Adaptive Data Generation Framework for Machine Learning Models in Nonlinear Structural Dynamics

Von der Fakultät für Bauingenieurwesen
der Rheinisch-Westfälischen Technischen Hochschule Aachen
zur Erlangung des akademischen Grades eines Doktors der Ingenieurwissenschaften
genehmigte Dissertation

vorgelegt von

Pavle Milicevic
aus Ilidža, Bosnien und Herzegowina

Berichter: Universitätsprofessor Dr.-Ing. habil. Sven Klinkel
Universitätsprofessor Dr.-Ing. habil Okyay Altay

Tag der mündlichen Prüfung: 17. April 2026

Diese Dissertation ist auf den Internetseiten der Universitätsbibliothek online verfügbar.

Veröffentlicht als Heft 24 (2026) in der Schriftenreihe
des Lehrstuhls für Baustatik und Baudynamik
der RWTH Aachen

Herausgeber:
Universitätsprofessor Dr.-Ing. habil. Sven Klinkel

Organisation und Verwaltung:
Rheinisch-Westfälische Technische Hochschule Aachen
Fakultät für Bauingenieurwesen
Lehrstuhl für Baustatik und Baudynamik
Mies-van-der-Rohe-Str. 1
52074 Aachen

Telefon: +49 241 80 25088
Telefax: +49 241 80 22303
E-Mail: sekretariat@lbb.rwth-aachen.de

© 2026 Pavle Milicevic

D 82 (Diss. RWTH Aachen University, 2026)

Alle Rechte, insbesondere das der Übersetzung in fremde Sprachen, vorbehalten. Ohne Genehmigung des Autors ist es nicht gestattet, dieses Heft ganz oder teilweise auf fotomechanischem Wege (Fotokopie, Mikrokopie) zu vervielfältigen oder in elektronischen Medien zu speichern.

ISSN 1437-0840
ISBN 978-3-946090-23-6

Acknowledgements

This work would not have been possible without the support and generosity of many people, to whom I am grateful.

I would like to thank Prof. Dr.Ing. Okyay Altay for giving me the opportunity to work with him, for his guidance and support throughout my PhD. I would like to thank Prof. Dr.Ing. Sven Klinkel for trusting me with the position and for making LBB an excellent place to work. Thank you also to Annette, whose help was offered so readily and so often. I am grateful to Prof. Dr.Ing. Jörg Blankenbach for chairing the examination committee.

A particular thank you goes to Prof. Dr. Jin-Song Pei, who introduced me to structural dynamics and trusted me as an undergraduate assistant. Her knowledge, scientific rigor, and generosity set a standard towards which I aspire.

To my colleagues at LBB, thank you for your help, warmth, and laughter. To my friends in Aachen and beyond, thank you for your support and for making wherever I was feel like home.

To my family and everyone who has always felt like the closest circle, thank you for the love and endless support across every distance. To my mother, my father and my brother, on whom I have always relied throughout my journey, from Mostar, Oklahoma and Aachen. You welcomed me with open arms time and again. Thank you for your love, your support and your sacrifice. To Ivana, who always reminded me of what matters. Hvala!

Aachen, April 2026 *Pavle Milicevic*

Abstract

Growing demand for structural vibration control devices, such as tuned mass dampers (TMDs), has increased the need for efficient, data-driven models that enhance design, analysis, and real-time applications including digital twins. Machine learning methods enable fast and accurate modeling of nonlinear dynamic systems, but require high-quality training data that is challenging and costly to obtain in structural dynamics.

Vibration control devices undergo mandatory laboratory testing before deployment, generating extensive experimental data currently used only for verification rather than developing predictive models. This dissertation presents a novel framework for *testing-integrated modeling* that simultaneously verifies device performance and generates high-fidelity data-driven models through adaptive experimental design.

The core contribution is the *Adaptive Data Generation (ADaGen)* framework, which formulates experimental data collection as an optimization problem in the space of experimental parameters. Unlike conventional testing protocols, ADaGen iteratively selects experimental conditions to maximize model accuracy using derivative-free optimization. This addresses the *dynamic sampling problem* inherent in testing devices with auxiliary mass, where desired states cannot be directly specified but must be reached through unknown system dynamics.

Numerical validation using a Duffing oscillator demonstrates that ADaGen generates an informative dataset with 19 experiments compared to 35 experiments required by conventional space-filling methods. Experimental validation employs a tuned liquid column damper prototype with Neural Ordinary Differential Equations, including a novel architecture extension that simultaneously identifies system dynamics and unknown modal mass parameters without direct force measurements. Furthermore, the presented extensions to the baseline framework address real-world constraints through interpretable machine learning initialization for reduced computational cost, cumulative data generation for multi-degree-of-freedom systems, and reference-free adaptive sampling that eliminates the need for comprehensive test datasets.

Kurzfassung

Die wachsende Nachfrage nach Schwingungsdämpfungssystemen für Bauwerke, wie z.B. Schwingungstilger (TMDs), hat den Bedarf an effizienten, datengetriebenen Modellen erhöht, die Entwurf, Analyse und Echtzeitanwendungen einschließlich digitaler Zwillinge verbessern. Methoden des maschinellen Lernens ermöglichen eine schnelle und genaue Modellierung nichtlinearer dynamischer Systeme, benötigen jedoch hochwertige Trainingsdaten, die in der Strukturdynamik schwierig und kostspielig zu erhalten sind.

Schwingungsdämpfungssysteme durchlaufen vor ihrem Einsatz obligatorische Labortests, die umfangreiche experimentelle Daten erzeugen, welche derzeit nur zur Verifikation und nicht zur Entwicklung prädiktiver Modelle verwendet werden. Diese Dissertation präsentiert einen neuartigen Ansatz für die *testintegrierte Modellierung*, der gleichzeitig die Geräteleistung verifiziert und hochpräzise datengetriebene Modelle durch adaptives Versuchsdesign erzeugt.

Der zentrale Beitrag ist das *Adaptive Data Generation (ADaGen)*-Framework, das die experimentelle Datenerfassung als Optimierungsproblem im Raum der experimentellen Parameter formuliert. Im Gegensatz zu konventionellen Testprotokollen wählt ADaGen iterativ experimentelle Bedingungen aus, um die Modellgenauigkeit mittels derivativfreier Optimierung zu maximieren. Dies adressiert das *dynamische Abtastproblem*, das bei Geräten mit Hilfsmasse auftritt. Die gewünschten Zustände werden nicht direkt spezifiziert, sondern müssen durch unbekannte Systemdynamik erreicht werden.

Die numerische Validierung anhand eines Duffing-Oszillators zeigt, dass ADaGen mit 19 Experimenten einen informativen Datensatz erzeugt, verglichen mit 35 Experimenten, die bei konventionellen raumfüllenden Methoden erforderlich sind. Die experimentelle Validierung verwendet einen Prototyp eines abgestimmten Flüssigkeitssäulendämpfers mit neuronalen gewöhnlichen Differentialgleichungen, einschließlich einer neuartigen Architekturerweiterung, die gleichzeitig Systemdynamik und unbekannte modale Massenparameter ohne direkte Kraftmessungen identifiziert. Darüber hinaus adressieren die vorgestellten Erweiterungen des Basis-Frameworks reale Randbedingungen durch interpretierbare Initialisierung des maschinellen Lernens zur Reduzierung der Rechenkosten, kumulative Datengenerierung für Systeme mit mehreren Freiheitsgraden und referenzfreie adaptive Abtastung, die die Notwendigkeit umfassender Testdatensätze eliminiert.

Contents

1 Introduction **1**

1.1 Motivation 1

1.2 State of the Art 2

1.2.1 Data-Driven Modeling in Structural Dynamics 2

1.2.2 Design of Experiments 4

1.2.3 Active Learning 6

1.3 Scope and Objectives 11

1.4 Outline 13

2 System Identification with Neural Networks **15**

2.1 Artificial Neural Networks 16

2.1.1 Artificial Neuron 17

2.1.2 Feedforward Neural Networks 19

2.1.3 Neural Ordinary Differential Equations 22

2.2 Bias-Variance Tradeoff 25

2.2.1 Underfitting and Overfitting 26

2.2.2 Model Complexity 26

2.2.3 Data Quality and Quantity 27

2.2.4 Improving the Bias-Variance Tradeoff 28

2.3 Summary 31

3 Design of Experiments and Active Learning **33**

3.1 Design of Experiments 34

3.1.1 Classical Design of Experiments 34

3.2 Space-Filling Sampling Methods 36

3.2.1 Random Sampling 36

3.2.2 Latin Hypercube Sampling . . . 37
3.2.3 Sobol Sequences . . . 38
3.2.4 Comparison of Space-Filling Methods . . . 39
3.3 Active Learning . . . 40
3.3.1 Active Learning Paradigm . . . 40
3.3.2 Uncertainty-Based Query Strategies . . . 41
3.3.3 Ensemble-Based Approaches for Neural Networks . . . 43
3.4 The Dynamic Sampling Problem . . . 45
3.4.1 Performance-Based Optimization . . . 46
3.5 Summary . . . 47

4 Adaptive Data Generation for Testing-Integrated Modeling 49
4.1 Problem Setup and Scope . . . 50
4.1.1 Mathematical Formulation . . . 52
4.2 Adaptive Data Generation Framework . . . 53
4.3 Training and Test Data Configuration . . . 54
4.3.1 Test Dataset . . . 55
4.3.2 Training Dataset . . . 55
4.3.3 Training Variance Reduction . . . 56
4.4 Pattern Search Optimization . . . 56
4.5 Summary . . . 59

5 Numerical Validation Using FNNs 61
5.1 Duffing Oscillator . . . 61
5.1.1 FNN Configuration . . . 62
5.1.2 Training and Test Datasets . . . 65
5.1.3 Sampling Space . . . 66
5.1.4 Investigated Unsupervised Sampling Methods . . . 69
5.1.5 Adaptive Data Generation Configuration . . . 71
5.1.6 Comparison with Unsupervised Sampling Methods . . . 73
5.2 Two-story Shear Frame with a Nonlinear Tuned Mass Damper . . . 74
5.3 Summary . . . 79

6 Experimental Validation with TLCD Prototype **81**
6.1 Experimental Setup . . . 82
6.2 Sampling Space and Experimental Parameters . . . 84
6.3 Pattern Search Parameters . . . 86
6.4 Extended Neural ODE Formulation for Forced Systems . . . 87
6.4.1 Neural ODE Configuration . . . 90
6.5 Results and Discussion . . . 90
6.5.1 Adaptation Cost Function Landscape . . . 90
6.5.2 Comparison with Batch Sampling Methods . . . 92
6.5.3 Ensemble Variance Reduction and Neural ODE Validation . . . 95
6.5.4 Sensitivity Analysis of Neural ODE Extensions . . . 96
6.5.5 Pattern Search Parameter Sensitivity . . . 100
6.5.6 Comparison with Genetic Algorithm . . . 101
6.6 Application Example . . . 103
6.7 Summary . . . 106

7 Extensions and Generalizations **107**
7.1 Introduction and Motivation . . . 107
7.2 Interpretable Neural Network Initialization . . . 108
7.2.1 The Initialization Problem . . . 108
7.2.2 Interpretable Machine Learning Framework . . . 108
7.2.3 Application to Duffing Oscillator . . . 109
7.2.4 Discussion . . . 112
7.3 Cumulative Data Generation . . . 113
7.3.1 Problem Formulation . . . 113
7.3.2 Cumulative Training Strategy . . . 114
7.3.3 Numerical Validation with SDOF System . . . 115
7.3.4 Integration with Adaptive Sampling Framework . . . 119
7.4 Extension to Multi-Degree-of-Freedom Systems . . . 120
7.4.1 Case Study 1: 2-DOF System with FNN . . . 121
7.4.2 Case Study 2: Neural ODEs for MDOF Systems . . . 123
7.5 Reference-Free Adaptive Sampling . . . 126
7.5.1 Multi-Validation Robust Optimization . . . 130
7.6 Summary . . . 135

8 Conclusions **137**

8.1 Key Contributions 137

8.2 Limitations and Future Directions 138

8.3 Closing Remarks 139

Nomenclature **141**

Bibliography **147**

Chapter 1

Introduction

1.1 Motivation

Growing environmental, economic, and architectural demands in civil engineering have led to increasingly slender, lightweight structures with low inherent damping, making them prone to excessive vibrations under wind and seismic loads. Additionally, the expanding demand for clean energy and rapid growth of wind power have amplified the need for vibration mitigation in wind turbines, which are subjected to multi-hazard dynamic loading and similarly exhibit low inherent damping [82]. These trends necessitate structural vibration control devices that provide supplementary damping and help maintain structural integrity and serviceability. Consequently, continued research on *tuned mass dampers* (TMDs) and their variations, such as *tuned liquid column dampers* (TLCDs), remains essential for civil engineering applications, including skyscrapers and wind turbines. A review of these devices is provided by Altay [8], while application examples can be found in [52].

Effective design and implementation of control devices requires accurate models that capture their dynamic behavior under realistic loading conditions. Traditionally, analyzing structures with installed control devices demands computationally intensive time-domain simulations using large high-fidelity finite element models across multiple loading scenarios. The complex damping mechanisms in these devices introduce strong nonlinearities [60], further increasing computational complexity and requiring accurate representation to simulate the combined structure-device system and determine appropriate design parameters [90, 91] and, when necessary, control strategies [152]. Therefore, efficient and accurate models of control devices are critically needed, which remains a challenging task for traditional physics-based analytical models.

In structural dynamics, experiments are necessary to determine the parameters of nonlinear systems, such as for optimal design of dampers [36] or within the scope of vibration-based identification [28, 79, 121]. Significant effort has been invested in developing experimental setups in civil engineering, such as using large-scale actuators and shaking tables.

With real-time hybrid simulation, these experiments can also be combined with numerical models, such as for investigating soil-structure interaction [37]. However, existing physical testing procedures are limited to pure performance assessment and typically do not result in obtaining a model of the tested specimen.

Structural control devices undergo mandatory laboratory performance testing, both during manufacturing and as required by relevant design codes, before being approved for use in structures [19, 76]. These tests verify that devices meet specified performance criteria such as force-displacement relationships under prescribed excitation protocols, but they do not result in any model of device behavior. Testing is typically conducted by human experts following standardized but repetitive protocols. This presents an opportunity to generate data-driven models in a testing-integrated fashion. However, currently these tests do not produce any versatile model, representing a fundamental gap where extensive experimental data is generated during testing, yet this data remains unexploited for model generation. Subsequent numerical simulations often rely on simplified analytical models with uncertain parameters or on parametric models requiring extensive calibration. If the testing phase could be redesigned to simultaneously verify performance and generate accurate models of the given specimens, it would eliminate redundant modeling efforts, reduce overall costs, and provide higher-fidelity models grounded in physical measurements rather than idealized assumptions.

1.2 State of the Art

1.2.1 Data-Driven Modeling in Structural Dynamics

Vibration control devices have traditionally been modeled using physics-based approaches, where governing equations are derived from first principles and model parameters are calibrated to match experimental observations. For example, viscous dampers are often represented by Kelvin-Voigt or Maxwell models [136], magnetorheological dampers with Bouc-Wen [62], while TMDs are modeled as spring-mass-damper systems with nonlinear restoring forces. However, these approaches face several limitations. First, even well-established models require extensive parameter identification from experimental data. Second, complex nonlinearities such as friction, fluid-structure interaction in TLCDs, or temperature-dependent behavior are difficult to capture with analytical expressions. Third, solving the resulting nonlinear equations often requires iterative numerical schemes like Newton-Raphson methods, which introduce computational overhead and potential convergence issues, limiting their applications in real-time control.

As an alternative to physics-based modeling, data-driven approaches using *machine learning* (ML) methods have gained attention [149]. Machine learning methods are increasingly popular for modeling nonlinear dynamical systems and have been implemented in various applications, such as anti-seismic materials and devices [114]. *Artificial neural networks* (ANNs) in civil engineering have been explored since the early 1990s [5], demonstrating

their potential for capturing complex structural behaviors. The work by Masri et al. [86] demonstrated the application of ANNs to Duffing oscillators, establishing the foundation for neural network-based identification of nonlinear dynamical systems. Rather than imposing predetermined functional forms, data-driven methods directly learn input-output relationships from measured data, allowing them to replicate various response characteristics. These models are suitable for fast and accurate modeling without requiring iterative solution procedures.

Several neural network architectures have proven effective for modeling dynamical systems. Simple *feedforward neural networks* (FNNs) can be trained to learn force-state mappings, directly relating system states such as displacement and velocity to restoring forces [86]. This approach is straightforward and computationally efficient, but requires direct measurement of forces, which can be challenging or impossible in many experimental setups. *Recurrent neural networks* (RNNs) and *long short-term memory networks* (LSTMs) have demonstrated capability in capturing temporal dependencies in dynamic systems [75]. Furthermore, *physics-informed neural networks* (PINNs) have recently emerged, which embed physical laws directly into the training process [115]. The comprehensive review by Cuomo et al. [32] reveals that most PINN research has focused on architecture customization rather than experimental design.

Neural Ordinary Differential Equations (Neural ODEs) [29] have emerged as particularly promising for modeling dynamical systems in engineering applications. They parameterize the right-hand side of an ordinary differential equation using a neural network, allowing them to learn continuous-time dynamics directly from time-series measurements. Unlike discrete-time models such as RNNs, Neural ODEs naturally represent the continuous mathematical framework of dynamics, making them well-suited for physical systems governed by differential equations. For structural dynamics applications, Neural ODEs offer several advantages. They can be trained using only measured system states without requiring direct force measurements, they naturally handle irregular sampling intervals, and they can generate predictions at arbitrary time resolutions. Furthermore, they can be integrated with physics-based constraints, leading to hybrid or gray-box models that combine data-driven flexibility with physical interpretability.

However, as pointed out by Bradley et al. [22], Neural ODEs struggle when the data does not capture critical system dynamics, requiring at least one experiment to contain sufficient information about the system's behavior such as resonance, nonlinear stiffness regimes, or damping characteristics for the model to generalize effectively. This observation underscores the importance of intelligent experimental design. Simply collecting large quantities of data is insufficient if that data does not adequately explore the system's operational space. A common bottleneck for all data-driven approaches is the required amount and quality of data necessary for training. Specifically in structural dynamics applications, generating real response data is difficult and expensive, which motivates the development of efficient data generation methods.

1.2.2 Design of Experiments

One of the major challenges in data-driven modeling is the substantial amount of experimental data needed for traditional supervised training methods. To keep the number of experiments economically feasible, increasing training efficiency is essential. In this context, the *informativeness* of sampled data is a key factor. The need for more efficient data collection has been addressed through *Design of Experiments* (DOE), which originates from statistics and aims to minimize experimental costs while maximizing information gain by parameterizing experiments and optimizing the sampling process [41, 42, 43]. Classical DOE theory was established by Fisher [42], who introduced fundamental concepts such as randomization, replication, and factorial design. Later extensions through optimal design theory [41, 113] provided mathematical frameworks for selecting experimental conditions that maximize the information content of collected data.

DOE methods are broadly classified into *unsupervised (batch) sampling* and *adaptive (sequential) sampling* methods. Batch sampling involves selecting all sampling points in advance before conducting experiments. Common examples include *random sampling*, *Latin hypercube sampling* (LHS), and space-filling sequences like the Sobol sequence [98]. These methods do not utilize information from previous experiments to influence future sampling decisions. While straightforward to implement, batch methods may result in inefficient exploration of the experimental space, particularly when the system behavior is unknown a priori.

In contrast, adaptive sampling methods adjust experimental parameters based on information obtained from previous experiments. This sequential approach allows for more efficient exploration of the experimental space by focusing on areas expected to yield the most valuable data. Adaptive strategies include variance-based and model bias-based methods, among others [44, 98]. Many of these techniques rely on models to predict the next set of sampling parameters. When a model uses its outputs to determine parameters for subsequent experiments, this process is known as *active learning* (AL) [98, 124]. Active learning helps in identifying the most informative data points to sample next, thereby optimizing the data collection process and reducing the total number of experiments required to achieve a desired model accuracy.

Active learning becomes particularly beneficial when data acquisition is feasible but expensive, which is the case for physical experiments in structural dynamics. Shields et al. applied a general AL framework to physical systems, specifically in large-scale boundary layer wind tunnel experiments [127]. Their work demonstrated transformative improvements in the rate of discovery, achieving in 300 experiments what traditional methods theoretically could not accomplish. This study provides strong empirical evidence that AL can fundamentally change how physical experiments are conducted, making it economically viable to explore complex phenomena that would otherwise be prohibitively expensive to investigate.

1.2.2.1 Excitation Signal Design for Dynamic Systems

For dynamic sampling of physical systems, it is necessary to navigate the system through unknown dynamics to explore target states. This can be achieved by designing excitation signals that drive the system across its operational range. In the context of system identification for nonlinear systems, various signal design approaches have been proposed. For linear systems, pseudorandom binary sequences (PRBS) have been widely used [98]. For nonlinear systems, more sophisticated approaches are required, such as amplitude-modulated pseudorandom binary sequences (APRBS) [50], modulated chirp signals [68] or optimized exponential swept-sine signals [101]. These methods aim to excite the system across a wide range of frequencies and amplitudes to capture the full nonlinear behavior.

Extensions of these methods involve sequential signal design [55], where previously measured data is used to enhance the diversity of new data to be collected. This iterative approach adjusts the excitation signal parameters based on analysis of the collected data to improve coverage of the system's operational space. However, these approaches are fundamentally batch methods in that they do not consider the current performance of a trained model when selecting the next experiment. They focus on signal diversity and coverage rather than model accuracy improvement.

Another approach to experimental design for system identification involves using the Fisher information matrix as a criterion for informativeness [53] and combined with the query-by-committee approaches [126], such as in [35]. Hametner et al. addressed the identification of both static and dynamic nonlinear systems using optimal experimental design based on Fisher information. The Fisher information matrix quantifies how much information an experiment is expected to provide about the model parameters. By maximizing a scalar function of this matrix such as its determinant or trace, experiments can be designed to maximize parameter identifiability. However, using the Fisher information matrix requires a predefined model structure and its linearization with respect to parameters, which can be problematic for nonlinear systems. As noted by Nelles [98], this approach assumes that the model structure is correct and that local linearization provides a good approximation, assumptions that may not hold for highly nonlinear systems or black-box models like neural networks.

Recent advances in Bayesian optimization (BO) have extended experimental design capabilities for dynamical systems. Xu et al. [145] developed the EGP-BO algorithm using weighted ensembles of Gaussian process surrogates for stochastic dynamical systems, while Tu et al. [135] applied BO with dynamic search space refinement to battery electro-thermal dynamics. For experimental design specifically, Petsagkourakis and Galvanin [112] developed safe model-based DOE using Gaussian processes with trust-region methods for kinetic models, and Anderson et al. [10] created GP experimental design for chance-constrained control. The classical GP-UCB algorithm by Srinivas et al. [131] provides theoretical no-regret guarantees, with the D-GPUCB variant by Wang et al. [137] achieving provably lower variance while maintaining these guarantees. However, these BO approaches typically require probabilistic surrogate models and uncertainty quan-

tification, which can be computationally expensive for complex dynamic systems. The framework developed in this work employs derivative-free optimization (Pattern Search) with deterministic neural networks, offering broader applicability without the computational overhead of maintaining probabilistic surrogates, while Chapter 7 explores BO-based extensions that eliminate the need for predefined test sets through multi-point cross-validation strategies.

1.2.3 Active Learning

Active learning seeks to enhance data collection by evaluating the current model's performance through measures of informativeness for data acquisition [84]. Multiple AL strategies can be utilized depending on the specific application. These strategies can be broadly classified into three main approaches based on the nature of available data and the associated costs of obtaining it [125]. First, *pool-based sampling*, where the most informative data is identified and labeled from a large pool of existing unlabeled data. This is the most common scenario in machine learning applications where data is abundant but labels are expensive. Second, *stream-based sampling*, where data arrives continuously in a stream and the learner must decide in real-time whether to query the label for each instance or discard it. This is common in online learning scenarios. Third, *query synthesis*, where no data exists initially and the learner must actively generate or request specific data points from scratch. This is the most relevant scenario for physical experiments where experimental conditions must be designed.

Active learning is well-established and widely used in computational studies. Settles provided comprehensive surveys of AL literature and methods [124, 125], establishing theoretical foundations and categorizing different query strategies. More recent work has extended these concepts to deep learning contexts [77, 118], where the challenge of applying AL to neural networks with millions of parameters has been addressed. Benchmark comparisons of different AL strategies for various machine learning tasks have been conducted [147], providing practical guidance on method selection. Kumar and Gupta surveyed AL query strategies across different problem types, including classification, regression, and clustering [72]. However, despite this extensive body of work in computational settings, applying AL to physical systems and experimental environments has proven to be significantly more challenging [127].

In the context of nonlinear dynamic systems, many AL strategies leverage uncertainty quantification, which is a natural output of probabilistic machine learning methods like Gaussian Processes (GPs). Zhao et al. developed an AL method based on GP regression to reduce the number of dissipative particle dynamics simulations needed for multiscale modeling of non-Newtonian fluids [151]. Their approach used the predictive variance from the GP to identify regions of the input space where the model was most uncertain, thereby guiding the selection of new simulation points. Although GPs have been effectively applied in AL due to their ability to provide uncertainty estimates, they are known for scalability

issues with large datasets, as the computational cost grows cubically with the number of data points.

Similarly, Belz et al. utilized unique features of local model networks (LMNs) to guide their AL strategy [16]. LMNs provide local model errors as a natural byproduct of their structure, allowing the identification of regions where the model performs poorly. Their goal-oriented AL approach sampled points in regions with significant local errors, thereby improving the model where it mattered most for the application at hand. This work demonstrated that model-specific features can be leveraged for AL without requiring full probabilistic frameworks.

In the specific context of PINNs, several AL strategies have been developed to improve training efficiency. Nabian et al. introduced importance sampling to boost the training performance of PINNs [97]. Their method identified training points that contributed most to the loss function and adaptively allocated more samples to these regions. Wu et al. conducted a comprehensive study comparing non-adaptive and residual-based adaptive sampling methods for PINNs [142], demonstrating that adaptive sampling can significantly improve convergence rates and final accuracy. Furthermore, Arthurs and King [11] developed an AL algorithm that automatically queries a finite element solver for additional training data in regions where the neural network performs poorly, enabling efficient training data acquisition for solving parametric Navier-Stokes equations. Decke et al. [34] combined PINNs with Graph Neural Networks using physics loss-based query strategies for AL in structural dynamics applications. Although these sampling methods have been effective in their respective applications, they are primarily designed for computational settings where function evaluations are relatively low-cost and the sampling is performed in a virtual space. Consequently, they are not directly suitable for physical experiments where each data point is expensive to obtain and the experimental conditions must be specified through controllable parameters such as excitation signals.

In the context of nonlinear dynamic systems, AL has seen recent theoretical advances establishing strong foundations for dynamical systems identification. Mania et al. [85] demonstrated that AL achieves superior identification rates for nonlinear systems compared to passive approaches. Building on these theoretical foundations, Lundby et al. [80] integrated deep AL directly into nonlinear system identification using neural network ensembles, where ensemble disagreement provides uncertainty quantification to guide data collection. Moreover, data generation in dynamical systems has been explored in control applications. Buisson-Fenet et al. proposed algorithms for AL of Gaussian process dynamics models [24]. Their work addressed the problem of learning the dynamics of an unknown system through active experimentation, using the uncertainty estimates from the GP to plan informative experiments. While their work offers significant improvements over batch data generation methods and is directly relevant to the problem of learning dynamics from experiments, the reliance on GP-based uncertainty quantification limits applicability. Deterministic neural network models, which do not provide intrinsic uncertainty estimates, cannot directly benefit from uncertainty-based AL strategies without

additional modifications such as ensemble methods or Bayesian neural networks, which come with their own computational costs.

1.2.3.1 Surrogate Modeling

Surrogate modeling has successfully integrated AL to reduce computational costs in structural dynamics. Kiani et al. employed pool-based query-by-committee AL to select the most informative ground motion records for training ANNs to predict structural responses [69]. In their approach, a large pool of ground motion records was available, and the challenge was to select which subset to use for computationally expensive response history analyses. By using a committee of neural networks trained on different subsets of data, they could identify ground motions where the committee disagreed most, indicating high uncertainty. Their results showed that this AL strategy significantly reduced the number of required response history analyses while improving the accuracy and stability of prediction models compared to larger randomly selected training datasets. Kapadia et al. [66] developed a method which combines dimensionality reduction with neural networks and uses error estimates to guide where new training data should be sampled. This approach does not require knowledge of governing equations and was validated on various fluid dynamics and nonlinear problems. For structural reliability analysis, Xiao et al. [144] combined multiple surrogate models with AL based on prediction uncertainty, achieving 52% efficiency improvement over standard methods. Similarly, in the aerodynamic context, Wang et al. [138] introduced an adaptive sampling method based on hyper-volume iteration that concentrates samples in regions with large gradients and reduces the number of high-fidelity evaluations. Yuan et al. introduced an AL framework based on the Kriging method and Bayesian inference for structural model updating [148]. They demonstrated that this strategy significantly increased the identification rate while reducing the required sample size by half compared to traditional approaches. The use of Kriging allowed them to build a surrogate model of the expensive finite element simulations and use the surrogate's uncertainty estimates to guide the selection of new parameter combinations to evaluate. However, this work, like the others mentioned, assumes the existence of a physics-based computational model that can be evaluated, which is not the case when the goal is to generate a data-driven model directly from physical experiments without any prior model.

1.2.3.2 Structural Health Monitoring

Active learning has been applied in various areas within the structural dynamics domain, though most applications differ from the query synthesis problem addressed in this dissertation. In Structural Health Monitoring (SHM), data streams are often available from installed sensors, and the challenge is to efficiently select which data to label and use for model training. Bull et al. developed AL approaches specifically for SHM applications [25].

Their semi-supervised AL framework addressed the scenario where large amounts of unlabeled vibration data are available, but obtaining labels, that is determining whether damage is present, is expensive. The AL strategy selectively queries labels for the most informative data points, significantly reducing labeling costs while maintaining detection accuracy.

Building on this work, Bull et al. introduced a probabilistic AL framework for online SHM [26], using GP classifiers to provide uncertainty estimates that guide the selection of data to label. Hughes et al. further extended these ideas to develop robust risk-based AL algorithms for enhanced decision support in SHM [61]. Their work explicitly considered decision risks in the AL loop, ensuring that the data acquisition strategy aligned with the ultimate goal of making reliable damage detection decisions rather than simply reducing model uncertainty. These applications have successfully demonstrated the value of AL in SHM, where continuous data streams are available and the challenge is efficient data selection rather than data generation.

Structural health monitoring has a mature optimal sensor placement literature where DOE and AL methods are employed [102]. To address multi-objective optimization challenges, Yang [146] developed an adaptive weight-updating algorithm that combines multiple sensor placement methods while maintaining their inherent characteristics through iterative optimization. However, these sensor placement methods optimize spatial measurement locations rather than adaptively selecting experiment parameters or training data for neural networks.

For digital twin development, Gardner et al. used AL strategies to update models based on uncertainty estimates [45]. Their work addressed the problem of maintaining an accurate digital twin as the physical system evolves over time, using AL to determine when and what type of measurements should be taken to update the model efficiently. However, this application assumes the existence of a physical system that can be queried at any time, which is different from the pre-deployment testing scenario addressed in this work.

Zhang et al. introduced an AL aided Bayesian nonparametric general regression network to reduce finite element model evaluations and enhance computational efficiency in model updating [150]. Their method combined the flexibility of Bayesian nonparametric regression with AL to efficiently explore the parameter space during model updating. By strategically selecting which finite element simulations to run next based on model uncertainty, they achieved significant computational savings. However, both of these applications assume the availability of computational models such as finite element models or ground motion databases from which data can be sampled, which is fundamentally different from generating data from physical experiments where no prior model exists.

For model calibration and updating using experimental data, Abbiati and Marelli presented a procedure for sequential design of experiments to improve the convergence of Bayesian model calibration [1]. Their adaptive sampling strategy was based on the uncertainty provided by a Kriging surrogate model that approximated the discrepancy between hybrid simulation and pure computer simulation. The goal was to identify model param-

eters that minimized this discrepancy. Furthermore, Abbiati et al. used AL together with Kriging metamodels to lower the experimental cost of hybrid simulation for the construction of safe/failure domain classifiers [2]. Their approach trained a classifier to distinguish between safe and failure regions of the design space using as few expensive hybrid simulations as possible. While AL is used in these contexts to reduce experimental costs, the fundamental difference from the present work is that a computational model is assumed to exist and the goal is parameter calibration or validation, rather than generating a complete predictive model from scratch without any prior model assumptions.

1.2.3.3 Applications in Experiments Beyond Structural Dynamics

An example where query synthesis methods have been developed in contexts similar to the present work is in self-driving laboratories, particularly in chemistry and materials science. Rapp et al. developed a self-driving laboratory for autonomous protein design [116], where a machine learning agent iteratively designed experiments, executed them in an automated laboratory, and used the results to update its model and design the next experiments. Their approach increased sampling efficiency and reduced experimental costs, leading to novel protein designs that were previously unattainable. Abolhasani and Kumacheva provided a broader review of self-driving laboratories in chemical and materials sciences [4], highlighting how the integration of machine learning, automation, and robotics is transforming experimental research by enabling closed-loop optimization where experiments are autonomously designed and executed.

However, these applications differ significantly from vibration control device testing in several important ways. In protein design or chemical synthesis, individual experiments can be directly evaluated against an objective function such as protein stability or reaction yield, and this evaluation provides immediate feedback for the next design iteration. In contrast, experimental testing of vibration mitigation devices in civil engineering requires both training data and a separate validation dataset to be generated from scratch [92]. The performance of the model cannot be evaluated on the same data used for training. Instead, a test set representative of the intended application must be defined, and the model's performance on this test set becomes the objective for the AL strategy. This distinction is critical. The AL strategy must generate data that enables the model to perform well on a predefined test set, rather than simply reducing uncertainty or maximizing coverage of the experimental space.

Additionally, in many of the applications discussed above, AL operates on static input-output mappings, where individual data points can be independently sampled. In contrast, for dynamic systems, data comes in the form of time series generated by applying excitation signals over a period of time. The experimental design parameters are not individual data points but rather the characteristics of these excitation signals such as frequency content, amplitude, and duration. This introduces additional complexity, as the informativeness of an experiment depends on the entire time history rather than a single point.

1.3 Scope and Objectives

The preceding review demonstrates that although fundamental components for intelligent experimental design in structural dynamics have been developed across various disciplines, their systematic integration for dynamic systems has not yet been realized. Notably, despite significant research efforts in related fields, a comprehensive data generation framework such as testing-integrated modeling remains absent in structural dynamics applications for vibration control devices. The following gaps in existing research can be identified.

1. **Uncertainty-dependent active learning.** Most AL frameworks for physical systems rely on uncertainty quantification from probabilistic models, which can be computationally expensive and may not scale well to complex dynamic systems. Furthermore, many modern neural network architectures used in practice are deterministic and do not provide intrinsic uncertainty estimates, limiting the applicability of uncertainty-based AL strategies. Critically, no published work explicitly combines derivative-free optimization methods with deterministic neural networks for query synthesis in structural dynamics. This gap enables broader applicability without computational overhead of uncertainty quantification.

2. **Sampling dynamic data.** Many AL approaches are designed for static input-output mappings rather than time-series data from dynamic systems. In physical experiments on dynamic systems, experimental conditions must be specified including excitation signals rather than individual data points, and the informativeness must be evaluated at the level of entire time histories rather than individual samples. While optimal input signal design has been extensively studied, these methods often optimize signal characteristics independently rather than adaptively based on current model performance. The integration of broader experimental design with model-guided AL for time-series experiments remains largely unaddressed.

3. **Batch methods for dynamic systems.** Existing input signal design methods for system identification are predominantly batch approaches that optimize signal characteristics to maximize information content. However, these methods rarely exploit information from previous experiments to adapt future tests, missing the opportunity for iterative improvement that AL provides. Even sequential approaches like Heinz et al. [55] often focus on data diversity rather than model accuracy as the optimization criterion. The fundamental paradigm shift from coverage-based to performance-based experimental design has not been realized in structural dynamics.

4. **Mandated testing protocols.** Current testing practices for vibration control devices generate extensive data during mandated performance tests but do not produce usable mathematical models. This represents an opportunity for integration,

as the same experimental infrastructure used for performance verification could simultaneously generate high-fidelity models if an intelligent data acquisition strategy were employed. While self-driving laboratories in chemistry such as those developed by Rapp et al. [116] and reviewed by Abolhasani and Kumacheva [4] have demonstrated automated experimental design, the unique requirements of structural dynamics such as separate training and validation datasets, time-series excitations, and performance on test sets create fundamentally different constraints that remain largely unaddressed by existing frameworks.

Addressing these gaps requires developing an AL framework specifically tailored to the constraints of physical testing, such as high per-experiment cost, sequential experimentation, dynamic data, and the need to generate both training and validation datasets from scratch. The primary objective of this dissertation is to develop and validate a systematic framework for testing-integrated modeling of nonlinear vibration control devices using adaptive data generation. The framework enables automated, efficient collection of experimental data during mandated performance testing while simultaneously generating accurate data-driven models of device behavior. The focus is on systems where specimen mass cannot be decoupled from restoring force, such as TMDs and TLCDs, hence requiring sampling of dynamic data rather than static force measurements.

For the intended testing-integrated modeling, the informativeness of data is controlled through parameters that define experimental conditions. These parameters may include, in the most direct approach, the characteristics of the excitation signal or, when applicable, other factors specific to the specimen such as damper configuration. If relevant, additional secondary effects such as ambient temperature or loading history can also be considered as experimental parameters. The key innovation is the use of a derivative-free optimization algorithm to adaptively select experimental conditions that maximize model performance on a predefined test set, without requiring uncertainty estimates from the model. This approach makes the framework applicable to any neural network architecture, including deterministic FNNs and Neural ODEs. Optimized data sampling is coupled with modeling of nonlinear system behavior using ANNs, creating a sampling strategy where the model's current performance directly informs the design of the next experiment.

This dissertation presents a comprehensive approach that addresses these limitations and incorporates both numerical and experimental validation. The main contributions of this work are as follows.

1. **A general active learning framework for physical experiments.** Development of an optimization-driven approach for data generation that does not rely on model uncertainty, making it applicable to various deterministic neural network architectures and suitable for expensive physical testing scenarios. The framework is general enough to be adapted to different types of vibration control devices and experimental setups.

2. **Enhanced Neural ODE architecture.** Development of an extended Neural ODE architecture capable of simultaneously identifying system dynamics and unknown modal mass. This architecture addresses the common scenario where the effective mass of the device is not precisely known a priori, with potential applications beyond vibration control devices to other mechanical systems where mass identification is required.

3. **Comprehensive validation.** Numerical validation using a Duffing oscillator as a benchmark system demonstrating superior performance compared to conventional batch sampling methods, as detailed in Chapter 5. Experimental validation using a TLCD prototype through shaking table tests, demonstrating the practical feasibility of the framework in a real laboratory setting, as presented in Chapter 6. Extensions of the proposed framework and their application in cases where the core assumptions cannot be assumed, such as availability of comprehensive test dataset, are explored in Chapter 7.

This research represents a first step toward automated, intelligent testing procedures that simultaneously satisfy regulatory requirements and generate high-fidelity models, reducing overall testing time and costs while improving model reliability. The approach bridges the gap between mandated performance testing and model development, combining both verification and predictive capability.

1.4 Outline

The remainder of this dissertation is organized as follows.

Chapter 2 establishes the theoretical foundations of data-driven modeling for dynamic systems, with detailed coverage of FNNs and Neural ODEs. These architectures are explained in the context of training procedures for representing nonlinear dynamical systems, with discussion of their advantages and limitations for system identification. Special attention is given to the formulation of Neural ODEs for mechanical systems and the training procedures required to ensure accurate representation of dynamics. The bias-variance tradeoff is analyzed through numerical examples, demonstrating that representative spatial coverage of training data often proves more important than absolute sample size.

Chapter 3 provides a comprehensive overview of experimental design methodologies, covering both classical DOE approaches and modern AL strategies. The theoretical framework for adaptive data acquisition is established, and relevant optimization algorithms for query synthesis are reviewed. Critical limitations of standard AL methods are identified for physical experiments with dynamic systems, particularly the dynamic sampling problem where desired measurement points cannot be directly specified but must be achieved through excitation parameter optimization.

Chapter 4 introduces the Adaptive Data Generation (AdaGen) framework for testing-integrated modeling. The methodology is presented in the context of vibration control device testing, covering all essential components including the definition of experimental parameters and test sets, the optimization algorithm for experimental parameter selection, training procedures for neural network models, and performance metrics. Implementation details are provided for FNN-based force-state mapping as a baseline approach for dynamics learning.

Chapter 5 presents numerical validation of the framework using a Duffing oscillator as a benchmark system with FNNs for force-state mapping. The adaptive approach is compared quantitatively against conventional batch sampling methods, demonstrating the advantages of the proposed framework. An application example illustrates how the identified model can be integrated into a two-story shear frame with a nonlinear TMD.

Chapter 6 provides experimental validation using a TLCD prototype. The framework is advanced through Neural ODE integration, and architectural modifications are introduced for the simultaneous identification of system dynamics and unknown modal mass parameters.

Chapter 7 addresses key limitations of the baseline framework through four methodological extensions. These include interpretable neural network initialization to reduce computational cost through physically informed parameters, cumulative data generation to enable strategic combination of multiple datasets, reference-free adaptive sampling that eliminates comprehensive test data requirements through cross-validation, and multi-validation robust optimization with Bayesian optimization.

Chapter 8 presents conclusions and directions for future research.

Chapter 2

System Identification with Neural Networks

Machine learning methods have gained significant traction as a potential alternative to classical methodologies in structural dynamics. They have proven to be a versatile tool for modeling complex, nonlinear relationships within systems. Since they are data-driven approaches that learn directly from measured data, they are well-suited for model updating and structural health monitoring applications. Furthermore, neural networks can incorporate both physical knowledge based on first principles and data-driven learning, enabling gray-box approaches that balance interpretability with flexibility.

This chapter introduces the machine learning approaches, particularly artificial neural networks, which are crucial to understanding the work and results in this thesis. The focus of this chapter is on FNNs and Neural ODEs, since the data generation methodology proposed in Chapter 4 is validated using these architectures in Chapter 5 and Chapter 6 respectively. Furthermore, this chapter introduces general aspects of ML methods necessary to understand the integration of machine learning with Design of Experiments (DOE) presented in Chapter 3, which addresses the challenge of efficiently collecting training data for these models in experimental settings where each measurement is time-consuming and requires expert oversight.

System identification in structural dynamics involves determining mathematical models that describe the dynamic behavior of structures from experimental data. Traditional approaches include physics-based white-box models derived from first principles, which require comprehensive knowledge of system properties, and various black-box data-driven methods that learn input-output relationships directly from measurements without explicit physical assumptions [78, 140]. Gray-box approaches combine physical knowledge with data-driven components, offering a balance between interpretability and flexibility.

For nonlinear structural systems such as vibration control devices, structures with geometric nonlinearities, or systems exhibiting hysteretic behavior, system identification presents particular challenges. The identification problem becomes significantly more complex due

to the need to capture nonlinear effects while maintaining model generalizability across the operational range. Classical parametric approaches often require assuming specific nonlinear forms, limiting their applicability when the underlying physics is complex or partially unknown [67, 100].

Neural networks offer a compelling alternative as universal function approximators capable of learning complex nonlinear mappings directly from data. As black-box models, they require minimal prior knowledge about system structure, making them particularly suitable for scenarios where physical parameters are difficult to measure or when modeling auxiliary components with unknown internal dynamics. The challenge, however, lies in collecting sufficient high-quality training data that captures the system's behavior across all relevant operating conditions. In experimental settings such as the tuned liquid column damper prototype studied in this thesis, each measurement is time-consuming and repetitive, requiring expert knowledge to determine which experiments should be performed. This practical constraint motivates the adaptive experimental design framework (ADaGen) developed in Chapter 4, which aims to maximize information gain while minimizing experimental effort.

This chapter introduces the neural network architectures employed for system identification in the subsequent chapters, with particular focus on FNNs and Neural ODEs. Understanding these architectures, their training procedures, and their limitations provides the foundation for the adaptive data generation methodology presented in Chapter 4.

2.1 Artificial Neural Networks

Artificial neural networks (ANNs) are a class of computational models that derive their architectural inspiration from biological neural systems. These networks represent a significant domain within computer science research, focused on developing computational systems capable of exhibiting human-like cognitive abilities for complex problem-solving and autonomous learning.

ANNs operate within the broader theoretical framework of Artificial Intelligence (AI), which encompasses both symbolic knowledge-based approaches and statistical Machine Learning (ML) methodologies. The distinctive characteristic of neural networks lies in their inherent capacity to extract patterns from training data and subsequently apply this knowledge to novel, previously unencountered scenarios, a process known as *generalization*. This learning process leverages highly parallelized computational structures, significantly enhancing training efficiency across various hardware implementations.

Neural networks can address diverse computational tasks including regression (modeling continuous input-output relationships), classification (categorical assignments), and clustering (unsupervised grouping). For system identification in structural dynamics, this work focuses exclusively on regression tasks, where neural networks learn to map measured system states (such as displacements and velocities) to corresponding responses or

forces. This supervised learning paradigm is particularly well-suited to modeling nonlinear dynamic systems where the governing equations may be complex, partially unknown, or computationally intractable.

The application of ANNs to structural dynamics has evolved considerably over recent decades. Early applications focused primarily on damage detection and structural health monitoring through pattern recognition and classification [14, 141]. More recently, ANNs have been increasingly employed for system identification and response prediction, leveraging their universal approximation capabilities to model complex nonlinear behaviors directly from experimental data. Modern architectures such as *recurrent neural networks* (RNNs) [40] and *long short-term memory networks* (LSTMs) [58] have demonstrated success in capturing temporal dependencies in structural responses [130], while *convolutional neural networks* (CNNs) [74] have proven effective for automated feature extraction from vibration signals [3]. Additionally, *radial basis function neural networks* (RBFNs) [95] have been successfully applied for frequency-domain damage identification in multimember structures [83].

Beyond these established architectures, specialized neural network formulations have emerged to address specific challenges in structural dynamics, including Physics-Informed Neural Networks (PINNs) that incorporate governing equations directly into training [115], Graph Neural Networks (GNNs) [123] for systems with complex connectivity [132], and Reservoir Computing approaches [63, 81] for efficient time-series prediction [103].

For the experimental system identification problem addressed in this thesis, two architectures are particularly well-suited. FNNs (Section 2.1.2) provide direct force-state mapping when restoring forces can be measured through load cells or strain gauges, enabling static characterization of nonlinear systems. Neural ODEs (Section 2.1.3) enable system identification from displacement and velocity measurements alone, eliminating the need for invasive force instrumentation. Their continuous-time formulation naturally represents the ordinary differential equations governing structural dynamics while naturally handling irregular sampling intervals. Both architectures accommodate the experimental constraints of the tuned liquid column damper system studied in Chapters 5 and 6, where measurement availability and experimental practicality determine which approach is most suitable. These architectures will be introduced in detail in subsequent sections, along with the fundamental principles governing their training and generalization capabilities.

2.1.1 Artificial Neuron

Artificial neurons are computational units inspired by biological neurons, first formalized by McCulloch and Pitts [88, 128]. Modern artificial neurons compute weighted sums of inputs, apply nonlinear activation functions, and serve as building blocks for neural networks. A major advancement came with the backpropagation algorithm [119], which enabled training of multi-layer networks through smooth, differentiable activation functions.

As illustrated in Fig. 2.1, an artificial neuron receives inputs from previous layers, processes them through weighted connections, and produces an output. Each connection has an associated weight w_i representing the strength of that input's contribution. The neuron computes the weighted sum of inputs, adds a bias term b, and applies an activation function $\sigma(\cdot)$:

$$\hat{y} = \sigma\left(\sum_{i=1}^{n} w_i x_i + b\right) \tag{2.1}$$

where n is the number of inputs, and $\sigma(\cdot)$ introduces nonlinearity enabling the network to model complex relationships.

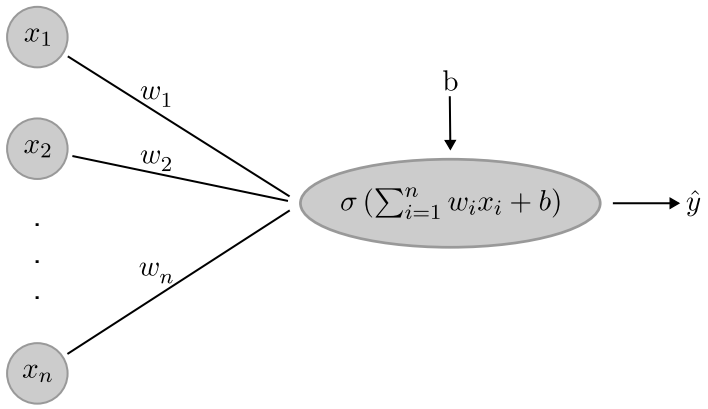

Figure 2.1: Schematic of an artificial neuron showing weighted inputs $w_i x_i$, bias term b, and activation function $\sigma(\cdot)$ producing output $\hat{y}$

Common activation functions used in this work are presented in Table 2.1. The hyperbolic tangent function is employed in the FNN implementation (Chapter 5), while the sigmoid function is used for interpretable neural network initialization (Chapter 7).

By connecting artificial neurons in different configurations, various neural network architectures can be constructed. This dissertation employs two architectures suited to the testing-integrated modeling problem. FNNs are used for static force-state mapping when forces are measurable (Chapter 5), and Neural ODEs are used for modeling continuous dynamics from state measurements alone (Chapter 6).

Activation Function	**Equation**	**Advantages**
Sigmoid	$\sigma(x) = \frac{1}{1+e^{-x}}$	Smooth gradient, bounded output between 0 and 1; used in interpretable machine learning.
Hyperbolic Tangent (tanh)	$\sigma(x) = \tanh(x) = \frac{2}{1+e^{-2x}} - 1$	Centered around zero, bounded output between -1 and 1; generally performs better than sigmoid for hidden layers.
ReLU (Rectified Linear Unit)	$\sigma(x) = \max(0,x)$	Computationally efficient, promotes sparsity, mitigates vanishing gradient problem.
Leaky ReLU	$\sigma(x) = \begin{cases} x, & x \geq 0 \\ \alpha x, & x < 0 \end{cases}$	Alleviates "dying ReLU" problem by allowing small, non-zero gradients when inactive; improves learning in deep networks.
Softmax	$\sigma(x_i) = \frac{e^{x_i}}{\sum_j e^{x_j}}$	Converts raw scores to probability distribution; essential for multi-class classification problems.

Table 2.1: Common activation functions with their mathematical formulations and functional advantages

2.1.2 Feedforward Neural Networks

For modeling purposes in this work, FNNs are considered as data-driven methods, a type of neural network with forward information propagation through layers, meaning no information is looped back or delayed. According to the universal approximation theorem, FNNs can approximate any continuous function to arbitrary precision, given sufficient neurons and appropriate activation functions [33, 59]. This makes them particularly suitable for force-state mapping and they are widely used in nonlinear dynamics applications for system identification and control [128]. They have also shown considerable resistance to noise [98], which can even enhance training robustness and prevent overfitting [18, 128].

Figure 2.2 illustrates a typical FNN architecture consisting of an input layer where function inputs x are presented; hidden layers which contain optimizable parameters that enable function approximation; and an output layer which produces the desired outputs $\hat{y}$. The parameters which are optimized in an FNN are usually weights w and bias terms b. Depending on how many neurons there are in each layer, the weights denote the strength of connection of the mth neuron in layer n to the neuron in the previous layer. The bias term is applied at each neuron, which receives inputs from a previous layer, processes

them, and passes the results forward to the next layer, as explained in Section 2.1.1.

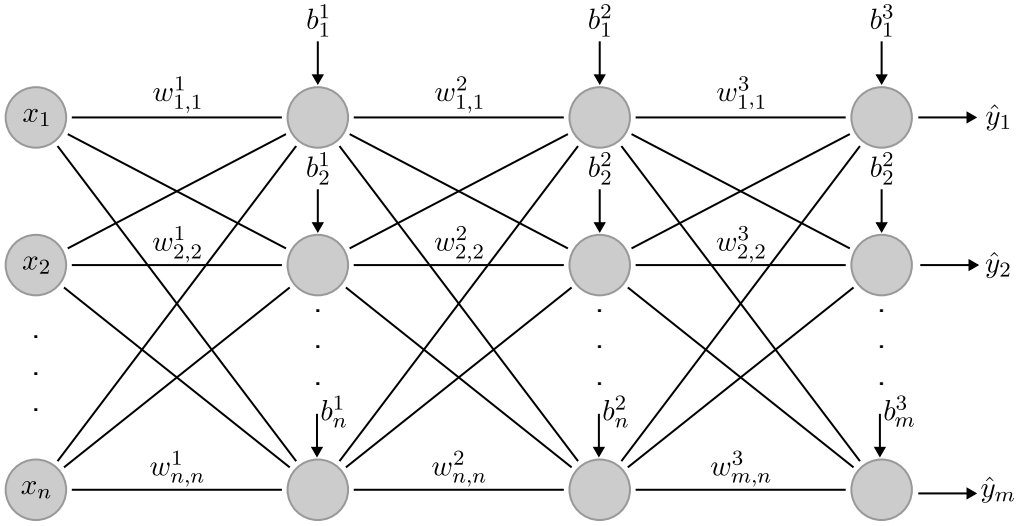

Figure 2.2: Graphical model of the artificial neural network

The structure of the neural network is commonly referred to as its architecture. For a network with L layers, the forward propagation can be expressed recursively in matrix form as:

$$\mathbf{a}^l = \sigma^l(\mathbf{W}^l\mathbf{a}^{l-1} + \mathbf{b}^l), \quad l = 1,2,\ldots,L \tag{2.2}$$

where $\mathbf{a}^l \in \mathbb{R}^{n_l}$ is the activation vector of the lth layer with n_l neurons; $\sigma^l(\cdot) : \mathbb{R}^{n_l} \to \mathbb{R}^{n_l}$ is the element-wise activation function applied at the lth layer; $\mathbf{W}^l \in \mathbb{R}^{n_l \times n_{l-1}}$ is the weight matrix connecting layers $l-1$ and l; $\mathbf{b}^l \in \mathbb{R}^{n_l}$ is the bias vector of the lth layer; and $\mathbf{a}^0 = \mathbf{x}$ represents the input layer. The final layer output $\mathbf{a}^L = \hat{\mathbf{y}}$ provides the network prediction.

Training of Feedforward Neural Networks FNNs are trained using optimization algorithms that adjust the network parameters (weights and biases) to minimize a predefined loss function. The most common training approach is backpropagation, which consists of two main phases, forward propagation and backward propagation [119].

During forward propagation, input data $\mathbf{x}$ passes through the network to generate predictions $\hat{\mathbf{y}} = f_\theta(\mathbf{x})$, where θ denotes all network parameters (weights and biases). These predictions are compared with the target values $\mathbf{y}$ to compute the error according to a loss function. For regression tasks, the mean squared error (MSE) is commonly employed as

$$J(\theta) = \frac{1}{N}\sum_{i=1}^{N} \|\mathbf{y}^{(i)} - \hat{\mathbf{y}}^{(i)}\|^2 = \frac{1}{N}\sum_{i=1}^{N} \|\mathbf{y}^{(i)} - f_\theta(\mathbf{x}^{(i)})\|^2 \tag{2.3}$$

where N is the number of training samples and $\|\cdot\|$ denotes the Euclidean norm. In backward propagation, the gradient of the loss function with respect to each network parameter is calculated using the chain rule of calculus, and the parameters are updated in the direction that reduces the error.

The parameter update rule typically follows the form

$$\theta_{t+1} = \theta_t - \eta \nabla_\theta J(\theta_t) \tag{2.4}$$

where θ_t represents the parameters at iteration t, η is the learning rate, and $\nabla_\theta J(\theta_t)$ is the gradient of the loss function J with respect to the parameters.

Common optimization algorithms include *Gradient Descent* (GD) which updates parameters using the full dataset for each iteration, *Stochastic Gradient Descent* (SGD) which updates parameters using a single randomly selected sample per iteration, and *Mini-batch SGD* which updates parameters using small random batches of samples. Furthermore, some advanced optimizers such as *Adam* [70], *RMSprop*, and *AdaGrad* adapt learning rates for each parameter based on historical gradient information. The choice of optimization algorithm affects both convergence speed and final solution quality. Batch gradient descent guarantees convergence to a local minimum for convex problems but can be slow for large datasets. SGD introduces stochasticity that can help escape poor local minima but exhibits noisy convergence. Mini-batch SGD balances these trade-offs, providing computational efficiency while maintaining reasonable convergence properties. Adaptive methods like Adam typically converge faster in practice but may require careful tuning of hyperparameters such as learning rate η, momentum coefficients, and batch size to achieve optimal performance.

Overfitting, which will be discussed in more detail in the following section, is a common problem with FNNs. To prevent overfitting, regularization techniques are commonly employed during training, including L_1 and L_2 regularization, dropout, and early stopping. Additionally, proper initialization of weights, such as Xavier/Glorot initialization [47], is crucial for effective training of deep networks.

For robust training and model evaluation, it is recommended to use three separate datasets. *Training data* is used for parameter optimization, *validation data* is used for model selection and hyperparameter tuning, and *test data* is used for final, unbiased performance evaluation.

Note that for proper training and testing, both training and test data should be representative and include data from all regions of the input space. This ensures both effective training and a reliable estimate of model generalization. Using the same dataset for validation and testing may yield overly optimistic performance estimates. The requirement for representative data across the full operating range, particularly challenging when experiments are time-consuming and expensive, motivates the active learning strategies presented in Chapter 4.

It is worth noting that FNN architectures with linear activation functions in the output layer and sigmoid or hyperbolic tangent activation functions in hidden layers are also referred to as multilayer perceptron (MLP) networks. The applications of MLPs in system modeling and identification can be found extensively in the literature [98, 128, 140] and more generally about ML in Goodfellow et al. [49].

2.1.3 Neural Ordinary Differential Equations

Neural Ordinary Differential Equations (Neural ODEs) represent a class of artificial neural networks particularly well-suited for modeling dynamic systems, as they naturally embed the ordinary differential equation framework required for modeling continuous-time dynamics. Unlike discrete-time recurrent architectures such as RNNs and LSTMs, Neural ODEs directly parameterize the continuous dynamics of a system, making them especially appropriate for structural dynamics applications within the scope of this thesis.

From Residual Networks to Neural ODEs To understand the conceptual foundation of Neural ODEs, it is useful to first consider their relationship to Residual Networks (ResNets) [54]. In ResNets, the transformation between consecutive hidden layers (or *states*) $\mathbf{h}_t$ and $\mathbf{h}_{t+1}$ is defined by a differentiable function f_θ such that

$$\mathbf{h}_{t+1} = \mathbf{h}_t + f_\theta(\mathbf{h}_t). \tag{2.5}$$

This formulation allows for discrete transitions through layers. The evolution can be interpreted through the explicit Euler integration scheme as $\mathbf{h}_{t+1} = \mathbf{h}_t + f_\theta(\mathbf{h}_t)\Delta t$, where $\Delta t = 1$. To obtain a continuous representation, a Neural ODE, $\Delta t \to 0$ is applied to obtain

$$\lim_{\Delta t \to 0} \frac{\mathbf{h}_{t+1} - \mathbf{h}_t}{\Delta t} = \frac{d\mathbf{h}(t)}{dt} = f_\theta(\mathbf{h}(t),t,\boldsymbol{\theta}) \tag{2.6}$$

where f_θ represents the internal neural network with parameters $\boldsymbol{\theta}$, including all weights and biases. This transforms the discrete layer transitions of ResNets into the continuous dynamics of Neural ODEs, providing a more natural framework for modeling physical systems governed by differential equations.

Mathematical Formulation Neural ODEs, introduced by Chen et al. [29], model continuous dynamical systems by parameterizing the time derivative of the system state using a neural network. The general formulation is expressed as

$$\dot{\mathbf{h}} = f_\theta(\mathbf{h},t,\boldsymbol{\theta}), \qquad \mathbf{h}(t_0) = \mathbf{h}_0, \tag{2.7}$$

where $\mathbf{h}(t) \in \mathbb{R}^n$ represents the state vector of the system at time t, $\mathbf{h}_0$ denotes the initial conditions at time t_0, and $f_\theta : \mathbb{R}^n \times \mathbb{R} \times \mathbb{R}^p \to \mathbb{R}^n$ is an FNN parameterized by weights and biases collected in the parameter vector $\boldsymbol{\theta} \in \mathbb{R}^p$.

The neural network f_θ typically consists of multiple layers, where each layer performs a linear transformation followed by a nonlinear activation function as

$$\mathbf{h}^l = \sigma^l(\mathbf{W}^l\mathbf{h}^{l-1} + \mathbf{b}^l), \tag{2.8}$$

where $\mathbf{h}^l$ represents the activation of the l-th layer, $\mathbf{W}^l$ and $\mathbf{b}^l$ are the weight matrix and bias vector for that layer, and $\sigma^l(\cdot)$ is the activation function. Common choices for

activation functions include hyperbolic tangent (tanh), rectified linear units (ReLU), or sigmoid functions, with the selection depending on the specific application and system characteristics as mentioned in Section 2.1.1.

Given an initial condition $\mathbf{h}(t_0)$, the state at any future time t can be predicted by numerically integrating Eq. (2.7) using standard ODE solvers such as the Runge-Kutta family of methods, Adams-Bashforth methods, or adaptive step-size solvers. The flexibility to use different integration schemes and time steps makes Neural ODEs particularly suitable for cases with irregular sampling intervals.

Training Neural ODEs Training Neural ODEs involves optimizing the network parameters $\boldsymbol{\theta}$ to minimize the discrepancy between predicted and observed system trajectories. For supervised learning with observed state trajectories $\{\mathbf{h}(t_i)\}_{i=1}^{N}$ at discrete time points $t_1, \ldots, t_N$, the loss function is formulated as

$$\mathcal{L}(\boldsymbol{\theta}) = \sum_{i=1}^{N} \|\mathbf{h}(t_i) - \hat{\mathbf{h}}(t_i; \boldsymbol{\theta})\|^2, \tag{2.9}$$

where $\hat{\mathbf{h}}(t_i; \boldsymbol{\theta})$ represents the predicted state obtained by numerically solving the ODE as

$$\hat{\mathbf{h}}(t_i; \boldsymbol{\theta}) = \text{ODESolve}(\mathbf{h}(t_0), f_\theta, t_0, t_i, \boldsymbol{\theta}). \tag{2.10}$$

The key challenge in training Neural ODEs lies in computing gradients of the loss function with respect to the parameters $\boldsymbol{\theta}$. Direct backpropagation through the ODE solver would require storing all intermediate states $\mathbf{h}(t)$ at each integration step during forward propagation, resulting in memory requirements that scale as $\mathcal{O}(NT)$, where N is the state dimension and T is the number of integration steps. For long time series or high sampling rates, this becomes computationally prohibitive. Instead, Chen et al. [29] introduced the *adjoint sensitivity method*, which computes gradients efficiently by solving an additional ODE backward in time. The adjoint state $\mathbf{a}(t) = \partial\mathcal{L}/\partial\mathbf{h}(t)$ satisfies

$$\frac{d\mathbf{a}}{dt} = -\mathbf{a}(t)^T \frac{\partial f_\theta}{\partial \mathbf{h}}, \qquad \mathbf{a}(t_N) = \frac{\partial \mathcal{L}}{\partial \mathbf{h}(t_N)}, \tag{2.11}$$

and the gradient with respect to parameters is obtained as

$$\frac{d\mathcal{L}}{d\boldsymbol{\theta}} = -\int_{t_N}^{t_0} \mathbf{a}(t)^T \frac{\partial f_\theta}{\partial \boldsymbol{\theta}} dt. \tag{2.12}$$

This adjoint method enables memory-efficient training by avoiding the storage of intermediate forward states, making Neural ODEs scalable to long time sequences and high-dimensional systems. The training procedure alternates between forward integration to compute predictions and loss, backward integration of the adjoint equations to compute gradients, and parameter updates using standard optimization algorithms such as Adam

or stochastic gradient descent.

Application to Structural Dynamics For structural dynamics applications, the state vector $\mathbf{h}$ typically represents the displacement and velocity of the system. For an SDOF oscillator, the state can be written as

$$\mathbf{h}(t) = \begin{bmatrix} x(t) \\ \dot{x}(t) \end{bmatrix}, \tag{2.13}$$

where $x(t)$ is the displacement and $\dot{x}(t)$ is the velocity. The neural network f_θ then learns to predict the time derivatives as

$$\frac{d}{dt}\begin{bmatrix} x(t) \\ \dot{x}(t) \end{bmatrix} = f_\theta\left(\begin{bmatrix} x(t) \\ \dot{x}(t) \end{bmatrix}, t, \boldsymbol{\theta}\right). \tag{2.14}$$

For mechanical systems, the first component of this derivative is simply the velocity ($\dot{x}$), while the second component represents the acceleration, which the neural network must learn from the system dynamics.

Neural ODEs offer several advantages for structural dynamics and system identification applications. Unlike force-state mapping approaches using FNNs, Neural ODEs can be trained using only displacement and velocity measurements, eliminating the need for load cells to measure restoring forces. This capability is particularly valuable for the tuned liquid column damper experiments presented in Chapter 6. Neural ODEs naturally represent continuous dynamics without discretization errors, allowing predictions at arbitrary time resolutions regardless of the sampling rate used during training. The ODE structure aligns with the mathematical description of mechanical systems, and extensions to the standard formulation allow explicit identification of physical parameters such as modal mass, as demonstrated in Chapter 6. Once trained, Neural ODEs can predict system behavior over extended time horizons by continuing the integration, making them suitable for simulation and analysis of controlled structures.

However, training Neural ODEs is more computationally expensive than training FNNs, as each forward pass requires numerical integration and each backward pass requires solving the adjoint equations. As noted by Bradley et al. [22], Neural ODEs require training data that captures the essential system dynamics across all relevant operating regimes to ensure proper generalization. If the data does not explore key dynamical regimes, the model may fail to generalize. Hence, the fundamental challenge of obtaining representative training data across the system's operating range addressed by the ADaGen framework in Chapter 4 is relevant for the Neural ODE. Chapter 6 demonstrates the application of an extended Neural ODE architecture to experimental data from a TLCD prototype, demonstrating the applicability of Neural ODEs within the ADaGen framework in a realistic experimental setup where physical parameters are not known a priori.

2.2 Bias-Variance Tradeoff

In data-driven modeling, the total model error can be decomposed into two fundamental components, *bias error* and *variance error*. This decomposition, thoroughly discussed by Nelles [98] and referenced in additional literature [17, 46, 139], is essential for understanding the trade-offs that arise when fitting models of varying complexity to observed data. While this concept applies to all modeling approaches, the present discussion focuses on artificial neural networks (ANNs) in the context of structural dynamics applications.

In structural dynamics, the bias-variance tradeoff has particular significance due to the nonlinear nature of many systems and the challenges in obtaining comprehensive training data. The complex relationship between system inputs and responses, combined with measurement noise and operational constraints, makes careful consideration of model complexity essential. Understanding these trade-offs enables practitioners to make informed decisions about model architecture, training procedures, and data collection strategies to achieve optimal performance for their specific applications.

The *bias error* reflects a model's inability to capture the underlying system dynamics. Models that are too simple or insufficiently flexible tend to underfit the data, resulting in persistently high bias error. For linear processes, if the model order exceeds the actual process order (e.g., a fifth-order model for a fourth-order process), the bias error can become negligible. However, in nonlinear scenarios, which are common in structural dynamics, the bias error can never be reduced to exactly zero but diminishes as the model gains more parameters and flexibility.

The *variance error* captures how sensitive a model is to fluctuations in the training data. High-variance models may fit the training set exceptionally well but fail to generalize to unseen data, a phenomenon known as overfitting. This behavior is often exacerbated by noisy measurements, imbalanced or limited design of experiments (DOE), and suboptimal parameter optimization. As the number of parameters increases, the variance error typically grows, making the model more susceptible to variations in the training set. Bias error is indicated in the training error and the variance error is indicated in the test error.

As shown in Fig. 2.3, bias and variance are inherently intertwined, where reducing bias error by increasing model complexity often simultaneously increases variance error, and vice versa. Note that Fig. 2.3a only illustrates this tradeoff for a specific model structure and does not represent the optimal solution across all possible architectures. Figure 2.3b demonstrates that for the same model structure, variance error can be reduced by adding more training samples, selecting a more informative training dataset, or reducing measurement noise. The relationship between model error and training data size follows the pattern that for sufficiently flexible models, the model error is proportional to $1/N$, where N is the number of training samples.

Hence, in the model that was chosen as flexible as possible, the total error would be dominated by the variance error. However, this is not desirable as complex models are both more computationally demanding and prone to overfitting. Technically, only overly

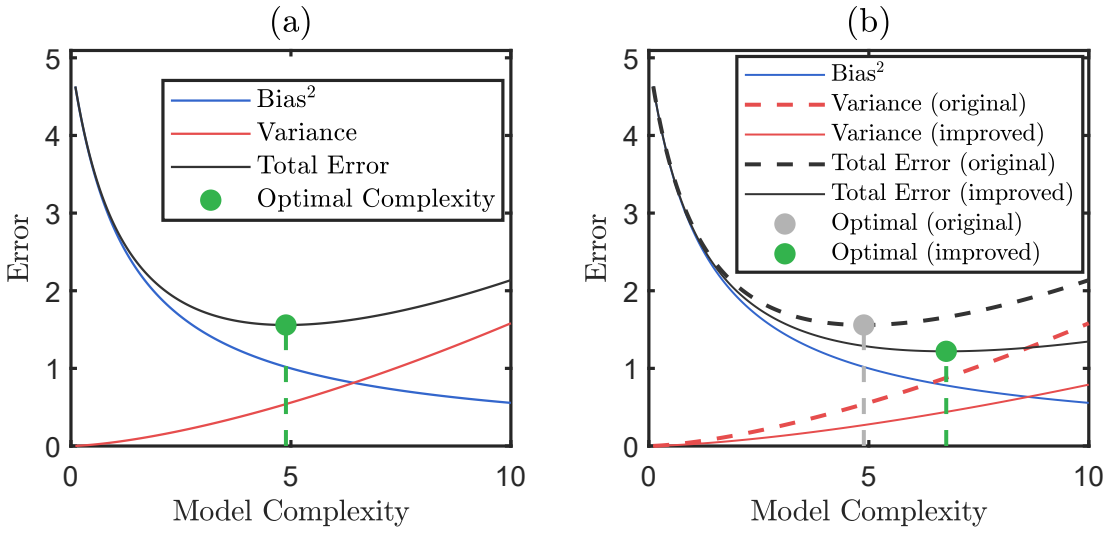

Figure 2.3: (a) The total model error can be decomposed into variance and bias error parts. (b) If the training data set contains more data points, more information, or less noise, the variance error can be decreased.

complex models are prone to overfitting as they may fail to generalize beyond the training data. Hence, choosing a model with an appropriate complexity and trained on sufficient training data is to be done as this reduces the risk of both underfitting and overfitting.

2.2.1 Underfitting and Overfitting

A standard approach to detecting the interplay of bias and variance errors involves separating training and test datasets. This is a simple and effective method to estimate these error components. During training, the model is evaluated and optimized on the training dataset, so the training error primarily reflects bias error. The variance error can be estimated by comparing performance on an independent test dataset.

Underfitting manifests as persistently high error on both training and test sets, suggesting a model that is too simple to capture the underlying system behavior. *Overfitting* appears when the model performs very well on the training set but poorly on the test set, creating a pronounced gap between training and test errors. This gap approximates the amount of overfitting and tends to be larger for more complex models while decreasing with the amount of training data.

2.2.2 Model Complexity

The number of parameters typically serves as a convenient proxy for model complexity, though not all parameters contribute equally to model flexibility. For ANNs, a network with more nodes is generally more complex and flexible than one with fewer nodes. However, this relationship is not strictly one-to-one, and the concept of *effective model*

complexity may differ from *nominal model complexity*. Effective complexity can also be influenced by factors such as the number of training iterations or regularization techniques employed.

Finding the appropriate model complexity is often more critical than selecting the specific model architecture. The optimal balance minimizes overall error by maintaining both bias and variance at acceptable levels, thereby reducing the risk of underfitting or overfitting while enabling effective generalization.

2.2.3 Data Quality and Quantity

Model performance depends critically on both the amount and quality of training data. While some levels of noise can improve generalization by diversifying the training set, excessive noise increases model error. For structural dynamics applications, training data should cover the full operating range of the system and include both linear and nonlinear response regions to represent diverse system states and conditions. Simply collecting large amounts of data from a single steady state provides no new information to the training set, despite containing more data points. Data points that lie close together and provide redundant information are insignificant for improving model performance. The distribution of data in the input space, addressed through proper design of experiments, is crucial for model quality.

Special attention must be paid to intrinsic correlations in structural dynamics, where most inputs cannot be controlled independently due to the dynamic nature of the processes under study.

For problems where the available data set is large enough and contains sufficient information about the system, it can be split into a training and test data set. However, for structural dynamics applications, different experiments can be used for training and testing of the model.

To illustrate the practical implications of data distribution and sample size on the bias-variance trade-off, a controlled experiment was conducted using a synthetic dataset. The true underlying model was defined as $y = \sin(1.5x)$, representing a nonlinear relationship between input x and output y. For all experimental conditions, a third-degree polynomial approximation was employed as

$$\hat{y} = \theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 \tag{2.15}$$

Since the model complexity remained constant across all experiments, the bias error component was consistent in each case. However, the data collection strategy was systematically varied to demonstrate the impact on variance error.

Figure 2.4 presents four distinct scenarios that highlight the interplay between data quality and quantity. In Fig. 2.4a, the polynomial model was fitted using $N = 50$ data points

collected from the narrow range $x \in [-1,0]$. This limited sampling region resulted in poor model generalization beyond the training data range. Similarly, in Fig. 2.4b, $N = 50$ data points were used, but sampled only from the range $x \in [1,2]$. As expected, this alternative limited sampling strategy produced dramatically different model parameters compared to the case in Fig. 2.4a, demonstrating the high variance error inherent in spatially constrained data collection due to inadequate representation of the underlying function.

In Fig. 2.4c, $N = 100$ data points uniformly distributed across the complete range of interest $x \in [-2,2]$ were used. This comprehensive sampling strategy yielded excellent model performance with low variance error, demonstrating that appropriate data distribution across the input space is essential for reducing model sensitivity to training data variations. Finally, in Fig. 2.4d, $N = 50$ data points uniformly distributed across the same range $x \in [-2,2]$ were used. Despite having half the sample size of Fig. 2.4c, the model performance remained excellent and comparable to the high-sample-size case. This indicates that data quality is often more important than data quantity. Both datasets contain equivalent information about the underlying function across the input space, but dataset (d) achieves this with 50% fewer observations. This demonstrates that well-distributed, representative samples can be more valuable than larger datasets with poor spatial coverage. Furthermore, the stark contrast between results obtained with all the datasets emphasizes that variance error is primarily driven by the representativeness of the training data rather than its absolute size. When training data fails to adequately sample the input space, the resulting model exhibits high sensitivity to the specific choice of training points, leading to poor generalization performance.

These findings have important implications for experimental design and data collection strategies as will be discussed in Chapter 3. Practitioners should prioritize representative sampling over maximizing sample size when resources are limited, ensuring that the training data captures the full range of input conditions expected in the target application domain.

2.2.4 Improving the Bias-Variance Tradeoff

Several strategies exist to improve model generalization and performance by addressing the bias-variance tradeoff. These approaches range from statistical model selection criteria to ensemble techniques that combine multiple model predictions.

Information Criteria Information criteria provide a principled approach to model selection by balancing model complexity against goodness of fit. As discussed in [98], various information criteria account for both model complexity and the number of data points. The general principle is to select the model with the lowest information criterion value, where the penalty term increases with model complexity while additional data decreases the effective penalty.

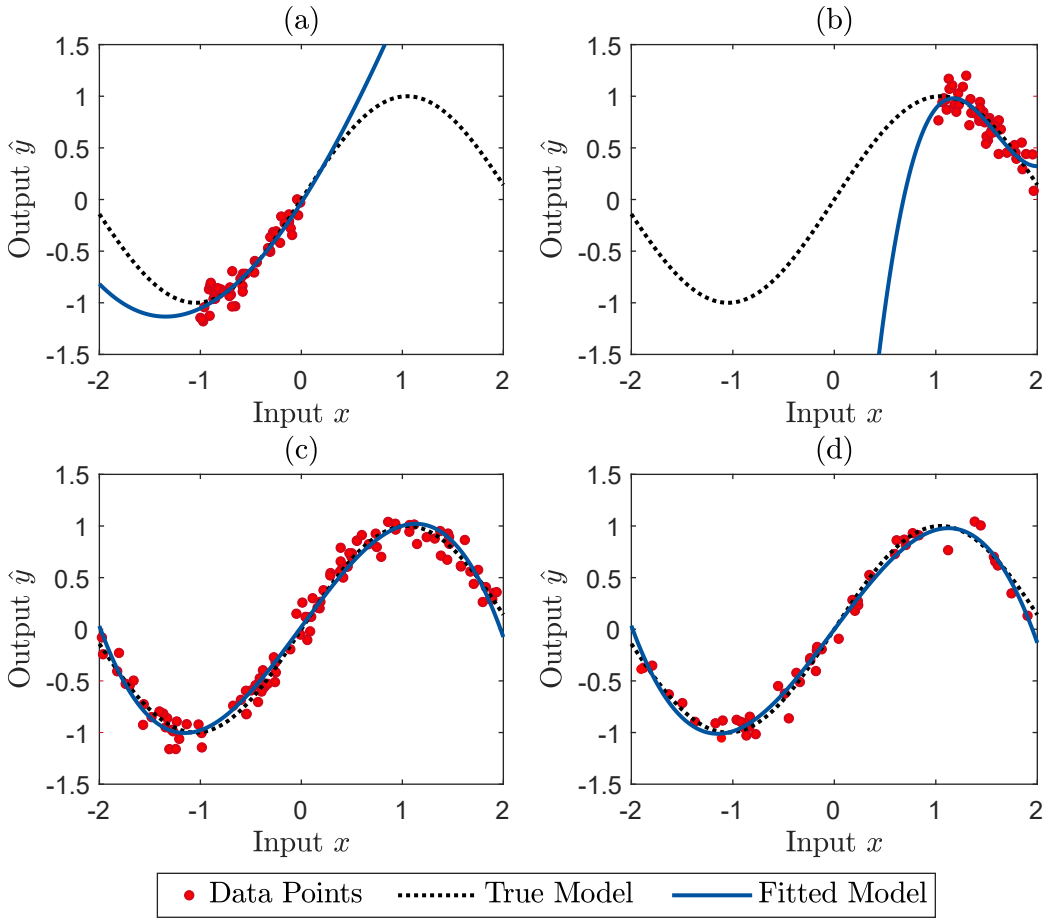

Figure 2.4: Illustration demonstrating the effect of data distribution on model variance: (a) poor spatial distribution (narrow range, N=50), (b) alternative poor distribution (different narrow range, N=50), (c) optimal distribution with high sample size (N=100), and (d) optimal distribution with moderate sample size (N=50). Despite having half the samples, case (d) achieves comparable performance to (c), demonstrating that representative spatial coverage is more important than sample quantity.

Cross-Validation and Bootstrapping When available data is sufficient, cross-validation techniques can provide robust estimates of model performance. Bootstrap methods [39] offer additional approaches for assessing model stability, though they are computationally demanding and can result in models that are difficult to interpret.

Regularization Techniques Regularization methods can help control model complexity and reduce overfitting by adding penalty terms to the loss function or employing techniques such as dropout in neural networks.

Model Pool Selection This approach, employed in this thesis, involves training multiple models in parallel (typically with different random initializations or slight architectural variations) and selecting the single best-performing model based on validation or

test performance. This approach provides robustness against poor local minima during optimization and allows exploration of the parameter space from different starting points. The computational cost during training is distributed across parallel processes, while deployment requires only the resources of a single model. The primary limitation is that only one model's knowledge is utilized, potentially discarding valuable information captured by other models in the pool. Hence, this approach, while computationally efficient for deployment, does not leverage the collective wisdom of multiple models.

Ensemble Methods Ensemble methods extend the model pool concept by combining predictions from multiple models rather than selecting a single model. The fundamental principle underlying ensemble effectiveness is the bias-variance decomposition. While individual models may exhibit high variance due to sensitivity to training data or initialization, ensemble methods reduce variance error by averaging out individual model fluctuations, leading to more stable and generalizable predictions [17]. Common aggregation strategies include simple averaging, weighted voting, or sophisticated meta-learning approaches. The diversity among ensemble members, achieved through different random initializations, varied architectures, distinct training procedures, or data sampling strategies such as bagging or boosting, is essential for realizing these benefits.

Modern ensemble implementations commonly consist of 10 to 100 individual models for practical applications. The computational architecture of ensemble training is inherently parallelizable, as individual models can be trained independently. However, ensemble methods require substantially increased computational and memory resources during inference (proportional to the ensemble size) and reduced model interpretability compared to single models. Advanced approaches such as the Dynamic Ensemble Learning algorithm automatically determine ensemble size and individual network architectures through constructive-pruning strategies, maintaining diversity through techniques like negative correlation learning [7].

For the experimental validation presented in Chapters 5 and 6, model pool selection is employed to ensure robust model identification while maintaining computational efficiency during deployment. Future extensions could investigate full ensemble methods for uncertainty quantification in structural health monitoring applications, where prediction confidence intervals are valuable for decision-making.

2.3 Summary

This chapter establishes the theoretical foundations of neural network architectures for system identification in structural dynamics. Two complementary architectures are introduced in detail: FNNs and Neural ODEs. The bias-variance tradeoff is introduced to motivate improving experimental data quality. This tradeoff presents a fundamental challenge where increasing model complexity reduces bias but increases variance, leading to overfitting. A numerical example demonstrates that performance depends critically on the quantity and quality of training data, with representative spatial coverage often being more important than sample size. For the tuned liquid column damper experiments, obtaining representative training data across the full operating range is essential for reliable modeling. This motivates the active learning and adaptive experimental design methodologies in Chapter 3 and the proposed methodology in Chapter 4. The neural network training principles introduced here provide the foundation for understanding how the proposed methodology guides experimental design, bridging machine learning theory with practical constraints in structural dynamics.

Chapter 3

Design of Experiments and Active Learning

Data-driven models can achieve performance only proportionate to the quality of their training data, making strategic data collection essential for successful modeling. As demonstrated in Section 2.2, model performance depends critically on both the quantity and informativeness of training data, with spatial coverage across the operational range often proving more important than absolute sample size. This chapter presents experimental design methodologies relevant to the adaptive data generation framework developed in Chapter 4, with a focus on space-filling sampling methods and active learning approaches for physical experiments.

Design of Experiments (DOE) provides systematic methodologies for planning experiments to maximize information gain while minimizing experimental costs [94]. Active Learning (AL) extends these concepts with adaptive strategies that iteratively refine data collection based on model performance [124]. While traditional DOE uses static experimental strategies specified before data collection begins, AL enables dynamic adjustment of sampling strategies as knowledge accumulates. This adaptive property is particularly valuable when experiments are expensive or time-consuming, as is the case for testing vibration control devices.

The integration of DOE and AL offers a framework for intelligent data collection. However, as will be demonstrated in Section 3.4, existing AL methods face fundamental limitations when applied to physical experiments involving dynamic systems, which motivate the optimization-based query synthesis framework presented in Chapter 4.

3.1 Design of Experiments

Design of experiments systematically plans experiments to extract maximum information with minimum runs, making it particularly critical in structural dynamics where experiments are time-consuming and require expert oversight. Pioneered by R.A. Fisher in the 1920s during agricultural experiments and expanded by George Box, Genichi Taguchi, and others, DOE has evolved into a fundamental methodology across engineering and sciences [20, 42, 94].

The fundamental objective of DOE is to efficiently establish relationships between input factors (parameters) and output responses of a system, enabling identification of the most influential factors affecting responses, determine optimal conditions for processes or systems, reduce variability while improving robustness, and minimize resource requirements. By strategically planning experimental configurations and loading conditions, optimal experimental design reduces model uncertainty while enhancing the reliability of subsequent analyses. Therefore, it is particularly critical in structural dynamics where key parameters such as damping ratios, stiffness coefficients, and modal frequencies must be estimated with high accuracy.

Core Principles DOE relies on several fundamental principles that control experimental variability and enable valid statistical conclusions:

- *Randomization*: Randomly allocating experimental units to different treatment conditions helps average out effects of uncontrolled or unknown variables, reducing systematic bias [42].
- *Replication*: Repeating experiments under identical conditions enables estimation of experimental error and increases precision of results. Replication allows for quantification of variability inherent in the experimental process [20].
- *Blocking*: Grouping experimental units into homogeneous blocks reduces influence of known but uncontrollable sources of variation, thereby improving precision. By accounting for systematic differences between blocks, this technique isolates the effects of primary interest from noise factors.
- *Orthogonality*: Orthogonal designs allow independent estimation of factor effects without interference, facilitating clear interpretation of results and simplifying analysis.

3.1.1 Classical Design of Experiments

Traditional DOE approaches assume all experiments are planned before data collection begins, following predetermined designs that distribute sample points according to fixed

criteria. These static batch methods have proven effective for many applications but cannot adapt based on observed results during the experimentation process.

Factorial Designs Full factorial designs investigate all possible combinations of factor levels, providing comprehensive coverage of the experimental space. For k factors each at l levels, a full factorial design requires l^k experiments. While providing complete information about main effects and interactions, the number of required experiments grows exponentially, often becoming impractical for systems with many factors [94].

Fractional factorial designs address this limitation by examining only a strategically selected subset of the full factorial combinations. These designs are based on the *sparsity of effects principle*, which assumes that systems are typically dominated by main effects and low-order interactions. Under this assumption, higher-order interactions are negligible. Fractional designs sacrifice information about higher-order interactions to reduce experimental cost [20, 143]. These designs are characterized by their resolution, which determines which effects are mixed with one another.

Response Surface Methodology Building upon factorial designs, Response Surface Methodology (RSM) employs statistical techniques to explore relationships between several input variables and one or more response variables, proving particularly useful for process optimization. Developed by Box and Wilson [21], RSM typically involves sequential experimentation including screening designs to identify important factors, steepest ascent procedures to approach the optimum region, and second-order designs to characterize the response surface near the optimum [94]. While RSM incorporates sequential stages, each stage itself consists of a predetermined batch of experiments.

Screening Designs When many potential factors must be evaluated with limited resources, screening designs efficiently identify the most influential variables among a large set of candidates. These designs prioritize the detection of main effects over the estimation of interactions. Plackett-Burman designs, which require $n = 4k$ runs for up to $n - 1$ factors, and definitive screening designs, which can estimate main effects and quadratic effects while providing partial information on two-factor interactions, provide economical approaches for this purpose [94].

Taguchi Methods Taguchi methods emphasize designing products and processes that are insensitive to noise factors. By employing orthogonal arrays that systematically combine controllable design factors with noise factors in the experimental layout, Taguchi methods aim to identify parameter settings that minimize sensitivity to variation. This approach introduces the concept of the signal-to-noise ratio as a performance metric to simultaneously optimize both the mean response and its variability [133].

While these classical DOE methods provide systematic frameworks for experimental planning, they share a fundamental limitation, which requires the experimental strategy to be

fully specified before data collection begins and cannot adapt based on observed results within a given stage. Although RSM incorporates multiple sequential stages, the experiments within each stage are predetermined. This constraint can lead to inefficient resource allocation when the response surface exhibits unexpected complexity or when certain regions prove more informative than anticipated. For the vibration control device testing addressed in this thesis, where each experiment is expensive and time-consuming, adaptive strategies that can adjust sampling based on accumulated knowledge offer significant potential advantages.

3.2 Space-Filling Sampling Methods

Space-filling designs aim to distribute sample points uniformly across the experimental domain, providing good coverage without requiring prior knowledge of the response surface. Unlike classical DOE methods that assume random experimental error and focus on factor-level combinations, space-filling designs are specifically developed for deterministic computer experiments or physical tests where the primary goal is to obtain a comprehensive representation of the system behavior with minimal experimental cost [120, 122].

The fundamental principle underlying space-filling designs is to distribute experimental points such that they provide maximum information about the entire experimental region while avoiding redundant or poorly distributed sampling. This is particularly crucial in structural dynamics applications where computational models or physical experiments may be expensive to evaluate, and the response surface may exhibit complex, nonlinear behavior across the parameter space. The space-filling methods serve as good baseline comparison strategies, representing best practices for non-adaptive sampling.

3.2.1 Random Sampling

Uniform random sampling generates points independently distributed across the domain, ensuring unbiased coverage. For a design space $\mathcal{X} = [a_1,b_1] \times [a_2,b_2] \times \cdots \times [a_d,b_d]$ in d dimensions, a random sample of size N consists of points $\mathbf{x}_k$ where each component is independently drawn as

$$x_{k,j} \sim \mathcal{U}(a_j,b_j), \quad \forall k \in \{1,2,\ldots,N\}, \quad j \in \{1,2,\ldots,d\} \tag{3.1}$$

where $\mathcal{U}(a_j,b_j)$ denotes the uniform distribution on the interval $[a_j,b_j]$. In practice, samples are generated using pseudo-random numbers $u \sim \mathcal{U}(0,1)$ and transforming them via

$$x = a + (b - a)u. \tag{3.2}$$

Random sampling is straightforward to implement and imposes no structural constraints on sample placement within the domain. The method is widely used in Monte Carlo

uncertainty quantification [27], machine learning [18], and global optimization [48]. However, for finite sample sizes, random sampling often produces clusters and gaps in the design space, potentially missing important regions. The expected number of samples required to achieve uniform coverage grows exponentially with dimensionality, making random sampling inefficient for high-dimensional problems.

3.2.2 Latin Hypercube Sampling

Latin Hypercube Sampling (LHS), introduced by McKay et al. [89], improves upon random sampling by ensuring better space-filling properties through stratification. For N samples in d dimensions, LHS divides each dimension into N equally probable intervals and samples exactly once from each interval. This guarantees that no two samples share the same interval in any dimension, ensuring uniform one-dimensional projections. Random sampling does not provide this guarantee for finite samples. Algorithm 1 shows how an LHS design matrix $\mathbf{X} \in \mathbb{R}^{N \times d}$ is constructed.

Algorithm 1 Latin Hypercube Sampling

Require: Number of samples N, dimension d, bounds [$\mathbf{a}$,$\mathbf{b}$]
Ensure: LHS design matrix $\mathbf{X} \in \mathbb{R}^{N \times d}$
for each dimension $j = 1, \ldots, d$ **do**
  Generate random permutation π_j of $\{1,2,\ldots,N\}$
  for each sample $k = 1, \ldots, N$ **do**
    Draw $u_{k,j} \sim \mathcal{U}(0,1)$
    Compute $x_{k,j} = \frac{\pi_j(k) - u_{k,j}}{N}$ ▷ Normalized coordinate
    Scale to bounds: $x_{k,j} \leftarrow a_j + (b_j - a_j) \cdot x_{k,j}$
  end for
end for
return $\mathbf{X} = [x_{k,j}]$

Optimized LHS Basic LHS ensures stratification but does not guarantee optimal space-filling in multidimensional space. The *maximin distance criterion* addresses this limitation by maximizing the minimum inter-point distance as

$$\mathcal{D}_{\text{maximin}} = \max_{\text{LHS}} \left\{ \min_{1 \le i < j \le N} \|\mathbf{x}_i - \mathbf{x}_j\|_2 \right\}. \tag{3.3}$$

Alternatively, the minimum squared Euclidean distance can serve as an optimization objective [96] as

$$d^2_{\min}(\mathbf{X}) = \min_{1 \le i < j \le N} \|\mathbf{x}_i - \mathbf{x}_j\|_2^2. \tag{3.4}$$

Various metaheuristic algorithms have been developed to construct optimized LHS designs, including simulated annealing [96], genetic algorithms [15], and enhanced stochastic

evolutionary methods [64]. These algorithms iteratively perturb an initial random LHS while maintaining the Latin hypercube constraint. The optimization is computationally challenging due to the discrete Latin hypercube constraint, the non-convex objective function, and the exponentially large search space.

LHS offers several advantages over random sampling. Each dimension is fully stratified, ensuring representative coverage across all input variables. Marginal distributions are uniform by construction, providing balanced one-dimensional projections. The sample size can be chosen independently of dimensionality, and LHS typically requires fewer samples than random sampling to achieve comparable coverage quality.

3.2.3 Sobol Sequences

Sobol sequences, introduced by Sobol [129], are quasi-random low-discrepancy sequences that provide more uniform multidimensional coverage than pseudorandom sampling methods. Unlike LHS, which ensures uniformity primarily in one-dimensional projections, Sobol sequences optimize multidimensional uniformity through carefully constructed point sets based on primitive polynomials over finite fields and corresponding direction numbers.

The quality of space-filling designs can be quantified through *discrepancy*, which measures how uniformly points are distributed compared to a theoretical uniform distribution. For Sobol sequences with N points in d dimensions, the star-discrepancy satisfies

$$D_N^* = O\left(\frac{(\log N)^d}{N}\right). \tag{3.5}$$

This convergence rate is significantly better than the $O(N^{-1/2})$ rate expected for random sampling, particularly for moderate dimensions where $d < 20$ [99]. The superior convergence rate translates to more efficient exploration of the design space and faster convergence for numerical integration and surrogate modeling tasks.

Construction Sobol sequences are constructed using the radical inverse function based on different prime bases for each dimension. For dimension j, the sequence uses a primitive polynomial over $\mathbb{F}_2$ (the finite field with two elements) and corresponding direction numbers $\{v_{j,k}\}_{k=1}^{\infty}$. The i-th point in dimension j is computed through bitwise operations on the binary representation of i as

$$x_{i,j} = \bigoplus_{k=1}^{m} b_k(i) \cdot v_{j,k}, \tag{3.6}$$

where $b_k(i)$ denotes the k-th bit in the binary representation of i, $\oplus$ represents the bitwise XOR operation, and $m = \lceil \log_2(i) \rceil$ is the number of bits required to represent i. The direction numbers $\{v_{j,k}\}$ must satisfy recurrence relations derived from primitive polynomials to ensure the low-discrepancy property [23].

Modern implementations use precalculated direction numbers that have been optimized to ensure good two-dimensional projections up to thousands of dimensions. These optimized direction numbers are essential for practical applications, as poor choices can lead to poor performance in higher dimensions.

Sobol sequences offer several advantages over random and Latin hypercube sampling. They provide superior multidimensional uniformity compared to random or pseudorandom sequences and ensure reproducibility through deterministic generation without random seed dependence. New points can be added while maintaining low-discrepancy properties, enabling sequential sampling if needed. The sequences maintain uniformity in all dimensional projections and achieve computational efficiency through bitwise operations.

While Sobol sequences excel for dimensions up to approximately 20, their advantage diminishes in very high-dimensional spaces. For structural dynamics applications in this thesis with typically fewer than 10 design variables (e.g., excitation amplitude and frequency), Sobol sequences remain highly effective. In the validation studies presented in Chapters 5 and 6, Sobol sequences serve as a robust baseline representing state-of-the-art space-filling methods. Their deterministic nature ensures reproducibility, while their superior uniformity makes them an appropriate benchmark for non-adaptive strategies. Outperforming Sobol sampling with equal experiment counts would demonstrate clear value of the proposed adaptive framework.

3.2.4 Comparison of Space-Filling Methods

The selection among space-filling methods depends on problem-specific considerations. Regarding dimensionality, all three methods perform adequately for low-dimensional problems ($d \leq 5$), with Sobol sequences offering marginal advantages. For moderate dimensions ($5 < d < 20$), Sobol sequences and optimized LHS provide superior coverage compared to random sampling. When exact reproducibility is essential (e.g., for benchmarking), Sobol sequences offer deterministic generation, whereas LHS and random sampling require careful seed management. If the sample size needs to be increased after initial experiments, Sobol sequences naturally accommodate additional points while maintaining low discrepancy. Random sampling requires minimal implementation effort, LHS offers a good balance between simplicity and performance, and Sobol sequences require access to direction number tables but are widely available in numerical libraries.

For the vibration control device testing addressed in this thesis, experimental parameters typically form a low-dimensional design space. In this regime, all three methods can provide adequate coverage, making them suitable as baseline comparisons for evaluating adaptive sampling approaches. However, these methods share a limitation that motivates the development of adaptive strategies. Space-filling methods allocate equal resources to all regions regardless of their importance for model accuracy. This can be inefficient when certain regions exhibit more complex response characteristics.

3.3 Active Learning

Active learning represents a paradigm shift from static experimental designs to dynamic, model-informed sampling strategies. Rather than specifying all experiments in advance following a predetermined plan, active learning iteratively selects new experiments based on information gained from previous results. This approach focuses experimental effort on regions expected to yield maximum improvement in model performance, making it particularly valuable when experiments are expensive, time-consuming, or resource-intensive.

3.3.1 Active Learning Paradigm

Active learning addresses contexts where obtaining labeled data requires significant cost, whether experimental effort, computational time, or human annotation, but the ability to strategically select which data points to label is available. The basic premise is that not all data points are equally informative for model improvement. By intelligently selecting which experiments to perform, active learning can achieve target model accuracy with substantially fewer experiments than passive sampling approaches [30, 124].

The active learning process typically follows an iterative cycle. First, a small set of initial experiments is performed, often using a space-filling design. A surrogate model is then trained on the available data. An acquisition function is defined to quantify the utility of potential new experiments based on the current model state. The experiment that maximizes this acquisition function is selected and performed. The model is updated with the new data, and the cycle repeats until a stopping criterion is satisfied, such as reaching a target accuracy or exhausting the experimental budget. This iterative process creates a feedback loop where each experiment is informed by all previous results, enabling more efficient exploration compared to static designs.

Active Learning Frameworks Active learning can be categorized into three main frameworks based on the availability of data and how the requests for data labels or *queries* are formulated [124].

Pool-based sampling assumes a large pool of unlabeled candidate points exists, from which the learner selects which points to label. This scenario is common when unlabeled data is abundant (e.g., images, sensor readings) but labeling requires expert effort. The pool-based approach allows for global optimization of the query strategy across all candidates but requires maintaining and searching through the candidate pool. Applications include structural health monitoring, where continuous sensor data are available but damage labels are expensive to obtain [25].

Stream-based sampling involves making real-time decisions about whether to query each data point as it arrives sequentially. The learner observes unlabeled instances one at a time and must immediately decide whether to request a label or discard the instance. This framework is suitable for online learning scenarios with continuous data streams, such as

real-time monitoring systems. The decision typically relies on an informativeness threshold, where points are queried if their expected information gain exceeds a predetermined level.

Query synthesis enables the learner to generate arbitrary query points anywhere in the continuous input space rather than selecting from existing candidates. For a design space $\mathcal{X}$, query synthesis solves

$$\mathbf{x}^* = \arg\max_{\mathbf{x}\in\mathcal{X}} A(\mathbf{x}), \tag{3.7}$$

where $A(\mathbf{x})$ is the acquisition function quantifying the expected utility of querying at $\mathbf{x}$. This optimization is performed over the continuous space $\mathcal{X}$, enabling queries at any feasible location.

For physical experiments with vibration control devices, the framework most closely resembles query synthesis. Excitation parameters such as amplitude, frequency, and duration can be specified continuously within feasible ranges. However, as discussed in Section 3.4, the problem is complicated by the *dynamic sampling challenge*. One cannot directly specify desired measurement points in the force-state space. Instead, excitation parameters must be optimized to drive the system through its unknown dynamics to achieve desired state coverage, which is fundamentally an inverse problem without closed-form solution.

3.3.2 Uncertainty-Based Query Strategies

The most prevalent active learning strategies use model uncertainty to guide sampling, operating on the principle that sampling in regions where the model is most uncertain will yield the greatest improvement. These approaches require probabilistic models or approximations that provide uncertainty estimates alongside predictions.

3.3.2.1 Gaussian Process Surrogates

Gaussian processes (GPs) provide probabilistic surrogate models that naturally quantify prediction uncertainty, making them particularly well-suited for uncertainty-based active learning [117]. A GP defines a distribution over functions specified by a mean function $\mu(\mathbf{x})$ and a covariance function (kernel) $k(\mathbf{x},\mathbf{x}')$ as:

$$f(\mathbf{x}) \sim \mathcal{GP}(\mu(\mathbf{x}),k(\mathbf{x},\mathbf{x}')). \tag{3.8}$$

Given observations $\mathcal{D} = \{(\mathbf{x}_i,y_i)\}_{i=1}^N$, the GP provides predictions with confidence intervals. For a new point $\mathbf{x}_*$, the predictive distribution is Gaussian as:

$$p(y_*|\mathbf{x}_*,\mathcal{D}) = \mathcal{N}(\mu_*(\mathbf{x}_*),\sigma_*^2(\mathbf{x}_*)), \tag{3.9}$$

where μ_* is the predictive mean and σ_*^2 quantifies prediction uncertainty. This uncertainty decreases near observed data and increases in unexplored regions, providing intuitive guidance for uncertainty-based exploration and active learning.

GP-based Bayesian optimization assumes the expensive function (experimental response) can be accurately modeled by a GP surrogate. For complex nonlinear dynamics, particularly systems with auxiliary masses exhibiting intricate force-state relationships, this assumption often fails. The GP must accurately represent the complete system response surface across the force-state space. For highly nonlinear systems like TLCDs with fluid-structure interaction or TMDs with nonlinear springs and friction, the GP may require prohibitive amounts of data to achieve adequate fidelity. Poor surrogate accuracy leads to suboptimal sampling decisions.

Building a GP surrogate of complete system response requires multi-dimensional output modeling for displacement, velocity, and force over time. For a time series with T samples and d state variables, the output dimension is $T \times d$. Multi-output GPs significantly increase complexity and computational cost, with kernel matrix operations scaling as $O((Td)^3)$. Standard GP operations scale as $O(N^3)$ with sample size N due to matrix inversion, limiting applicability for large datasets. Furthermore, GP performance critically depends on kernel choice, and standard kernels may poorly represent structural dynamics responses with resonances, nonlinear stiffness regimes, and complex damping behaviors.

3.3.2.2 Acquisition Functions

Acquisition functions formalize the trade-off between *exploration*, sampling uncertain regions to improve global model quality, and *exploitation*, sampling near-optimal regions to refine the optimum. Common acquisition functions include the following.

Expected Improvement (EI) For optimization problems, expected improvement measures the expected amount by which a new sample will improve over the current best observation f_{best} [65] as:

$$\text{EI}(\mathbf{x}) = \mathbb{E}[\max(f_{\text{best}} - f(\mathbf{x}),0)]. \tag{3.10}$$

For a Gaussian predictive distribution with mean $\mu(\mathbf{x})$ and standard deviation $\sigma(\mathbf{x})$, this can be computed analytically as:

$$\text{EI}(\mathbf{x}) = (f_{\text{best}} - \mu(\mathbf{x}))\Phi(Z) + \sigma(\mathbf{x})\phi(Z), \tag{3.11}$$

where $Z = (f_{\text{best}} - \mu(\mathbf{x}))/\sigma(\mathbf{x})$, $\Phi(\cdot)$ is the standard normal cumulative distribution function, and $\phi(\cdot)$ is the standard normal probability density function.

Upper Confidence Bound (UCB) UCB balances mean prediction and uncertainty through an explicit parameter [131] as:

$$\mathrm{UCB}(\mathbf{x}) = \mu(\mathbf{x}) + \beta\sigma(\mathbf{x}), \tag{3.12}$$

where $\beta > 0$ controls the exploration-exploitation trade-off. Larger β favors exploration of uncertain regions, while smaller β focuses on exploitation of regions with good predicted performance.

Probability of Improvement (PI) PI computes the probability that a new sample will improve over the current best as:

$$\mathrm{PI}(\mathbf{x}) = \Phi\left(\frac{f_{\text{best}} - \mu(\mathbf{x}) - \xi}{\sigma(\mathbf{x})}\right), \tag{3.13}$$

where $\xi \geq 0$ is a small exploration parameter that encourages sampling points slightly away from the current best. While computationally simple, PI tends to be more conservative than EI, often focusing on high-probability improvements rather than potentially large but uncertain gains.

Entropy-Based Criteria Information-theoretic approaches select points that maximize expected information gain, measured through entropy reduction. For model parameters θ, the expected information gain from observing y at point $\mathbf{x}$ is [56]:

$$\mathrm{IG}(\mathbf{x}) = H[\theta] - \mathbb{E}_{y|\mathbf{x}}[H[\theta|y,\mathbf{x}]], \tag{3.14}$$

where $H[\cdot]$ denotes differential entropy. Entropy Search (ES) and its variant Predictive Entropy Search (PES) aim to reduce uncertainty about the location of the global optimum rather than directly minimizing the objective function [57].

These acquisition functions have proven effective for many computational and experimental optimization problems. However, they face limitations when applied to physical testing of dynamic systems with deterministic neural network models, as detailed in Section 3.4.

3.3.3 Ensemble-Based Approaches for Neural Networks

Deterministic neural networks, such as standard FNNs and vanilla Neural ODEs, do not provide built-in uncertainty estimates. To enable uncertainty-based active learning with these models, ensemble methods approximate model uncertainty through prediction variance across multiple trained models.

Ensemble Construction An ensemble consists of M models $\{f_m(\mathbf{x};\theta_m)\}_{m=1}^{M}$ trained with different random initializations of network parameters, bootstrap samples of the

training data, architectural variations such as different numbers of neurons or layers, or training procedures including different optimization algorithms or hyperparameters.

The ensemble prediction is typically the mean as:

$$\bar{f}(\mathbf{x}) = \frac{1}{M} \sum_{m=1}^{M} f_m(\mathbf{x}; \theta_m), \tag{3.15}$$

and prediction variance approximates model uncertainty as:

$$\sigma^2_{\text{ensemble}}(\mathbf{x}) = \frac{1}{M} \sum_{m=1}^{M} (f_m(\mathbf{x}; \theta_m) - \bar{f}(\mathbf{x}))^2. \tag{3.16}$$

This variance can then be used in acquisition functions similar to GP-based approaches, treating $\sigma_{\text{ensemble}}(\mathbf{x})$ as a proxy for model uncertainty.

Two distinct strategies exist for utilizing multiple trained models. *Model selection* involves training multiple models in parallel, typically with different random initializations, and selecting the single best performer based on validation metrics. This approach provides robustness against poor local minima during optimization and allows exploration of the parameter space from different starting points. The computational cost during training is distributed across parallel processes, while deployment requires only the resources of a single model. However, only one model's knowledge is utilized, potentially discarding valuable information captured by other models.

Ensemble averaging combines predictions from multiple models, typically through simple averaging or weighted voting. This reduces prediction variance by averaging out individual model fluctuations, often improving generalization [17]. The diversity among ensemble members, which is essential for realizing these benefits, can be achieved through different initializations, architectures, training procedures, or data sampling strategies. However, ensemble methods require substantially increased computational and memory resources during training, proportional to ensemble size, and may reduce model interpretability.

Neural Network Ensemble Limitations For deterministic neural networks, uncertainty quantification through ensembles introduces several challenges. High prediction variance may indicate either lack of training data (epistemic uncertainty that active learning aims to address) or inherent model limitations such as insufficient capacity or poor architecture choice. The uncertainty measure cannot distinguish between these cases, potentially guiding sampling to regions where the model cannot improve regardless of additional data.

Additionally, prediction variance from an ensemble includes both epistemic uncertainty and aleatoric variance from the training process due to initialization, stochastic optimization, and gradient noise. This training variance component does not reflect true epistemic

uncertainty and contaminates the uncertainty estimate, potentially misleading sampling decisions.

Reliable uncertainty quantification requires large ensembles for stable variance estimates or Bayesian neural networks with sampling-based inference. This computational overhead is particularly problematic when each network training already requires substantial time. For Neural ODEs, each forward pass requires numerically solving ODEs and each backward pass requires solving adjoint equations, making ensemble training expensive for real-time iterative experimental design.

3.4 The Dynamic Sampling Problem

While the methods presented above have proven effective for many computational experiments and certain types of physical testing, they face limitations when applied to experimental testing of dynamic systems with neural network models.

Traditional DOE and most AL methods assume *static sampling*, where desired input conditions can be directly specified and corresponding outputs measured. For example, in materials testing one can directly set temperature and pressure to target values and measure resulting properties. Similarly, computational experiments allow evaluation at any chosen parameter values without additional complexity. Vibration control device testing, however, often requires *dynamic sampling* through unknown system dynamics. As illustrated in Fig. 3.1, one cannot directly specify desired state combinations such as particular displacement-velocity pairs. Instead, experiment parameters including excitation signals must be designed to drive the system through its unknown dynamics to achieve desired measurement states.

For example, when modeling the force-state relationship of a TMD, the goal is to obtain measurements covering the operational range of displacements and velocities to train a neural network model. However, the displacement $x(t)$ and velocity $\dot{x}(t)$ at any time depend on the excitation history and the unknown system dynamics being learned. Different excitation signals produce different state trajectories through the force-state space, and no direct mapping exists from desired force-state pairs to required excitation parameters.

Hence, for dynamic systems, the sampling problem transforms into the optimization of experimental parameters rather than direct selection of measurement points in the model space. This distinction between sampling space (experimental parameters) and model space is crucial for understanding the problem and developing a sampling strategy, as shown in Fig. 3.1. Effective data generation requires optimization in the sampling space to achieve the desired coverage in the model space, governed by unknown nonlinear dynamics that are the subject of identification. This inverse problem structure distinguishes vibration device testing from many applications where active learning has been successfully applied.

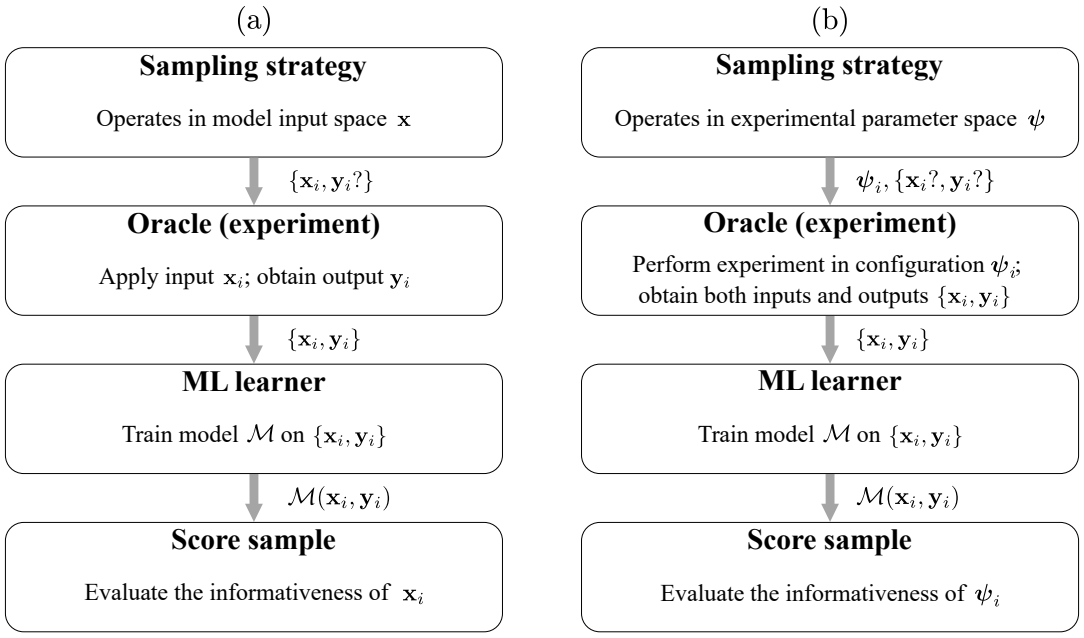

Figure 3.1: Comparison of sampling paradigms. (a) Static sampling where desired measurements can be directly specified and obtained. (b) Dynamic sampling where excitation parameters must be optimized to drive the system through unknown dynamics to achieve desired measurement coverage.

In the dynamic sampling, standard uncertainty-based active learning approaches face additional limitations when applied to physical experiments with deterministic neural networks. For example, even if uncertainty could be reliably quantified in the force-state space where the model operates, translating this into optimal excitation signal parameters remains profoundly challenging. This would require a forward model predicting which states will be reached by different excitation parameters, essentially a surrogate of the system dynamics, which is precisely what is being learned. This circular dependency requires building a surrogate of the surrogate, modeling both the system dynamics and the neural network model uncertainty simultaneously. The complexity of this nested modeling substantially diminishes the practical appeal of uncertainty-based query synthesis.

3.4.1 Performance-Based Optimization

The limitations detailed above suggest an alternative strategy that circumvents uncertainty quantification entirely by directly optimizing excitation parameters to minimize model error on a predefined test set representing the target application. This performance-based approach offers several advantages. It works with any machine learning architecture without requiring probabilistic extensions, directly minimizes the metric that matters (model error on representative test scenarios), and operates naturally in experimental parameter space without requiring inverse mapping from desired states to excitation signals.

This approach requires a predefined test set representing the operational range and

derivative-free optimization methods suitable for expensive, noisy, black-box functions. The test set requirement is usually satisfied in structural engineering applications, where vibration control devices undergo mandatory performance testing against standardized loading conditions before deployment [19]. Rather than treating performance testing as separate from model development, the proposed framework integrates these activities. The same loading conditions used to verify device performance serve as the test set guiding model training.

For optimization, derivative-free methods such as pattern search provide practical solutions, offering convergence guaranties while requiring minimal function evaluations. The deterministic nature of pattern search also ensures reproducibility of results, which is valuable for scientific validation. This optimization-based query synthesis framework, presented in detail in Chapter 4, provides a principled approach to adaptive experimental design that addresses the limitations of traditional AL methods while remaining practical for implementation in structural dynamics testing and performance evaluation.

3.5 Summary

This chapter presents experimental design methodologies for adaptive data generation in physical testing of dynamic systems, progressing from static to adaptive approaches. Space-filling methods, such as random or Latin hypercube sampling, provide systematic exploration of design spaces but cannot adapt based on experimental results. Active learning extends traditional DOE through adaptive strategies that iteratively refine sampling based on accumulated knowledge, with uncertainty-based methods like Bayesian optimization showing effectiveness in many computational domains. However, Gaussian process surrogates struggle with highly nonlinear structural responses and multi-dimensional outputs. For deterministic neural networks, ensemble-based uncertainty estimates cannot distinguish between insufficient data and model limitations. Furthermore, these approaches face challenges when applied to physical experiments with dynamic systems. The dynamic sampling problem prevents the direct specification of measurement points, requiring instead the optimization of experimental parameters.

These limitations motivate the optimization-based framework developed in Chapter 4, which directly optimizes excitation parameters to minimize model error on application-relevant test sets using derivative-free optimization. This performance-based approach operates in experimental parameter space, remains model-agnostic, and integrates mandatory performance testing with model development to create an efficient adaptive experimental design strategy for structural dynamics applications.

Chapter 4

Adaptive Data Generation for Testing-Integrated Modeling

This chapter presents a general methodology for autonomous and data-efficient experimental data generation in structural dynamics. Building on the theoretical background introduced in Chapters 2 and 3, where design of experiments and active learning methodologies were discussed, this chapter addresses a critical gap in existing approaches. Most established methodologies have been developed for cases where data is already available, such as an existing pool of data or stream of data. Furthermore, many methods are model-specific and cannot be readily transferred across different applications. An additional challenge, as explained in Section 4.1, is the requirement to sample dynamic data indirectly. In this chapter, the proposed framework, termed Adaptive Data Generation (ADaGen) is introduced.

The first step toward establishing a general methodology is to focus on SDOF systems with nonlinear, but memory-free restoring forces, such as vibration mitigation devices with auxiliary mass. This focus is motivated by the prevalence of such systems in civil engineering applications, particularly vibration mitigation devices, where accurate modeling of the restoring force is essential for control algorithm design and performance prediction. The methodology is presented within this constrained scope to ensure clarity and rigorous validation, with extensions to more complex systems addressed in Chapter 7.

The ADaGen framework, first conceptualized in [92], enables testing-integrated modeling by iteratively collecting experimental data to train machine learning models. The framework considers a device available for experimental testing with no pre-existing data or information. The data set must be generated from laboratory experiments, and the proposed framework aims to achieve two primary objectives:

1. To collect a measurement dataset containing sufficient information about the system in the desired operational range, such that an accurate machine learning model can be generated.

2. To minimize the number of necessary experiments.

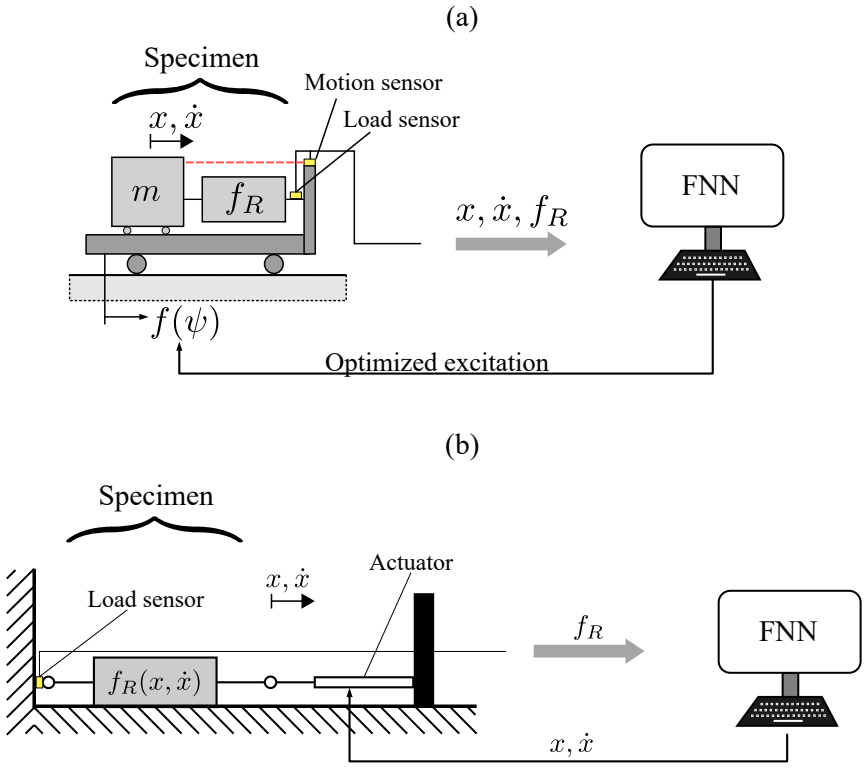

Figure 4.1: (a) Setup 1: A nonlinear oscillator is mounted on a shaking table and the restoring force-state mapping is learned by a FNN. The sampling strategy controls the excitation signal to generate response data required for optimum model training. (b) Setup 2: A nonlinear specimen is attached to an actuator. Similar to Setup 1, the restoring force-state mapping is being learned by an FNN.

4.1 Problem Setup and Scope

The proposed framework addresses the modeling of nonlinear oscillating systems with auxiliary mass, such as vibration mitigation devices or substructures mounted on shaking tables, as illustrated in Fig. 4.1a. Before presenting the mathematical formulation, the scope and underlying assumptions of the methodology are established. This chapter presents a baseline framework configuration for the testing setup described below. The framework is modular in design, meaning that the machine learning model can be replaced with alternative architectures as demonstrated in Chapter 6, and both the optimization algorithm and sampling strategy can be modified as explored in Chapter 7.

The framework is designed for indirect sampling scenarios where the restoring force cannot be directly commanded. As shown in Fig. 4.1a, an excitation signal must be applied to the system to drive it through the desired states, from which force-displacement-velocity data are measured. This contrasts with direct identification setups (Fig. 4.1b) where prescribed displacements and velocities directly yield restoring forces. The indirect nature of the problem, illustrated in Fig. 3.1, necessitates sampling in the space of excitation parameters $\boldsymbol{\psi}$ rather than in the state space, which distinguishes this approach from

standard active learning methods that assume direct access to the feature space.

The core assumptions underlying the ADaGen framework are:

(i) *System class.* The methodology targets SDOF systems with nonlinear but memory-free restoring forces. This restriction encompasses a wide range of vibration control devices in civil engineering, including tuned mass dampers, viscous dampers, and certain classes of base isolators. MDOF systems are not addressed in this baseline formulation, but are considered in Section 7.4.

(ii) *Test data availability.* A representative test dataset $\mathcal{D}_T$ must be available that spans the system's operational range. This test set typically originates from standardized performance evaluation protocols and serves as a benchmark for model evaluation throughout the adaptive process. The test set is established once prior to framework initialization and remains fixed throughout the optimization procedure. The case where this cannot be assumed is explored in Section 7.5.

(iii) *Information sufficiency.* The framework assumes that a single training dataset $\mathcal{D}_{train}(\boldsymbol{\psi}_{opt})$, generated through a single experiment with optimized parameters $\boldsymbol{\psi}_{opt}$, can contain sufficient information to model the system across the desired operating range when an appropriate ANN architecture is employed. This assumption enables the formulation of data generation as a static optimization problem over the experiment parameter space, rather than requiring cumulative data collection across multiple experiments. The validity of this assumption is demonstrated numerically in Chapter 5 and experimentally in Chapter 6. Furthermore, cumulative training approach is considered in Section 7.3.

(iv) *Black-box modeling.* Neural networks are initialized randomly without incorporating prior knowledge about the system. This choice ensures generality across different setups and enables fully autonomous data generation. Interpretable machine learning approach is explored in Section 7.2.

These assumptions collectively define a class of problems where adaptive experimental design can provide substantial efficiency gains over conventional testing strategies. The modular nature of the framework allows for relaxation or modification of these assumptions in specific applications, as demonstrated in subsequent chapters. With these assumptions established, the mathematical formulation of the problem can be presented.

4.1.1 Mathematical Formulation

The general equation of motion for the oscillator setup (Setup 1, cf. Fig. 4.1a) can be written as:

$$m\ddot{\mathbf{x}} + \mathbf{f}_R(\dot{\mathbf{x}},\mathbf{x}) = \mathbf{u}(\boldsymbol{\psi})\,, \tag{4.1}$$

where the mass m is assumed to be known, $\mathbf{x} \in \mathbb{R}^{N\times 1}$ is the displacement vector recorded at N time steps, a dot denotes the derivative with respect to time, and $\mathbf{f}_R \in \mathbb{R}^{N\times 1}$ is the restoring force including both linear and nonlinear components. The applied excitation signal $\mathbf{u} \in \mathbb{R}^{N\times 1}$ is generated according to the experiment parameters $\boldsymbol{\psi} \in \mathbb{R}^M$, where M represents the dimension of the experiment parameter space, also termed the sampling space. The experiment parameters $\boldsymbol{\psi}$ can include excitation signal characteristics such as amplitude, frequency, and excitation type, as well as testing conditions such as ambient temperature, specimen configuration, specimen orientation, and excitation application point. The dimension M defines the complexity of the optimization problem that must be solved.

The restoring force $\mathbf{f}_R$ is modeled with an ANN denoted as $\mathcal{M}(\dot{\mathbf{x}},\mathbf{x},\boldsymbol{\theta})$, where $\boldsymbol{\theta}$ is the matrix containing model parameters. As shown in Fig. 4.1a, the model is trained and assessed using data collected through experiments on the specimen. Both the training data $\mathcal{D}_i\{\dot{\mathbf{x}}_\mathbf{i},\mathbf{x}_\mathbf{i},\mathbf{f}_{\mathbf{R},\mathbf{i}}\}$ and the test data $\mathcal{D}_T\{\dot{\mathbf{x}}_\mathbf{T},\mathbf{x}_\mathbf{T},\mathbf{f}_{\mathbf{R},\mathbf{T}}\}$ consist of measured velocity and displacement responses with the corresponding restoring force. The problem is formulated as supervised learning, requiring a fully labeled data set.

In an ideal scenario for supervised learning, the training set $\mathcal{D}_{inf}$ contains sufficient information about the system's behavior in the range of interest such that a suitably chosen model $\mathcal{M}$ can achieve satisfactory accuracy when trained on $\mathcal{D}_{inf}$. However, the second requirement is to minimize experimental effort. The aim is to find a training set that contains sufficient information while containing as few data points as possible. Such a training set is referred to as the *optimal training data set* and denoted as $\mathcal{D}_{opt}$.

The training data set $\mathcal{D}_{opt}$ is collected by performing experiments such that when a model is trained using $\mathcal{D}_{opt}$ and tested on a separate comprehensive data set $\mathcal{D}_T$, it achieves modeling accuracy within an error tolerance of ϵ. The obtained model $\mathcal{M}$ then closely represents the behavior of the system in its operating range. Supplying excessive information might result in non-diverse data, such as disproportionately large amounts of data in the linear range, which can lead to suboptimal training of the model. Conversely, a very small dataset will lack the necessary information. The training dataset should therefore contain sufficient information about the behavior of the system in the operational range of interest, as the model quality is fundamentally limited by the provided training set.

The goal is to identify an optimal training set $\mathcal{D}_{opt}$ that encapsulates enough information to allow the model $\mathcal{M}$ to be trained successfully such that it meets modeling accuracy with an error tolerance of ϵ when evaluated on an independent test data set $\mathcal{D}_T$, while simultaneously minimizing the number of experiments i needed to find this optimal training data set.

4.2 Adaptive Data Generation Framework

To determine the optimal training dataset $\mathcal{D}_{opt}$ while minimizing the required number of experimental trials, the proposed framework converts the data generation process into a structured optimization problem. The model error evaluated on the test dataset is quantified through the adaptation cost function $\mathcal{L}_T$, where $\mathcal{L}_T : \mathbb{R}^M \longrightarrow \mathbb{R}$. This function serves as the objective function in the optimization problem. Each evaluation of $\mathcal{L}_T$ represents the model's assessment following a single experiment and subsequent training on the generated data.

As the training set $\mathcal{D}_i$ is not cumulative, the model $\mathcal{M}(\dot{\mathbf{x}},\mathbf{x},\boldsymbol{\theta})$ is trained using a single training set $\mathcal{D}_i$ of fixed size, where each set is generated using distinct experimental parameters $\boldsymbol{\psi}_i$. This design choice enables the adaptation cost function to be interpreted as a quantitative measure of the information content within a training dataset, where lower function values indicate datasets with higher informational value and greater learning potential. This interpretation aligns with information-theoretic principles, where the most informative datasets enable the model to achieve superior generalization performance with minimal training data.

The complete algorithmic procedure is presented in Algorithm 2. The framework begins with initialization of the test dataset and experimental parameters. At each iteration, an experiment is conducted with the current parameters $\boldsymbol{\psi}_i$, generating a training dataset $\mathcal{D}_i$. An ensemble of models is trained on this dataset to reduce training variance, and the best-performing model is evaluated on the test set to compute the adaptation cost function $\mathcal{L}_T$. If the error tolerance is met, the algorithm terminates with the optimal training dataset $\mathcal{D}_{opt}$. If the maximum number of iterations is reached or other stopping criteria are satisfied without meeting the error tolerance, the algorithm terminates with the best training dataset found but does not designate it as optimal.

Several considerations must be acknowledged regarding the framework's optimization characteristics and practical implementation. The solution obtained through this methodology does not guarantee convergence to the global minimum of the adaptation cost function. The optimization process may settle into local minima, which is a common limitation of gradient-free optimization approaches in complex landscapes.

As illustrated in Fig. 4.2, the concept of $\mathcal{D}_{opt}$ should not be interpreted as a single, universally optimal training set. Rather, $\mathcal{D}_{opt}$ represents a family of training sets, each generated using excitation signal parameters $\boldsymbol{\psi}_{opt}$ that satisfy the condition $\mathcal{L}_T(\boldsymbol{\psi}_{opt}) < \epsilon$, where ϵ denotes the predefined tolerance threshold for acceptable model performance. This reflects the inherent complexity of the optimization landscape and acknowledges that multiple parameter configurations may yield satisfactory results within the specified bounds.

Algorithm 2 Adaptive Data Generation Framework for experimental parameter optimization

Require: Test parameters $\boldsymbol{\psi}_T$, error tolerance ϵ, sampling space dimension M, ensemble size N_{ens}, maximum iterations i_{max}

Ensure: Optimal training dataset $\mathcal{D}_{opt}$ and trained model $\mathcal{M}_{opt}$ (if tolerance is met), or best dataset found $\mathcal{D}_{best}$ and model $\mathcal{M}_{best}$

Conduct experiment with test parameters $\boldsymbol{\psi}_T$
Obtain test dataset $\mathcal{D}_T = \{\dot{\mathbf{x}}_T, \mathbf{x}_T, \mathbf{f}_{R,T}\}$
Initialize experimental parameters $\boldsymbol{\psi}_1 \in \mathbb{R}^M$ (randomly or based on prior knowledge)
Initialize model parameters $\boldsymbol{\theta}_0$
Set iteration counter $i \leftarrow 1$
Initialize best error: $\mathcal{L}_{best} \leftarrow \infty$
Set convergence flag `optimal_found` $\leftarrow$ `False`
while $i \leq i_{max}$ **and** `optimal_found` = `False` **do**
    Conduct experiment with parameters $\boldsymbol{\psi}_i$
    Obtain training dataset $\mathcal{D}_i = \{\dot{\mathbf{x}}_i, \mathbf{x}_i, \mathbf{f}_{R,i}\}$
    for $j = 1$ to N_{ens} **do**
        Initialize model $\mathcal{M}_j$ with random parameters $\boldsymbol{\theta}_{j,0}$
        Split $\mathcal{D}_i$ into training (80%) and validation (20%) subsets
        Train $\mathcal{M}_j$ on training subset using supervised learning
        Apply early stopping based on validation subset performance
    end for
    Select best model from ensemble: $\mathcal{M}_i \leftarrow \arg\min_j \text{ValidationError}(\mathcal{M}_j)$
    Evaluate $\mathcal{M}_i$ on test set $\mathcal{D}_T$
    Compute adaptation cost function: $\mathcal{L}_T(\boldsymbol{\psi}_i) = \text{Error}(\mathcal{M}_i, \mathcal{D}_T)$
    if $\mathcal{L}_T(\boldsymbol{\psi}_i) < \mathcal{L}_{best}$ **then**
        $\mathcal{L}_{best} \leftarrow \mathcal{L}_T(\boldsymbol{\psi}_i)$
        $\mathcal{D}_{best} \leftarrow \mathcal{D}_i$
        $\mathcal{M}_{best} \leftarrow \mathcal{M}_i$
    end if
    if $\mathcal{L}_T(\boldsymbol{\psi}_i) \leq \epsilon$ **then**
        $\mathcal{D}_{opt} \leftarrow \mathcal{D}_i$
        $\mathcal{M}_{opt} \leftarrow \mathcal{M}_i$
        `optimal_found` $\leftarrow$ `True`
    else
        Update parameters via Pattern Search: $\boldsymbol{\psi}_{i+1} \leftarrow \text{PatternSearch}(\mathcal{L}_T, \boldsymbol{\psi}_i)$
        $i \leftarrow i + 1$
    end if
end while

4.3 Training and Test Data Configuration

The proposed framework operates with two distinct datasets: a training dataset $\mathcal{D}_i$ and a test dataset $\mathcal{D}_T$. The framework assumes the availability of a specimen for performance testing prior to its operational deployment.

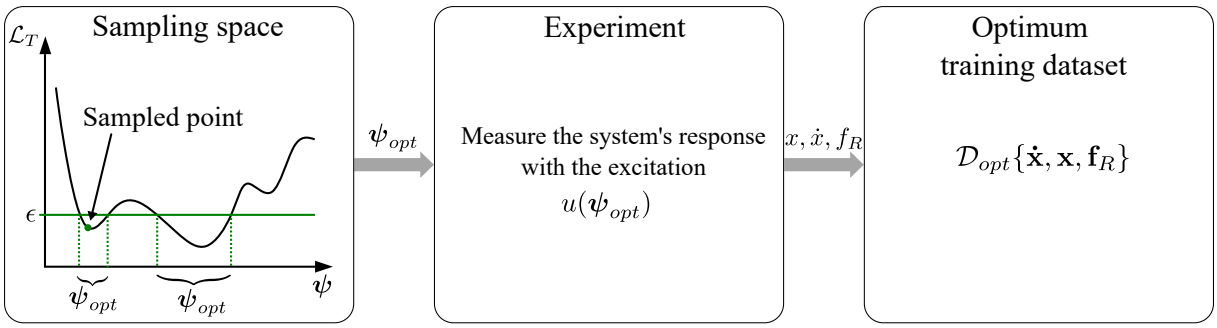

Figure 4.2: The optimal training dataset $\mathcal{D}_{opt}$ is produced by an excitation signal with parameters $\boldsymbol{\psi}_{opt}$ such that the adaptation cost function satisfies $\mathcal{L}_T(\boldsymbol{\psi}_{opt}) < \epsilon$ (cf. Algorithm 2).

4.3.1 Test Dataset

The test dataset $\mathcal{D}_T$ serves as a benchmark throughout the optimization process, enabling consistent evaluation of model accuracy across different training iterations. This dataset should ideally be derived from previous performance evaluation campaigns of the target specimen, reflecting realistic operational conditions and regulatory requirements. Performance tests are characterized by user-defined criteria encompassing loading conditions, specimen configurations, and operational constraints. Since these performance requirements are typically predetermined by application specifications and relevant codes, the experimental parameters for the test set $\boldsymbol{\psi}_T$ are known in advance.

The experimental parameters can be specimen-specific or general in nature, including excitation signal characteristics such as type, parameters, and application point, environmental conditions such as temperature, and measurement specifications such as sensor type and placement. For the purposes of this thesis, the analysis focuses exclusively on loading conditions for performance testing, enabling strategic evaluation of the proposed framework within a single configuration in Chapters 5 and 6. Further generalizations are addressed in Chapter 7. The test dataset corresponds to the operational range of the system and leverages existing experimental performance assessment procedures, such as earthquake time histories commonly used for testing control devices.

4.3.2 Training Dataset

The training dataset $\mathcal{D}_i$ is obtained through separate parameterized experiments designed specifically for model training purposes. During the training process, this dataset undergoes additional partitioning into training and validation subsets, with the ratio determined by the user and the splitting performed by the training algorithm. The data splitting strategy depends on the nature of the mapping being learned. For static mappings, input-output data pairs can be selected randomly from the training dataset. For dynamic, time-dependent systems, the training algorithm selects initial points and their corresponding prediction horizon points in a systematic manner to preserve temporal

dependencies.To ensure valid performance evaluation, the test dataset must remain independent from all training datasets. This independence is achieved by utilizing different types of experiment configurations, such as types of excitation signals, for generating the test and training responses, which is necessary for obtaining unbiased performance assessments and preventing overfitting during model evaluation.

4.3.3 Training Variance Reduction

As explained in Chapter 2, the bias-variance tradeoff is a major consideration in the ADaGen framework. Artificial neural networks with identical architectures and training data but different initial parameter values can yield substantially different test errors. Since the goal is to improve data quality and use model error as a source of information for error reduction, the training variance component must be reduced as much as possible. This is achieved through model selection or ensemble methods, where training is repeated multiple times.

The framework assumes that residual error after variance reduction is primarily due to data quality rather than training convergence issues. This assumption enables the interpretation of the adaptation cost function as a reliable measure of training dataset informativeness, which is essential for effective optimization of experimental parameters.

4.4 Pattern Search Optimization

The identification of an optimal training set $\mathcal{D}_{opt}$ while minimizing the required number of experiments is formulated as an optimization problem wherein the experimental parameters $\boldsymbol{\psi}_i$ are adjusted to minimize the model error $\mathcal{L}_T(\boldsymbol{\psi}_i)$. Given that specific parameter configurations yield more informative datasets, the error function exhibits at least one global minimum alongside potential local minima. The model error depends on both data quality and errors from bias and variance components. Therefore, the error structure is inherently stochastic, requiring the implementation of derivative-free optimization methodologies.

While various optimization algorithms designed for stochastic problems are compatible with the proposed framework, including particle swarm optimization [51] or genetic algorithms, such population-based methods typically require extensive sampling, which contradicts the goal of minimizing experiments. Population-based optimization methods might yield superior results but demand significantly more function evaluations, thereby increasing experimental costs. In contrast, Pattern Search algorithms have demonstrated efficiency in applications where function evaluations are computationally or experimentally expensive [12].

Pattern Search methods represent a well-established class of direct search algorithms with proven convergence properties under mild continuity assumptions [134]. At each iteration,

PS evaluates the objective function on a mesh of trial points constructed around the current best solution. The mesh is defined by a set of search directions and a step size parameter.

Within the proposed framework, the adaptation cost function $\mathcal{L}_T(\boldsymbol{\psi})$ is minimized, where $\boldsymbol{\psi}$ represents a candidate solution with dimension M corresponding to the number of design variables. At each iteration, PS generates a finite set of exploratory vectors $\boldsymbol{\psi}^e$, collectively referred to as a mesh, expressed as:

$$ {}^k\boldsymbol{\psi}_i^e = \boldsymbol{\psi}_i + \Delta \cdot {}^k\boldsymbol{v} \,, \tag{4.2} $$

where ${}^k\boldsymbol{v}$ represents a set of k vectors defining the search pattern, and Δ denotes the step length or mesh size. The mesh is centered at $\boldsymbol{\psi}$ and defined by the number of independent design variables M and the positive basis set.

For a two-dimensional problem with minimal basis and mesh size $\Delta = 1$, the mesh consists of $M + 1 = 3$ vectors:

$$ {}^1\boldsymbol{\psi}_{i+1}^e = \boldsymbol{\psi}_i + 1 \cdot \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \quad {}^2\boldsymbol{\psi}_{i+2}^e = \boldsymbol{\psi}_i + 1 \cdot \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \quad {}^3\boldsymbol{\psi}_{i+3}^e = \boldsymbol{\psi}_i + 1 \cdot \begin{bmatrix} -1 \\ -1 \end{bmatrix}. \tag{4.3} $$

The algorithm proceeds through iterative mesh updates based on trial point evaluations. When a trial point yields an improved solution satisfying $\mathcal{L}_T({}^k\boldsymbol{\psi}_{i+k}^e) < \mathcal{L}_T(\boldsymbol{\psi}_i)$, the mesh center is relocated to $\boldsymbol{\psi}_{i+k} = {}^k\boldsymbol{\psi}_{i+k}^e$, and the mesh is expanded with $\Delta = \tau\Delta$ where $\tau \geq 1$. Conversely, when trial points fail to produce improvement such that $\mathcal{L}_T({}^k\boldsymbol{\psi}_{i+k}^e) \geq \mathcal{L}_T(\boldsymbol{\psi}_i)$, the mesh contracts with $\Delta = \delta\Delta$ where $0 < \delta < 1$, while the solution candidate remains unchanged as $\boldsymbol{\psi}_{i+k} = \boldsymbol{\psi}_i$.

The algorithm terminates under three conditions: when the adaptation cost function reaches the specified tolerance $\mathcal{L}_T(\boldsymbol{\psi}_i) < \epsilon$, when the mesh size falls below the minimum threshold $\Delta < \Delta_{tol}$, or when the maximum number of function evaluations i is reached. Figure 4.3 illustrates the convergence behavior of PS algorithm, where the subscript indicates the total number of experiments and the superscript denotes the trial point being evaluated within the mesh.

For the proposed framework, the mesh should contain the minimum allowable number of points to minimize experimental requirements, as each objective function evaluation corresponds to one experiment. Additionally, when the objective function exhibits higher sensitivity to specific design parameters, the mesh should be scaled accordingly to implement smaller step sizes in sensitive directions. In this work, a constant mesh size is maintained and the sampling space is normalized, enabling Pattern Search operation in a standardized environment with a uniform mesh size.

The algorithm can be configured to evaluate all mesh points or employ a greedy approach where the mesh center immediately moves to the first trial point with an improved solution. To prevent redundant data collection, previously evaluated points are stored in memory.

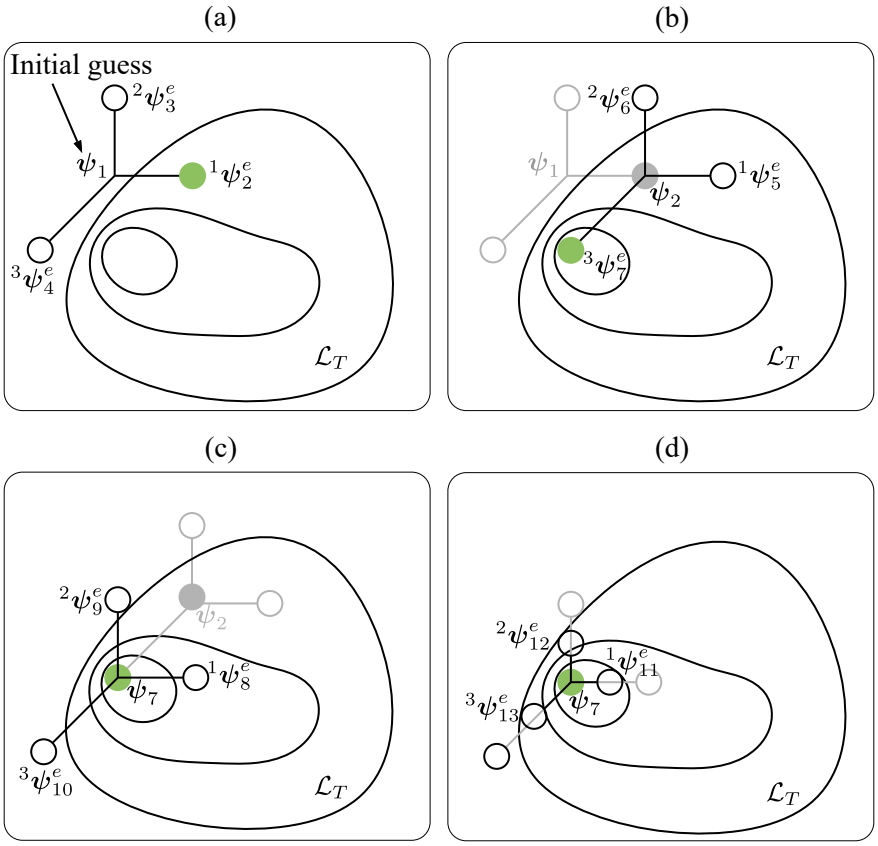

Figure 4.3: Example of convergence of pattern search optimization algorithm. (a) The initial guess $\boldsymbol{\psi}_1$ serves as both candidate solution and mesh center. Three trial points are evaluated. (b) The mesh is scaled and new trial points are evaluated. (c) No improved solution is found among trial points, maintaining the current candidate solution. (d) The mesh contracts and new evaluation points are generated.

Since the adaptation cost function is based on model performance on the test dataset, it includes variance error from the training process. For the adaptation cost function to be stable and reliable, training variance must be reduced using methods such as model selection (Chapter 2). Without this variance reduction, the adaptation cost function would include excessive noise that would prevent effective optimization.

The formulation presented here is adapted specifically for the proposed data generation framework, with theoretical foundations established in the literature through Generalized Pattern Search by Torczon [134] and Generative Set Search methods by Kolda et al. [71]. The mesh size Δ serves as a measure of convergence progress. When Δ becomes sufficiently small, $\boldsymbol{\psi}$ must be near a point where no search direction yields improvement, providing a natural stopping criterion without requiring explicit gradient computations. For non-smooth or discontinuous functions, which may arise from measurement noise in physical experiments, modified convergence results exist showing convergence to points where Clarke subdifferentials satisfy necessary optimality conditions [13].

4.5 Summary

This section introduces the ADaGen framework, which combines ANNs with an active-learning strategy that formulates data generation as an optimization problem, initially conceptualized in [92]. The framework assumes a dynamic experimental setup as the data source, such as a TMD mounted on a shaking table. Experimental conditions are represented by a set of parameters $\boldsymbol{\psi}$, where performing an experiment with parameters $\boldsymbol{\psi}_i$ yields a dataset $\mathcal{D}_i$. By sequentially varying these parameters, datasets are collected to iteratively improve their information content.

In each iteration, the model is trained on the gathered response data and evaluated on a fixed test set. For TMDs, suitable test data can be obtained from experiments prescribed by relevant codes, which specify testing conditions including harmonic excitation signals and representative earthquake records. The experimental parameters are then updated using an optimization algorithm to generate system responses that improve model accuracy. While any gradient-free optimization algorithm can be employed, the Pattern Search (PS) algorithm was selected for its sampling efficiency, as the objective is to obtain a fully informative dataset $\mathcal{D}_{\text{opt}}$ with the minimum number of experiments. The adaptive loop continues until the model prediction error reaches a specified tolerance, ensuring that sufficient information is captured about the system response.

Chapter 5

Numerical Validation Using FNNs

This chapter presents a numerical validation of the proposed ADaGen framework. The numerical validation is focused on modeling the nonlinear restoring force of oscillating systems, such as damping devices or substructures, using force-state mapping. In particular, the Duffing oscillator system is considered. The force-state map is approximated by an FNN, where the data is assumed to be obtained through shaking table testing of the specimen and this here only simulated, no actual experiments were performed. The FNN evaluates its accuracy on an independent data set, and requests, if necessary, more data in order to improve its modeling accuracy.

This chapter is primarily based on the insights and results of the publication by Milicevic and Altay [92] and demonstrates the proposed framework on a numerical example. In this case the chosen system is a Duffing oscillator, a dynamic system which is commonly used for benchmarking data-driven modeling methods, such as in [75]. In Sections 5.1.6 and 5.1.4, the proposed framework is then compared to several unsupervised sampling methods. In Section 5.2, an engineering application example is presented. The application example serves as a demonstration of how the modeled force-state map of a nonlinear TMD can be integrated and used in the simulation of a two-story shear frame. All examples presented in this section are simulated with Matlab [87].

5.1 Duffing Oscillator

The advantages of using a force-state mapping are that it fully describes the system dynamics. Furthermore, if the system has a local nonlinearity or is a nonlinear substructure within a larger structure, the modeled force-state behavior can be readily integrated in a numerical simulation [31]. For the Duffing oscillator—a canonical benchmark widely validated in the literature through both numerical studies and physical experiments including shaking table tests—substantial work exists on identification using methods such as sparse regression and genetic programming, though remarkably little addresses optimal experimental design with neural networks.

To generate the data, the system is first excited by some excitation signal $\mathbf{f}(\boldsymbol{\psi})$ and the velocity and displacement responses as well as the restoring responses are then collected by sensors. The displacements can be measured with laser sensors and velocity can be derived from the displacements. The restoring force measurement can be done with a load cell, such as in [38], or with strain gauges, which are calibrated to determine the shear force at the connection point with the shaking table [37]. As mentioned before, the operating range of the system is assumed to be known, such as from initial performance assessment tests, and the experiments are conducted in this range.

To evaluate the proposed methodology, the Duffing oscillator is used, since it is a typical system utilized to evaluate nonlinear approaches in civil engineering. In this example, the primary goal is to map the restoring force of the system given from the state response. The continuous form for the restoring force of the Duffing oscillator is given by:

$$f_R(x,\dot{x}) = \alpha\dot{x} + \beta x + \gamma x^3\,, \tag{5.1}$$

where the system is mass normalized, hence $m = 1$ and other parameters include the damping coefficient $\alpha = 1.25$; linear stiffness coefficient $\beta = (2\pi)^2$; and nonlinear stiffness coefficient $\gamma = 10$. The calculated linear frequency of the system is 1 Hz. The system is simulated in Matlab using 4th order Runge-Kutta method and with a time step of 0.01 s.

Figure 5.1 illustrates the normalized force-state mapping that will be approximated using the FNN $\mathcal{M}(\dot{\mathbf{x}},\mathbf{x},\boldsymbol{\theta})$. This response is calculated through Eq. (5.1) using a *sine sweep* input signal and serves as the benchmark for model evaluation in this section. The mapping clearly exhibits nonlinear characteristics, contrasting with linear systems where the force-state relationship would manifest as a planar surface, meaning that Fig. 5.1d would not exhibit the cubic nonlinearity. It should be emphasized, however, that direct sampling from this surface is not possible since the system states are not directly accessible. Consequently, data collection must be performed by subjecting the system to dynamic excitation, where both the resulting states and corresponding restoring forces remain unknown a priori. This sampling procedure occurs within a distinct space, as detailed further in Section 5.1.3.

5.1.1 FNN Configuration

The force-state mapping is approximated using an FNN architecture consisting of two hidden layers, with each layer containing two neurons. While more complex network structures could be implemented for this system modeling task, as shown in [86], such architectures would result in increased model variance and subsequently introduce additional noise into the problem. Given that an ensemble modeling approach is employed to mitigate this noise, adopting a more complex architecture would not yield substantial improvements in the proposed framework's performance, while significantly increasing the computational overhead during training. The network utilizes *hyperbolic tangent* activation functions and optimization is performed using *mean squared error* as the loss function.

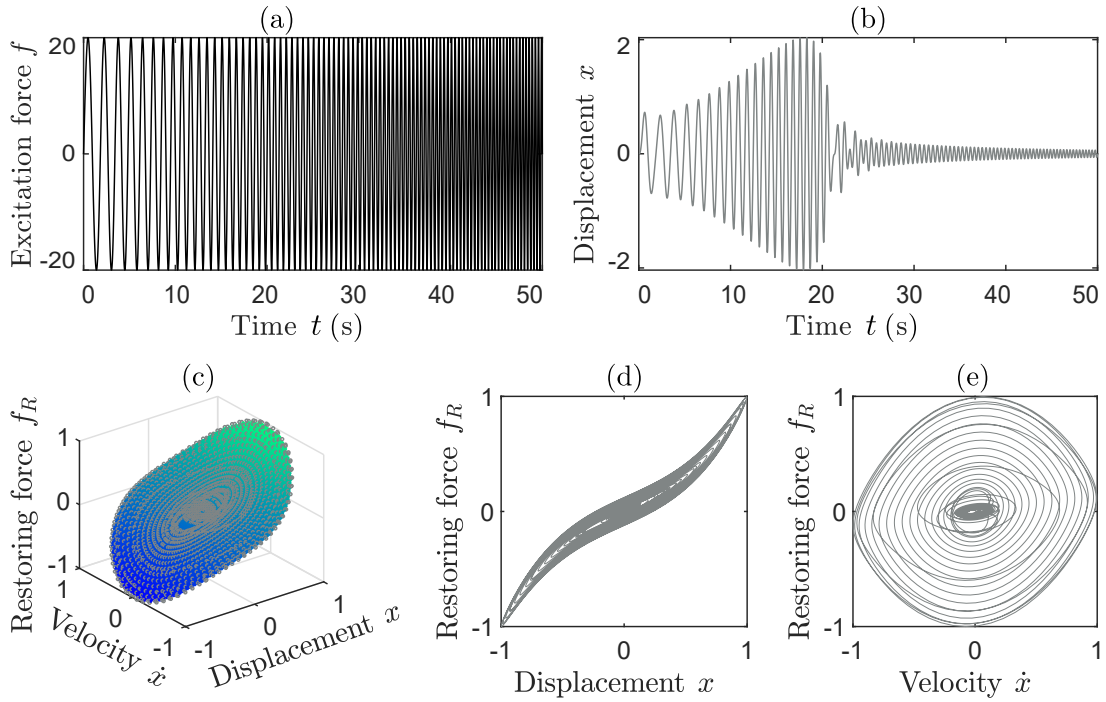

Figure 5.1: (a) Sine-sweep (chirp) excitation signal (b) analytically calculated displacement response of the Duffing oscillator; (c) normalized force-state map of the oscillator. The shading represents the values of the restoring force, ranging from −1 (dark) to 1 (light). (d) Normalized restoring force vs. displacement plot and **e** normalized restoring force vs. velocity. plot

Training is conducted via the *Levenberg-Marquardt* optimization algorithm, implemented through Matlab's *trainlm* function with default parameter settings. To improve the robustness of the training procedure and simulate potential measurement noise, 3% noise is incorporated into the restoring force data prior to the training.

A further important aspect involves minimizing the variance error that stems exclusively from the training algorithm, network architecture, and parameters of the selected FNN, as outlined in both Chapter 2 and Chapter 4. As illustrated in Fig. 5.2, for this study a *model selection pool* approach is used, where multiple FNNs are trained in parallel using the same datasets, with the network with the best modeling error being selected for dataset quality assessment. While some residual bias and variance errors will remain, their influence is assumed to be minimal. Hence, the final error is assumed to be a reliable source of information when comparing the quality of collected datasets. It is noteworthy that this example exclusively employs force-state mappings, hence the utilization of FNNs. Nevertheless, alternative modeling strategies exist that can characterize system response without direct measurement of displacement and restoring force, such as approaches that map excitation signals to displacement, velocity, and acceleration responses [73].

A model selection pool used in the examples in this section contains 10 model members. Each member model has the same structure and hyperparameters, but different training initialization, such as initial model parameters. During the training, 80% of the data is used as the training data set and 20% of the data is used as the validation data set. The

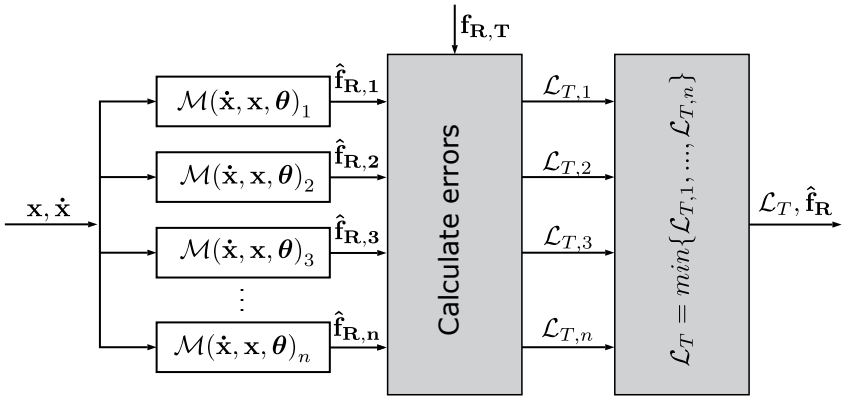

Figure 5.2: FNN model selection pool for obtaining the model with the most successful training; by using a model selection, information about the quality of the training data set has been made more reliable.

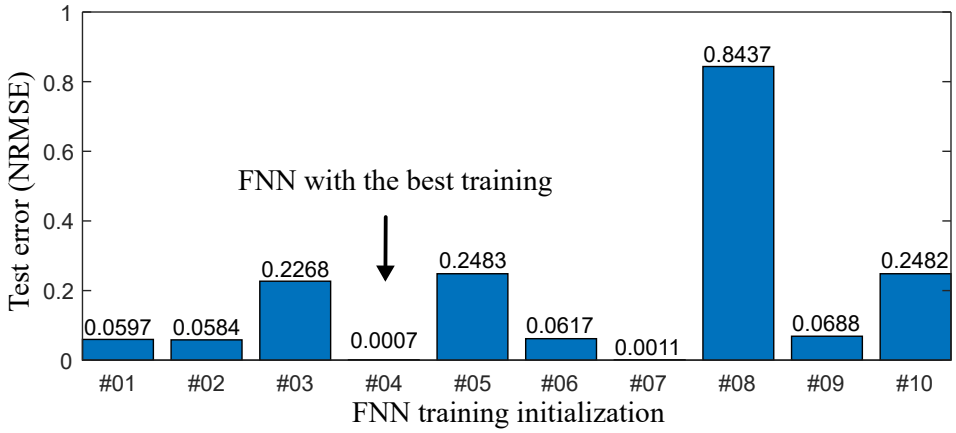

Figure 5.3: The same data set is used for the training of feedforward neural network models with different initialization, yielding different test errors. Using the model ensemble allows to choose the model with the best training.

model performance is evaluated with normalized root mean square error (NRMSE). The necessity to use the model ensemble is justified by training the model several times with the same training data set and different training initialization. From the results shown in Fig. 5.3, it can be seen that the variance in the results is vast, where the model with the best performance has the NRMSE of 0.0007 and the one with the worst performance has the NRMSE of 0.8437 which is orders of magnitude higher.

Furthermore, Figs. 5.4 and 5.5 show the impact of this training variance on the result and help understand the significance of utilizing the model selection procedure. Figs. 5.4 and 5.5 show the performances of the FNN models # 04, # 05 and # 08 on the sine-sweep test excitation signal (cf. Fig. 5.1a) respectively as time history and force vs. displacement plots. It is evident that a single network without training variance reduction cannot be used as a reliable source of information regarding the quality of the training dataset. This would yield an unreliable and noisy objective function which could

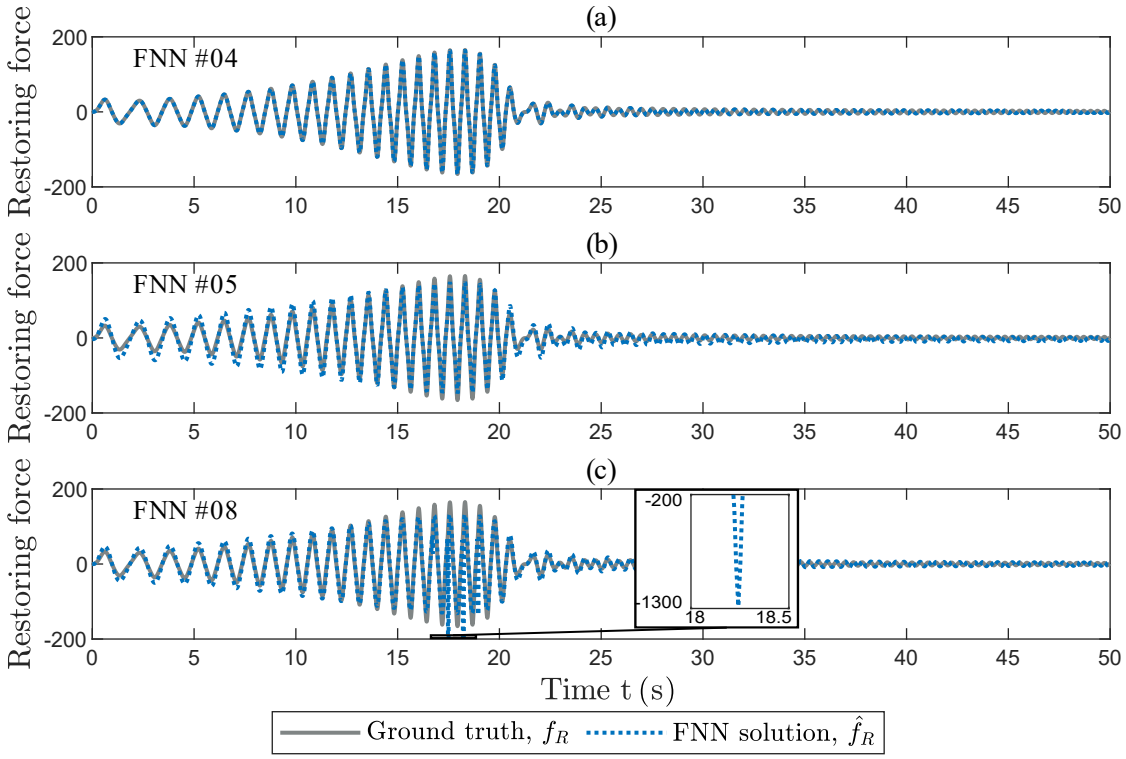

Figure 5.4: Restoring force time histories showing the performance of the FNN model on the test set with different training success, corresponding to the NRMSE of 0.0007 (a), 0.2483 (b) and 0.8437 (c), (cf. Fig. 5.3).

not be used for optimization to find the best experiment parameters.

5.1.2 Training and Test Datasets

Training datasets are generated using 20 s duration *sine signals* as excitation signals. Figure 5.6a shows a representative sine signal example. The signal is characterized by parameters $\boldsymbol{\psi} = [f_0 \;\; \Omega]^\top$, where f_0 represents the amplitude and Ω denotes the excitation frequency. It is worth noting that the response data encompasses the transient behavior of the oscillator, as illustrated in Fig. 5.6b. As a result, amplitude identification becomes less critical compared to steady-state response scenarios. Nevertheless, it must be stressed that the problem formulation assumes users possess no prior knowledge regarding the response characteristics of the tested specimen.

Alternative excitation signals with potentially superior properties, including pseudo-random binary signals (PRBS) and amplitude modulated pseudo-random binary signals (APRBS) [50, 98], could be employed for training set generation. However, sine signals are chosen for methodological demonstration, serving both educational purposes and ensuring a fair comparison with unsupervised approaches by eliminating signal-induced randomness that would arise from PRBS and APRBS excitation.

The test dataset is generated using a 50 s long *swept sine* excitation with an amplitude

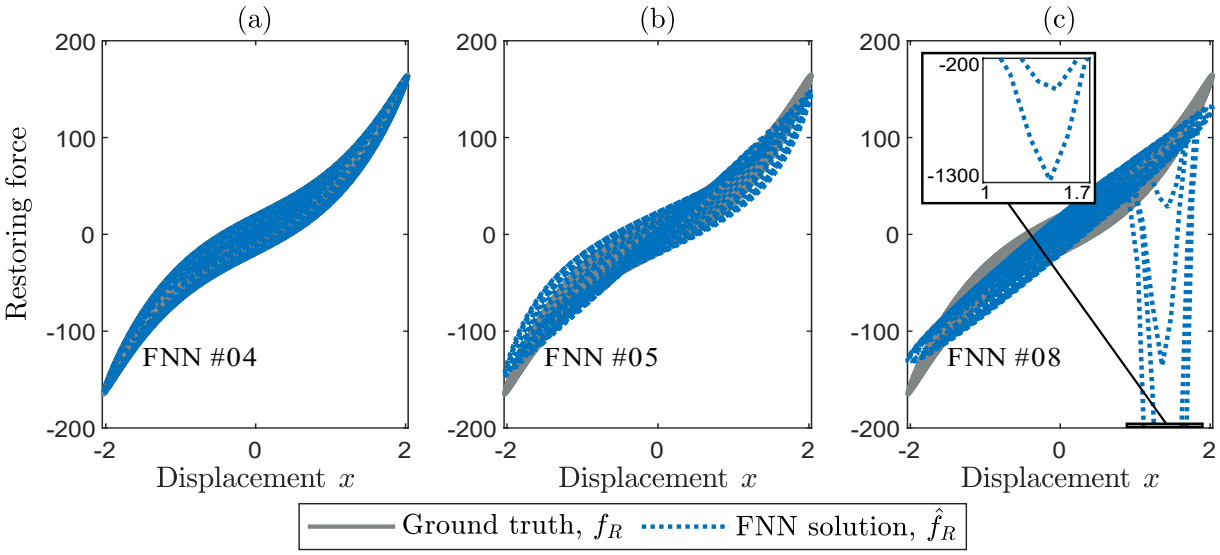

Figure 5.5: Force vs. displacement plots showing the performance of the FNN model on the test set with different training success, corresponding to the NRMSE of 0.0007 (a), 0.2483 (b) and 0.8437 (c), (cf. Fig. 5.3).

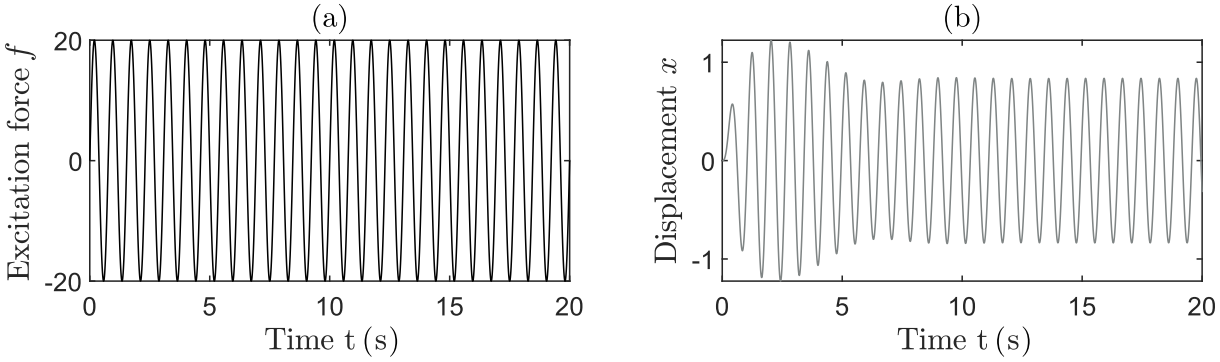

Figure 5.6: (a) An example of the sine excitation signal used to obtain the training data. Excitation amplitude and frequency are $f_0 = 20$ and $\Omega = 1.3$, respectively.(b) Analytically calculated corresponding displacement response of the Duffing oscillator.

of 20 and frequency sweep ranging from 0.5 Hz to 3 Hz. This test dataset is illustrated in Fig. 5.1a. It is clear that the selected test dataset encompasses the system's nonlinear behavior within its operational range, making it representative of the overall system characteristics. It should be emphasized that multiple test datasets could be implemented in this configuration to enhance model accuracy further. It should be highlighted that the test dataset remains fixed and unchanged throughout the entire proposed framework.

5.1.3 Sampling Space

Data sampling is performed within the excitation signal parameter space. Therefore, using sine signals for excitation results in a two-dimensional sampling space. When each

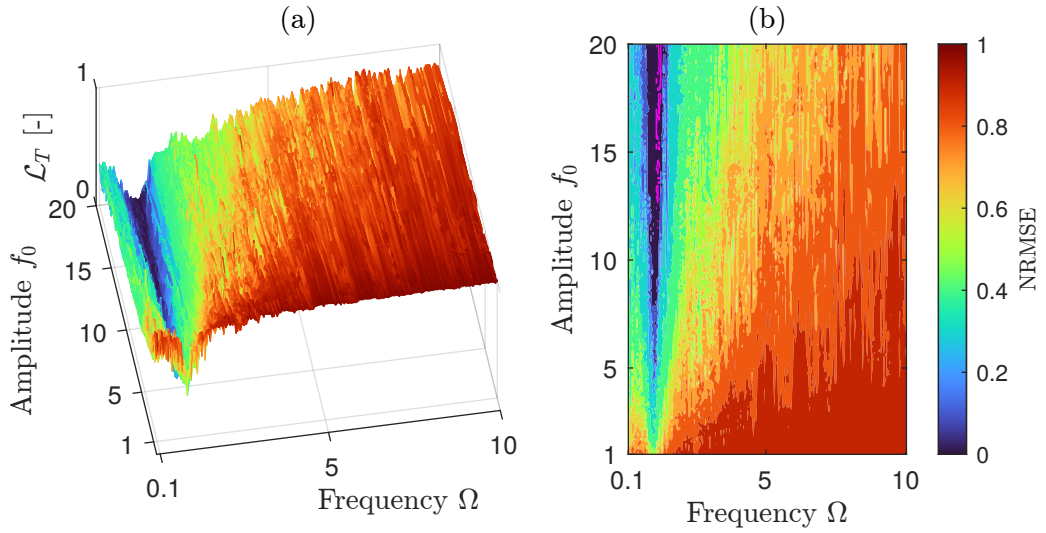

Figure 5.7: (a) Error surface produced by evaluating training sets on a uniform grid with respect to excitation signal parameters. The colorbar shows the normalized root mean square error. (b) Contour plot corresponding to the error surface; the surface is convex and there are excitation parameter combinations which can lead to optimally trained high accuracy models of the Duffing. oscillator

point in the sampling space is assigned a quality or information measure for sample $\boldsymbol{\psi}_i$, the adaptation cost function $\mathcal{L}_T$ is constructed. As previously introduced, this adaptation cost function corresponds to the NRMSE of the model assessed on the test dataset, following training with data obtained from the system's response under sine signal excitation characterized by parameters $\boldsymbol{\psi}_i$.

Figure 5.7 shows the error surface illustrating the adaptation cost function $\mathcal{L}_T$ as a function of excitation signal parameters $\boldsymbol{\psi}_i$ within the system's operating range of amplitudes $f_0 = [1 \dots 20]$ and excitation frequencies $\Omega = [0.1 \dots 10]$ Hz. This error surface is generated by evaluating each parameter set $\boldsymbol{\psi}_i$ in the operating range on a uniform grid using increments of 0.1 for amplitude and 0.05 for frequency.

Figure 5.7 shows that the surface exhibits convex characteristics with a global minimum, although it is overall not convex. In addition, the noise present in the surface shows that there is still a certain amount of error coming from both bias and variance components. However, the existence of this clear global minimum region confirms the feasibility of formulating an optimization problem to identify the optimal training dataset. Under low excitation amplitudes, the system exhibits linear behavior, and superior training data can be obtained using excitation frequencies corresponding to the system's linear natural frequency at approximately $\Omega = 1$ Hz. More accurate models with smaller prediction error values are achievable when training with higher amplitude data, where the system's nonlinear properties become more pronounced in the response data. In this regime, the optimal training dataset is possible for excitation frequencies within a broader frequency band compared to the low-amplitude linear case. It should be reiterated that error surface evaluation is not included in the proposed framework and serves purely for visualization

purposes. In the current problem formulation, both the error surface and oscillator parameters remain unknown and must be explored experiment-by-experiment.

The stochastic nature of the error surface provides justification for employing the proposed gradient-free optimization approach. It should be noted that modifications to the model architecture, hyperparameters, or training algorithm are expected to change the characteristics of the error surface. Furthermore, increased mismatch between the model architecture and the target system is anticipated to result in a more stochastic error surface due to higher training variance. However, the error surface shape reflects the underlying system characteristics and should remain consistent for the Duffing Oscillator. Based on the authors' experience, the error surface is likely to maintain its properties, such as having a global minimum, for systems exhibiting different nonlinear behaviors, including piecewise linear (PWL) systems, as these systems generate distinct responses when subjected to different excitation signals. Consequently, a set of excitation signal or general experiment parameters will always exist that produces more informative system responses. The error surface configuration will depend on system properties, including natural frequency and the particular excitation conditions that drive the system through its most informative states. For instance, when examining a hardening (PWL) system in this particular setup, optimal training signal parameter combinations must drive the system to experience both initial and hardening stiffness regimes while providing adequate coverage of the transition region to achieve the optimal training dataset. Any combination of excitation signal parameters satisfying these requirements would yield an optimal training dataset. For systems exhibiting more complex material behaviors, the proposed data generation framework may require adaptation to alternative neural network architectures. As an example, long short-term memory (LSTM) networks could be employed for systems with memory characteristics as mentioned in Chapter 2.

To further clarify the interpretation of Fig. 5.7, two distinct samples extracted from the error surface are examined. The first sample $\boldsymbol{\psi}_1 = [13.36 \quad 0.62]^\top$ generates the first training set $\mathcal{D}_1 = \{\dot{\mathbf{x}}_\mathbf{1}, \mathbf{x}_\mathbf{1}, \mathbf{f}_{\mathbf{R},\mathbf{1}}\}$. The second sample $\boldsymbol{\psi}_2 = [19.26 \quad 1.25]^\top$ produces the second training set $\mathcal{D}_2 = \{\dot{\mathbf{x}}_\mathbf{2}, \mathbf{x}_\mathbf{2}, \mathbf{f}_{\mathbf{R},\mathbf{2}}\}$, which represents an optimal training set according to the definition provided in Section 4.2 and satisfies the target tolerance $\mathcal{L}_T = \epsilon = 0.01$. Hence, this dataset is designated as $\mathcal{D}_{opt}$. The initial training set $\mathcal{D}_1$, which fails to satisfy the specified tolerance, is randomly selected from the surface and exhibits an error of $\mathcal{L}_T = 0.08$. This randomly sampled training set is labeled as $\mathcal{D}_S$. Upon evaluation on the test set $\mathcal{D}_T = \{\dot{\mathbf{x}}_\mathbf{T}, \mathbf{x}_\mathbf{T}, \mathbf{f}_{\mathbf{R},\mathbf{T}}\}$, these datasets generate outputs $\hat{\mathbf{f}}_{\mathbf{R},\mathbf{S}}$ and $\hat{\mathbf{f}}_{\mathbf{R},\mathbf{opt}}$, respectively, which are compared to the ground truth $\mathbf{f}_{\mathbf{R},\mathbf{T}}$ using NRMSE to obtain $\mathcal{L}_T$. The influence of tolerance error is illustrated in Fig. 5.8, which presents the time histories of the restoring forces $\hat{\mathbf{f}}_{\mathbf{R},\mathbf{S}}$ and $\hat{\mathbf{f}}_{\mathbf{R},\mathbf{opt}}$. This analysis demonstrates that two training sets, despite containing equivalent data quantities, carry varying amounts of information for model training purposes. Moreover, Fig. 5.8c reveals that the selected error tolerance ϵ results in $\mathcal{D}_{opt}$ that enables accurate replication of both linear and nonlinear system behavior.

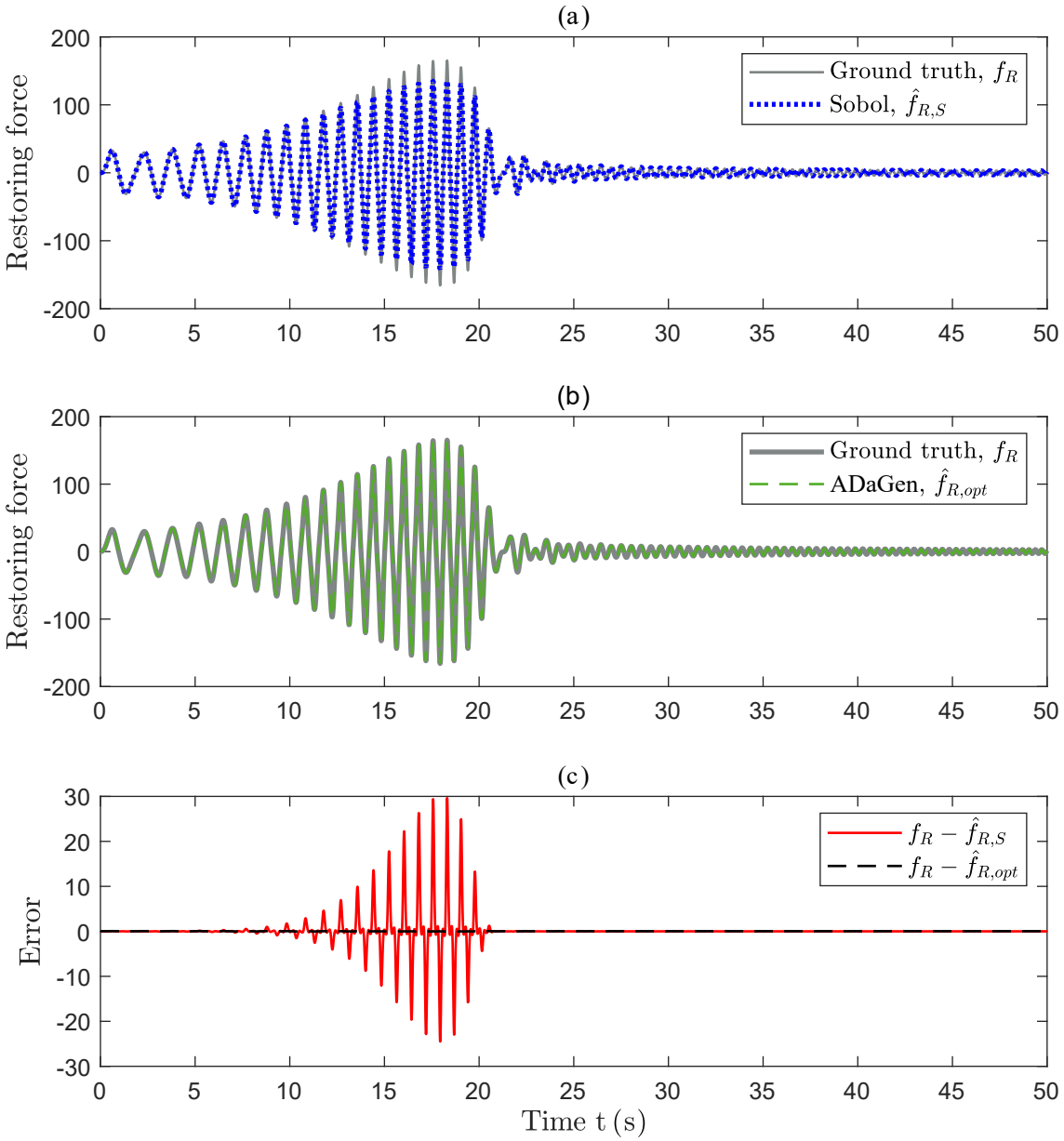

Figure 5.8: (a) Time history of the restoring force $\hat{f}_{R,S}$ produced by the model trained with randomly sampled data set $\mathcal{D}_S$ compared to the ground truth. (b) Time history of the restoring force $\hat{f}_{R,opt}$ produced by the model trained with an optimum data set $\mathcal{D}_{opt}$ compared to the ground truth. (c) Errors showing that $\mathcal{D}_S$ and $\mathcal{D}_{opt}$ lead to different quality of the model, although they have the same size.

5.1.4 Investigated Unsupervised Sampling Methods

Three different unsupervised sampling methods are taken as a reference for comparison: *uniform random sampling*, *Latin hypercube sampling* (LHS) and *Sobol sequence sampling*. In the parameter space, a maximum of 50 points are sampled with each method as depicted in Fig. 5.10. In each experimental campaign, 10, 20, 30 and 50 points are assumed to be sampled, and each campaign is repeated 1000 times to obtain statistically viable results,

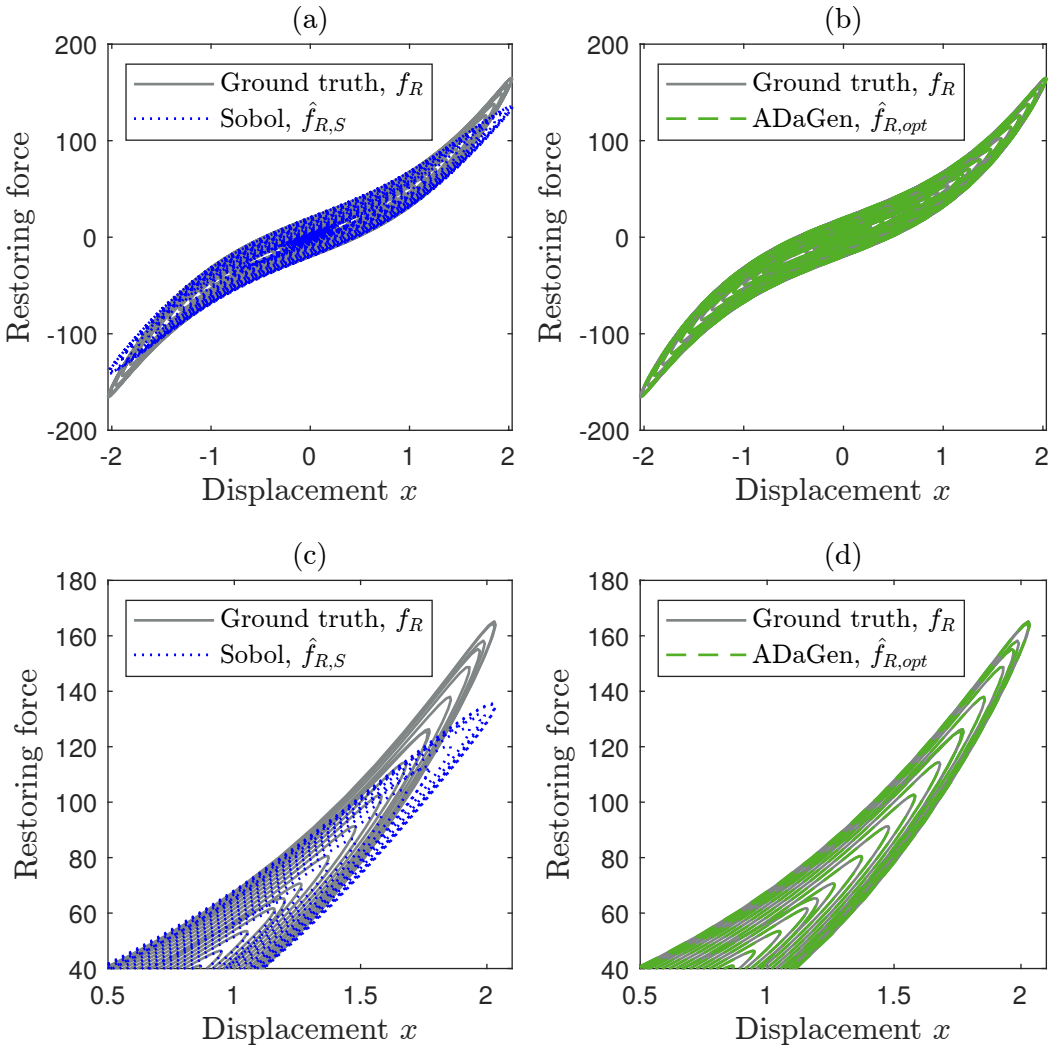

Figure 5.9: (a) Restoring force vs. displacement plot produced by the model trained with a data set $\mathcal{D}_S$ obtained with Sobol sequence, compared to the ground truth. (b) Restoring force vs. displacement plot produced by the model trained with an optimum data set $\mathcal{D}_{opt}$ compared to the ground truth. (c) Enhanced view of the plot shown in Fig. 5.9a. (d) enhanced view of the plot shown in Fig. 5.9b.

since the distribution of points differs in each instance of these methods. This corresponds to a total of 4000 experiments for each sampling method.

For each number of points, the frequency at which the optimum data set $\mathcal{D}_{opt}$ is found is recorded. For example, for the *uniform random* method with 10 points, the simulation is executed 1000 times and the occurrence of finding the optimum data set $\mathcal{D}_{opt}$ is recorded each time. This process is then repeated with 20, 30, and 50 sampling points with the expectation that the results would improve as the number of samples is increased. The same procedure is applied for other sampling methods.

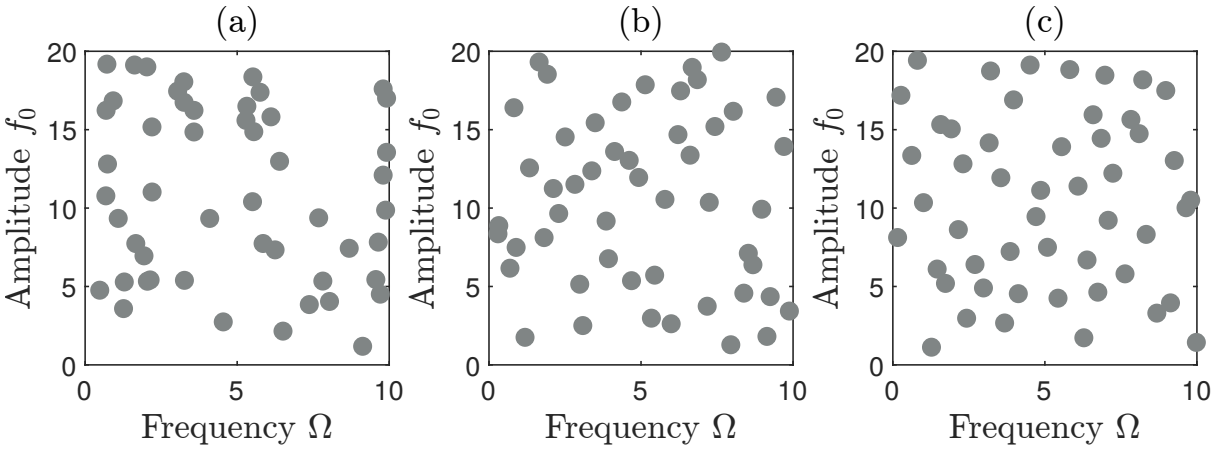

Figure 5.10: Parameters sampled by (a) *Uniform random sampling*; (b) *Latin hypercube sampling*; and (c) *Sobol sequence sampling.*

5.1.5 Adaptive Data Generation Configuration

The PS optimization algorithm is implemented using the function *patternsearch* in Matlab. The algorithm's objective function takes the excitation signal amplitude and frequency as inputs and provides NRMSE (adaptation cost function) as the output. When the objective function receives the excitation signal parameters, it generates the sine signal with these parameters, executes the numerical simulation of the response and performs the training of the model selection pool. The model pool is then evaluated and the NRMSE of the pool member with the best performance is returned as the output together with the NRMSE value. The first sample is selected randomly, and every subsequent sample is determined by the PS algorithm. Similar to unsupervised methods, the proposed framework is executed 1000 times with different starting points in the sampling space. The detailed simulation parameters are presented in Table 5.1. It is observed that normalizing the sampling space for PS and selecting a large starting mesh results in performance improvement. This is confirmed by the results showing that the modified PS finds the optimum training data set on average after 19 experiments, while the default algorithm finds it on average with 23 experiments.

Table 5.1: Pattern search simulation parameters used in the study.

Parameters	**Values**
Initial mesh size	8
Poll method	Minimal basis set
Mesh tolerance	0.01
Scale mesh	No
Use complete poll	No
Function tolerance	0.01
Use cache	Yes

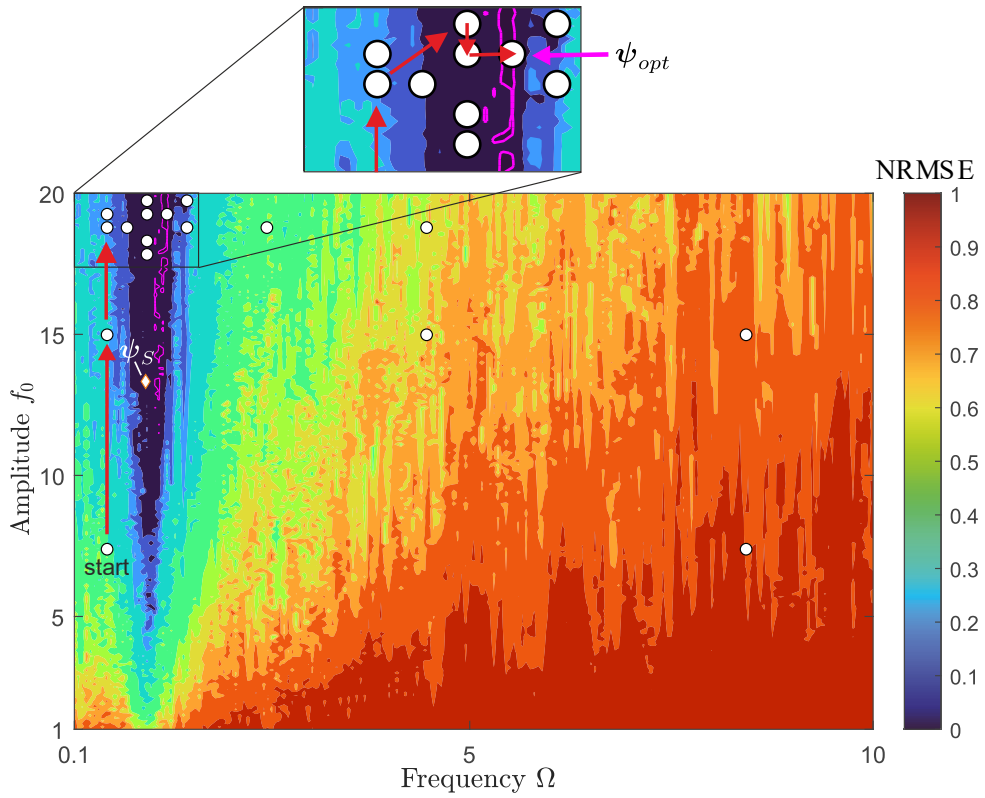

Figure 5.11: One example of convergence of the proposed framework to the optimum excitation parameters $\boldsymbol{\psi}_{opt}$ encircled by the magenta line which produce the optimum training data set $\mathcal{D}_{opt}$ are found after 17 experiments.

An example of the converged solution is shown in Fig. 5.11, where the PS algorithm requires 17 experiments to identify the optimal data set. For fair comparison purposes, a solution obtained through the Sobol sequence utilizing the same experiment count was plotted and denoted as $\boldsymbol{\psi}_S$. These two examples are employed in subsequent sections for additional illustrative examples. From the convergence behavior of the proposed framework displayed in Fig. 5.11, it is observed that the algorithm starts with a random point positioned far from the global minimum, and during the approach toward the global minimum, sampling becomes denser in the global minimum region. This result demonstrates that the dynamic data collection for model training is transformed into an optimization problem within a static sampling space. It should be noted that this transformation is only feasible due to the assumption that an optimal training data set $\mathcal{D}_{opt}$ exists which can be generated using sine excitation with parameters $\boldsymbol{\psi}$. Should this assumption not hold true, the optimal training data set $\mathcal{D}_{opt}$ would require construction at each iteration. In such a case, the error surface presented in Fig. 5.7 would evolve with each data batch added to the training set $\mathcal{D}$. This remains theoretically feasible with the proposed framework, where the PS algorithm would be applied in the context of dynamic function optimization and is explored in Chapter 7. However, at this stage, the proposed framework is presented in its simplest form for the sake of brevity.

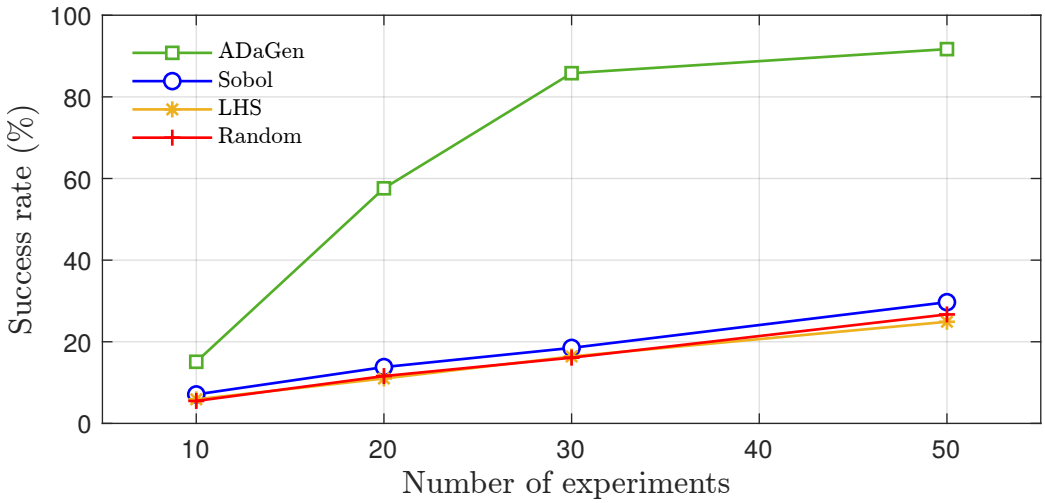

Figure 5.12: Plot showing the number of experiments required to achieve a certain success rate for each method. The proposed framework clearly outperforms the unsupervised methods since its success rate is more than twice higher for 10 experiments and continues to increase with the number of experiments.

5.1.6 Comparison with Unsupervised Sampling Methods

For comparison between the proposed adaptive sampling and unsupervised sampling methods (Uniform random sampling, Latin hypercube sampling (LHS), Sobol sequence sampling) the number of times the optimum training data set is found in 1000 simulations with a prescribed number of experiments ($10-50$) is recorded. Fig. 5.12 shows the results, where after 10 experiments, the proposed method finds the optimum training data set 151 times, which is equivalent to a 15% success rate. At least 23 experiments are required by the best unsupervised method, Sobol sequence, to achieve the same training success. As the number of samples increases, the performance difference grows and becomes pronounced at 50 experiments, where almost 100% success rate is achieved by the proposed framework, while at most 40% success can be reached by Sobol sequence. Random sampling and LHS show similar performance, whereas Sobol sampling demonstrates slightly better performance among unsupervised methods. The difference between LHS and random sampling is expected to increase in favor of LHS as the sampling space dimensions increase. This occurs because LHS sampling possesses equal space-filling properties in all dimensions. However, these results confirm that for functions where the evaluations are extremely expensive, such as for experiments, the adaptive methods vastly outperform the batch sampling methods. This is confirmed by the proposed ADaGen performance.

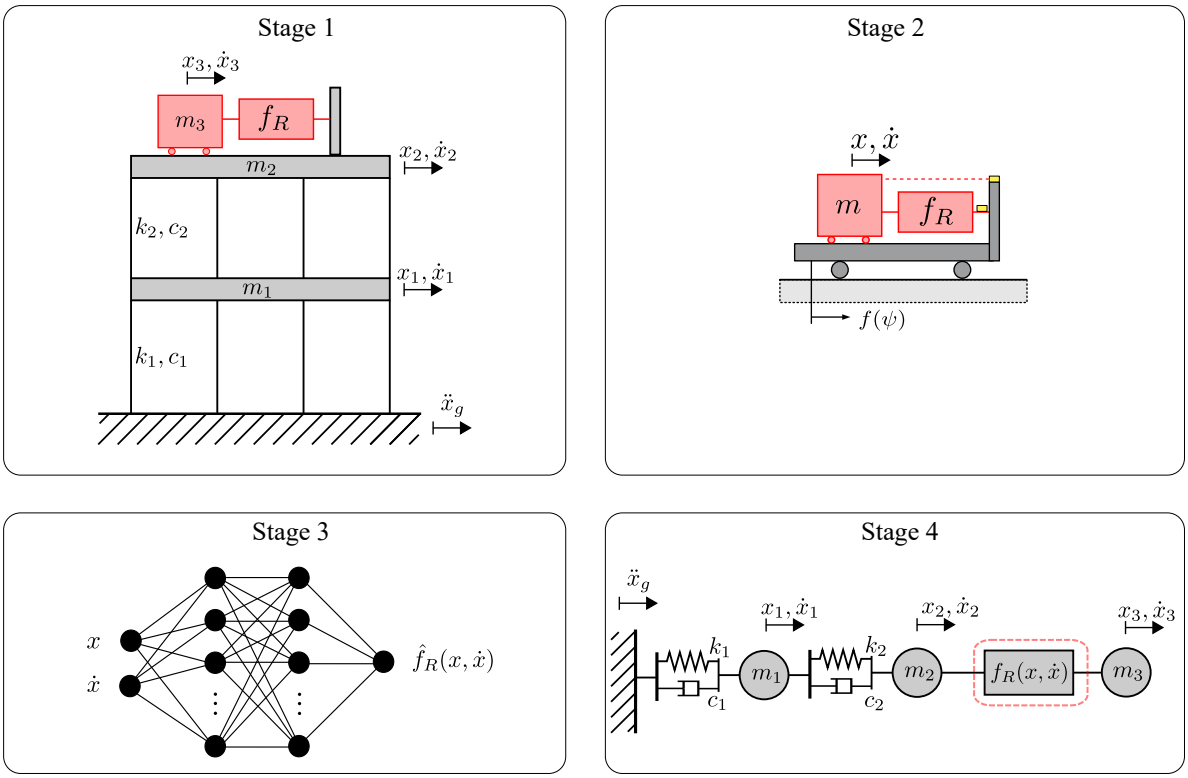

Figure 5.13: Two-story shear frame with a nonlinear TMD modeled by the proposed framework.

5.2 Two-story Shear Frame with a Nonlinear Tuned Mass Damper

To examine how the modeled restoring force can be incorporated into structural analysis, a two-story shear frame equipped with a nonlinear TMD on the top floor is examined, cf. Fig. 5.13. The system experiences ground acceleration $\ddot{x}_g$ and the corresponding equations of motion are formulated as

$$-m_1\ddot{x}_g(t) = m_1\ddot{x}_1 + k_1x_1 + k_2(x_1 - x_2) + c_1\dot{x}_1 + c_2(\dot{x}_1 - \dot{x}_2), \tag{5.2a}$$

$$-m_2\ddot{x}_g(t) = m_2\ddot{x}_2 - k_2(x_1 - x_2) - c_2(\dot{x}_1 - \dot{x}_2) - f_R, \tag{5.2b}$$

$$-m_3\ddot{x}_g(t) = m_3\ddot{x}_3 + f_R, \tag{5.2c}$$

where f_R represents the restoring force of the TMD including both linear and nonlinear components and is given by:

$$f_R = \beta(x_3 - x_2) + \gamma(x_3 - x_2)^3 + \alpha(\dot{x}_3 - \dot{x}_2). \tag{5.3}$$

The nonlinear TMD parameters correspond to those of the Duffing oscillator having the restoring force described in Eq. (5.1). The parameter values for both the structure and TMD are listed in Table 5.2. The structure has the first and second natural frequencies

of 0.40 Hz and 1.05 Hz respectively. The TMD is designed to control the second natural frequency in this case study. The structure undergoes excitation from the North-South component of the *El Centro Earthquake* recorded at the Imperial Valley Irrigation District substation in El Centro, California, on May 18, 1940. The seismic record is amplified by factors of $\phi = 10$, 20, 50 and 300 to excite the system and to evaluate the performance limits of the proposed framework compared to competitive methods.

Table 5.2: System parameters used in the simulation.

Parameter	**Value**	**Explanation**
Structure:		
m_1,m_2	15	Mass of rigid floors
c_1,c_2	1.25	Damping coefficient of floor columns
k_1,k_2	250	Stiffness coefficient of floor columns
TMD:		
m_3	1	Mass of the TMD
α	1.25	Damping coefficient
β	$(2\pi)^2$	Stiffness coefficient
γ	10	Nonlinear stiffness coefficient

As depicted in Fig. 5.13, the numerical example involves an engineering application executed in four stages. During the *first stage*, the structural response is calculated as the reference solution using Eqs. (5.2 - 5.3) and the parameters from Table 5.2, cf. Fig. 5.14. The *ode45* solver is utilized for this computation. For the sake of brevity, only the displacement response of the second floor is presented. During the *second stage*, the TMD restoring force is modeled using results from Section 5.1.5 through the proposed data generation framework and the Sobol sequence for data sampling as shown in Fig. 5.11. In the *third stage*, the developed TMD model is integrated into the numerical simulation of the structure, replacing Eq. (5.3). Subsequently, in the *fourth stage*, the structural response to the earthquake excitation is simulated and compared against the ground truth obtained in first stage.

Fig. 5.14 shows the displacement time histories of the second floor. Fig. 5.15 illustrates the TMD displacements, where the deterioration in model performance becomes more apparent. It is evident that the FNN model acquired through the proposed data generation framework can adequately model the TMD restoring force for scaling factors $\phi = 10$, 20 and 50. Therefore, no divergence exists between the modeled response and the reference solution. The initial decline in model performance occurs at $\phi = 300$, which significantly exceeds its design range corresponding to $\phi = 20$. Conversely, the FNN model trained using the Sobol sequence training set performs poorly even at $\phi = 20$, with performance degrading rapidly for $\phi = 50$ and $\phi = 300$, as demonstrated in Figs. 5.14 and 5.15.

Fig. 5.16 illustrates the relationship between restoring force and TMD displacement. The generalization performance of the FNN model developed using the proposed framework

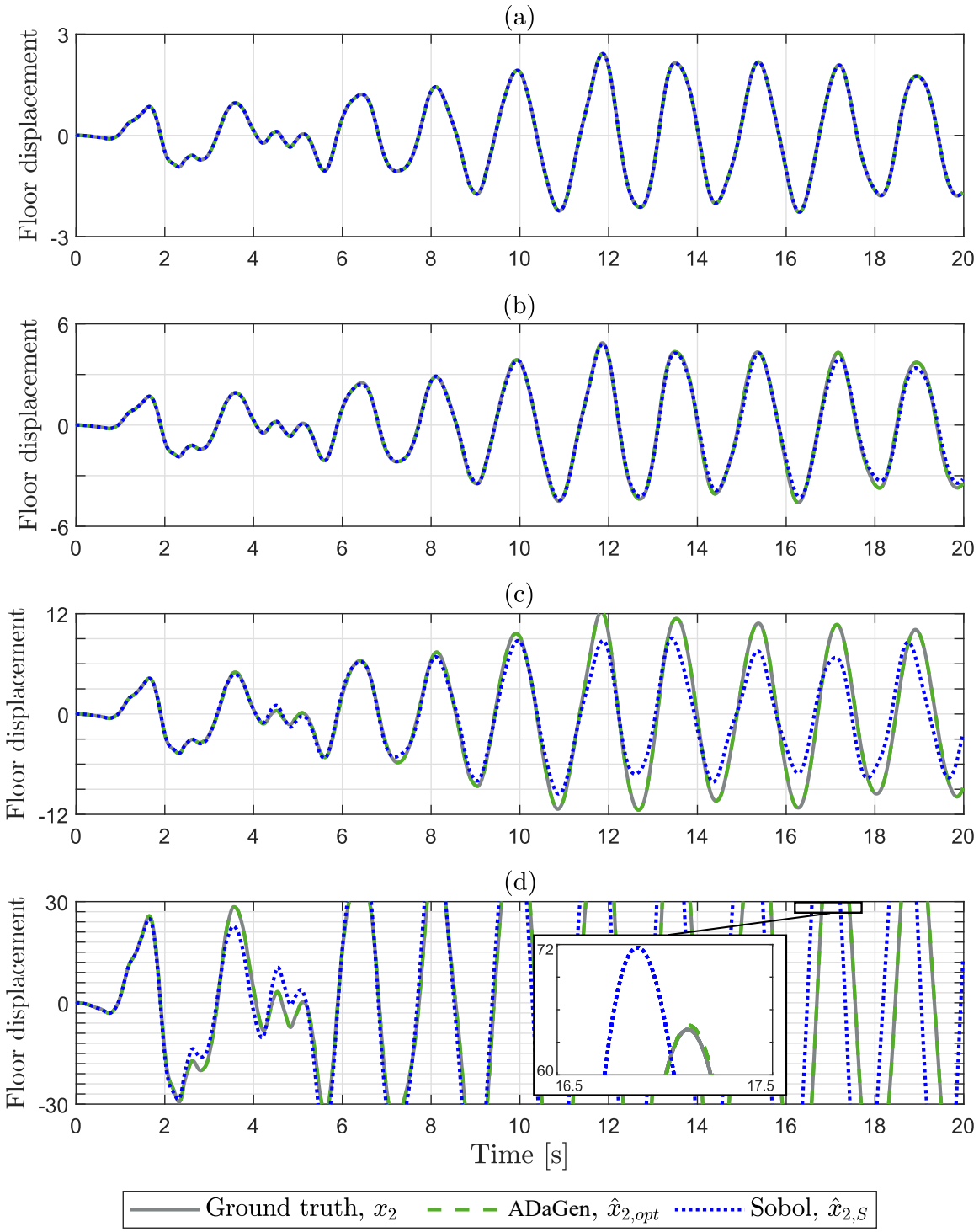

Figure 5.14: Plots showing the displacement responses of the 2nd floor using an FNN trained with the data sets obtained with proposed framework ($x_{2,opt}$) and Sobol sequence ($x_{2,S}$) when the excitation signal is scaled by $\phi = 10$ (a), 20 (b), 50 (c) and 300 (d) times.

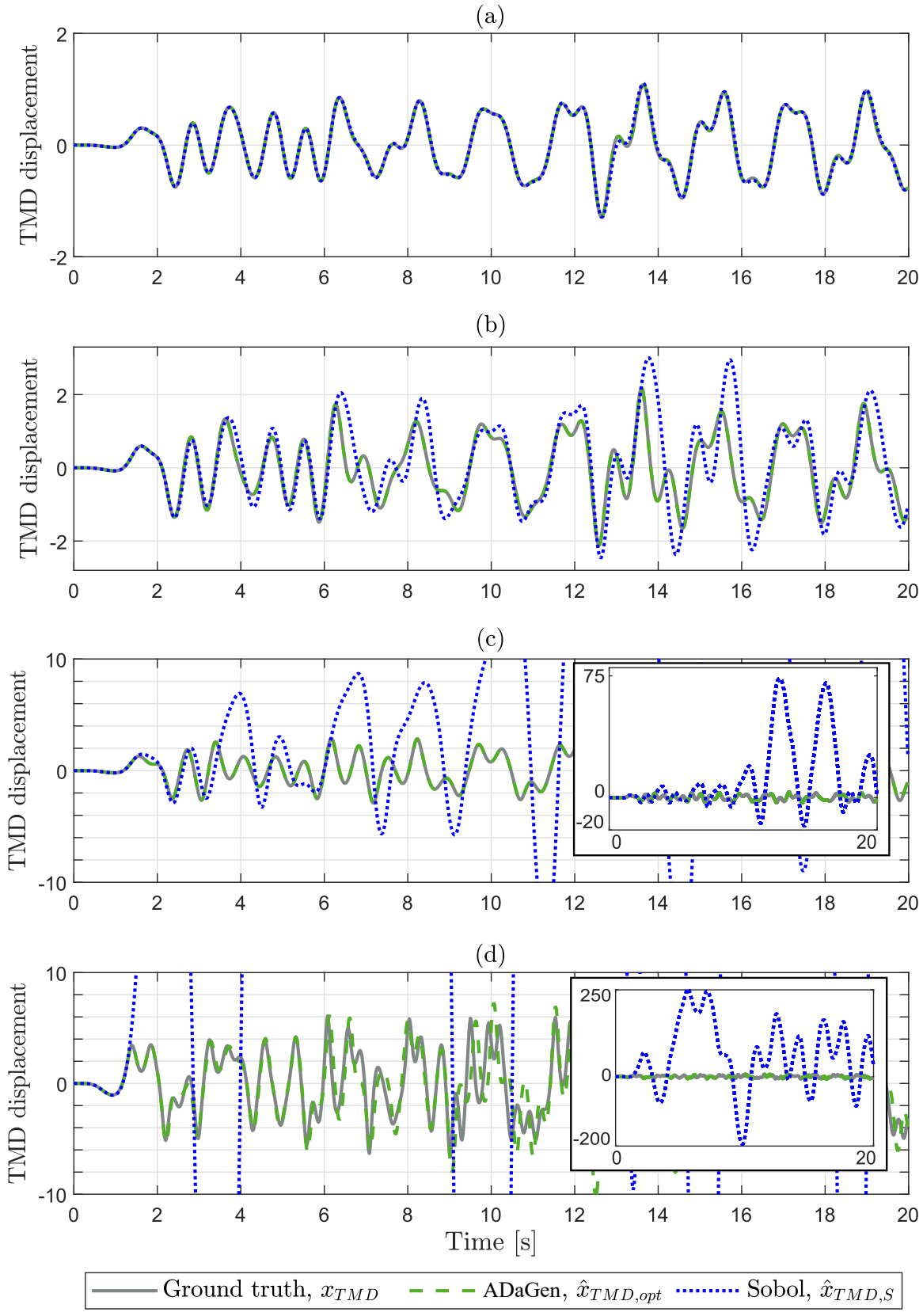

Figure 5.15: Plots showing the displacement responses of the TMD using an FNN trained with the data sets obtained with proposed framework ($x_{TMD,opt}$) and Sobol sequence ($x_{TMD,S}$) when the excitation signal is scaled by $\phi = 10$ (a), 20 (b), 50 (c) and 300 (d) times.

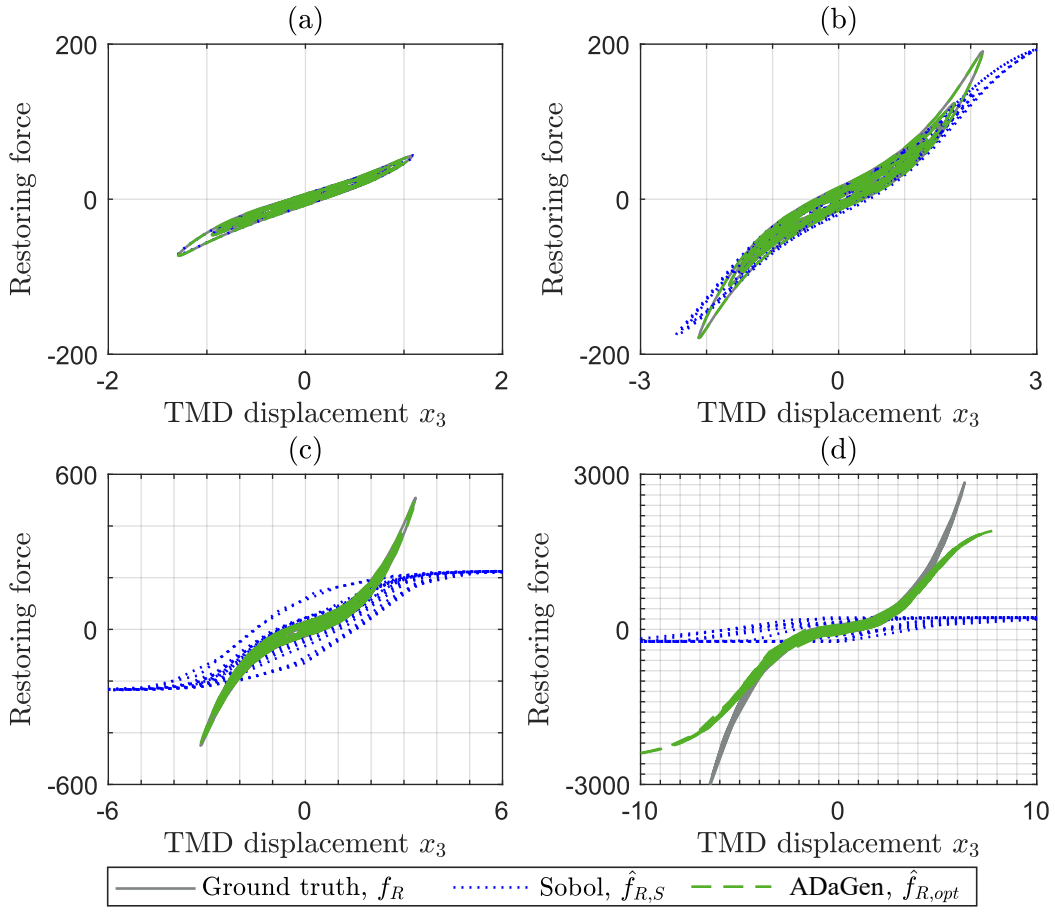

Figure 5.16: Restoring force vs. displacement plots of the TMD using an FNN trained with the data sets obtained with proposed framework ($\hat{f}_{R,opt}$) and Sobol sequence ($\hat{f}_{R,S}$) when the excitation signal is scaled by $\phi = 10$ (a), 20 (b), 50 (c) and 300 (d) times.

surpasses that of the model trained with the Sobol sequence. When the Sobol sequence model's restoring force is applied in the simulation, the model fails to capture the nonlinear behavior of the TMD where hardening effects become pronounced. This results in substantial overestimation of displacements and underestimation of restoring forces. Consequently, these findings validate the benefits of utilizing model performance information to sequentially sample new data points through optimization methods.

5.3 Summary

This chapter validates the proposed ADaGen framework using the Duffing oscillator as a numerical benchmark. The framework learns force–state mappings through FNNs trained with data from simulated shaking table experiments. To address training variance, which ranged from NRMSE of 0.0007 to 0.8437 under identical data but different initializations, a model selection pool of 10 networks is employed. The optimization problem is formulated in a two-dimensional sampling space of sine excitation amplitude and frequency, where the adaptation cost function measures model performance on a fixed swept-sine test set. Visualization of this cost function reveals a clear global minimum, confirming that equal-sized datasets can carry vastly different amounts of information for training.

Using pattern search optimization to adaptively select excitation parameters, the framework identifies optimal training datasets in an average of 10 experiments across 1000 simulations. After only 10 experiments, ADaGen achieves 15% success rate while the best unsupervised baseline (Sobol sequence) requires 23 experiments to match this performance. At 50 experiments, ADaGen reaches nearly 100% success compared to 40% for Sobol sequence. Uniform random and Latin hypercube sampling perform similarly but fall behind Sobol, confirming the expectations from Section 3.2.

To assess practical applicability, the learned restoring force model is integrated into a two-story shear frame with a nonlinear TMD under earthquake excitation. The ADaGen-trained model accurately reproduces structural response even beyond the design range, while the Sobol-trained model degrades rapidly at design excitation levels. These results demonstrate that adaptive sampling transforms expensive experimental data collection into a tractable optimization problem, producing compact datasets that yield accurate and generalizable models for simulation.

Chapter 6

Experimental Validation with TLCD Prototype

This chapter is primarily based on the insights and results of the publication by Milicevic and Altay [93] and validates the ADaGen framework experimentally using a TLCD prototype. Unlike the numerical validation in Chapter 5, this study employs Neural ODEs instead of FNNs to model system dynamics from displacement measurements alone. This extension significantly broadens the framework's applicability to devices where direct force measurement is impractical or where restoring forces cannot be meaningfully separated from inertial effects. The chapter introduces three key advances beyond the baseline framework presented in Chapter 4:

1. *Neural ODE integration*: Extension from force-state FNN mapping to continuous-time dynamics modeling, enabling identification from system states alone without requiring force transducers.
2. *Unknown modal mass identification*: Modification of the Neural ODE architecture to simultaneously learn system dynamics and modal mass parameter. This extension addresses a critical limitation for devices with auxiliary masses where effective mass cannot be easily measured or estimated.
3. *Experimental validation*: Demonstration of the complete adaptive experimental workflow on a physical system, including real measurement noise, actuator constraints, and operational limitations inherent in laboratory testing.

The chapter is organized as follows. Section 6.1 describes the TLCD prototype and experimental infrastructure. Section 6.2 defines the sampling space and experimental parameters. Section 6.4.1 presents the Neural ODE configuration including the architectural modifications. Section 6.5 presents experimental campaign results, visualizing the adaptation cost function landscape and comparing adaptive sampling against conventional batch methods. Section 6.6 demonstrates integration of the identified model into coupled structural simulation for performance evaluation.

6.1 Experimental Setup

The ADaGen framework is applied to a TLCD prototype. A TLCD is defined as a U-shaped vessel partially filled with Newtonian liquid that dissipates structural vibrations through fluid motion and orifice head loss. TLCDs find widespread application in civil engineering structures, including skyscrapers and wind turbines. They can be deployed independently or integrated with complementary control systems. While the prototype is an omnidirectional TLCD (O-TLCD) with semi-active control capabilities [90, 91, 152], only passive operation with all cells open is considered here. This configuration choice ensures that validation focuses on the adaptive experimental design methodology rather than on particular control strategies. For brevity, the proposed framework is demonstrated using a single configuration, and the TLCD is excited uniaxially.

Figure 6.1 shows the experimental configuration. The TLCD is mounted on a uniaxial shaking table ① with four L-arms, each divided into five cells ②. The middle cell is permanently open, while the four control cells can be opened or closed by regulators ④ to modify damping characteristics, providing the damper with semi-active vibration control capabilities as described in [91]. Ultrasonic sensors ③ measure vertical liquid displacement at 100 Hz sampling rate. The idle liquid height can be adjusted via an inlet/outlet pipe ⑤. For this experimental campaign, the idle liquid height is $L_V = 125$ mm and the horizontal distance between the column axes is $L_H = 520$ mm. The cells c_1 and c_2 are 70 mm wide, while cell c_3 is 50 mm wide. All cells have a length of $d_3 = 50$ mm.

For physical context, the fluid motion in a TLCD is described by a streamline, with the resulting acceleration balanced by pressures and flow losses, which are lumped together considering pipe friction, directional changes, and turbulence. The TLCD equation of motion derived from the unsteady Bernoulli equation is:

$$\ddot{u} + \delta|\dot{u}|\dot{u} + \omega_D^2 u = \gamma_1 \ddot{x}, \tag{6.1}$$

where u is the liquid displacement, δ is the head loss coefficient, ω_D is the natural circular frequency, γ_1 is a geometric factor coupling table acceleration $\ddot{x}$ to liquid motion, and $\ddot{x}$ is the shaking table acceleration. The damper is characterized by its natural circular frequency ω_D. Another critical parameter is the nonlinear damping term $\delta|\dot{u}|\dot{u}$, where the head loss coefficient δ accounts for flow losses due to pipe friction and turbulence. Although analytical formulations have been developed, they exhibit limited precision when applied to complex TLCD geometries, as demonstrated in [91].

Estimating the geometric factor γ_1 is equally challenging. During the derivation of the equation of motion, this factor is introduced to consolidate the damper's geometric parameters as:

$$\gamma_1 = \frac{L_H}{2L_V + \frac{A_V}{A_H} L_H}, \tag{6.2}$$

where A_V/A_H represents the ratio of the cross-sectional areas of the vertical and horizontal parts of the damper. Given that L_H and L_V represent approximations of the actual

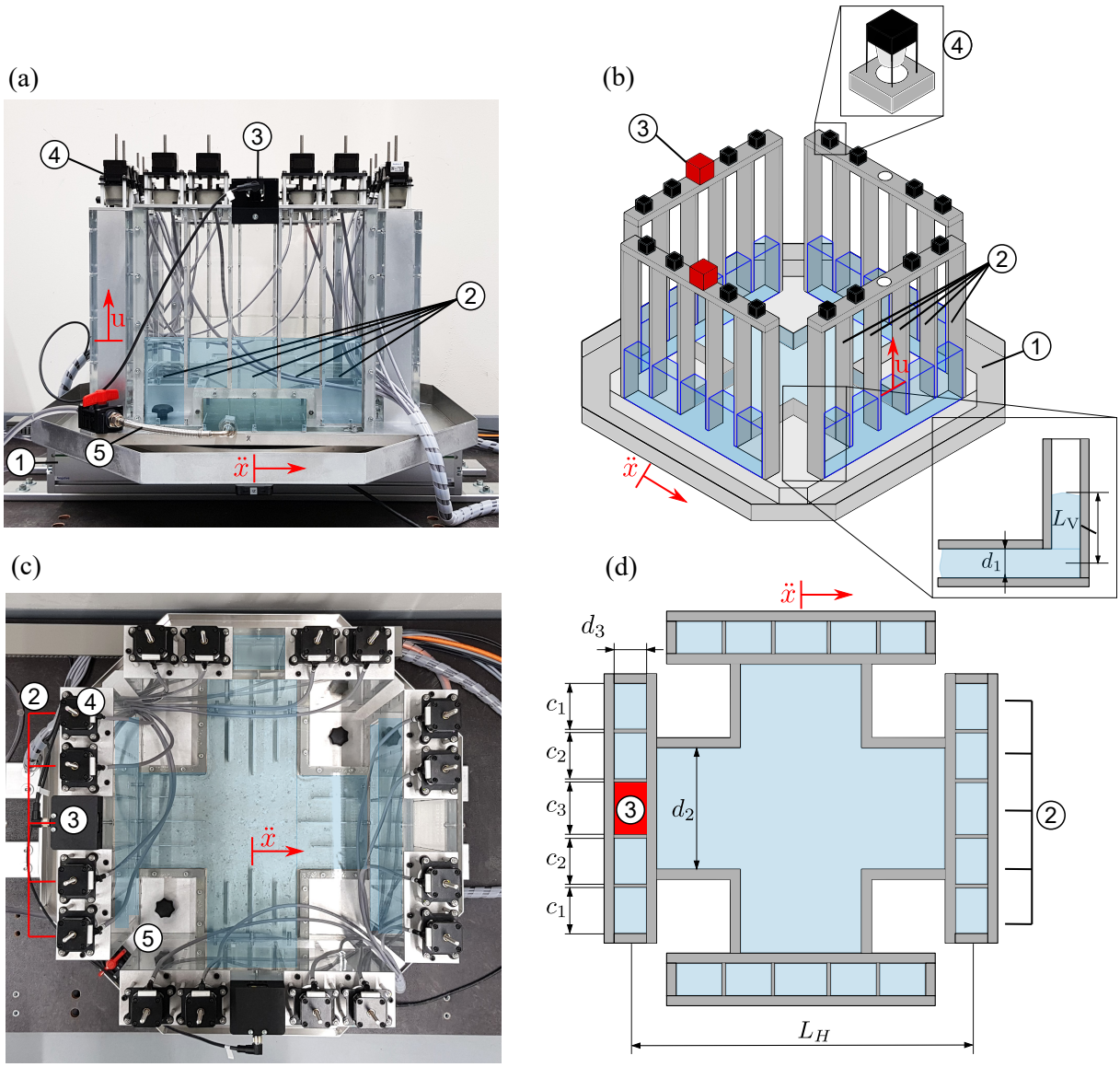

Figure 6.1: Experimental setup of O-TLCD with four L-arms connected to a uniaxial shaking table ① shown in front view (a) and plan view (c), with corresponding sketches (b) and (d). Each L-arm is divided into five cells ② equipped with adjustable stoppers ④. Liquid motion is measured by ultrasonic sensors ③ and idle liquid height can be adjusted via inlet/outlet pipe ⑤.

fluid flow path, their determination becomes especially difficult in complex TLCD configurations, directly affecting the precision of γ_1. Nevertheless, precise determination of this factor is critical because it represents the damper's modal mass, which is a parameter fundamental to the structural design process when incorporating TLCDs. The extended Neural ODE formulation can directly estimate the geometric factor $\hat{\gamma}_1$ to circumvent this challenge, as detailed in Section 6.4. It is emphasized that this equation of motion is not used within the ADaGen framework and is assumed to be unknown. It is provided solely for physical interpretation and to illustrate the complexity of the system.

6.2 Sampling Space and Experimental Parameters

Following the ADaGen methodology, the experimental parameters that define the sampling space must be specified. For this study, the excitation signal applied to the shaking table is parameterized while keeping other experimental conditions constant. Sine signals are employed, characterized by displacement amplitude x_0 and cyclic excitation frequency f_S, forming a two-dimensional sampling space $\boldsymbol{\psi} = [x_0 \quad f_S]^\top$.

While sine signals are less information-rich than more powerful excitations such as APRBS for system identification purposes, they are deliberately selected here for two reasons: (i) the two-parameter space enables visualization of the adaptation cost function, facilitating interpretation of optimization behavior; and (ii) their deterministic nature ensures fair comparison with batch sampling methods by eliminating signal-induced randomness. Furthermore, employing signals which contain randomness can damage the prototype or the experimental setup due to their unpredictability. By retaining the transient response in the signal, it is assumed that it is sufficiently powerful to extract the necessary information. The goal is not to design the optimum excitation signal but rather to sample the most informative response data. Using a simple harmonic function allows for two-dimensional parameterization, enabling later visualization of the adaptation cost function $\mathcal{L}_T(\boldsymbol{\psi})$, which would not be feasible with higher-dimensional parameterization.

The test dataset is generated using a sine sweep excitation with constant acceleration amplitude of 0.4 ms^{-2} and frequency range of $[0.40 \ldots 1.00]$ Hz, shown in Fig. 6.2. This swept-sine signal effectively extracts comprehensive information about the system, serving as a reliable measure of model accuracy and generalization capability. It provides broader frequency coverage than individual sine signals and therefore serves as a benchmark for model generalization. The test set remains fixed throughout all experiments, providing a consistent metric for comparing dataset quality.

Figure 6.3 illustrates the data acquisition workflow. The sampling parameters $\boldsymbol{\psi} = [x_0 \quad f_S]^\top$ define a sine excitation $x = x_0 \sin(2\pi f_S t)$ applied by the shaking table. Due to displacement constraints of the shaking table, amplitude is specified in terms of displacement x_0, though acceleration $\ddot{x}$ is used in the training dataset. For training the Neural ODEs, the excitation signal is translated into an acceleration $\ddot{x}$ time history. The TLCD response is captured by the ultrasonic sensor measuring vertical liquid displacement u at 100 Hz. To obtain the complete training dataset $\mathcal{D} = \{\ddot{\mathbf{x}}, \mathbf{u}, \dot{\mathbf{u}}\}$, the raw measurements are processed as follows. Offsets and high-frequency noise are removed from the measured liquid displacement u. Displacement data $\mathbf{u}$ is numerically differentiated to obtain velocity $\dot{\mathbf{u}}$. Finally, signals are trimmed to 20 s duration. Note that displacement $\mathbf{u}$ and velocity $\dot{\mathbf{u}}$ serve as direct inputs to the internal neural network, while excitation $\ddot{\mathbf{x}}$ is added to the network output before integration, as indicated in Eq. 6.5 and distinguished by color in Figs. 6.3 and 6.2.

The sampling space of the experimental parameters is defined based on the damper's operational range, which is determined by the maximum allowable liquid displacement.

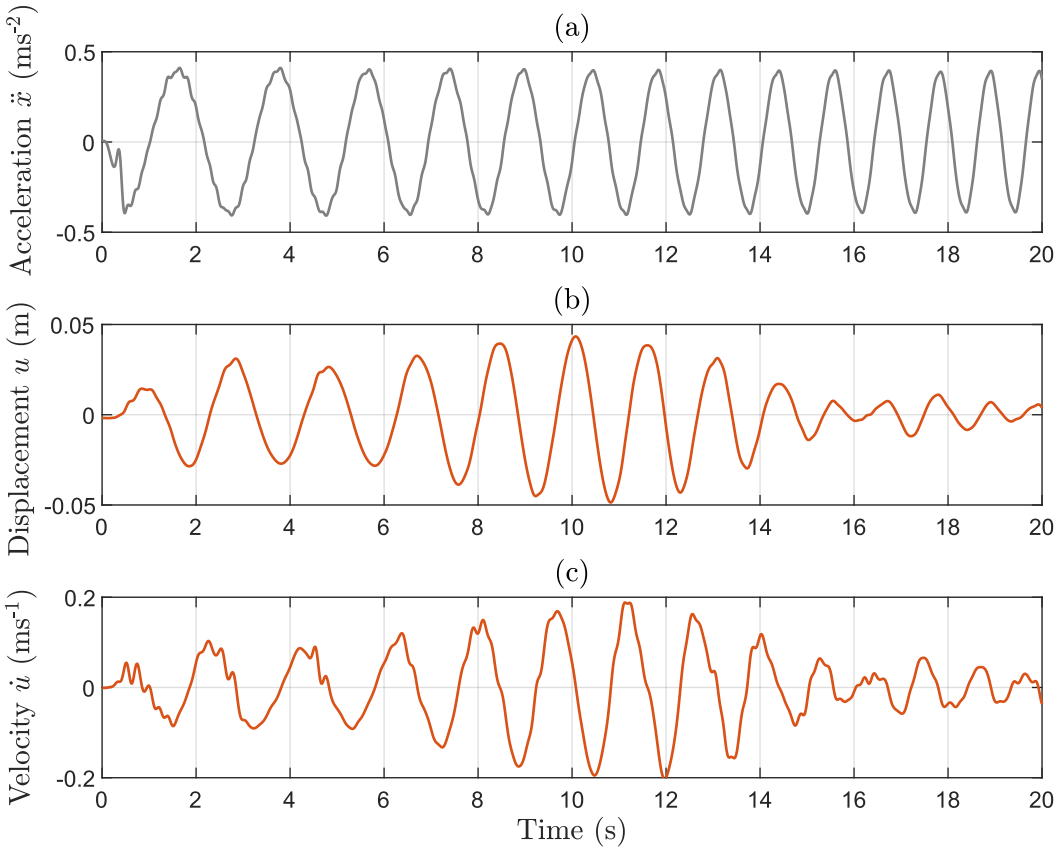

Figure 6.2: Test dataset obtained using sine sweep excitation.

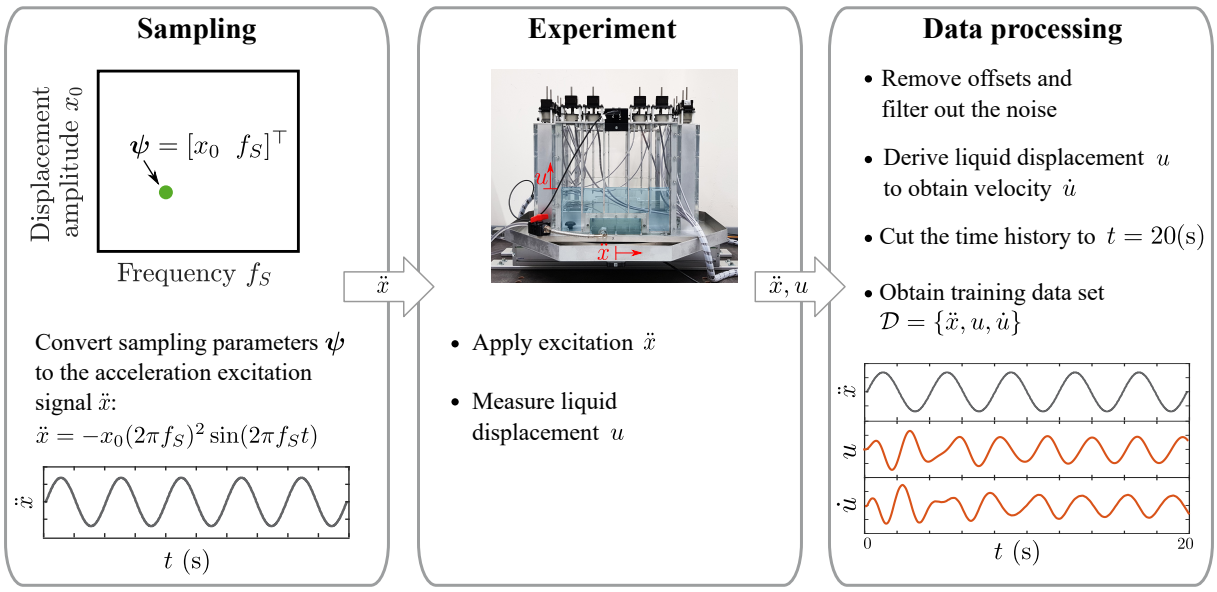

Figure 6.3: Workflow showing training dataset acquisition from a sampling point.

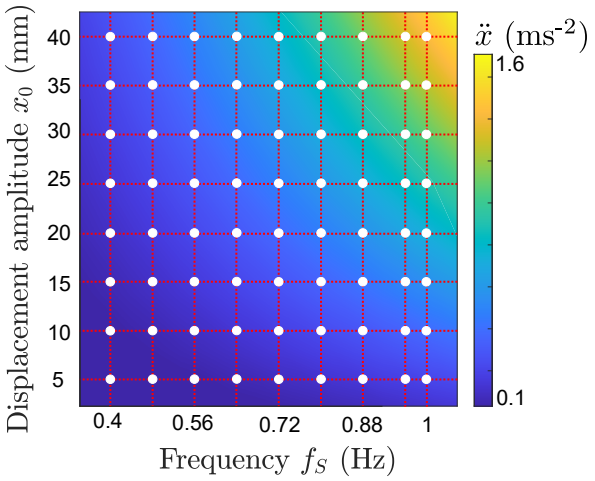

Figure 6.4: Grid showing the collected datasets in the sampling space and the relationship between excitation frequency f_S, acceleration amplitude $\ddot{x}_0$, and displacement amplitude x_0.

The specimen's operational range spans frequencies from 0.4 Hz to 1.0 Hz and displacement amplitudes from 5 mm to 40 mm. To gain a comprehensive understanding of the results, the experimental parameters are initially sampled uniformly before applying the active learning framework. The displacement amplitudes are sampled in the range of 5–40 mm with increments of 5 mm, and excitation frequencies are sampled in the range of 0.40–0.96 Hz with increments of 0.08 Hz, and additionally at 1.00 Hz. These sampling points are represented as dots in Fig. 6.4. With active learning, the optimization algorithm is restricted to selecting only from these points. If a sample is requested that does not correspond to these points, the point closest to the requested one is used. This restriction limits the framework's effectiveness but allows accurate comparison with unsupervised sampling methods by utilizing the same set of available points.

One of the challenges that arises from the experimental setup and the parameterization is that using excitation displacement amplitude as a parameter creates non-uniform forcing across frequencies. Figure 6.4 shows the relationship between displacement amplitude x_0, acceleration amplitude $\ddot{x}_0$, and excitation frequency f_S. The color gradient indicates that acceleration amplitude increases quadratically with frequency ($\ddot{x}_0 = (2\pi f_S)^2 x_0$), meaning higher frequencies apply greater forces to the system.

6.3 Pattern Search Parameters

The generalized pattern search optimization algorithm, as implemented in Matlab's *patternsearch* function, is used as the optimization algorithm. The adaptation cost function $\mathcal{L}_T$ serves as the optimization objective, taking experiment parameters $\boldsymbol{\psi}_i = [x_{0_i} \quad f_{S_i}]^\top$ as input and returning the mean absolute error (MAE) of the best ensemble member evaluated on the test set. As described in Chapter 4, the first sampling point $\boldsymbol{\psi}_1$ is selected randomly, with all subsequent points determined by the PS algorithm.

The parameter space is normalized to [0,10] for both amplitude and frequency to ensure uniform mesh behavior and ensure algorithmic generality across different TLCD configurations. A minimal basis set is used for the poll method. To minimize experiments, an opportunistic (greedy) polling strategy is employed by disabling the *complete poll* option, where the mesh center moves immediately to the first trial point yielding improvement. The *cache option* is enabled to prevent redundant evaluation of previously tested points, allowing the evaluation record to be kept such that the same point is not evaluated twice.

The convergence criteria were carefully selected to balance accuracy requirements with practical experimental constraints. After each successful iteration, the mesh is expanded by a factor of 2, and after an unsuccessful iteration it is contracted by 0.5. The error threshold was set to 0.25 MAE, which approximately translates to 0.1 normalized root mean square error (NRMSE), corresponding to approximately 90% goodness of fit. Normalization is performed using standard deviation. This threshold was determined to be appropriate as it allows for both measurement noise and uncertainty as well as numerical processing and integration errors. The minimum allowed mesh size was set to 0.2, which corresponds to a distance in the measurement as described in Section 6.2. The algorithm was allowed to run until either the minimum mesh size was reached or the minimum error threshold was satisfied, without restrictions on the maximum number of experiments.

6.4 Extended Neural ODE Formulation for Forced Systems

While the standard Neural ODE formulation introduced in Section 2.1.3 is effective for autonomous systems, many structural dynamics problems involve forced vibrations where external excitations drive the system response. The TLCD application requires two critical extensions to the classical Neural ODE framework, namely modeling forced vibrations and identifying unknown modal mass parameters. This section presents these extensions and their theoretical justification.

The classical Neural ODE formulation from Eq. (2.7) models autonomous dynamical systems. However, for forced vibration systems, explicit accommodation of external forcing is required. To understand the necessity of this extension, consider the state-space representation of a forced dynamical system:

$$\dot{\mathbf{h}}(t) = \mathbf{A}\mathbf{h}(t) + \mathbf{B}F(t), \tag{6.3}$$

where $\mathbf{A}$ is the state matrix containing system parameters, $\mathbf{B}$ is the input matrix, and $F(t)$ is the external forcing function. For structural systems, both $\mathbf{A}$ and $\mathbf{B}$ contain mass-dependent terms. For a linear SDOF system with mass m, stiffness k, and damping

coefficient c, these become:

$$\mathbf{A} = \begin{bmatrix} 0 & 1 \\ -\frac{k}{m} & -\frac{c}{m} \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} 0 \\ \frac{1}{m} \end{bmatrix}. \tag{6.4}$$

This formulation assumes all system parameters can be learned implicitly through the network weights in the standard Neural ODE framework. However, when the mass parameter is unknown and cannot be easily measured or estimated, which is often the case, the standard Neural ODE faces an identifiability issue. The network cannot uniquely determine both the system dynamics (matrix $\mathbf{A}$) and the input scaling (matrix $\mathbf{B}$) simultaneously, as these are coupled through the unknown mass parameter.

To address this limitation, the Neural ODE formulation is extended by explicitly parameterizing the input scaling as a learnable parameter:

$$\dot{\mathbf{h}} = f_\theta(\mathbf{h},t,\boldsymbol{\theta}) + \theta_m \mathbf{B} F(t), \tag{6.5}$$

where $F(t)$ is the measured external force excitation, $\mathbf{B}$ is the input matrix structure, and θ_m is a scalar parameter accounting for the unknown modal mass. For the SDOF TLCD system, $\theta_m = -m^{-1}$ is used corresponding to the inverse of the modal mass, and $\mathbf{B} = [0\ 1]^T$. The state vector for the TLCD is $\mathbf{h} = [\mathbf{u}(t)\ \dot{\mathbf{u}}(t)]^\top$, where u denotes the vertical liquid displacement and $\dot{u}$ the corresponding velocity.

The internal FNN parameters $\boldsymbol{\theta}$ consist of weights $\mathbf{W}$ and biases $\mathbf{b}$, structured in layers as described in Section 2.1.3:

$$\mathbf{h}^l = \sigma(\mathbf{W}^l \mathbf{h}^{l-1} + \mathbf{b}^l), \tag{6.6}$$

where $\sigma(\cdot)$ is the activation function of the lth layer. Importantly, θ_m is not embedded within the neural network layers but is instead optimized simultaneously with the network parameters $\boldsymbol{\theta}$ using the same loss function defined in Eq. (2.9). During training, both $\boldsymbol{\theta}$ and θ_m are updated via backpropagation through the adjoint method (Section 2.1.3), ultimately enabling joint identification of dynamics and modal mass from displacement measurements alone.

This extended formulation offers three key benefits for system identification. The neural network f_θ is freed to concentrate exclusively on capturing intrinsic dynamical behavior. Meanwhile, the explicit parameter θ_m handles mass-dependent input scaling, facilitating direct modal mass estimation. By decoupling these learning objectives, the modified architecture overcomes the identifiability limitations that prevent standard Neural ODEs from simultaneously recovering both system dynamics and mass parameters in forced vibration problems.

Figure 6.5 illustrates the training procedure for the extended Neural ODE architecture. The training process operates iteratively through the following steps:

1. The dataset $\mathcal{D}$ contains measured system states $\mathbf{h}(t)$ and input forces $F(t)$ at discrete time points $t = t_1, \ldots, t_n$. For each training iteration i, an initial condition $\mathbf{h}(t_{s,i})$

and corresponding forcing $F(t_{s,i})$ are randomly selected from the training dataset and used as input for the Neural ODE.

2. Using the initial condition from step 1, numerical integration is performed from $t_{s,i}$ to a prediction time $t_{p,i}$ to predict the solution. The integration incorporates the FNN f_θ, which models the system's restoring acceleration, and the parameter θ_m, which scales the input force.

3. After solving the ODE in step 2, the predicted state $\hat{\mathbf{h}}(t_{p,i})$ and the current estimate of θ_m are obtained from the ODE solution.

4. The predicted response $\hat{\mathbf{h}}(t_{p,i})$ from step 3 is compared to the true measured state $\mathbf{h}(t_{p,i})$, and the loss $\mathcal{L}_N$ is computed according to Eq. (2.9).

5. The loss from step 4 is used to update both the parameters $\boldsymbol{\theta}$ and θ_m via backpropagation. Gradients are computed via the adjoint method and used to update both network weights $\boldsymbol{\theta}$ and mass parameter θ_m.

6. A new iteration is initiated with fresh initial conditions selected from the dataset. The process repeats with new initial conditions for the next training iteration until convergence.

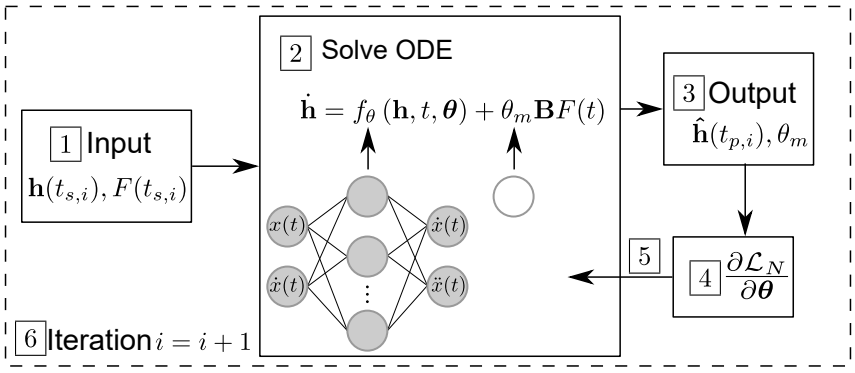

Figure 6.5: Extended Neural ODE architecture and training process for forced vibration systems with unknown modal mass. The neural network f_θ models intrinsic restoring dynamics while the mass parameter θ_m scales external forcing. The numbered steps ①–⑥ correspond to the iterative training procedure: ① random selection of initial conditions and forcing from dataset; ② forward integration over prediction horizon; ③ obtaining predicted state and mass parameter; ④ loss computation; ⑤ gradient computation and parameter updates via adjoint method; ⑥ iteration with new initial conditions.

For the TLCD application, once θ_m is identified through the training process, the Neural ODE provides the geometric factor as $\hat{\gamma}_1 = -\theta_m$. This geometric factor is essential for coupled structural analysis as demonstrated in Section 6.6, where the identified TLCD model is integrated into a structural simulation. The sensitivity analysis presented in Section 6.5.4 validates the necessity of both components of this extended formulation for accurate system identification.

6.4.1 Neural ODE Configuration

The TLCD dynamics are modeled using a Neural ODE with one hidden layer containing 20 neurons and sigmoid activation, followed by a linear output layer. This network size balances the expected complexity of the modeled TLCD against training variance, following principles discussed in Chapter 2. The architecture is deliberately not optimized, but initial tests suggest it is sufficiently large to capture system complexity while remaining compact enough to avoid introducing significant variance error. Since the Neural ODE consists of an internal FNN that essentially learns a force-state-like mapping and the anticipated nonlinearity has no memory, deep architectures are unnecessary, as discussed in [111].

Training parameters are as follows: 10,000 iterations, batch size 200, prediction horizon $N_t = 40$ (selected to handle measurement noise present in the dataset), ADAM optimizer with learning rate 0.01 and gradient decay 0.9. The training loss is obtained using the mean absolute error as the error function. An ensemble of 10 Neural ODEs with identical architecture but different random initializations is employed to reduce training variance, consistent with the ensemble approach in Chapter 5. After training, all ensemble members are evaluated on a test set, and the best-performing member is selected as the model.

6.5 Results and Discussion

6.5.1 Adaptation Cost Function Landscape

To demonstrate the performance of the proposed framework and visualize the optimization problem solved by the ADaGen framework, the adaptation cost function landscape is constructed through a comprehensive experimental campaign. It should be emphasized that the error surface is created solely for illustrative purposes and is not part of the methodology, nor is it assumed to be known. Training datasets are collected uniformly across the TLCD operational range, covering the parameter grid shown in Fig. 6.6a. For each parameter combination $\boldsymbol{\psi}$, the Neural ODE ensemble is trained sequentially with the response data corresponding to the sampling points. After each training, the best member is evaluated on the fixed test set to obtain $\mathcal{L}_T(\boldsymbol{\psi})$. This discrete evaluation of the adaptation cost function produces what can be visualized as an error surface. As demonstrated in Section 5.1.3, each training dataset $\mathcal{D}$ is produced based on a unique pair of sampling parameters $\boldsymbol{\psi}$, such that $\mathcal{D}(\boldsymbol{\psi})$, the adaptation cost function can be expressed in terms of the training set $\mathcal{L}_T(\mathcal{D})$, and hence in terms of the training set parameters $\mathcal{L}_T(\boldsymbol{\psi})$. By plotting each corresponding $\mathcal{L}_T(\boldsymbol{\psi})$ for each $\boldsymbol{\psi}$ in the sampling range, the error surface or visualization of the objective function is obtained.

Figure 6.6a illustrates the error surface, where dark blue indicates highly informative training datasets (low test error) and red indicates poor datasets (high test error). Each

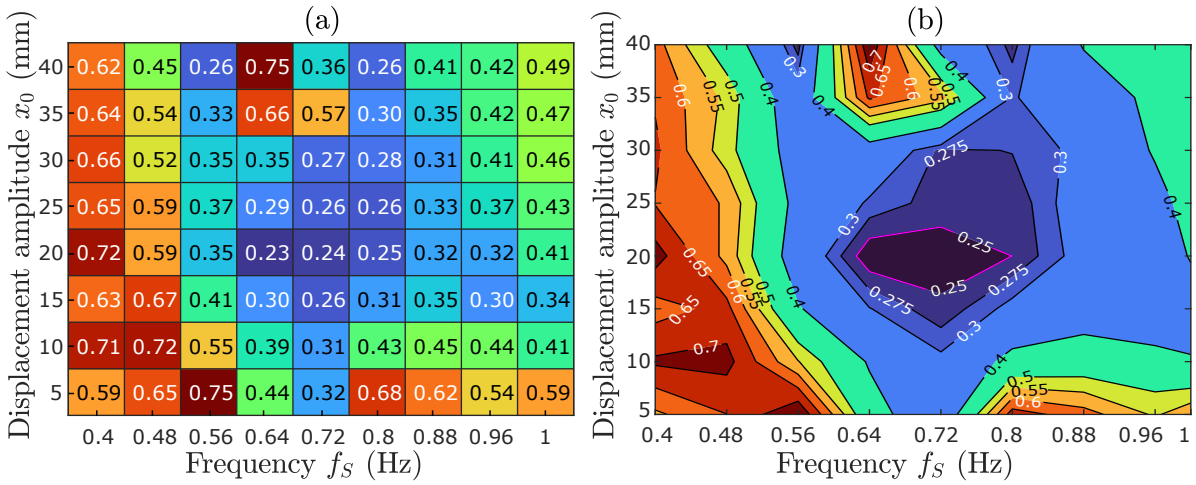

Figure 6.6: (a) Error surface representing the adaptation cost function landscape showing information content of each sampling point. Squares correspond to the training sets obtained using sine excitation with the displayed amplitude and frequency. Values indicate mean absolute error (MAE) on the test set after training with that dataset, with red indicating high error. Dark blue (low error) indicates highly informative datasets. (b) Linear interpolation of (a), providing a smooth representation of the error surface.

number inside the square on the graph shows the mean absolute error of the model evaluated on the test set after training with that training set. Hence, the surface regions shown in blue are the regions of high model accuracy. As described in Section 6.4.1, the errors shown in Fig. 6.6a correspond to the best-trained member of the Neural ODE ensemble. Figure 6.6b presents a linearly interpolated continuous representation obtained through linear interpolation of the discrete results. The region within the magenta boundary denotes the global minimum area where any sampling point yields an optimal training dataset satisfying the error tolerance criterion.

From the error surface shown in Fig. 6.6, several key observations can be made. First, the error surface has a minimum, which justifies the assumption that there is a region in the sampling space that is more informative and that optimal training datasets exist. Second, despite ensemble variance reduction, the surface shows local variations and noise, justifying the use of derivative-free PS optimization. Finally, it is evident that the best performance is at $\boldsymbol{\psi} = [20 \text{ mm } 0.64 \text{ Hz}]^\top$. Knowing that the natural frequency of the system, evaluated in a separate study, is 0.64 Hz, this expectation is met. However, the minimum model error does not align with the maximum excitation amplitude, contrary to intuitive expectations, underlining the need for the optimization algorithm. This discrepancy arises due to the simultaneous identification of the geometric scaling factor γ_1 and the state matrix by the Neural ODE. As the excitation amplitude increases, the transient stage of the response signal becomes less prominent, leading to non-unique solutions and, consequently, a less accurate model.

Figure 6.7 demonstrates that the optimal training dataset is attainable. The Neural ODE trained with $\mathcal{D}_{opt}$ accurately reproduces both displacement and velocity responses

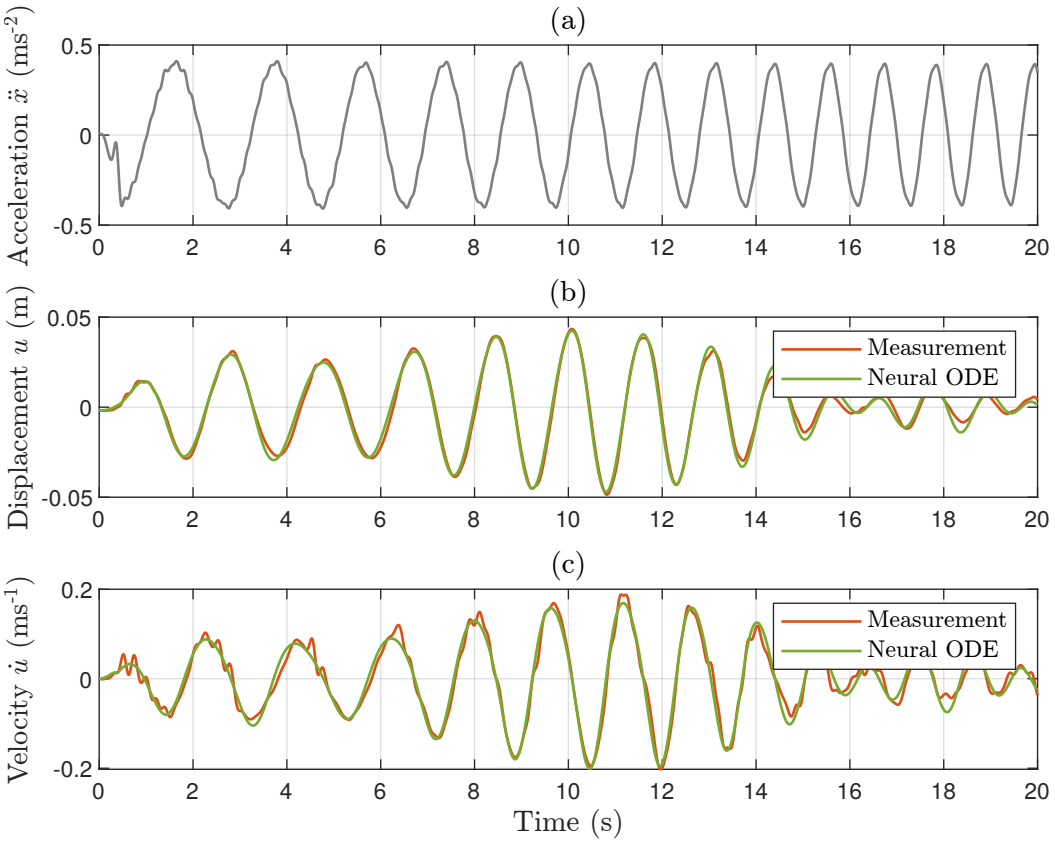

Figure 6.7: Plots showing the test dataset obtained using sine sweep excitation shown in (a) with the damper displacement (b) and velocity (c) shown in green. Furthermore, (b) and (c) show the performance of the Neural ODE when trained with the optimal training dataset obtained by the proposed framework. The model accurately captures dynamics across the full frequency range.

on the swept-sine test set, capturing the system's dynamics across the full frequency range. This confirms that sine signal excitation, despite being suboptimal for general system identification, contains sufficient information for this application when parameter selection is adaptive.

6.5.2 Comparison with Batch Sampling Methods

The ADaGen framework's efficiency is quantified by comparison against three conventional batch sampling methods: uniform random sampling, Latin hypercube sampling (LHS), and Sobol sequence sampling. Comparison with unsupervised sampling methods is a common approach for evaluating active learning techniques, and it has been shown that active learning methods do not always outperform them [147].

Similarly to Section 5.1.4, to obtain a statistically viable picture, the sampling methods are run 1000 times, each time sampling 5, 10, 15, and 20 points. For each method and sample size, the success rate (the fraction of campaigns finding $\mathcal{D}_{opt}$) is determined by

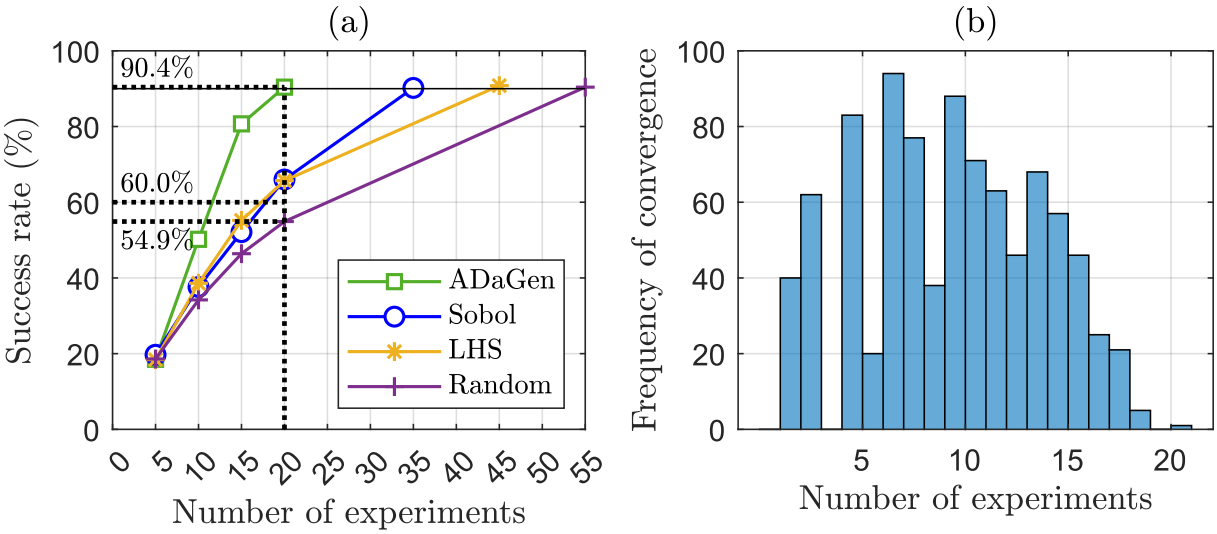

Figure 6.8: (a) Success rate of the proposed active learning framework compared to unsupervised sampling methods. ADaGen achieves 90% success with 20 experiments, while the best batch method requires at least 35 experiments for equivalent performance. (b) Histogram showing the frequency of convergence for the proposed framework across 1000 random initializations.

evaluating the model error and identifying the number of successfully trained models out of 1000 runs. For example, using random sampling with 20 points, the simulation is run 1000 times with different random distributions of 20 points, and $\mathcal{D}_{opt}$ was successfully identified 549 times, hence a success rate of 54.9%. The same procedure is applied to the other sampling methods for the aforementioned number of points, and the results are shown in Fig. 6.8a. The figure also shows the number of sampling points required by each method to achieve a success rate of at least 90%.

For the ADaGen framework, a comparable statistical analysis is conducted. For a fair comparison, the proposed active learning framework is initialized 1000 times with the Halton sequence sampling. This approach ensures uniform distribution of starting points and eliminates any bias that could result from initial sampling. For each starting point, PS optimization is executed, and the number of experiments (sampling points) required to identify the optimal training dataset is counted for each sequence to obtain the success rate. After running all 1000 simulations, the histogram shown in Fig. 6.8b is obtained, which shows the frequency of success of the ADaGen framework. From this histogram, the cumulative histogram is calculated and shown in Fig. 6.8a, plotted alongside the batch method success rates, enabling direct performance comparison.

Figure 6.8a reveals performance advantages for adaptive sampling. After 20 experiments, ADaGen achieves 90.4% success rate, while the best batch methods (LHS and Sobol) reach only 66%, and random sampling performs at 55%. The ADaGen framework finds $\mathcal{D}_{opt}$ with an average of 10 experiments, as shown in Fig. 6.9d, representing a 2× efficiency gain over batch methods at the 90% success threshold. All methods eventually reach 90% success: random sampling at 55 experiments, LHS and Sobol at 35-45 experiments.

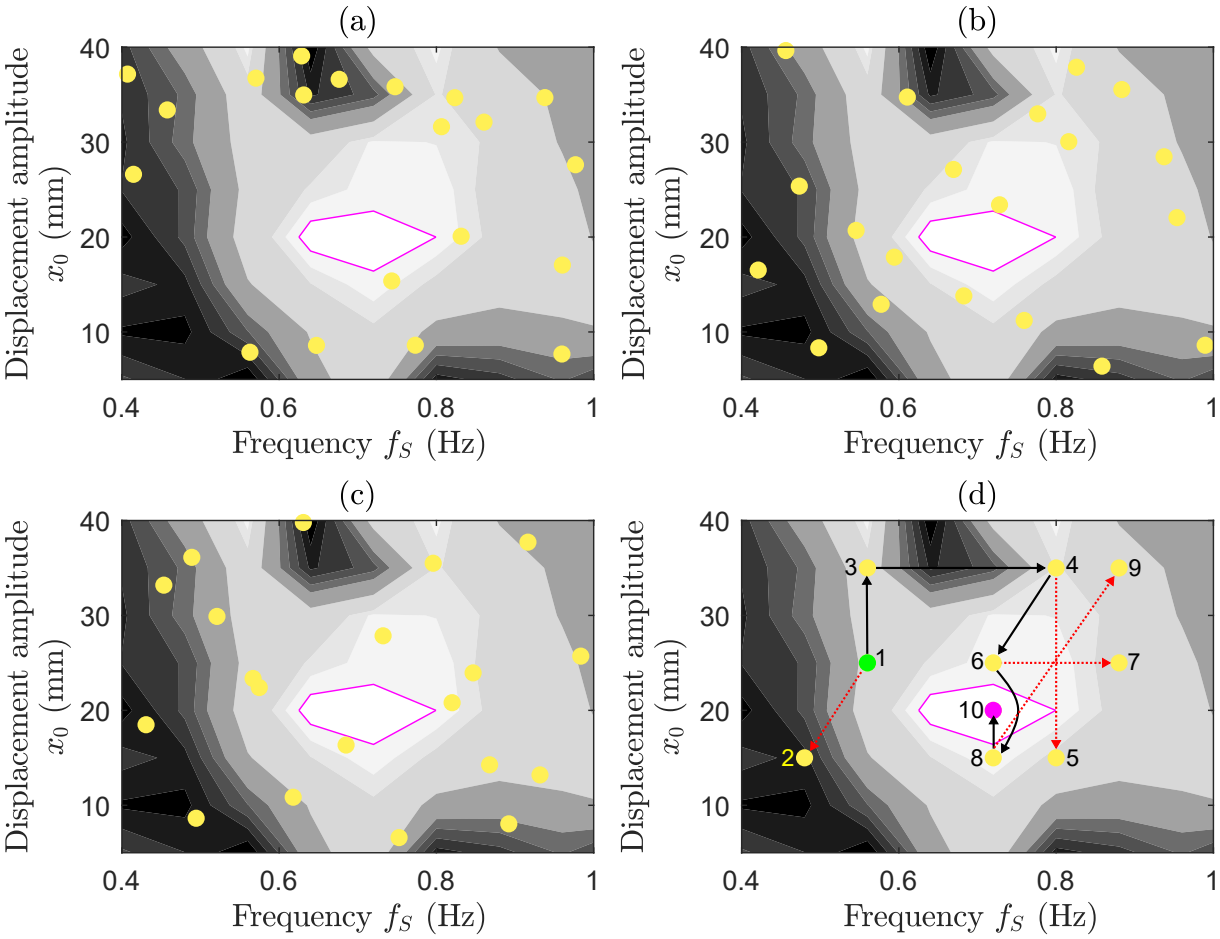

Figure 6.9: Figures showing an instance of sampling point distribution using (a) random, (b) Latin hypercube, and (c) Sobol sequence sampling methods. Figure (d) shows an example of typical convergence with the proposed active learning framework. The framework starts with an initial sampling point (green) and finds the optimal training dataset (magenta). Since the proposed approach requires on average 10 experiments (cf. Fig. 6.8b) to achieve the defined success rate, this case is depicted in the figure.

Figure 6.9 illustrates the placement of points produced by unsupervised methods and the proposed framework on the error surface. Batch methods (a-c) distribute points across the entire parameter space according to their respective strategies, with no adaptation based on observed performance. ADaGen (d) concentrates sampling in the promising region once initial exploration identifies the approximate minimum location, exemplifying efficient information-directed experimentation.

It should be mentioned that the efficiency of unsupervised sampling methods is enhanced by the constrained sampling space. Expanding the sampling space would make it more challenging for these methods to effectively cover the area. In higher dimensions, uniform space coverage becomes exponentially more expensive. Thus, the observed advantage is likely conservative, with greater gains expected for problems with more experimental parameters.

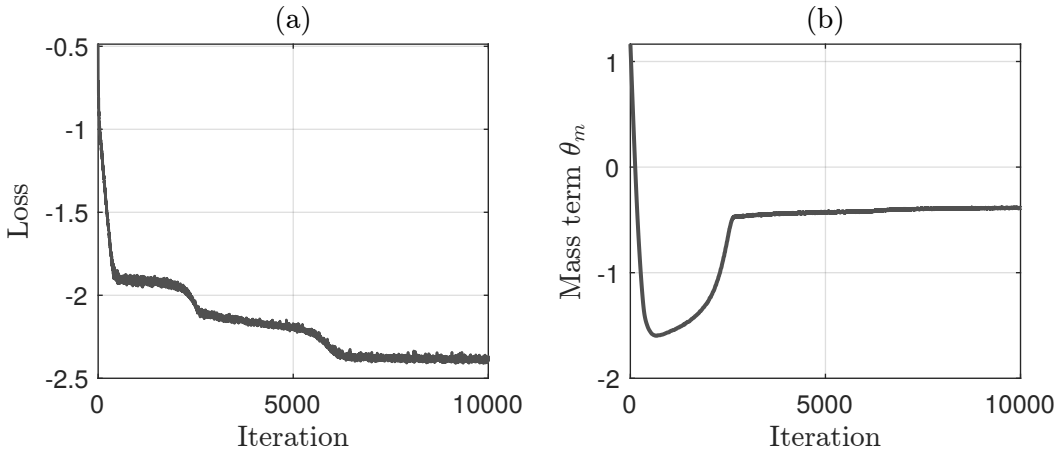

Figure 6.10: (a) Training history of the Neural ODE for 10,000 iterations calculated as mean absolute error and shown on a logarithmic scale. (b) Plot showing the estimated value of θ_m from which the geometric factor is determined as $\hat{\gamma}_1 = -\theta_m = 0.40$.

6.5.3 Ensemble Variance Reduction and Neural ODE Validation

To validate the extended Neural ODE formulation for unknown mass identification and the ensemble approach for variance reduction, training results are shown in detail. Figure 6.10a shows the training loss obtained using the mean absolute error as the error function for the optimal training dataset, indicating that the model is learning effectively from the data. The loss function exhibits characteristic step-wise convergence over 10,000 iterations. The training starts with rapid decline, reaching steady convergence after around 7000 iterations. Since measurement noise is low, approximately 3%, it does not have a large impact on model training.

Figure 6.10b shows the training history of the mass parameter θ_m. The plot shows the estimated value from which the geometric factor is determined as $\hat{\gamma}_1 = -\theta_m$. It can be seen that it reaches a stable value during the training of the Neural ODE. The identified value converges to $\hat{\gamma}_1 = 0.39$, exhibiting strong agreement with predictions from analytical methods presented in [91]. This demonstrates successful simultaneous identification of both system dynamics and modal mass. In the presented case, the Neural ODE model was able to identify a meaningful geometric factor, even when initialized with a physically impossible negative value, demonstrating the robustness of the proposed approach.

Furthermore, the effectiveness of using an ensemble of Neural ODEs to eliminate the training variance is investigated and demonstrated in Fig. 6.11. Figure 6.11a presents the error surface when a single Neural ODE is utilized for each parameter combination with random initialization. This figure highlights the variance and potential instability in the training process when relying on a single model. The surface exhibits substantial noise and irregularities, making optimization unreliable.

Figure 6.11b demonstrates the improvement in the error surface when the network ensemble is used and shows the dramatically improved surface when employing the 10-member

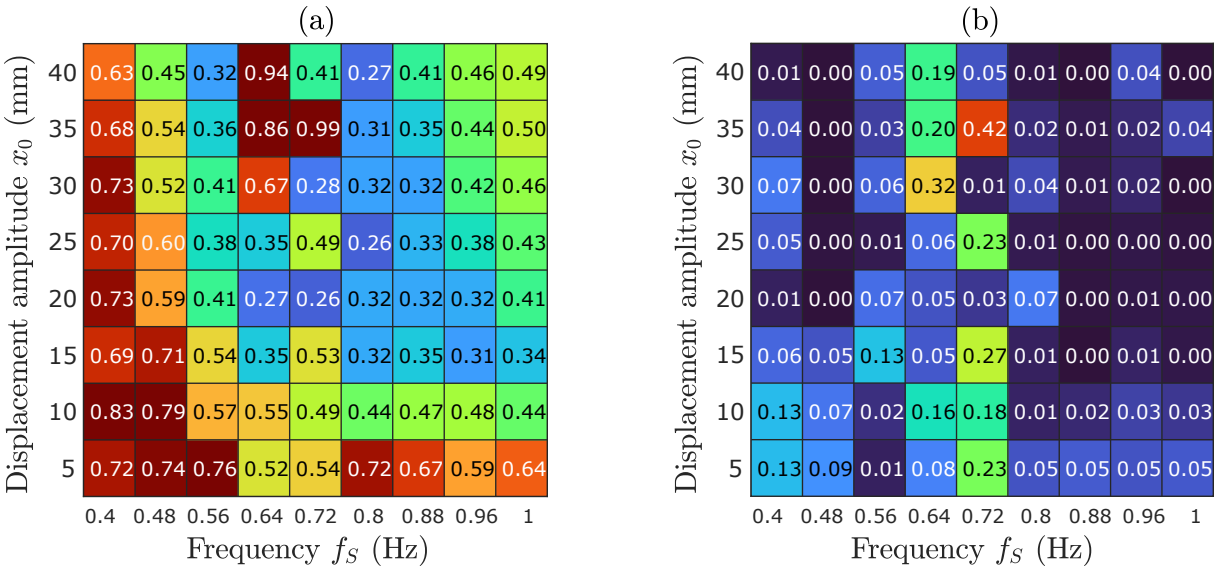

Figure 6.11: The difference in the obtained error surface when a single Neural ODE is used (a) and when an ensemble of Neural ODEs is used (b). The ensemble is used to reduce the variance error coming from the training algorithm.

ensemble with best-member selection, indicating a more stable outcome. The noise is significantly reduced, revealing the underlying structure with a clear global minimum. The shown values are calculated by subtracting the error surface obtained using the ensemble shown in Fig. 6.6a from the error surface obtained using a single Neural ODE in Fig. 6.11a. This comparison confirms that ensemble methods are essential for stable adaptation cost function evaluation, without which the ADaGen framework would struggle to identify optimal experimental parameters, confirming the results from Section 5.1.3.

6.5.4 Sensitivity Analysis of Neural ODE Extensions

To verify the generalization capability of the identified model beyond sine excitation, the optimal Neural ODE is evaluated on a broadband random noise signal, as shown in Fig. 6.12. Despite the substantial difference from the sine training signals, the model accurately predicts both displacement and velocity responses. This demonstrates that the identified dynamics capture the underlying physical behavior rather than merely memorizing the training signal characteristics, confirming robust generalization to unseen excitation patterns.

A sensitivity study is conducted to systematically evaluate the individual contributions of the forced-input term and the modal mass parameter θ_m in the extended Neural ODE formulation. The necessity of each component is demonstrated through controlled numerical experiments. Three distinct training cases are designed to separate the effects of forced excitation and modal mass parameterization, as presented in Table 6.1. All trained models are evaluated on identical sine-sweep reference data to ensure consistent comparison of generalization performance.

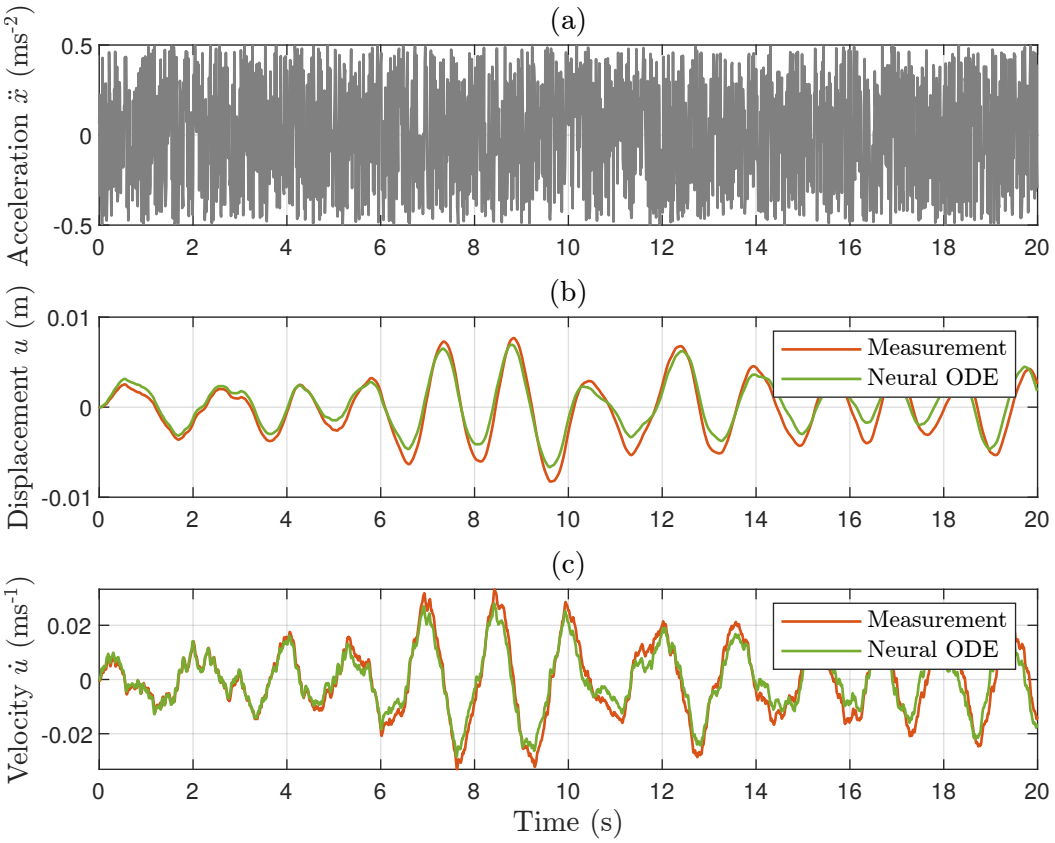

Figure 6.12: Performance of the Neural ODE model in predicting broadband noise signal: (a) random noise excitation, (b) comparison of displacement responses, (c) comparison of velocity responses. The model demonstrates high accuracy despite the significant difference from the sine training signal.

Case	Training Excitation	Modal Mass (θ_m) Known
1	Initial conditions only	Yes
2	Initial conditions only	No (learned)
3	Harmonic excitation	No (learned)

Table 6.1: Sensitivity study design to isolate the effects of forced excitation and modal mass parameter identification.

The comparison between Cases 1 and 2 isolates the contribution of the modal mass parameter when forced excitation is absent during training. As illustrated in Figure 6.13 (Case 1), when the modal mass is provided, training on initial conditions alone yields excellent performance on the reference signal. The model successfully captures the system dynamics and accurately predicts both displacement and velocity responses. On the other hand, Figure 6.14 (Case 2) reveals the necessity of the modal mass parameter. When θ_m is unknown, the model cannot achieve proper response scaling, despite capturing the general dynamic behavior. The predicted responses show correct temporal patterns but incorrect amplitudes, establishing that the modal mass parameter is essential for accurate forced response prediction.

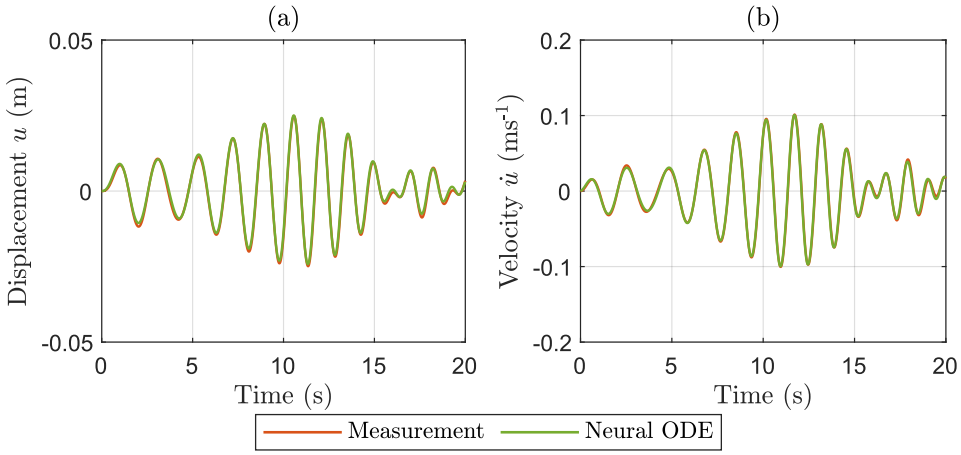

Figure 6.13: Case 1: training with initial conditions and known modal mass. Comparison of displacement (a) and velocity (b) responses. The model successfully learns system dynamics from free response data.

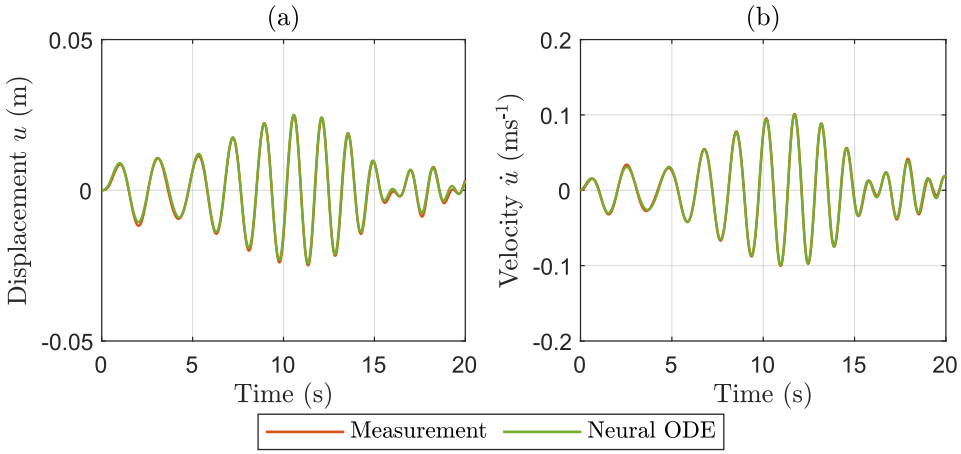

Figure 6.14: Case 2: training with initial conditions and unknown modal mass. Comparison of displacement (a) and velocity (b) responses. Temporal patterns are captured correctly, but amplitude scaling is incorrect due to the missing mass parameter.

In Figure 6.15 (Case 3), which represents the proposed active learning framework, the model successfully learns both the system dynamics and identifies the modal mass. Training with harmonic excitation while learning θ_m simultaneously enables accurate dynamics learning and correct mass identification. This establishes that forced excitation data is necessary for modal mass identification since free vibration data alone is insufficient.

The training process is further examined when θ_m is learned jointly with the neural network parameters. Figure 6.16 compares the training loss convergence with and without modal mass parameter learning. The results indicate that including θ_m as a learnable parameter introduces only a marginal increase in training iterations, with negligible computational overhead, demonstrating the efficiency of the joint learning approach. Furthermore, the evolution of the modal mass parameter during training is presented in

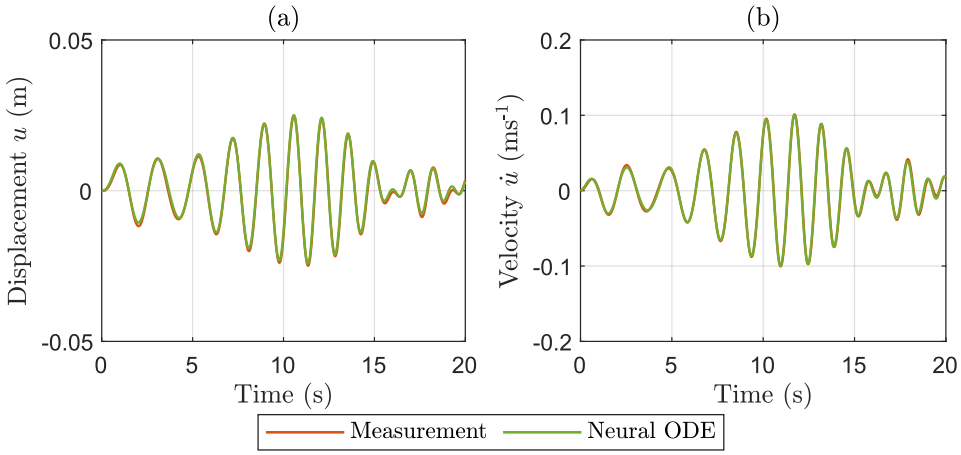

Figure 6.15: Case 3: training with harmonic excitation and unknown modal mass. Comparison of displacement (a) and velocity (b) responses. The model successfully learns both system dynamics and modal mass.

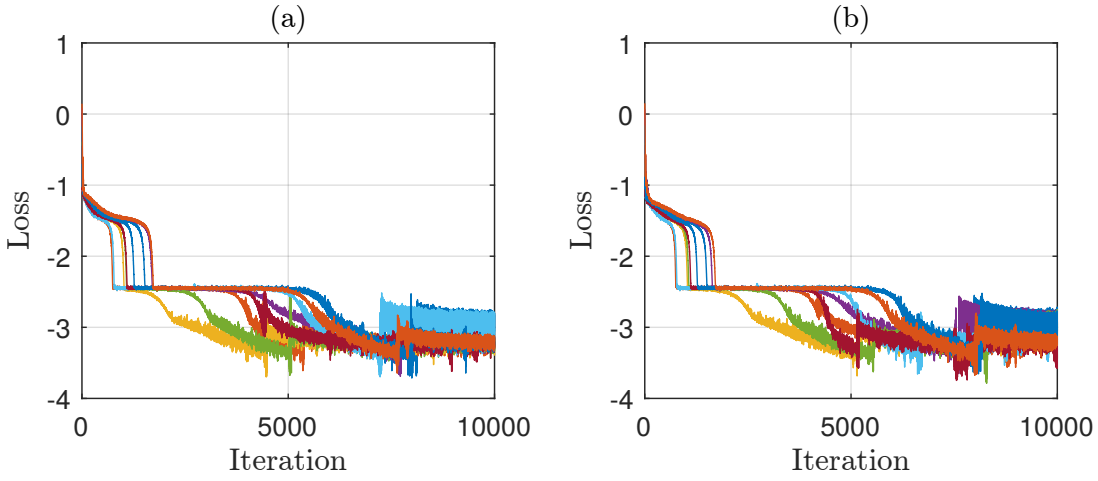

Figure 6.16: Training loss convergence comparison with (a) and without (b) modal mass parameter learning. Each color represents a run with a different initial value. The identification of the modal mass introduces minimal additional computational cost.

Figure 6.17. The parameter reliably converges to the correct value across multiple training runs, confirming the stability and robustness of the identification process when using informative excitation signals.

This sensitivity study establishes several key findings. First, the comparison of Cases 1 and 2 confirms that θ_m is essential for accurate amplitude prediction in forced systems. Second, the comparison of Cases 2 and 3 demonstrates that forced excitation data is necessary for modal mass identification. Third, the joint learning of network parameters and θ_m is computationally efficient and numerically stable. Finally, the modal mass parameter reliably converges to the correct value when trained with informative excitation signals. These results validate both extensions to the standard Neural ODE formulation as sufficient for accurate dynamical system identification with unknown mass parameters.

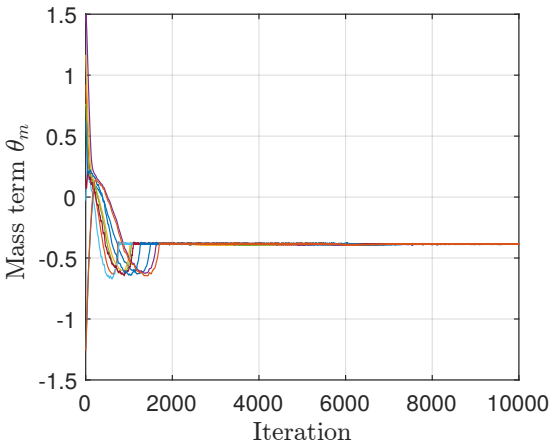

Figure 6.17: Modal mass parameter convergence history during training across multiple runs with varying initial values (different colors). The reliable convergence to the correct value, regardless of initialization, demonstrates the robustness of the identification process.

6.5.5 Pattern Search Parameter Sensitivity

A sensitivity analysis of the PS optimization parameters is conducted to evaluate the robustness of the optimization and justify the selected algorithm settings. All experiments are performed on the same TLCD identification problem, where the error surface shown in Figure 6.6 serves as the objective function. The analysis systematically examines three key parameters:

- *Initial mesh size*: 0.5, 1, 2, 4, and 8 (in scaled parameter space [0,10])
- *Poll method*: Four different polling strategies were tested, varying in the number and direction of search points evaluated at each iteration
- *Initial guesses*: 1000 initial points uniformly distributed using Halton Sequence Sampling

Figure 6.18(a) illustrates the sensitivity to initial mesh size. The initial mesh size substantially influences optimization performance, with the optimal mesh size of 4 achieving the best overall performance, validating the choice adopted in this work. Medium-range mesh sizes (2-4) provide an optimal balance between exploration capability and convergence precision. Conversely, very small mesh sizes (0.5) lead to premature local convergence. This observation aligns with previous findings from Section 5.1.5.

Figure 6.18(b) presents a comparison of the examined poll methods. The four investigated poll methods differ in how they generate candidate points around the current best solution. Poll Method #1 evaluates 3 points using a minimal positive basis with directions along the positive coordinate axes and one negative diagonal direction. Poll Method #2 evaluates 4 points using a positive basis that includes all positive and negative coordinate directions (i.e., $2N$ directions in an N-dimensional space). Poll Methods #3 and

#4 employ adaptive mesh strategies that dynamically adjust the search directions based on local geometry: Poll Method #3 uses $N + 1$ directions with Mesh Adaptive Direct Search (MADS), while Poll Method #4 uses an orthogonalized mesh with $2N$ directions, enabling more systematic exploration of the search space.

Results demonstrate that the choice of poll method can significantly influence algorithm performance. Poll Method #2 achieves the highest overall success rate but requires more experiments. Poll Method #1, used in Section 6.5.2, reaches 90% success rate with fewer average function evaluations, providing superior efficiency. Poll Method #1 uses 3 poll points versus 4 for Poll Method #2, explaining the efficiency advantage when fewer experiments are required. Both Poll Methods #3 and #4 show inferior performance for this specific problem. Therefore, the choice of Poll Method #1 represents an optimal balance between reliability and computational efficiency.

In Figure 6.18(c), Poll Method #2 is further evaluated with different initial mesh sizes to examine whether its performance could be enhanced. Testing Poll Method #2 with varying initial mesh sizes confirms that the mesh size effect is consistent across poll methods. Nevertheless, the reference configuration (Poll Method #1, mesh size of 4) yields the best overall performance.

Finally, Figure 6.18(d) presents the initial guess distribution analysis. The spatial distribution of converged and non-converged initial points reveals a clear correlation between proximity to the global optimum and convergence success. Initial points in the range of $[0.5 \ldots 0.6]$ Hz exhibit higher failure rates, likely due to mesh size causing the algorithm to overshoot potential solutions in this region. A few scattered non-converged points are observed, which can be attributed to the stochastic nature of the optimization landscape. These initial points were also used to obtain the convergence statistics shown in Figure 6.9.

The findings confirm that the chosen pattern search configuration was deliberately selected to minimize function evaluations while maintaining effectiveness. The sensitivity analysis validates this choice by demonstrating that the selected parameters (mesh size 4, Poll Method #1) provide near-optimal performance. Further tuning of these parameters yields marginal improvements in success rate, but at the cost of a substantially higher number of function evaluations, i.e., experiments. As noted, the parameter space was scaled to [0,10] for both amplitude and frequency to ensure algorithmic generality across different TLCD configurations. This scaling approach, combined with the validated parameter settings, establishes a robust optimization framework that can be adapted to other problems.

6.5.6 Comparison with Genetic Algorithm

For the proposed active learning framework, pattern search was selected due to its demonstrated efficiency in other applications with respect to the number of required function evaluations, which in this context correspond directly to physical experiments. While more advanced optimization algorithms could also be used when experimental budgets

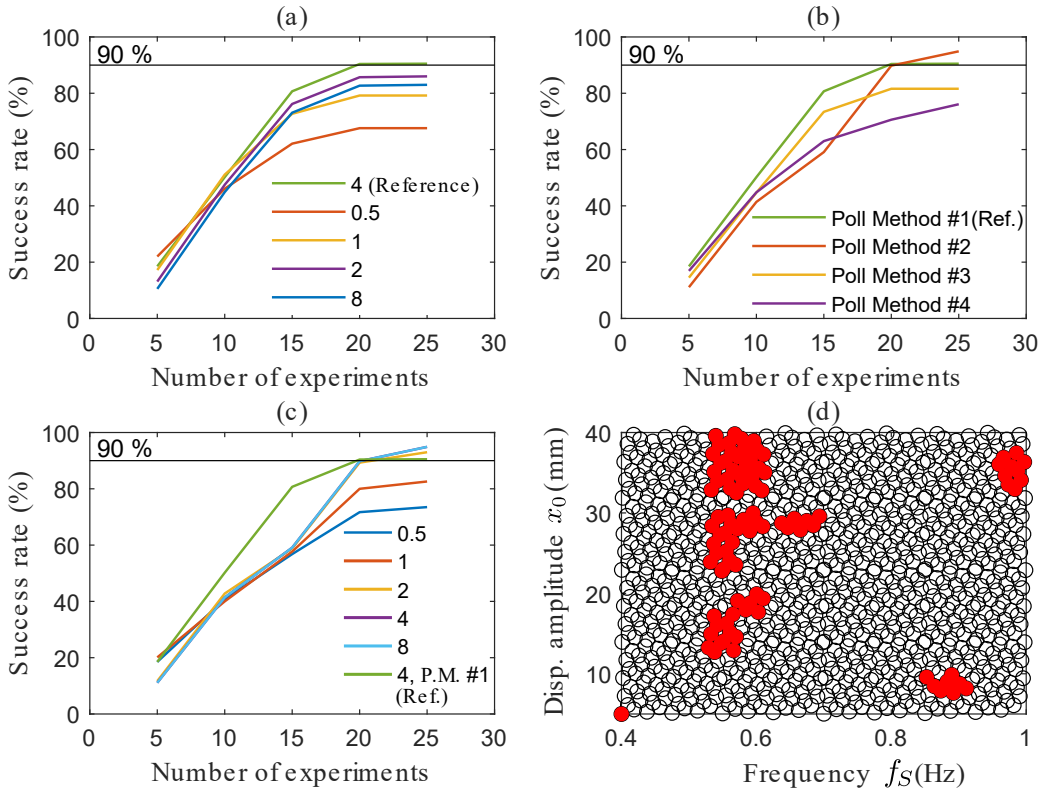

Figure 6.18: Sensitivity analysis of pattern search optimization parameters: (a) influence of initial mesh size on convergence behavior; (b) comparison of different poll methods; (c) combined influence of poll method and mesh size; (d) spatial distribution of converged (hollow) and non-converged (red) initial points in the parameter space.

and timelines allow, algorithms such as pattern search offer clear advantages when each function evaluation requires substantial time and resources.

To justify the choice of optimization algorithm, a comparison between pattern search and a genetic algorithm is conducted using identical problem setups and convergence criteria, as described in Section 6.5.2. Both algorithms are tested in 1000 independent optimization runs to ensure statistical significance.

The results in Figure 6.19 demonstrate clear advantages of pattern search for this application. Both algorithms are configured to achieve a success rate of 90% across 1000 runs to ensure a fair comparison. Figure 6.19(a) shows the distribution of function evaluations for both algorithms. The pattern search exhibits a more concentrated distribution around the median value, whereas the genetic algorithm shows substantially greater variability, as reflected by its higher standard deviation and a larger number of outliers that require excessive function evaluations. This variability makes pattern search more predictable and therefore better suited for experimental planning, where resource allocation and time scheduling are critical constraints.

Quantitatively, pattern search requires significantly fewer function evaluations on average, with a mean of 9.0 and a standard deviation of 4.2, compared to the genetic algorithm's

mean of 14.5 and a standard deviation of 11.9. The median number of function evaluations further highlights this efficiency advantage: pattern search required only 9.0 evaluations compared to 13.0 for the genetic algorithm, representing approximately a 31% reduction in experimental cost.

This empirical comparison validates the choice of pattern search as the optimization algorithm for the proposed framework, demonstrating its superior efficiency and reliability for experimental parameter identification in this application.

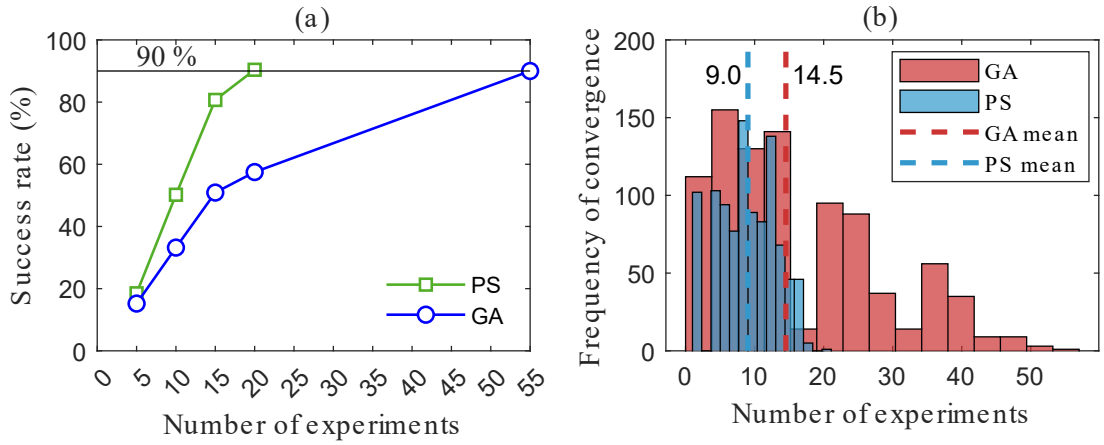

Figure 6.19: (a) Comparison of the success rates of the pattern search (PS) and the genetic algorithm (GA). (b) Histogram showing the frequency of convergence for the pattern search and the genetic algorithm for 1000 runs.

6.6 Application Example

This section demonstrates how the Neural ODE model identified through the ADaGen framework can be integrated into coupled structural simulations for performance evaluation. The goal is to assess whether the damper properties are appropriate for the structure. In particular, an SDOF system subjected to a dynamic force is investigated, Fig. 6.20a. The objective is to assess whether the modeled TLCD prototype is suitable for mitigating vibrations of a single-degree-of-freedom structure subjected to dynamic excitation. No damper optimization is performed; rather, the focus is on utilizing the obtained model within a numerical simulation framework. The application process is similar for MDOF systems with either single or multiple TLCDs. For damper optimization procedures, the reader is referred to [9].

The equation of motion representing the structure reads as follows:

$$\ddot{x} + 2D_s\omega_s\dot{x} + \omega_s^2 x = -\mu(\ddot{x} - \hat{\gamma}_2\ddot{u}) + f, \tag{6.7}$$

where the damping ratio of the structure is $D_s = 1\%$, the natural frequency is $\omega_s = 4.22$ rad/s corresponding to $f_s = 0.67$ Hz, and the mass ratio of the damper liquid mass

to the modal mass of the structure is $\mu = m_d/m_s = 11.7/180 = 6.5\%$. Additionally, $f = F(t)/m_s$ represents the time-dependent, mass-normalized excitation. Corresponding to the test signal with which the Neural ODE model was obtained, a sine-sweep signal is chosen in the range of 0.40–1.00 Hz as excitation, shown in Fig. 6.20b.

When coupled with the structure, a second geometric factor, $\hat{\gamma}_2$, has to be estimated. Similar to the first geometric factor, γ_1, it depends on the TLCD's geometry. Since the cross-sectional areas, A_V and A_H, can be easily measured, they are assumed to be known. However, as mentioned before, L_H and L_V are approximations of the liquid flow direction and are particularly challenging to determine for complex TLCD layouts. Therefore, L_H is assumed based on the TLCD's geometry and the missing L_V is estimated by substituting the first geometric factor, $\hat{\gamma}_1$, estimated by the Neural ODE, in the corresponding Eq. (6.2).

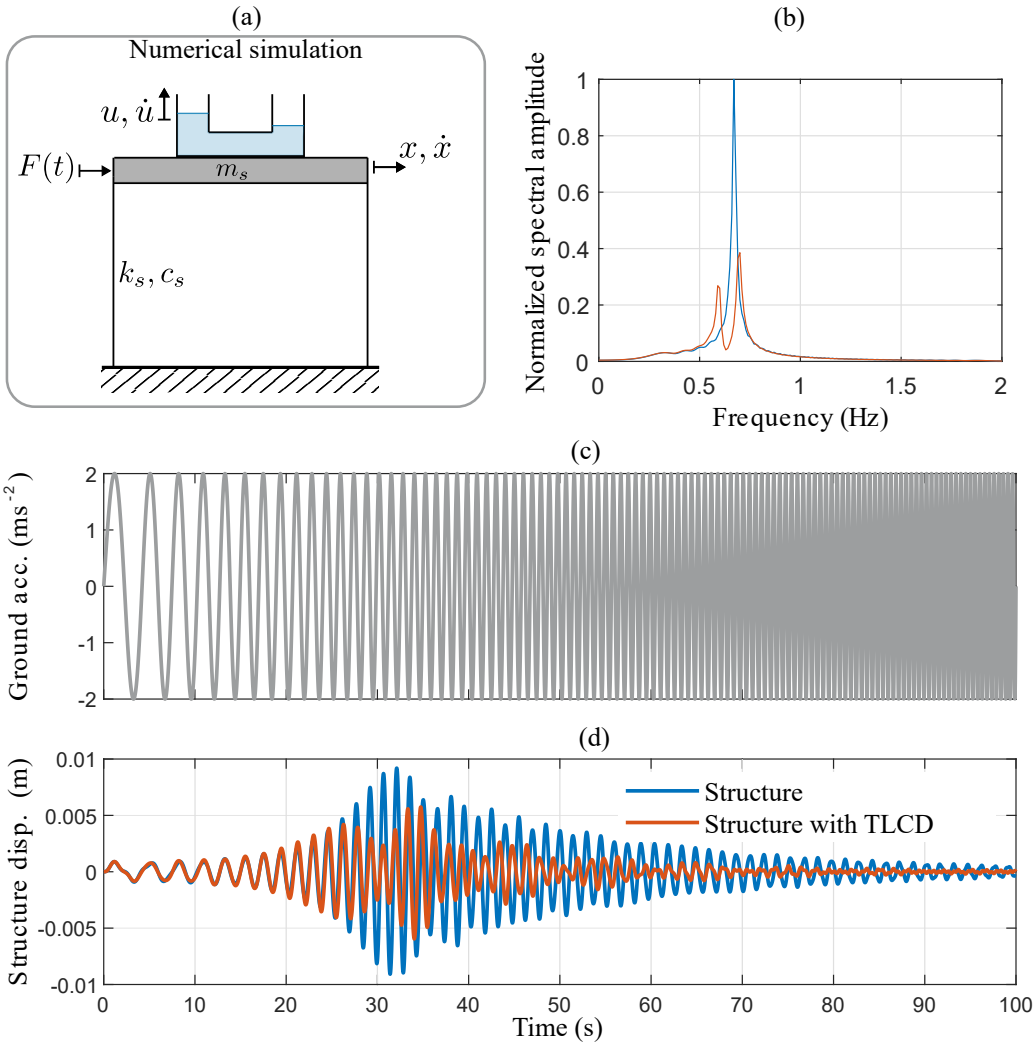

Figure 6.20: (a) The studied structure with the TLCD used in simulation. (b) The frequency response of the structure with and without damper. (c) Sine-sweep excitation signal used for evaluation.(d) Displacement of the structure with the model of the TLCD compared to an SDOF structure without a damper.

Once L_V is identified, the second geometric factor can then be estimated from:

$$\hat{\gamma}_2 = \frac{L_H}{2L_V + \frac{A_H}{A_V} L_H}. \tag{6.8}$$

In this way, any error introduced by the approximation of L_H is effectively removed.

In Eq. (6.7), the acceleration response of the TLCD, $\ddot{u}$, is estimated by the Neural ODE model. Corresponding to Eq. (6.1), this model uses the acceleration of the structure, $\ddot{x}$, multiplied by the first geometric factor, $\hat{\gamma}_1$, as its dynamic force input. A simultaneous inference of the Neural ODE model with Eq. (6.7) yields the dynamic response of the structure-TLCD system.

The example structure has mass $m_s = 180$ kg, stiffness $k_s = 3200$ N/m, and damping coefficient $c_s = 15$ Ns/m. It can be seen from Fig. 6.20c that the frequency response of the structure is significantly improved when compared to the uncontrolled case. This is further highlighted in Fig. 6.20d, which shows the displacement time histories of the structure with and without the TLCD, demonstrating substantial vibration mitigation achieved by the TLCD during the swept excitation. The peak displacement is reduced significantly, confirming the damper's effectiveness.

This example demonstrates that Neural ODE models identified through the ADaGen framework can be directly integrated into structural analysis software for performance prediction and design verification. While the TLCD parameters could be further optimized for this specific structure, such optimization is beyond the scope of this study. The identified model enables realistic assessment of device effectiveness under operational conditions without requiring extensive physical prototyping or iterative testing cycles.

6.7 Summary

This chapter validates the ADaGen framework experimentally using a TLCD prototype, demonstrating three key advances beyond the numerical validation in Chapter 5.

First, Neural ODE integration extends the framework from FNN force-state mapping to continuous-time dynamics modeling. The modified architecture enables learning from displacement measurements alone, eliminating force transducers and broadening applicability to devices where direct force measurement is impractical. Unknown modal mass identification is achieved through explicit parameterization of the mass-dependent input scaling factor θ_m. Sensitivity studies confirm both modifications are necessary and sufficient for accurate system identification, with the identified geometric factor closely matching theoretical expectations. The parameter converges reliably across different initializations, even from physically impossible values, and the model generalizes accurately to broadband random excitation despite training exclusively on sine signals.

Experimental validation on the physical system confirms the framework's effectiveness. Error surface visualization reveals a clear global minimum, validating the optimization-based approach. ADaGen achieves 90% success with 20 experiments compared to 35-45 required by the best batch methods (random, LHS, Sobol), finding optimal datasets with an average of 10 experiments. Sensitivity analyses validate key algorithmic choices, showing that chosen pattern search parameters provide optimal balance between exploration and exploitation. Comparison with genetic algorithms demonstrates 31% reduction in average experimental cost with more predictable convergence. The ensemble approach proves essential for reducing training variance, as shown by comparing single-model versus ensemble-based error surfaces.

The identified Neural ODE model integrates directly into coupled structural simulations. The application example demonstrates significant vibration reduction for an SDOF structure under dynamic excitation, confirming that ADaGen-generated models are immediately applicable for design verification and performance prediction. Overall, this experimental validation demonstrates that ADaGen efficiently transforms data collection into an optimization problem, yielding compact, high-information datasets.

Chapter 7

Extensions and Generalizations

7.1 Introduction and Motivation

The adaptive data generation framework presented in Chapter 4 and validated in Chapters 5 and 6 has demonstrated significant improvements in data efficiency for structural dynamics applications. However, the framework operates under several critical assumptions that, while enabling the initial development and validation, limit its applicability to broader classes of problems. This chapter systematically addresses these limitations and explores extensions that enhance the framework's versatility and practical utility.

While the assumptions made in Chapter 4 enabled rigorous validation of the core concepts, real-world structural dynamics applications frequently violate one or more of these constraints. For instance, comprehensive test data may be unavailable due to safety constraints, cost limitations, or physical inaccessibility of certain operating regimes. Similarly, complex nonlinear systems or MDOF structures may require multiple experiments to adequately sample their dynamic behavior. The random initialization of neural networks, while theoretically justified by universal approximation theorems, introduces significant variance that complicates the optimization process and increases computational demands.

This chapter addresses these limitations through four key extensions. These extensions follow a improve the baseline framework by making it more efficient through Interpretable Machine Learning (IML) in Section 7.2, handling cases where it cannot work as-is and requiring a novel cumulative training approach described in Section 7.3, to expanding its scope to MDOF systems (Section 7.4), to operating in its most challenging scenario where no test set is available (reference-free) in Section 7.5.

These extensions represent critical steps toward a more robust and versatile methodology capable of addressing complex, multi-dimensional problems in structural dynamics while maintaining the adaptive capabilities demonstrated in previous chapters. The exploration proceeds systematically, building from interpretable model initialization through cumulative training strategies to fully exploratory approaches that eliminate the test data requirement entirely.

7.2 Interpretable Neural Network Initialization

7.2.1 The Initialization Problem

The adaptive data generation framework relies on evaluating model performance to distinguish between training datasets of varying quality. However, randomly initialized neural networks introduce significant performance variance that can obscure meaningful differences between datasets. If we consider training ten networks on the same data, as was done in the model pool in Chapters 5 and 6, their test errors may vary greatly depending solely on initialization. This variance directly undermines optimization efficiency. If differences between datasets are smaller than initialization-induced variance, the optimization algorithm cannot reliably identify superior training configurations.

The baseline ADaGen framework addressed this challenge through ensemble and model pool methods, hence training multiple networks per candidate dataset and selecting the best performer. While effective, this approach multiplies computational cost linearly with ensemble size. For the studies in Chapters 5 and 6, ten-network ensembles were required to achieve reliable optimization, representing a tenfold computational overhead per framework iteration.

This section demonstrates that *interpretable initialization* eliminates ensemble requirements entirely. By initializing network parameters purposefully rather than randomly, a single network achieves comparable performance to ten-network model pool at 10% of the computational cost. The key insight is that a physically informed initialization provides a quasi-deterministic starting point that exhibits low variance across training runs, enabling reliable single-network optimization. Since the interpretable initialization has already been researched and developed within the context of Interpretable Machine Learning (IML), the goal of this section is its integration within the active learning context.

7.2.2 Interpretable Machine Learning Framework

Interpretable machine learning , developed by Pei and colleagues [104, 105, 107, 108, 109, 110], provides a constructive framework for neural network design grounded in the universal approximation theorem [33, 59]. While this theorem guarantees that single hidden layer networks can approximate any continuous function arbitrarily well, it provides no guidance on architecture selection or parameter initialization. IML fills this gap through systematic techniques that assign physical meaning to network weights and biases. The framework employs three primary construction methods:

Taylor Series Expansion Neural networks approximate polynomials by exploiting Taylor series expansions of sigmoid activation functions $\sigma(z) = (1+e^{-z})^{-1}$ around selected points. By choosing specific weights and biases, individual neurons represent polynomial

terms of desired orders. Proper input/output scaling is critical, as approximation accuracy degrades away from expansion points.

Geometric Construction This method exploits sigmoid geometric properties: small weights with zero bias produce linear behavior, large weights with zero bias approximate step functions, and various weight/bias combinations generate symmetric or antisymmetric functions. The approach extends naturally to multi-input cases while maintaining parameter interpretability after training [107].

Function Decomposition Complex functions decompose into sums of simpler components that are individually easier to approximate. For structural dynamics, restoring forces naturally split into displacement-dependent spring components and velocity-dependent damping terms—a decomposition that aligns with physical understanding [106].

These techniques culminate in a *three-prototype framework* [105, 110] designed for ten typical engineering nonlinearities: linear, cubic, bilinear, multi-slope, fractional power, softening cubic, dead space, hard saturation, saturation, and stiction. The framework employs a three-stage procedure: (I) select the appropriate prototype matching target function characteristics, (II) choose the specific variant for the given function class, and (III) adjust scales, weights, and biases to achieve the desired transformation. The three prototypes are:

- *Prototype 1* (3 nodes): Sigmoid function with constant removal for softening to hard-limiting nonlinearities,
- *Prototype 2* (4 nodes): Taylor series-based with cubic approximation for hardening to dead-space nonlinearities,
- *Prototype 3* (3 nodes): Partial derivative-based using subtracted sigmoids for anti-symmetrical functions.

The approach applies directly to systems exhibiting nonlinearities within Pei's ten-class framework which cover the majority of common structural dynamics applications. For these systems, IML initialization is immediately deployable without modification.

7.2.3 Application to Duffing Oscillator

The Duffing oscillator from Section 5.1 provides a suitable test case for evaluating IML integration with ADaGen framework. The restoring force, expressed as:

$$f_R(x,\dot{x}) = \beta x + \gamma x^3 + c\dot{x} \tag{7.1}$$

decomposes naturally into spring stiffness (displacement-dependent) and damping (velocity-dependent) components, which is precisely the functional separation IML leverages.

The proposed initialization combines Prototype 2 for the nonlinear spring force and Prototype 3 for the damping force. Specifically, four nodes capture the combined linear-cubic stiffness $\beta x + \gamma x^3$. Although Prototype 2 is primarily designed for cubic functions, training adapts it to accommodate the linear contribution. Three nodes model the linear damping force $c\dot{x}$ using Prototype 3's antisymmetric construction.

The complete architecture, illustrated in Figure 7.1, consists of seven hidden layer nodes with two inputs (x, $\dot{x}$) and one output ($\hat{f}_R$). Initialization values are provided in Table 7.1. This initialization is *quasi-deterministic*. Hence, while incorporating physical insight systematically, some engineering judgment remains in exact parameter values and scaling choices.

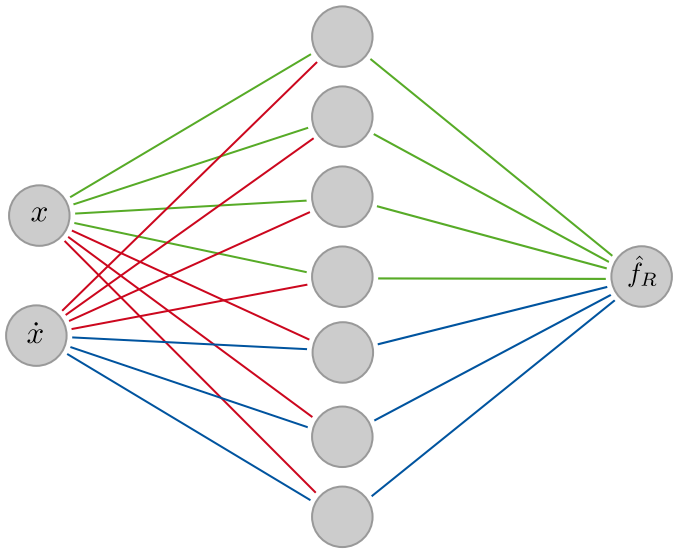

Figure 7.1: Architecture of the IML-initialized neural network. Green nodes connections: spring stiffness component; blue connections: damping component; red connections: zero-valued weights completing the fully connected structure.

Table 7.1: IML initialization parameters for Duffing oscillator.

Parameter	**Values**
Input layer weights	$\begin{bmatrix} 5 & 5 & 1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & -1 \end{bmatrix}^\top$
Input layer biases	$[5, -5, 0, 0, 0, 0, 0]^\top$
Output layer weights	$[1, 1, -1, -1, 1, -0.5, -0.5]^\top$

The IML-initialized network is integrated into the adaptive data generation framework and compared against the model pool-based approach from Chapter 5. The setup for the numerical study is identical to Chapter 5. The sampling space is normalized, a large

initial mesh size for pattern search, and the same Duffing oscillator and PS parameters used in Section 5.1.

Figure 7.2 presents the error surface for the IML-initialized network. The transformation from random initialization is significant. The landscape exhibits smooth, convex structure with a clearly defined global minimum, characteristics that facilitate efficient optimization compared to the noisy, multi-modal surfaces typical of randomly initialized networks, where local minima and flat regions impede convergence. The error surface shown in Figure 7.2 closely resembles the one obtained with the model pool, at a tenth of the computational cost and the added benefit of the interpretability and quasi-deterministic initialization, which is significant for structural dynamics applications.

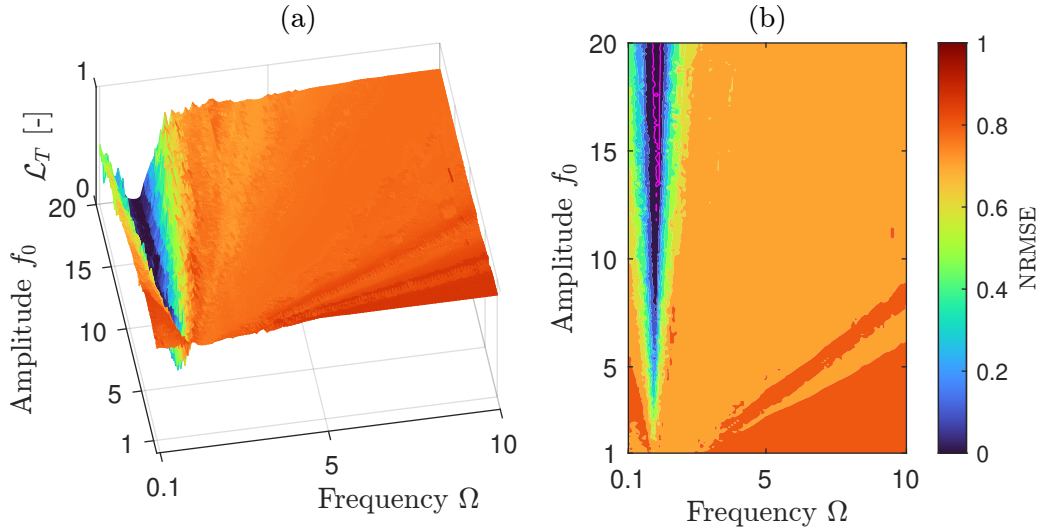

Figure 7.2: (a) Error surface for IML-initialized neural network. (b) Contour plot of surface for IML-initialized neural network. Magenta contours indicate regions satisfying the accuracy requirement ($\epsilon < 0.01$).

Furthermore, Fig. 7.3 demonstrates the central finding. The single IML-initialized network achieves superior performance to the ten-member model pool while requiring only 10% of the computational cost. The IML-enabled approach achieves 95% success rate within 12 experiments, compared to 90% for the baseline framework. Median convergence occurs after 8 experiments versus 10 for model pool. This computational advantage compounds across optimization iterations. For the Duffing oscillator case requiring approximately 10 optimization iterations, IML reduces total computational cost by tenfold while maintaining or exceeding ensemble accuracy. Both adaptive approaches significantly outperform space-filling methods (LHS, Sobol), which require substantially more experiments to achieve comparable accuracy, but IML achieves this efficiency without ensemble overhead.

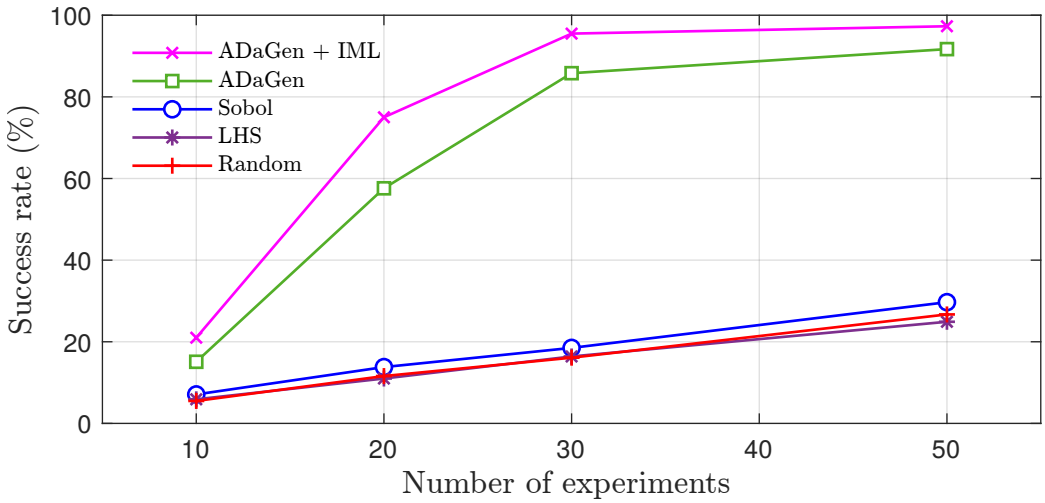

Figure 7.3: Comparison in performance of ADaGen with IML-initialization, ADaGen with model pool selection, and unsupervised methods. The x-axis shows the number of experiments required for the success rate shown on y-axis.

7.2.4 Discussion

The integration of IML with adaptive data generation provides several significant advantages. Firstly, replacing random initialization with physically informed parameters, IML eliminates the primary source of training variance. Single-network optimization becomes reliable without ensembles or model pools, enabling deterministic performance evaluation during dataset comparison. Secondly, the tenfold reduction in per-iteration cost directly translates to faster optimization, making the framework practical for real-time or resource-constrained applications. For systems requiring multiple optimization iterations, this advantage compounds significantly.

Network parameters retain physical meaning throughout training, facilitating model diagnosis and providing engineering insight into system behavior. When training fails or produces unexpected results, interpretable parameters enable targeted troubleshooting impossible with black-box approaches.

The three-prototype framework itself resulted from systematic analysis of common engineering nonlinearities; extension to additional classes follows the same principled approach. The initialization remains quasi-deterministic rather than fully deterministic, requiring engineering judgment for parameter selection. However, this moderate subjectivity is vastly preferable to the complete arbitrariness of random initialization, and the resulting variance reduction justifies the additional effort.

Integration with Broader Framework IML initialization has the potential to enhance all subsequent extensions presented in this chapter. Cumulative data generation (Section 7.3) benefits from variance reduction when evaluating multiple training datasets. MDOF extensions (Section 7.4) gain computational efficiency critical for higher-

dimensional optimization. Reference-free sampling (Section 7.5) requires particularly reliable performance evaluation, making variance elimination essential.

The approach represents a paradigm shift from treating neural networks as black-box universal approximators to viewing them as structured models encoding physical knowledge. This shift aligns machine learning with engineering practice, where prior knowledge guides model development rather than being discarded in favor of purely data-driven approaches. For adaptive data generation, this alignment proves essential since optimization requires reliable performance gradients that random initialization cannot consistently provide.

However, the IML approach has limitations. It requires prior knowledge of the functional form of system nonlinearities, knowledge that may not always be available. The initialization is quasi-deterministic rather than fully deterministic, requiring some engineering judgment. Finally, while effective for the classes of nonlinearities studied by Pei et al., extension to more complex nonlinear behaviors may require developing new prototypes. Despite these limitations, IML represents a promising direction for enhancing adaptive data generation, particularly for well-characterized nonlinearities common in structural dynamics applications.

7.3 Cumulative Data Generation

7.3.1 Problem Formulation

The baseline ADaGen framework presented in Chapter 4 operates under the assumption that a single experiment with optimized parameters $\boldsymbol{\psi}_{opt}$ generates a training dataset containing sufficient information to model the system across its operational range. This assumption can be violated in several practical scenarios. Firstly, some setups can reliably capture only steady-state behavior and provide insufficient amplitude diversity for comprehensive model training. Furthermore, MDOF structures with multiple significant natural frequencies require targeted excitation at each mode for adequate dynamic characterization. Large-amplitude excitations may be prohibited in certain frequency ranges due to structural integrity concerns, necessitating multiple moderate-amplitude experiments. Finally, physical devices may have distinct operating modes (e.g., TLCDs with different liquid heights) that cannot be simultaneously sampled. To address these limitations, this section introduces *cumulative data generation*, a strategy for strategically combining multiple training datasets to achieve comprehensive system coverage.

To better illustrate this, the Duffing oscillator from Section 5.1 is considered. Figure 7.4 compares two training datasets obtained with identical excitation parameters $\boldsymbol{\psi} = [f_0,\Omega] = [15,1]$:

- *Dataset* $\mathcal{D}_a$ includes the complete system response including both transient and steady-state responses. The dataset spans amplitudes from zero to maximum transient values before settling, providing excellent coverage of the force-state space.

- *Dataset* $\mathcal{D}_b$ contains only the steady-state periodic response, sampling the force-state space at a single amplitude. Despite containing the same number of data points as $\mathcal{D}_a$, this dataset provides dramatically less information due to limited amplitude diversity.

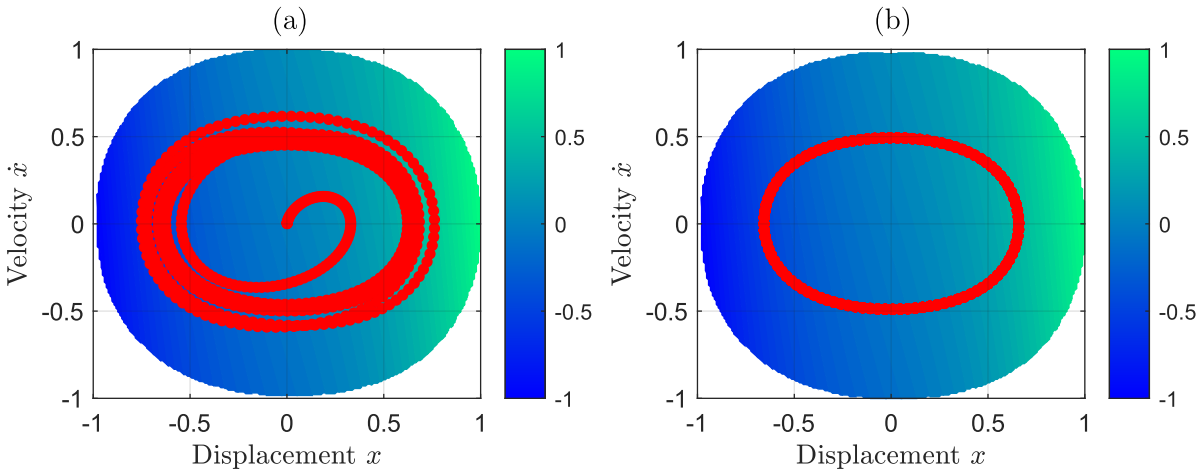

Figure 7.4: Training data visualization in force-state space. Blue-green surface: target restoring force model. Red markers: training data points. (a) Dataset $\mathcal{D}_a$ with transient and steady-state response provides comprehensive amplitude coverage. (b) Dataset $\mathcal{D}_b$ with steady-state response only samples a single amplitude, yielding insufficient information for model training despite identical data point count.

Dataset $\mathcal{D}_a$ provides a unique, well-conditioned mapping from states $(x,\dot{x})$ to forces f_R, enabling successful neural network training with low variance error coming from the dataset itself. In contrast, dataset $\mathcal{D}_b$ yields a non-unique, ill-conditioned mapping that produces high variance error regardless of neural network capacity or training procedure. This example motivates the need for cumulative approaches when single-experiment data sufficiency cannot be guaranteed.

7.3.2 Cumulative Training Strategy

To address the issue discussed in the problem statement, a cumulative training strategy is introduced. The training dataset is denoted with $\mathcal{D}_i(\boldsymbol{\psi}_i)$ and generated with experiment parameters $\boldsymbol{\psi}_i$. A *cumulative training dataset* is formed by combining N_{exp} individual datasets:

$$\mathcal{D}_{\text{cum}} = \bigcup_{i=1}^{N_{\text{exp}}} \mathcal{D}_i(\boldsymbol{\psi}_i), \tag{7.2}$$

where each $\boldsymbol{\psi}_i$ is unique and strategically selected to provide complementary information.

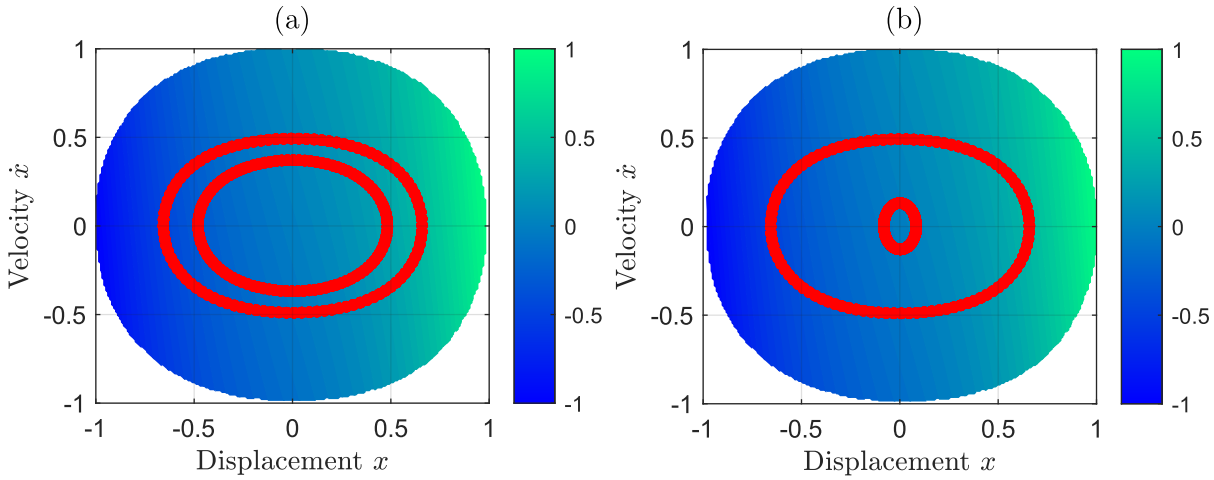

Figure 7.5: Effect of parameter variation in cumulative datasets (steady-state data only). Red markers show combined training data from two experiments with different parameter combinations: (a) $\boldsymbol{\psi}_1 = [15,1]$ and $\boldsymbol{\psi}_2 = [10,1]$; and (b) $\boldsymbol{\psi}_1 = [15,1]$ and $\boldsymbol{\psi}_2 = [10,2]$.

The adaptive data generation objective is then modified to find the optimal set of experiment parameters:

$$\{\boldsymbol{\psi}_{1,opt}, \ldots, \boldsymbol{\psi}_{N_{\text{exp}},opt}\} = \underset{\{\boldsymbol{\psi}_1,\ldots,\boldsymbol{\psi}_{N_{\text{exp}}}\}}{\arg\min} \mathcal{L}_T(\mathcal{M}(\mathcal{D}_{\text{cum}})), \tag{7.3}$$

subject to:

$$\|\boldsymbol{\psi}_i - \boldsymbol{\psi}_j\| \geq d_{\min} \quad \forall i \neq j, \tag{7.4}$$

where $d_{\min}$ ensures sufficient parameter separation and $\mathcal{L}_T$ denotes the test error. The diversity constraint prevents selecting redundant nearby points that overfit to specific dynamics rather than providing complementary information.

Figure 7.5 illustrates how parameter selection affects information content for two-experiment cumulative datasets with steady-state data only. When both experiments vary only amplitude at fixed frequency, $\boldsymbol{\psi}_1 = [15,1]$ and $\boldsymbol{\psi}_2 = [10,1]$ (space is normalized to the restoring force), the resulting dataset provides excellent amplitude diversity but limited frequency content. When both amplitude and frequency vary, $\boldsymbol{\psi}_1 = [15,1]$ and $\boldsymbol{\psi}_2 = [10,2]$, the cumulative dataset achieves balanced diversity across both dimensions and spans the force-state space more comprehensively. This reveals that strategic variation of both excitation parameters is necessary for optimal cumulative dataset formation.

7.3.3 Numerical Validation with SDOF System

The cumulative training strategy is evaluated quantitatively using the Duffing oscillator under the realistic constraint that only steady-state response data can be measured. This is a common limitation in experimental setups that reach equilibrium quickly or where transient behavior cannot be reliably captured. To intensify the challenge, the parameter

domain is restricted to $f_0 \in [1,15]$ and $\Omega \in [0.1,4]$, while models are evaluated on a test signal with amplitude $f_0 = 20$ that extends beyond the training range. The parameters of the Duffing oscillator, test signal and the setup are the same as introduced in Chapter 5, Section 5.1.

Figure 7.6 compares error surfaces for two training data scenarios. When both transient and steady-state data are available, as shown in Fig. 7.6a, the error surface exhibits a clear global minimum structure with minimum error $\mathcal{L}_T^{\min} = 0.0026$ at $\boldsymbol{\psi} = [14.5,1.2]$. Multiple parameter combinations achieve the tolerance threshold $\mathcal{L}_T < 0.01$, indicated by magenta contours. In contrast, when only steady-state data can be used (Fig. 7.6b), the minimum achievable error increases to $\mathcal{L}_T^{\min} = 0.0203$ at $\boldsymbol{\psi} = [13.5,1.2]$, and no parameter combination meets the tolerance threshold. The error surface topology resembles the transient-inclusive case, but overall error margins are approximately eight times higher. This confirms that steady-state data alone provides insufficient information for accurate modeling, motivating the cumulative approach.

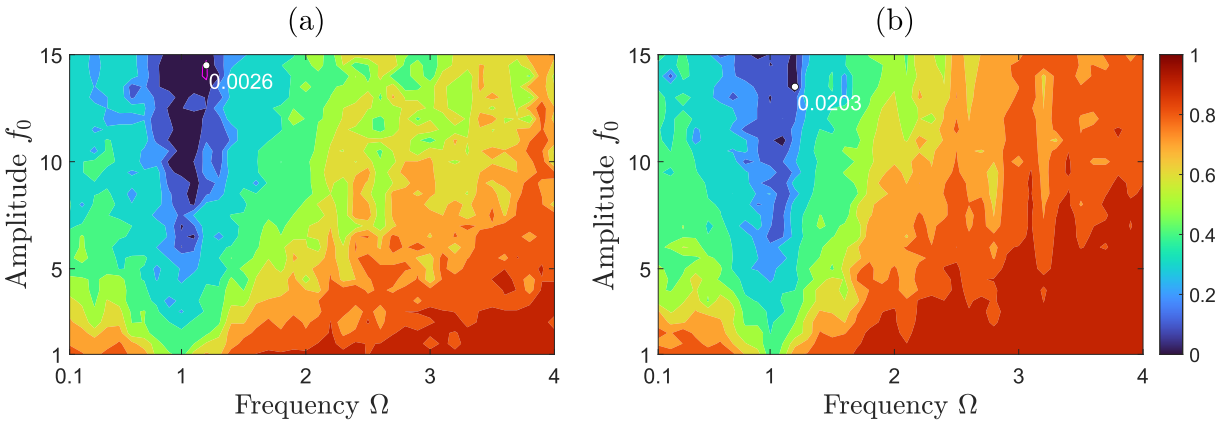

Figure 7.6: Error surfaces for single-dataset training in restricted parameter domain. (a) Transient and steady-state data: multiple parameter combinations meet accuracy requirements (magenta contours indicate $\mathcal{L}_T < 0.01$). (b) Steady-state data only: no parameter combination achieves tolerance threshold.

7.3.3.1 Cumulative Training Results

To evaluate cumulative training viability, two-experiment datasets are constructed by systematically combining sampling points. For each initial point $\boldsymbol{\psi}_i$, cumulative datasets $\mathcal{D}_{\text{cum}} = \mathcal{D}_i(\boldsymbol{\psi}_i) \cup \mathcal{D}_k(\boldsymbol{\psi}_k)$ are formed by pairing with all other points k in the sampling space. A feedforward neural network is trained on each cumulative dataset and evaluated on the test signal, producing a cumulative error surface that shows the performance when combining the fixed initial point with datasets from all possible second points.

Six representative initial points spanning the range of single-dataset information content are selected, from point (a) near the global minimum with lowest single-dataset error to point (c) near maximum error, as shown in Table 7.2.

Table 7.2: Selected initial points for cumulative training analysis.

Point	**Amplitude f_0**	**Frequency Ω [Hz]**	**Single-dataset error $\mathcal{L}_T$**
a	13.5	1.2	0.0203
b	10.0	2.0	0.6796
c	2.0	3.5	0.9632
d	15.0	0.1	0.4449
e	4.0	1.2	0.3994
f	13.5	3.7	0.8428

Figure 7.7 presents the cumulative error surfaces for all six initial points. The results demonstrate that cumulative training fundamentally transforms the optimization landscape. All cumulative strategies achieve errors below the $\epsilon = 0.01$ threshold that no single-dataset configuration could meet. Initial point (a), despite having the best single-dataset performance, achieves minimum cumulative error $\mathcal{L}_T^{\min} = 0.0040$ when combined with $\boldsymbol{\psi}_2 = [15.0,1.2]$. While slightly higher than the best transient-data result of 0.0026 from Figure 7.6(a), the cumulative approach provides a vastly larger region of acceptable parameter combinations, offering robustness against parameter selection uncertainty.

Several critical patterns emerge from these surfaces. Each cumulative error surface exhibits a characteristic floor at approximately the error level of the initial point, reflecting an information preservation property: $\mathcal{L}_T(\mathcal{D}_i \cup \mathcal{D}_k) \leq \max(\mathcal{L}_T(\mathcal{D}_i), \mathcal{L}_T(\mathcal{D}_k))$, with equality when $\mathcal{D}_k$ provides no new information. Points near the initial point in parameter space sometimes produce slightly higher cumulative errors than the initial point alone, visible as small error increases near $\boldsymbol{\psi}_i$ in Figure 7.7(a). This occurs when adding redundant data at similar amplitudes and frequencies causes overfitting rather than improved generalization. Optimal second points provide complementary information from different parameter space regions. Even initial points with poor single-dataset performance, such as points (c) and (f) with $\mathcal{L}_T > 0.84$, achieve acceptable cumulative performance when combined appropriately, demonstrating robustness of the cumulative strategy.

All initial sampling points achieve acceptable accuracy when appropriately combined with complementary datasets, demonstrating robustness of the cumulative training strategy. Even when optimization initializes from configurations yielding poor single-dataset performance, strategic combination with well-selected second experiments consistently produces models meeting accuracy requirements. This robustness proves particularly valuable in practical applications where prior knowledge of optimal sampling locations may be limited or where initial experiments must be conducted conservatively due to safety constraints.

Table 7.3 summarizes the optimal second point for each initial condition. Most optimal cumulative datasets combine experiments near the system's natural frequency ($\Omega \approx 1.2$ Hz) with high amplitudes, regions that excite significant dynamic response and provide rich force-state space coverage.

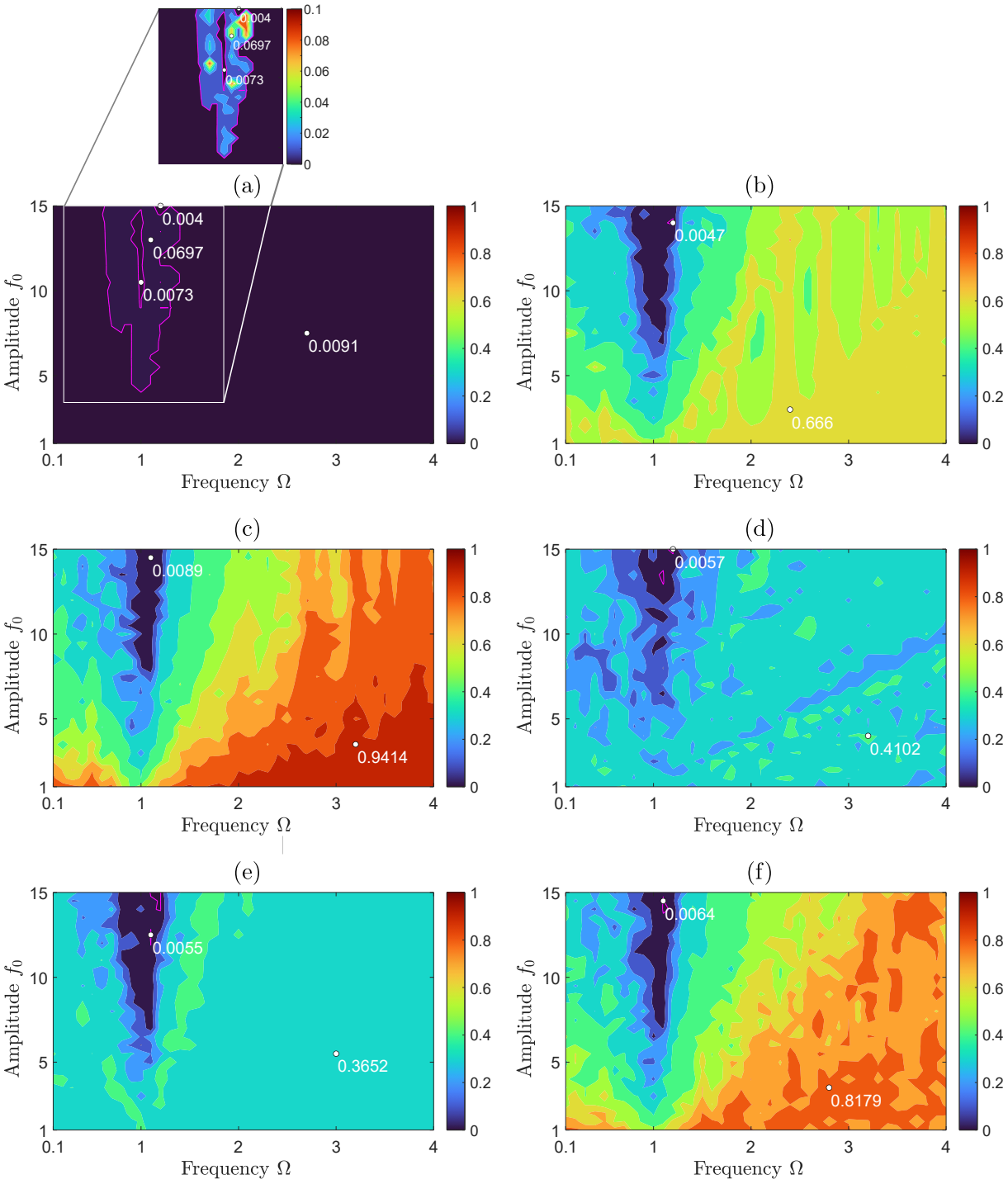

Figure 7.7: Cumulative error surfaces for six initial sampling points, as shown in Table 7.2. Each surface shows test error when combining the initial point dataset with datasets from all other parameter combinations. Magenta contours indicate $\mathcal{L}_T < 0.01$ threshold. All cumulative strategies achieve acceptable accuracy, as shown in Table 7.3.

Table 7.3: Optimal cumulative dataset configurations.

Initial point	Minimum error $\mathcal{L}_T^{\min}$	Optimal second point f_0	Ω [Hz]
a	0.0040	15.0	1.2
b	0.0047	14.0	1.2
c	0.0089	14.5	1.1
d	0.0057	15.0	1.2
e	0.0055	12.5	1.1
f	0.0064	14.5	1.1

7.3.4 Integration with Adaptive Sampling Framework

The cumulative training strategy integrates naturally into the adaptive data generation framework through sequential optimization. Algorithm 3 formalizes the procedure. The first optimization identifies the globally optimal single-dataset configuration using the standard framework from Chapter 4. Subsequent optimizations then identify complementary datasets from different parameter space regions, with each new dataset selected to minimize test error when combined with all previously collected data.

Algorithm 3 Cumulative Adaptive Data Generation

Input: Initial sampling strategy, maximum experiments $N_{\max}$, tolerance ϵ
Output: Optimal experiment parameters $\{\boldsymbol{\psi}_{1,opt}, \ldots, \boldsymbol{\psi}_{N_{opt},opt}\}$
Initialize: $\mathcal{D}_{\text{cum}} \leftarrow \emptyset$, $n \leftarrow 0$
while $n < N_{\max}$ **and** $\mathcal{L}_T(\mathcal{M}(\mathcal{D}_{\text{cum}})) > \epsilon$ **do**
    $n \leftarrow n + 1$
    if $n = 1$ **then**
        Find $\boldsymbol{\psi}_{1,opt}$ using standard AdaGen framework (cf. Chapter 4)
        $\mathcal{D}_{\text{cum}} \leftarrow \mathcal{D}_1(\boldsymbol{\psi}_{1,opt})$
    else
        Construct objective: $f(\boldsymbol{\psi}) = \mathcal{L}_T(\mathcal{M}(\mathcal{D}_{\text{cum}} \cup \mathcal{D}_n(\boldsymbol{\psi})))$
        Find $\boldsymbol{\psi}_{n,opt}$ using pattern search with constraint $\|\boldsymbol{\psi}_n - \boldsymbol{\psi}_i\| \geq d_{\min}$ for $i < n$
        $\mathcal{D}_{\text{cum}} \leftarrow \mathcal{D}_{\text{cum}} \cup \mathcal{D}_n(\boldsymbol{\psi}_{n,opt})$
    end if
    Train model $\mathcal{M}(\mathcal{D}_{\text{cum}})$ and evaluate $\mathcal{L}_T$
end while
return $\{\boldsymbol{\psi}_{1,opt}, \ldots, \boldsymbol{\psi}_{n,opt}\}$

The hierarchical structure enables rapid convergence in practice. The first optimization typically identifies the globally optimal single-dataset configuration, establishing a strong baseline. Subsequent optimizations identify complementary datasets that fill information gaps. In many cases, the second dataset can be selected directly from points already evaluated during the first optimization, eliminating additional experiments. The diversity

constraint $\|\boldsymbol{\psi}_n - \boldsymbol{\psi}_i\| \geq d_{\min}$ proves critical for preventing the selection of redundant nearby points that provide minimal additional information.

The cumulative strategy transforms the adaptive framework from beneficial to essential for scenarios where single-experiment data sufficiency cannot be guaranteed. The steady-state constraint example demonstrates this transformation. Whereas no single experiment achieved acceptable accuracy, strategic two-experiment combinations consistently succeeded. This principle extends naturally to more complex scenarios, including MDOF systems where multiple natural frequencies demand targeted excitation, as explored in the following section.

7.4 Extension to Multi-Degree-of-Freedom Systems

The cumulative data generation strategy introduced in Section 7.3 proves not merely beneficial but essential for MDOF systems. Unlike SDOF systems where single experiments could provide sufficient information under favorable conditions, MDOF structures with multiple natural frequencies and coupled dynamics fundamentally preclude comprehensive characterization through isolated experiments. This section extends the adaptive framework to MDOF systems and demonstrates that the cumulative approach transitions from an optional enhancement to a necessary requirement.

In order to extend the approach to MDOF systems, the two-degree-of-freedom shear building structure shown in Fig. 7.8 is considered. The system comprises two masses connected by springs and dampers, with equations of motion given by:

$$m_1\ddot{x}_1 = F_1 - k_1x_1 - k_2(x_1 - x_2) - c_1\dot{x}_1 - c_2(\dot{x}_1 - \dot{x}_2), \tag{7.5}$$

$$m_2\ddot{x}_2 = F_2 + k_2(x_1 - x_2) + c_2(\dot{x}_1 - \dot{x}_2), \tag{7.6}$$

where subscripts 1 and 2 denote the first and second floors, respectively. The system parameters yield natural frequencies $f_1 = 0.97$ Hz and $f_2 = 4.03$ Hz. Figure 7.8 also shows multiple possible loading configurations such as ground acceleration $\ddot{u}_g$ affecting both masses proportionally, or direct forces $F_1(t)$ and $F_2(t)$ applied to specific floors. This simple system already exhibits the key challenge of MDOF identification. Here, adequate excitation of multiple well-separated natural frequencies requires targeted experiments across the frequency spectrum.

The increased complexity relative to SDOF systems manifests in three critical ways. First, restoring forces depend on states from both degrees of freedom, requiring neural network architectures that can capture coupled dynamics. Second, multiple natural frequencies must be adequately excited within training data to enable accurate prediction across the operational range. Third, loading location significantly affects system observability, with some excitation configurations providing richer information than others. These factors necessitate careful consideration of both neural network architecture and experimental design strategy.

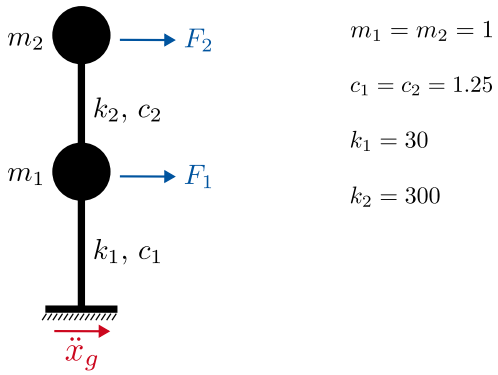

Figure 7.8: Two-degree-of-freedom system. Loading options include ground excitation $\ddot{x}_g$ or floor-level forces $F_1(t)$ and $F_2(t)$. System natural frequencies are $f_1 = 0.97$ Hz and $f_2 = 4.03$ Hz.

The investigation proceeds through two complementary case studies. The first employs FNNs with direct restoring force measurements, establishing baseline performance under idealized measurement conditions. The second uses Neural ODEs to learn system dynamics from displacement and velocity time series alone, representing the more realistic scenario where internal forces are not directly measurable. Together, these studies demonstrate the necessity of cumulative approaches for MDOF system identification.

7.4.1 Case Study 1: 2-DOF System with FNN

The first case study considers an idealized scenario where restoring forces for both degrees of freedom are directly measurable. While this configuration is typically unattainable in practice for continuous structures modeled as MDOF systems, it serves to establish baseline performance and isolate the effects of multi-modal dynamics from measurement constraints. A FNN with two hidden layers of two neurons each maps the complete state vector to restoring forces:

$$[f_{R,1}, f_{R,2}]^\top = f_\theta([x_1, \dot{x}_1, x_2, \dot{x}_2]^\top). \tag{7.7}$$

Training datasets $\mathcal{D} = \{x_1, \dot{x}_1, f_{R,1}, x_2, \dot{x}_2, f_{R,2}\}$ are generated across parameter ranges $f_0 \in [1,20]$ in increments of 0.2 and $\Omega \in [0.1,10]$ Hz in increments of 0.1, spanning both natural frequencies. The test dataset employs a 50-second sine sweep from 0.5 to 5 Hz with amplitude $f_0 = 20$, capturing system response across the first natural frequency and into the transition region toward the second mode. Model selection from ten parallel training runs with different random initializations identifies the best performer based on training convergence, and test error quantifies training dataset quality through NRMSE averaged across both degrees of freedom.

Figure 7.9 presents error surfaces for two distinct excitation configurations. The first configuration, shown in Fig. 7.9a, applies ground acceleration, which is equivalent to forces $F_1 = -m_1\ddot{x}_g$ and $F_2 = -m_2\ddot{x}_g$ proportional to each mass. The second configuration

excites only the top floor with $F_2 \neq 0$ and $F_1 = 0$, representing a realistic experimental scenario where access to individual DOFs may be limited.

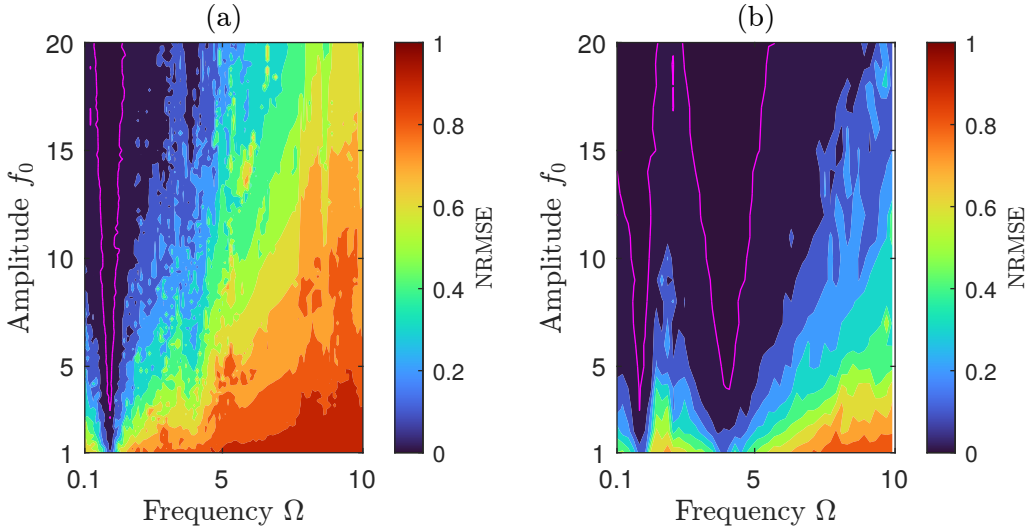

Figure 7.9: Error surfaces for the 2-DOF system identified using FNNs with complete restoring force measurements under (a) ground excitation configuration and (b) Top-floor excitation. Magenta contours indicate regions satisfying the accuracy requirement $\mathcal{L}_T < 0.01$. Both surfaces exhibit pronounced minima near system natural frequencies at $f_1 = 0.97$ Hz and $f_2 = 4.03$ Hz, with top-floor excitation producing more distinct low-error bands aligned with these resonances.

The error surface topology reveals several critical characteristics of MDOF system identification. Both excitation configurations produce pronounced minima near the system natural frequencies, particularly evident in the top-floor excitation case where distinct low-error bands appear at $\Omega \approx 1$ Hz and $\Omega \approx 4$ Hz, as can be seen from Fig. 7.9b. Ground excitation yields a more diffuse minimum region, as shown in Figure 7.9a, suggesting that simultaneous excitation of both masses provides broader frequency content but less targeted modal information. The top-floor excitation produces sharper gradients in the error surface, facilitating more efficient optimization through pattern search algorithms. Most importantly, both surfaces demonstrate that single experiments conducted away from natural frequencies produce models with high test errors.

These results confirm that the ADaGen framework extends naturally to MDOF systems when complete force measurements are available. The error surface convexity near minima facilitates efficient convergence using the ADaGen framework established for SDOF systems. However, this idealized scenario with direct measurement of all internal forces is generally unrealistic for operational structures. Continuous structures modeled through spatial discretization as MDOF systems rarely allow direct measurement of restoring forces at each degree of freedom. This limitation motivates the second case study employing Neural ODEs, which learn system dynamics directly from observable displacement and velocity.

7.4.2 Case Study 2: Neural ODEs for MDOF Systems

When restoring force measurements are unavailable, Neural ODEs provide an alternative approach by learning system dynamics directly from displacement and velocity time series, as demonstrated in Chapter 6. This represents the operationally realistic scenario where only states are measured through accelerometers or displacement transducers, while internal forces remain inaccessible. The Neural ODE architecture learns to predict accelerations given current states, with the known external forcing term added explicitly. This formulation respects the physical structure of the problem while allowing the network to discover the coupled restoring force relationships through training, as described in detail in Section 6.4.

Cumulative training with Neural ODEs requires careful handling of temporal constraints due to the time-depth parameter N_t, which defines the prediction horizon during training. When combining multiple datasets $\mathcal{D}_i$ and $\mathcal{D}_k$ collected from different experiments, initial conditions for training trajectories must satisfy temporal constraints to ensure sufficient future time steps for computing multi-step prediction losses. Figure 7.10 illustrates this constraint structure. The datasets are concatenated temporally, with Dataset 1, shown in blue, exhibiting larger amplitude oscillations at lower frequency and Dataset 2 showing smaller amplitude response at higher frequency. Gray shaded regions mark invalid starting points where initial time t_0 cannot be selected, specifically the final N_t points of each dataset. This constraint arises because at least N_t future time steps must be available from any starting point to compute trajectory-based loss functions over the specified prediction horizon. The green trajectory in Dataset 1 demonstrates a valid training example: starting at $t_0^{(i)}$ and extending N_t steps into the future, ensuring sufficient data for multi-step prediction during training iteration i.

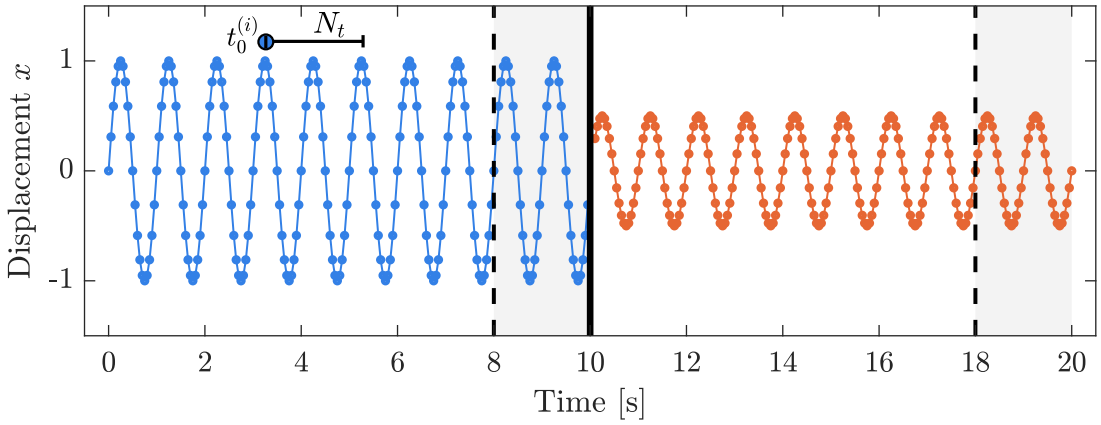

Figure 7.10: Temporal constraints for cumulative Neural ODE training. Two datasets collected with different experimental parameters are concatenated. The time-depth parameter N_t restricts admissible initial conditions for training trajectories. Gray shaded regions at the end of each dataset indicate points where t_0 cannot be selected due to insufficient future time steps. Transition regions between datasets require special handling to maintain temporal continuity.

As illustrated in Figure 7.10, admissible initial time points t_0 are constrained to the set

$$t_0 \in \{t_1, t_2, \ldots, t_{N-N_t}\}, \tag{7.8}$$

where N denotes the total number of time steps in the dataset. This constraint ensures that at least N_t future time steps remain available from any starting point, enabling computation of trajectory-based loss functions over the specified prediction horizon. Transition regions between datasets require special consideration to maintain temporal continuity, as discontinuities in the combined trajectory can adversely affect training stability. In practice, the training algorithm samples initial conditions uniformly from admissible regions while respecting these temporal boundaries.

7.4.2.1 Results and Analysis

Figure 7.11 presents the single-dataset and cumulative error surfaces for four representative initial sampling points. To reduce computational cost while maintaining experimental realism, the investigation evaluates a restricted parameter domain with $f_0 \in [1,4]$ in increments of 0.25 and $\Omega \in [0.5,5]$ Hz in increments of 0.1. The single-dataset error surface in Fig. 7.11a reveals that only few single-experiment configurations within the restricted parameter range achieve the accuracy tolerance of $\mathcal{L}_T < 0.01$. The minimum error obtained from any single experiment is $\mathcal{L}_T^{\min} = 0.0098$, which is directly at the required threshold.

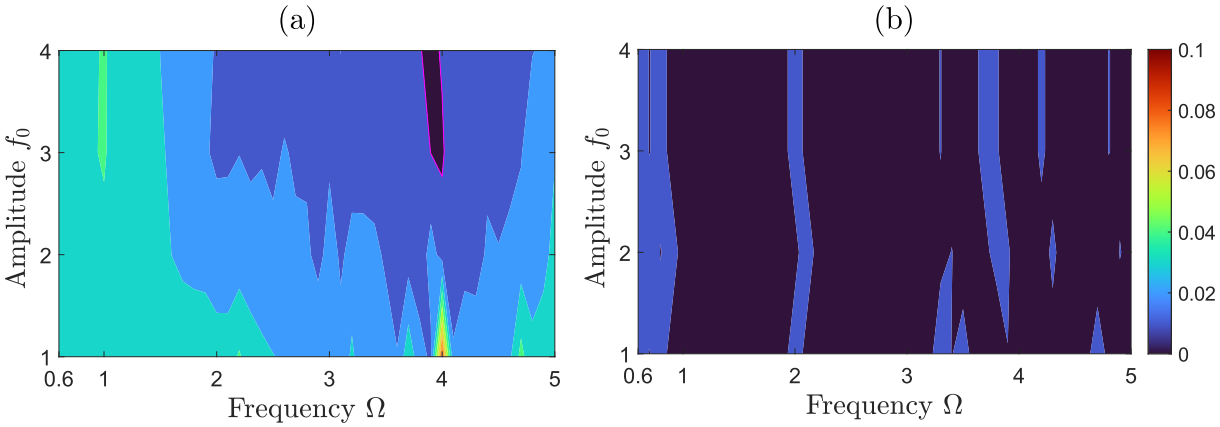

Figure 7.11: Error surfaces for Neural ODE identification of the 2-DOF system with top-floor excitation. Panel (a) shows the single-dataset error surface where no parameter combination satisfies $\mathcal{L}_T < 0.01$, demonstrating the inadequacy of single-experiment approaches.

The error surfaces exhibit a pronounced frequency-driven structure. Steep gradients along the frequency axis contrast with relatively gradual variations along the amplitude axis, indicating that excitation frequency exerts stronger influence on cumulative performance than excitation amplitude. This frequency dominance reflects the resonance structure inherent to linear systems, where excitation near natural frequencies generates the most informative dynamic responses. The relatively weak amplitude dependence suggests that

for the linear system examined here, moderate amplitude variations provide sufficient nonlinearity in the force-state relationship to enable accurate identification, whereas frequency content determines which modes are adequately excited.

The single-dataset error surface reveals that one experiment configuration approaches but does not quite achieve the accuracy tolerance, with minimum error $\mathcal{L}_T^{\min} = 0.0098$. This near-success likely results from the system's linear dynamics combined with training data that includes transient response, which provides sufficient amplitude diversity for Neural ODE training even from a single experiment. However, the acceptable parameter region remains extremely limited. The cumulative error surface in Fig. 7.11b demonstrates that strategic combination of two experiments consistently achieves acceptable accuracy. Each panel shows the error obtained when combining a fixed initial sampling point with datasets from all other parameter combinations across the sampling space. The magenta contours delineating regions below the tolerance threshold reveal that cumulative training dramatically expands the acceptable parameter space. The best single-dataset configurations, when combined with complementary second experiments, achieve even lower errors than observed in the single-experiment case, demonstrating that cumulative approaches provide both robustness through expanded acceptable regions and superior peak performance through strategic experiment combination.

The pronounced frequency-driven structure reflects the linear system's resonance characteristics. Excitation frequency dominates performance because it determines which modes are adequately excited, while amplitude variations play a secondary role for this linear system. This frequency dominance would likely shift for nonlinear systems where amplitude-dependent behaviors introduce additional complexity requiring balanced variation of both parameters. The multi-frequency complementarity principle ensures that cumulative datasets capture system response characteristics across multiple modes, even when complete modal information cannot be obtained from individual experiments. This transition from SDOF to MDOF fundamentally alters the problem structure. While SDOF systems permit comprehensive characterization through carefully designed single experiments, MDOF systems with well-separated natural frequencies may require multiple targeted experiments to capture complete modal behavior.

7.5 Reference-Free Adaptive Sampling

The most restrictive assumption underlying the baseline framework is the availability of a comprehensive test dataset $\mathcal{D}_{\mathrm{T}}$ that spans the system's operational range and provides objective evaluation of training data quality. This assumption is violated in several realistic scenarios where developing such test data is impractical or impossible. During initial system characterization, the appropriate operational range may be unknown, making it difficult to define what constitutes comprehensive coverage. For prototype devices or untested structures, certain operating conditions may be unsafe to test directly, yet modeling these regimes may be essential for safety analysis. When exploring extreme or rare loading scenarios such as earthquake response or impact dynamics, generating representative test data may be prohibitively expensive or simply infeasible. Developing methods that can operate without comprehensive test datasets extends the framework's applicability to these challenging but practically relevant scenarios, enabling adaptive sampling in exploratory contexts.

The fundamental challenge in reference-free adaptive sampling is distinguishing informative training datasets from uninformative ones without ground truth. The key insight enabling progress is that relative information content between training datasets can be assessed through cross-validation, even when absolute quality cannot be determined. To illustrate this, two candidate training datasets $\mathcal{D}_1(\boldsymbol{\psi}_1)$ and $\mathcal{D}_2(\boldsymbol{\psi}_2)$ generated with different experimental parameters are considered. By training a model $\mathcal{M}_1$ on $\mathcal{D}_1$ and evaluating it on $\mathcal{D}_2$ to obtain error $\mathcal{L}_{1\to2}$, then training model $\mathcal{M}_2$ on $\mathcal{D}_2$ and evaluating on $\mathcal{D}_1$ to obtain error $\mathcal{L}_{2\to1}$, the asymmetry in cross-validation errors reveals relative information content. If $\mathcal{L}_{2\to1} > \mathcal{L}_{1\to2}$, this indicates that $\mathcal{D}_1$ contains more information than $\mathcal{D}_2$, as the model trained on $\mathcal{D}_1$ generalizes better to $\mathcal{D}_2$ than the reverse. This asymmetry principle forms the foundation for reference-free optimization. Hence, by iteratively seeking training datasets that generalize well to all other evaluated datasets, the framework can identify informative parameter configurations without requiring ground truth validation.

Algorithm 4 formalizes the reference-free adaptive sampling procedure using cross-validation as the optimization objective. The algorithm begins by sampling at least two diverse initial points in the parameter space, typically using space-filling designs such as LHS to ensure adequate coverage. For each sampling point $\boldsymbol{\psi}_i$, a training dataset $\mathcal{D}_i$ is generated and a model $\mathcal{M}_i$ is trained. Cross-validation errors $\mathcal{L}_{i\to j}$ are computed for all pairs, and the point yielding the model with superior generalization performance is selected as the starting point for pattern search optimization. The pattern search then proceeds by evaluating poll points around the current best solution, generating new training datasets, training models, and computing cross-validation errors. A poll point becomes the new mesh center if its trained model generalizes better to the current best dataset than the current best model generalizes to the poll dataset. The algorithm terminates when the pattern search mesh size falls below a minimum threshold or the experimental budget is exhausted.

Algorithm 4 Reference-Free Adaptive Sampling using Cross-Validation

Input: Minimum mesh size $\Delta_{\min}$, maximum experiments $N_{\max}$
Output: Optimal experiment parameters $\boldsymbol{\psi}_{opt}$, trained model $\mathcal{M}_{opt}$
Sample at least two diverse initial points: $\boldsymbol{\psi}_1$, $\boldsymbol{\psi}_2$
Collect datasets: $\mathcal{D}_1 \leftarrow \mathcal{D}(\boldsymbol{\psi}_1)$, $\mathcal{D}_2 \leftarrow \mathcal{D}(\boldsymbol{\psi}_2)$
Train models: $\mathcal{M}_1 \leftarrow \text{Train}(\mathcal{D}_1)$, $\mathcal{M}_2 \leftarrow \text{Train}(\mathcal{D}_2)$
Compute cross-validation: $\mathcal{L}_{1\to2} \leftarrow \text{Eval}(\mathcal{M}_1,\mathcal{D}_2)$, $\mathcal{L}_{2\to1} \leftarrow \text{Eval}(\mathcal{M}_2,\mathcal{D}_1)$
if $\mathcal{L}_{2\to1} > \mathcal{L}_{1\to2}$ **then**
    $\boldsymbol{\psi}_{\text{best}} \leftarrow \boldsymbol{\psi}_1$ ▷ Dataset 1 more informative
else
    $\boldsymbol{\psi}_{\text{best}} \leftarrow \boldsymbol{\psi}_2$
end if

Initialize pattern search at $\boldsymbol{\psi}_{\text{best}}$ with mesh size Δ_0
while $\Delta > \Delta_{\min}$ **and** experiments $< N_{\max}$ **do**
    Generate poll set $\mathcal{P}(\boldsymbol{\psi}_{\text{best}},\Delta)$
    for $\boldsymbol{\psi}_{\text{poll}} \in \mathcal{P}$ **do**
        $\mathcal{D}_{\text{poll}} \leftarrow \mathcal{D}(\boldsymbol{\psi}_{\text{poll}})$
        $\mathcal{M}_{\text{poll}} \leftarrow \text{Train}(\mathcal{D}_{\text{poll}})$
        $\mathcal{L}_{\text{best}\to\text{poll}} \leftarrow \text{Eval}(\mathcal{M}_{\text{best}},\mathcal{D}_{\text{poll}})$
        $\mathcal{L}_{\text{poll}\to\text{best}} \leftarrow \text{Eval}(\mathcal{M}_{\text{poll}},\mathcal{D}_{\text{best}})$
        if $\mathcal{L}_{\text{poll}\to\text{best}} > \mathcal{L}_{\text{best}\to\text{poll}}$ **then**
            $\boldsymbol{\psi}_{\text{best}} \leftarrow \boldsymbol{\psi}_{\text{poll}}$ ▷ Successful poll
            break ▷ Continue with new center
        end if
    end for
    Update mesh size per pattern search rules
end while
return $\boldsymbol{\psi}_{\text{best}}$, $\mathcal{M}_{\text{best}}$

To evaluate the cross-validation approach and identify the conditions under which it succeeds or fails, a study is conducted using the Duffing oscillator from Section 5.1. The experimental campaign consists of 1000 independent optimization runs, each initialized from different randomly sampled point pairs distributed to ensure comprehensive parameter space coverage. Since the cross-validation approach provides only relative comparisons between datasets without absolute quality measures, direct error surface evaluation is impossible. However, by leveraging the ground truth error surface established in Section 5.1, the convergence behavior can be objectively assessed by determining whether each run terminates in the known global minimum region. Success is defined as convergence to parameter combinations where test error falls below the tolerance threshold of $\mathcal{L}_T < 0.01$, as identified by the reference error surface. This validation provides statistical insight into the method's reliability and reveals the parameter space characteristics that enable or prevent successful convergence.

Figure 7.12 presents the results of this comprehensive evaluation, showing both the initial

point distribution and the final converged locations for all 1000 optimization runs. The initial points in Figure 7.12(a) are distributed uniformly across the parameter space using the Hammersley sequence, ensuring unbiased coverage and eliminating systematic initialization bias. The final converged points in Figure 7.12(b) reveal a success rate of 201 out of 1000 runs, corresponding to 20.1% of attempts reaching the global minimum region as established in Fig. 5.7. While this success rate may appear modest, it establishes a crucial proof of concept. Adaptive sampling guided solely by cross-validation can indeed identify highly informative training datasets without any ground truth reference. This success rate represents a baseline performance for the simplest cross-validation strategy, demonstrating feasibility while simultaneously highlighting opportunities for refinement and improvement.

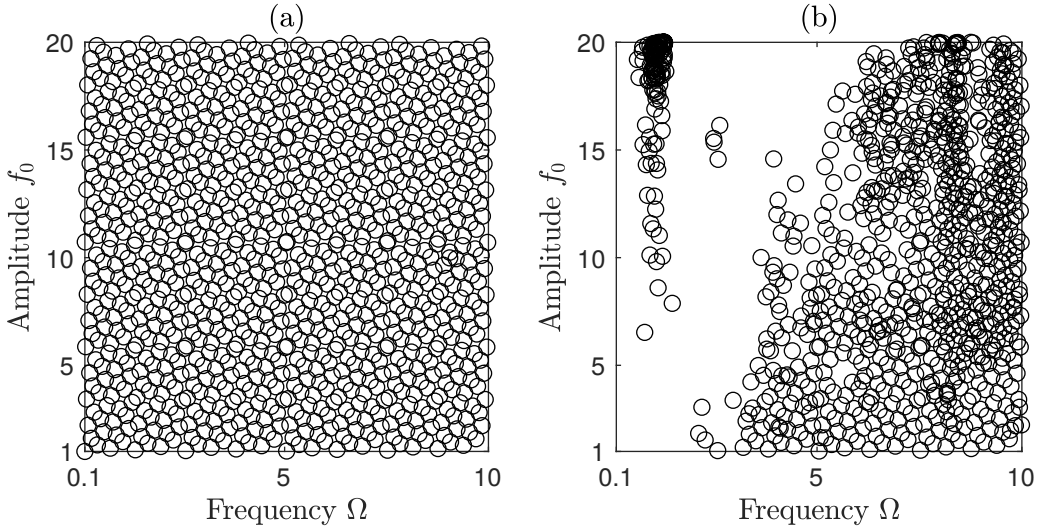

Figure 7.12: Reference-free adaptive sampling with cross-validation objective. (a) Initial point distribution. (b) Final converged points.

The spatial distribution of successful and unsuccessful runs reveals systematic patterns in the conditions enabling effective cross-validation guidance. Successful convergence exhibits strong correlation with initial point location, particularly when at least one initial point lies in regions of high error surface gradient near the system's natural frequency. Conversely, runs initialized exclusively in the flat high-frequency region $\Omega \in [5,10]$ Hz rarely succeed, as these parameter combinations produce qualitatively similar training datasets that yield comparable cross-validation errors regardless of relative information content. This observation exposes the limitation of the cross-validation criterion which relies on measurable differences in generalization performance between candidate datasets, and such differences vanish in flat error surface regions where all nearby parameter combinations produce similarly uninformative data.

Figure 7.13 illustrates representative convergence trajectories of successful and unsuccessful optimization runs. The successful case shown in Figure 7.13(a) begins with at least one initial point positioned in a high-gradient error surface region, where training datasets

exhibit substantial variation in information content. The cross-validation errors reliably reflect these differences, providing consistent gradient information that guides the pattern search toward increasingly informative parameter combinations. The convergence trajectory follows the error surface topology, demonstrating that cross-validation can serve as an effective surrogate for test error when sufficient gradient information exists.

In contrast, the unsuccessful case presented in Figure 7.13(b) illustrates the case when both initial points fall in flat error surface regions. The resulting training datasets, despite being generated with different parameter combinations, provide similar dynamic information due to the weak or similar system response. The cross-validation errors fail to distinguish meaningfully between these qualitatively similar datasets, providing insufficient gradient to guide optimization. The pattern search stagnates, unable to escape the flat region and locate more informative parameter combinations. This behavior confirms that cross-validation fails in regions where training dataset quality varies slowly or remains uniformly poor.

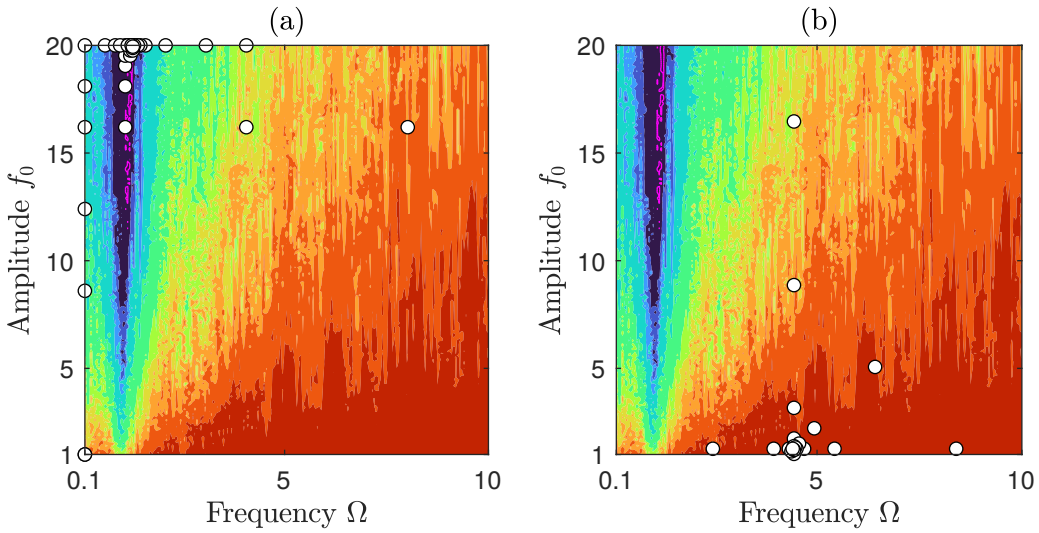

Figure 7.13: Representative convergence trajectories overlaid on test error surface. (a) Successful case where initial point in high-gradient region enables systematic descent to global minimum. (b) Unsuccessful case where initial points in flat region produce similar training datasets, the algorithm does not converge.

7.5.1 Multi-Validation Robust Optimization

The limitations observed in pure cross-validation motivate augmentation with information-theoretic measures that can provide complementary guidance when cross-validation gradients vanish. Information criteria such as the Akaike Information Criterion (AIC) and Bayesian Information Criterion (BIC) have been extensively studied for model selection in statistics, but their application to experimental design in structural dynamics remains limited. The AutoSID framework developed for process systems engineering represents one of few cases where information criteria and cross-validation are systematically combined for experimental design, employing Bayesian optimization to guide input signal design for system identification [6]. However, the structural dynamics literature lacks explicit multi-point validation strategies designed to improve neural network generalization across diverse operating conditions.

The multi-validation robust optimization framework developed here addresses this gap by replacing the binary cross-validation comparison with a comprehensive evaluation across multiple validation datasets distributed throughout the parameter space. Rather than comparing two candidate training datasets, the approach evaluates each candidate against a grid of N_{val} validation conditions spanning the expected operational range. This changes the optimization objective from seeking datasets that outperform individual alternatives to seeking datasets that perform consistently well across diverse conditions, thereby promoting robust generalization rather than comparative superiority.

The multi-validation objective function combines three complementary performance metrics that together capture different aspects of model quality. Hence, several validation datasets $\{\mathcal{D}_{\text{val}}^{(1)},\mathcal{D}_{\text{val}}^{(2)},\ldots,\mathcal{D}_{\text{val}}^{(N_{\text{val}})}\}$ are generated at parameter combinations $\boldsymbol{\psi}_{\text{val}}^{(i)}$ distributed across the parameter space using Latin hypercube sampling. For each candidate training dataset $\mathcal{D}_{\text{i}}(\boldsymbol{\psi})$, an ensemble of M neural networks $\{\mathcal{M}_1,\mathcal{M}_2,\ldots,\mathcal{M}_M\}$ with varying architectures is trained to capture different aspects of the underlying dynamics and reduce sensitivity to local minima. The validation error on each validation dataset is computed as:

$$\mathcal{L}_{\text{val}}^{(i)}(\boldsymbol{\psi}) = \frac{1}{N_i}\sum_{j=1}^{N_i}\left(y_j^{(i)} - \frac{1}{M}\sum_{k=1}^{M}\mathcal{M}_k(x_j^{(i)})\right)^2, \tag{7.9}$$

where N_i is the number of samples in validation dataset i and $\mathcal{L}_{\text{val}}^{(i)}(\boldsymbol{\psi})$ is the error averaged across the ensemble to reduce variance. Furthermore, the following individual validation errors are then aggregated into a robust objective function that balances multiple competing criteria:

$$\mathcal{L}_{\text{robust}}(\boldsymbol{\psi}) = \omega_{\text{avg}}\bar{\mathcal{L}}_{\text{val}} + \omega_{\max}\max_i \mathcal{L}_{\text{val}}^{(i)} + \omega_{\text{std}}\sigma_s(\mathcal{L}_{\text{val}}), \tag{7.10}$$

where $\bar{\mathcal{L}}_{\text{val}} = \frac{1}{N_{\text{val}}}\sum_{i=1}^{N_{\text{val}}}\mathcal{L}_{\text{val}}^{(i)}$ represents average performance across all validation conditions, $\max_i \mathcal{L}_{\text{val}}^{(i)}$ penalizes worst-case performance to prevent failures on any validation set, and $\sigma_s(\mathcal{L}_{\text{val}})$ promotes consistency by penalizing high variability in performance across dif-

ferent operating conditions. The weighting factors ω_{avg}, $\omega_{\max}$, and ω_{std} control the relative importance of these objectives and can be tuned based on application requirements. For the numerical studies presented here, values of $\omega_{\text{avg}} = 0.6$, $\omega_{\max} = 0.25$, and $\omega_{\text{std}} = 0.15$ are chosen to prioritize average performance while maintaining adequate robustness and consistency.

The optimization problem seeks to minimize this robust objective over the admissible parameter space Ψ:

$$\boldsymbol{\psi}_{opt} = \arg\min_{\boldsymbol{\psi} \in \Psi} \mathcal{L}_{\text{robust}}(\boldsymbol{\psi}), \tag{7.11}$$

subject to parameter bounds $\boldsymbol{\psi}_{\text{lb}} \leq \boldsymbol{\psi} \leq \boldsymbol{\psi}_{\text{ub}}$. The computational expense of evaluating this objective function is substantial, as each evaluation requires generating a training dataset, training M neural networks, and computing validation errors on N_{val} datasets. However, since the goal here is to minimize the number of experiments (evaluated sampling points) and not the computational effort, Bayesian optimization with Gaussian process surrogates is employed. This also demonstrates how the baseline ADaGen framework can be modified and customized to the given problem. Bayesian optimization with Gaussian process surrogates provides an efficient alternative by learning a probabilistic model of the objective function from past evaluations and using this model to intelligently select the next parameter combination to evaluate. The Gaussian process captures both the mean behavior and uncertainty in the objective function, enabling acquisition functions to balance exploitation of known good regions against exploration of uncertain regions.

Algorithm 5 presents the complete multi-validation robust optimization procedure. The algorithm begins by generating the validation grid using Latin hypercube sampling to ensure coverage of the parameter space without clustering of the sampling points. An ensemble of neural networks with varying hidden layer sizes between 10 and 30 neurons is initialized to capture different model complexities and promote diversity. The robust objective function incorporates a degeneracy check that assigns a large penalty of 10^6 to parameter combinations producing training data with variance below 10^{-6}, preventing optimization from selecting configurations that yield uninformative responses. The Bayesian optimization proceeds iteratively, with the Gaussian process surrogate progressively refining its model of the objective landscape as more parameter combinations are evaluated. After the specified number of iterations, the optimal parameters $\boldsymbol{\psi}_{opt}$ are identified from the Gaussian process predictions, a final ensemble is trained on data generated with these parameters, and the ensemble is pruned to the top 25 networks based on diversity-weighted performance to balance accuracy with model variety.

Algorithm 5 Multi-Validation Robust Optimization

Input: Parameter bounds $[\boldsymbol{\psi}_{\text{lb}}, \boldsymbol{\psi}_{\text{ub}}]$, ensemble size M, validation grid size N_{val}, max iterations N_{iter}
Output: Optimal parameters $\boldsymbol{\psi}_{opt}$, trained ensemble $\{\mathcal{M}_k^{opt}\}$
// Generate validation datasets
Sample N_{val} points $\{\boldsymbol{\psi}_{\text{val}}^{(1)}, \ldots, \boldsymbol{\psi}_{\text{val}}^{(N_{\text{val}})}\}$ using Latin hypercube design
for $i = 1$ to N_{val} **do**
    Generate validation dataset: $\mathcal{D}_{\text{val}}^{(i)} \leftarrow \texttt{generateData}(\boldsymbol{\psi}_{\text{val}}^{(i)})$
end for
// Initialize neural network ensemble
for $k = 1$ to M **do**
    Initialize network $\mathcal{M}_k$ with architecture from {10, 15, 20, 25, 30} neurons
end for
// Define robust objective function
function RobustObjective($\boldsymbol{\psi}$)
    Generate training dataset: $\mathcal{D}_{\text{train}} \leftarrow \texttt{generateData}(\boldsymbol{\psi})$
    if $\text{var}(\mathcal{D}_{\text{train}}) < 10^{-6}$ **then**
        return 10^6 ▷ Degenerate data penalty
    end if
    Train ensemble: $\{\mathcal{M}_k\} \leftarrow \texttt{trainEnsemble}(\mathcal{D}_{\text{train}})$
    for $i = 1$ to N_{val} **do**
        Compute ensemble prediction on $\mathcal{D}_{\text{val}}^{(i)}$
        $\mathcal{L}_{\text{val}}^{(i)} \leftarrow \text{MSE}(\text{ensemble prediction}, \mathcal{D}_{\text{val}}^{(i)})$
    end for
    $\bar{\mathcal{L}} \leftarrow \text{mean}(\{\mathcal{L}_{\text{val}}^{(i)}\})$
    $\mathcal{L}_{\max} \leftarrow \max(\{\mathcal{L}_{\text{val}}^{(i)}\})$
    $\sigma_{\mathcal{L}} \leftarrow \text{std}(\{\mathcal{L}_{\text{val}}^{(i)}\})$
    return $0.6\bar{\mathcal{L}} + 0.25\mathcal{L}_{\max} + 0.15\sigma_{\mathcal{L}}$
end function
// Bayesian optimization
Initialize Gaussian process surrogate with squared exponential kernel
Configure acquisition function: Expected Improvement Plus (exploration ratio = 0.5)
for iteration = 1 to N_{iter} **do**
    Query next point: $\boldsymbol{\psi}_{\text{next}} \leftarrow \arg\max_{\boldsymbol{\psi}} \text{AcquisitionFunction}(\boldsymbol{\psi})$
    Evaluate: $\mathcal{L}_{\text{next}} \leftarrow \texttt{RobustObjective}(\boldsymbol{\psi}_{\text{next}})$
    Update Gaussian process with $(\boldsymbol{\psi}_{\text{next}}, \mathcal{L}_{\text{next}})$
end for
$\boldsymbol{\psi}_{opt} \leftarrow \arg\min_{\boldsymbol{\psi}} \text{GP-predicted-objective}(\boldsymbol{\psi})$
Generate final dataset: $\mathcal{D}_{\text{train}}^{opt} \leftarrow \texttt{generateData}(\boldsymbol{\psi}_{opt})$
Train final ensemble: $\{\mathcal{M}_k^{opt}\} \leftarrow \texttt{trainEnsemble}(\mathcal{D}_{\text{train}}^{opt})$
Prune ensemble to top 25 networks by diversity-weighted performance
return $\boldsymbol{\psi}_{opt}$, $\{\mathcal{M}_k^{opt}\}$

The multi-validation framework is evaluated on the Duffing oscillator from Section 5.1

with parameter space $f_0 \in [0.1,20]$ and $\Omega \in [0.1,10]$ Hz. Five validation datasets are generated using Latin hypercube sampling to ensure coverage across the parameter space, with parameters listed in Table 7.4. The study was conducted in Matlab using Bayesian optimization with the Expected Improvement acquisition function, whose exploration ratio was set to 0.5. The experimental budget was set to 20, chosen to balance exploration thoroughness against computational cost given the expensive ensemble training required at each evaluation.

Table 7.4: Validation dataset parameters (Latin hypercube sampling)

Point	**Amplitude f_0**	**Frequency Ω [Hz]**
1	14.4	0.8
2	8.1	5.4
3	19.0	7.0
4	5.4	10.0
5	0.2	2.3

Figure 7.14 presents results which show that the multi-validation criterion successfully identifies optimum training datasets without requiring comprehensive test data. The convergence trajectory in Fig. 7.14(a) shows the robust objective value decreasing systematically over the 20 iterations, indicating successful learning and refinement. The parameter space exploration visualized in Fig. 7.14(b) reveals intelligent sampling behavior, with early iterations broadly exploring the parameter domain and later iterations concentrating evaluations near the emerging optimal region. The squares marking validation dataset locations are deliberately avoided by the optimization, as the robust objective discourages selecting training parameters identical to validation points. The optimization converges to $\boldsymbol{\psi}_{opt} = [18.89,1.19]$ Hz, corresponding to high-amplitude excitation near the system's natural frequency. This result is consistent with findings from Section 5.1 that resonant excitation provides maximally informative training data.

The error surface which GP surrogate learned during optimization, shown as the background color map in Figure 7.14(b), closely resembles the ground truth error surface from Section 5.1 despite being constructed from only 20 function evaluations. This agreement confirms that the robust objective captures the essential structure of the adaptation cost function. Furthermore, Table 7.5 documents the validation errors across all five validation points for the optimized model, demonstrating balanced performance with errors ranging from 0.0035 to 0.0091. This consistency across diverse parameter combinations confirms that the robust objective successfully promotes generalization rather than overfitting to specific conditions. The variation in individual validation errors reflects the inherent difficulty differences across the parameter space, with points near the natural frequency naturally achieving lower errors than those in high-frequency regions where system response is weaker and less informative.

These reference-free methods represent complementary tools rather than replacements for the baseline ADaGen framework with comprehensive test data. When ground truth

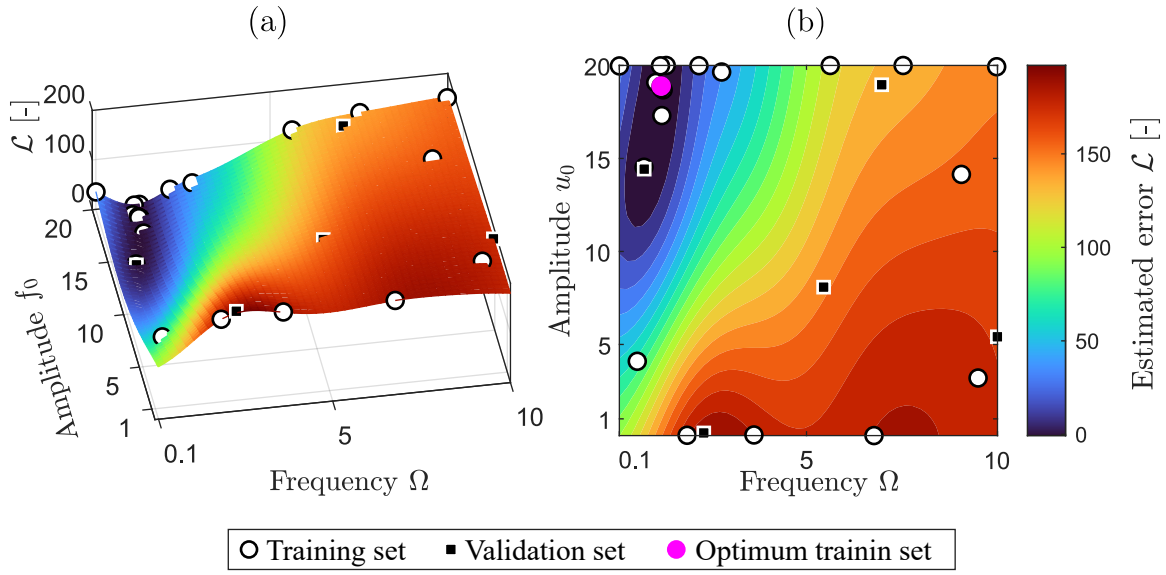

Figure 7.14: Multi-validation optimization results for Duffing oscillator. (a) GP surrogate of the error surface (cf. Fig. 5.7). (b) Contour plot of the GP surrogate of the error surface.

Table 7.5: Validation errors for multi-validation optimization

Validation set	Model Error
1	0.0083
2	0.0047
3	0.0091
4	0.0075
5	0.0035

validation is available, it provides unambiguous quality assessment and enables straightforward optimization convergence. When comprehensive testing is infeasible, reference-free methods offer viable alternatives that preserve the adaptive sampling paradigm. The multi-validation approach demonstrates particular promise for exploratory applications where the operational range is initially unknown, as the validation grid can be adaptively refined as understanding improves. Integration with the cumulative training strategies from Section 7.3 provides a natural path forward. Multi-validation optimization can identify promising initial experiments, followed by cumulative refinement if additional data proves necessary. The validation grid itself guides subsequent experiment selection by identifying parameter regions where performance remains inadequate, enabling systematic expansion of the training dataset until acceptable coverage is achieved. Future work should investigate adaptive validation grid refinement strategies, hybrid objective functions combining cross-validation with information-theoretic measures, and efficient parallel implementations to reduce the computational burden of ensemble training and validation evaluation.

7.6 Summary

This chapter addresses the assumptions and idealized conditions in the baseline framework to handle real-world structural dynamics challenges. While the baseline framework's assumptions of comprehensive test data availability, single-experiment sufficiency, SDOF system scope, and black-box ANN initialization enable rigorous validation of core concepts, they limit applicability to scenarios encountered in engineering practice. The four extensions presented systematically address each limitation while preserving the framework's advantage of learning from experimental data to guide subsequent experiments adaptively.

Interpretable ANN initialization through IML eliminates the computational burden of ensemble approaches by replacing random parameter initialization with values grounded in engineering knowledge. The resulting reduction in training variance enables single-network optimization at one-tenth the computational cost of the baseline ten-network ensemble approach while maintaining accuracy. Cumulative data generation addresses the reality that single experiments often cannot characterize complex nonlinear systems across their operational range. The optimization strategy first identifies a globally optimal dataset, then sequentially adds complementary experiments from different parameter space regions to achieve comprehensive coverage. For SDOF systems, this transforms scenarios where no single experiment meets accuracy requirements into robust solutions requiring only a few strategically selected experiments. The approach extends naturally to MDOF systems where multiple natural frequencies demand targeted excitation, with optimal cumulative datasets combining experiments near different system modes to capture complete dynamic response.

Reference-free adaptive sampling enables framework operation when comprehensive test data is unavailable. Cross-validation using relative performance comparisons between candidate training datasets establishes proof-of-concept feasibility but reveals critical dependence on initialization in parameter space regions with sufficient gradient structure. Multi-validation robust optimization addresses this limitation by evaluating candidates against multiple validation conditions distributed across the parameter space. This achieves more reliable convergence through Bayesian optimization with GP surrogates, identifying effective training datasets where the learned surrogate closely approximates the true error surface despite limited function evaluations.

These extensions transform the ADaGen framework from a validated research concept into a versatile methodology for diverse structural dynamics applications. IML initialization reduces computational cost while maintaining accuracy, cumulative strategies enable comprehensive system characterization when single experiments prove insufficient, MDOF extensions capture multi-modal dynamics through targeted excitation, and reference-free methods preserve adaptive sampling capabilities when comprehensive validation data cannot be obtained. Together, these advances demonstrate that adaptive data generation can efficiently guide model development across a wide range of practical scenarios.

Chapter 8

Conclusions

This dissertation establishes adaptive data generation as an efficient approach for experimental characterization of nonlinear vibration control devices. The central contribution is the ADaGen framework, which transforms physical testing from passive verification into active model generation by treating data collection as an optimization problem where experimental parameters are iteratively refined to minimize model error.

8.1 Key Contributions

The framework addresses limitations in conventional testing approaches. Traditional space-filling methods sample the parameter space uniformly without considering which experiments provide the most information for modeling. The ADaGen framework demonstrates that strategic, adaptive selection of experimental parameters reduces testing effort compared to batch sampling methods while achieving superior model accuracy. Pattern search optimization efficiently navigates the experimental parameter space to identify configurations that generate maximally informative training datasets.

The theoretical foundation addresses critical gaps in active learning for structural dynamics. Conventional uncertainty-based approaches fail for physical experiments with dynamic systems because desired states cannot be directly specified but must be reached through unknown system dynamics. The performance-based optimization strategy circumvents these limitations by operating directly in the experimental parameter space using deterministic neural network architectures without requiring probabilistic uncertainty quantification.

Numerical validation using the Duffing oscillator established baseline performance under controlled conditions. Across 1000 independent optimization runs, ADaGen achieved a 90% success rate in identifying optimal training datasets within approximately 19 experiments, compared to the 45 experiments required by Sobol sequence sampling. These efficiency gains translated to experimental validation using a tuned liquid column damper prototype under realistic conditions, including measurement noise, actuator constraints,

and unknown dynamics. The identified models were integrated directly into coupled structural simulations, demonstrating immediate practical applicability.

The Neural ODE extension enables system identification from displacement measurements alone without requiring force transducers. The modified architecture simultaneously learns system dynamics and unknown modal mass parameters by explicitly parameterizing mass-dependent input scaling. For tuned liquid column dampers, this yields the geometric factor essential for coupled structural analysis but challenging to determine analytically. The identified geometric factor of 0.40 closely matched theoretical expectations.

Framework extensions address key assumptions in the baseline methodology. Interpretable machine learning initialization eliminates ensemble requirements through physics-informed parameter assignment, achieving equivalent performance at one-tenth the computational cost. Cumulative data generation enables comprehensive characterization when single experiments prove insufficient, particularly for MDOF systems with multiple natural frequencies. Reference-free sampling through multi-validation robust optimization enables operation when comprehensive test datasets are unavailable. Gaussian process surrogates learn the optimization landscape from limited evaluations, making the approach practical even without reference data.

8.2 Limitations and Future Directions

Several limitations merit acknowledgment. First, the baseline framework targets systems with memory-free restoring forces. However, devices exhibiting substantial hysteresis require architectures that capture path-dependent behavior, necessitating modifications not yet addressed. Second, validation has focused on problems with 2–3 experimental parameters, and extension to high-dimensional spaces may encounter optimization convergence challenges. Finally, online adaptation strategies enabling real-time model updating as data streams arrive would extend applicability to high-throughput testing environments, but remain unexplored.

Future work should pursue experimental validation of the framework extensions, particularly cumulative strategies on physical MDOF structures and reference-free sampling under realistic conditions. Transfer learning approaches that leverage knowledge from previous similar devices could dramatically reduce the number of required experiments for new systems. Hybrid optimization strategies combining pattern search for exploration with Bayesian optimization for refinement may improve efficiency. Extending the methodology to hysteretic systems through recurrent architectures represents both a theoretical challenge and a practical necessity for metallic yielding dampers and friction-based devices.

8.3 Closing Remarks

This work demonstrates that adaptive experimental design through optimization-guided testing offers substantial efficiency gains compared to conventional approaches while maintaining or improving model accuracy. Validation studies confirm that the framework operates effectively under realistic experimental conditions. Extensions addressing interpretable initialization, cumulative training, MDOF systems, and test-free sampling establish versatility across diverse scenarios.

The ADaGen framework provides theoretical foundations and practical tools for integrating physical testing with data-driven modeling. As structural systems grow more complex and computational capabilities advance, this integration becomes increasingly essential. The methodology presented here establishes a foundation for testing-integrated modeling across structural dynamics applications. Strategic experimental design guided by model performance can fundamentally improve how physical testing programs are conceptualized and executed.

Nomenclature

Roman Symbols

$A(\mathbf{x})$	Acquisition function quantifying expected utility of querying at point $\mathbf{x}$
$\mathbf{A}$	State matrix in state-space representation
A_H	Cross-sectional area of horizontal part of TLCD
A_V	Cross-sectional area of vertical part of TLCD
$\mathbf{a}^l$	Activation vector of the l-th layer in neural network
$\mathbf{a}(t)$	Adjoint state in Neural ODE training
$\mathbf{B}$	Input matrix in state-space representation
$\mathbf{b}^l$	Bias vector of the l-th layer in neural network
c	Damping coefficient
$d_{\min}$	Minimum distance constraint for parameter separation
d_3	Cell length in TLCD
D_s	Damping ratio of structure
D_N^*	Star-discrepancy for quasi-random sequences
$\mathcal{D}$	Training dataset
$\mathcal{D}_i$	Training dataset from experiment i
$\mathcal{D}_T$	Test dataset
$\mathcal{D}_{opt}$	Optimal training dataset
$\mathcal{D}_{cum}$	Cumulative training dataset
$\mathcal{D}_{val}^{(i)}$	Validation dataset i
f	Mass-normalized excitation force
f_0	Excitation amplitude
f_R	Restoring force
f_s	Natural frequency of structure
f_S	Cyclic excitation frequency
f_θ	Neural network function with parameters θ
$F(t)$	External forcing function
$F_1(t)$,$F_2(t)$	Forces applied to specific floors (MDOF system)
$\mathbf{h}^l$	Activation of the l-th layer
k	Stiffness coefficient
L_H	Horizontal distance between column axes in TLCD
L_V	Idle liquid height in TLCD
m	Mass
m_d	Mass of damper liquid
m_s	Modal mass of structure
M	Ensemble size; dimension of experiment parameter space
n	Number of inputs to artificial neuron; state dimension

N	Number of time steps; sample size
N_{exp}	Number of experiments
N_{val}	Number of validation datasets
N_t	Time-depth parameter; prediction horizon
t	Time
t_0	Initial time
u	Liquid displacement in TLCD
$\dot{u}$	Liquid velocity in TLCD
$\ddot{u}$	Liquid acceleration in TLCD
$\ddot{u}_g$	Ground acceleration
$\mathbf{W}^l$	Weight matrix connecting layers $l-1$ and l
w_i	Weight associated with input i
x	Displacement
x_0	Displacement amplitude
$\dot{x}$	Velocity
$\ddot{x}$	Acceleration
$\ddot{x}_0$	Acceleration amplitude
$\mathbf{x}$	Input vector; displacement vector
y	Output; target value
$\hat{y}$	Predicted output
Z	Standardized variable in Expected Improvement

Greek Symbols

α	Damping coefficient (Duffing oscillator)
β	Linear stiffness coefficient; exploration parameter
γ	Nonlinear stiffness coefficient
γ_1	Geometric factor coupling table acceleration to liquid motion
$\hat{\gamma}_1$	Estimated geometric factor
γ_2	Geometric factor for coupled structural simulation
$\hat{\gamma}_2$	Estimated geometric factor for coupled structural simulation
Δ	Mesh size in pattern search
Δ_{min}	Minimum mesh size
δ	Head loss coefficient; mesh contraction factor
ϵ	Error tolerance
η	Learning rate
θ	Network parameters (weights and biases)
θ_m	Mass parameter in Neural ODE
$\boldsymbol{\theta}$	Parameter vector
$\mathcal{L}$	Loss function
$\mathcal{L}_T$	Adaptation cost function; test error

$\mathcal{L}_{robust}$	Robust objective function
μ	Mass ratio; predictive mean
μ_*	Predictive mean at new point
ξ	Exploration parameter in Probability of Improvement
$\sigma(\cdot)$	Activation function
σ^2	Predictive variance
σ_*^2	Predictive variance at new point
$\sigma^2_{\text{ensemble}}$	Ensemble prediction variance
τ	Mesh expansion factor
ϕ	Scaling factor; standard normal PDF
Φ	Standard normal CDF
$\boldsymbol{\psi}$	Experiment parameters; sampling parameters
$\boldsymbol{\psi}_i$	Experiment parameters for iteration i
$\boldsymbol{\psi}_{opt}$	Optimal experiment parameters
$\boldsymbol{\psi}_T$	Test parameters
ω_D	Natural circular frequency of damper
ω_s	Natural circular frequency of structure
Ω	Excitation frequency

Abbreviations

ADaGen	Adaptive Data Generation
AI	Artificial Intelligence
AIC	Akaike Information Criterion
AL	Active Learning
ANN	Artificial Neural Network
APRBS	Amplitude Modulated Pseudo-Random Binary Signal
BIC	Bayesian Information Criterion
CDF	Cumulative Distribution Function
CNN	Convolutional Neural Network
DOE	Design of Experiments
EI	Expected Improvement
ES	Entropy Search
FNN	Feedforward Neural Network
GA	Genetic Algorithm
GD	Gradient Descent
GNN	Graph Neural Network
GP	Gaussian Process
IML	Interpretable Machine Learning
LHS	Latin Hypercube Sampling
LSTM	Long Short-Term Memory

MADS	Mesh Adaptive Direct Search
MAE	Mean Absolute Error
MDOF	Multi-Degree-of-Freedom
ML	Machine Learning
MLP	Multilayer Perceptron
MSE	Mean Squared Error
NRMSE	Normalized Root Mean Square Error
ODE	Ordinary Differential Equation
O-TLCD	Omnidirectional Tuned Liquid Column Damper
PDF	Probability Density Function
PES	Predictive Entropy Search
PI	Probability of Improvement
PINN	Physics-Informed Neural Network
PRBS	Pseudo-Random Binary Signal
PS	Pattern Search
RBFN	Radial Basis Function Neural Network
ReLU	Rectified Linear Unit
ResNet	Residual Network
RNN	Recurrent Neural Network
RSM	Response Surface Methodology
SDOF	Single-Degree-of-Freedom
SGD	Stochastic Gradient Descent
TLCD	Tuned Liquid Column Damper
TMD	Tuned Mass Damper
UCB	Upper Confidence Bound

Mathematical Notation

$\mathbb{R}$	Set of real numbers
$\mathbb{R}^n$	n-dimensional real vector space
$\mathcal{N}(\mu,\sigma^2)$	Normal distribution with mean μ and variance σ^2
$\mathcal{U}(a,b)$	Uniform distribution on interval $[a,b]$
$\mathcal{GP}(\mu,k)$	Gaussian process with mean function μ and kernel k
$\|\cdot\|$	Euclidean norm
$\|\cdot\|_2$	L_2 norm
∇	Gradient operator
$\partial/\partial t$	Partial derivative with respect to time
$\dot{(\cdot)}$	Time derivative
$\ddot{(\cdot)}$	Second time derivative
$\arg\min$	Argument that minimizes
$\arg\max$	Argument that maximizes

$\mathbb{E}[\cdot]$	Expected value
$H[\cdot]$	Differential entropy
$\oplus$	Bitwise XOR operation
$\bigcup$	Set union
$\in$	Element of
$\times$	Cartesian product
$\sim$	Distributed as
$\propto$	Proportional to
$\mathcal{O}(\cdot)$	Big-O notation (computational complexity)

Bibliography

[1] G. Abbiati and S. Marelli. Sequential experimental design of hybrid simulations for Bayesian calibration of computational simulators. In: *Proceedings of the 17th World Conference on Earthquake Engineering*, Sendai, Japan, Sep 2021.

[2] G. Abbiati, S. Marelli, C. Ligeikis, R. Christenson, and B. Stojadinović. Training of a classifier for structural component failure based on hybrid simulation and kriging. *Journal of Engineering Mechanics*, 148(1), 2022. DOI: 10.1061/(ASCE)EM.1943-7889.0002048.

[3] O. Abdeljaber, O. Avci, S. Kiranyaz, M. Gabbouj, and D. J. Inman. Real-time vibration-based structural damage detection using one-dimensional convolutional neural networks. *Journal of Sound and Vibration*, 388:154–170, Feb. 2017. DOI: 10.1016/j.jsv.2016.10.043.

[4] M. Abolhasani and E. Kumacheva. The rise of self-driving labs in chemical and materials sciences. *Nature Synthesis*, 2(6):483–492, jun 2023.

[5] H. Adeli. Neural Networks in Civil Engineering: 1989–2000. *Computer-Aided Civil and Infrastructure Engineering*, 16(2):126–142, Mar. 2001. DOI: 10.1111/0885-9507.00219.

[6] A. Ahmed, E. A. del Rio-Chanona, and M. Mercangöz. Comparative study of machine learning and system identification for process systems engineering dynamics. *Industrial & Engineering Chemistry Research*, 64(8):4450–4478, 2025. DOI: 10.1021/acs.iecr.4c03264.

[7] S. Alam, N. Siddique, and H. Adeli. A dynamic ensemble learning algorithm for neural networks. *Neural Computing and Applications*, 32(12):8675–8684, 2020. DOI: 10.1007/s00521-019-04359-7.

[8] O. Altay. *Vibration Mitigation Systems in Structural Engineering*. CRC Press, Boca Raton, Florida, 2021. DOI: 10.1201/9781315122243.

[9] O. Altay and S. Klinkel. A semi-active tuned liquid column damper for lateral vibration control of high-rise structures: Theory and experimental verification. *Structural Control and Health Monitoring*, 25(12), 10 2018. DOI: 10.1002/stc.2270.

[10] S. Anderson, K. Byl, and J. P. Hespanha. Experiment design with gaussian process regression with applications to chance-constrained control, 2025. DOI: 10.48550/arXiv.2502.14066.

[11] C. J. Arthurs and A. P. King. Active training of physics-informed neural networks to aggregate and interpolate parametric solutions to the navier–stokes equations. *Journal of Computational Physics*, 438:110364, 2021. DOI: 10.1016/j.jcp.2021.110364.

[12] C. Audet and J. E. J. Dennis. Analysis of generalized pattern searches. *SIAM Journal on Optimization*, 13(3):889–903, 2003. DOI: 10.1137/S1052623400378742.

[13] C. Audet and J. E. J. Dennis. Mesh adaptive direct search algorithms for constrained optimization. *SIAM Journal on Optimization*, 17(1):188–217, 2006. DOI: 10.1137/040603371.

[14] O. Avci, O. Abdeljaber, S. Kiranyaz, M. Hussein, M. Gabbouj, and D. J. Inman. A review of vibration-based damage detection in civil structures: From traditional methods to Machine Learning and Deep Learning applications. *Mechanical Systems and Signal Processing*, 147:107077, Jan. 2021. DOI: 10.1016/j.ymssp.2020.107077.

[15] S. J. Bates, J. Sienz, and V. V. Toropov. Formulation of the optimal latin hypercube design of experiments using a permutation genetic algorithm. *45th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics & Materials Conference*, pp. 1–7, 2004. DOI: 10.2514/6.2004-2011.

[16] J. Belz, K. Bamberger, O. Nelles, and T. Carolus. Goal-oriented active learning with local model networks. *International Journal of Computational Methods and Experimental Measurements*, 6(4):785–796, Nov. 2017. DOI: 10.2495/CMEM-V6-N4-785-796.

[17] C. M. Bishop. *Neural Networks for Pattern Recognition*. Clarendon Press, Oxford, 1995.

[18] C. M. Bishop. *Pattern Recognition and Machine Learning*. Information Science and Statistics. Springer, New York, 2006. ISBN 978-0-387-31073-2.

[19] A. Bogdanovic, Z. Rakicevic, and E. Noroozinejad Farsangi. Shake table tests and numerical investigation of a resilient damping device for seismic response control of building structures. *Structural Control and Health Monitoring*, 26(11):e2443, 2019. DOI: https://doi.org/10.1002/stc.2443.

[20] G. E. Box, J. S. Hunter, and W. G. Hunter. *Statistics for Experimenters: Design, Innovation, and Discovery*. Wiley-Interscience, Hoboken, NJ, 2nd edition, 2005.

[21] G. E. P. Box and K. B. Wilson. On the experimental attainment of optimum conditions. *Journal of the Royal Statistical Society, Series B*, 13(1):1–45, 1951.

[22] W. Bradley, R. Volkovinsky, and F. Boukouvala. Enabling global interpolation, derivative estimation and model identification from sparse multi-experiment time series data via neural ODEs. *Eng. Appl. Artif. Intell.*, 130:107611, 2024. DOI: 10.1016/j.engappai.2023.107611.

[23] P. Bratley and B. L. Fox. Algorithm 659: Implementing Sobol's quasirandom sequence generator. *ACM Transactions on Mathematical Software*, 14(1):88–100, 1988. DOI: 10.1145/42288.214372.

[24] M. Buisson-Fenet, F. Solowjow, and S. Trimpe. Actively learning gaussian process dynamics. In: *Learning for dynamics and control*, pp. 5–15, 2020. DOI: 10.48550/arXiv.1911.09946.

[25] L. Bull, K. Worden, G. Manson, and N. Dervilis. Active learning for semi-supervised structural health monitoring. *Journal of Sound and Vibration*, 437:373–388, 2018. DOI: 10.1016/j.jsv.2018.08.040.

[26] L. Bull, T. Rogers, C. Wickramarachchi, E. Cross, K. Worden, and N. Dervilis. Probabilistic active learning: An online framework for structural health monitoring. *Mechanical Systems and Signal Processing*, 134, 2019. DOI: 10.1016/j.ymssp.2019.106294.

[27] R. E. Caflisch. Monte carlo and quasi-monte carlo methods. *Acta Numerica*, 7:1–49, 1998. DOI: 10.1017/S0962492900002804.

[28] F. Casciati and L. Faravelli. Experimental investigation on the aging of the base isolator elastomeric component. *Acta Mechanica*, 223(8):1633–1643, 2012. DOI: 10.1007/s00707-012-0636-x.

[29] T. Q. Chen, Y. Rubanova, J. Bettencourt, and D. K. Duvenaud. Neural ordinary differential equations. In: *Advances in Neural Information Processing Systems*, pp. 6571–6583, 2018.

[30] D. A. Cohn, Z. Ghahramani, and M. I. Jordan. Active learning with statistical models. *Journal of Artificial Intelligence Research*, 4:129–145, 1996.

[31] E. F. Crawley and K. J. O'donnell. Force-state mapping identification of nonlinear joints. *AIAA journal*, 25(7):1003–1010, 1987. DOI: https://doi.org/10.2514/3.9733.

[32] S. Cuomo, V. S. Di Cola, F. Giampaolo, G. Rozza, M. Raissi, and F. Piccialli. Scientific machine learning through physics-informed neural networks: Where we are and what's next. *Journal of Scientific Computing*, 92(3):88, 2022. DOI: 10.1007/s10915-022-01939-z.

[33] G. Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of control, signals and systems*, 2(4):303–314, 1989.

[34] J. Decke, A. Heinen, B. Sick, and C. Gruhl. Active learning with physics-informed graph neural networks on unstructured meshes. In: *CEUR Workshop Proceedings*, vol. 3770, pp. 68–76, Vilnius, Lithuania, 2024. IAL@ECML-PKDD'24: 8th International Workshop & Tutorial on Interactive Adaptive Learning.

[35] M. Deflorian and S. Zaglauer. Design of experiments for nonlinear dynamic system identification. *Journal of Process Control*, 21(10):1476–1489, 2011. DOI: 10.1016/j.jprocont.2011.06.009.

[36] A. Di Matteo, T. Furtmüller, C. Adam, and A. Pirrotta. Optimal design of tuned liquid column dampers for seismic response control of base-isolated structures. *Acta Mechanica*, 229(2):437–454, 2018. DOI: https://doi.org/10.1007/s00707-017-1980-7.

[37] H. Ding, A. Kaup, J.-T. Wang, L.-Q. Lu, and O. Altay. Real-time hybrid simulation framework for the investigation of soil-structure interaction effects on the vibration control performance of shape memory alloys. *Engineering Structures*, 243:112621, 2021. DOI: 10.1016/j.engstruct.2021.112621.

[38] H. Ding, J.-T. Wang, O. Altay, and L.-Q. Lu. Multilayer toroidal tuned liquid column dampers for seismic vibration control of structures. *Structures*, 33:406–422, 2021. DOI: 10.1016/j.istruc.2021.04.041.

[39] B. Efron. *The Jackknife, the Bootstrap and Other Resampling Plans*, vol. 38. SIAM, 1982.

[40] J. L. Elman. Finding structure in time. *Cognitive Science*, 14(2):179–211, 1990. DOI: 10.1207/s15516709cog1402_1.

[41] V. V. Fedorov. *Theory of Optimal Experiments*. Academic Press, New York, 1972. Translated from Russian and edited by W. J. Studden and E. M. Klimko.

[42] R. A. Fisher. *The Design of Experiments*. Oliver & Boyd, Edinburgh, 1935.

[43] R. A. Fisher. *Statistical Methods, Experimental Design and Scientific Inference: A Re-Issue of Statistical Methods for Research Workers, The Design of Experiments and Statistical Methods and Statistical Inference*. Oxford University Press, Oxford, 1990.

[44] J. N. Fuhg, A. Fau, and U. Nackenhorst. State-of-the-Art and Comparative Review of Adaptive Sampling Methods for Kriging. *Archives of Computational Methods in Engineering*, 28(4):2689–2747, Jun 2021.

[45] P. Gardner, M. Dal Borgo, V. Ruffini, A. Hughes, Y.-C. Zhu, and D. Wagg. Towards the Development of an Operational Digital Twin. *Vibration*, 3:235–265, 09 2020. DOI: 10.3390/vibration3030018.

[46] S. Geman, E. Bienenstock, and R. Doursat. Neural Networks and the Bias/Variance Dilemma. *Neural Computation*, 4(1):1–58, 1992.

[47] X. Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. In: *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pp. 249–256. JMLR Workshop and Conference Proceedings, 2010.

[48] D. E. Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley, 1989. ISBN 978-0-201-15767-3.

[49] I. Goodfellow, Y. Bengio, and A. Courville. *Deep learning*. MIT Press, Cambridge, Massachusetts, 2016.

[50] M. Gringard and A. Kroll. On the parametrization of APRBS and multisine test signals for the identification of nonlinear dynamic TS-models. In: *2016 IEEE Symposium Series on Computational Intelligence (SSCI)*, pp. 1–8, Dec. 2016. DOI: 10.1109/SSCI.2016.7849984.

[51] Q. Guo, X. Yang, K. Li, and D. Li. Parameters identification of magnetorheological damper based on particle swarm optimization algorithm. *Engineering Applications of Artificial Intelligence*, 143:110016, 2025. DOI: 10.1016/j.engappai.2025.110016.

[52] M. Gutierrez Soto and H. Adeli. Tuned Mass Dampers. *Arch. Comput. Methods Eng.*, 20(4):419–431, 2013. DOI: 10.1007/s11831-013-9091-7.

[53] C. Hametner, M. Stadlbauer, M. Deregnaucourt, S. Jakubek, and T. Winsel. Optimal experiment design based on local model networks and multilayer perceptron networks. *Eng. Appl. Artif. Intell.*, 26(1):251–261, 2013. DOI: 10.1016/j.engappai.2012.05.016.

[54] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, June 2016.

[55] T. O. Heinz and O. Nelles. Iterative Excitation Signal Design for Nonlinear Dynamic Black-Box Models. *Procedia Computer Science*, 112:1054–1061, 2017. DOI: 10.1016/j.procs.2017.08.112.

[56] P. Hennig and C. J. Schuler. Entropy search for information-efficient global optimization. *Journal of Machine Learning Research*, 13:1809–1837, 2012.

[57] J. M. Hernández-Lobato, M. W. Hoffman, and Z. Ghahramani. Predictive entropy search for efficient global optimization of black-box functions. In: *Advances in Neural Information Processing Systems*, pp. 918–926, 2014.

[58] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997. DOI: 10.1162/neco.1997.9.8.1735.

[59] K. Hornik, M. Stinchcombe, and H. White. Multilayer feedforward networks are universal approximators. *Neural networks*, 2(5):359–366, 1989.

[60] X. Huang and B. Yang. Towards novel energy shunt inspired vibration suppression techniques: Principles, designs and applications. *Mechanical Systems and Signal Processing*, 182, 2023. DOI: 10.1016/j.ymssp.2022.109496.

[61] A. Hughes, L. Bull, P. Gardner, N. Dervilis, and K. Worden. On robust risk-based active-learning algorithms for enhanced decision support. *Mechanical Systems and Signal Processing*, 181:109502, 2022. DOI: 10.1016/j.ymssp.2022.109502.

[62] M. Ismail, F. Ikhouane, and J. Rodellar. The Hysteresis Bouc-Wen Model, a Survey. *Archives of Computational Methods in Engineering*, 16:161–188, 06 2009. DOI: 10.1007/s11831-009-9031-8.

[63] H. Jaeger. The "echo state" approach to analysing and training recurrent neural networks. *GMD Technical Report*, 148:34, 2001.

[64] R. Jin, W. Chen, and A. Sudjianto. An efficient algorithm for constructing optimal design of computer experiments. *Journal of Statistical Planning and Inference*, 134 (1):268–287, 2005. DOI: 10.1016/j.jspi.2004.02.014.

[65] D. R. Jones, M. Schonlau, and W. J. Welch. Efficient global optimization of expensive black-box functions. *Journal of Global Optimization*, 13(4):455–492, 1998. DOI: 10.1023/A:1008306431147.

[66] H. Kapadia, L. Feng, and P. Benner. Active-learning-driven surrogate modeling for efficient simulation of parametric nonlinear systems. *Computer Methods in Applied Mechanics and Engineering*, 419:116657, 2024. DOI: 10.1016/j.cma.2023.116657.

[67] G. Kerschen, K. Worden, A. Vakakis, and J.-C. Golinval. Past, present and future of nonlinear system identification in structural dynamics. *Mechanical Systems and Signal Processing*, 20(3):505–592, 2006. DOI: 10.1016/j.ymssp.2005.04.008.

[68] M. Khalid, R. Yusof, M. Joshani, H. Selamat, and M. Joshani. Nonlinear Identification of a Magneto-Rheological Damper Based on Dynamic Neural Networks. *Computer-Aided Civil and Infrastructure Engineering*, 29(3), 2014. DOI: 10.1111/mice.12005.

[69] J. Kiani, C. Camp, S. Pezeshk, and N. Khoshnevis. Application of pool-based active learning in reducing the number of required response history analyses. *Computers & Structures*, 241, 2020. DOI: 10.1016/j.compstruc.2020.106355.

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

[71] T. G. Kolda, R. M. Lewis, and V. Torczon. Optimization by Direct Search: New Perspectives on Some Classical and Modern Methods. *SIAM Review*, 45(3):385–482, Jan. 2003. DOI: 10.1137/S003614450242889.

[72] P. Kumar and A. Gupta. Active learning query strategies for classification, regression, and clustering: A survey. *Journal of Computer Science and Technology*, 35(4): 913–945, July 2020. DOI: 10.1007/s11390-020-9487-4.

[73] Z. Lai, C. Mylonas, S. Nagarajaiah, and E. Chatzi. Structural identification with physics-informed neural ordinary differential equations. *Journal of Sound and Vibration*, 508:116196, Sept. 2021. DOI: 10.1016/j.jsv.2021.116196.

[74] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural Computation*, 1(4):541–551, 1989. DOI: 10.1162/neco.1989.1.4.541.

[75] S. Li and Y. Yang. A recurrent neural network framework with an adaptive training strategy for long-time predictive modeling of nonlinear dynamical systems. *Journal of Sound and Vibration*, 506:116167, 2021. DOI: https://doi.org/10.1016/j.jsv.2021.116167.

[76] W.-I. Liao, I. Mualla, and C.-H. Loh. Shaking-table test of a friction-damped frame structure. *The Structural Design of Tall and Special Buildings*, 13:45 – 54, 03 2004. DOI: https://doi.org/10.1002/tal.232.

[77] P. Liu, L. Wang, R. Ranjan, G. He, and L. Zhao. A survey on active deep learning: From model driven to data driven. *ACM Comput. Surv.*, 54(10s), sep 2022. DOI: https://doi.org/10.1145/3510414.

[78] L. Ljung. *System identification: theory for the user*. Prentice Hall information and system sciences series. Prentice Hall PTR, Upper Saddle River, NJ, 2nd edition, 1999. ISBN 978-0-13-656695-3.

[79] C.-H. Loh, K. J. Loh, Y.-S. Yang, W.-Y. Hsiung, and Y.-T. Huang. Vibration-based system identification of wind turbine system: Vibration-based System Identification of Turbine Blades. *Structural Control and Health Monitoring*, 24(3):e1876, 2017. DOI: 10.1002/stc.1876.

[80] J. Lundby, M. K. Ebbesen, and L. Imsland. Deep active learning for nonlinear system identification. *arXiv preprint arXiv:2303.08085*, 2023.

[81] W. Maass, T. Natschläger, and H. Markram. Real-time computing without stable states: A new framework for neural computation based on perturbations. *Neural Computation*, 14(11):2531–2560, 2002. DOI: 10.1162/089976602760407955.

[82] M. Machado and M. Dutkiewicz. Wind turbine vibration management: An integrated analysis of existing solutions, products, and open-source developments. *Energy Reports*, 11:3756–3791, 2024. DOI: 10.1016/j.egyr.2024.03.014.

[83] R. Machavaram and K. Shankar. Structural damage identification using improved RBF neural networks in frequency domain. *Advances in Structural Engineering*, 15 (10):1689–1710, Oct. 2012. DOI: 10.1260/1369-4332.15.10.1689.

[84] D. J. MacKay. Information-based objective functions for active data selection. *Neural Computation*, 4(4):590–604, 1992. DOI: https://doi.org/10.1162/neco.1992.4.4.590.

[85] H. Mania, M. I. Jordan, and B. Recht. Active learning for nonlinear system identification with guarantees. *Journal of Machine Learning Research*, 21(1):1–47, 2020.

[86] S. F. Masri, A. G. Chassiakos, and T. K. Caughey. Identification of nonlinear dynamic systems using neural networks. *Journal of Applied Mechanics*, 60(1):123–133, 1993. DOI: 10.1115/1.2900734.

[87] MATLAB. *version 9.11.0 (R2021b)*. The MathWorks Inc., Natick, Massachusetts, 2021.

[88] W. S. McCulloch and W. Pitts. A logical calculus of the ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics*, 5:115–133, 1943.

[89] M. D. McKay, R. J. Beckman, and W. J. Conover. A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics*, 21(2):239–245, 1979. DOI: doi.org/10.2307/1268522.

[90] B. Mehrkian and O. Altay. Mathematical modeling and optimization scheme for omnidirectional tuned liquid column dampers. *Journal of Sound and Vibration*, 484, 2020. DOI: 10.1016/j.jsv.2020.115523.

[91] B. Mehrkian and O. Altay. Omnidirectional liquid column vibration absorbers for multi-story buildings. *Journal of Building Engineering*, 62, 2022. DOI: 10.1016/j.jobe.2022.105306.

[92] P. Milicevic and O. Altay. Data generation framework for inverse modeling of nonlinear systems in structural dynamics applications. *Acta Mechanica*, 235(3): 1493–1515, mar 2024. DOI: https://doi.org/10.1007/s00707-023-03532-3.

[93] P. Milicevic, S. Stemmler, and O. Altay. Active learning with neural ordinary differential equations for data-efficient modeling of tuned mass dampers. *Engineering Applications of Artificial Intelligence*, 173:114487, 2026. DOI: 10.1016/j.engappai.2026.114487.

[94] D. C. Montgomery. *Design and Analysis of Experiments*. John Wiley & Sons, Hoboken, NJ, 10 edition, 2017.

[95] J. Moody and C. J. Darken. Fast learning in networks of locally-tuned processing units. *Neural Computation*, 1(2):281–294, 1989. DOI: 10.1162/neco.1989.1.2.281.

[96] M. D. Morris and T. J. Mitchell. Exploratory designs for computational experiments. *Journal of Statistical Planning and Inference*, 43(3):381–402, 1995. DOI: 10.1016/0378-3758(94)00035-T.

[97] M. A. Nabian, R. J. Gladstone, and H. Meidani. Efficient training of physics-informed neural networks via importance sampling. *Computer-Aided Civil and Infrastructure Engineering*, 36(8):962–977, 2021. DOI: 10.1111/mice.12685.

[98] O. Nelles. *Nonlinear System Identification: From Classical Approaches to Neural Networks, Fuzzy Models, and Gaussian Processes*. Springer International Publishing, Cham, 2nd edition, 2020. ISBN 978-3-030-47438-6. DOI: 10.1007/978-3-030-47439-3.

[99] H. Niederreiter. *Random Number Generation and Quasi-Monte Carlo Methods*. CBMS-NSF Regional Conference Series in Applied Mathematics. Society for Industrial and Applied Mathematics, 1992. ISBN 978-0-89871-295-7. DOI: 10.1137/1.9781611970081.

[100] J. Noël and G. Kerschen. Nonlinear system identification in structural dynamics: 10 more years of progress. *Mechanical Systems and Signal Processing*, 83:2–35, 2017. DOI: 10.1016/j.ymssp.2016.07.020.

[101] A. Novák, L. Simon, F. Kadlec, and P. Lotton. Nonlinear system identification using exponential swept-sine signal. *IEEE Transactions on Instrumentation and Measurement*, 59(8):2220–2229, 2010. DOI: 10.1109/TIM.2009.2031836.

[102] W. Ostachowicz, R. Soman, and P. Malinowski. Optimization of sensor placement for structural health monitoring: A review. *Structural Health Monitoring*, 18(3): 963–988, 2019. DOI: 10.1177/1475921719825601.

[103] J. Pathak, B. Hunt, M. Girvan, Z. Lu, and E. Ott. Model-free prediction of large spatiotemporally chaotic systems from data: A reservoir computing approach. *Physical Review Letters*, 120(2):024102, 2018. DOI: 10.1103/PhysRevLett.120.024102.

[104] J.-S. Pei. *Parametric and nonparametric identification of nonlinear systems*. Columbia University, 2001.

[105] J.-S. Pei and E. C. Mai. Constructing multilayer feedforward neural networks to approximate nonlinear functions in engineering mechanics applications. *Journal of Applied Mechanics*, 75, 2008.

[106] J.-S. Pei and A. W. Smyth. New approach to designing multilayer feedforward neural network architecture for modeling nonlinear restoring forces. II: Applications. *Journal of Engineering Mechanics*, 132, 2006. DOI: 10.1061/(ASCE)0733-9399(2006)132:12(1301.

[107] J.-S. Pei and A. W. Smyth. New approach to designing multilayer feedforward neural network architecture for modeling nonlinear restoring forces. I: Formulation. *Journal of engineering mechanics*, 132, 2006. DOI: 10.1061/(ASCE)0733-9399(2006)132:12(1290).

[108] J.-S. Pei, J. P. Wright, and A. W. Smyth. Mapping polynomial fitting into feedforward neural networks for modeling nonlinear dynamic systems and beyond. *Computer Methods in Applied Mechanics and Engineering*, 194, 2005. DOI: 10.1016/j.cma.2004.12.010.

[109] J.-S. Pei, E. C. Mai, J. P. Wright, and A. W. Smyth. Neural network initialization with prototypes-function approximation in engineering mechanics applications. In: *2007 International Joint Conference on Neural Networks*. IEEE, 2007.

[110] J.-S. Pei, E. C. Mai, J. P. Wright, and S. F. Masri. Mapping some basic functions and operations to multilayer feedforward neural networks for modeling nonlinear dynamical systems and beyond. *Nonlinear Dynamics*, 71, 2013.

[111] J.-S. Pei, D. F. Hougen, S. T. Kanneganti, J. P. Wright, E. C. Mai, A. W. Smyth, S. F. Masri, A. Derkevorkian, F. Gay-Balmaz, and L. Komini. Interpretable machine learning for function approximation in structural health monitoring. In: A. Cury, D. Ribeiro, F. Ubertini, and M. D. Todd (eds.), *Structural Health Monitoring Based on Data Science Techniques*, pp. 369–388. Springer International Publishing, 2022.

[112] P. Petsagkourakis and F. Galvanin. Safe model-based design of experiments for nonlinear systems using Gaussian processes. *Computers & Chemical Engineering*, 151:107339, 2020. DOI: 10.1016/j.compchemeng.2021.107339.

[113] L. Pronzato and A. Pázman. *Design of Experiments in Nonlinear Models*, vol. 212 of *Lecture Notes in Statistics*. Springer, New York, 2013. DOI: 10.1007/978-1-4614-6363-4.

[114] G. Quaranta, W. Lacarbonara, and S. F. Masri. A review on computational intelligence for identification of nonlinear dynamical systems. *Nonlinear Dynamics*, 99 (2):1709–1761, 2020. DOI: https://doi.org/10.1007/s11071-019-05430-7.

[115] M. Raissi, P. Perdikaris, and G. E. Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational physics*, 378:686–707, 2019. DOI: 10.1016/j.jcp.2018.10.045.

[116] J. T. Rapp, B. J. Bremer, and P. A. Romero. Self-driving laboratories to autonomously navigate the protein fitness landscape. *Nature Chemical Engineering*, 1 (1):97–107, jan 2024. DOI: 10.1038/s44286-023-00002-4.

[117] C. E. Rasmussen and C. K. Williams. Gaussian processes for machine learning. *The MIT Press*, 2006.

[118] P. Ren, Y. Xiao, X. Chang, P.-Y. Huang, Z. Li, B. B. Gupta, X. Chen, and X. Wang. A survey of deep active learning. *ACM Computing Surveys*, 54(9):1–40, Oct. 2021. DOI: 10.1145/3472291.

[119] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, 1986. DOI: 10.1038/323533a0.

[120] J. Sacks, W. J. Welch, T. J. Mitchell, and H. P. Wynn. Design and analysis of computer experiments. *Statistical Science*, 4(4):409–423, 1989. DOI: 10.1214/ss/1177012413.

[121] S. M. S. Sadati, A. S. Nobari, and T. Naraghi. Identification of a nonlinear joint in an elastic structure using optimum equivalent linear frequency response function. *Acta Mechanica*, 223(7):1507–1516, 2012. DOI: 10.1007/s00707-012-0656-6.

[122] T. J. Santner, B. J. Williams, and W. I. Notz. *The Design and Analysis of Computer Experiments*. Springer Series in Statistics. Springer, New York, 2003. DOI: 10.1007/978-1-4757-3799-8.

[123] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini. The graph neural network model. *IEEE Transactions on Neural Networks*, 20(1):61–80, 2008. DOI: 10.1109/TNN.2008.2005605.

[124] B. Settles. *Active Learning Literature Survey*. Computer Sciences, Technical Report 1648. University of Wisconsin, Madison, Wisconsin, 2009.

[125] B. Settles. *Active Learning*. Number 18 in Synthesis lectures on artificial intelligence and machine learning. Morgan & Claypool, San Rafael, Calif., 2012. ISBN 978-1-60845-725-0. DOI: https://doi.org/10.2200/S00429ED1V01Y201207AIM018.

[126] H. S. Seung, M. Opper, and H. Sompolinsky. Query by committee. In: *Proceedings of the Fifth Annual Workshop on Computational Learning Theory (COLT)*, pp. 287–294. ACM, 1992. DOI: 10.1145/130385.130417.

[127] M. D. Shields, K. Gurley, R. Catarelli, M. Chauhan, M. Ojeda-Tuz, and F. J. Masters. Active learning applied to automated physical systems increases the rate of discovery. *Scientific Reports*, 13(1), May 2023. DOI: 10.1038/s41598-023-35257-7.

[128] N. H. Siddique and H. Adeli. *Computational intelligence: synergies of fuzzy logic, neural networks, and evolutionary computing*. John Wiley & Sons, Chichester, West Sussex, United Kingdom, 2013. ISBN 978-1-118-33784-4.

[129] I. M. Sobol'. On the distribution of points in a cube and the approximate evaluation of integrals. *USSR Computational Mathematics and Mathematical Physics*, 7(4):86–112, 1967. DOI: 10.1016/0041-5553(67)90144-9.

[130] S. Sony, S. Gamage, A. Sadhu, and J. Samarabandu. Vibration-based multiclass damage detection and localization using long short-term memory networks. *Structures*, 35:436–451, Feb. 2022. DOI: 10.1016/j.istruc.2021.10.088.

[131] N. Srinivas, A. Krause, S. M. Kakade, and M. Seeger. Gaussian process optimization in the bandit setting: No regret and experimental design. In: *Proceedings of the 27th International Conference on Machine Learning (ICML)*, pp. 1015–1022, 2010.

[132] A. Sánchez-González, J. Godwin, T. Pfaff, R. Ying, J. Leskovec, and P. Battaglia. Learning to simulate complex physics with graph networks. *International Conference on Machine Learning*, pp. 8459–8468, 2020.

[133] G. Taguchi. Introduction to quality engineering: Designing quality into products and processes. *Asian Productivity Organization*, 1986.

[134] V. Torczon. On the Convergence of Pattern Search Algorithms. *SIAM Journal on Optimization*, 7(1):1–25, Feb. 1997. DOI: 10.1137/S1052623493250780.

[135] H. Tu, X. Lin, Y. Wang, and H. Fang. System Identification for Lithium-Ion Batteries with Nonlinear Coupled Electro-Thermal Dynamics via Bayesian Optimization, 2024. URL `https://arxiv.org/abs/2405.20219`.

[136] N. Vaiana, S. Sessa, F. Marmo, and L. Rosati. A class of uniaxial phenomenological models for simulating hysteretic phenomena in rate-independent mechanical systems and materials. *Nonlinear Dynamics*, 93(3):1647–1669, Aug. 2018. DOI: 10.1007/s11071-018-4282-2.

[137] W. Wang and X. Zhang. On the suboptimality of gp-ucb under polynomial effective optimism, 2023. URL `https://arxiv.org/abs/2312.01386`.

[138] X. Wang, B. Ni, L. Zeng, and Y. Liu. An adaptive sampling strategy for construction of surrogate aerodynamic model. *Aerospace Science and Technology*, 112:106594, 2021. DOI: 10.1016/j.ast.2021.106594.

[139] S. M. Weiss and C. A. Kulikowski. *Computer Systems that Learn*. Morgan Kaufmann Publishers, San Francisco, 1991.

[140] K. Worden. *Nonlinearity in Structural Dynamics: Detection, Identification and Modelling*. CRC Press, Boca Raton, 2019. DOI: https://doi.org/10.1201/9780429138331.

[141] K. Worden, G. Manson, and N. Fieller. Damage detection using outlier analysis. *Journal of Sound and Vibration*, 229(3):647–667, 2000. DOI: 10.1006/jsvi.1999.2514.

[142] C. Wu, M. Zhu, Q. Tan, Y. Kartha, and L. Lu. A comprehensive study of non-adaptive and residual-based adaptive sampling for physics-informed neural networks. *arXiv:2207.10289*, 2022. DOI: https://doi.org/10.48550/arXiv.2207.10289.

[143] C. F. J. Wu and M. S. Hamada. *Experiments: Planning, Analysis, and Optimization*. John Wiley & Sons, Hoboken, NJ, 2 edition, 2011.

[144] T. Xiao, C. Park, L. Ouyang, and Y.-Z. Ma. An active learning Bayesian ensemble surrogate model for structural reliability analysis. *Quality and Reliability Engineering International*, 38(7):3579–3597, 2022. DOI: 10.1002/qre.3152.

[145] D. Xu, J. Lu, and Y. Bar-Shalom. Ensemble Gaussian process bayesian optimization for stochastic dynamical systems. *IEEE Transactions on Automatic Control*, 70(1): 412–427, 2025. DOI: 10.1109/TAC.2024.3384621.

[146] C. Yang. An adaptive sensor placement algorithm for structural health monitoring based on multi-objective iterative optimization using weight factor updating. *Mechanical Systems and Signal Processing*, 151:107363, 2021. DOI: https://doi.org/10.1016/j.ymssp.2020.107363.

[147] Y. Yang and M. Loog. A benchmark and comparison of active learning for logistic regression. *Pattern Recognition*, 83:401–415, Nov. 2018. DOI: 10.1016/j.patcog.2018.06.004.

[148] Y. Yuan, F. T. Au, D. Yang, and J. Zhang. Active learning structural model updating of a multisensory system based on kriging method and bayesian inference. *Computer-Aided Civil and Infrastructure Engineering*, 38(3):353–371, 2023. DOI: https://doi.org/10.1111/mice.12822.

[149] B. Zaparoli Cunha, C. Droz, A.-M. Zine, S. Foulard, and M. Ichchou. A review of machine learning methods applied to structural dynamics and vibroacoustic. *Mechanical Systems and Signal Processing*, 200, 2023. DOI: 10.1016/j.ymssp.2023.110535.

[150] W.-J. Zhang, K.-V. Yuen, and W.-J. Yan. Active learning aided bayesian nonparametric general regression for model updating using modal data. *Mechanical Systems and Signal Processing*, 204, 2023. DOI: 10.1016/j.ymssp.2023.110830.

[151] L. Zhao, Z. Li, B. Caswell, J. Ouyang, and G. E. Karniadakis. Active learning of constitutive relation from mesoscopic dynamics for macroscopic modeling of non-Newtonian flows. *Journal of Computational Physics*, 363:116–127, June 2018. DOI: 10.1016/j.jcp.2018.02.039.

[152] M. Zimmer, S. Moritz, O. Altay, and D. Abel. Hierarchical model predictive vibration control for high-rise structures using semi-active tuned liquid column damper. *Structural Control and Health Monitoring*, 29(10):e3034, 2022. DOI: 10.1002/stc.3034.

Schriftenreihe

des Lehrstuhls für Baustatik und Baudynamik der RWTH Aachen

01 (2013): Okyay Altay, Flüssigkeitsdämpfer zur Reduktion periodischer und stochastischer Schwingungen turmartiger Bauwerke, 2013, ISBN: 978-3-946090-01-4.

02 (2015): Francesca Taddei, Numerical Investigation of Soil-Structure Interaction for Onshore Wind Turbines Grounded on a Layered Soil, 2015, ISBN: 978-3-946090-00-7.

03 (2015): Wolfgang Dornisch, Interpolation of Rotations and Coupling of Patches in Isogeometric Reissner–Mindlin Shell Analysis, 2015, ISBN: 978-3-946090-02-1.

04 (2015): Konstantinos Mykoniou, Dynamic analysis of multiple liquid-storage tanks, 2015, ISBN: 978-3-946090-03-8.

05 (2015): Benedikt Kohlhaas, Ein Finite-Elemente-Modell zur Analyse des Verhaltens von Formgedächtnisfaserkompositen mit beliebiger Mikrostruktur, 2015, ISBN: 978-3-946090-04-5.

06 (2016): Lin Chen, Numerical Models for the Analysis of Soil, Structure and Their Interaction, 2016, ISBN: 978-3-946090-05-2.

07 (2016): Julia Rosin, Seismische Auslegung von Tankbauwerken, 2016, ISBN: 978-3-946090-06-9.

08 (2019): Sreelakshmy Rajan, Probabilistic Seismic Safety Analysis of Multicomponent Systems, 2019, ISBN: 978-3-946090-07-6.

09 (2020): Maximilian Praster, Entwicklung eines adaptiven FE2 Ansatzes zur Simulation von thermomechanisch beanspruchten Faser-Matrix-Kompositen, 2020, ISBN: 978-3-9460901-08-3.

10 (2020): Margarita Chasapi, Nonlinear Formulations and Coupling of Patches for Isogeometric Analysis of Solids in Boundary Representation, 2020, ISBN: 978-3-946090-09-0.

11 (2021): Philipp Michel, Boden-Bauwerk-Fluid-Interaktion flüssigkeitsgefüllter Tankbauwerke auf nachgiebigen, vielfach geschichteten Böden unter seismischer Einwirkung, 2021, ISBN: 978-3-946090-10-6.

12 (2021): Okyay Altay, Structural Vibration Control with Semi-Active and Shape Memory Alloy Based Systems, 2021, ISBN: 978-3-946090-11-3.

13 (2021): Niklas Boesen, Trag- und Verformungsverhalten von unbewehrten Mauerwerksscheiben unter Berücksichtigung der Interaktion mit der Gebäudestruktur, 2021, ISBN: 978-3-946090-12-0.

14 (2022): Simon Schleiter, Entwicklung einer numerischen Methode zur Identifikation von Bauwerk-Dämpfer-Systemparametern, 2022, ISBN: 978-3-946090-13-7.

15 (2022): Georgia Kikis, Locking and Brittle Fracture in Isogeometric Reissner-Mindlin Plate and Shell Analysis, 2022, ISBN: 978-3-946090-14-4.

16 (2022): Sven Klinkel, Simon Klarmann, Forschungskolloquium Baustatik-Baupraxis, 2022, ISBN: 978-3-946090-15-1.

17 (2022): Andreas Kaup, Constitutive Modeling of Superelastic Shape Memory Alloy Damping Considering Dynamic Effects, 2022, ISBN: 978-3-946090-16-8.

18 (2022): Thomas Kubalski, Modellierung von seismisch beanspruchten Mauerwerksausfachungen in Stahlbetonrahmentragwerken, 2022, ISBN: 978-3-946090-17-5.

19 (2023): Rainer Reichel, A polytope finite element formulation for the analysis of nonlinear mechanical behavior of composite materials, 2023, ISBN: 978-3-946090-18-2.

20 (2024): Leonie Mester, Computational Homogenisation and Multiscale Modelling Employing an Image-based Approach for the Structural Analysis of Shells, 2024, ISBN: 978-3-946090-19-9.

21 (2024): Arturo Méndez Salas, NURBS–based Perfectly Matched Layer Method in Transient Elastodynamics, 2024, ISBN: 978-3-946090-20-5.

22 (2025): Bjorn Sauren, Polytopal Mixed Finite Elements for Nearly–Incompressible Finite Elasticity: Discretization, Approximation, Stabilization, 2025, ISBN: 978-3-946090-21-2.

23 (2025): Niklas Lenzen, Artificial Neural Network-Assisted Parameter Identification and Response Modeling of Superelastic Shape Memory Alloys Subjected to Dynamic Loads, 2025, ISBN: 978-3-946090-22-9.

Lebenslauf

Name	Pavle Milicevic
Geburtsdatum	26.07.1994
Geburtsort	Ilidža, Bosnien und Herzegowina
2011 – 2013	Schulausbildung United World College, Mostar IB Diploma
2013 – 2017	Studium des Bauingenieurwesens The Universitz of Oklahoma, Norman Bachelor of Science
2018 – 2020	Studium des Bauingenieurwesens Rheinisch-Westfälische Technische Hochschule Aachen Master of Science
2020 – 2025	Wissenschaftlicher Mitarbeiter Lehrstuhl für Baustatik und Baudynamik Rheinisch-Westfälische Technische Hochschule Aachen