\mu}(x)$ remains unaltered. The steps involved in the STD calibration process are outlined in Algorithm 5. For calibration, a separate dataset should be used that is distinct from the training and test datasets to ensure calibration generalization [184], and therefore, a validation dataset is utilized for the calibration. In multi-output regression tasks, every DE output can be calibrated independently using the number of scaling parameters s equal to the output dimension (this is implemented by the for-loop in line 3 of Algorithm 5). In conclusion, this study uses a straightforward STD calibration method for uncertainty calibration, which involves tuning scalar parameters without modifying trained NNs.

Algorithm 5 STD calibration procedure

1: Prepare calibration dataset X (with input x and output y).

2: Define candidates of scaling factor: S

3: **for** $i = 1 : length(y)$ **do** ▷ Loop for output dimension of DE

4: $$s_i = \underset{s \in S}{\operatorname{argmin}}\left(\frac{\log\left(s\hat{\sigma}_i(x)\right)^2}{2} + \frac{\left(y_i - \hat{\mu}_i(x)\right)^2}{2\left(s\hat{\sigma}_i(x)\right)^2} + \frac{\log 2\pi}{2}\right)$$

5: **end for**

6: Utilize s_i to calibrate estimated uncertainty over ith output

7: ▷ Use $s_i\hat{\sigma}_i$ in lieu of $\hat{\sigma}_i$.

4.3 Application of DE to aerodynamic performance regression task

This section applies the DE method to a real-world engineering problem of predicting aerodynamic coefficients for a specific missile configuration with varying flow conditions. It aims to validate the performance of DE in multi-output regression tasks since no comprehensive study has been conducted on this topic. The section evaluates both the regression and uncertainty estimation performance of DE and investigates the impact of M, the number of NNs used for the ensemble.

4.3.1 Data preparation and training details

After obtaining the training dataset, the next step is to determine the structure of the probabilistic NN to be used for the ensemble: following the work by Lakshminarayanan et al. [49], the identical architectures of the probabilistic NNs are used for ensembling in this study. To this end, grid search is carried out with hyperparameters such as the number of layers, number of nodes, and size of the mini-batch. Other hyperparameters such as the optimizer algorithm, initial learning rate, and total epochs are selected as Adam, 10^{-3}, and 13,000, respectively. The results of the tuning are available in 4.6.2.1. A probabilistic NN with 7 hidden layers and 128 nodes is selected based on both NLL and RMSE, and the mini-batch size is set to 512. Subsequently, different values of the hyperparameter M (2, 4, 8, and 16) are adopted, with each corresponding DE model referred to as DE-2, DE-4, DE-8, and DE-16 in this manuscript. That is, for DE-16, 16 probabilistic NNs with 7 hidden layers and 128 nodes are trained with a mini-batch size of 512, sharing identical hyperparameters with other DE models except M.

GPR models with different kernels are also trained for their hyperparameter tuning. To this end, Matérn 5/2, radial basis function, rational quadratic, and dot-product kernels are examined [188], and their results also can be found in 4.6.2.2. In addition, not only single-output GPR models are tested, but also the multi-output GPR (MOGPR) with radial basis function kernel is trained for more comprehensive comparison with DE [188, 189, 190, 191]. Among them, GPR with Matérn 5/2 kernel shows the best performance, and is therefore selected for the comparison with DE throughout this paper. The required training times for DE with selected NN structure (7 hidden layers and 128 nodes) and GPR (Matérn 5/2 kernel) models using Intel(R) Xeon(R) CPU @ 2.20GHz are as follows: 10.9 hours for GPR, 2.4 hours for DE-2, 5 hours for DE-4, 9.7 hours for DE-8, and 19.4 hours for DE-16. The loss function history of the trained models is shown in Fig. 4.5.

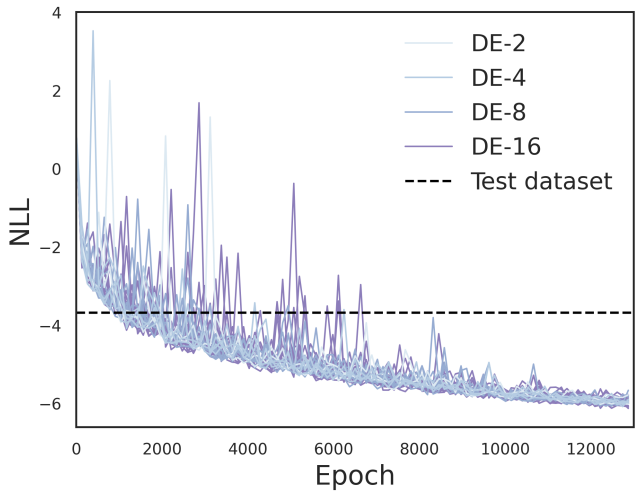


Figure 4.5: Loss history of all trained models. NLL calculated by the test dataset is adopted the results of the hyperparameter tuning (Table 4.3 in Sec. 4.6.2).

4.3.2 Evaluation of regression performance

In this section, the regression performances of all selected models are presented using a test dataset that is not used in model training. Before going into details, DE-2 (which required the least training time among the DE models) is compared with GPR to highlight the efficiency of the DE models. Fig. 4.6 shows the results of kernel density estimation (KDE), which demonstrates the generalization performance of the models by visualizing the distributions of the test data in terms of NLL and RMSE (those of all six QoIs are averaged to be shown in this figure). For both criteria, the obvious superiority of DE-2 can be identified: most of the test data is concentrated in the lower error region in DE-2. More specifically, the KDE of NLL shows that the density peak of DE-2 represented by a star with long dashed line is located at NLL of -4.6, while that of GPR is located at -3.1. When it comes to RMSE, the peak of DE-2 is at RMSE of 0.003 while GPR is at 0.09. The medians of the error metrics are also shown as circles with dotted lines. For both metrics, those of DE-2 are much lower than those of GPR, indicating that DE-2 performs better than GPR overall. The most interesting point here is that although DE-2 requires only 22% of the training time of GPR, it achieves superior regression accuracy.

Fig. 4.7 provides the comprehensive results of the regression performance. Fig. 4.7a shows the NLL results of all models with respect to the six aerodynamic QoIs, and their averaged NLL is also shown at the right end. Throughout all QoIs, GPR shows inferior regression accuracy than all other DE models. The results on NLL could be expected as each NN in the DE model is trained to minimize NLL. However, the results on RMSE in Fig. 4.7b are highly inspiring: they also achieve higher regression accuracy even in terms of RMSE. Considering that numerous engineers use RMSE to evaluate regression models, the fact

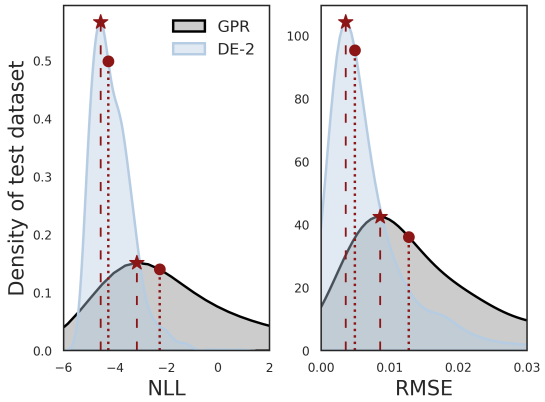
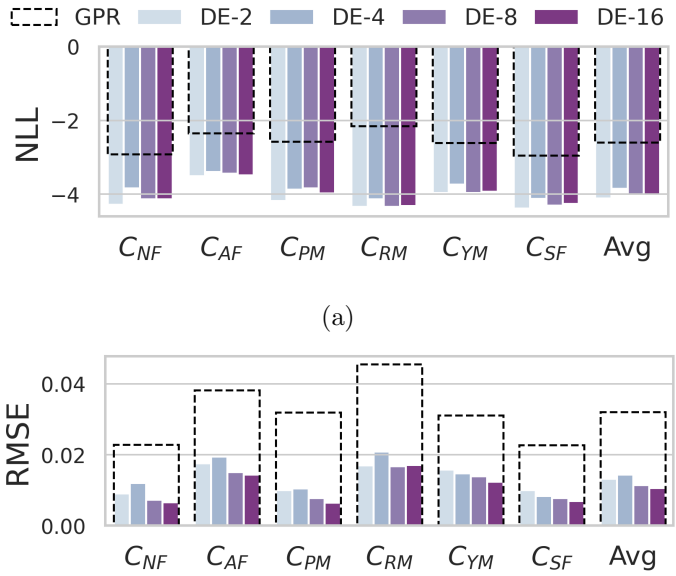


Figure 4.6: Comparison of regression accuracy between GPR and DE-2: kernel density estimation (KDE) of test dataset with respect to NLL and RMSE (averaged values of all six QoIs). The stars and circles represent the maximum and median points of each model, respectively.

that the average RMSE of DE models is less than half that of GPR is quite encouraging. Also, contrary to the claim that DE-5 would be sufficient in the work first proposed DE approach [49], DE-2 seems to be sufficient enough in this study, at least in terms of predictive accuracy: its values between all DE models are insignificant. However, the conventional belief is that the more models used in the ensemble, the more accurate the prediction will be due to the robustness that comes from averaging multiple predictions. The underlying reason for this counter-intuitive result (that is, insignificant differences in predictive accuracy between DE models) is judged to be the insufficient diversity between individual models due to the strategy adopted by DE: identical dataset and model architecture [49]. However, note that blindly ensuring excessive diversity by using different datasets and model architectures should be done with caution, since it can degrade UQ performance (which will be mentioned in **Remark 2** of Sec. 4.3.4).



(a)

(b)

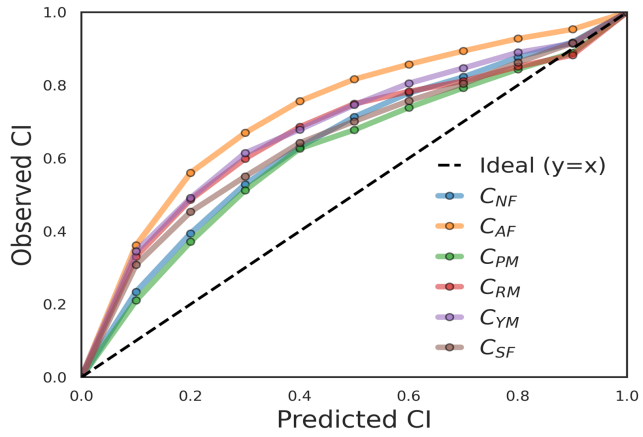
Figure 4.7: Comparison of regression accuracy between GPR and all DE models: comprehensive results in terms of all aerodynamic QoIs. (a) NLL, (b) RMSE.

4.3.3 Evaluation of UQ performance

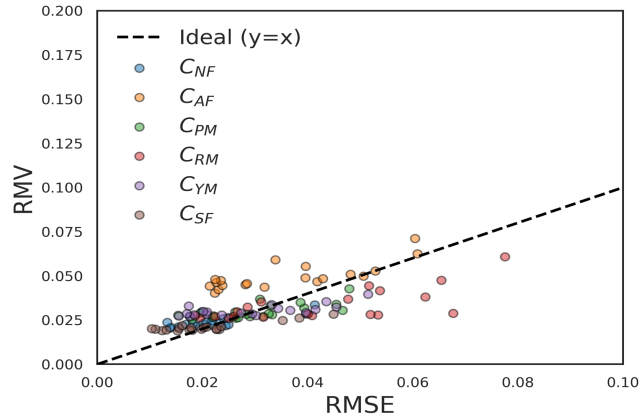
This section examines the quality of the predictive uncertainty, using AUCE and ENCE criteria for the quantitative investigation. For this purpose, reliability plots should be drawn first, using the test dataset split in Sec. 4.3.1 (dataset size of 980). Also, as in Algorithm 3, CI-based reliability plots require the set of CI candidates (P) and error-based reliability plots in Algorithm 4 need the number of bins (B). In this study, $P = \{0.1, 0.2, ..., 0.9\}$ and $B = 20$ are chosen.

Fig. 4.8 shows the results of GPR, and it appears that GPR has a satisfactory uncertainty quality with respect to the error-based reliability plot (Fig. 4.8b), while the CI-based plot (Fig. 4.8a) shows relatively poor quality. In a

CI-based plot, since the predicted CI (x-axis) is underestimated compared to the actual observed CI (y-axis), it can be inferred that GPR is trained to be "underconfident": it is underconfident itself, so it overestimates its uncertainty.



(a)



(b)

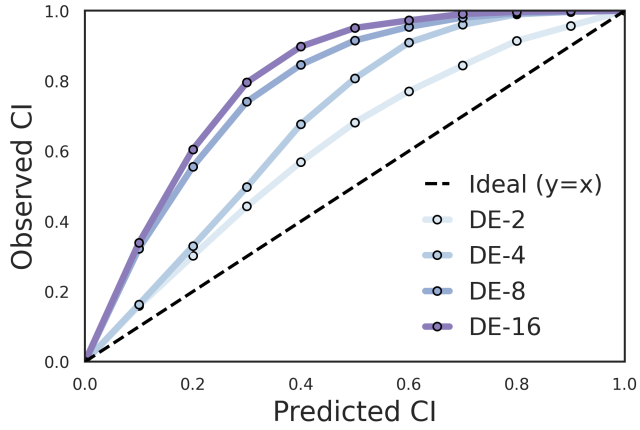
Figure 4.8: Reliability plots of GPR: (a) CI-based reliability plot, (b) Error-based reliability plot.

The results of the DE models are then shown in Fig. 4.9. Note that unlike GPR in Fig. 4.8, only the results of output C_{SF} are visualized to highlight the

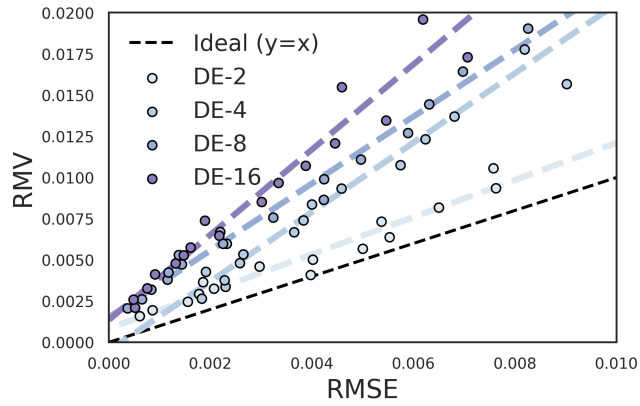
differences between DE models: comprehensive results can be found in Fig. 4.14 in 4.6.3. For DE-2, the CI-based plot (Fig. 4.9a) shows a similar trend to that of GPR, while the error-based plot (Fig. 4.9b) shows slightly better quality. Meanwhile, a notable trend is observed along the increase of M: as it increases, the uncertainty quality with respect to both reliability plots apparently degrades. More specifically, both types of plots move upward away from the $y = x$ ideal line as M increases, indicating that DE models tend to become "underconfident". Considering that DE-16 requires about 8 times as much training time as DE-2, it can be confirmed that using large M values for the ensemble does not necessarily lead to better results, but rather the opposite in terms of uncertainty quality. In this context, assuming that the performance of DE-5 will be between DE-4 and DE-8, it can be inferred that using $M = 5$ as suggested by Lakshminarayanan et al. [49] does not guarantee sufficient UQ quality in this case. In fact, the insight behind this underconfident tendency when ensembling networks in classification tasks can be found in Rahaman et al. [165], while the corresponding tendency in regression has not been proven. Accordingly, in the next section, we provide the mathematical explanation for this underconfident tendency in regression tasks.

4.3.4 Theoretical derivation: underconfidence of DE in regression tasks

This section is for the mathematical derivation of why the ensemble of NNs becomes underconfident, as discovered in the previous section. For this purpose, the deviation from calibration (DC) score is introduced as in Rahaman et al. [165]; their work focused only on classification tasks, so their DC score consisted of the Brier score and the entropic term. Meanwhile, since our work focuses on the regression task, we adopted the different DC score consisting of the MSE



(a)



(b)

Figure 4.9: Reliability plots of DE: for simplicity, only the C_{SF} results of different DE models are shown. (a) CI-based reliability plot, (b) Error-based reliability plot. In (b), to clearly show the decreasing tendency of UQ quality with increasing M, the linear regression model of the scatter points of each DE model is shown as a dashed line with the corresponding color.

and predictive variance. In this context, the following proposition and its proof can be considered as one of the contributions of this paper.

Proposition. *When DC score is defined as follows,*

$$DC(\mu, \sigma) \equiv (y - \mu)^2 - \sigma^2 \tag{4.9}$$

DC score of the ensemble becomes less than or equal to the averaged DC score of the individual NNs.

$$DC(\hat{\mu}, \hat{\sigma}) \leq \frac{1}{M} \sum_{i=1}^{M} DC(\mu_i, \sigma_i) \tag{4.10}$$

Proof. The averaged DC score of the individual NNs (right-hand side of the Eq. 4.10) can be expressed as:

$$\begin{aligned}
\frac{1}{M} \sum_{i=1}^{M} DC(\mu_i, \sigma_i) &= \frac{1}{M} \sum_{i=1}^{M} (y^2 - 2y\mu_i + {\mu_i}^2 - {\sigma_i}^2) \\
&= y^2 - 2y\hat{\mu} + \frac{1}{M} \sum_{i=1}^{M} {\mu_i}^2 - \frac{1}{M} \sum_{i=1}^{M} {\sigma_i}^2 \\
&= (y^2 - 2y\hat{\mu} + \hat{\mu}^2 - \hat{\sigma}^2) + (\frac{1}{M} \sum_{i=1}^{M} {\mu_i}^2 - \hat{\mu}^2) + (\hat{\sigma}^2 - \frac{1}{M} \sum_{i=1}^{M} {\sigma_i}^2) \\
&= \underbrace{(y^2 - 2y\hat{\mu} + \hat{\mu}^2 - \hat{\sigma}^2)}_{=DC(\hat{\mu}, \hat{\sigma})} + 2 \underbrace{(\frac{1}{M} \sum_{i=1}^{M} {\mu_i}^2 - \hat{\mu}^2)}_{=Var(\mu_i)} \quad (\because \text{Eq. 4.4})
\end{aligned} \tag{4.11}$$

Hence,

$$DC(\hat{\mu}, \hat{\sigma}) = \frac{1}{M} \sum_{i=1}^{M} DC(\mu_i, \sigma_i) - \underbrace{2 \cdot Var(\mu_i)}_{\geq 0} \tag{4.12}$$

□

Remark 1. *The DC used in the above proposition indicates the degree of calibration. When DC equals 0, it means that the estimated uncertainty σ^2 exactly matches the MSE, $(y-\mu)^2$. If $DC < 0$, the uncertainty is overestimated compared to the MSE, which is an underconfident case. Therefore, the proposition that the DC score decreases after ensembling has mathematically explained the underconfidence of DE models observed in Sec. 4.3.3.*

Remark 2. *In Sec. 4.3.2, it was mentioned that introducing excessive diversity to individual NNs can lead to degraded UQ performance as a trade-off for predictive accuracy. This can be easily inferred by the term $Var(\mu_i)$ in Eq. 4.12: the more variance NNs have, the more underconfidence their ensemble shows.*

4.4 DE models with STD calibration

The underconfidence tendency of DE models in regression tasks is observed and explained in the previous section. This section suggests the use of post-hoc STD calibration to mitigate this undesirable tendency and examines its effects.

4.4.1 STD calibration of DE models

The findings presented in Sec. 4.3.3 suggest that, despite the prevailing view that DE models are well-calibrated, this is not always the case, as illustrated in this straightforward multi-output regression task within an engineering domain. To address this issue, we propose using the STD calibration method on the trained DE models. This technique, as described in Algorithm 5, is straightforward and practical, as it requires only a single for-loop and leverages the existing models without additional training. This makes it a feasible option for our study, which focuses on the application of DE in engineering, where practicality is crucial.

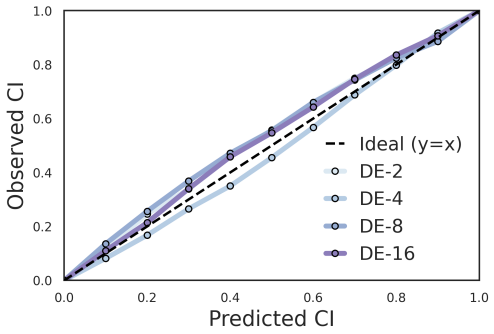
Algorithm 5 first requires a set of candidates for scaling factors, S. Since the scaling factor of 1 corresponds to the case without calibration, the candidates s are set around 1. Accordingly, $s = 10^x$ are chosen as candidates, where x are 100 uniformly distributed points from -2 to 0.18, so that the resulting range of scaling factors to explore is from 0.01 to 1.5. Note that with s less than 1, underconfident models that overestimate the standard deviations (uncertainty) can be calibrated. Finally, the STD calibration is performed using validation dataset split in Section 4.3.1 (dataset size of 980) and the optimized scaling factors for each DE model with respect to each output (QoI) are summarized in Table 4.1. The STD calibration for all models is performed within 60 seconds, which is negligible compared to their training time.

Table 4.1: Optimized scaling factors for STD calibration

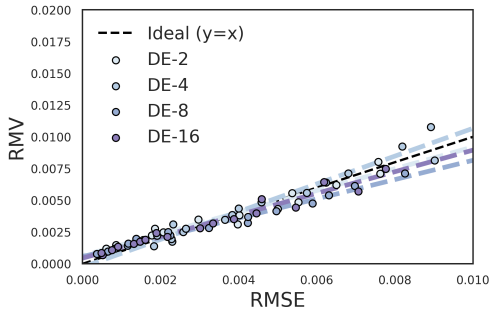
Methods	Optimized scaling factors						
	C_{NF}	C_{AF}	C_{PM}	C_{RM}	C_{YM}	C_{SF}	**Avg**
DE-2	0.549	1.061	0.608	1.009	0.824	0.578	**0.771**
DE-4	0.385	0.405	0.284	0.472	0.257	0.270	**0.345**
DE-8	0.147	0.270	0.133	0.270	0.155	0.140	**0.186**
DE-16	0.103	0.199	0.088	0.189	0.120	0.108	**0.135**

Herein, the scaling factors for all six aerodynamic coefficients and their average value in each model are presented. The most notable point is that almost all s values are less than 1 and they decrease as M increases: see the bold values in Table 4.1 to confirm their average trend. Taken together with the results from Section 4.3.3 that DE models overestimate their σ^2 (become underconfident) as M increases, one might expect optimized $s < 1$ to mitigate this underconfident tendency. And Fig. 4.10 proves that this actually happens: reliability plots of the DE models after STD calibration are drawn with the test dataset. Note that the validation dataset used during the STD calibration should not be reused in this process for generalization purposes. When compared to the previous plots in Fig. 4.9, the obvious improvement due to the calibration technique can be observed. See 4.6.4 for comprehensive results on the calibration effects with respect to all six QoIs.

Then, the quantitative effects of the calibration in terms of AUCE and ENCE will be analyzed, and from now on DE before and after calibration will be referred to as DE-bef and DE-aft, respectively. The AUCE and ENCE of the GPR will also be presented for the comparison, but please note that the GPR can be considered inherently STD-calibrated since its training algorithm already aims to minimize NLL as in the STD calibration process. This means that the



(a)



(b)

Figure 4.10: Reliability plots of DE after STD calibration: (a) CI-based reliability plot, (b) Error-based reliability plot. The noticeable effects of STD calibration can be found when compared with the corresponding figure before STD calibration, Fig. 4.9.

GPR does not require additional STD calibration for a fair comparison with DE-aft because it can be seen as having already undergone STD calibration. Finally, the results are summarized in Fig. 4.11. It consists of the sub-figures, where the row indicates each UQ metric, the column indicates each QoI, and the x-axis in each sub-figure indicates whether the DE undergoes STD calibration (as explained, GPR metrics have a constant value along the x-axis regardless of the STD calibration). Before the calibration, the AUCE (upper row) of GPR is

between DE models: DE-2 is better than GPR, DE-4 is similar, and DE-8 and DE-16 are worse. However, the STD calibration completely changes this situation: AUCE of all DE models for all aerodynamic QoIs decreases dramatically. For all outputs, DE-aft clearly outperforms GPR. The significant improvement of ENCE (lower row) due to the calibration of DE can also be verified. DE models show worse performance than GPR without calibration, but this gap narrows and even reverses, as seen in the rightmost subplot "Avg", which shows the average performance of all outputs. In summary, vanilla DE outperformed GPR in terms of training efficiency and regression accuracy, but not in terms of quality of estimated uncertainty. However, when used with a simple post-hoc STD calibration (which requires negligible additional post-processing time), DE demonstrated its strong potential as an alternative to GPR in terms of training time, prediction accuracy, and also UQ quality.

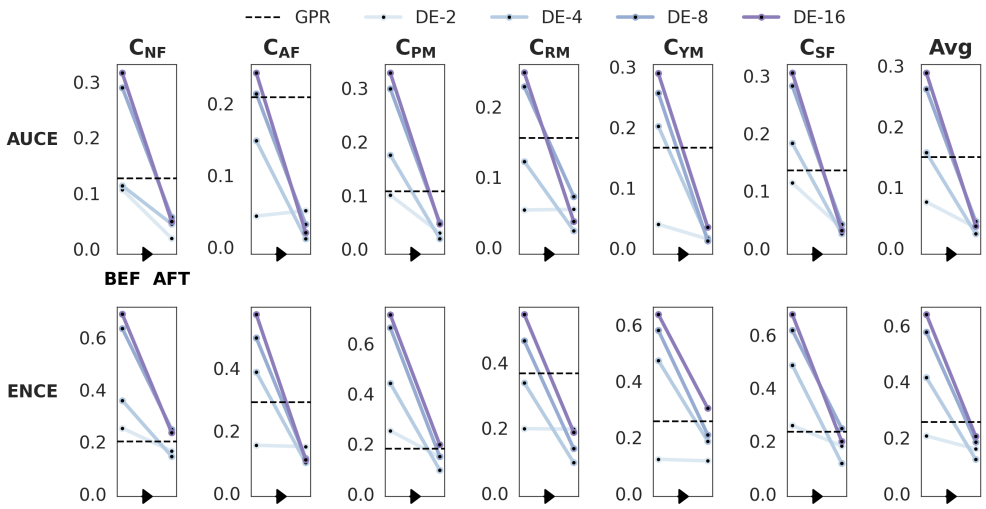


Figure 4.11: AUCE and ENCE of DE models before and after STD calibration. Those of GPR are also shown for comparison.

4.4.2 Effects of STD calibration on Exploratory Behavior in Bayesian optimization

Since the scaling factors are optimized to have values less than 1 during the STD calibration process (Table 4.1), it is obvious that the overall predictive uncertainty of DE models would decrease. To provide a more intuitive understanding of the practical implications of calibration, this section aims to briefly point out that applying STD calibration to DE can lead to different exploratory behavior during Bayesian optimization. Specifically, the importance of calibration is highlighted by comparing the next query candidates before and after STD calibration obtained in the first iteration of Bayesian optimization. Note that only the first iteration is implemented in this paper for the following two reasons. First, the goal of this section is simply to show how calibrated DE leads to different exploratory behavior than vanilla DE. Second, the purpose of this section is not to claim that the final converged results of Bayesian optimization can be different depending on the calibration. Rather, it is to point out that the intended balance between exploitation and exploration may not be realized due to the miscalibrated uncertainty of the vanilla DE, which may affect the convergence history of the Bayesian optimization.

Before moving on to Bayesian optimization, CIs of the 68% confidence level predicted by DE-16 model are shown in Fig. 4.12 to visually understand the impact of calibration. Only one input variable, AoA, is used for the illustration. And its value is standardized to distinguish between its ID (in-distribution) region and the OOD (out-of-distribution) region: in Fig. 4.12, the ID region is defined as the area containing 95% of the train data, while the OOD region is the remaining area. Overall, both results—those obtained before and after STD calibration—show diverging CIs in OOD and relatively narrow CIs in ID for all six QoIs. However, as expected from the scaling factors less than 1, the

CIs from the DE-aft models become significantly narrower than those from the DE-bef models, indicating that these discrepancies will lead to differences in the subsequent Bayesian optimization process.

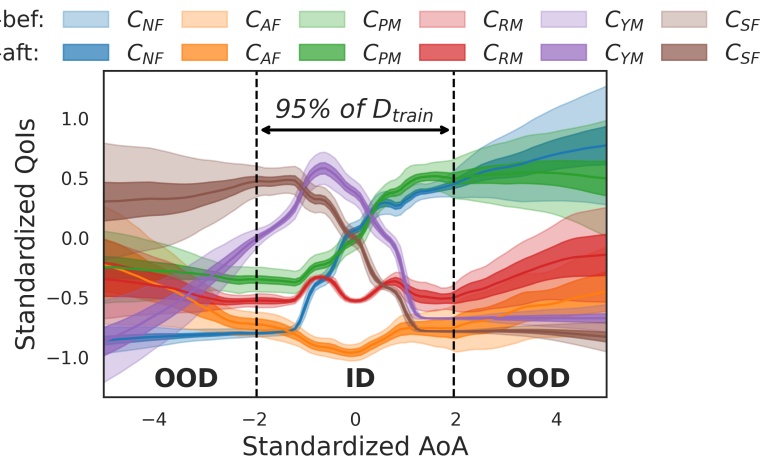


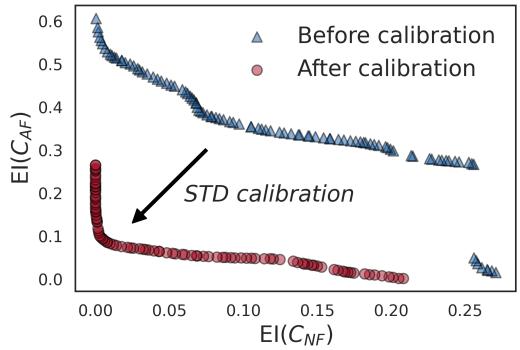
Figure 4.12: CIs of 68% confidence level predicted by DE-16: comparison between before and after STD calibration.

Then, the multi-objective Bayesian optimization problem is defined is adopted to practically investigate their effects on Bayesian optimization: maximization of both C_{NF} and C_{AF} within five varying input parameters (Ma, ϕ, δp, δr, and AoA). These optimizations, coupled with the expected improvement (EI) acquisition function, are performed separately for DE-bef and DE-aft models. The former searches for the maximum EI point where EI is calculated from the uncertainty quantified by the DE-bef model, while the latter does so using the uncertainty quantified by DE-aft. To find the Pareto solutions of $EI(C_{NF})$ and $EI(C_{AF})$, NSGA-II in the Python package pymoo is utilized [98, 18, 192]. inally, the obtained Pareto solutions from the first iteration are shown in Fig. 4.13a. Since the uncertainty estimated by DE-bef and DE-aft are different as

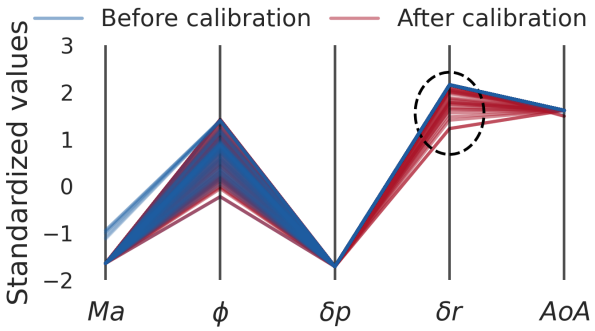
shown in Fig. 4.12, the Pareto solutions of $EI(C_{NF})$ and $EI(C_{AF})$ are also different: EI values of both QoIs after calibration are much smaller than those before calibration.

In Bayesian optimization, however, the most valuable information to the user is not the EI value itself (Fig. 4.13a). More important are the values of the input variable sets (Fig. 4.13b) obtained from the EI Pareto solutions: they are the next query candidates, the main purpose of implementing Bayesian optimization. Additional experiments/simulations will be performed on these candidate queries, indicating that their selection has a significant impact on the convergence of the iterative Bayesian optimization process. If unintended candidates are obtained due to inaccurate UQ and therefore inaccurate EI calculation, the convergence of Bayesian optimization can be much different from the intention of the user. That is, the intended balance between exploitation and exploration during Bayesian optimization may differ due to unintentionally overestimated/underestimated uncertainty. To inspect its unintended exploratory behavior more specifically, the parallel coordinates plot (PCP) in Fig. 4.13b shows how the first query candidates in Bayesian optimization can vary due to the STD calibration in the DE model. This PCP has five vertical lines corresponding to each input variable, and the y-axis indicates their standardized values. Each red/blue line represents each point of the Pareto solutions in Fig. 4.13a. Comparing them, large variations are found especially in the input variable δr. That is, Bayesian optimization coupled with DE-bef discourages exploration of the variable δr (which was not intended by the user), while DE-aft encourages exploration within δr (which was the original intention). In conclusion, whether the DE is calibrated by STD calibration or not can result in exploration characteristics that differ from the user's intended balance between exploitation and exploration in Bayesian optimization, which cautions against

blindly applying vanilla DE models to Bayesian optimization in regression tasks.



(a) Pareto solutions obtained from multi-objective EI optimization



(b) PCP of design variables in Pareto solutions

Figure 4.13: Effects of STD calibration for DE models on Bayesian optimization results.

4.5 Summary

This chapter comprehensively investigated the state-of-the-art approximate Bayesian inference approach, DE. It is applied to the multi-output regression task, which is the most common task in the engineering fields: a simple test case is adopted where aerodynamic QoIs of the specific missile configuration are predicted under varying flow conditions. DE models with different numbers of NNs are trained and then examined in the following order. First, their regression performance and the quality of estimated uncertainty are scrutinized while being compared with GPR. Then, a simple post-hoc STD calibration method is proposed to be applied to miscalibrated DE models. Finally, the effectiveness of the calibration on DE is highlighted by the improvement of two UQ quality criteria and the different exploratory behavior in Bayesian optimization before and after calibration. The key findings of this study can be summarized as follows:

1. The effect of the number of NNs used in ensemble, M, is comprehensively investigated in the simple multi-output regression task. For regression accuracy, DE models show superior performance to GPR in terms of RMSE and NLL, while showing indistinguishable differences among themselves. For UQ quality, however, they show the obvious trend toward underconfidence as M increases, both in terms of AUCE and ENCE criteria. The mathematical proof of why DE tends to be miscalibrated in regression tasks is also derived.

2. The post-hoc STD calibration method, which simply modifies the estimated uncertainty from DE, is proposed to be applied to miscalibrated DE models. Finally, the reliability of the UQ performance after calibration is dramatically improved for both AUCE and ENCE, also surpassing

that of GPR.

3. The impact of the calibration approach on the exploratory behavior in Bayesian optimization is examined. Finally, whether or not the DE is calibrated via STD calibration can result in completely different exploration characteristics when extended to Bayesian optimization, which cautions against blindly applying vanilla DE models to Bayesian optimization in regression tasks.

4. We have demonstrated that by applying a simple post-hoc STD calibration technique that requires negligible additional post-processing time, DE models can have enormous potential compared to GPR, which is the most commonly used regression model for UQ in engineering. These results are summarized in Table 4.2, where the DE-2 model after STD calibration outperforms GPR in terms of regression performance (−56% NLL & −55% RMSE), reliability of UQ (−77% AUCE & −38% ENCE), and training efficiency (−78% training time).

Table 4.2: Comprehensive comparison between GPR and DE-2

Metrics		GPR	DE-2	
			Before calibration	After calibration
Regression	NLL	-2.653 (−%)	-4.145 (↓ **56**%)	-4.145 (↓ **56**%)
	RMSE	0.029 (−%)	0.013 (↓ **55**%)	0.013 (↓ **55**%)
UQ	AUCE	0.150 (−%)	0.076 (↓ 49%)	0.034 (↓ **77**%)
	ENCE	0.256 (−%)	0.206 (↓ 20%)	0.159 (↓ **38**%)
Training time [s]		39081 (−%)	8640 (↓ **78**%)	8640+30 (↓ **78**%)

The presented DE framework has great promise in two engineering applications. First, DE with STD calibration has the potential to replace the most com-

mon regression model, GPR, owing to its following advantages: more scalable to large datasets, higher regression accuracy, and last but not least, more reliable uncertainty estimation. Second, DE with STD calibration can be leveraged in Bayesian optimization by ensuring a reliable balance between exploitation and exploration due to its trustworthy UQ performance. Although the application of this framework has been demystified using the simple multi-output regression task, it can be easily applied and extended to high-dimensional input/output problems since it is based on the deep neural network structures and no special assumptions have been made for this specific problem. For future work, a more comprehensive investigation of DE models will be conducted, such as their scalability to other practical engineering regression problems. Also, since the extension of DE to the whole Bayesian optimization was outside the focus of our paper, comparing the convergence history of DE-bef and DE-aft over the entire iterations can be a future work.

4.6 Additional results

4.6.1 Controversial issues on MC-dropout

Osband [156] pointed out that what MC-dropout (MCD) estimates is a risk, not an uncertainty, and also emphasized the pitfalls of MCD when used as a naive tool for estimating uncertainty. Moreover, its algorithm does not perform adequately even in very simple examples [193, 194], and its posterior samples are often too spiky to provide a reliable predictive uncertainty trend [154, 48, 193, 195, 51], which makes it unattractive to be exploited for Bayesian optimization.

4.6.2 Hyperparameter tuning results in Sec. 4.3.1

4.6.2.1 Results of DE models

The results of the hyperparameter tuning for DE models performed in Sec. 4.3.1 are shown in Table 4.3. Three hyperparameters are used: the number of hidden layers ($N_{layer} \in \{3, 5, 7\}$), the number of nodes in each hidden layer ($N_{node} \in \{32, 64, 128\}$), and the size of the mini-batch ($N_{batch} \in \{512, 1024, 2048\}$). The corresponding regression performance in terms of NLL and RMSE of all hyperparameter combinations are shown (total training time of 6944 seconds). Since $N_{layer} = 7, N_{node} = 128, N_{batch} = 512$ shows the best RMSE performance, it is selected as the best hyperparameter combination.

Table 4.3: Results of hyperparameter tuning: several structures of probabilistic NN used in the DE model are tested.

N_{layer}	N_{node}	N_{batch}	NLL	RMSE	Time [s]
3	32	512	-2.50	0.125	183
		1024	-2.22	0.200	**174**
		2048	-2.13	0.183	183
	64	512	-2.98	0.076	202
		1024	-2.69	0.087	179
		2048	-2.52	0.100	213
	128	512	-3.37	0.036	256
		1024	-3.34	0.040	257
		2048	-2.55	0.052	271
5	32	512	-2.62	0.137	187
		1024	-2.23	0.173	214
		2048	-2.25	0.137	202
	64	512	-3.21	0.039	236
		1024	-2.79	0.055	210
		2048	-2.01	0.085	255
	128	512	**-3.83**	0.017	342
		1024	-3.53	0.019	327
		2048	-3.16	0.035	349
7	32	512	-2.49	0.088	212
		1024	-2.36	0.119	224
		2048	-2.17	0.104	244
	64	512	-3.38	0.031	286
		1024	-3.05	0.042	247
		2048	-2.45	0.062	262
	128	**512**	-3.82	**0.016**	428
		1024	-3.56	0.024	399
		2048	-3.13	0.035	402

4.6.2.2 Results of GPR models

The results of the hyperparameter tuning for GPR models performed in Sec. 4.3.1 are shown in Table 4.4. For conventional single-output GPR (SOGPR), Matérn 5/2, radial basis function, rational quadratic, and dot-product kernels are explored. Additionally, multi-output GPR (MOGPR) with radial basis function is also tested. The corresponding regression performance in terms of NLL and RMSE of all GPR models are summarized in Table 4.4 (total training time is 205493 seconds, which is significantly longer than Sec. 4.6.2.1). MOGPR requires the least training time, but single-output GPR with Matérn 5/2 is selected since it shows the best performance with respect to NLL and RMSE.

Table 4.4: Results of hyperparameter tuning: several GPR models are tested.

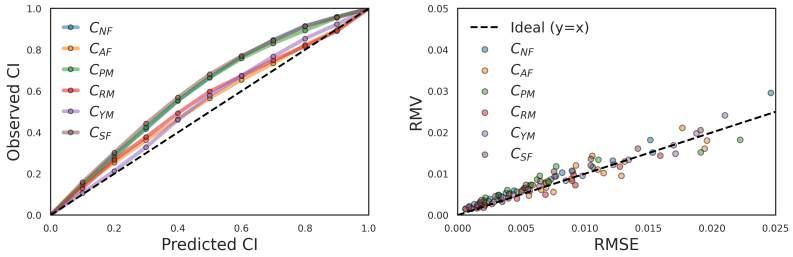
Kernels	NLL	RMSE	Time [s]
Matérn 5/2	**-2.653**	**0.029**	39081
Radial basis function (SOGPR)	-1.657	0.039	64250
Radial basis function (MOGPR)	-1.4236	0.044	**8136**
Rational quadratic	-0.747	0.061	66363
Dot-product	0.440	0.547	27663

4.6.3 Additional results in Sec. 4.3.3

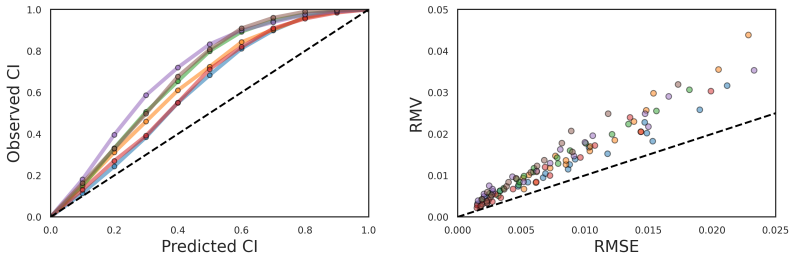
Fig. 4.9 in Sec. 4.3.3 shows the reliability plots with respect to only one QoI, C_{SF}. In this section, more comprehensive results are provided as Fig. 4.14. For each vanilla DE model, all six QoIs are shown with different colors. Again, the trend of underconfidence as M increases can be seen from DE-2 to DE-16.

4.6.4 Additional results in Sec. 4.4.1

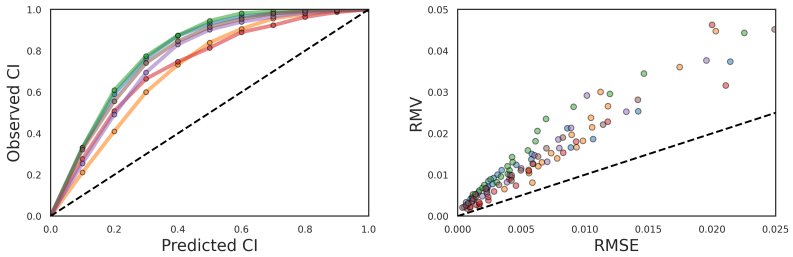
In 4.6.3, the reliability plots of DE models before STD calibration (vanilla DE models) are shown; this section shows the results after STD calibration as Fig. 4.15. It is shown that all DE models become well-calibrated after calibration, even for the DE-16 model, which was the most miscalibrated DE model.



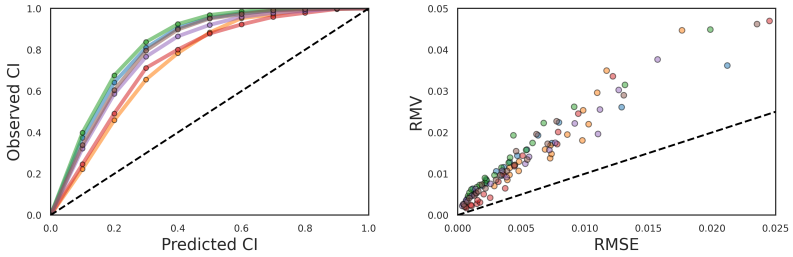
(a) DE-2



(b) DE-4

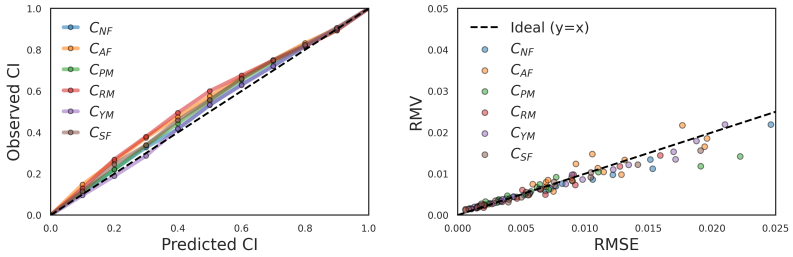


(c) DE-8

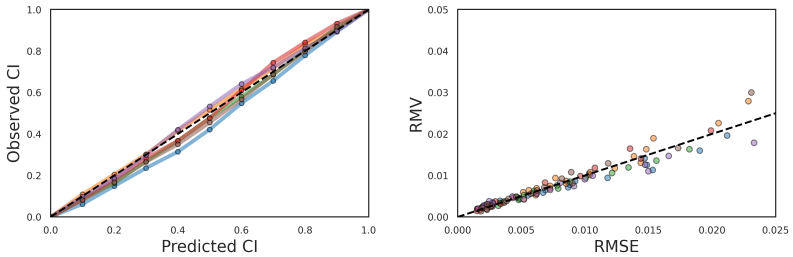


(d) DE-16

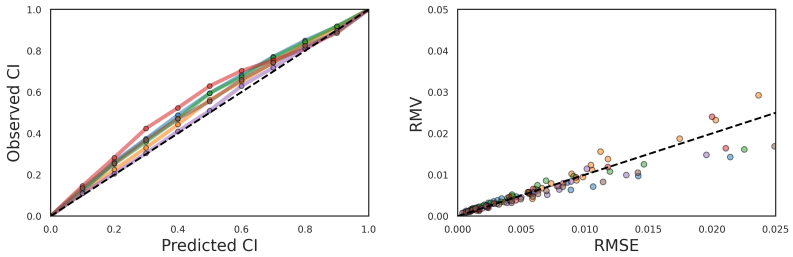
Figure 4.14: Reliability plots of vanilla DE models: (left) CI-based reliability plots, (right) error-based reliability plots.



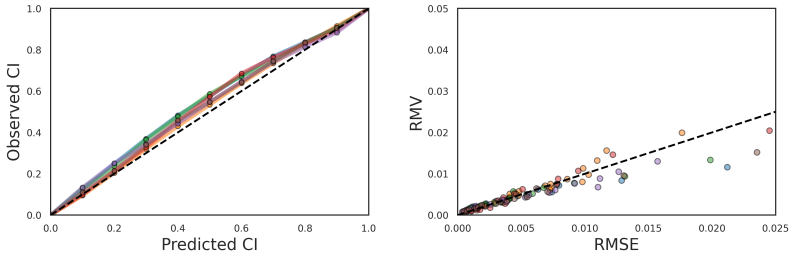
(a) DE-2



(b) DE-4



(c) DE-8



(d) DE-16

Figure 4.15: Reliability plots of DE models after STD calibration: (left) CI-based reliability plots, (right) error-based reliability plots.

Chapter 5

Concluding remarks

5.1 Summary of the dissertation

This dissertation aims to address three major bottlenecks that hinder the effective use of regression models in the aerodynamic design process. First, the curse of dimensionality poses a challenge due to the high-dimensional input space required for aerodynamic design resulting from geometric deformation freedom and various flight conditions. The exponential growth in the number of possible configurations within the input space makes it difficult to thoroughly explore the design space, leading to a sparsity problem and inaccurate capture of complex input-output relationships. In addition, the curse of dimensionality makes it impractical to apply gradient-free optimizers to optimization tasks, limiting the ability to find reasonable solutions. Second, popular regression models are primarily designed for single-output prediction, which limits their applicability in high-dimensional output spaces. Training these models independently for each output not only increases training time linearly with the

output dimension but also neglects correlations within outputs, limiting their accuracy in modeling complex physical systems. Finally, reliable UQ is essential to account for the inherent risk in the aerodynamic design process as neglecting uncertainty can lead to suboptimal designs and unexpected failures. Therefore, coupling regression models with UQ techniques allows engineers to obtain not only predictive values, but also variance or confidence intervals, enabling informed decision-making and evaluation of the reliability of design parameters. By addressing these bottlenecks, this dissertation focuses on improving the effectiveness and reliability of regression models in aerodynamic design.

The first bottleneck, the high-dimensional input space in aerodynamic design, was addressed by the DR technique that finds the low-dimensional latent representation of the high-dimensional original data (Chapter 2). The DR technique can significantly reduce the dimensionality of the input space to a level suitable for effective training of the regression models. By alleviating the curse of dimensionality, the feasibility of applying gradient-free optimizers to the reduced input space was also investigated. For this purpose, the inverse design optimization of the wind turbine airfoil was taken as a case study. Finally, a novel inverse design optimization framework with a two-step deep learning approach, which refers to the successive coupling of VAE and MLP, was proposed. Here, VAE generates a realistic target distribution, and MLP predicts the QoIs and shape parameters from the generated distribution. Then, the target distribution is optimized based on this two-step approach. To increase the accuracy, active learning was used to retrain the models with newly added designs. In addition, transfer learning was coupled to reduce the computational cost of retraining, thereby increasing the accuracy of the framework with efficient computational resources. The proposed framework was validated using two constrained optimization problems: single-objective and multi-objective airfoil optimizations of

the tip region of a megawatt-class wind turbine blade. In the single objective optimization, the predictive accuracy of the trained MLP model and the validity of the trained VAE model for generating realistic data were verified. The multi-objective results verified a continuous mapping between the inputs and outputs of the framework, which enables successful optimization through the two-step approach. Furthermore, this mapping was confirmed to accurately reflect the rapid changes in QoIs, which frequently occur in real-world engineering applications. In summary, the results of the optimizations showed that the proposed framework for inverse design optimization via a two-step deep learning approach is accurate, efficient, and flexible enough to be applied to any other regression task with a high-dimensional design space.

To address the second bottleneck, the high-dimensional output space, this dissertation focused on the application of ROM techniques (Chapter 3). ROM specializes in predicting high-dimensional QoIs by treating a high-fidelity CFD simulation as a black-box function and learning simplified models in a data-driven manner. Similar to dealing with the high-dimensional input space, the DR process is required to construct a latent space in the ROM. This dissertation specifically investigated the influence of this latent space, which acts as an intermediary for predicting high-dimensional data, on ROM performance. To validate this approach, the prediction of flow fields around a transonic airfoil was used as a case study. In this study, a novel framework was proposed: a physics-aware ROM based on physics-aware LVs. These LVs are interpretable and information-intensive, extracted using the β-VAE. The validation process for this framework follows a systematic approach. First, the process of extracting physically meaningful LVs was thoroughly investigated by quantitatively estimating their independence and information intensity. Next, the actual physical meanings associated with these LVs were investigated in detail. Finally, the ef-

fectiveness of the proposed physics-aware ROM was compared to conventional ROMs and its superiority was established. Through these validation steps, the dissertation successfully demonstrated the effectiveness of the physics-aware ROM framework for the 2D transonic benchmark problem. The proposed framework not only addressed the challenges posed by the high-dimensional output space, but also provided valuable insights into the independence, information intensity, and physical interpretations of the extracted LVs. This research contributed to the improvement of the regression task of high-dimensional data and to the understanding of the underlying physics in aerospace design.

The last bottleneck, reliable and efficient UQ of the regression model, was addressed by the DE approach (Chapter 4). Since DE is based on neural networks, it offers all of the following: universal approximation capability, scalability to large datasets, and multi-output regression. Last but not least, it is able to quantify the predictive uncertainty by a simple modification of the conventional MLP structure. This dissertation aimed to validate this simple and scalable DE method for multi-output regression tasks, which are the most common problems in practical engineering disciplines. To this end, a simple test case was adopted where aerodynamic QoIs of the specific missile configuration are predicted under varying flow conditions. DE models with different numbers of NNs were trained and then investigated in the following order. First, their regression performance and the quality of the estimated uncertainty were investigated and compared with GPR. Then, a simple post-hoc STD calibration method was proposed to be applied to miscalibrated DE models. Finally, the effectiveness of calibration on DE was highlighted by the improvement of two UQ quality criteria and the difference in Bayesian optimization results before and after calibration.

5.2 Limitations of the dissertation

This dissertation attempted to speed up the aerodynamic design process by alleviating the following three bottlenecks: 1) high-dimensional input space, 2) high-dimensional output space, and 3) reliable and efficient UQ. However, since the case study for each bottleneck was selected independently, the comprehensive case study can be the future work of this dissertation. In fact, there is an ongoing project to create a Python library named "**SubDamian**: **SU**rrogate-**B**ased **DA**ta **MI**ning **AN**alysis." It is currently under construction (`https://github.com/sunwoong-yang/SubDamian`), including all the methods used in this dissertation. Specifically, it will be divided into three categories: surrogate (regression) models, dimensionality reduction, and data mining techniques. For the category of regression models, the most popular models used in engineering disciplines are covered, including GPR, MLP, and DE, which are used in this study. For the dimensionality reduction category, various DR approaches are considered, including POD, AE, VAE, and β-VAE, all of which are also adopted in this dissertation. By using regression models and dimensionality reduction techniques simultaneously, the ROM performed in this dissertation can be implemented. Finally, since this dissertation focused on the regression models, the post-processing methods including data mining analysis and optimization based on them will also be incorporated for the practicality of this library. For the data mining analysis, various techniques such as sensitivity analysis, analysis of variance, self-organizing maps, parallel coordinate plot, and decision tree will be incorporated [19]. With this library, the comprehensive case study that copes with all three bottlenecks mentioned in this thesis, that is, high-dimensional input and output with reliable and efficient UQ, can be explored as future work.

Also, the three case studies adopted in this dissertation for each bottleneck, pose limitations in terms of their engineering practicality. In this regard, the following are the suggestions for future work of each case study that can promote their practicality in engineering-oriented problems.

For the case study of the high-dimensional input space, the inverse design of the airfoil was performed (Chapter 2). However, it is known that training the network for the inverse design alone can be inefficient due to the non-uniqueness mapping from the performance distribution to the design parameters [196, 197]. Although the non-uniqueness mapping did not significantly affect the effectiveness of the inverse design framework in this case study, it can be an obstacle with respect to the scalability of the proposed framework. For example, the extension to a 3-dimensional flow analysis or high-fidelity CFD solvers may lead to the impossibility of a one-to-one correspondence due to the complex physical characteristics. In this context, attaching a pre-trained forward network at the end of the inverse network can be a solution [197]. During training, only the inverse network is trained, while the pre-trained network is frozen: the loss function is the discrepancy between the performance distribution input to the inverse network and the predicted performance distribution output from the forward network.

Then, in Chapter 3, the flow field around the transonic airfoil was predicted for the case study of the high-dimensional output space. It is assumed that the generating factors were known, which is not the case in real engineering problems. Therefore, the physics-aware latent variables cannot be straightforwardly determined in engineering applications; in this study, the 1000-VAE model could be selected for physics-aware ROM because there were two generating factors and the corresponding β-VAE activated only two latent variables. However, there is no need to extract the exact physical generating factors for ROM. That

is, ROM with β-VAE does not require the use of the exact generating factors of the dataset: the main goal of physics-aware ROM was to find and leverage compact latent space, unlike conventional AE-based ROM, and to prevent the naive use of entangled and therefore physics-unaware latent space during the ROM process. In this context, physics-aware ROM was summarized in Algorithm 1 to emphasize that it does not require the extraction of exact generating factors, but only makes use of the compact and therefore physics-aware latent space. Therefore, this framework can be extended to real-world engineering applications regardless of whether the generating factors are known in advance or not. In this regard, future work can be more practical applications to various real-world problems to demonstrate its effectiveness and generality.

Finally, the prediction of the missile aerodynamic performance was conducted in Chapter 4. Despite the use of the low-fidelity flow solver, the scalability of the proposed UQ approach based on STD-calibrated DE can be considered insignificant with respect to the fidelity of the flow solver, since only the general trends of the physics are considered through the form of the aerodynamic coefficients. However, there need to be further studies on the aleatory uncertainty and the dimension of the output space. This study exploited a deterministic computer solver (Missile Datcom), so the ability of DE to decompose the predictive uncertainty into the aleatory and epistemic uncertainty cannot be verified. Also, since the case study was adopted as a multi-output regression task with six QoIs, a high-dimensional prediction task (such as in Chapter 3) needs to be performed to investigate the scalability of the proposed UQ framework in the much higher output space dimension.

5.3 Embarking on a journey towards acceleration of 3D aerodynamic simulations

The academic progress sparked by this dissertation will evolve into the following visionary framework: **uncertainty-aware multi-fidelity reduced-order modeling** aiming at **acceleration of 3D aerodynamic simulations**. And this framework will stretch its boundaries to the computationally expensive aerodynamic design process of the 3D configurations with the following three phases. 1) Harness AI-driven generative modeling to create diverse 3D design configurations. 2) Deploy multi-fidelity reduced-order modeling to evaluate the aerodynamic performances of the generated designs. 3) Recommend design candidates taking into account the uncertainty inherent in evaluated performances. Beyond academia, the breakthrough pioneered by this design framework has the potential to illuminate a path to the realization of digital twins that will resonate across industries as a beacon of innovation.

Chapter 6

References

[1] R. J. McGhee and W. D. Beasley, "Low-speed aerodynamic characteristics of 17-percent-thick airfoil section designed for general aviation applications," no. NASA TN D-7428, 1973.

[2] J. P. Slotnick, A. Khodadoust, J. Alonso, D. Darmofal, W. Gropp, E. Lurie, and D. J. Mavriplis, "Cfd vision 2030 study: a path to revolutionary computational aerosciences," Tech. Rep., 2014.

[3] L. Huang, Z. Gao, and D. Zhang, "Research on multi-fidelity aerodynamic optimization methods," *Chinese Journal of Aeronautics*, vol. 26, no. 2, pp. 279–286, 2013.

[4] Y. Kim, S. Lee, K. Yee, and D.-H. Rhee, "High-to-low initial sample ratio of hierarchical kriging for film hole array optimization," *Journal of Propulsion and Power*, vol. 34, no. 1, pp. 108–115, 2018.

[5] Z.-H. Han and S. Görtz, "Hierarchical kriging model for variable-fidelity surrogate modeling," *AIAA Journal*, vol. 50, no. 9, pp. 1885–1896, 2012.

[6] X. Zhang, F. Xie, T. Ji, Z. Zhu, and Y. Zheng, "Multi-fidelity deep neural network surrogate model for aerodynamic shape optimization," *Computer Methods in Applied Mechanics and Engineering*, vol. 373, p. 113485, 2021.

[7] A. Jameson and J. Alonso, "Automatic aerodynamic optimization on distributed memory architectures," in *34th Aerospace Sciences Meeting and Exhibit*, 1996, p. 409.

[8] R. B. Langtry and F. R. Menter, "Correlation-based transition modeling for unstructured parallelized computational fluid dynamics codes," *AIAA journal*, vol. 47, no. 12, pp. 2894–2906, 2009.

[9] P. Khayatzadeh and S. Nadarajah, "Aerodynamic shape optimization of natural laminar flow (nlf) airfoils," in *50th AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition*, 2012, p. 61.

[10] G. K. Kenway, C. A. Mader, P. He, and J. R. Martins, "Effective adjoint approaches for computational fluid dynamics," *Progress in Aerospace Sciences*, vol. 110, p. 100542, 2019.

[11] B. Christianson, "Reverse accumulation and attractive fixed points," *Optimization Methods and Software*, vol. 3, no. 4, pp. 311–326, 1994.

[12] A. Griewank and C. Faure, "Piggyback differentiation and optimization," in *Large-scale PDE-constrained optimization*. Springer, 2003, pp. 148–164.

[13] M. B. Giles, M. C. Duta, J.-D. Muller, and N. A. Pierce, "Algorithm developments for discrete adjoint methods," *AIAA journal*, vol. 41, no. 2, pp. 198–205, 2003.

[14] S. Xu, D. Radford, M. Meyer, and J.-D. Müller, "Stabilisation of discrete steady adjoint solvers," *Journal of computational physics*, vol. 299, pp. 175–195, 2015.

[15] T. A. Albring, M. Sagebaum, and N. R. Gauger, "Efficient aerodynamic design using the discrete adjoint method in su2," in *17th AIAA/ISSMO multidisciplinary analysis and optimization conference*, 2016, p. 3518.

[16] S. Jeong, M. Murayama, and K. Yamamoto, "Efficient optimization design method using kriging model," *Journal of Aircraft*, vol. 42, no. 5, pp. 1375–1375, 2005.

[17] S. Nikolopoulos, I. Kalogeris, and V. Papadopoulos, "Non-intrusive surrogate modeling for parametrized time-dependent partial differential equations using convolutional autoencoders," *Engineering Applications of Artificial Intelligence*, vol. 109, p. 104652, 2022.

[18] S. Yang, S. Lee, and K. Yee, "Inverse design optimization framework via a two-step deep learning approach: application to a wind turbine airfoil," *Engineering with Computers*, pp. 1–17, 2022.

[19] S. Yang and K. Yee, "Design rule extraction using multi-fidelity surrogate model for unmanned combat aerial vehicles," *Journal of Aircraft*, pp. 1–15, 2022.

[20] Y.-E. Kang, S. Yang, and K. Yee, "Physics-aware reduced-order modeling

of transonic flow via β-variational autoencoder," *Physics of Fluids*, vol. 34, no. 7, p. 076103, 2022.

[21] J. R. Martins and A. Ning, *Engineering design optimization*. Cambridge University Press, 2021.

[22] M. D. Buhmann, *Radial basis functions: theory and implementations*. Cambridge university press, 2003, vol. 12.

[23] N. Cristianini, J. Shawe-Taylor *et al.*, *An introduction to support vector machines and other kernel-based learning methods*. Cambridge university press, 2000.

[24] J. Wang, "An intuitive tutorial to gaussian processes regression," *arXiv preprint arXiv:2009.10862*, 2020.

[25] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. Cambridge, Massachusetts: MIT press, 2016.

[26] E. VanDerHorn and S. Mahadevan, "Digital twin: Generalization, characterization and implementation," *Decision Support Systems*, vol. 145, p. 113524, 2021.

[27] O. U. Espinosa Barcenas, J. G. Quijada Pioquinto, E. Kurkina, and O. Lukyanov, "Surrogate aerodynamic wing modeling based on a multilayer perceptron," *Aerospace*, vol. 10, no. 2, p. 149, 2023.

[28] K. Balla, R. Sevilla, O. Hassan, and K. Morgan, "An application of neural networks to the prediction of aerodynamic coefficients of aerofoils and wings," *Applied Mathematical Modelling*, vol. 96, pp. 456–479, 2021.

[29] G. Sun, Y. Sun, and S. Wang, "Artificial neural network based inverse design: Airfoils and wings," *Aerospace Science and Technology*, vol. 42, pp. 415–428, 2015.

[30] D. Lee, Y.-E. Kang, D.-H. Kim, and K. Yee, "Aeroelastic design and comprehensive analysis of composite rotor blades through cluster-based kriging," *AIAA Journal*, vol. 60, no. 10, pp. 5984–6004, 2022.

[31] S. Chae, K. Yee, C. Yang, T. Aoyama, S. Jeong, and S. Obayashi, "Helicopter rotor shape optimization for the improvement of aeroacoustic performance in hover," *Journal of Aircraft*, vol. 47, no. 5, pp. 1770–1783, 2010.

[32] W. Song and A. J. Keane, "Surrogate-based aerodynamic shape optimization of a civil aircraft engine nacelle," *AIAA journal*, vol. 45, no. 10, pp. 2565–2574, 2007.

[33] S. Obayashi, S. Jeong, and K. Chiba, "Multi-objective design exploration for aerodynamic configurations," in *35th AIAA fluid dynamics conference and exhibit*, 2005, p. 4666.

[34] S. Obayashi, S. Jeong, K. Chiba, and H. Morino, "Multi-objective design exploration and its application to regional-jet wing design," *Transactions of the Japan Society for Aeronautical and Space Sciences*, vol. 50, no. 167, pp. 1–8, 2007.

[35] M. Kanazaki, K. Tanaka, S. Jeong, and K. Yamamoto, "Multi-objective aerodynamic exploration of elements setting for high-lift airfoil using kriging model," *Journal of Aircraft*, vol. 44, no. 3, pp. 858–864, 2007.

[36] Z. Lyu, G. K. Kenway, and J. R. Martins, "Aerodynamic shape optimization investigations of the common research model wing benchmark," *AIAA journal*, vol. 53, no. 4, pp. 968–985, 2015.

[37] W. Chen, "Data-driven geometric design space exploration and design synthesis," Ph.D. dissertation, University of Maryland, College Park, 2019.

[38] J. Li and M. Zhang, "On deep-learning-based geometric filtering in aerodynamic shape optimization," *Aerospace Science and Technology*, vol. 112, p. 106603, 2021.

[39] B. Wang and T. Chen, "Gaussian process regression with multiple response variables," *Chemometrics and Intelligent Laboratory Systems*, vol. 142, pp. 159–165, 2015.

[40] Z. Han, Y. Liu, J. Zhao, and W. Wang, "Real time prediction for converter gas tank levels based on multi-output least square support vector regressor," *Control Engineering Practice*, vol. 20, no. 12, pp. 1400–1409, 2012.

[41] D. Kuznar, M. Mozina, and I. Bratko, "Curve prediction with kernel regression," in *Proceedings of the 1st workshop on learning from multi-label data*, 2009, pp. 61–68.

[42] A. J. Burnham, J. F. MacGregor, and R. Viveros, "Latent variable multivariate regression modeling," *Chemometrics and Intelligent Laboratory Systems*, vol. 48, no. 2, pp. 167–180, 1999.

[43] T. Murata, K. Fukami, and K. Fukagata, "Nonlinear mode decomposition

with convolutional neural networks for fluid dynamics," *Journal of Fluid Mechanics*, vol. 882, p. A13, 2020.

[44] K. Fukami, K. Fukagata, and K. Taira, "Super-resolution reconstruction of turbulent flows with machine learning," *Journal of Fluid Mechanics*, vol. 870, pp. 106–120, 2019.

[45] K. Fukami, Y. Nabae, K. Kawai, and K. Fukagata, "Synthetic turbulent inflow generator using machine learning," *Physical Review Fluids*, vol. 4, no. 6, p. 064603, 2019.

[46] C.-A. Cheng and B. Boots, "Variational inference for gaussian process models with linear complexity," *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[47] H. Wang, B. van Stein, M. Emmerich, and T. Bäck, "Time complexity reduction in efficient global optimization using cluster kriging," in *Proceedings of the Genetic and Evolutionary Computation Conference*, 2017, pp. 889–896.

[48] Y. Gal *et al.*, "Uncertainty in deep learning," 2016.

[49] B. Lakshminarayanan, A. Pritzel, and C. Blundell, "Simple and scalable predictive uncertainty estimation using deep ensembles," *Advances in neural information processing systems*, vol. 30, 2017.

[50] J. Fernández, M. Chiachío, J. Chiachío, R. Muñoz, and F. Herrera, "Uncertainty quantification in neural networks by approximate bayesian computation: Application to fatigue in composite materials," *Engineering Applications of Artificial Intelligence*, vol. 107, p. 104511, 2022.

[51] D. Zhang, L. Lu, L. Guo, and G. E. Karniadakis, "Quantifying total uncertainty in physics-informed neural networks for solving forward and inverse stochastic problems," *Journal of Computational Physics*, vol. 397, p. 108850, 2019.

[52] A. H. Ibrahim and S. N. Tiwari, "A variational method in design optimization and sensitivity analysis for aerodynamic applications," *Engineering Computations*, vol. 20, no. 1, pp. 88–95, 2004.

[53] V. Sekar, M. Zhang, C. Shu, and B. C. Khoo, "Inverse design of airfoil using a deep convolutional neural network," *AIAA Journal*, vol. 57, no. 3, pp. 993–1003, 2019.

[54] S. A. Renganathan, R. Maulik, and J. Ahuja, "Enhanced data efficiency using deep neural networks and gaussian processes for aerodynamic design optimization," *Aerospace Science and Technology*, vol. 111, p. 106522, 2021.

[55] K. Daneshkhah and W. Ghaly, "Aerodynamic inverse design for viscous flow in turbomachinery blading," *Journal of Propulsion and Power*, vol. 23, no. 4, pp. 814–820, 2007.

[56] Z. Li and X. Zheng, "Review of design optimization methods for turbomachinery aerodynamics," *Progress in Aerospace Sciences*, vol. 93, pp. 1–23, 2017.

[57] K. Lane and D. Marshall, "Inverse airfoil design utilizing cst parameterization," in *48th AIAA Aerospace Sciences Meeting Including the New Horizons Forum and Aerospace Exposition*. American Institute of Aeronautics and Astronautics, 2010.

[58] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural networks*, vol. 2, no. 5, pp. 359–366, 1989.

[59] A. Kharal and A. Saleem, "Neural networks based airfoil generation for a given using bezier-parsec parameterization," *Aerospace Science and Technology*, vol. 23, no. 1, pp. 330–344, 2012.

[60] X. Wang, S. Wang, J. Tao, G. Sun, and J. Mao, "A pca-ann-based inverse design model of stall lift robustness for high-lift device," *Aerospace Science and Technology*, vol. 81, pp. 272–283, 2018.

[61] S. Obayashi and S. Takanashi, "Genetic optimization of target pressure distributions for inverse design methods," *AIAA Journal*, vol. 34, no. 4, pp. 881–886, 1996.

[62] H. J. Kim and O. H. Rho, "Aerodynamic design of transonic wings using the target pressure optimization approach," *Journal of Aircraft*, vol. 35, no. 4, pp. 671–677, 1998.

[63] Y. Zhu, Y. Ju, and C. Zhang, "Proper orthogonal decomposition assisted inverse design optimisation method for the compressor cascade airfoil," *Aerospace Science and Technology*, vol. 105, p. 105955, 2020.

[64] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[65] A. R. Barron, "Universal approximation bounds for superpositions of a sigmoidal function," *IEEE Transactions on Information theory*, vol. 39, no. 3, pp. 930–945, 1993.

[66] X. Meng, H. Babaee, and G. E. Karniadakis, "Multi-fidelity bayesian neural networks: Algorithms and applications," *Journal of Computational Physics*, vol. 438, p. 110361, 2021.

[67] A. L. Maas, A. Y. Hannun, A. Y. Ng *et al.*, "Rectifier nonlinearities improve neural network acoustic models," in *Proc. icml*, vol. 30, no. 1. Atlanta, Georgia, USA, 2013, p. 3.

[68] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *nature*, vol. 323, no. 6088, pp. 533–536, 1986.

[69] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization," *Journal of Machine Learning Research*, vol. 12, pp. 2121–2159, 2011.

[70] T. Tieleman, G. Hinton *et al.*, "Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude," *COURSERA: Neural networks for machine learning*, vol. 4, no. 2, pp. 26–31, 2012.

[71] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[72] P. Ghosh, M. S. Sajjadi, A. Vergari, M. Black, and B. Schölkopf, "From variational to deterministic autoencoders," *arXiv preprint arXiv:1903.12436*, 2019.

[73] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, "Extracting and composing robust features with denoising autoencoders," in *Proceedings of the 25th international conference on Machine learning*, 2008, pp. 1096–1103.

[74] C. Doersch, "Tutorial on variational autoencoders," *arXiv preprint arXiv:1606.05908*, 2016.

[75] I. Higgins, L. Matthey, A. Pal, C. Burgess, X. Glorot, M. Botvinick, S. Mohamed, and A. Lerchner, "beta-vae: Learning basic visual concepts with a constrained variational framework," 2016.

[76] A. Pati and A. Lerch, "Attribute-based regularization of latent spaces for variational auto-encoders," *Neural Computing and Applications*, vol. 33, no. 9, pp. 4429–4444, 2021.

[77] R. Tibshirani, "Regression shrinkage and selection via the lasso," *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 58, no. 1, pp. 267–288, 1996.

[78] B. Settles, *Active Learning*, ser. Synthesis Lectures on Artificial Intelligence and Machine Learning, 2012, vol. 6, no. 1.

[79] X. Yang, X. Cheng, Z. Liu, and T. Wang, "A novel active learning method for profust reliability analysis based on the kriging model," *Engineering with Computers*, 2021.

[80] G. Pinto, R. Messina, H. Li, T. Hong, M. S. Piscitelli, and A. Capozzoli, "Sharing is caring: An extensive analysis of parameter-based transfer learning for the prediction of building thermal dynamics," *Energy and Buildings*, vol. 276, p. 112530, 2022.

[81] C. Tan, F. Sun, T. Kong, W. Zhang, C. Yang, and C. Liu, "A survey on deep transfer learning," in *Artificial Neural Networks and Machine Learning–ICANN 2018: 27th International Conference on Artificial Neu-*

ral Networks, Rhodes, Greece, October 4-7, 2018, Proceedings, Part III 27. Springer, 2018, pp. 270–279.

[82] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 10, pp. 1345–1359, 2010.

[83] Y. Li, W. Jiang, G. Zhang, and L. Shu, "Wind turbine fault diagnosis based on transfer learning and convolutional autoencoder with small-scale data," *Renewable Energy*, vol. 171, pp. 103–115, 2021.

[84] B. Moghadassian and A. Sharma, "Designing wind turbine rotor blades to enhance energy capture in turbine arrays," *Renewable Energy*, vol. 148, pp. 651–664, 2020.

[85] L. E. Kollar and R. Mishra, "Inverse design of wind turbine blade sections for operation under icing conditions," *Energy Conversion and Management*, vol. 180, pp. 844–858, 2019.

[86] B. Moghadassian and A. Sharma, "Inverse design of single-and multi-rotor horizontal axis wind turbine blades using computational fluid dynamics," *Journal of Solar Energy Engineering*, vol. 140, no. 2, p. 021003, 2018.

[87] M. Drela, "Xfoil: an analysis and design system for low reynolds number airfoils," in *Low Reynolds number aerodynamics*. Springer, 1989, pp. 1–12.

[88] A. Ceruti, "Meta-heuristic multidisciplinary design optimization of wind turbine blades obtained from circular pipes," *Engineering with Computers*, vol. 35, no. 2, pp. 363–379, 2018.

[89] W. J. Zhu, W. Z. Shen, and J. N. Sørensen, "Integrated airfoil and blade design method for large wind turbines," *Renewable Energy*, vol. 70, pp. 172–183, 2014.

[90] S. Mohammadi, M. Hassanalian, H. Arionfard, and S. Bakhtiyarov, "Optimal design of hydrokinetic turbine for low-speed water flow in golden gate strait," *Renewable Energy*, vol. 150, pp. 147–155, 2020.

[91] E. Tandis and E. Assareh, "Inverse design of airfoils via an intelligent hybrid optimization technique," *Engineering with Computers*, vol. 33, no. 2, pp. 361–374, 2017.

[92] S. Barone, "Gear geometric design by b-spline curve fitting and sweep surface modelling," *Engineering Computations*, vol. 17, no. 1, pp. 66–74, 2001.

[93] Y. Li, K. Wei, W. Yang, and Q. Wang, "Improving wind turbine blade based on multi-objective particle swarm optimization," *Renewable Energy*, vol. 161, pp. 525–542, 2020.

[94] F. Grasso, "Usage of numerical optimization in wind turbine airfoil design," *Journal of Aircraft*, vol. 48, no. 1, pp. 248–255, 2011.

[95] H. Hamad and A. Al-Smadi, "Space partitioning in engineering design via metamodel acceptance score distribution," *Engineering Computations*, vol. 23, no. 2, pp. 175–185, 2007.

[96] L. Liu, H. Moayedi, A. S. A. Rashid, S. S. A. Rahman, and H. Nguyen, "Optimizing an ann model with genetic algorithm (ga) predicting load-settlement behaviours of eco-friendly raft-pile foundation (erp) system," *Engineering with Computers*, vol. 36, pp. 421–433, 2019.

[97] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: Nsga-ii," *IEEE Transactions on Evolutionary Computation*, vol. 6, pp. 182–197, 2002.

[98] J. Blank and K. Deb, "Pymoo: Multi-objective optimization in python," *IEEE Access*, vol. 8, pp. 89 497–89 509, 2020.

[99] M. Vinokur, "On one-dimensional stretching functions for finite-difference calculations," *Journal of Computational Physics*, vol. 50, pp. 215–234, 1983.

[100] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," *arXiv preprint arXiv:1502.01852*, 2015.

[101] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga *et al.*, "Pytorch: An imperative style, high-performance deep learning library," *Advances in neural information processing systems*, vol. 32, 2019.

[102] N. G. Verhaagen, "Leading-edge radius effects on aerodynamic characteristics of 50-degree delta wings," *Journal of Aircraft*, vol. 49, pp. 521–531, 2012.

[103] S. McKinley and M. Levine, "Cubic spline interpolation," *College of the Redwoods*, vol. 45, no. 1, pp. 1049–1060, 1998.

[104] H.-M. Gutmann, "A radial basis function method for global optimization," *Journal of Global Optimization*, vol. 19, no. 3, pp. 201–227, 2001.

[105] C. E. Rasmussen, "Gaussian processes in machine learning," in *Summer school on machine learning*. Springer, 2003, pp. 63–71.

[106] E. Schulz, M. Speekenbrink, and A. Krause, "A tutorial on gaussian process regression: Modelling, exploring, and exploiting functions," *Journal of Mathematical Psychology*, vol. 85, pp. 1–16, 2018.

[107] L. Sirovich, "Turbulence and the dynamics of coherent structures. i. coherent structures," *Quarterly of Applied Mathematics*, vol. 45, no. 3, pp. 561–571, 1987.

[108] D. Amsallem, M. J. Zahr, and C. Farhat, "Nonlinear model order reduction based on local reduced-order bases," *International Journal for Numerical Methods in Engineering*, vol. 92, no. 10, pp. 891–916, 2012.

[109] D. J. Lucia, P. I. King, and P. S. Beran, "Domain decomposition for reduced-order modeling of a flow with moving shocks," *AIAA Journal*, vol. 40, no. 11, pp. 2360–2362, 2002.

[110] Y.-E. Kang, S. Shon, and K. Yee, "Local non-intrusive reduced order modeling based on soft clustering and classification algorithm," *International Journal for Numerical Methods in Engineering*, 2022.

[111] R. Dupuis, J.-C. Jouhaud, and P. Sagaut, "Surrogate modeling of aerodynamic simulations for multiple operating conditions using machine learning," *AIAA Journal*, vol. 56, no. 9, pp. 3622–3635, 2018.

[112] M. A. Kramer, "Nonlinear principal component analysis using autoassociative neural networks," *AIChE Journal*, vol. 37, no. 2, pp. 233–243, 1991.

[113] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *science*, vol. 313, no. 5786, pp. 504–507, 2006.

[114] M. Milano and P. Koumoutsakos, "Neural network modeling for near wall turbulent flow," *Journal of Computational Physics*, vol. 182, no. 1, pp. 1–26, 2002.

[115] H. Eivazi, H. Veisi, M. H. Naderi, and V. Esfahanian, "Deep neural networks for nonlinear model order reduction of unsteady flows," *Physics of Fluids*, vol. 32, no. 10, p. 105104, 2020.

[116] J. Zhang and X. Zhao, "Machine-learning-based surrogate modeling of aerodynamic flow around distributed structures," *AIAA Journal*, vol. 59, no. 3, pp. 868–879, 2021.

[117] K. Hasegawa, K. Fukami, T. Murata, and K. Fukagata, "Machine-learning-based reduced-order modeling for unsteady flows around bluff bodies of various shapes," *Theoretical and Computational Fluid Dynamics*, vol. 34, no. 4, pp. 367–383, 2020.

[118] J. Wang, C. He, R. Li, H. Chen, C. Zhai, and M. Zhang, "Flow field prediction of supercritical airfoils via variational autoencoder based deep learning framework," *Physics of Fluids*, vol. 33, no. 8, p. 086108, 2021.

[119] N. T. Mücke, S. M. Bohté, and C. W. Oosterlee, "Reduced order modeling for parameterized time-dependent PDEs using spatially and memory aware deep learning," *Journal of Computational Science*, vol. 53, p. 101408, 2021.

[120] P. Wu, S. Gong, K. Pan, F. Qiu, W. Feng, and C. Pain, "Reduced order model using convolutional auto-encoder with self-attention," *Physics of Fluids*, vol. 33, no. 7, p. 077107, 2021.

[121] R. Maulik, B. Lusch, and P. Balaprakash, "Reduced-order modeling of

advection-dominated systems with recurrent neural networks and convolutional autoencoders," *Physics of Fluids*, vol. 33, no. 3, p. 037106, 2021.

[122] A. Gruber, M. Gunzburger, L. Ju, and Z. Wang, "A comparison of neural network architectures for data-driven reduced-order modeling," *Computer Methods in Applied Mechanics and Engineering*, vol. 393, p. 114764, 2022.

[123] T. Kadeethum, F. Ballarin, Y. Choi, D. O'Malley, H. Yoon, and N. Bouklas, "Non-intrusive reduced order modeling of natural convection in porous media using convolutional autoencoders: comparison with linear subspace techniques," *Advances in Water Resources*, p. 104098, 2022.

[124] Y. Burda, R. Grosse, and R. Salakhutdinov, "Importance weighted autoencoders," *arXiv preprint arXiv:1509.00519*, 2015.

[125] C. P. Burgess, I. Higgins, A. Pal, L. Matthey, N. Watters, G. Desjardins, and A. Lerchner, "Understanding disentangling in *beta*-vae," *arXiv preprint arXiv:1804.03599*, 2018.

[126] H. Eivazi, S. Le Clainche, S. Hoyas, and R. Vinuesa, "Towards extraction of orthogonal and parsimonious non-linear modes from turbulent flows," *Expert Systems with Applications*, p. 117038, 2022.

[127] C. K. Sønderby, T. Raiko, L. Maaløe, S. K. Sønderby, and O. Winther, "Ladder variational autoencoders," *Advances in Neural Information Processing Systems*, vol. 29, 2016.

[128] S. H. Park and J. H. Kwon, "Implementation of kw turbulence models in an implicit multigrid method," *AIAA journal*, vol. 42, no. 7, pp. 1348–1357, 2004.

[129] Y. Hong, D. Lee, K. Yee, and S. H. Park, "Enhanced high-order scheme for high-resolution rotorcraft flowfield analysis," *AIAA Journal*, vol. 60, no. 1, pp. 144–159, 2022.

[130] S. Yang and K. Yee, "Comment on "novel approach for selecting low-fidelity scale factor in multifidelity metamodeling"," *AIAA Journal*, vol. 60, no. 4, pp. 2713–2715, 2022.

[131] J. Herman and W. Usher, "SALib: An open-source python library for sensitivity analysis," *The Journal of Open Source Software*, vol. 2, no. 9, jan 2017. [Online]. Available: https://doi.org/10.21105/joss.00097

[132] S. Yang and K. Yee, "Towards quantifying calibrated uncertainty via deep ensembles in multi-output regression task," *arXiv preprint arXiv:2303.16210*, 2023.

[133] H. Zhang, W. W. Chen, A. Iyer, D. W. Apley, and W. Chen, "Uncertainty-aware mixed-variable machine learning for materials design," *Scientific reports*, vol. 12, no. 1, pp. 1–13, 2022.

[134] D. R. Jones, M. Schonlau, and W. J. Welch, "Efficient global optimization of expensive black-box functions," *Journal of Global Optimization*, vol. 13, no. 4, pp. 455–492, 1998.

[135] J. Snoek, H. Larochelle, and R. P. Adams, "Practical bayesian optimization of machine learning algorithms," *Advances in neural information processing systems*, vol. 25, 2012.

[136] S. Shin, Y. Lee, M. Kim, J. Park, S. Lee, and K. Min, "Deep neural network model with bayesian hyperparameter optimization for prediction

of nox at transient conditions in a diesel engine," *Engineering Applications of Artificial Intelligence*, vol. 94, p. 103761, 2020.

[137] K. Shimoyama, K. Sato, S. Jeong, and S. Obayashi, "Updating kriging surrogate models based on the hypervolume indicator in multi-objective optimization," *Journal of Mechanical Design*, vol. 135, no. 9, p. 094503, 2013.

[138] S. Rhode, "Non-stationary gaussian process regression applied in validation of vehicle dynamics models," *Engineering Applications of Artificial Intelligence*, vol. 93, p. 103716, 2020.

[139] B. Wang, W. Wang, Z. Qiao, G. Meng, and Z. Mao, "Dynamic selective gaussian process regression for forecasting temperature of molten steel in ladle furnace," *Engineering Applications of Artificial Intelligence*, vol. 112, p. 104892, 2022.

[140] H. Yang, S. H. Hong, R. ZhG, and Y. Wang, "Surrogate-based optimization with adaptive sampling for microfluidic concentration gradient generator design," *RSC advances*, vol. 10, no. 23, pp. 13 799–13 814, 2020.

[141] N. Quirante, J. Javaloyes-Antón, and J. A. Caballero, "Hybrid simulation-equation based synthesis of chemical processes," *Chemical Engineering Research and Design*, vol. 132, pp. 766–784, 2018.

[142] N. Quirante and J. A. Caballero, "Optimization of a sour water stripping plant using surrogate models," in *Computer Aided Chemical Engineering*. Elsevier, 2016, vol. 38, pp. 31–36.

[143] T. Keßler, C. Kunde, K. McBride, N. Mertens, D. Michaels, K. Sundmacher, and A. Kienle, "Global optimization of distillation columns using

explicit and implicit surrogate models," *Chemical Engineering Science*, vol. 197, pp. 235–245, 2019.

[144] W. Zhong, C. Qiao, X. Peng, Z. Li, C. Fan, and F. Qian, "Operation optimization of hydrocracking process based on kriging surrogate model," *Control Engineering Practice*, vol. 85, pp. 34–40, 2019.

[145] K. Sugimura, S. Jeong, S. Obayashi, and T. Kimura, "Kriging-model-based multi-objective robust optimization and trade-off rule mining of a centrifugal fan with dimensional uncertainty," *Journal of computational science and technology*, vol. 3, no. 1, pp. 196–211, 2009.

[146] K. Park, J. Jung, and S. Jeong, "Multi-objective shape optimization of airfoils for mars exploration aircraft propellers," *International Journal of Aeronautical and Space Sciences*, pp. 1–15, 2022.

[147] J. Muñoz, B. López, F. Quevedo, S. Garrido, C. A. Monje, and L. E. Moreno, "Gaussian processes and fast marching square based informative path planning," *Engineering Applications of Artificial Intelligence*, vol. 121, p. 106054, 2023.

[148] B. S. Yıldız, "Slime mould algorithm and kriging surrogate model-based approach for enhanced crashworthiness of electric vehicles," *International Journal of Vehicle Design*, vol. 83, no. 1, pp. 54–68, 2020.

[149] ——, "Marine predators algorithm and multi-verse optimisation algorithm for optimal battery case design of electric vehicles," *International Journal of Vehicle Design*, vol. 88, no. 1, pp. 1–11, 2022.

[150] N. Namura, S. Obayashi, and S. Jeong, "Efficient global optimization of

vortex generators on a supercritical infinite wing," *Journal of Aircraft*, vol. 53, no. 6, pp. 1670–1679, 2016.

[151] D. J. MacKay, "Information-based objective functions for active data selection," *Neural computation*, vol. 4, no. 4, pp. 590–604, 1992.

[152] ——, "Probable networks and plausible predictions-a review of practical bayesian methods for supervised neural networks," *Network: computation in neural systems*, vol. 6, no. 3, p. 469, 1995.

[153] J. Fernández, J. Chiachío, M. Chiachío, J. Barros, and M. Corbetta, "Physics-guided bayesian neural networks by abc-ss: Application to reinforced concrete columns," *Engineering Applications of Artificial Intelligence*, vol. 119, p. 105790, 2023.

[154] Y. Gal and Z. Ghahramani, "Dropout as a bayesian approximation: Representing model uncertainty in deep learning," in *international conference on machine learning*. PMLR, 2016, pp. 1050–1059.

[155] T. Deruyttere, V. Milewski, and M.-F. Moens, "Giving commands to a self-driving car: How to deal with uncertain situations?" *Engineering applications of artificial intelligence*, vol. 103, p. 104257, 2021.

[156] I. Osband, "Risk versus uncertainty in deep learning: Bayes, bootstrap and the dangers of dropout," in *NIPS workshop on bayesian deep learning*, vol. 192, 2016.

[157] J. Hron, A. G. d. G. Matthews, and Z. Ghahramani, "Variational gaussian dropout is not bayesian," *arXiv preprint arXiv:1711.02989*, 2017.

[158] J. Hron, A. Matthews, and Z. Ghahramani, "Variational bayesian

dropout: pitfalls and fixes," in *International Conference on Machine Learning*. PMLR, 2018, pp. 2019–2028.

[159] L. L. Folgoc, V. Baltatzis, S. Desai, A. Devaraj, S. Ellis, O. E. M. Manzanera, A. Nair, H. Qiu, J. Schnabel, and B. Glocker, "Is mc dropout bayesian?" *arXiv preprint arXiv:2110.04286*, 2021.

[160] F. K. Gustafsson, M. Danelljan, and T. B. Schon, "Evaluating scalable bayesian deep learning methods for robust computer vision," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, 2020, pp. 318–319.

[161] S. Fort, H. Hu, and B. Lakshminarayanan, "Deep ensembles: A loss landscape perspective," *arXiv preprint arXiv:1912.02757*, 2019.

[162] Y. Ovadia, E. Fertig, J. Ren, Z. Nado, D. Sculley, S. Nowozin, J. Dillon, B. Lakshminarayanan, and J. Snoek, "Can you trust your model's uncertainty? evaluating predictive uncertainty under dataset shift," *Advances in neural information processing systems*, vol. 32, 2019.

[163] A. Ashukha, A. Lyzhov, D. Molchanov, and D. Vetrov, "Pitfalls of in-domain uncertainty estimation and ensembling in deep learning," *arXiv preprint arXiv:2002.06470*, 2020.

[164] A. G. Wilson and P. Izmailov, "Bayesian deep learning and a probabilistic perspective of generalization," *Advances in neural information processing systems*, vol. 33, pp. 4697–4708, 2020.

[165] R. Rahaman *et al.*, "Uncertainty quantification and deep ensembles," *Advances in Neural Information Processing Systems*, vol. 34, pp. 20 063–20 075, 2021.

[166] X. Wu and M. Gales, "Should ensemble members be calibrated?" *arXiv preprint arXiv:2101.05397*, 2021.

[167] J. de Becdelievre and I. Kroo, "A bayesian approach to collaborative optimization with application to tailless aircraft range maximization," in *AIAA AVIATION 2021 FORUM*, 2021, p. 3063.

[168] S. Pawar, O. San, P. Vedula, A. Rasheed, and T. Kvamsdal, "Multi-fidelity information fusion with concatenated neural networks," *Scientific Reports*, vol. 12, no. 1, p. 5900, 2022.

[169] M. Pocevičiūtė, G. Eilertsen, S. Jarkman, and C. Lundström, "Generalisation effects of predictive uncertainty estimation in deep learning for digital pathology," *Scientific Reports*, vol. 12, no. 1, pp. 1–15, 2022.

[170] E. Ilg, O. Cicek, S. Galesso, A. Klein, O. Makansi, F. Hutter, and T. Brox, "Uncertainty estimates and multi-hypotheses networks for optical flow," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 652–667.

[171] J. Linmans, J. van der Laak, and G. Litjens, "Efficient out-of-distribution detection in digital pathology using multi-head convolutional neural networks." in *MIDL*, 2020, pp. 465–478.

[172] R. Egele, R. Maulik, K. Raghavan, B. Lusch, I. Guyon, and P. Balaprakash, "Autodeuq: Automated deep ensemble with uncertainty quantification," in *2022 26th International Conference on Pattern Recognition (ICPR)*. IEEE, 2022, pp. 1908–1914.

[173] R. Maulik, R. Egele, K. Raghavan, and P. Balaprakash, "Quantifying uncertainty for deep learning based forecasting and flow-reconstruction using

neural architecture search ensembles," *arXiv preprint arXiv:2302.09748*, 2023.

[174] O. Solopchuk and A. Zénon, "Active sensing with artificial neural networks," *Neural Networks*, vol. 143, pp. 751–758, 2021.

[175] M.-H. Laves, S. Ihler, J. F. Fast, L. A. Kahrs, and T. Ortmaier, "Recalibration of aleatoric and epistemic regression uncertainty in medical imaging," *arXiv preprint arXiv:2104.12376*, 2021.

[176] T. Gneiting and A. E. Raftery, "Strictly proper scoring rules, prediction, and estimation," *Journal of the American statistical Association*, vol. 102, no. 477, pp. 359–378, 2007.

[177] T. Hastie, R. Tibshirani, J. H. Friedman, and J. H. Friedman, *The elements of statistical learning: data mining, inference, and prediction.* Springer, 2009, vol. 2.

[178] A. Kendall and Y. Gal, "What uncertainties do we need in bayesian deep learning for computer vision?" *Advances in neural information processing systems*, vol. 30, 2017.

[179] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger, "On calibration of modern neural networks," in *International conference on machine learning.* PMLR, 2017, pp. 1321–1330.

[180] G. Scalia, C. A. Grambow, B. Pernici, Y.-P. Li, and W. H. Green, "Evaluating scalable uncertainty estimation methods for deep learning-based molecular property prediction," *Journal of chemical information and modeling*, vol. 60, no. 6, pp. 2697–2717, 2020.

[181] S. Hu, N. Pezzotti, and M. Welling, "Learning to predict error for mri reconstruction," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer, 2021, pp. 604–613.

[182] V. Kuleshov, N. Fenner, and S. Ermon, "Accurate uncertainties for deep learning using calibrated regression," in *International conference on machine learning*. PMLR, 2018, pp. 2796–2804.

[183] M. P. Naeini, G. Cooper, and M. Hauskrecht, "Obtaining well calibrated probabilities using bayesian binning," in *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

[184] D. Levi, L. Gispan, N. Giladi, and E. Fetaya, "Evaluating and calibrating uncertainty prediction in regression tasks," *Sensors*, vol. 22, no. 15, p. 5540, 2022.

[185] B. Phan, R. Salay, K. Czarnecki, V. Abdelzad, T. Denouden, and S. Vernekar, "Calibrating uncertainties in object localization task," *arXiv preprint arXiv:1811.11210*, 2018.

[186] B. Zadrozny and C. Elkan, "Obtaining calibrated probability estimates from decision trees and naive bayesian classifiers," in *Icml*, vol. 1. Citeseer, 2001, pp. 609–616.

[187] ——, "Transforming classifier scores into accurate multiclass probability estimates," in *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, 2002, pp. 694–699.

[188] C. K. Williams and C. E. Rasmussen, *Gaussian processes for machine learning*. MIT press Cambridge, MA, 2006, vol. 2, no. 3.

[189] M. A. Alvarez, L. Rosasco, N. D. Lawrence *et al.*, "Kernels for vector-valued functions: A review," *Foundations and Trends® in Machine Learning*, vol. 4, no. 3, pp. 195–266, 2012.

[190] Q. Lin, J. Hu, Q. Zhou, Y. Cheng, Z. Hu, I. Couckuyt, and T. Dhaene, "Multi-output gaussian process prediction for computationally expensive problems with multiple levels of fidelity," *Knowledge-Based Systems*, vol. 227, p. 107151, 2021.

[191] Q. Lin, J. Hu, L. Zhang, P. Jin, Y. Cheng, and Q. Zhou, "Gradient-enhanced multi-output gaussian process model for simulation-based engineering design," *AIAA Journal*, vol. 60, no. 1, pp. 76–91, 2022.

[192] N. Öztürk, A. R. Yıldız, N. Kaya, and F. Öztürk, "Neuro-genetic design optimization framework to support the integrated robust design optimization process in ce," *Concurrent Engineering*, vol. 14, no. 1, pp. 5–16, 2006.

[193] I. Osband, C. Blundell, A. Pritzel, and B. Van Roy, "Deep exploration via bootstrapped dqn," *Advances in neural information processing systems*, vol. 29, 2016.

[194] T. Pearce, N. Anastassacos, M. Zaki, and A. Neely, "Bayesian inference with anchored ensembles of neural networks, and application to exploration in reinforcement learning," *arXiv preprint arXiv:1805.11324*, 2018.

[195] C. Riquelme, G. Tucker, and J. Snoek, "Deep bayesian bandits showdown: An empirical comparison of bayesian deep networks for thompson sampling," *arXiv preprint arXiv:1802.09127*, 2018.

[196] D. Liu, Y. Tan, E. Khoram, and Z. Yu, "Training deep neural networks

for the inverse design of nanophotonic structures," *Acs Photonics*, vol. 5, no. 4, pp. 1365–1369, 2018.

[197] S. Kim, M. Jwa, S. Lee, S. Park, and N. Kang, "Deep learning-based inverse design for engineering systems: multidisciplinary design optimization of automotive brakes," *Structural and Multidisciplinary Optimization*, vol. 65, no. 11, p. 323, 2022.

국문 초록

데이터 기반 기법을 통한 공력 설계 효율화

양선웅
서울대학교 대학원
항공우주공학과

최근 고성능 컴퓨팅 기술의 발달로 높은 계산 비용을 요구하는 공기역학 설계 프로세스의 가속화에 대한 관심이 높아지고 있다. 하지만 계산 기술 및 시뮬레이션 기법의 발전에도 불구하고 공기역학 설계 프로세스의 계산 비용은 여전히 큰 걸림돌로 남아 있다. 해당 프로세스의 반복적인 특성과 고충실도 전산 유체 역학 해석자의 필요성은 짧은 설계 수행 시간의 달성을 어렵게 만들고 창의적인 설계 탐색을 방해한다. 그럼에도 불구하고 성공적으로 설계된 공기역학적 물체의 높은 가치 덕분에 엔지니어들은 정확한 유동 해석에 자원을 투자하고 있다. 이러한 문제를 해결하기 위해 그들은 여러 가지 기법을 연구해 왔으며, 주어진 데이터로부터 학습된 회귀 모델은 높은 비용이 요구되는 유동 시뮬레이션이나 실험을 대체할 수 있는 방법으로서 인기를 얻고 있다. 회귀 모델은 고성능 컴퓨팅의 남용을 방지하고 설계 프로세스 효율성을 개선하는 데 도움이 되며, 성능 값 예측, 설계 최적화, 지식 추출 등을 목적으로 공학 분야에서 광범위하게 사용되고 있다.

하지만 공기 역학 설계 프로세스에서 회귀 모델의 활용을 방해하는 몇 가지 장애물이 있다. 첫째, 공기역학 설계에는 고차원의 입력 공간이 필요한 경우가 많은데 이 때 회귀 모델은 차원의 저주에 직면한다. 입력 변수의 수가 증가하면 입력 공간 내에서 가능한 디자인의 경우의 수가 기하급수적으로 증가한다. 이로 인해

사용 가능한 데이터의 수가 고차원 공간을 대표하기에 충분하지 않은 sparsity 문제가 발생한다. 둘째, 널리 사용되는 대부분의 회귀 모델은 단일 출력을 예측하기 위해 고안되었기 때문에 고차원 출력 공간에서는 사용이 제한된다. 다중 출력 회귀 문제에서는 각 출력에 대해 회귀 모델이 독립적으로 학습되므로 필요한 학습 시간이 출력 차원에 따라 선형적으로 증가하고 출력들 간의 상관관계는 전혀 고려되지 않는다. 마지막으로, 신뢰할 수 있고 효율적인 불확실성 정량화 과정이 회귀 모델과 결합되어 공기 역학 설계 프로세스 과정에서 발생할 수 있는 내재적 불확실성이 고려되어야 한다. 회귀 모델의 경우 예측 값이 실제 값과 일치하지 않을 가능성이 항상 존재하기 때문에 예측 값의 신뢰성에 대한 정보를 제공하기 위한 불확실성 정량화 과정이 특히 중요하나 기존의 가우시안 프로세스 회귀, 베이지안 신경망 등의 기법들은 효율성에 있어 큰 한계를 갖는다.

이러한 측면에서 본 논문의 의의는 다음과 같이 요약될 수 있다:

1. 공력 설계 과정에서의 고차원 입력 공간을 차원 축소 기법을 사용하여 저차원 잠재 공간으로 축소하고자 하였고, 축소된 잠재 공간에서의 gradient-free 최적화 기법의 적용 가능성을 확인하고자 하였다. 이를 위해 풍력 터빈 에어포일의 역설계 문제가 사례로서 활용되었다. 결과적으로 딥러닝 기반 차원 축소 기법을 통해 기존의 고차원 입력 공간을 성공적으로 축소할 수 있음을 확인하고, 축소된 차원에 유전 알고리즘을 적용하여 gradient-free 최적화 기법의 활용 가능성 또한 검증하였다.

2. 고차원 데이터의 예측을 위해 차수 감수 모델링 기법을 활용하였다. 이를 위해 잠재 공간을 생성하는 차원 축소 프로세스가 필요한데, 본 연구에서는 차수 감수 모델링 과정에서 잠재 공간이 매개체 역할을 한다는 점에서 잠재 공간은 필연적으로 차수 감수 모델링 성능에 영향을 미친다는 사실에 주목하였다. 이에 따라 천음속 에어포일 주변의 유동장 예측을 연구 사례로 선정하고, 다양한 머신러닝 기반 차원 축소 기법을 바탕으로 구축된 잠재 공간들의 물리적 해석 가능성을 조사하였다. 또한 잠재 공간의 해석가능성이

차수 감수 모델링 성능에 미치는 영향을 종합적으로 분석하고, 최종적으로 차수 감수 모델링을 통한 고차원 출력 공간 예측의 정확성과 효율성에 잠재 공간이 미치는 영향을 확인하였다.

3. 신뢰성 있고 효율적인 불확실성 정량화가 가능한 회귀 모델을 연구하였다. 이를 위해 미사일 형상의 공기역학적 성능을 예측하는 문제에서 딥 앙상블 접근법을 활용하였다. 구체적으로 본 연구는 실제 공학 분야에서 빈번하게 요구되는 다중 출력 회귀 작업에 대해 간단하고 확장이 쉬운 딥 앙상블 방법을 검증하는 것을 목표로 하였다. 검증 과정에서 딥 앙상블을 통해 정량화된 불확실성의 낮은 신뢰도를 발견하였으며, 만족스럽지 못한 불확실성 품질을 보정하기 위해 간단한 사후 보정 방법을 모델에 적용하였다. 결과적으로 공학 분야에서 가장 빈번하게 사용되는 가우시안 프로세스 회귀에 비해 딥 앙상블 기법이 더 신뢰성 있고 효율적인 불확실성 정량화가 가능함을 확인하였고, 끝으로 제안된 보정 기법이 딥 앙상블 기반 베이지안 최적화에 미치는 영향을 살펴보았다.

주요어: 공력 설계, 데이터 기반 기법, 예측 모델링, 제너레이티브 모델링, 차수 감소 모델링, 불확실성 정량화

학번: 2020-34020