

Accelerated Multi-Objective Design Optimization of MEMS Using Surrogate Models

Ali Nazari

A Thesis submitted to the Faculty of Graduate Studies of
The University of Manitoba
in partial fulfilment of the requirements of the degree of

Master of Science

Department of Electrical and Computer Engineering
University of Manitoba
Winnipeg

Copyright © 2024 by Ali Nazari

To

My Mom,

Dad,

and Lovely Sisters, Fatemeh and Rezvan

for their presence in my life

Acknowledgements

I would like to thank my **mother, father, and sisters, Fatemeh** and Rezvan, who without their support it would have been impossible for me to finish my journey as a student. I want you to know that I am eternally indebted to you for your unconditional love, support, kindness, and sacrifices. Thank you!

I would like to thank **Phiona** who by providing valuable insights when I first started my master's, streamlined my familiarity with the project. She was always kind, thoughtful, and supportive and did her best to guide me in the right direction when I was at a stage where the whole project was still overwhelming to me.

I would like to thank National Research Council Canada (NRC) and University of Manitoba Graduate Fellowship (UMGF) for providing the financial support for this project.

Dr. Yunli Wang played a major role in providing invaluable technical insights where I encountered obstacles in my research path and for that, I am grateful.

Dr. Byoungyoul Park was one the most caring people I encountered while doing my study at the U of M. He was so kind as to provide engineering research direction for this project and for that, I am grateful.

The last three years when I was a student have been by far the most eventful period in my life as went through a rollercoaster of emotional, physical, and mental hardships. I was lucky to have my dear friends by my side who were always there for me. Dear **Ariyan, Milad, Javad, Armin, and Nazanin**, I deeply thank you for your company and presence by my side when I needed you the most.

When I was applying for a master's position, I would have never imagined that my supervisor, **Dr. Cyrus Shafai** would be one of the (if not THE) nicest, caring, and knowledgeable people I have ever met in my life. I am lucky, honoured and forever grateful to have been one of your students. If 20 years from now I find myself in a position where I feel like I have become someone like you, at that moment, I would simply smile. THANK YOU CYRUS!

Abstract

This study presents a comprehensive multi-objective optimization framework specifically designed for micro-electromechanical systems (MEMS). The framework integrates both traditional and adaptive optimization techniques, named Surrogate-Assisted Multi-Objective Optimization (SAMOO) and Adaptive-SAMOO (A-SAMOO), respectively. These optimization schemes were tested on multiple MEMS devices with varying physics and complexities, demonstrating their robustness and versatility, particularly in handling cases with discrete design variables and strict objective constraints.

Additionally, the study addresses several shortcomings of traditional approaches such as considering objective constraints and providing multiple design options for users, which enhances the flexibility and practical application of the results. The importance of preprocessing objectives is also emphasized. Results show that adaptive optimization outperforms traditional offline methods, delivering a greater number and higher quality of optimal solutions. This performance boost highlights the advantages of online learning in enhancing the accuracy, speed, and diversity of solutions in MEMS optimization. This detailed, step-by-step framework serves as a valuable guide for researchers and practitioners looking to optimize MEMS designs from scratch, offering a more reliable and effective approach to multi-objective optimization in MEMS applications.

Contents

List of Abbreviations	xviii
1 Introduction	1
1.1 Motivation	1
1.2 Shortcomings of Current Approaches	3
1.3 Contribution of This Study	4
1.4 Thesis Structure	8
2 Background	9
2.1 General Surrogate-Assisted Optimization Procedure	9
2.2 Previous Studies	13
2.3 Summary	20
3 Actuators	22
3.1 U-Shaped Lorentz Force Actuator	22

3.2	Lorentz Force Actuator with Springs	26
3.3	Thermal Actuator	29
3.4	Summary	32
4	Methodology	34
4.1	Surrogate-Assisted Multi-Objective Optimization (SAMOO)	35
4.1.1	Initial Sampling of the Design Space	37
4.1.2	FEM Modelling	39
4.1.3	Preprocessing	44
4.1.4	Surrogate Modelling	45
4.1.5	Optimization	49
4.2	A-SAMOO	57
4.2.1	Overview of the Optimization Procedure	57
4.2.2	Optimization	58
4.3	Parameters	63
4.4	Summary	64
5	U-Shaped Lorentz Force Actuator Results	66
5.1	Initial Sampling	66
5.2	Surrogate Modelling	67
5.3	Optimization	70
5.4	Summary	76
6	Lorentz Force Actuator with Springs Results	78

6.1	Initial Sampling	78
6.2	Surrogate Modelling	80
6.3	Optimization	84
6.4	Summary	88
7	Thermal actuator results	90
7.1	Initial Sampling	90
7.2	Surrogate Modelling	91
7.3	Optimization	94
7.4	Summary	99
8	Conclusion	100
8.1	Achieved Aims and Objectives	100
8.2	Limitations	103
Appendix A	Deliverables	105
A.1	Publication	105
A.2	ai4d Python Package	106
Appendix B	Supplementary Figures	108
B.1	U-Shaped Lorentz Force Actuator	108
B.2	Lorentz Force Actuator with Springs	115
B.3	Thermal Actuator	131
	References	141

List of Figures

2.1	The trade-off between fidelity (accuracy) and computation cost of different function evaluation methods	12
2.2	Two different CNN surrogate model input images used in the work of Di Barbra and colleagues.	14
2.3	Finite element meshing of the comb-drive electrostatic actuator optimized by Chereches and colleagues.	15
2.4	Schematics of actuators optimized using ASDEMO algorithm	17
2.5	Schematic of the MEMS capacitive accelerometer optimized by Saghir and colleagues	18
2.6	Schematics of the RF-MEMS suspended inductor	19
2.7	Schematics of RF-MEMS cantilever-based series switch	20
3.1	Schematic of the U-shaped Lorentz force actuator.	23
3.2	Schematic of the Lorentz force actuator with springs.	27
3.3	Thermal actuator schematics	30

4.1	Flowchart of the SAMOO algorithm	36
4.2	An example of LHS sampling in a three-dimensional design space	38
4.3	Flowchart of NSGA-II.	51
4.4	Pareto front of a car purchase problem	54
4.5	Visualization of different points associated with a MOO problem	56
4.6	Flowchart of A-SAMOO	60
5.1	MAPE of surrogate models with different training sizes and preprocessing for the U-shaped Lorentz force actuator	68
5.2	Actual vs predicted plot for the U-shaped Lorentz force actuator with a training size of 200	70
5.3	Actual vs predicted plot for the U-shaped Lorentz force actuator with a training size of 200: After SAMOO optimization	72
5.4	Actual vs predicted plot for the U-shaped Lorentz force actuator with a training size of 200: After A-SAMOO optimization	73
5.5	MAPE after optimization for the U-shaped Lorentz force actuator	74
5.6	Number of true optimal solutions for the U-shaped Lorentz force actuator . . .	75
5.7	Hypervolume of predicted solutions for the U-shaped Lorentz force actuator .	76
6.1	MAPE of surrogate models with different training sizes and preprocessing for the Lorentz force actuator with springs	80
6.2	Actual vs predicted plot (log scale) for the Lorentz force actuator with springs with a training size of 200	82

6.3	Actual vs predicted plot for the Lorentz force actuator with springs with a training size of 200	83
6.4	Actual vs predicted plot for the Lorentz force actuator with springs with a training size of 200: After SAMOO optimization	84
6.5	Actual vs predicted plot for the Lorentz force actuator with springs with a training size of 200: After A-SAMOO optimization	85
6.6	MAPE after optimization for the Lorentz force actuator with springs	86
6.7	Number of true optimal solutions for the Lorentz force actuator with springs .	87
6.8	Hypervolume of predicted solutions for the Lorentz force actuator with springs	88
7.1	MAPE of surrogate models with different training sizes and preprocessing for the Thermal actuator	92
7.2	Actual vs predicted plot (log scale) for the Thermal actuator with a training size of 200	93
7.3	Actual vs predicted plot for the Thermal actuator with a training size of 200: After SAMOO optimization.	94
7.4	Actual vs predicted plot for the Thermal actuator with a training size of 200: After A-SAMOO optimization.	95
7.5	Mean Absolute Percentage Error (MAPE) after optimization for the Thermal actuator	96
7.6	Number of true optimal solutions for the Thermal actuator	97
7.7	Hypervolume of predicted solutions for the Thermal actuator	98
A.1	ai4d Python package developed for this project	107

B.1	Actual vs predicted plot for the U-shaped Lorentz force actuator with a training size of 20.	108
B.2	Actual vs predicted plot for the U-shaped Lorentz force actuator with a training size of 20: After Adaptive (A-SAMOO) Optimization.	109
B.3	Actual vs predicted plot for the U-shaped Lorentz force actuator with a training size of 20: After Regular (SAMOO) Optimization.	109
B.4	Actual vs predicted plot for the U-shaped Lorentz force actuator with a training size of 50.	110
B.5	Actual vs predicted plot for the U-shaped Lorentz force actuator with a training size of 50: After Adaptive (A-SAMOO) Optimization.	110
B.6	Actual vs predicted plot for the U-shaped Lorentz force actuator with a training size of 50: After Regular (SAMOO) Optimization.	111
B.7	Actual vs predicted plot for the U-shaped Lorentz force actuator with a training size of 100.	111
B.8	Actual vs predicted plot for the U-shaped Lorentz force actuator with a training size of 100: After Adaptive (A-SAMOO) Optimization.	112
B.9	Actual vs predicted plot for the U-shaped Lorentz force actuator with a training size of 100: After Regular (SAMOO) Optimization.	112
B.10	Actual vs predicted plot for the U-shaped Lorentz force actuator with a training size of 400.	113
B.11	Actual vs predicted plot for the U-shaped Lorentz force actuator with a training size of 400: After Adaptive (A-SAMOO) Optimization.	113

B.12 Actual vs predicted plot for the U-shaped Lorentz force actuator with a training size of 400: After Regular (SAMOO) Optimization.	114
B.13 Actual vs predicted plot for the Lorentz force actuator with springs with a training size of 20.	115
B.14 Actual vs predicted plot (log scaled) for the Lorentz force actuator with springs with a training size of 20.	116
B.15 Actual vs predicted plot for the Lorentz force actuator with springs with a training size of 20: After Adaptive (A-SAMOO) Optimization.	117
B.16 Actual vs predicted plot for the Lorentz force actuator with springs with a training size of 20: After Regular (SAMOO) Optimization.	118
B.17 Actual vs predicted plot for the Lorentz force actuator with springs with a training size of 50.	119
B.18 Actual vs predicted plot (log scaled) for the Lorentz force actuator with springs with a training size of 50.	120
B.19 Actual vs predicted plot for the Lorentz force actuator with springs with a training size of 50: After Adaptive (A-SAMOO) Optimization.	121
B.20 Actual vs predicted plot for the Lorentz force actuator with springs with a training size of 50: After Regular (SAMOO) Optimization.	122
B.21 Actual vs predicted plot for the Lorentz force actuator with springs with a training size of 100.	123
B.22 Actual vs predicted plot (log scaled) for the Lorentz force actuator with springs with a training size of 100.	124

B.23 Actual vs predicted plot for the Lorentz force actuator with springs with a training size of 100: After Adaptive (A-SAMOO) Optimization.	125
B.24 Actual vs predicted plot for the Lorentz force actuator with springs with a training size of 100: After Regular (SAMOO) Optimization.	126
B.25 Actual vs predicted plot for the Lorentz force actuator with springs with a training size of 400.	127
B.26 Actual vs predicted plot (log scaled) for the Lorentz force actuator with springs with a training size of 400.	128
B.27 Actual vs predicted plot for the Lorentz force actuator with springs with a training size of 400: After Adaptive (A-SAMOO) Optimization.	129
B.28 Actual vs predicted plot for the Lorentz force actuator with springs with a training size of 400: After Regular (SAMOO) Optimization.	130
B.29 Actual vs predicted plot for the thermal actuator with a training size of 20. . . .	131
B.30 Actual vs predicted plot for the thermal actuator with a training size of 50. . . .	132
B.31 Actual vs predicted plot for the thermal actuator with a training size of 50: After Adaptive (A-SAMOO) Optimization.	133
B.32 Actual vs predicted plot for the thermal actuator with a training size of 50: After Regular (SAMOO) Optimization.	134
B.33 Actual vs predicted plot for the thermal actuator with a training size of 100. . . .	135
B.34 Actual vs predicted plot for the thermal actuator with a training size of 100: After Adaptive (A-SAMOO) Optimization.	136
B.35 Actual vs predicted plot for the thermal actuator with a training size of 100: After Regular (SAMOO) Optimization.	137

B.36	Actual vs predicted plot for the thermal actuator with a training size of 400. . .	138
B.37	Actual vs predicted plot for the thermal actuator with a training size of 400: After Adaptive (A-SAMOO) Optimization.	139
B.38	Actual vs predicted plot for the thermal actuator with a training size of 400: After Regular (SAMOO) Optimization.	140

List of Tables

1.1	Comparison of this work with existing literature	7
3.1	Design variables of the U-Shaped Lorentz force actuator	25
3.2	Optimization objectives of the U-shaped Lorentz force actuator	26
3.3	Design variables of the Lorentz force actuator with springs	28
3.4	Optimization objectives of the Lorentz force actuator with springs	29
3.5	Design variables of the thermal actuator	31
3.6	Optimization objectives of the thermal actuator	32
4.1	The parameters used for modelling of Lorentz force actuators in COMSOL Multiphysics	43
4.2	NSGA-II Parameters	64
5.1	Objective ranges and computation times of initial samples for U-shaped Lorentz force actuator	67

6.1	Objective ranges and computation times of initial samples for Lorentz force actuator with springs	79
7.1	Objective ranges and computation times of initial samples for Thermal actuator	91

List of Abbreviations

R² coefficient of determination

A-SAMOO Adaptive Surrogate-Assisted Multi-Objective Optimization

ANN Artificial neural network (ANN)

API application programming interface

ASDEMO self-adaptive surrogate model-assisted differential evolution for MEMS optimization

CCECE Canadian Conference on Electrical and Computer Engineering

CFD computational fluid dynamics

CNN convolutional neural network

DE Differential Evolution

EA evolutionary algorithm

FEM finite element method

GEKPLS Gradient-enhanced Kriging Partial Least Squares

GP Gaussian process

GPR Gaussian process regression

KPLS Kriging Partial Least Squares

LARS least angle regression

LHS Latin hypercube sampling

MAE mean absolute error

MAPE mean absolute percentage error

MEMS micro-electromechanical systems

MOO multi-objective optimization

NSGA-II Non-dominated Sorting Genetic Algorithm II

QI quality indicator

RANSAC Random Sample Consensus

RF-MEMS radio frequency micro-electromechanical systems

RMSE root mean squared error

SAMOO Surrogate-Assisted Multi-Objective Optimization

SAO surrogate-assisted optimization

SBO surrogate-based optimization

SQP sequential quadratic programming

Introduction

1.1 | Motivation

Micro-electromechanical systems (MEMS) have become indispensable in various devices such as biomedical apparatus [1, 2], wearables [3], lasers [4], robots [5, 6], autonomous vehicles [7], drones [8], smartphones [9], and telescopes [10, 11], owing to their miniature size, energy efficiency, high sensitivity, and ease of integration and control [12]. These systems are built using semiconductor manufacturing techniques and can perform sensing, actuation, or processing tasks, often in a compact and efficient form. A common real-world example of MEMS is the accelerometer in smartphones. This tiny device detects the orientation and movement of the phone, enabling features like screen rotation, step counting, and shake-to-wake functionality. It works by measuring the forces acting on a small, movable mass inside the sensor.

To maximize MEMS performance, design parameter optimization is generally conducted before fabrication. However, the increasing complexity of modern MEMS

devices often challenges the capabilities of traditional design expertise, making a robust optimization process imperative.

Evolutionary algorithms (EAs) can be used to explore the design space, with finite element method (FEM) analyses employed to derive objective values [13, 14]. Many optimization algorithms demand a large number of evaluations [15, 16]. The number of samples required to optimize a MEMS device is highly dependent on the number of design variables and the complexity of the design. As the number of design variables increases or the design becomes more intricate, the search space grows exponentially, necessitating a significantly larger number of samples to explore it adequately.

This often results in the need for thousands of FEM analyses to achieve a satisfactory optimization outcome. However, conducting such a large number of FEM simulations is computationally prohibitive, as each simulation can be time-intensive. Even if a single MEMS simulation take only a few minutes [17], evaluating thousands of designs could take days or even weeks, rendering such optimization procedure impractical. For such problems, it is essential to employ an optimization approach that can find optimal solutions using only a limited number of expensive evaluations. Therefore, improving the efficiency of the design and optimization process is an important research focus in the MEMS community.

One of the primary approaches to address this issue is to replace the computationally expensive function with a more efficient one that approximates the original problem with acceptable accuracy. This substitute is referred to as a "surrogate model" or "meta model" in the literature. An optimization procedure that employs this method is known as "surrogate-assisted optimization (SAO)" or "surrogate-based optimization (SBO)" [18, 19].

1.2 | Shortcomings of Current Approaches

Despite significant advancements in application of SBO to MEMS design optimization [20–22], several shortcomings remain to be addressed:

- **Single-objective focus:** Many studies focus on single-objective optimization, which does not address the complexity of integrating multiple design objectives [23–25].
- **Static Surrogate models:** Traditional SBO methods often employ static surrogate models, meaning the surrogate model is trained only once and remains unchanged as the optimization progresses. This approach can result in poor accuracy of the model in the most favorable regions of the design space, which in turn can negatively impact the quality and precision of the optimization results.
- **Limited preprocessing discussion:** The impact of preprocessing design variables and objectives on the accuracy of surrogate predictions and optimization results is seldom addressed.
- **Inadequate Constraint Handling:** Many MEMS SAO frameworks fail to adequately manage constraints imposed by specific application requirements [26, 27].
- **Computational Intensity:** Despite using surrogates, some approaches still face high computational costs due to the need for extensive initial simulations to build accurate models.

1.3 | Contribution of This Study

In this study, surrogate-assisted multi-objective optimization was conducted by utilizing a limited number of simulations to train surrogate models with datasets of varying sizes and preprocessing methods. An evolutionary algorithm is then applied to identify the best design configurations, ensuring that optimization objectives remain within predefined acceptable ranges. To further enhance the optimization process, adaptive optimization was incorporated, continuously refining the surrogate models for improved accuracy.

These methods were applied to three different MEMS devices:

1. **U-shaped Lorentz-force actuator:** A MEMS device that generates vertical deflection by creating a Lorentz force when current flows through its U-shaped conductor in the presence of a magnetic field.
2. **Lorentz-force actuator with springs as arms:** A variation of the Lorentz-force actuator where flexible spring-like arms provide additional control over the characteristics of the device.
3. **Thermal actuator:** A MEMS device that produces motion through differential thermal expansion, where resistive heating causes one arm to expand more than another, resulting in bending and hence, generating a deflection.

The primary contribution of this study lies in the development and validation of a generalized multi-objective optimization method suitable for diverse MEMS applications. Specifically, the contribution contributions include:

1. **Integration of adaptive optimization:** By incorporating adaptive optimization, the surrogate models are dynamically updated during the optimization process. This continuous refinement aims to increase the accuracy of the surrogate models, reduces computational overhead, and accelerate convergence to optimal solutions. The adaptive approach ensures that the models remain relevant and precise as new data is integrated, effectively addressing the limitations of static surrogate models.
2. **Validation across multiple devices:** The developed optimization methods are tested on various MEMS devices to ensure the robustness and versatility of those approaches, indicating that they can be effectively utilized for a wide range of MEMS designs.
3. **Providing a set of optimal solutions:** In this work, since we conduct multi-objective optimization, the goal is not to identify a single optimal design but to generate a set of non-dominated solutions. These solutions represent the optimal trade-offs between conflicting objectives, allowing users to explore and select the design that best aligns with their specific requirements or preferences. In MOO, there is no single "best" solution because improving one objective inevitably requires compromising on others. Instead, we provide a diverse set of optimal solutions, empowering users to make informed decisions based on their unique priorities and application needs. This approach ensures flexibility and adaptability in choosing the most suitable design.
4. **Addressing shortcomings of traditional approaches:** This study addresses several key shortcomings of traditional SBO methods:

- **Multi-objective optimization:** Unlike many existing studies that focus on single-objective optimization [21, 23, 25], the approach provided in this study considers multiple design objectives, capturing the complexity and trade-offs inherent in MEMS design.
 - **Preprocessing of design variables and objectives:** The impact of preprocessing objectives on the accuracy of surrogate predictions and optimization result are explored. This aspect is often overlooked in existing research, yet it is crucial for enhancing the performance and reliability of the surrogate models and hence, optimization process.
 - **Constraint handling:** This framework incorporates constraint handling mechanisms to ensure that the optimization process adheres to the specific requirements and limitations of each MEMS application.
5. **Efficient computational strategies:** Through the use of surrogate models and adaptive optimization, the computational intensity typically associated with FEM evaluations is significantly reduced, making the optimization process more efficient.
6. **Comprehensive Framework:** The methods developed in this study form a comprehensive optimization framework that can be generalized to various MEMS applications, making it a valuable tool for MEMS designers.

Table 1.1 provides a comparison of the present study with the existing literature.

Table 1.1: Comparison of this work with existing literature

Article	# Features	# Objectives	Surrogate	Optimizer	Optimized Device	Notes
[28]	3	2	CNN	NSGA-II	Electrostatic micromotor	Image inputs to surrogate
[20]	4	2	Kriging	NSGA-II	Comb-drive	Cross-validation of surrogate
[23]	4	1	Kriging	DE	Calorimetric wall shear stress sensor	Results verified with a redundant optimizer
[21]	9	1	Kriging	DE	Shell-type dome actuator, corrugated membrane actuator	Promising regions highly populated
[25]	7	1	Kriging	Customized DE	Corrugated membrane actuator	Use of derivatives in initial sampling by sequential quadratic programming (SQP), constraints handled by a penalty factor
[24]	4, 10	1, 1	Kriging	DE	Thermal actuator, corrugated membrane actuator	Weighted sum of 2 constraint violations serves as the objective.
[27]	5	5	Kriging	Desirability function approach, Gradient descent	Capacitive accelerometer	Multi-objective optimization, sensitivity analysis is conducted, objective constraints are handled, does not intend to provide the user with multiple choices
[22]	4	3	Regression tree	NSGA-II	RF-MEMS suspended inductor, RF-MEMS switch	Multi-objective optimization, provides user with multiple Pareto-optimal solutions to choose from
This work	6, 7, or 8	3, 5, 6	Gaussian process regression (Kriging)	NSGA-II	U-shaped Lorentz force actuator, Lorentz force actuator with spring arms, Thermal actuator	Multi-objective, adaptive-optimization, provides the user with multiple choices (a non-dominated solution set), constraint handling, study of data pre-processing

1.4 | Thesis Structure

The structure of the remaining sections is as follows: Chapter 2 provides a review of previous studies on surrogate-assisted design optimization of MEMS. Chapter 3 discusses the operational principles of the actuators and defines the problems. Chapter 4 details methodology used in this work. Chapters 5, 6, and 7 present the results and discuss the findings for U-shaped Lorentz force, Lorentz force with springs, and thermal actuator, respectively. Chapter 8 summarizes the key findings and provides detail on some of the limitation of the present study. Project deliverables are mentioned in Appendix A. Lastly, Appendix B includes supplementary figures whose placement within the main body of the thesis would otherwise disrupt its flow.

Background

This chapter provides a review of previous works on surrogate-assisted optimization of MEMS. First, an overview of a general surrogate-assisted multi-objective optimization procedure will be introduced. Then, several examples of using SAO for MEMS design optimization will be provided along with their notable contributions and limitation.

2.1 | General Surrogate-Assisted Optimization Procedure

Optimization problems often involve balancing multiple objectives that may conflict with one another. For instance, consider the design of a car: engineers might aim to minimize fuel consumption while simultaneously maximizing safety features. Achieving one objective, such as improved fuel efficiency, might involve using lighter materials, which could compromise the car's structural strength and safety. This trade-off is a hallmark of multi-objective optimization, where the goal is to identify solutions that strike a balance between competing objectives.

To formalize this, a general MOO framework defines objectives, constraints, and design variables, enabling the systematic evaluation of potential solutions. Mathematically, a general multi-objective optimization procedure can be defined as

$$\begin{aligned}
 &\text{Minimize } f_m(\mathbf{x}), \quad m = 1, 2, \dots, M; \\
 &\text{subject to:} \\
 &\quad g_j(\mathbf{x}) \geq 0, \quad j = 1, 2, \dots, J; \\
 &\quad h_k(\mathbf{x}) = 0, \quad k = 1, 2, \dots, K; \\
 &\quad x_i^{(L)} \leq x_i \leq x_i^{(U)}, \quad i = 1, 2, \dots, n.
 \end{aligned} \tag{2.1}$$

in which M , J , K , and n represent the number of objectives, inequality constraints, equality constraints, and design variables, respectively [29]. A solution x in the design space, $\mathbb{R}^n$, is considered "feasible" if it satisfies both the inequality and equality constraints and falls within the specified bounds. For each solution x , there exists a corresponding point $\mathbf{y} = \mathbf{f}(\mathbf{x}) \in \mathbb{R}^M$ in the objective space.

Throughout this work, the terms "solution" and "design" will be used interchangeably to refer to any point in the design space, x . The goal of the optimization process is to identify these feasible points that satisfy the constraints and minimize (or maximize) the objectives. In the MOO formulation, usually only minimization is considered. This is due to the fact that the equations for maximization can be simply obtained by multiplying objectives by -1 and perform a similar transformation for inequality constraints [30].

Evolutionary algorithms (EA) have been widely used for optimization of computationally expensive functions in different areas such as building design and

performance analysis [31], electrical machines [32], aircraft design [33], radio frequency (RF) [34], and many other engineering applications [35, 36]. The appeal of EAs lies in their versatility in solving MOO problems, as they can handle diverse variable types (continuous, combinatorial, or mixed) without assumptions on constraints, convexity or differentiability of the objectives [37, 38]. MEMS design optimization literature is no exception and many studies have relied on EAs [13, 39–41].

During a general optimization procedure, the performance of different solutions is assessed through "fitness function evaluations" $f(\mathbf{x})$. Based on the information obtained from these evaluations, the optimizer advances in a direction it believes will yield optimal solutions. One of the primary difficulties in solving MOO problems is that the optimization process can require a substantial number of function evaluations [37].

In many applications, such as computational fluid dynamics (CFD) [33] and computational electromagnetics [42], the complexity of the problem and hence the high computational cost make evaluating a large number of designs impractical.

As mentioned in chapter 1, surrogate models can address this issue. Surrogate models are constructed from a limited set of FEM simulations and intend to replicate the behaviour of more complex systems. For most practical purposes, in the context of optimization, it is just another term for regression models. These models allow rapid predictions of objective values without the need for full simulations, thereby significantly reduce compute times, as will be discussed in chapter 5, chapter 6, and chapter 7.

The trade-off between accuracy and computational time for various models used in optimization is illustrated in Figure 2.1 [18]. As one would expect, faster models generally exhibit lower accuracy in approximating the original model. This trade-off is governed by

the available computational budget and the acceptable level of modelling error for a given application.

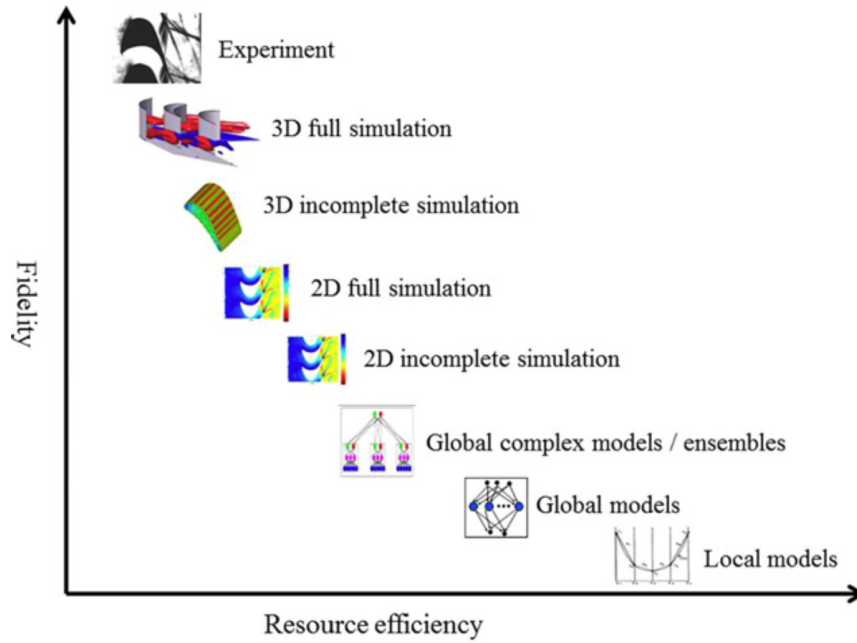


Figure 2.1: The trade-off between fidelity (accuracy) and computation cost of different function evaluation methods [18].

Gaussian process regression (GPR) models are the most commonly used function estimation techniques in the context of surrogate-assisted optimization. GPR is a machine learning method used for predicting objectives and modelling uncertainty. It assumes that data points follow a Gaussian distribution and builds a smooth predictive model by learning from known data while providing confidence intervals for predictions.

Unlike many traditional regression techniques that provide only point estimates for predictions, GPR offers a unique advantage: it provides not only predictions for a given

input but also a measure of uncertainty for the prediction, typically represented as a variance or standard deviation. This uncertainty is derived from the properties of the Gaussian process itself and the data available for training. This dual capability makes GPR particularly valuable for applications where understanding the confidence in predictions is as important as the predictions themselves. Their ability to provide a reasonable function estimation when trained on a small dataset is well-suited for applications where obtaining a large number of training samples is not feasible. Surrogate-assisted optimization falls into that category.

In the literature sometimes GP regression is used interchangeably with Kriging [43] although there are some differences between the two that is mostly related to the assumptions on distribution of data being modelled [44]. The origin of Kriging traces back to geostatistics and as such it is often used in geology while GP regression is a more common approach in machine learning [45]. In this work, GPR and Kriging are used interchangeably.

2.2 | Previous Studies

Using combination of terms "surrogate", "MEMS", and "optimization" various previous works were identified. Gathering a list of all related studies is not an easy task as different terminologies are used in the literature. Also, there is not clear boundary for definition of surrogate itself. As a result, the following constraints were considered to define a clear boundary for works that have conducted SBO for MEMS design:

- There should be clear mention of MEMS. Although some devices are technically MEMS it does not seem rational to consider them as such if the authors do not.
- The surrogate model must refer to a regression model that is constructed from FEM. There are MEMS design optimization studies that try to replace computationally expensive FEM with simpler mathematical models. However, since the models do not require FEM samples for surrogate model construction the respective studies are not considered.

Di Barba and colleagues used Non-dominated Sorting Genetic Algorithm II (NSGA-II) [46] optimizer coupled with convolutional neural network (CNN) surrogate model for bi-objective optimization of a MEMS micro-motor. The inputs of the surrogate model were images of a segment of the rotor. Two different input images are provided in Figure 2.2.

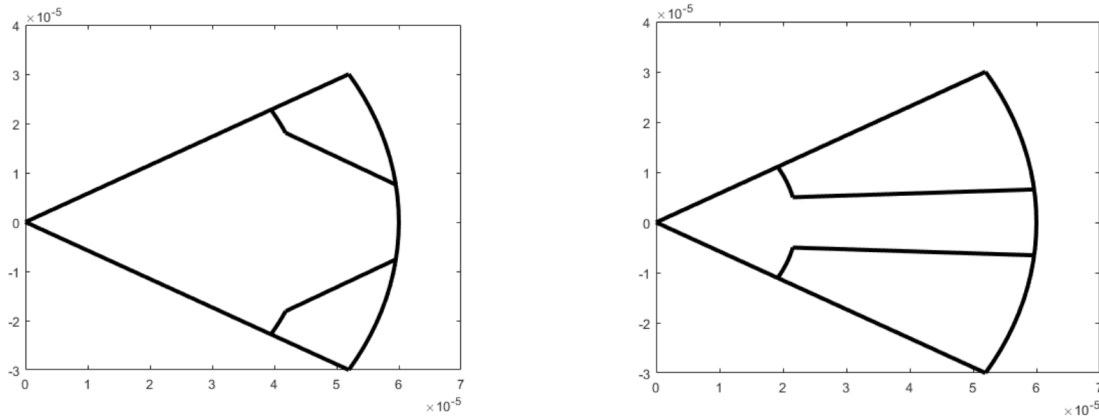


Figure 2.2: Two different CNN surrogate model input images used in the work of Di Barbra and colleagues [28]. Dimensions are in m .

The study considers three design variables. Although the use of surrogate model has reduced the number of required FEM analyses by 88%, the number of FEM evaluations

used for surrogate training that is 1200 is relatively large. This is mostly due to the fact that neural networks generally require more training data compared to GPR.

Giehler and colleagues [23] conducted surrogate-assisted design optimization for MEMS calorimetric wall shear stress sensors. 32 initial FEM evaluations were used to train an Ordinary Kriging [47] surrogate model. During the optimization process utilizing Differential Evolution (DE) [48], additional sampling was performed, resulting in a total of 221 samples. The surrogate predictions were evaluated against both FEM models and experimental results, showing a high degree of accuracy. This study focused on optimizing a single objective.

Design optimization of a comb-drive electrostatic MEMS micro-actuator depicted in Figure 2.3 was performed by Chereches and colleagues [20].

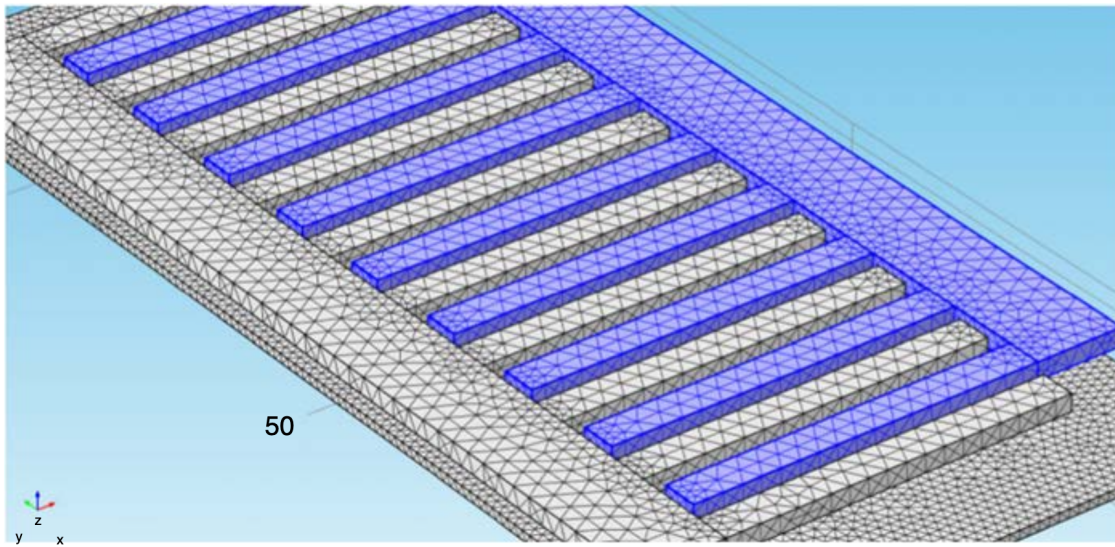


Figure 2.3: Finite element meshing of the comb-drive electrostatic actuator optimized by Chereches and colleagues [20].

Two objectives were optimized by finding optimal values of four design variable

using the NSGA-II optimizer and Kriging surrogate model. No equality or inequality constraints were set for objectives. The utilization of surrogate models resulted in a reduction of compute time to just half of that observed in the previous work of authors [49], which did not incorporate any surrogate models.

Akinsolu and colleagues [24] introduced a surrogate-assisted optimization method specifically for MEMS called self-adaptive surrogate model-assisted differential evolution for MEMS optimization (ASDEMO). The performance of this method was tested on various benchmark test problems as well as two MEMS designs, thermal and corrugated membrane actuators illustrated in Figure 2.4.

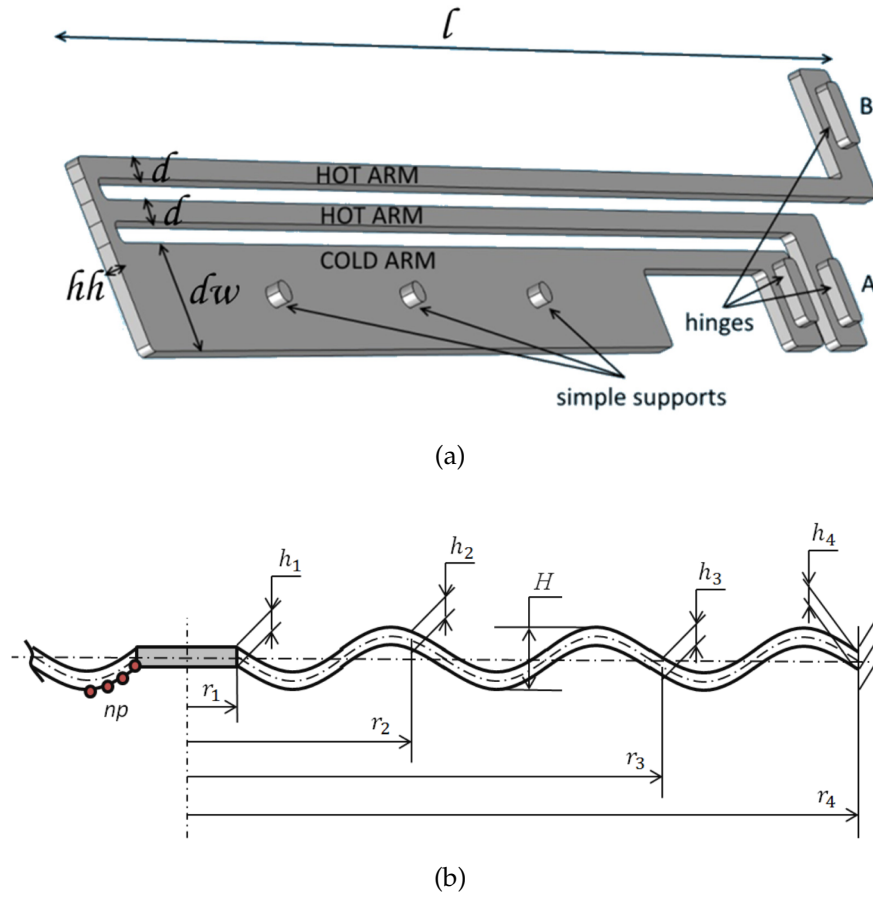


Figure 2.4: Schematics of the (a) thermal actuator and (b) corrugated membrane actuator optimized using ASDEMO algorithm [24].

The DE optimizer and the Kriging surrogate model employed in this study were utilized to optimize a single objective, which was calculated by a weighted sum of two objective constraints. Meaning that although two objectives were being optimized simultaneously, the optimization procedure itself was inherently single-objective; hence, intricacies of multi-objective problems are not considered.

Saghir and colleagues performed MOO for a 2-degree-of-freedom MEMS capacitive

accelerometer depicted in Figure 2.5.

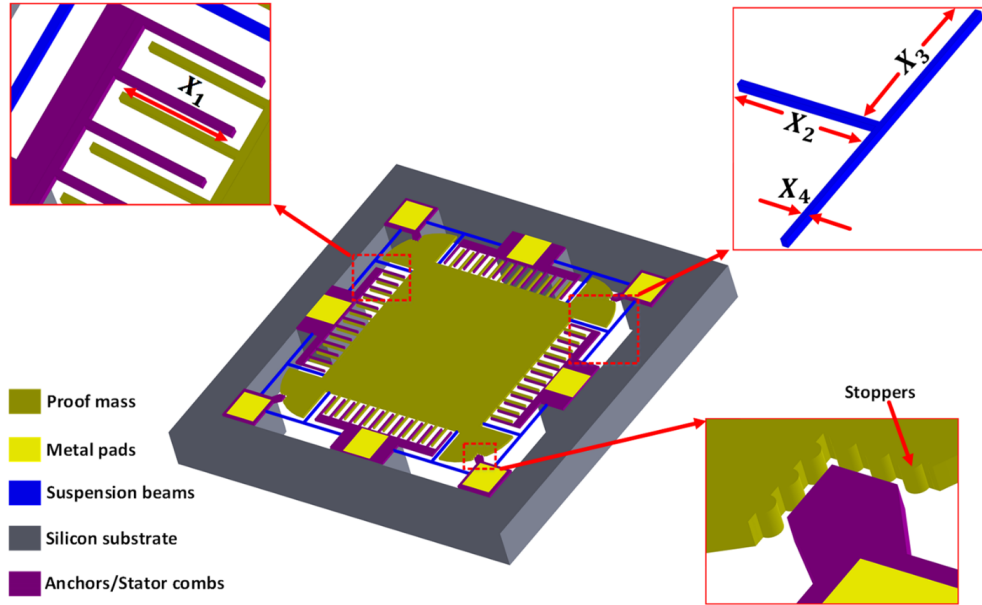


Figure 2.5: Schematic of the MEMS capacitive accelerometer optimized by Saghir and colleagues [27].

First, the FEM simulations are conducted to train GPR surrogates. Then, using a desirability function [50] each predicted objective is mapped to a value called desirability that ranges between 0 and 1, correspond to the least and most optimal values, respectively. Geometric mean of all desirability values is then calculated which serves as the optimization objective for a gradient descent optimizer. As a result, the optimizer only deals with a scalar value; meaning that after the optimization procedure only the single best design would be available. This contrasts to true multi-objective optimization procedures in which after optimization the user is provided with multiple design to choose from based on their preferences.

Multi-objective design optimization of two RF-MEMS devices, suspended ring inductor (Figure 2.6) and switch (Figure 2.7), was reported by Bajwa and colleagues [22]. The surrogate model and optimizer used in this work were regression trees [51] and NSGA-II, respectively. Compared to most studies mentioned in this chapter, this is a comprehensive study as it performs true multi-objective optimization and includes constraint handling. However, the surrogate models used here are static; meaning that they are trained only once and remain the same during the optimization procedure.

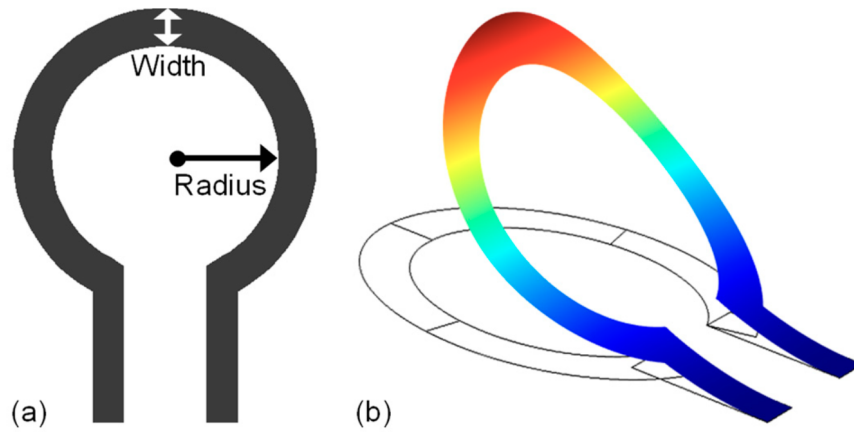


Figure 2.6: Schematics of the RF-MEMS suspended inductor [22]: (a) top view under planar condition, and (b) under deformation.

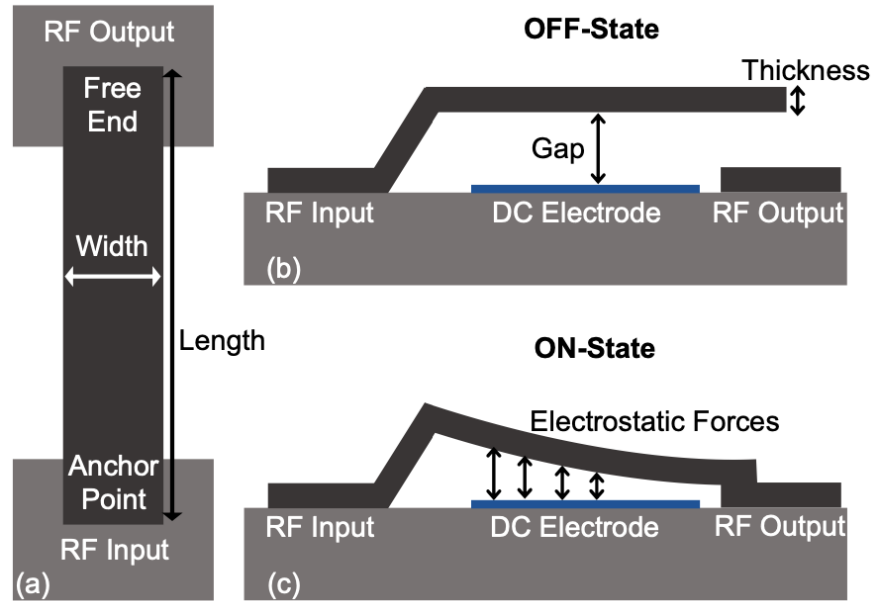


Figure 2.7: Schematics of RF-MEMS cantilever-based series switch [22] : (a) top view, (b) side view in OFF-state, and (c) side view in ON-state.

2.3 | Summary

This chapter presents a comprehensive review of surrogate-assisted optimization (SAO) methods for MEMS design. It begins by outlining a general multi-objective optimization framework, detailing how surrogate models can significantly reduce computational costs for evaluating complex MEMS designs. In SAO, surrogate models serve as efficient approximations, often using regression techniques like Gaussian process regression (Kriging) to predict objectives based on a limited number of FEM simulations. Evolutionary algorithms, such as NSGA-II and DE, are commonly integrated with these surrogates, allowing rapid optimization across various engineering domains, including MEMS.

Several applications of SAO in MEMS, such as micro-motors, actuators, sensors, and RF devices, are discussed in terms of their methodologies, strengths, and limitations. Although each case study emphasizes the computational advantage over traditional optimization schemes where only FEM is utilized, several challenges persist. These include large training data, static surrogates, not considering objectives constraint and true multi-objective optimization.

Actuators

In this chapter, the fundamental working principles of the MEMS actuators studied in this work are explored, namely, the U-shaped Lorentz force actuator, the Lorentz force actuator with springs (serpentine), and the thermal actuator. Understanding how these actuators operate is essential for optimizing their performance and integrating them into various applications. Each section of this chapter is dedicated to one the actuators, providing insights into its functionality, design variables, objectives, and any constraints and considerations that must be taken into account before the optimization procedure.

3.1 | U-Shaped Lorentz Force Actuator

The first MEMS device to be optimized is a U-shaped Lorentz force actuator, depicted in Figure 3.1. When exposed to an external magnetic field, passing a current through the actuator and hence, the crossbar, generates a Lorentz force. This force moves the crossbar and, consequently, the attached load vertically. The primary purpose of this actuator is to

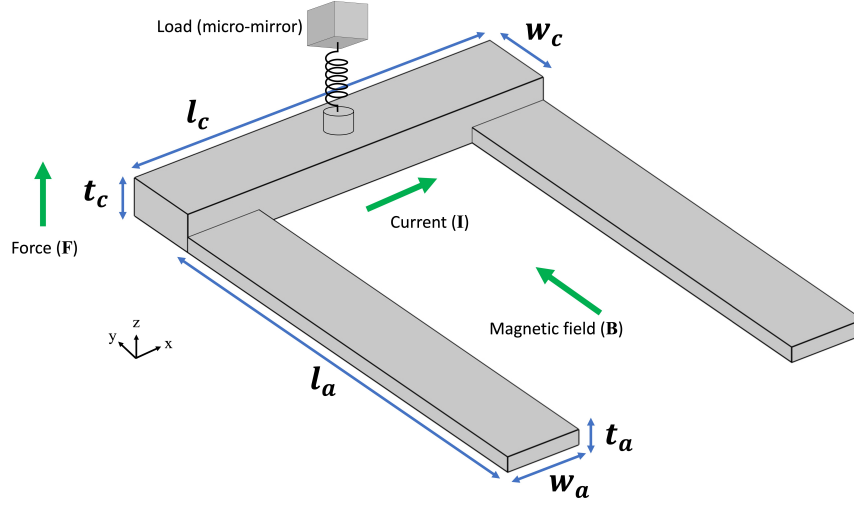


Figure 3.1: Schematic of the Lorentz force actuator.

create a deflection in the load, enabling precise movement or adjustment. In this work, the load is assumed to be a micro-mirror, modelled as a mass-spring system to account for its flexibility. Such a micro-mirror can be used in applications like the correction of electromagnetic waves incoming from space, making it a critical component in telescopes, for example.

The generated deflection is the primary purpose of this actuator. In this work, the desired deflection is set to a fixed value, and the optimization process focuses on identifying the objectives and design parameters that enable achieving this target deflection. By optimizing these objectives, the actuator's performance is tailored to meet the specific requirements necessary for the desired actuation.

Mathematically, the generated force is calculated using the equation:

$$|\vec{F}| = |\vec{I} \times \vec{B}| = BIL \cdot \sin \theta \quad (3.1)$$

where $\vec{F}$ is the generated force, I is the current, $\vec{L}$ is the length of the crossbar (l_c in Figure 3.1), $\vec{B}$ is the magnetic field, and θ is the angle between the magnetic field and the crossbar. In this work, the magnetic field is assumed to be provided via a permanent magnet in a way that the provided magnetic field is perpendicular to the crossbar. Since magnetic field and the crossbar are perpendicular for the studied actuator, the generated force is calculated as

$$|\vec{F}| = BIL \quad (3.2)$$

The current passing through the arms does not participate in the generation of force as the current flow direction is parallel to the magnetic field (the sine term in Equation 3.1 becomes 0.)

The current pathway typically consists of a thin conductive layer covering the entire structure, which is anchored to the substrate at the ends of each arm. Detailed information on the operational mechanics and fabrication process of a similar actuator, where vertical movement is transferred to a deformable mirror for waveform control, can be found in [52].

The actuator in question is defined by seven design variables, each with specific ranges outlined in Table 3.1.

It is assumed that the load is flexible, and as a result, its spring constant is one of the design variables. All other design variables are geometric features, depicted in Figure 3.1. For the optimization process, three objectives are considered:

- **Stress:** Stress corresponds to the von Mises stress that could predict the yielding point (point beyond which the structure deforms permanently) of the structure.

Table 3.1: Design variables of the U-Shaped Lorentz force actuator

Variable	Symbol	Lower bound	Upper bound	Unit
Arm length	l_a	200	500	μm
Arm width	w_a	20	200	μm
Arm thickness	t_a	5	20	μm
Crossbar length	l_c	500	3000	μm
Crossbar width	w_c	50	400	μm
Crossbar thickness	t_c	25	50	μm
Mirror spring constant	k_m	1	10	Nm^{-1}

- **Resonant frequency:** Resonant frequency [53], is the natural frequency at which the system oscillates with the greatest amplitude.
- **Reaction force:** Reaction force is the force applied to the crossbar when it pushes the load.

The displacement is set to a fixed value of $1 \mu m$ in each direction, totalling $2 \mu m$. The deformed structure undergoes a solid mechanics analysis to determine the resulting stress and the required reaction force. This approach enables the calculation of both the force necessary to achieve the specified deformation and the stress experienced by the structure. Resonant frequency is simulated through eigenfrequency study. The goal of the design optimization is to identify various design configurations that simultaneously maximize the resonant frequency while minimizing both stress and reaction force.

As is typical in multi-dimensional optimization problems, these objectives often conflict with one another. Furthermore, constraints have been established for both the resonant frequency and reaction force. The objectives and their corresponding constraints are detailed in Table 3.2.

Table 3.2: Optimization objectives of the U-shaped Lorentz force actuator

Objective	Constraint	Unit
Stress	N/A	Pa
Resonant frequency	Higher than 5×10^3	Hz
Reaction force	Lower than 2×10^{-5}	N

3.2 | Lorentz Force Actuator with Springs

The second MEMS studied in this work is a more complex version of the U-shaped Lorentz force actuator. The main distinction is that in this model, the crossbar is attached to serpentine structures that act as springs, hence name Lorentz force actuator with springs. This structure is much similar to the MEMS considered in [52].

Figure 3.2 shows the schematic of this design. As it can be viewed, it shares most of the design variables with that of the U-shaped structure with the exception that a new integer variable, n_s , is added to account for the number of spring turns. Hence, in total, there are 8 design variables, listed in Table 3.3.

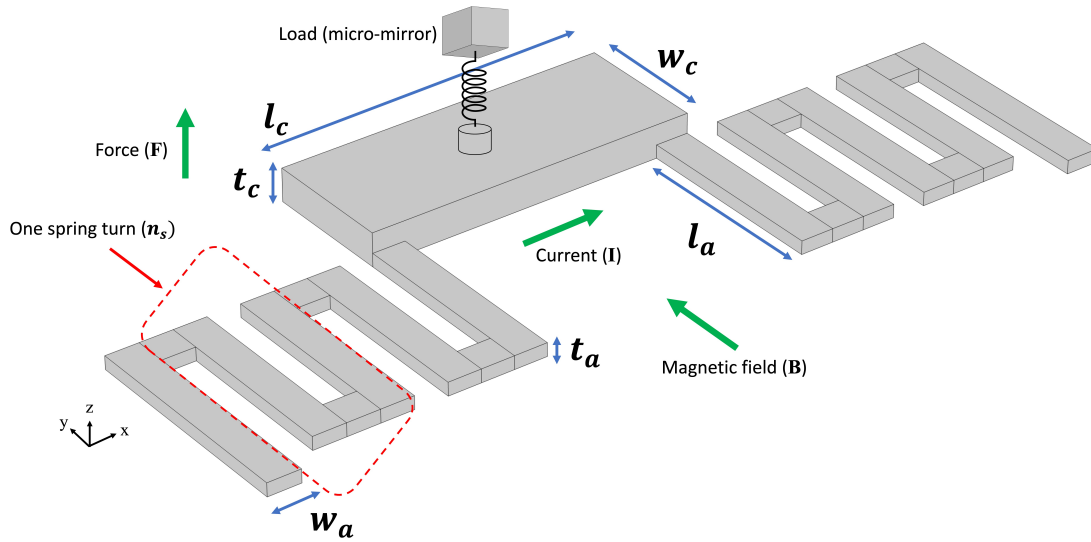


Figure 3.2: Schematic of the Lorentz force actuator with springs.

The addition of the mentioned design variable allows to test the performance of the optimization schemes in cases where discrete non-geometric parameters are considered. The device depicted in Figure 3.2 has two spring turns; one of them highlighted for reference. At each ends of the device, there is a bar that acts as an anchor, connecting the whole structure to the substrate.

In addition to the objective listed in Table 3.2, three new objectives are considered for this actuator:

- **Temperature:** Corresponds to the maximum temperature in the structure with respect to that of the ambient (25 °C or 298 K), resulting from resistive heating caused by the passing current.
- **Current:** Refers to the current required for the operation of the actuator.

Table 3.3: Design variables of the Lorentz force actuator with springs

Variable	Symbol	Lower bound	Upper bound	Unit
Arm length	l_a	200	500	μm
Arm width	w_a	20	200	μm
Arm thickness	t_a	5	20	μm
Crossbar length	l_c	500	3000	μm
Crossbar width	w_c	50	400	μm
Crossbar thickness	t_c	25	50	μm
Mirror spring constant	k_m	1	10	Nm^{-1}
Spring turns	n_s	1	10	-

■ **Voltage:** The voltage that needs to be applied to the structure for its operation.

As evident from their definition, all three new objectives are to be minimized. Eigenfrequency and solid mechanics studies are similar to those of the U-shaped actuator mentioned in section 3.1. Also, thermal study of the device is necessary in order to calculate the additional objectives. The temperature objective is also constrained. Table 3.4 provides the complete list of objectives and their limits. The addition of an objective constraint and design variable along with the increased structural complexity compared to section 3.1, significantly complicates the optimization process.

Table 3.4: Optimization objectives of the Lorentz force actuator with springs

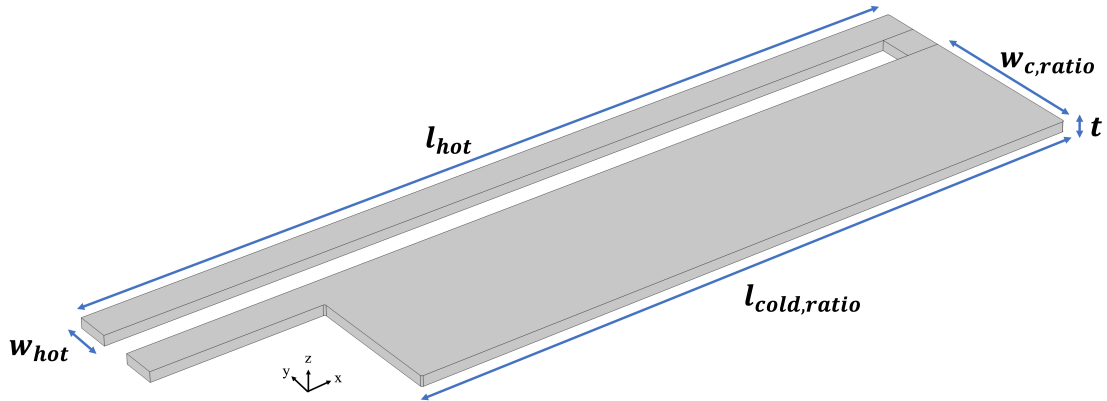
Objective	Constraint	Unit
Stress	N/A	Pa
Resonant frequency	Higher than 5×10^3	Hz
Reaction force	Lower than 2×10^{-5}	N
Temperature difference	Lower than 10	K
Current	N/A	A
Voltage	N/A	V

3.3 | Thermal Actuator

Another MEMS considered for the optimization in this work is the thermal actuator. As it can be seen in Figure 3.3 (a), the actuator consists of two primary hot and cold arms. The difference in cross-sectional area of the arms mean that the narrower one reaches significantly higher temperatures and as a result, expands to a larger degree. This differential expansion results in a bending motion and hence deformation, which is the core functionality of such devices. Figure 3.3 (b) illustrates this deformation (expansion is not to scale.)

Thermal energy is provided via resistive heating, resulting from the application of a current passing through the structure. A detailed explanation of the working principles of a similar structure is provided in [54].

The primary purpose of this actuator, similar to the Lorentz force actuators, is to



(a) Schematic of the thermal actuator



(b) Top view of the actuator in operation. Deformation is exaggerated.

Figure 3.3: Thermal actuator schematics

generate a deflection. However, unlike the Lorentz force actuators, this device is not designed to account for an attached load. Instead, the optimization is centred exclusively on the actuator's intrinsic deformation characteristics.

For the optimization, six design variables, provided in Table 3.5, are considered. The hot arm length, specifically, is a crucial parameter as it directly affects the amount of thermal expansion and consequently the actuation force generated by the device. By varying the length of the hot arm within a specified range, the actuator's displacement

and force characteristics can be optimized.

The width and length of the cold arm are determined via ratios of those of the hot arm. By using ratios, the relative dimensions of the cold arm can easily be adjusted to modify the structure geometries without being constrained by absolute measurements. This also expands the types of design variables considered in this work.

Table 3.5: Design variables of the thermal actuator

Variable	Symbol	Lower bound	Upper bound	Unit
Hot arm length	l_{hot}	100	500	μm
Cold/hot arm length ratio	$l_{cold, ratio}$	0.2	0.9	-
Hot arm width	w_{hot}	3	30	μm
Cold/hot width ratio	$w_{cold, ratio}$	1.1	10	-
Thickness	t	1	5	μm
Voltage	v	0.01	10	V

Five objectives are chosen for the optimization, shown in Table 3.6. Unlike the Lorentz force actuators, a higher limit of 300 MPa is set for the maximum von Mises stress of the structure. The minimum deformation constraint is chosen to 5 μm . An objective constraint is also applied to the temperature with a higher limit of 800 K. Temperature refers to the temperature difference of the hottest spot in the structure, under operation, to that of the ambient condition. Based on the discussion above, this occurs in the middle of the hot arm. Explanation of other objectives is provided in previous sections. All objectives are to be minimized, except for the resonant frequency and deformation.

Table 3.6: Optimization objectives of the thermal actuator

Objective	Constraint	Unit
Stress	Lower than 3×10^8	Pa
Resonant frequency	N/A	Hz
Deformation	Higher than 5	μm
Temperature difference	Lower than 800	K
Current	N/A	A

3.4 | Summary

This chapter presents an introduction to the MEMS actuators that will be subjected to optimization: the U-shaped Lorentz force actuator, the Lorentz force actuator with spring arms, and the thermal actuator. While the U-shaped and spring-arm actuators operate on similar principles, they exhibit key differences. Notably, the actuator with spring arms includes a discrete design variable, n_s , representing the number of springs at each end of the crossbar. Such variables, to the best of the author's knowledge, are barely (if any) discussed in the literature before in the context of MEMS design optimization. Additionally, this actuator requires an extra study to determine the temperature difference, current, and voltage objectives.

The thermal actuator operates on entirely different mechanics, further showcasing the versatility of the approach of this work. Each actuator has unique types, ranges, and

constraints on their design variables and objectives. The successful application of surrogate-assisted optimization to these diverse MEMS structures indicates the broad applicability and robustness of the developed optimization framework for various MEMS devices.

Methodology

In this chapter, the detailed steps for the optimization of the aforementioned MEMS devices are presented. Two distinct multi-objective design parameter optimization methods for MEMS will be discussed: Surrogate-Assisted Multi-Objective Optimization (SAMOO) and Adaptive Surrogate-Assisted Multi-Objective Optimization (A-SAMOO). It is important to note that certain steps are applicable to both SAMOO and A-SAMOO. Therefore, these steps are explained in the SAMOO section and not repeated for A-SAMOO.

This chapter is focused on detailing the approaches and techniques employed throughout this work rather than presenting specific results as they would be mentioned in the following chapters. However, to provide a clearer context and rationale for the chosen methodologies, we discuss whether certain methods proved effective or not along with a brief discussion on their behaviour during preliminary evaluations. These insights are not intended to showcase final results but to justify the decisions made in selecting and implementing the methodologies. By highlighting the strengths,

limitations, and performance characteristics of the explored methods, we aim to establish a solid foundation for understanding why the adopted approach aligns with the objectives of this research.

4.1 | Surrogate-Assisted Multi-Objective Optimization (SAMOO)

Figure 4.1 illustrates the flowchart of the SAMOO algorithm, providing an overview of its key building blocks. These steps include:

1. **Initial sampling of the design space:** A relatively small number of designs are sampled from the design space and evaluated using computationally expensive simulations to obtain their respective objective values.
2. **Preprocessing:** Objectives are scaled using various methods to study the effect of preprocessing on the quality of surrogates.
3. **Surrogate modelling:** Surrogate models are trained on the scaled datasets to imitate the original model. This model now "predicts" the performance of each design.
4. **Choosing the best model:** Based on the prediction accuracy of surrogates trained on differently scaled data, the best model is chosen to advance to the next step.
5. **Optimization:** An optimization algorithm is utilized to obtain the optimal solutions using the trained model as the evaluation function.

6. **Evaluating optimization results:** The optimal solutions found are evaluated via the original model (FEM) to ensure their actual performance is close enough to their predicted value calculated by the surrogate.

Since the surrogate evaluation are much faster to compute, the optimizer can use a large number of function evaluations without consuming heavy compute and time resources. Each step of this process is detailed in the subsequent sections.

The entirety of section 4.1 is dedicated to explaining and expanding on each of these building blocks in detail.

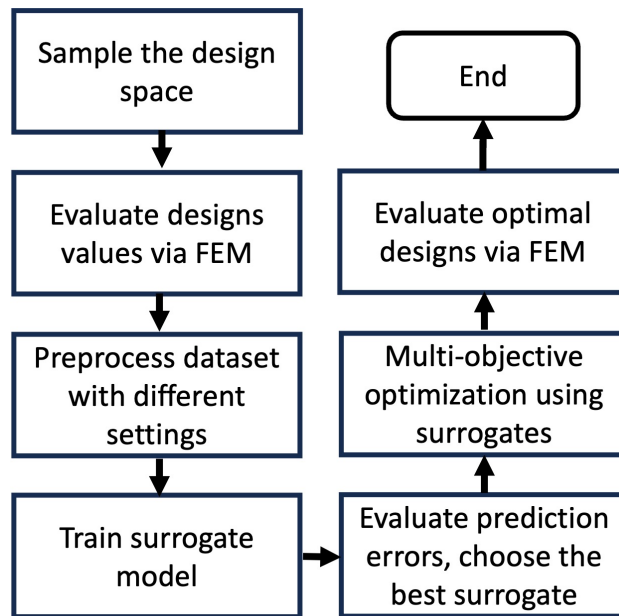


Figure 4.1: Flowchart of the SAMOO algorithm

4.1.1 | Initial Sampling of the Design Space

In order to construct a surrogate model, one has to provide the model with a set of training samples. These samples (designs) are drawn from the design space and then evaluated by the computationally expensive simulations.

It is crucial to select the initial samples to be evaluated in a manner that they cover all regions of the design space while being uniformly distributed. The reason being that a well-distributed initial sampling ensures that the surrogate model can effectively capture the underlying behaviour of the objective functions and constraints across the entire design space. If certain regions of the design space are underrepresented, the model may fail to approximate the performance metrics accurately in those areas, leading to poor predictive accuracy and potentially suboptimal designs. A uniform distribution also minimizes bias in the training data, enabling the surrogate model to generalize effectively and avoid overfitting to localized regions. Furthermore, an inadequate sampling strategy can result in a need for additional costly simulations to fill gaps in the data, thereby increasing computational expense and delaying the optimization process.

The simplest sampling strategy would be to choose design variable values randomly. This method, albeit computationally cheap, does not ensure proper coverage of the design space. Latin hypercube sampling (LHS) [56] is the most commonly used technique among different sampling strategies. Suppose that we are to sample n points from the design space. First, each dimension of the design space is split into n sections. LHS chooses the samples so that if we were to exit the design space from each dimension along the hypercubes (multi-dimensional volume), we would not encounter other

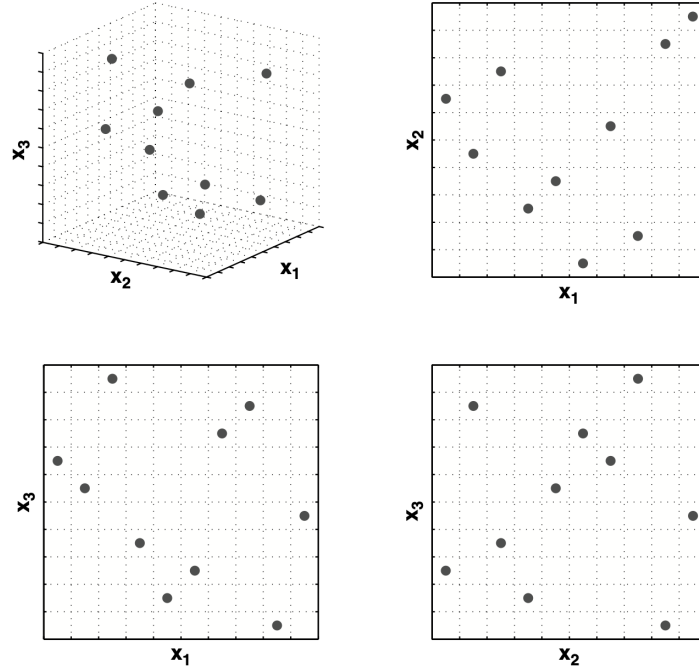


Figure 4.2: An example of Latin Hypercube Sampling (LHS) in a three-dimensional design space. The 2-D projections of each pair of dimensions are shown to illustrate that no row or column contains more than one design point. [55]

samples [55]. Figure 4.2 [55] illustrates an LHS sampling strategy in which ten samples are chosen from a three-dimensional design space. When looking at the location of samples on the 2-D plot of each dimension pair, it is evident that at most one samples occupies each row or column.

Latin hypercube sampling by itself does not ensure space-filling. In the example above, if we choose all ten points along the three-dimensional diagonal of the design space, the sampling strategy would satisfy LHS conditions. However, it is evident that many regions of the design space are not sampled. In order to properly fill the space, maximin criterion

[57] can be used along with LHS. As the name suggests, this criterion seeks to maximize the minimum distance between points, promoting an even distribution across the design space.

While larger sampling sizes lead to superior representation of the underlying simulations, this improvement comes at the cost of additional computational resources. Thus, the choice of sample size requires balancing model accuracy with computational efficiency.

During preliminary studies, various other sampling strategies were tested such as random sampling and LHS with other optimization criteria other than maximin. The comparison between these methods was conducted using L^2 -discrepancy [58] which measures the uniformity of sampling. Although the results were in favour of LHS, no tangible performance gain was observed when the datasets generated using LHS were used for surrogate model training. Nevertheless, LHS is chosen as the sampling strategy since it provides a more uniform representation of the design space.

In this work, LHS from scikit-optimize Python framework [59] is employed to generate a representative sample set for the design space that has built-in support for the the maximin criterion.

4.1.2 | FEM Modelling

Finite Element Method (FEM) is a numerical technique used to solve complex engineering and physical problems by breaking down a larger system into smaller, simpler parts called finite elements. These elements are connected at specific points called nodes, forming a

mesh that represents the geometry of the problem. FEM works by solving equations over each element to approximate the behaviour of the entire system. It is widely used in fields that require modelling of complex devices because it allows engineers to analyze intricate designs and predict their performance under various conditions with high accuracy.

The meshing size for the simulations was carefully chosen to balance convergence and computational efficiency. Larger mesh sizes were found to occasionally fail due to significant inaccuracies in the results, making them unsuitable for reliable optimization. On the other hand, smaller mesh sizes did not lead to noticeable changes in results but significantly increased computation times. More specifically, the chosen meshing sizes, provided later in subsection 4.1.2.1 and subsection 4.1.2.2, are tested to be small enough that finer sizes do not make significant changes to the simulation results. This is due the fact that the main contribution of this work is the optimization process itself and time-consuming sampling would only hinder extensive studies. More details on the meshing is provided later in subsection 4.1.2.1 and subsection 4.1.2.2.

In this study, simulations were performed using COMSOL Multiphysics version 6.2 on a Mac Studio equipped with an M1 Ultra processor and 64 GB of RAM. COMSOL provides an application programming interface (API) implemented in Java, enabling integration with external code. The simulations were initiated in Python using the MPh package [60], which leverages the aforementioned API through a Python-Java interface. Significant enhancements were introduced to the existing Python-COMSOL interface code to ensure consistent and efficient sampling, including the implementation of parallel simulation capabilities. This parallelization was critical for managing the computational demands of the study and reduced the overall sampling time by

approximately a factor of eight compared to sequential sampling.

Several technical challenges were encountered during this phase of the study, the details of which are beyond the scope of this discussion. However, one of the major contributions of this work is the development of a Python module titled "ai4d." This module is not limited to managing COMSOL simulations but offers a comprehensive framework for the entire optimization workflow in this chapter. This includes initial sampling, surrogate modelling, and multi-objective optimization.

Certain values used in the model construction, such as the magnetic field and deformation in the case of Lorentz actuators, were predefined as constants within COMSOL. However, these values can be readily modified through the developed code, providing flexibility for various simulation scenarios. The design variables of the COMSOL model were defined parametrically. During the sampling loop conducted by ai4d module, the value of these parameters were set to those of the design that was to be evaluated. Then, FEM studies were conducted. The simulation objectives were identified as 'Derived Values' within COMSOL, which were then extracted back into python for further analysis. For simplicity, basic objective manipulations—such as converting absolute temperatures to temperature differences relative to ambient conditions—were performed directly within the COMSOL environment.

4.1.2.1 | Lorentz Force Actuators

Since these actuators have applications for adaptive optics, the load is considered to be a mirror membrane composed of Au/Si₃N₄/Au layers. Based on the volume of this layers,

the mirror mass is computed. The mirror is then modelled as a mass and spring system attached to the crossbar via a small pillar depicted in Figure 3.1 and Figure 3.2. A thin film Al layer, covering the whole structure, provides the pathway for current. The actuator itself is made of polycrystalline silicon. The magnetic field required for the operation is assumed to be provided externally. This value is used for the calculation of required current and voltage. The crossbar vertical displacement in one direction is set to a fixed value and hence, the total displacement of the cross bar is twice the value.

A trial and error process was conducted to obtain the meshing configuration that balanced the convergence and consistency of simulations and computational times. Meshing is calibrated for "General physics" and set to "Predefined Coarse" size. Parameters used in modelling and their values are listed in Table 4.1.

Table 4.1: The parameters used for modelling of Lorentz force actuators in COMSOL Multiphysics

Parameter	Value	Unit
Crossbar vertical displacement in one direction	1	μm
Al layer thickness	1	μm
External magnetic field	0.7	T
Mirror area	1×1	mm
Mirror Au layer thickness (each one)	0.15	μm
Mirror Si ₃ N ₄ layer thickness	0.5	μm
Au density	19300	kg/m ³
Si ₃ N ₄ density	2850	kg/m ³
Calculated mirror mass	7.215×10^{-9}	kg
Polycrystalline silicon parameters		
Heat capacity at constant pressure	678	J/(kgK)
Density	2320	kg/m ³
Thermal conductivity	34	W/(mK)
Young's modulus	160×10^9	Pa
Poisson's ratio	0.22	-

4.1.2.2 | Thermal Actuator

Similar to Lorentz force actuators, the material chosen for the thermal actuator structure is polycrystalline silicon with the same parameters as those listed in Table 4.1. The width and length of the junction between the two arms is equal to the width of the hot arm.

At micro-scale, heat transfer via radiation is negligible [61, 62]. Also, conduction dominates convection [63]. Hence, for the thermal modelling it is assumed that the only form of heat transfer is via conduction to the substrate. Meshing type is set to "Physics-controlled" with the automatic size of "Extra fine".

4.1.3 | Preprocessing

As discussed before, preprocessing objectives for training surrogate models, as well as its role in the optimization process, is a relatively unexplored area in surrogate-assisted optimization of MEMS. In this study, three different preprocessing techniques were considered for the objectives before integrating them into the surrogate models:

- Normalization, defined as:

$$y_{norm} = \frac{y - y_{min}}{y_{max} - y_{min}} \quad (4.1)$$

This equation scales the objectives between 0 and 1, using the minimum and maximum values from the initial samples. However, because optimization results may extend beyond this initial range, some objectives might end up with negative scaled values. This can complicate the evaluation of surrogate modelling and optimization outcomes. To address this, 1 was added to the scaled objectives to ensure all values remain positive, resulting in ranges between 1 and 2.

- Log transformation where values undergo natural logarithm transformation. For instance, if the value of the resonant frequency objective is 4000 *Hz*, the scaled value would be 8.29.

- Log transformation followed by normalization

Preliminary results showed that scaling the design variables had less of an impact compared to scaling the objectives. Consequently, only the combination of log transformation and normalization were used for the design variables as described above.

During preliminary studies, filtering based on standard deviation was explored as a preprocessing method to refine the dataset. For example, a sample with any objective value greater than three standard deviations from the mean was discarded, as it was considered an outlier. However, this approach was not pursued further, as it resulted in the loss of samples that were computationally expensive to obtain, given that they were derived from FEM simulations.

4.1.4 | Surrogate Modelling

During our preliminary studies, numerous surrogate models (mostly from scikit-learn [64] and SMT [65] python frameworks) were tested. Some of which are:

- Linear regression,
- Artificial neural network (ANN),
- Ridge regression [66],
- Lasso regression [67, 68],
- Least angle regression (LARS) [69],
- Random Sample Consensus (RANSAC) [70],

- Support vector regression (SVR),
- Nu Support Vector Regression (NuSVR),
- Second order (quadratic) polynomial approximation,
- Kriging Partial Least Squares (KPLS) [71],
- Gradient-enhanced Kriging Partial Least Squares (GEKPLS) [72],

However, SAMOO uses GPR for mainly three reasons. One, our experiments showed that the prediction accuracy of the surrogates listed above are either lower or on par with GPR. Second, as mentioned in section 2.1, GPR has the unique feature that it also provides an estimate of its predictions. Third, since A-SAMOO uses this surrogate, using GPR in SAMOO enables a more meaningful comparison between the two.

4.1.4.1 | Gaussian Process Regression (GPR)

At its core, GPR assumes that the data originates from a Gaussian process, which consists of a set of random variables with a joint Gaussian distribution [73]. This assumption allows GPR to model complex, non-linear relationships between inputs and outputs effectively.

The covariance function, or kernel, is a fundamental component of GPR that defines the similarity between different input points. It determines how much influence one data point has on another and controls the smoothness and generalization properties of the resulting regression function.

GP regression works as follows: If we are interested in modelling function $f(\mathbf{x})$, $\mathbf{x} \in \mathbb{R}^d$, GP assumes that only noisy observations of f , $y = f(\mathbf{x}) + \epsilon$, are available where both $f(\mathbf{x})$ and ϵ (noise) are random variables with Gaussian distributions of $N(0, k(\mathbf{x}, \mathbf{x}'))$, and $N(0, \sigma_\epsilon^2)$, respectively. The correlation between each pair of function values at $\mathbf{x}$ and $\mathbf{x}'$ is defined by a covariance function (kernel). In this work, squared exponential kernel is used defined as:

$$k(\mathbf{x}, \mathbf{x}') = \sigma_f^2 \exp\left(-\frac{1}{2\lambda^2} |\mathbf{x} - \mathbf{x}'|^2\right) \quad (4.2)$$

in which σ_f^2 is the variance of the function to be modelled and λ is a hyperparameter called the length-scale. We have collected a set of n training observations, $\{\mathbf{X}_t, \mathbf{y}_t\}$ such that $\mathbf{X}_t \in \mathbb{R}^{n \times d}$ and $\mathbf{y}_t \in \mathbb{R}^n$. We intend to predict the value of $\mathbf{f}_\star$ for a set of m new inputs $\mathbf{X}_\star \in \mathbb{R}^{m \times d}$. The predictions can be drawn from a Gaussian distribution with the following mean and variance:

$$\text{mean}(\mathbf{f}_\star) = \mathbf{K}_{\star,t} [\mathbf{K}_t + \sigma_\epsilon^2 \mathbf{I}]^{-1} \mathbf{y}_t \quad (4.3)$$

$$\text{var}(\mathbf{f}_\star) = \mathbf{K}_\star - \mathbf{K}_{\star,t} [\mathbf{K}_t + \sigma_\epsilon^2 \mathbf{I}]^{-1} \mathbf{K}_{t,\star} \quad (4.4)$$

in which $\mathbf{K}_{\star,t} \in \mathbb{R}^{m \times n}$ is the covariance matrix between each pair of new and training inputs, $\mathbf{K}_t \in \mathbb{R}^{n \times n}$ is the covariance matrix between each pair of training inputs and $\mathbf{K}_{t,\star} \in \mathbb{R}^{n \times m}$ is the covariance matrix between each pair of training and new inputs. Each element of covariance matrices, denoted as $\mathbf{K}(i, j)$ represents the covariance between the i -th and j -th points calculated via Equation 4.2. $\mathbf{I} \in \mathbb{R}^{n \times n}$ is also the identity matrix. In practice, (4.3) is used as the prediction. The hyperparameters λ , σ_f^2 , and σ_ϵ^2 are collectively

denoted at θ and are approximated by maximizing the log marginal likelihood:

$$\log p(\mathbf{y}|\mathbf{X}, \theta) = -\frac{1}{2}\mathbf{y}^T \mathbf{K}_y^{-1} \mathbf{y} - \frac{1}{2} \log |\mathbf{K}_y| - \frac{n}{2} \log(2\pi) \quad (4.5)$$

where $\mathbf{K}_y = \mathbf{K}_t + \sigma_\epsilon^2 \mathbf{I}$. More information on Gaussian process regression can be found in [73, 74].

One could argue that the GPR model used in this work performs its own internal optimization during the training process, as it attempts to minimize the specified "loss function" in Equation 4.5. This optimization ensures that the model parameters, such as the kernel hyperparameters, are fine-tuned to best represent the training data.

Gaussian process regression is not without limitations. The reliability of uncertainty estimates hinges on several factors. The choice of kernel function is crucial; a poor choice can lead to overconfident predictions in data-sparse regions or underconfident predictions where the model should be certain. Furthermore, the computational complexity of GPR, particularly the inversion of the covariance matrix, scales cubically with the number of training points, posing a challenge for large datasets [75]. The complexity, however, is not much concerning for application where obtaining a large number of training samples is not practical.

This study utilizes the GP implementation in Scikit-learn Python framework [64]. Additionally, it is important to note that a separate GPR model is trained for each objective, and these models operate completely independently of one another.

4.1.4.2 | Prediction Accuracy Metric

After the training of the surrogate models is complete, it is crucial to evaluate their predictive accuracy. Various error metrics were utilized to assess the accuracy of the surrogates, such as root mean squared error (RMSE), mean absolute error (MAE), and coefficient of determination (R^2). The primary error metric used, which is also reported in this work, is mean absolute percentage error (MAPE), defined as:

$$MAPE = \frac{1}{n} \sum_{i=1}^n \left| \frac{y_t - y_p}{y_t} \right| \quad (4.6)$$

where y_t and y_p refer to the true and predicted value of an objective. The choice of MAPE is driven by its ability to convey the average discrepancy between predicted and actual values in a clear, percentage-based format. This feature of MAPE is particularly useful as it allows for direct comparison of errors across different objectives, regardless of their individual scales. As mentioned before, models are trained on different sets of objective values of varying scales. Following this step, the model with the lowest prediction error is selected to move forward to the optimization stage.

4.1.5 | Optimization

In this phase, the surrogate-assisted multi-objective optimization algorithm explores the design space, identifies potential solutions, assesses their performance via the surrogate model (as opposed to the generic approach of using FEM), and determines the optimal designs based on these evaluations.

4.1.5.1 | NSGA-II

The optimization algorithm used in this study is Non-dominated Sorting Genetic Algorithm II (NSGA-II) [46]. NSGA-II, an evolutionary algorithm, is extensively employed in the optimization of engineering problems [76], including MEMS design optimization [20, 22]. The flowchart of NSGA-II algorithm is presented in Figure 4.3. The step distinguished using dashed border is where the surrogate model is used.

NSGA-II begins by generating a initial population of solutions using random sampling, LHS or any other sampling strategy. It employs non-dominated sorting, a method that classifies solutions based on Pareto dominance, assigning lower ranks to more optimal solutions. Starting from the lowest rank, solutions are added to the set that would be used for child generation. If only a subset of a front (solution set) can make it to this set due to limited number of population size, then, solutions would be selected based on their crowding distance mentioned in Algorithm 1. Crowding distance is a measure of solution density in the objective space and is used to maintain diversity among solutions.

Algorithm 1 Crowding distance (D) used in NSGA-II

```

 $i \in 1 \dots n$  solutions,  $j \in 1 \dots m$  objectives,  $y$  objective values
 $D_{i \dots n} \leftarrow 0$ 
for  $j = 1 \dots m$  do
    Sort  $y^j$ 
     $D_1 = D_n = \infty$  ▷ Ensure survival of boundary points
    for  $i = 2 \dots (n - 1)$  do
         $D_i = D_i + (y_{i+1}^j - y_{i-1}^j) / (y_{max}^j - y_{min}^j)$ 
    end for
end for

```

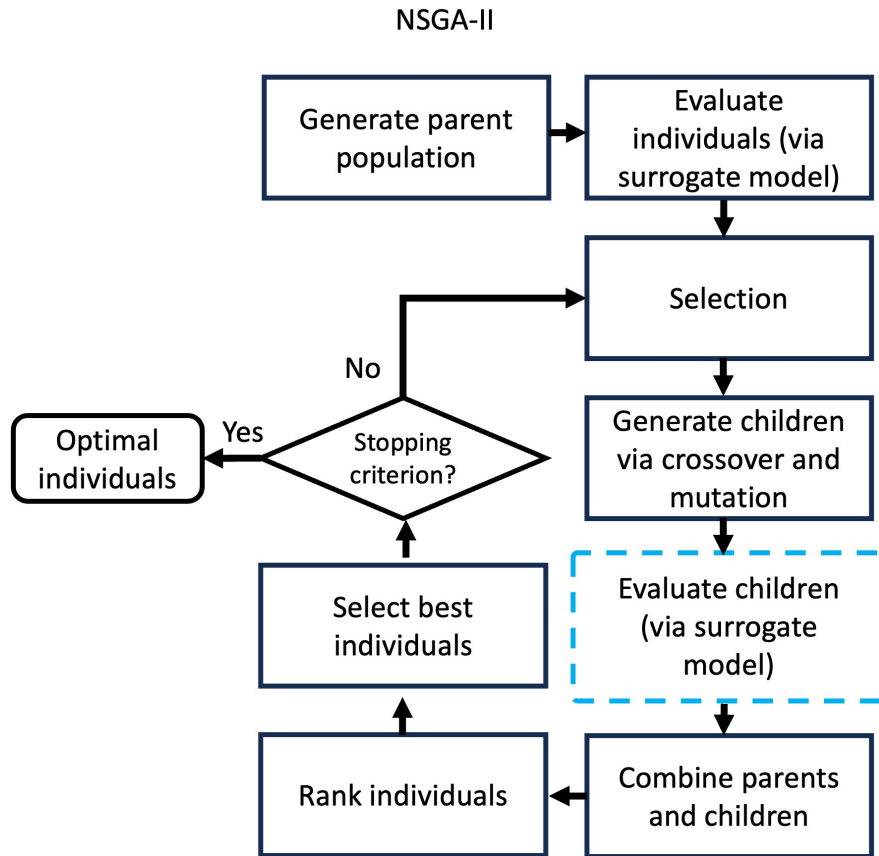


Figure 4.3: Flowchart of NSGA-II.

The algorithm selects parents for "breeding" based on these criteria and then applies genetic operators, crossover and mutation, to produce offspring. This process iteratively evolves the population over generations, progressively moving towards an optimal set of diverse, non-dominated solutions. The number of function evaluations required for the optimization is a multiplication of offspring size and the number of generations (iterations). It is suggested that for a fixed number of function evaluations, reducing the size of the offspring population while increasing the number of generations enhances the

effectiveness of the NSGA-II algorithm [77].

NSGA-II implementation found in the Pymoo Python library [78], which has the capability to manage objective constraints, is utilized in this study.

4.1.5.2 | Evaluation of Optimization Results

Performance metrics, or quality indicators (QIs), are crucial in the field of multi-objective optimization. As the name indicates, these metrics are used to evaluate quality of optimization solutions and hence, the optimization process itself.

To the best of the author's knowledge, QIs are not mentioned in previous works regarding surrogate-assisted optimization of MEMS. This seems rather surprising as without QIs, it is difficult to assess and compare results of optimization processes. As will be discussed, even when QIs are utilized, the task of comparing different solution sets is not straightforward.

Although plotting can provide valuable insights and serve as the most basic QI [79, 80], it has major limitations:

- Plotting is not simple when the number of objectives is more than three.
- It cannot quantify the difference between solution sets.
- If the algorithm is run multiple times, the user cannot easily decide which run to plot.

Nevertheless, parallel coordinate, spider web (radar), and scatter plot matrix are suitable plotting layouts that can be used.

The Pareto front and the non-dominated solution set are related but distinct concepts in multi-objective optimization. The "Pareto front" refers to the best possible set of non-dominated solutions in the entire objective space. These solutions represent the true optimal trade-offs between objectives, where improving one objective would require compromising at least one other. The Pareto front serves as a benchmark of ideal performance, showing the boundary of optimality in the objective space.

In contrast, the "non-dominated solution set" refers to the set of solutions found by an optimization algorithm during a specific run or process. These solutions may or may not fully represent the true Pareto front, as they depend on the algorithm's efficiency, accuracy, and the constraints of the problem. While the non-dominated solution set is a subset of all evaluated solutions that are not dominated by any other in the set, it might only approximate the true Pareto front rather than defining it entirely.

Thus, the Pareto front represents the theoretical optimum in the objective space, while the non-dominated solution set is the practical outcome of a given optimization effort, which may approach but not always perfectly align with the true Pareto front.

In MOO, the quality of a solution set generally means that it can well represent the Pareto front of the problem. Pareto front is defined as the set of all Pareto optimal solutions. A solution is Pareto optimal if no other feasible solutions exist that would improve some objective without causing a simultaneous degradation in at least one other objective. In other words, moving from one point on the Pareto front to another involves a trade-off between conflicting objectives.

Let $\mathbf{x}$ and $\mathbf{y}$ be two solutions in the set Z . We say that solution $\mathbf{x}$ weakly (Pareto) dominates solution $\mathbf{y}$ (denoted as $\mathbf{x} \preceq \mathbf{y}$) if for every objective i (where $1 \leq i \leq m$) the

inequality $x_i \leq y_i$ holds. Furthermore, if there is at least one objective j such that $x_j < y_j$, then $\mathbf{x}$ is said to dominate $\mathbf{y}$ (denoted as $\mathbf{x} \prec \mathbf{y}$). A solution $\mathbf{x} \in Z$ is considered Pareto optimal (or efficient) if no other solution $\mathbf{y} \in Z$ strictly dominates it. The collection of all Pareto optimal solutions for a given multi-objective optimization problem is known as the Pareto front (or Pareto optimal frontier) [81].

Figure 4.4 [81] demonstrates a solution set for a car purchase problem where performance is to be maximized while price is minimized. Black and white points refer to the Pareto optimal and dominated subsets. As it can be viewed, there is no point in the Pareto optimal subset that is "better" than any other point without compromising one of the objectives.

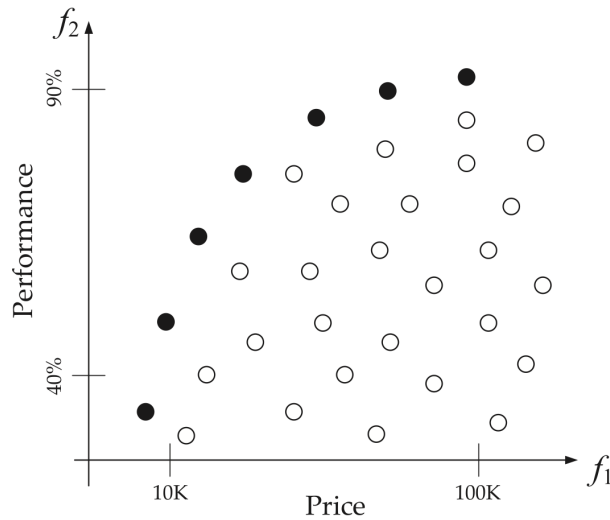


Figure 4.4: A solution set for a car purchase problem. Black points are the Pareto optimal solutions in this set [81].

The most commonly used QI in the literature (apart from plotting and simple

measure like average of objectives) is the hypervolume indicator. It is defined as the union of hypervolumes (n-dimensional volumes) created between each point in the set and a reference point. Formally, for a solution set A and reference point r :

$$HV(A) = \lambda \left(\bigcup_{a \in A} \{x \mid a \prec x \prec r\} \right) \quad (4.7)$$

where λ is the Lebesgue measure. The Lebesgue measure is a mathematical concept used to assign a "volume" or "size" to subsets of a given space, particularly in $\mathbb{R}^n$. It is a generalization of the intuitive notions of length, area, and volume. For subsets of $\mathbb{R}$, $\mathbb{R}^2$, or $\mathbb{R}^n$:

- The Lebesgue measure of an interval in $\mathbb{R}$, such as $[a, b]$, is its length, $b - a$.
- For rectangles in $\mathbb{R}^2$ or $\mathbb{R}^n$, it is the product of the lengths of its sides (area or volume).
- For more complex sets, the Lebesgue measure extends this concept by approximating the set using countable unions of such intervals or rectangles.

There are multiple reasonable selections for the reference point such as the nadir and worst objective points (or vectors when more than three objectives are considered). Figure 4.5 [82] visualizes these points in the objective space.

The hypervolume metric inherently assumes minimization for all objectives. Therefore, if an objective is to be maximized, its corresponding dimension in both the reference point and the objective values is multiplied by -1. This transformation effectively converts the maximization problem into a minimization problem, ensuring that the hypervolume calculation remains valid and consistent across all objectives.

It should be noted that, based on the definition provided above, when evaluating the quality of two different solution sets using hypervolume, it is essential to ensure that the reference point used in the calculation remains consistent. This is because the hypervolume metric is sensitive to the choice of the reference point, as it measures the volume of the objective space dominated by the solution set relative to this point. Using different reference points would lead to incomparable results, potentially misrepresenting the relative quality of the solution sets.

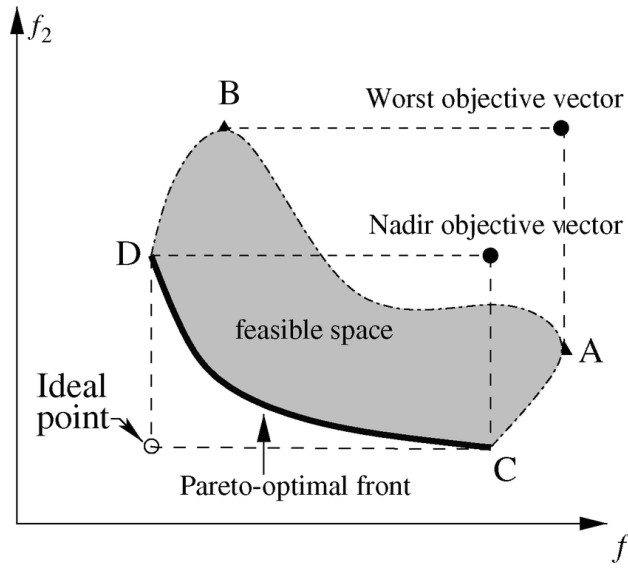


Figure 4.5: Visualization of different points associated with a MOO problem [82]

When multiple solution sets are to be compared with respect to each other, a common practice is to combine these sets and choose the reference point from this constructed set [83]. The worst objective point of the combined solution set is used as the reference point for hypervolume calculation.

Following the optimization process, a set of non-dominated solutions are obtained.

However, since these designs have been evaluated using surrogate models, validating them through actual simulations is essential to determine their actual objective values. The number of these non-dominated solutions that meet the predefined objective constraints is reported. Additionally, to assess the accuracy of the provided optimization procedure, the discrepancy between the surrogate-evaluated solutions and their true objective values is measured using MAPE.

4.2 | A-SAMOO

4.2.1 | Overview of the Optimization Procedure

In the previous optimization approach, the surrogate model was trained only once using the initial set of samples generated via finite element method simulations. However, this approach has a significant drawback. For some optimization problems, the optimal solutions may be concentrated within a small region of the design space. Since samples in the training dataset are spread across the entire design space, many of them will fall in areas that do not contribute much to representation of optimal regions. This leads to a substantial waste of computational resources on FEM simulations that refine the surrogate model's accuracy in irrelevant regions. Therefore, any strategy that focuses sampling efforts on the most promising regions of the design space could significantly reduce computational costs. More importantly, it would enhance the accuracy of the surrogate model in regions that are likely to yield optimal solutions. This form of "adaptive" sampling can be dynamically integrated into the optimization process.

Adaptive Surrogate-Assisted Multi-Objective Optimization (A-SAMOO) seeks to leverage online learning to continuously refine the surrogate model within optimal regions as the optimization progresses. Initially, a limited set of samples is chosen from the design space and evaluated using FEM simulations to create a preliminary surrogate model. This initial model guides the optimization process similarly to section 4.1. However, unlike static modelling, the surrogate model here is continuously updated throughout the optimization. The updating is based on new FEM simulations, but only in regions already indicated as potentially optimal.

The benefits of this adaptive technique can be viewed from two key perspectives. First, with a fixed budget for FEM simulations, the surrogate model becomes more accurate where it is most needed, likely leading to a higher quality set of solutions. Second, if additional computational resources are available, they can be dedicated exclusively to refining the model in these promising regions, further improving the accuracy and optimality of the results.

Most steps outlined in section 4.1 remain applicable in this context. The primary difference lies in the optimization phase which by its nature is the defining feature of this approach. Therefore, only the optimization step is presented here.

4.2.2 | Optimization

The A-SAMOO scheme explained here substitutes the dashed stage in Figure 4.3 which corresponds to objective evaluation. The pseudo-code and flowchart of A-SAMOO are demonstrated in Algorithm 2 and Figure 4.6, respectively.

Algorithm 2 Objective Evaluation in NSGA-II using A-SAMOO

```

Set  $n\_eval\_per\_gen$ , ▷ Number of FEM evaluations per generation
Set  $n\_reserved\_evals \leftarrow 0$ , ▷ Initiate number of reserved FEM evaluations
Set  $std\_threshold$ , ▷ Standard deviation threshold
while optimization not finished do
  Get design variables to be evaluated from NSGA-II
   $n\_to\_eval \leftarrow n\_adaptive\_samples + n\_reserved\_evals$ 
  for each evaluation in  $n\_to\_eval$  do
    Predict objectives and uncertainties using surrogate model
    Find optimal (non-dominated and constraint-compliant) solutions
    if no optimal solutions found then
       $n\_reserved\_evals \leftarrow$  remaining evaluations
      break
    end if
    Select solution with maximum uncertainty for evaluation
    if maximum uncertainty  $>$  threshold then
       $n\_reserved\_evals \leftarrow$  remaining evaluations
      break
    end if
    if using reserved evaluations then
       $n\_reserved\_evals \leftarrow n\_reserved\_evals - 1$ 
    end if
    Evaluate selected solution using FEM simulation
    Update and retrain surrogate model with new data
  end for
  Predict objectives using the updated surrogate model
  Return evaluated samples to NSGA-II
end while

```

Within the optimization loop, NSGA-II generates a set of design variables and objectives that need evaluation. The number of FEM evaluations for each generation is calculated as the number of FEM evaluations assigned to each generation plus any reserved evaluations. Reserved evaluations corresponds to previous generations that did

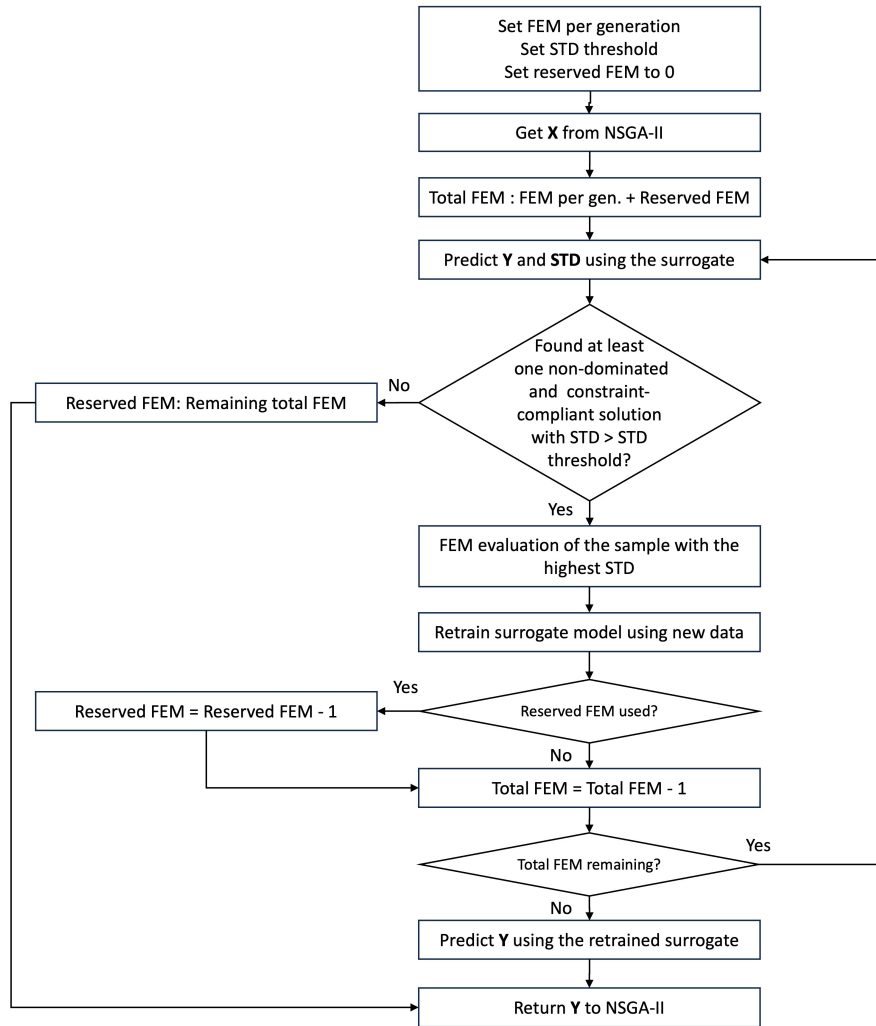


Figure 4.6: Flowchart of A-SAMOO

not perform any FEM. This mechanism will be explained later.

During the evaluation process, the surrogate model predicts the objectives and their associated uncertainties in term of standard deviation. The algorithm identifies solutions that are both non-dominated and compliant with the given objective constraints. If no such solutions are found, the remaining evaluations are reserved for future generations to

conserve computational resources. Predicted objectives are simply returned to NSGA-II for ranking and selection of individuals that will make it to the next generation.

If optimal solutions are identified, the algorithm selects the one with the highest uncertainty for evaluation. This approach targets areas where the surrogate model is least certain, thereby improving the model's accuracy in regions where it could benefit the most from higher precision. Before conducting the FEM evaluation, the algorithm checks whether the maximum uncertainty is larger than the predefined threshold. If it is not, further evaluations are deemed unnecessary and the remaining evaluations are reserved for future use. If reserved evaluations are utilized, their count is decremented accordingly.

The rationale for defining a standard deviation threshold is twofold. Firstly, the threshold ensures that only those designs with sufficiently high uncertainty in the surrogate model's predictions are selected for evaluation. This means that evaluations are only conducted when the surrogate model's uncertainty is significant, thereby preventing unnecessary evaluations of samples whose objectives can already be predicted with reasonable accuracy. In essence, the threshold discourages the evaluation of samples that do not add meaningful information to the optimization process.

Secondly, the configuration of the GPR, particularly the choice of the kernel, results in a negligible standard deviation for the training samples. This low standard deviation indicates high confidence in the model's predictions for these samples. By setting a threshold, the algorithm prevents designs that are already in the training set from being re-evaluated, as their predicted uncertainties fall far below the defined threshold.

Selected solution is then evaluated using FEM simulations. The objectives obtained

are used to update and retrain the surrogate model, incorporating new data to refine its predictions and reduce uncertainties in targeted regions.

It is important to note that when multiple samples can be assigned for FEM evaluation within a single generation ($n_to_eval > 1$), the process of re-evaluation and surrogate model updating is performed sequentially rather than in parallel. This means that each sample is evaluated one at a time, and after each FEM simulation, the surrogate model is updated accordingly.

Consider a scenario where $n_to_eval = 5$. Initially, the surrogate model predicts standard deviations for the objectives, identifying five samples as optimal candidates for evaluation. The sample with the highest uncertainty is selected first and evaluated using FEM simulation. The surrogate model is then retrained with this new data. Following this update, the loop repeats: the updated surrogate model predicts the standard deviations for all designs again. In this new evaluation, it may be discovered that none of the remaining four samples from the initial five now satisfy the objective constraints. As a result, the number of reserved evaluations is incremented by four, and the predicted results are returned to NSGA-II for further optimization steps.

This sequential approach is deliberate and serves a strategic purpose, as demonstrated in the example. Especially in the early generations, when the model is most uncertain, a single evaluation can substantially alter the predicted objective values and uncertainties, potentially making further evaluations within that generation unnecessary. The benefit of this method is that it allows the optimization process to adapt dynamically based on the most current information, reducing redundant evaluations and focusing resources more effectively.

While this approach does increase the frequency of surrogate model training and prediction steps, these operations are computationally inexpensive compared to FEM evaluations. Thus, the additional computational overhead is minimal or even negligible, ensuring that the overall efficiency and speed of the optimization process are maintained.

After completing the evaluations for the generation, the surrogate model predicts the objectives for the updated solutions. These evaluated samples are then fed back into the NSGA-II algorithm, guiding next stages of the optimization in this generation, ranking and selection.

4.3 | Parameters

LHS is employed to generate five datasets within the design space with the sampling sizes of 20, 50, 100, 200, and 400. Five-fold cross-validation is used to assess prediction accuracy of surrogate models. The parameters associated with NSGA-II optimization algorithm are listed in Table 4.2. Interested readers can refer to [84] for more information on crossover and mutation defined in Table 4.2.

Table 4.2: NSGA-II Parameters

Parameter	Value
Population size	50
Offspring size	50
Number of generations	50
Crossover Probability	0.9
Crossover eta	20
Mutation eta	20

4.4 | Summary

This chapter presents two approaches for optimizing the design parameters of MEMS devices: Surrogate-Assisted Multi-Objective Optimization (SAMOO) and Adaptive Surrogate-Assisted Multi-Objective Optimization (A-SAMOO). The SAMOO method involves generating a dataset through finite element method simulations and training a surrogate model to approximate the behaviour of these simulations efficiently. Initial samples are selected using LHS combined with the maximin criterion to ensure a comprehensive coverage of the design space. The surrogate model, constructed using Gaussian process regression, is trained to predict both the objective values and their uncertainties, allowing for efficient optimization without the need for extensive FEM simulations. The performance of the surrogate model is evaluated using MAPE. NSGA-II, an evolutionary algorithm, is employed to guide the search for optimal solutions.

In contrast, the A-SAMOO approach dynamically updates the surrogate model throughout the optimization process, focusing on regions of the design space that are most likely to yield optimal solutions. This adaptive sampling strategy addresses the limitations of static sampling by concentrating computational resources on promising areas, thereby improving model accuracy where it is most needed and reducing unnecessary FEM evaluations. The adaptive method's iterative refinement ensures that the optimization process is more efficient, with higher-quality solutions achieved with fewer simulations. A-SAMOO also utilizes NSGA-II to handle multi-objective optimization, leveraging its ability to maintain diversity among solutions through non-dominated sorting and crowding distance mechanisms. Moreover, this chapter emphasizes the importance of quality indicators for evaluating the optimization results; a subject barely mentioned in the MEMS surrogate-assisted optimization literature.

U-Shaped Lorentz Force Actuator Results

This chapter presents the results of the developed optimization schemes for the U-shaped Lorentz force actuator. Key parameters utilized in the analysis are introduced and explained in detail. These parameters will remain consistent in the subsequent chapters, which present the results for the serpentine Lorentz force actuator and the thermal actuator, unless explicitly stated otherwise. This detailed discussion establishes a foundation for understanding the analyses presented in the remainder of this work.

5.1 | Initial Sampling

As mentioned in chapter 4, the optimization scheme starts with sampling of the design space followed by their evaluation using COMSOL Multiphysics simulations. The objective ranges and the computation times are provide in Table 5.1. These values are obtained from the combined set of all training sizes; hence the combined set has a size of 770.

The maximum stress obtained (worst value as stress is to be minimized) is 47 MPa with an average of 10.1 MPa over all sampled designs. For polycrystalline silicon at microscale, the ultimate and endurance strength are both in the GPa range [85]. Hence, the maximum stress on the structure of the actuator for all designs is far below the yielding strength of polycrystalline silicon. This was found during the initial studies and was the motivation for not setting a constraint on von Mises stress. Resonant frequencies range from hundreds Hz to tens of thousand Hz . As a results, unlike stress, some designs must be ignored during the optimization only for their low frequency values as the constraint on this objective is 5 kHz . The condition for the reaction force is worst since even the average value is $121 \text{ }\mu\text{N}$, almost two orders of magnitude higher than the constraint of $20 \text{ }\mu\text{N}$. The average simulation time for each FEM evaluation is 6 seconds.

Table 5.1: Objective ranges and computation times of initial samples for U-shaped Lorentz force actuator

Parameter	Unit	Minimum	Maximum	Average
Stress	Pa	1.92×10^6	4.70×10^7	1.01×10^7
Resonant frequency	Hz	8.49×10^2	4.51×10^4	8.95×10^3
Reaction force	N	3.03×10^{-6}	7.92×10^{-4}	1.21×10^{-4}
Simulation time	s	4.53	9.82	6.02

5.2 | Surrogate Modelling

The designs and their respective objective values are used to construct GPR surrogate models, which are subsequently utilized by the optimizer algorithm. Figure 5.1 illustrates

the prediction performance of surrogates trained on different sampling sizes in terms of MAPE. The MAPE values presented are the results of five-fold cross-validation. In this procedure, the training set is divided into five subsets; the surrogate is trained on four subsets, and the MAPE is recorded for the remaining subset. This process is repeated five times and the error values shown in Figure 5.1 represent the average of all subsets.

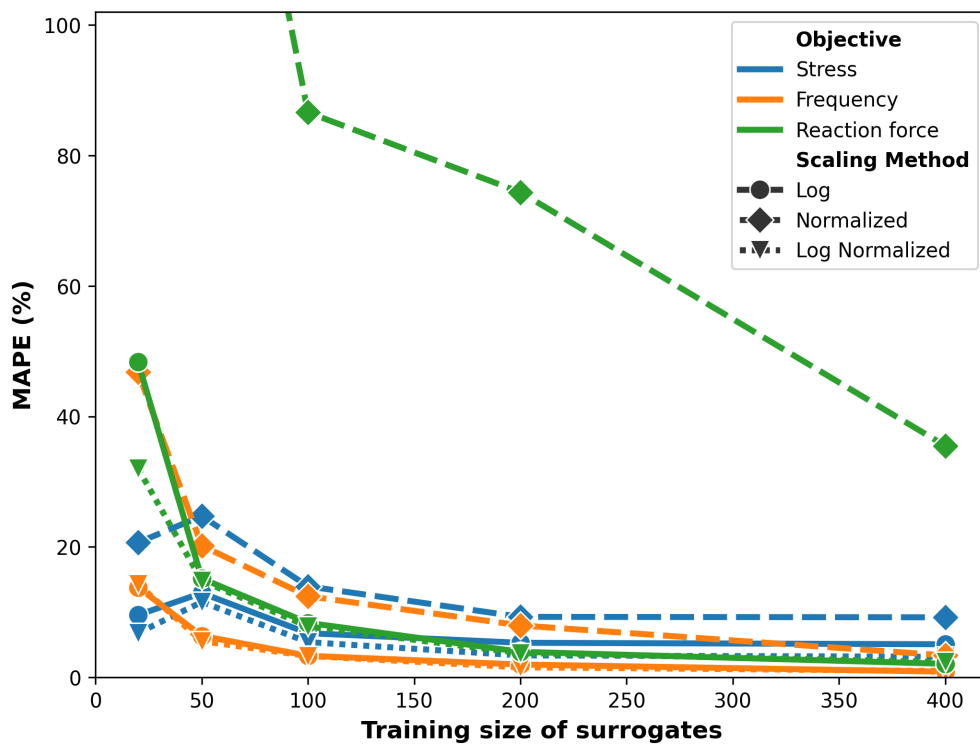


Figure 5.1: MAPE of surrogate models with different training sizes and preprocessing for the U-shaped Lorentz force actuator

The y-axis is capped at 100% to prevent data points with large MAPE values from reducing the visibility of the better-performing lines, which would make interpreting the results more challenging.

The error values generally show a decreasing trend as the training size increases. This aligns with intuition, as a larger dataset provides more information for the model to learn from, improving its performance.

Surrogates trained on normalized objectives have been found to perform the worst, particularly for the reaction force. Even the best MAPE in this case is 35.5%, which is considered completely unacceptable. For the other two objectives, the error values are significantly lower, although they remain noticeably higher compared to the other two scaling methods.

The prediction performance of surrogates trained using log scaling and a combination of log scaling with normalization are quite similar. However, the error tends to be lower when normalization is applied. Consequently, only the surrogates trained on log-scaled and normalized objectives will be used in the optimization procedure.

Applying a logarithmic transformation to input features and objectives in GPR can indeed be beneficial in certain scenarios, particularly when the data exhibit exponential growth patterns or significant variance. A log transformation stabilizes variance and often makes relationships between variables more linear, aligning better with the assumptions of GPR and potentially improving model performance. However, it is important to emphasize that log transformation is not a requirement for using GPR. Its effectiveness depends on the specific characteristics of the data.

Another reason for the improved results observed after applying a log transformation could be the large range of objective values related to the physical characteristics of the actuator. By compressing these ranges, the log transformation can make optimization and model training more efficient, leading to better performance.

Figure 5.2 presents the actual versus predicted plot of objective values for the surrogate model trained using the best-performing scaling method. A training size of 200 is selected for illustration. Similar figures for other sampling sizes are provided in section B.1. This figure clearly demonstrates that the accuracy of the best-performing surrogate model is sufficiently high, making it a viable alternative to FEM simulations.

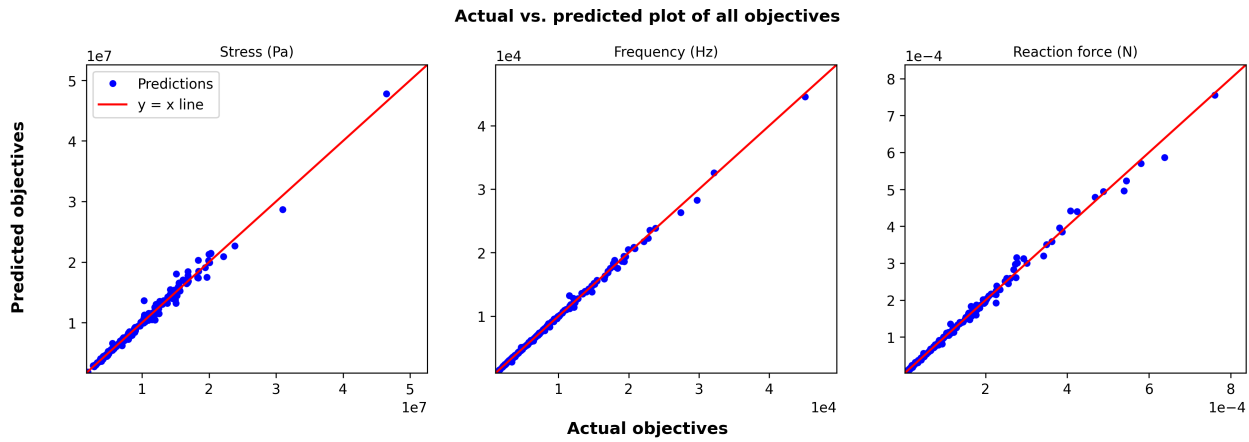


Figure 5.2: Actual vs predicted plot for the U-shaped Lorentz force actuator with a training size of 200

5.3 | Optimization

After optimization is conducted using both SAMOO and A-SAMOO, the resulting solutions are evaluated through simulation to assess their quality. Since the objectives used in the optimization process are obtained through surrogate models, a final FEM re-evaluation is necessary to accurately compare different sets of solutions. Even if direct comparison is not the main focus of the study, re-evaluating at least a subset of solutions that are of particular interest before fabrication is a good approach.

All the parameters used in SAMOO were previously provided in chapter 4. For A-SAMOO, the adaptive sampling parameters include one FEM evaluation per generation and a standard deviation threshold of 0.0001. The rationale for limiting FEM evaluations to one per generation was to minimize the total number of FEM simulations required. Regarding the standard deviation threshold, the value of 0.0001 was determined through a trial-and-error process based on the fact that objectives are log-normalization at this stage. This value was chosen because it was found that it is significantly larger than the standard deviation of the training samples, thereby ensuring that training points are not redundantly resampled.

As discussed in section 4.2, this threshold also ensures that only points with inaccuracies exceeding an acceptable level are evaluated using FEM during the optimization process. The choice of 0.0001 reflects the fact that the log-scaled objectives fall between 1 and 2. If the objectives were scaled differently, the threshold would require adjustment to a value that aligns appropriately with the new scales.

Figure 5.3 shows the actual versus predicted plots after optimization for surrogates trained on 200 designs using SAMOO. Figures for other sampling sizes are provided in section B.1. As seen in the figure, the MAPE is significantly higher compared to Figure 5.2.

This is likely because the optimal solutions are concentrated in a small portion of the design space, and there are insufficient training points in that region. As discussed in subsection 4.2.1, the initial sampling is conducted across the entire design space, which results in many data points contributing to higher model accuracy in unoptimized regions, while the promising areas remain undersampled. A straightforward solution

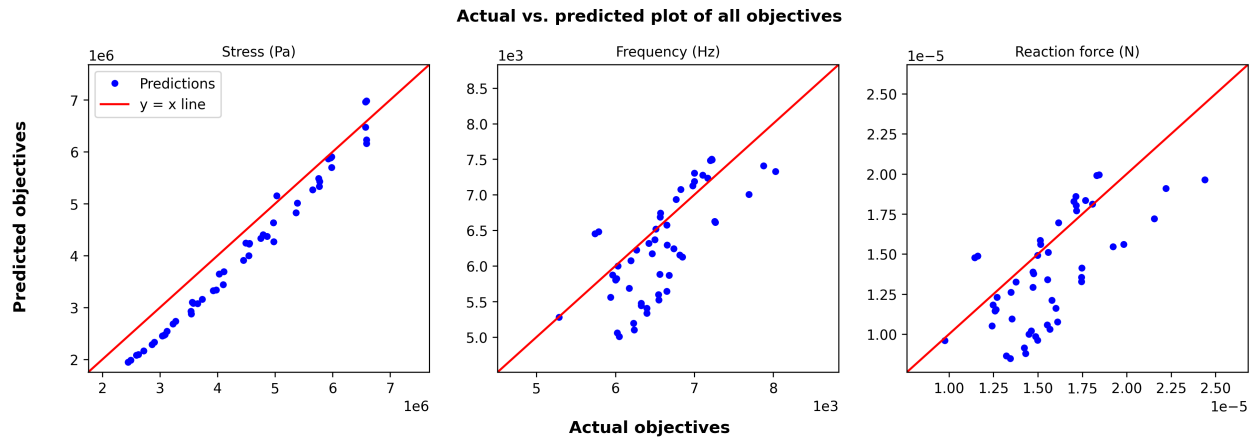


Figure 5.3: Actual vs predicted plot for the U-shaped Lorentz force actuator with a training size of 200: After SAMOO optimization

could be to increase the training size; however, this would increase computation time and still leave a relatively small number of samples in the most promising regions.

For A-SAMOO, the results differ significantly, as shown in Figure 5.4. Utilizing only 50 additional FEM evaluations during the optimization loop has markedly improved model accuracy. The MAPE values are just 0.87%, 0.47%, and 1.02% for stress, resonant frequency, and reaction force, respectively.

The model has adaptively updated itself in promising regions that could benefit from higher accuracy. Another important outcome is that the model is less vulnerable to being misled by inaccurate surrogate predictions, thereby reducing the risk of being directed into the sub-optimal regions.

Figure 5.5 demonstrates the MAPE of all objectives after the optimization for both SAMOO and A-SAMOO. As previously shown, the error for A-SAMOO is significantly lower and remains nearly constant across all training sizes. This is likely because the

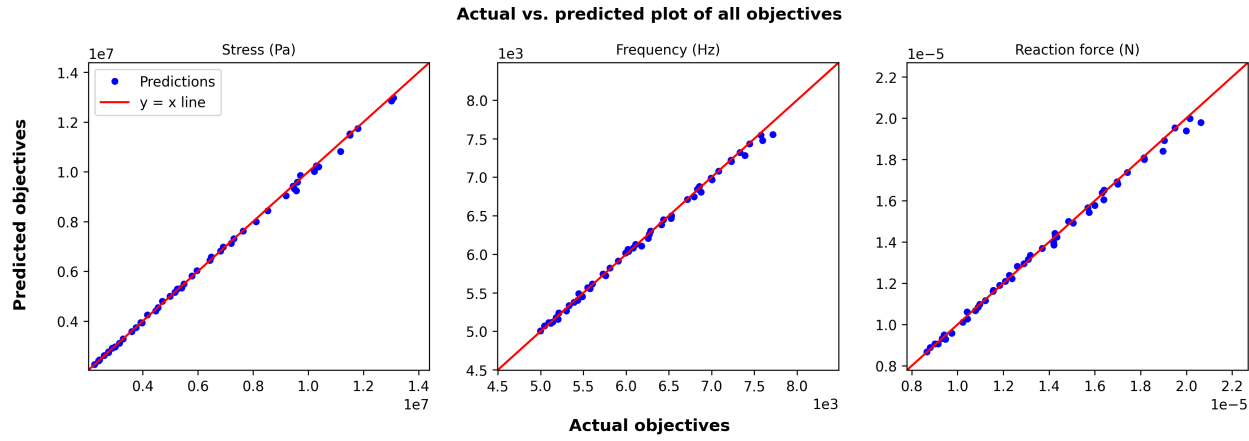


Figure 5.4: Actual vs predicted plot for the U-shaped Lorentz force actuator with a training size of 200: After A-SAMOO optimization

50 additional samples evaluated during the optimization loop were highly effective in improving the surrogate model's accuracy, making the initial accuracy of the surrogate less critical. It is also important to note that A-SAMOO, even with the smallest surrogate model (size 20), still performed significantly better in terms of accuracy compared to the largest model used in SAMOO which was trained on 20 times more data.

Another important aspect of optimization result quality is the number of optimal designs produced. As mentioned in section 4.3, the population size for the NSGA-II optimizer was set to 50. Both SAMOO and its adaptive variant predicted that all the solutions they returned are non-dominated and compliant with objective constraints. However, Figure 5.6, which depicts the number of optimal solutions suggests otherwise.

The term "optimal" refers to the union of solutions that satisfy objective constraints and are non-dominated with respect to each other. Once again, A-SAMOO performs significantly better, as the number of its optimal solutions is very close to the maximum

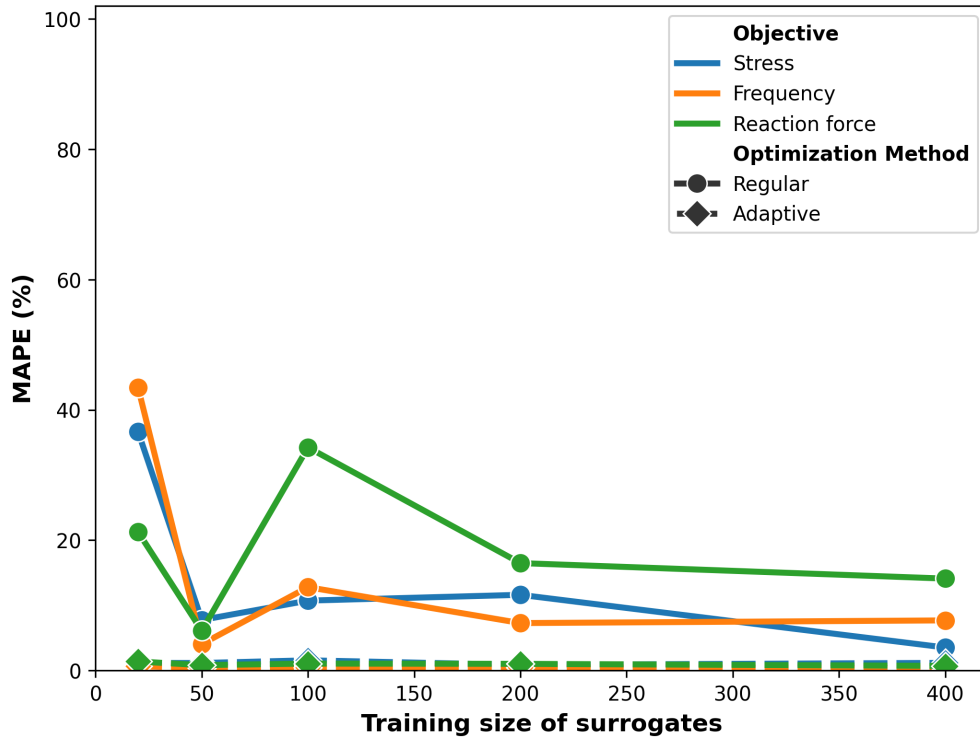


Figure 5.5: MAPE after optimization for the U-shaped Lorentz force actuator

possible, which is the population size of 50. In contrast, the numbers are lower for SAMOO. This discrepancy is due to the lower prediction quality of its surrogate model, which led to the generation of samples that were falsely deemed compliant with objective constraints or non-dominated.

The hypervolume quality metrics for the solution sets of both algorithms are shown in Figure 5.7. The "Predicted" values represent the hypervolume measured immediately after optimization, while the "True" values correspond to the results after FEM re-evaluation. This plot can be interpreted from two perspectives.

Firstly, for each scheme, the predicted hypervolume is greater than the actual value,

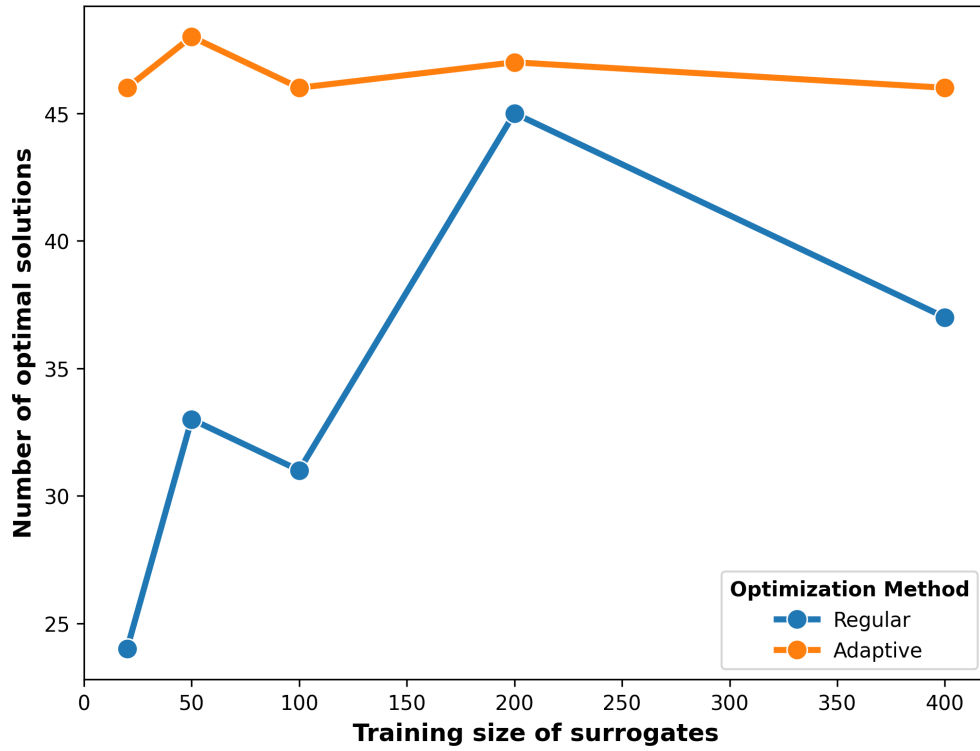


Figure 5.6: Number of true optimal solutions for the U-shaped Lorentz force actuator

indicating that the optimizer overestimated the quality of its solutions. However, the discrepancy is much larger for SAMOO, particularly for the smallest training size. In contrast, the true and predicted hypervolumes of A-SAMOO are almost overlapping, suggesting that online learning significantly aids the optimizer in providing a more accurate estimate of its results.

Secondly, regarding the hypervolume values themselves, those for A-SAMOO are consistently higher, indicating that the quality of its solutions is superior. Similar to the observations in Figure 5.6, even the smallest adaptive model outperforms the largest surrogate model used in the other algorithm.

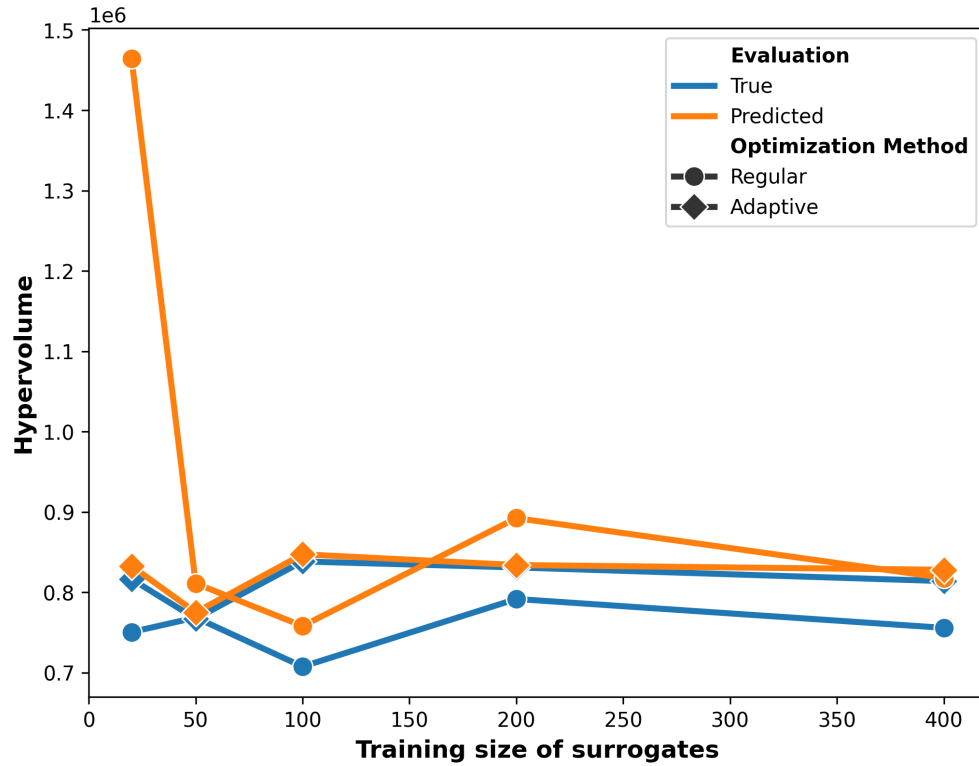


Figure 5.7: Hypervolume of predicted solutions for the U-shaped Lorentz force actuator

5.4 | Summary

This chapter focused on the optimization results for the U-shaped Lorentz force actuator, highlighting the performance of different optimization strategies. Surrogate models were constructed using various scaling methods and were evaluated based on their prediction accuracy. The results indicated that surrogates trained using a combination of log scaling and normalization provided superior accuracy across different training sizes. Models trained on normalized objectives alone performed poorly, particularly in predicting reaction force, where the error rates were unacceptably high.

The optimization results from SAMOO and A-SAMOO showed notable differences in solution quality. A-SAMOO achieved significantly lower error rates and maintained consistent prediction accuracy across all training sizes, even with the smallest surrogate model. In terms of the number of true optimal solutions, A-SAMOO performed better, producing solutions that were closer to the maximum possible count. In contrast, SAMOO had larger discrepancies between predicted and actual values, especially regarding hypervolume metrics. The adaptive approach in A-SAMOO, which included additional FEM evaluations during the optimization loop, allowed it to adapt more effectively to promising regions, resulting in higher-quality optimization.

Overall, the results demonstrated that A-SAMOO significantly enhanced optimization performance, providing more accurate predictions and a greater number of optimal solutions compared to the traditional SAMOO approach. This underscores the value of adaptive sampling strategies in improving the quality and reliability of surrogate-assisted optimization processes.

Lorentz Force Actuator with Springs Results

6.1 | Initial Sampling

The minimum, maximum, and average objective values of sampled designs for the Lorentz force actuator with springs are listed in Table 6.1. Similar to section 5.1, the dataset from which these values were calculated is constructed by combining all initial sampling datasets with varying sizes. The average von Mises stress is only 3.82 MPa , almost 2.6 times lower than that of the U-shaped actuator. For the resonant frequency, values are generally lower compared to those reported in Table 5.1; although the difference is not significant. This was predictable as the Lorentz force actuator with springs is much more flexible, resulting in lower frequencies. Although the minimum and maximum reaction forces found are in the vicinity of their respective values for the U-shaped actuator, interestingly, the average is almost one order of magnitude lower.

Temperature, current, and voltage were not optimization objectives for the simpler

U-shaped actuator, so no comparison will be reported for these parameters. The temperature difference range for the Lorentz force actuator with springs is quite broad, spanning from less than 1 K to over 1100 K, with an average temperature of 28.5 K. Current and voltage values are typically in the mA and mV ranges, respectively. The computation times are significantly higher for this actuator, particularly in terms of maximum time, which is nearly seven times greater than that for the U-shaped Lorentz force actuator. This increase was expected, given the added complexity of the model due to the inclusion of an additional study for thermal modelling.

Table 6.1: Objective ranges and computation times of initial samples for Lorentz force actuator with springs

Parameter	Unit	Minimum	Maximum	Average
Stress	Pa	2.14×10^5	2.54×10^7	3.82×10^6
Resonant frequency	Hz	3.74×10^2	1.47×10^4	2.52×10^3
Reaction force	N	1.00×10^{-6}	1.73×10^{-4}	1.47×10^{-5}
Temperature difference	K	1.89×10^{-2}	1.15×10^3	2.85×10^1
Current	A	5.70×10^{-4}	2.88×10^{-1}	1.60×10^{-2}
Voltage	V	1.37×10^{-3}	7.89×10^{-1}	4.39×10^{-2}
Simulation time	s	8.91	67	26.5

6.2 | Surrogate Modelling

Mean absolute percentage error values for surrogates of different training sizes and objective preprocessing methods are plotted in Figure 6.1. The trend is similar to that of the previously studied U-shaped actuator. Surrogates trained on normalized objective values constructed the worst-performing models. In contrast, the other two preprocessing methods—log scaling and a combination of log scaling with normalization—produced models with significantly lower MAPE, performing much better than mere normalization.

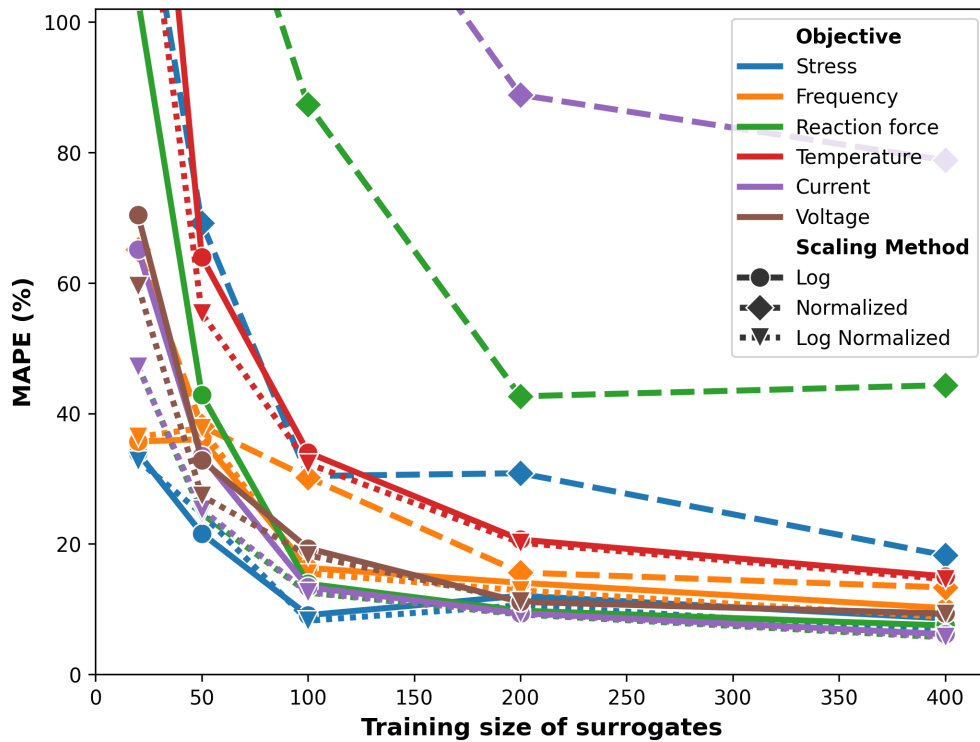


Figure 6.1: MAPE of surrogate models with different training sizes and preprocessing for the Lorentz force actuator with springs

It can be concluded that log scaling of objectives significantly enhances the prediction performance of surrogate models. As with the U-shaped actuator, normalized reaction force values were the most challenging objective to predict accurately.

There are some notable distinctions between the error plots of the U-shaped and complex Lorentz force actuators, Figure 5.1, and Figure 6.1, respectively. Firstly, for the actuator with springs, the error values are generally higher across all cases. This behaviour can be attributed to the increased complexity of the actuator with springs, which requires more data points for the surrogate model to accurately capture the underlying mechanics. Secondly, for the actuator with springs, models performed best in predicting reaction forces when trained with log-scaled plus normalized data. In contrast, for the simpler U-shaped actuator, resonant frequency was the easiest to predict. This difference is likely because adding springs to the structure introduces complexities to frequency modelling, more so than to stress and reaction force.

As shown in Figure 6.2, the actual versus predicted plot of objectives for the surrogate trained on 200 samples indicates higher MAPE for the more complex structure. However, most data points still lie close to the $y = x$ line, indicating reasonable predictive accuracy despite the increased complexity. Please note that due to larger range of objectives, especially temperature and voltage, axes are plotted in log-scale. Using unscaled axes, as it can be seen in Figure 6.3, to assess the accuracy of surrogates can be deceiving. Similar plots for other sampling sizes can be found in section B.2.

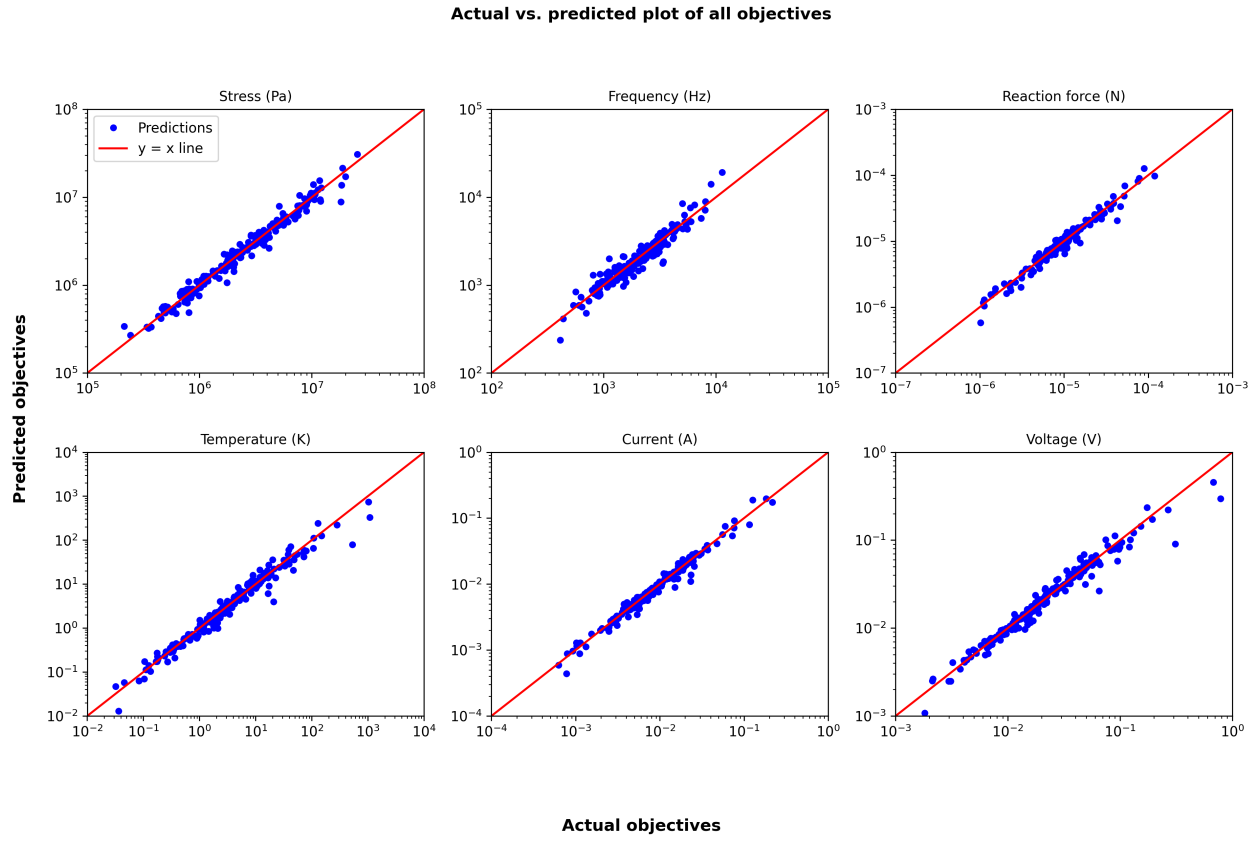


Figure 6.2: Actual vs predicted plot (log scale) for the Lorentz force actuator with springs with a training size of 200

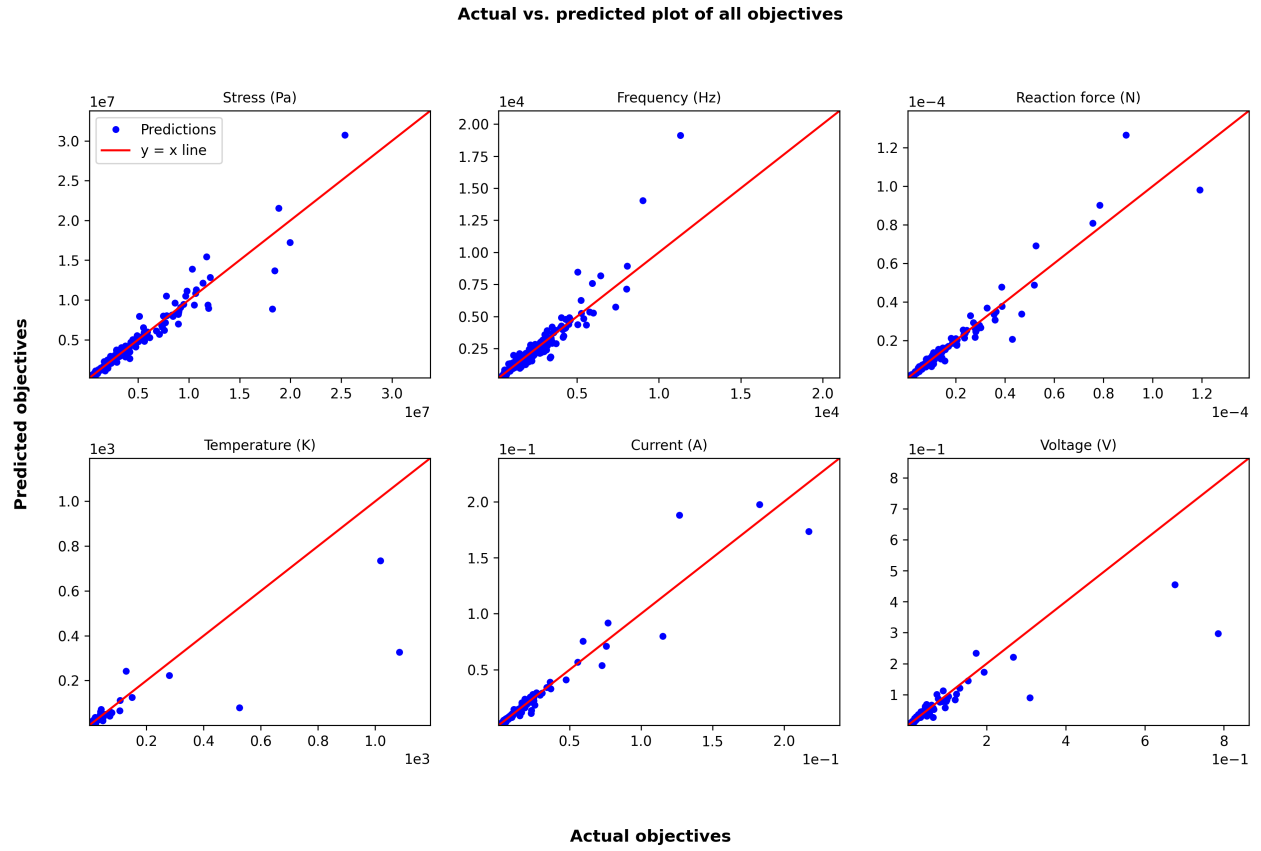


Figure 6.3: Actual vs predicted plot for the Lorentz force actuator with springs with a training size of 200

6.3 | Optimization

After re-evaluating the results of both methods using COMSOL, their respective prediction plots are presented in Figure 6.4 and Figure 6.5. From Figure 6.4, it is evident that the prediction accuracy for the Lorentz force actuator with springs is significantly worse than that of the U-shaped actuator. The decline in predictive performance is particularly severe for stress and frequency. This poor prediction capability can be attributed to several factors.

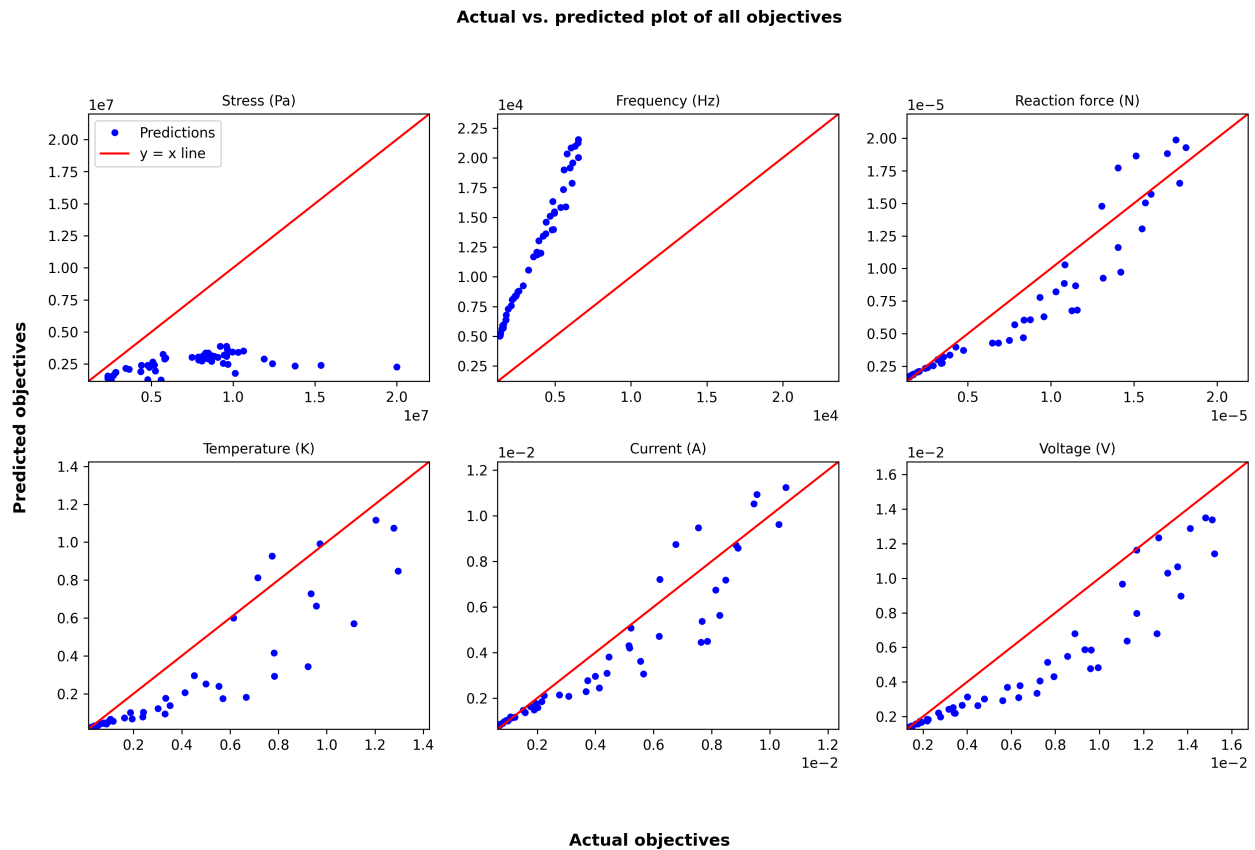


Figure 6.4: Actual vs predicted plot for the Lorentz force actuator with springs with a training size of 200: After SAMOO optimization

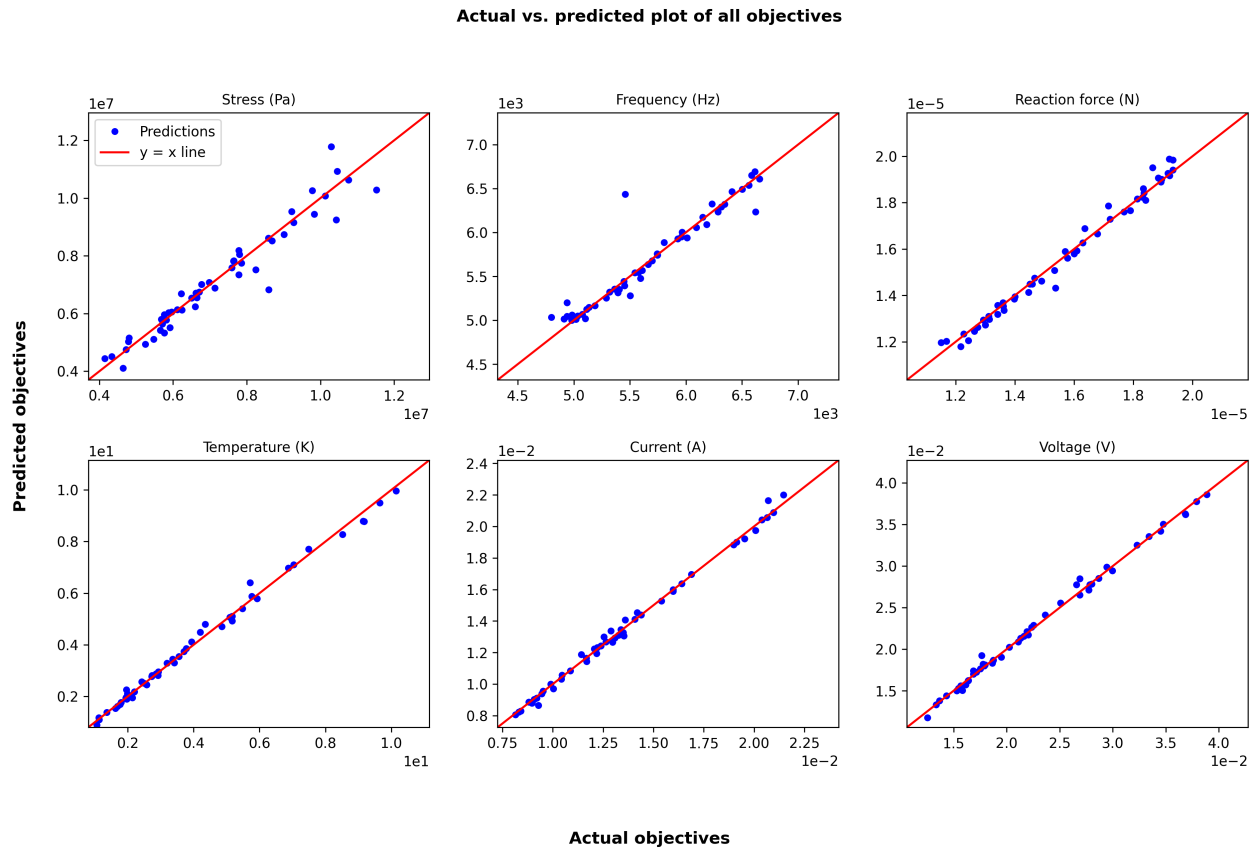


Figure 6.5: Actual vs predicted plot for the Lorentz force actuator with springs with a training size of 200: After A-SAMOO optimization

Firstly, the optimization process begins with surrogates of lower quality compared to those used for the simpler actuator. Secondly, the addition of one more design variable and three objectives significantly expands both the design and objective spaces, particularly for the latter. As a result, the surrogates require a substantially larger number of samples to make reasonable predictions. This can also be inferred from the fact that a larger objective space, with the same number of samples, is less likely to have sufficient data points in any given subsection of the space.

However, the results for A-SAMOO are far more robust against these effects. As shown in Figure 6.5, the data points remain very close to the $y = x$ line, indicating much higher surrogate performance.

These findings are also consistent with the MAPE values illustrated in Figure 6.6. The error values for SAMOO solution sets are so high that the predictions could be considered almost meaningless. In contrast, A-SAMOO has successfully generated design and objective pairs that closely align with their true values, although the errors are still larger than those reported in Figure 5.5.

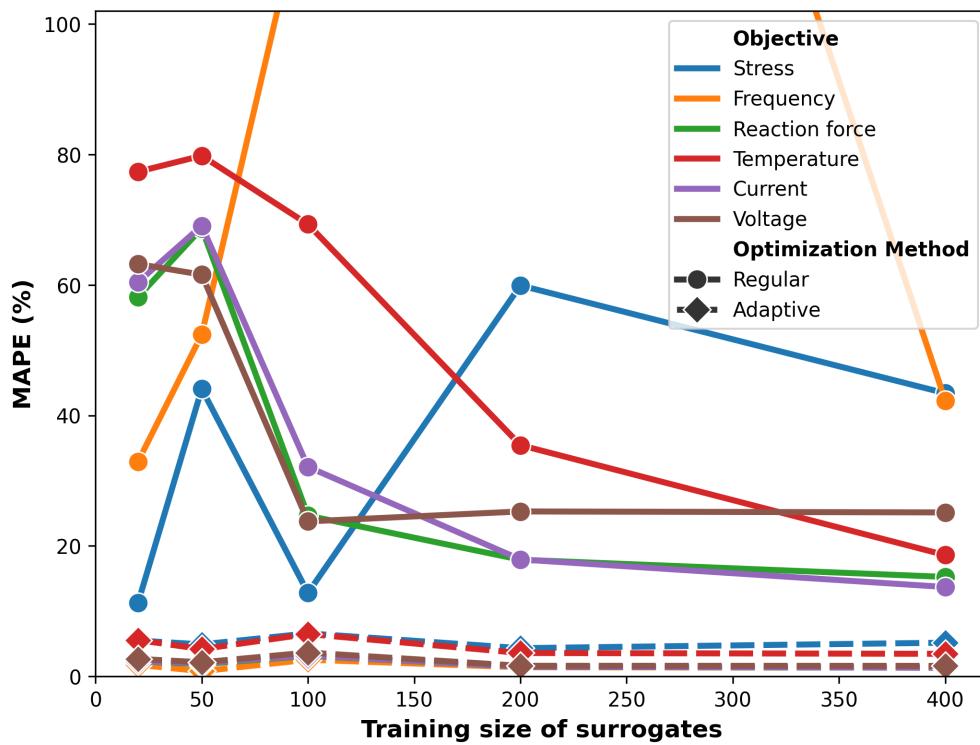


Figure 6.6: MAPE after optimization for the Lorentz force actuator with springs

The number of optimal solutions found using both optimization strategies is presented in Figure 6.7. Once again, A-SAMOO significantly outperformed the regular SAMOO approach, yielding a much higher number of desirable solutions. In the case of SAMOO, even with the largest dataset, the number of optimal solutions was less than half the population size.

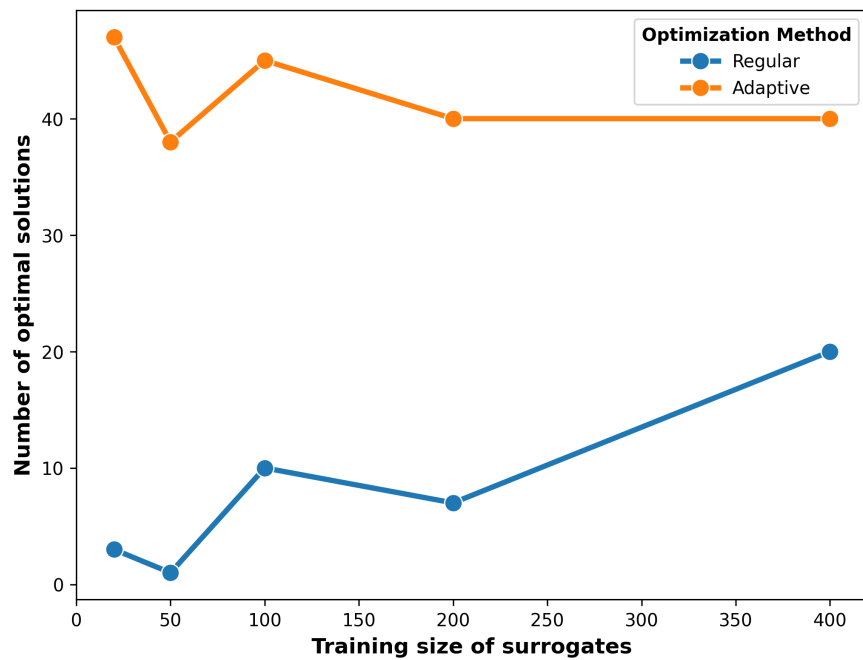


Figure 6.7: Number of true optimal solutions for the Lorentz force actuator with springs

The hypervolume plot, shown in Figure 6.8, further substantiates the superior solution quality of A-SAMOO. In cases where online learning was not employed, the predicted hypervolume was completely different from the actual value, indicating that the optimizer significantly overestimated its performance. In contrast, the true and predicted hypervolumes for the adaptive mechanism were much closer than those

reported in Figure 5.7. Consistently, these values are larger than those of SAMOO with the same number of training size. Despite the challenges posed by a more complex model with a greater number of objectives and design variables, A-SAMOO successfully delivered high-quality optimization results.

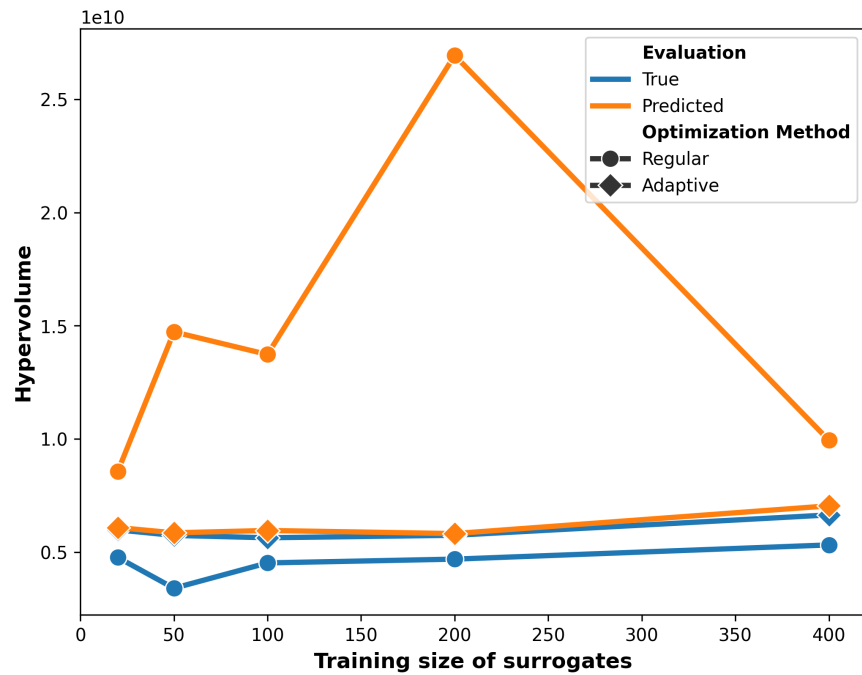


Figure 6.8: Hypervolume of predicted solutions for the Lorentz force actuator with springs

6.4 | Summary

This chapter explored the optimization results for the Lorentz force actuator with springs, focusing on how surrogate models and optimization techniques performed with this more complex design. The initial sampling revealed distinct characteristics: the actuator with

springs had a notably lower average von Mises stress than the U-shaped actuator, while resonant frequencies were lower due to the increased flexibility. Reaction forces, though similar in their minimum and maximum values to the U-shaped actuator, showed a much lower average. The inclusion of additional objectives introduced further complexity to the model, resulting in longer computation times.

Analysis of surrogate model performance highlighted that normalization alone led to the poorest predictions, particularly for challenging objectives like reaction force. Surrogates trained with log scaling or a combination of log scaling and normalization demonstrated superior predictive accuracy. However, errors remained higher than those observed for the U-shaped actuator, underscoring the increased complexity and data requirements of the spring-based design. When evaluating optimization performance, A-SAMOO proved to be more effective than SAMOO, delivering a significantly greater number of true optimal solutions and more accurate predictions. The hypervolume metric further emphasized the advantage of the adaptive approach, with A-SAMOO showing larger values and a closer match between predicted and actual hypervolume, indicating its effectiveness in handling the larger design and objective spaces.

Thermal actuator results

7.1 | Initial Sampling

Objective ranges for the thermal actuator are summarized in Table 7.1. Unlike the Lorentz force actuators, the thermal actuator experiences significantly higher stress levels, with maximum values reaching up to 2.82 GPa . This level of stress could potentially exceed the yielding strength of polycrystalline silicon. Also, the stress range for this actuator is the largest observed so far, spanning more than six orders of magnitude.

Resonant frequencies for the thermal actuator are predominantly in the hundreds of kHz range. However, given the distinct structural and functional characteristics of this actuator compared to previous designs, direct comparisons of resonant frequency values are not particularly meaningful.

Deformation is a primary objective of interest for the thermal actuator, as generating a displacement is the main application of the actuator. The average deformation observed is $5.29 \text{ }\mu\text{m}$, which is marginally above the defined constraint of $5 \text{ }\mu\text{m}$, suggesting that

optimizing this parameter could be particularly challenging.

The temperature difference range is notably broad, extending from 20 K to an extreme of 18,400 K. Such high temperature values are invalid and result from the non-linearities not being accounted for in the thermal modelling. With an average temperature almost an order of magnitude higher than the constraint, the optimizer will likely face significant challenges in improving temperature values to remain within acceptable limits. Current values and computation times do not exhibit any remarkable characteristics.

Table 7.1: Objective ranges and computation times of initial samples for Thermal actuator

Parameter	Unit	Minimum	Maximum	Average
Stress	Pa	8.50×10^3	9.88×10^9	2.82×10^9
Resonant frequency	Hz	2.27×10^4	3.43×10^6	4.07×10^5
Deformation	μm	5.00×10^{-6}	1.32×10^2	5.29
Temperature difference	K	2.00×10^1	1.84×10^4	6.15×10^3
Current	A	1.30×10^{-5}	3.76×10^{-1}	2.90×10^{-2}
Simulation Time	s	4.44	29.1	22

7.2 | Surrogate Modelling

Figure 7.1 illustrates MAPE for surrogate models trained on various training sizes for the thermal actuator. In this case, the choice of scaling method has a significant impact on the predictive performance of the surrogate models. As observed previously, a combination

of log scaling and normalization has once again proven to be the most effective scaling strategy.

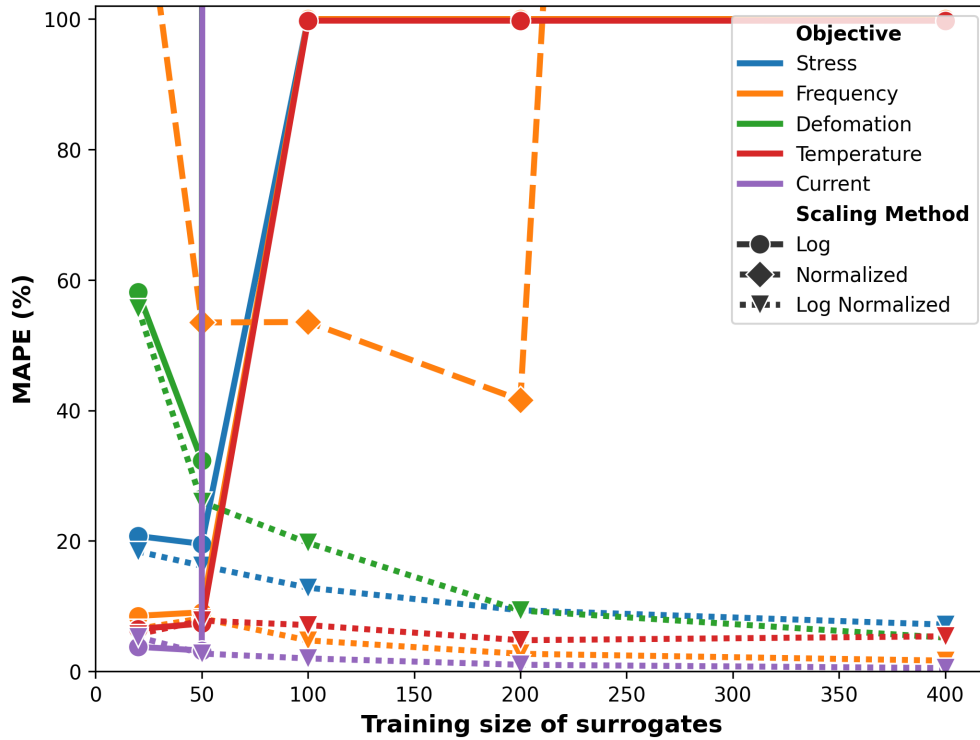


Figure 7.1: MAPE of surrogate models with different training sizes and preprocessing for the Thermal actuator

A key distinction from the similar plots for the previous two actuators (Figure 5.1 and Figure 6.1) is that log scaling alone does not perform as well as the combined scaling method for the thermal actuator. From the training sizes of 100 and larger, GPR models provided with log scaled or normalized dataset failed to converge. This discrepancy is likely due to the considerably larger ranges for the objectives. With the exception of resonant frequency, all other objectives have ranges that extend at least three orders of

magnitude, with deformation spanning up to eight orders of magnitude. While the exact reasons behind the significant effect of normalization for this actuator are not immediately clear, it is evident that the vast range of values plays a critical role. Further studies would be required to fully understand why normalization has a more pronounced impact on the performance of surrogate models for this particular actuator.

Actual vs. predicted plots of objectives for surrogates with the training size of 200 are provided in Figure 7.2. These plots also emphasize the prediction capabilities of

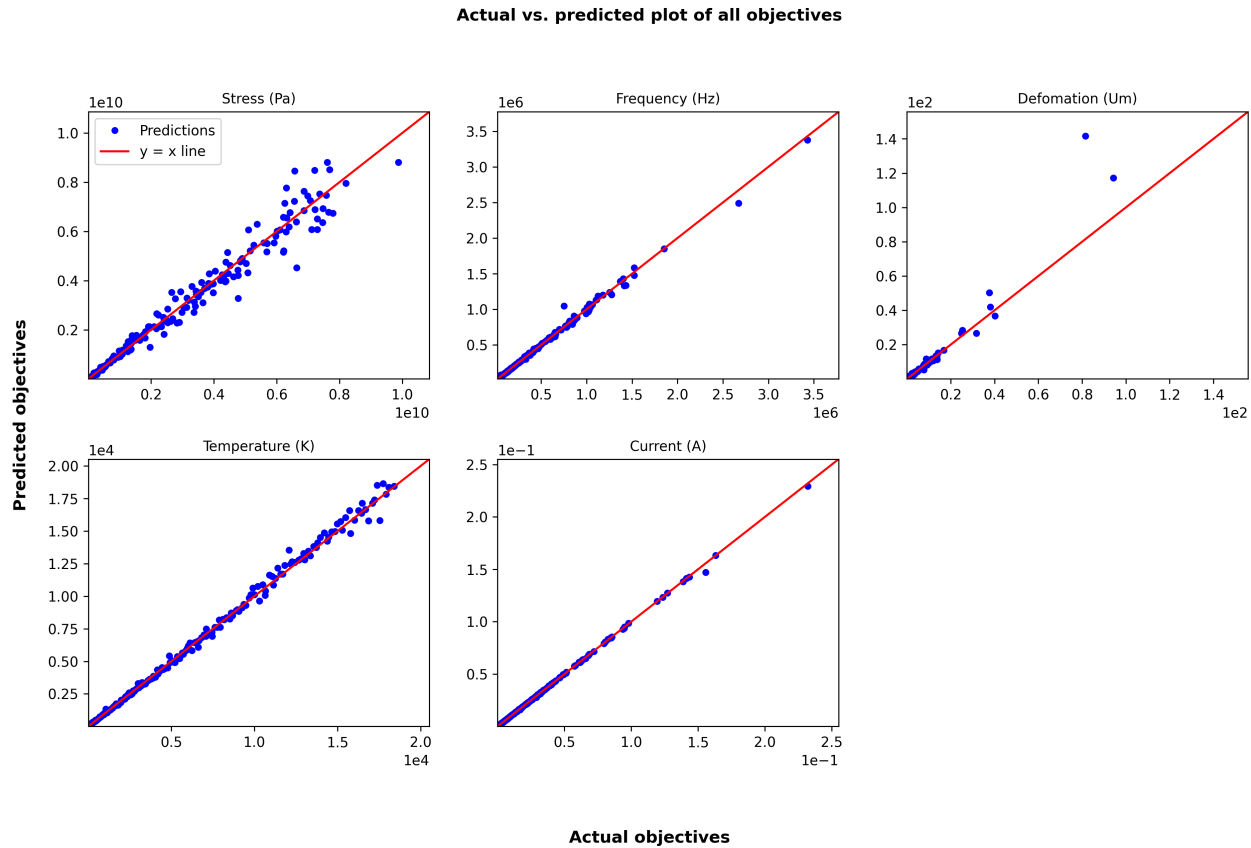


Figure 7.2: Actual vs predicted plot (log scale) for the Thermal actuator with a training size of 200

surrogates trained on objectives that are both log scaled and normalized. The only noticeable takeaway is that current predictions are remarkably accurate. Similar plots for other sampling sizes can be seen in section B.3.

7.3 | Optimization

Prediction plots for solutions of both optimization methods, SAMOO and A-SAMOO, are illustrated in Figure 7.3 and Figure 7.4, respectively.

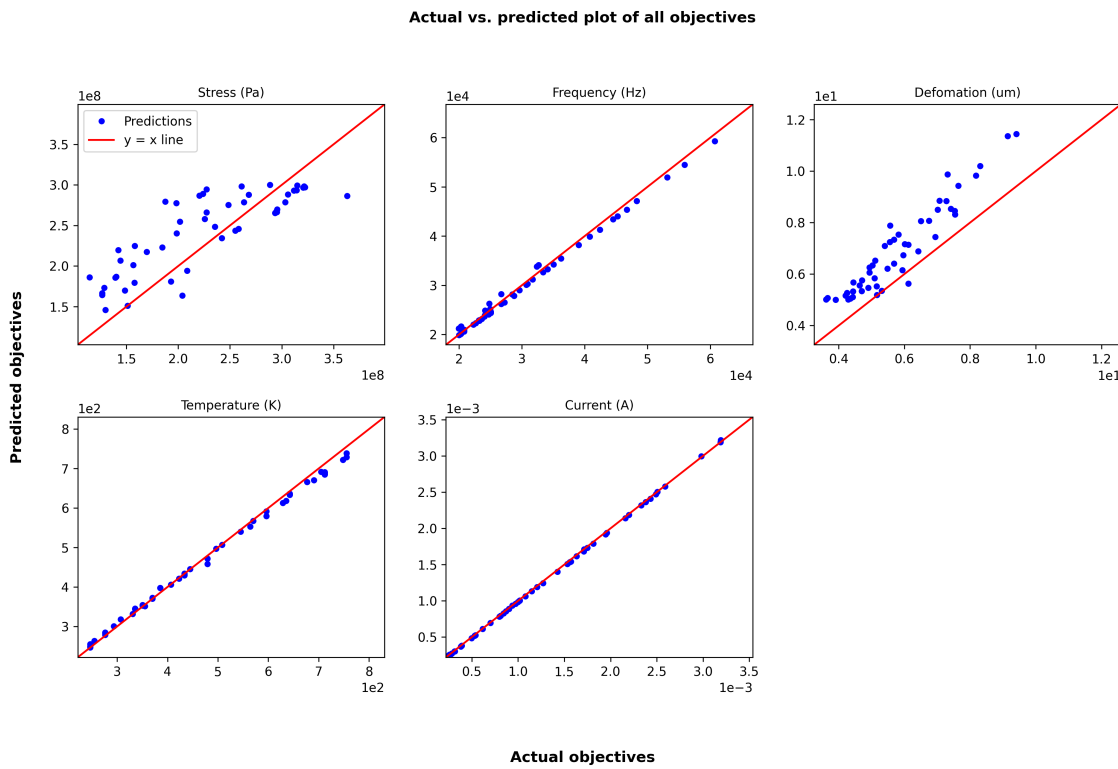


Figure 7.3: Actual vs predicted plot for the Thermal actuator with a training size of 200: After SAMOO optimization.

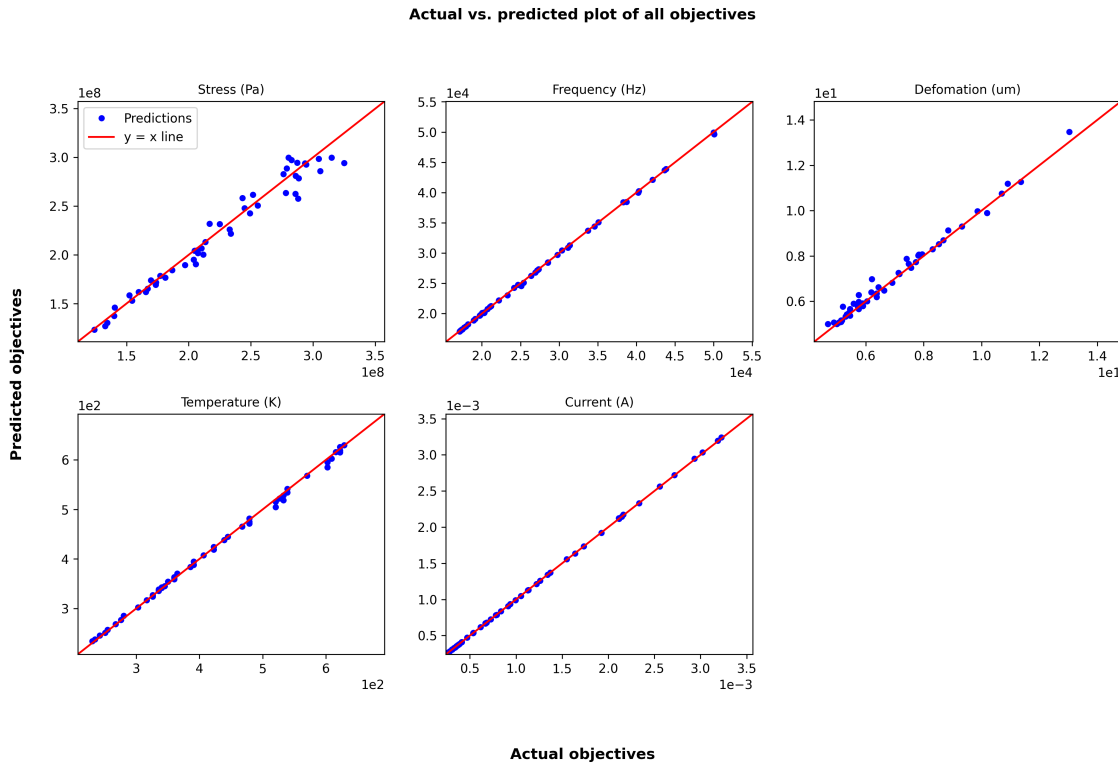


Figure 7.4: Actual vs predicted plot for the Thermal actuator with a training size of 200: After A-SAMOO optimization.

In both cases, resonant frequency, temperature and current are predicted very accurately. For stress and deformation, however, that is not the case as understandably, adaptive optimization has resulted in far better predictions.

MAPE plot shown in Figure 7.5 reveals interesting insights. When the initial training size of surrogates is 20 samples, the optimizer is unable to provide even a single optimal design; making reporting of MAPE in that case irrelevant. This is the first time in this study that such limitation is encountered; even for the adaptive version of optimization. Although it cannot be argued with certainty, this is most likely due to the fact that for

stress and deformation objective values are generally much worse than the constraints (refer to Table 7.1.) Consequently, a poor surrogate can easily lead the optimizer stray.

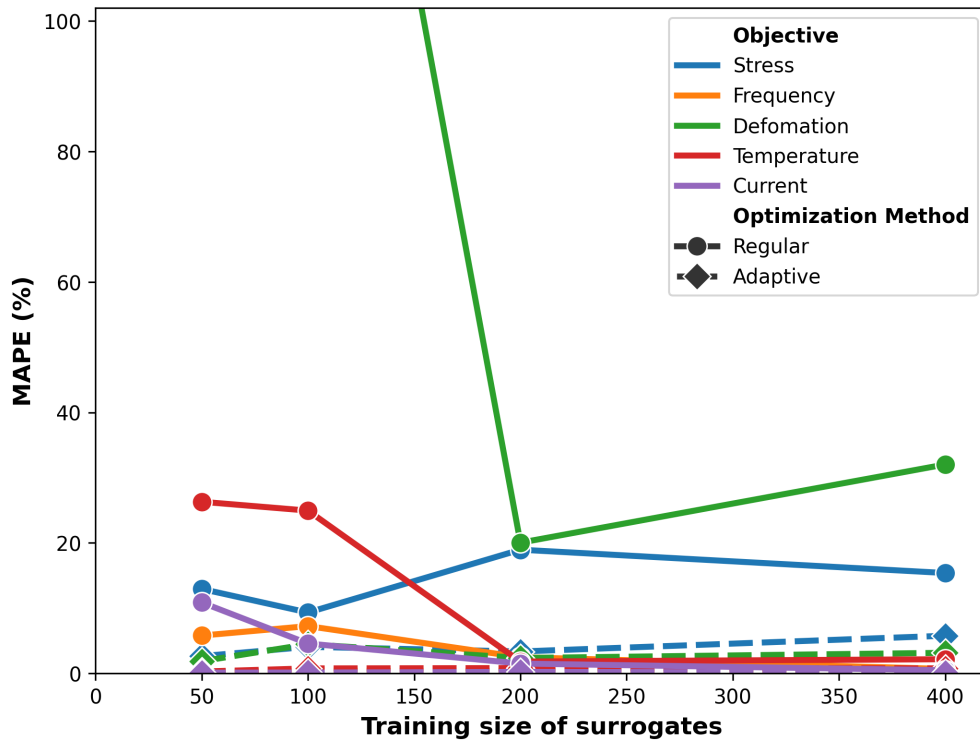


Figure 7.5: Mean Absolute Percentage Error (MAPE) after optimization for the Thermal actuator

Figure 7.6 provides a plot of the number of optimal solutions against the training sizes of surrogates used in the optimization. The results are consistent with previous findings, demonstrating that online training of surrogates in the A-SAMOO technique significantly aids the NSGA-II optimizer in generating a greater number of optimal solutions compared to traditional offline learning.

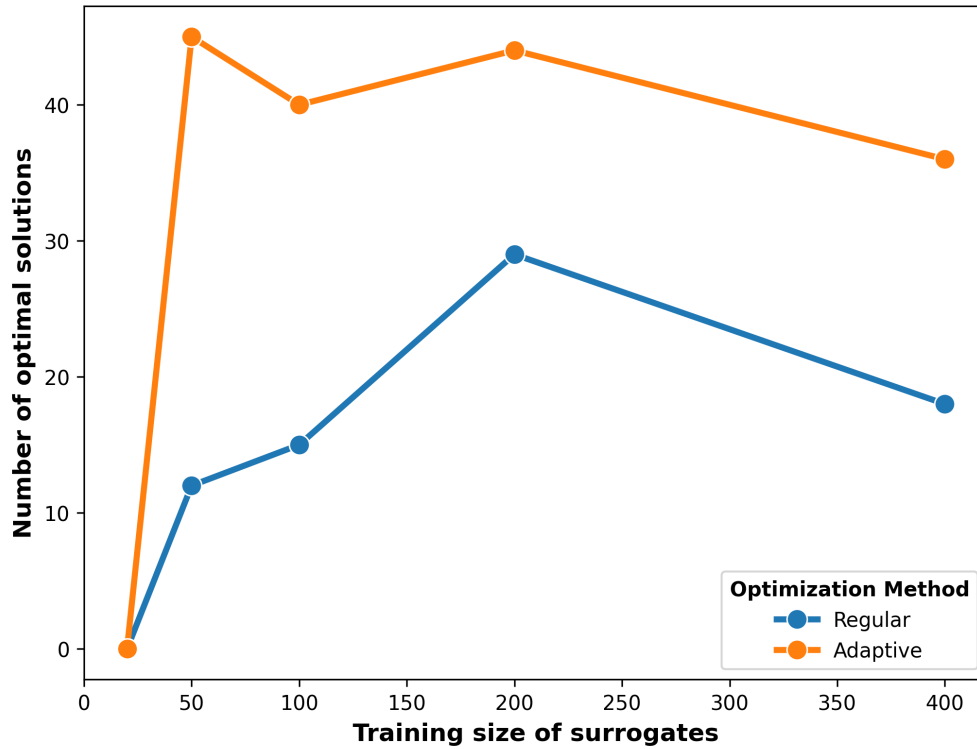


Figure 7.6: Number of true optimal solutions for the Thermal actuator

The hypervolume of the solution sets, depicted in Figure 7.7, further corroborates the benefits of adaptive surrogate modelling. For most training sizes, the discrepancy between actual and predicted hypervolume values is quite large for SAMOO, indicating an overestimation of solution quality when using static, offline-trained surrogates. However, an interesting anomaly appears when the surrogate training size is 400: the actual hypervolume for regular optimization is slightly higher than that of the adaptive approach. This observation can be explained by considering two factors. Firstly, the difference between the actual hypervolumes of the two approaches is not substantial enough to definitively challenge the performance advantage of A-SAMOO. Secondly, the

inherent inaccuracies of surrogate models can sometimes lead to unpredictable optimizer behaviour.

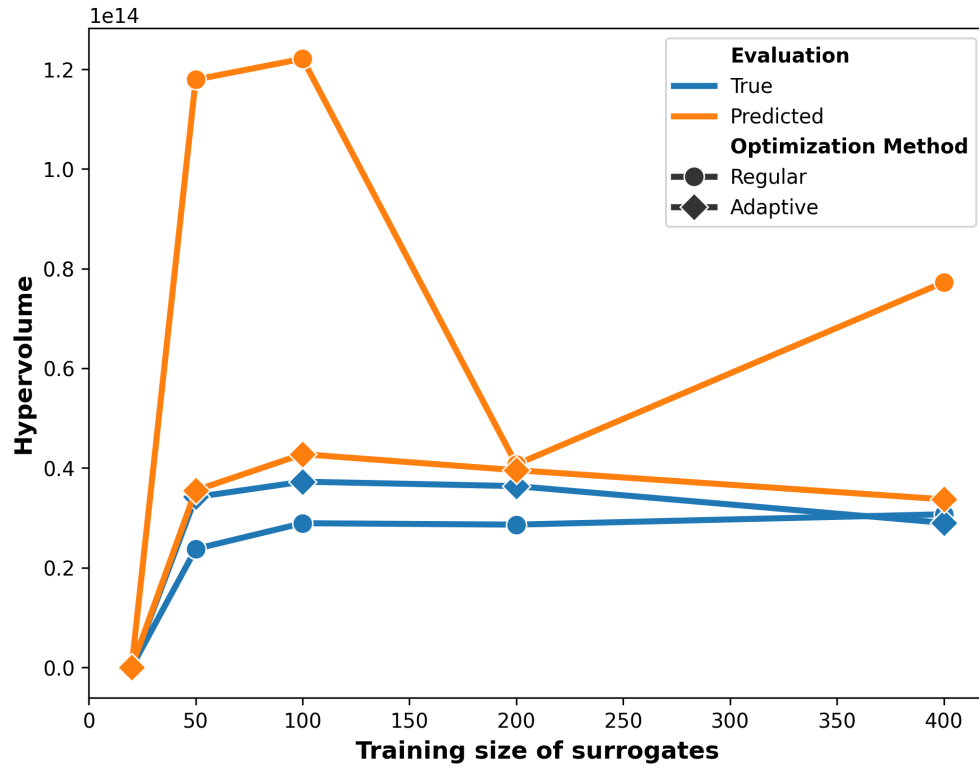


Figure 7.7: Hypervolume of predicted solutions for the Thermal actuator

As seen in the corresponding figures for the Lorentz force actuators, such inaccuracies often result in lower actual hypervolumes. However, in this case, it is likely that the relatively inaccurate surrogate predictions happened to be in a favourable direction for optimization. For example, the actual stress value design was 200 *MPa* and the surrogate predicted a slightly better value of 180 *MPa* (lower stress being the objective). This explanation is further supported by observations across all actuators, where in the actual

vs. predicted plots, some data points fall below the $y = x$ line (or above it, for objectives that are maximized), indicating cases where surrogate predictions are better than the actual values in the context of the optimization goals.

7.4 | Summary

For the thermal actuator, it is observed that a combination of log scaling and normalization consistently delivers the best predictive performance, particularly due to the wide range of objective values. This effect is more prominent compared to Lorentz force actuators as the surrogate models trained with this combined approach show significantly better accuracy than those using log scaling alone.

A-SAMOO approach shows better performance in generating a greater number of optimal solutions. Its prediction accuracy is also much greater compared to SAMOO, particularly for stress and deformation objectives, which are the most challenging objectives to optimize. Due to the complexities of the model, an initial training size of 20 did not result in any optimal design in both optimization cases. Also, at a surrogate training size of 400, regular optimization slightly outperforms adaptive optimization in terms of hypervolume, likely due to favourable inaccuracies in surrogate predictions.

Conclusion

8.1 | Achieved Aims and Objectives

The motivation behind this work, as discussed in chapter 1, stems from identifying major gaps in the literature on surrogate-assisted optimization for MEMS devices. The primary focus of this study has been to develop a comprehensive multi-objective optimization framework specific to MEMS, aimed at addressing these gaps. This framework is designed to be robust, covering most critical aspects of the optimization process, thereby enabling MEMS designers to efficiently utilize the proposed methodology to derive optimized designs suitable for the fabrication process.

The objectives of this study can be considered as achieving the contribution goals outlined earlier. The contributions claimed in section 1.3 are revisited here, with key findings highlighted to support each of these points."

1. **Integration of Adaptive Optimization:** Unlike the majority of previous research, which predominantly relied on classic offline optimization where surrogate models

remain static throughout the process, this work implemented both regular (SAMOO) and adaptive (A-SAMOO) optimization approaches. Across all cases, adaptive optimization consistently outperformed regular optimization both in terms of the number of optimal solutions and their quality. This performance boost emphasizes the potential of online learning to significantly improve the optimization process for MEMS devices in terms of accuracy, speed, and diversity of solutions.

2. **Validation Across Multiple Devices:** Another major contribution of this study is the validation across multiple devices. Optimization methods were rigorously tested on three distinct types of actuators: the U-shaped Lorentz force actuator, the Lorentz force actuator with springs at both ends, and the thermal actuator. Each device differed in design variables, objectives, objective constraints, and, notably, complexity. The chosen devices and their parameters were specifically selected to test various aspects of the optimization process. For example, the Lorentz force actuator with springs included a discrete design variable related to the number of spring turns. In contrast, most previous studies primarily considered continuous variables and maintained the overall shape of the devices while only adjusting their dimensions. Despite this added complexity, the provided approach successfully produced high-quality solutions for this device.

Furthermore, the thermal actuator presented a unique challenge due to its stringent objective constraints for stress and deformation, with a large proportion of initial designs failing to meet these limitations. Despite these constraints, the A-SAMOO

strategy was able to generate acceptable optimization solutions. This validation across multiple devices with varying complexities and constraints demonstrates the robustness and versatility of the proposed optimization framework in effectively handling a diverse range of MEMS designs.

3. **Addressing Shortcomings of Traditional Approaches:** As mentioned above, even when strict objective constraints were imposed, the present optimization framework effectively produced optimal designs. This capability contrasts with many prior studies where handling objective constraints is often overlooked or inadequately addressed. The robustness of this framework ensures that constraints are integrated into the optimization process, enhancing the reliability of the resulting designs.

Additionally, this study is capable of providing multiple design options for users at the end of the optimization process. In contrast to some previous studies where multiple objectives were technically optimized, the final output often reduced to optimizing a single value derived through averaging or weighting methods. Such strategies inherently limit the variety of choices available to MEMS designers.

4. **Preprocessing of Design Variables and Objectives:** In much of the existing MEMS surrogate-assisted optimization literature, the effects of preprocessing are largely overlooked. However, it was demonstrated that the scaling of objectives can significantly impact the accuracy of surrogate models. In some cases, the improper selection of scaling methods led to completely unacceptable optimization results, underscoring the critical role of appropriate preprocessing. By carefully choosing

the right scaling techniques, the framework provided here ensures more accurate predictions and hence solutions of higher quality.

5. **Comprehensive Framework:** This study provides a detailed step by step multi-objective optimization procedure for MEMS. There are no such comprehensive studies to the best of the author's knowledge. Hence, the framework introduced here can be used by researchers to enhance their MEMS designs from scratch.

8.2 | Limitations

Some of the limitations of this study are listed as follows:

1. **Selection of solutions** After the optimization process, the responsibility falls on the user, also known as the decision maker (DM), to select the most appropriate design(s). This study does not provide specific guidance on how to make these selections efficiently. For readers interested in strategies for evaluating and selecting optimal solutions, further information can be found in [83].
2. **FEM analysis error** The focus of this study was to develop an optimization framework rather than to achieve the absolute best solutions. Consequently, the simulations were configured for faster run times, potentially sacrificing accuracy. Using finer meshing sizes in the simulation models might yield different results.

3. **Real-world representation of devices** For similar reasons, the physical models used in this study do not fully replicate real-world conditions. For example, the effects of the device substrate are largely ignored in the COMSOL Multiphysics models, which could lead to discrepancies between simulated and real-world performance.

Deliverables

A.1 | Publication

A significantly simpler version of this work was submitted to Canadian Conference on Electrical and Computer Engineering (CCECE) 2024 held in Kingston, Ontario, titled **"Surrogate-Assisted Multi-Objective Design Optimization of a Lorentz Force Actuator [86]"**. The paper discusses the regular offline learning approach applied to the U-shaped actuator. There are some minor details that differ from the work presented here, for instance, the kernel used in Gaussian process regression, and the preprocessing methods used. Regardless, the procedure is more or less the same as the one presented here defined as "SAMOO". The publication was presented in-person and was awarded **"2nd Runner Up - Best Paper Award"**.

As the introduction of A-SAMOO represents a major contribution of this work, the results are intended to be published as a journal paper. The draft for this paper is completed.

A.2 | ai4d Python Package

In order to perform the experiments, a Python package named "**ai4d**" was developed. The code is currently hosted on Github in a private repository and will be made available upon request. The structure of the package is illustrated in Figure A.1. One of the major technical challenges of this project was to reliably communicate with the FEM software, COMSOL Multiphysics. ai4d provides parallel-computing capabilities for sampling that dramatically reduces the average simulation times when multiple designs are to be evaluated. By running each simulation on one CPU core, almost **8 times** faster sampling times compared to the sequential case were achieved on a Mac Studio with M1 Ultra CPU and 64 GB of RAM.

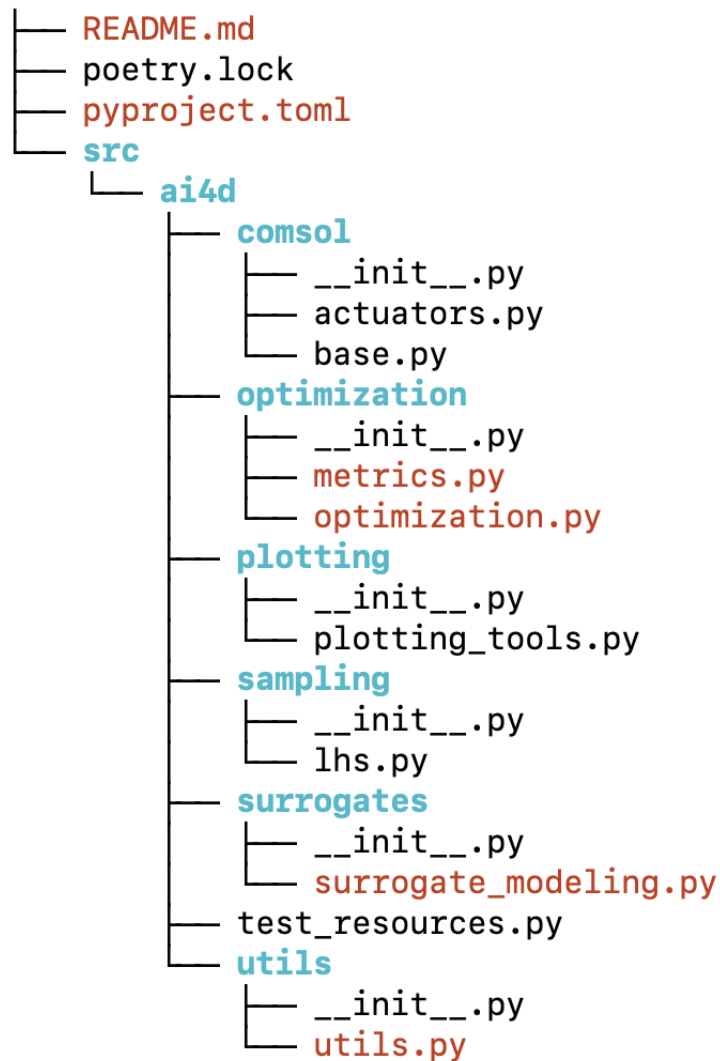


Figure A.1: ai4d Python package developed for this project

Supplementary Figures

B.1 | U-Shaped Lorentz Force Actuator

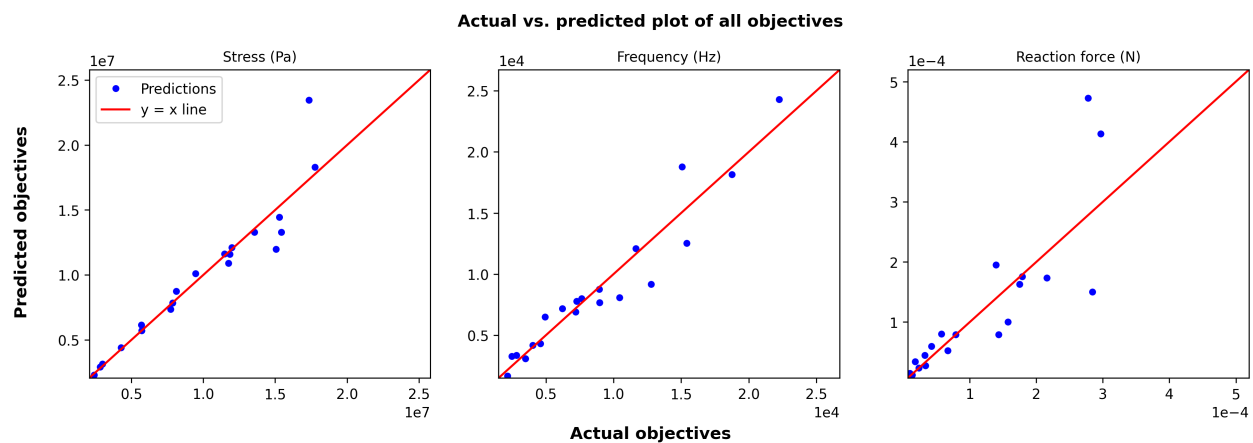


Figure B.1: Actual vs predicted plot for the U-shaped Lorentz force actuator with a training size of 20.

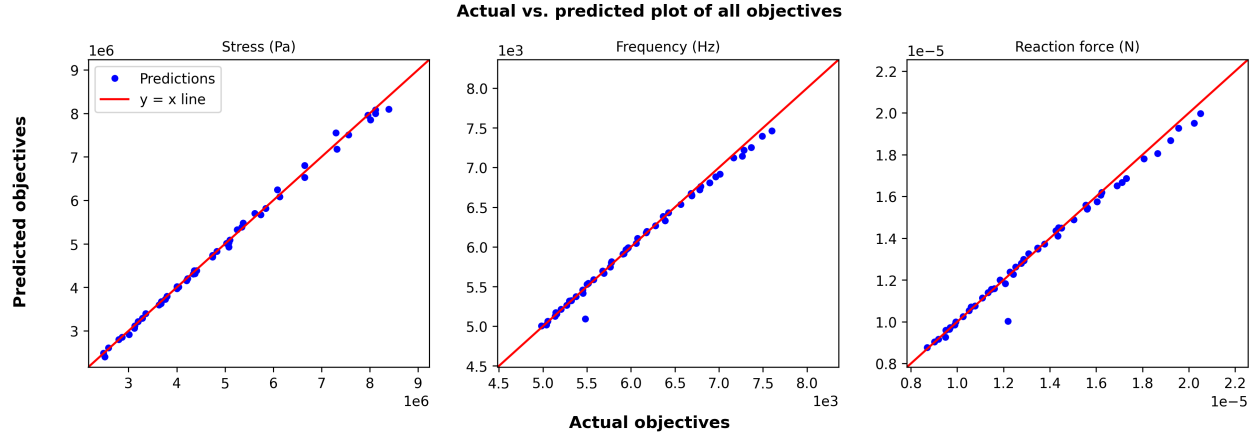


Figure B.2: Actual vs predicted plot for the U-shaped Lorentz force actuator with a training size of 20: After Adaptive (A-SAMOO) Optimization.

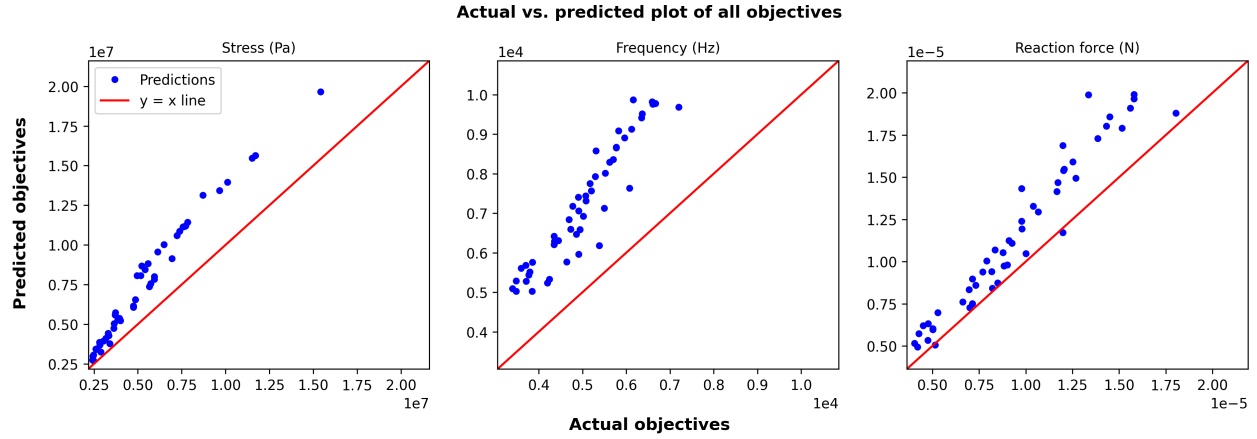


Figure B.3: Actual vs predicted plot for the U-shaped Lorentz force actuator with a training size of 20: After Regular (SAMOO) Optimization.

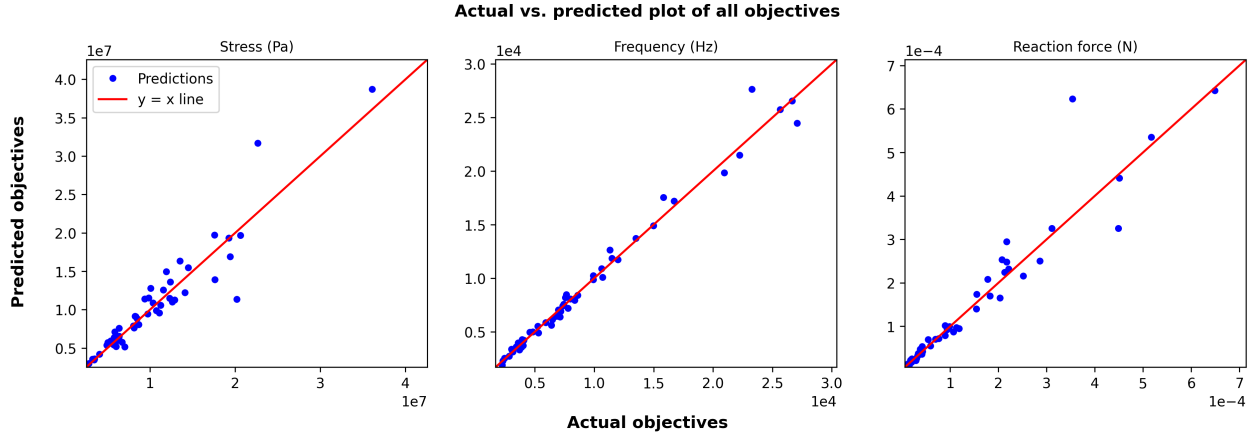


Figure B.4: Actual vs predicted plot for the U-shaped Lorentz force actuator with a training size of 50.

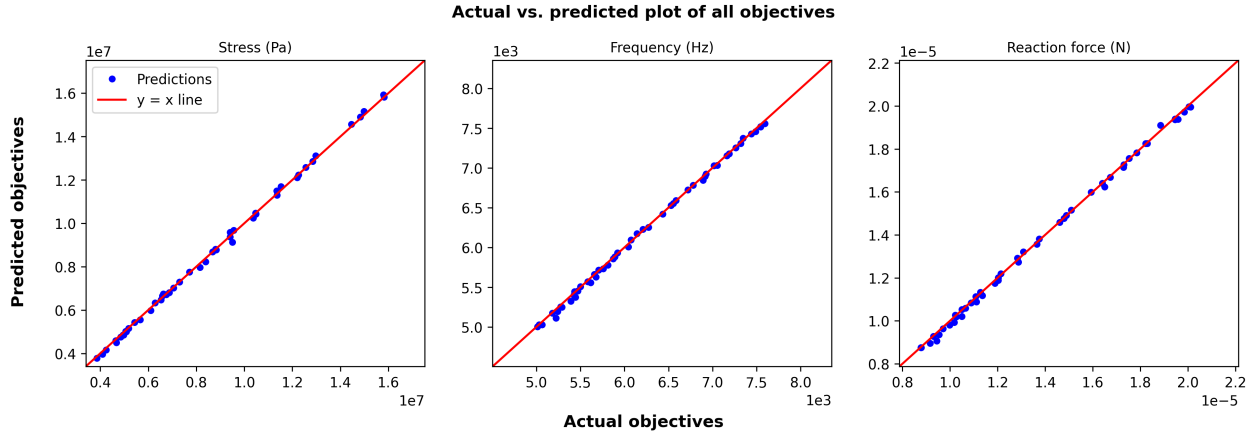


Figure B.5: Actual vs predicted plot for the U-shaped Lorentz force actuator with a training size of 50: After Adaptive (A-SAMOO) Optimization.

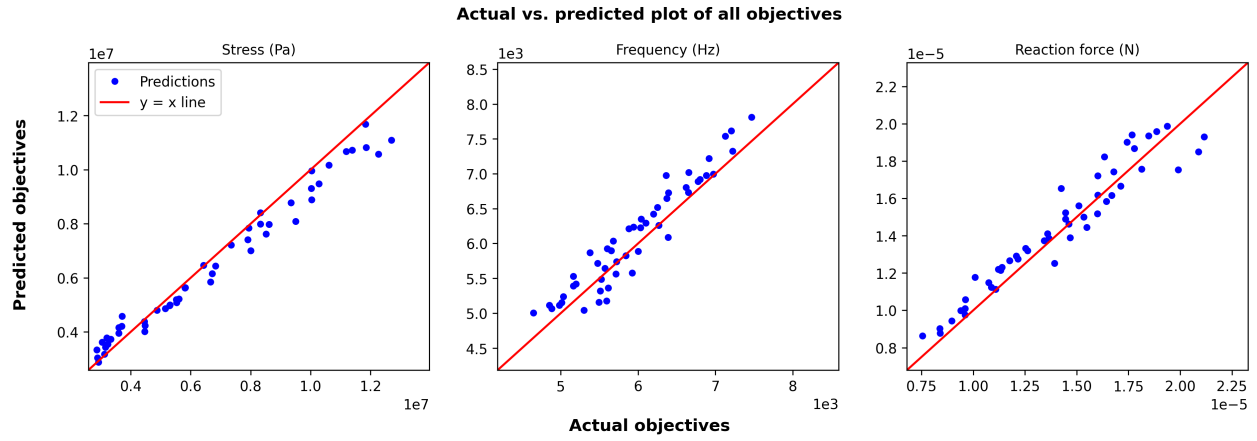


Figure B.6: Actual vs predicted plot for the U-shaped Lorentz force actuator with a training size of 50: After Regular (SAMOO) Optimization.

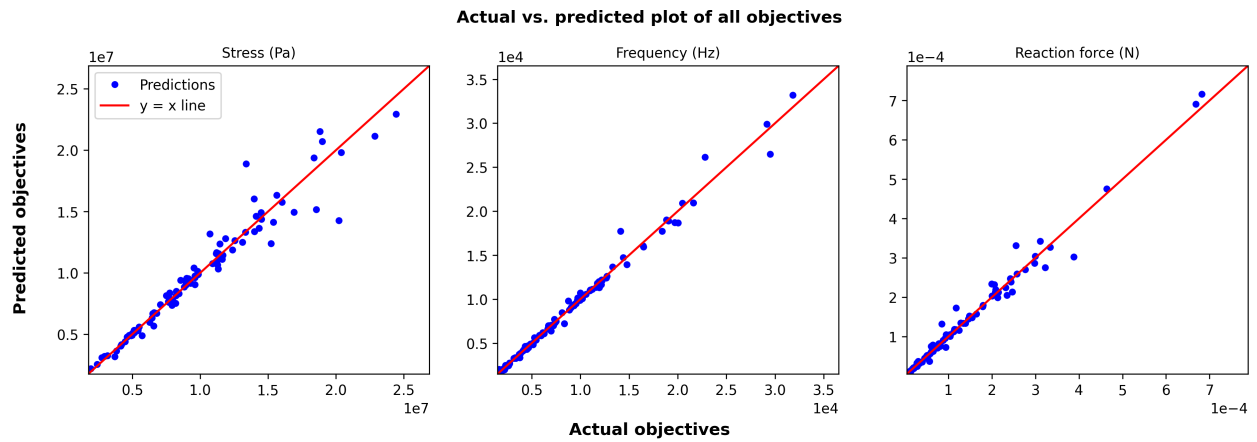


Figure B.7: Actual vs predicted plot for the U-shaped Lorentz force actuator with a training size of 100.

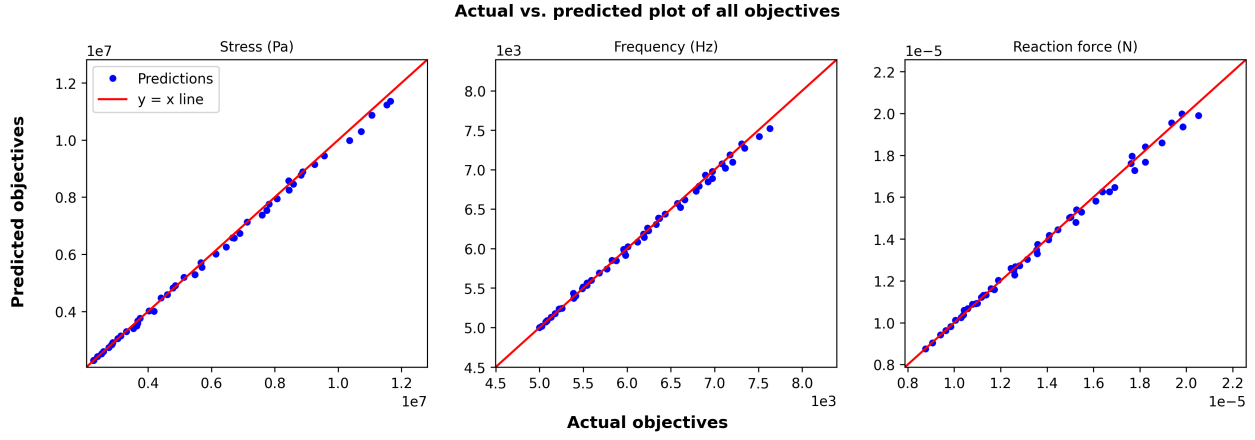


Figure B.8: Actual vs predicted plot for the U-shaped Lorentz force actuator with a training size of 100: After Adaptive (A-SAMOO) Optimization.

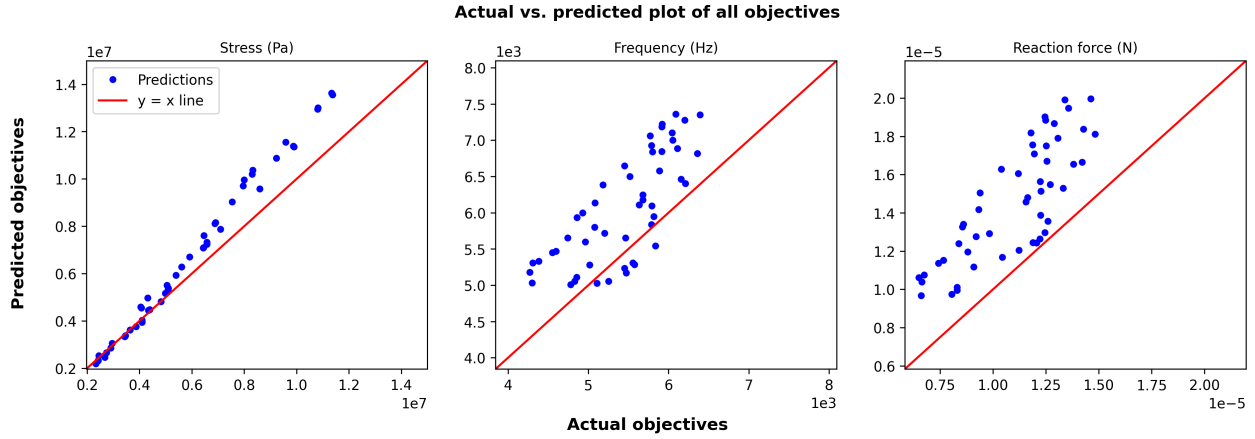


Figure B.9: Actual vs predicted plot for the U-shaped Lorentz force actuator with a training size of 100: After Regular (SAMOO) Optimization.

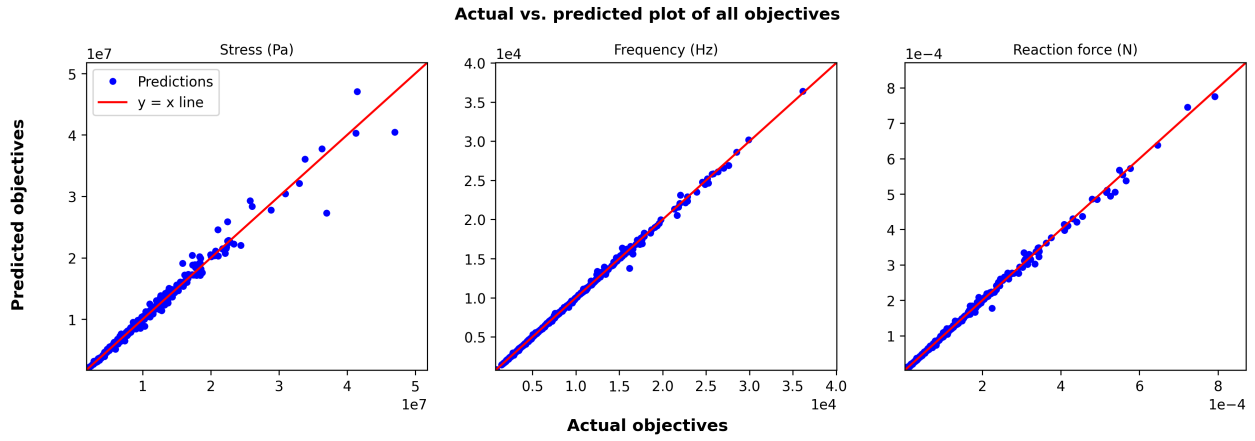


Figure B.10: Actual vs predicted plot for the U-shaped Lorentz force actuator with a training size of 400.

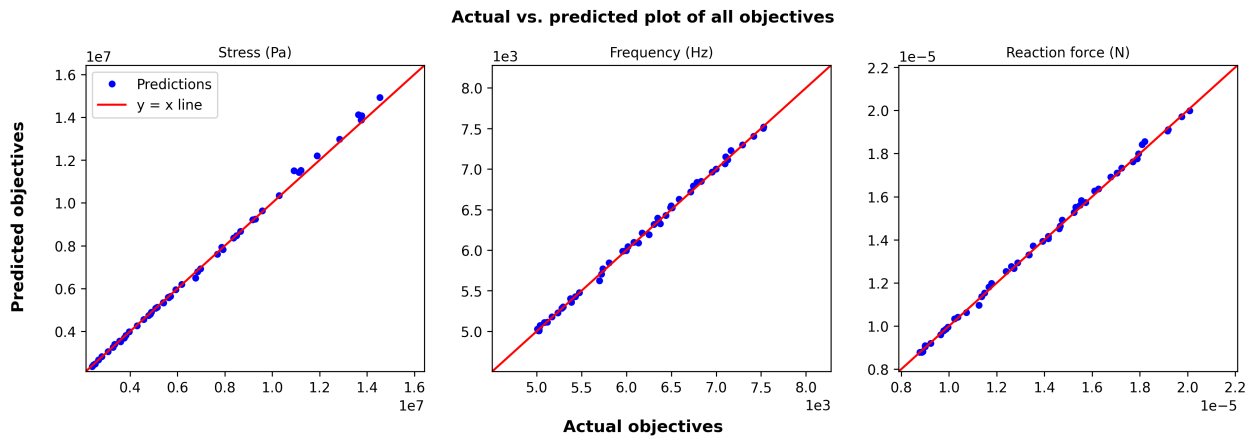


Figure B.11: Actual vs predicted plot for the U-shaped Lorentz force actuator with a training size of 400: After Adaptive (A-SAMOO) Optimization.

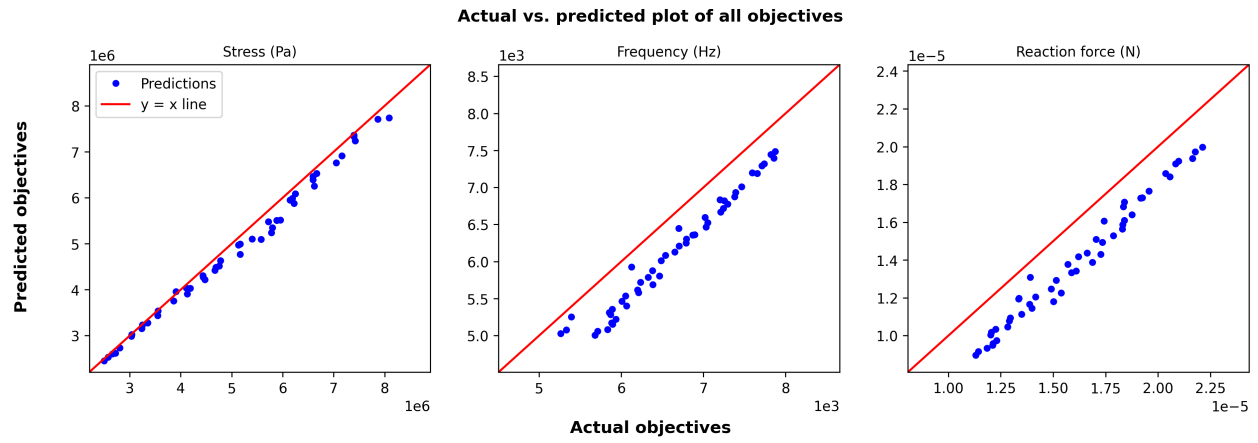


Figure B.12: Actual vs predicted plot for the U-shaped Lorentz force actuator with a training size of 400: After Regular (SAMOO) Optimization.

B.2 | Lorentz Force Actuator with Springs

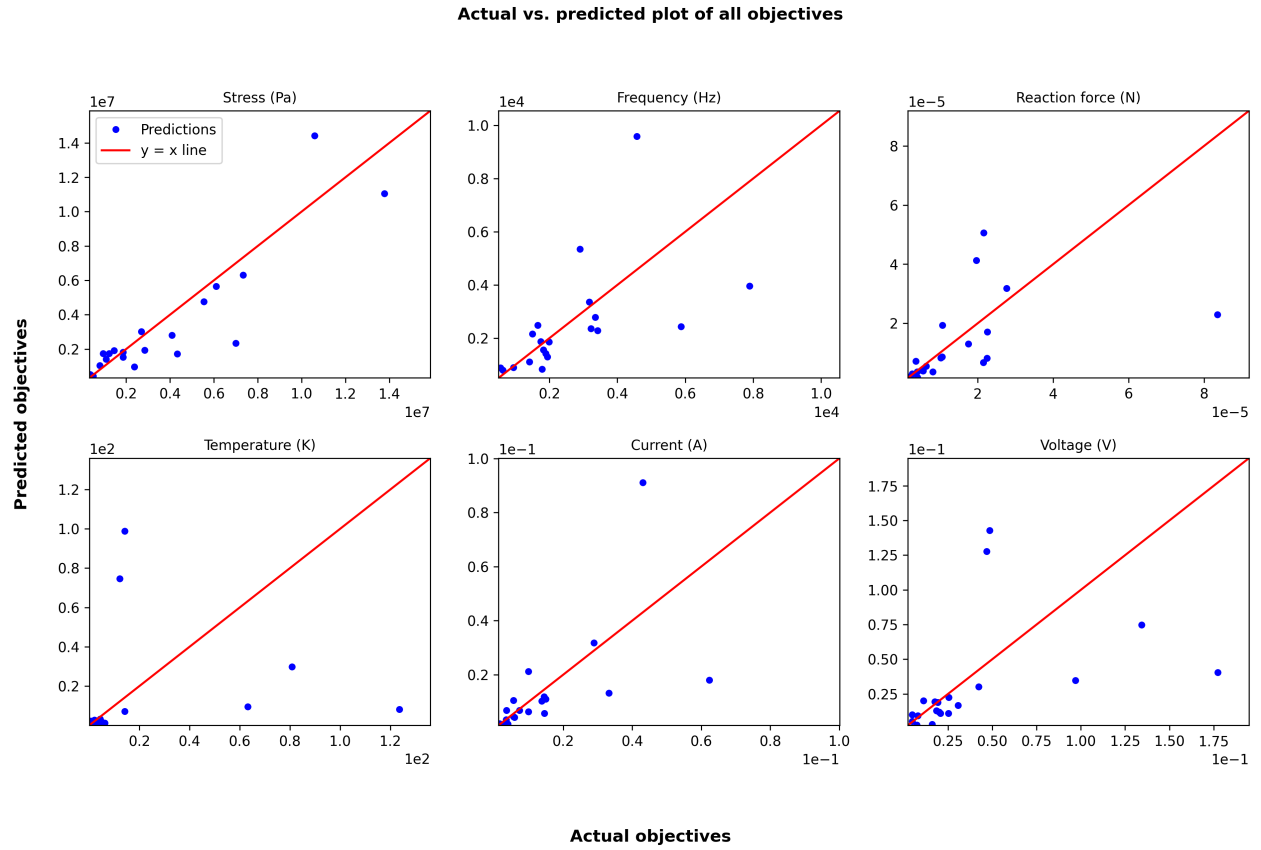


Figure B.13: Actual vs predicted plot for the Lorentz force actuator with springs with a training size of 20.

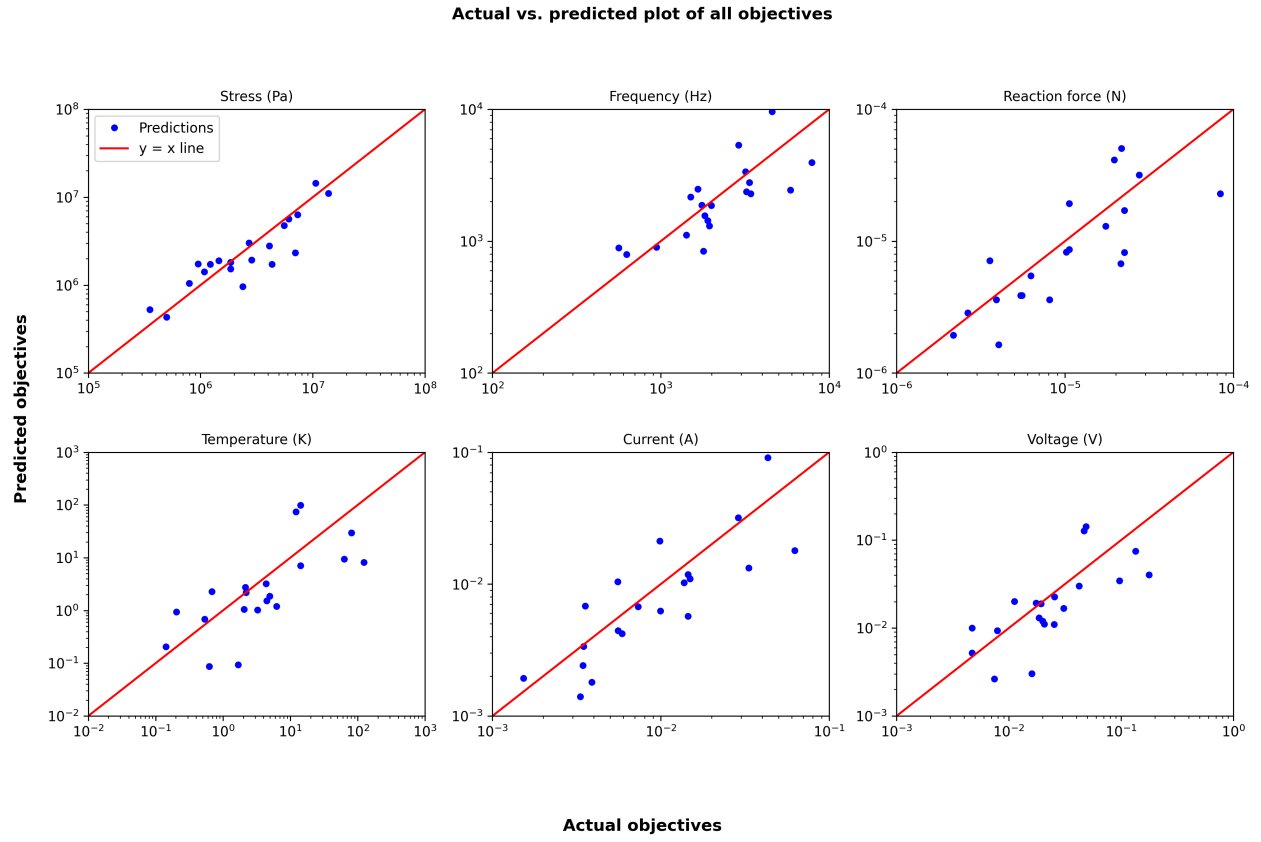


Figure B.14: Actual vs predicted plot (log scaled) for the Lorentz force actuator with springs with a training size of 20.

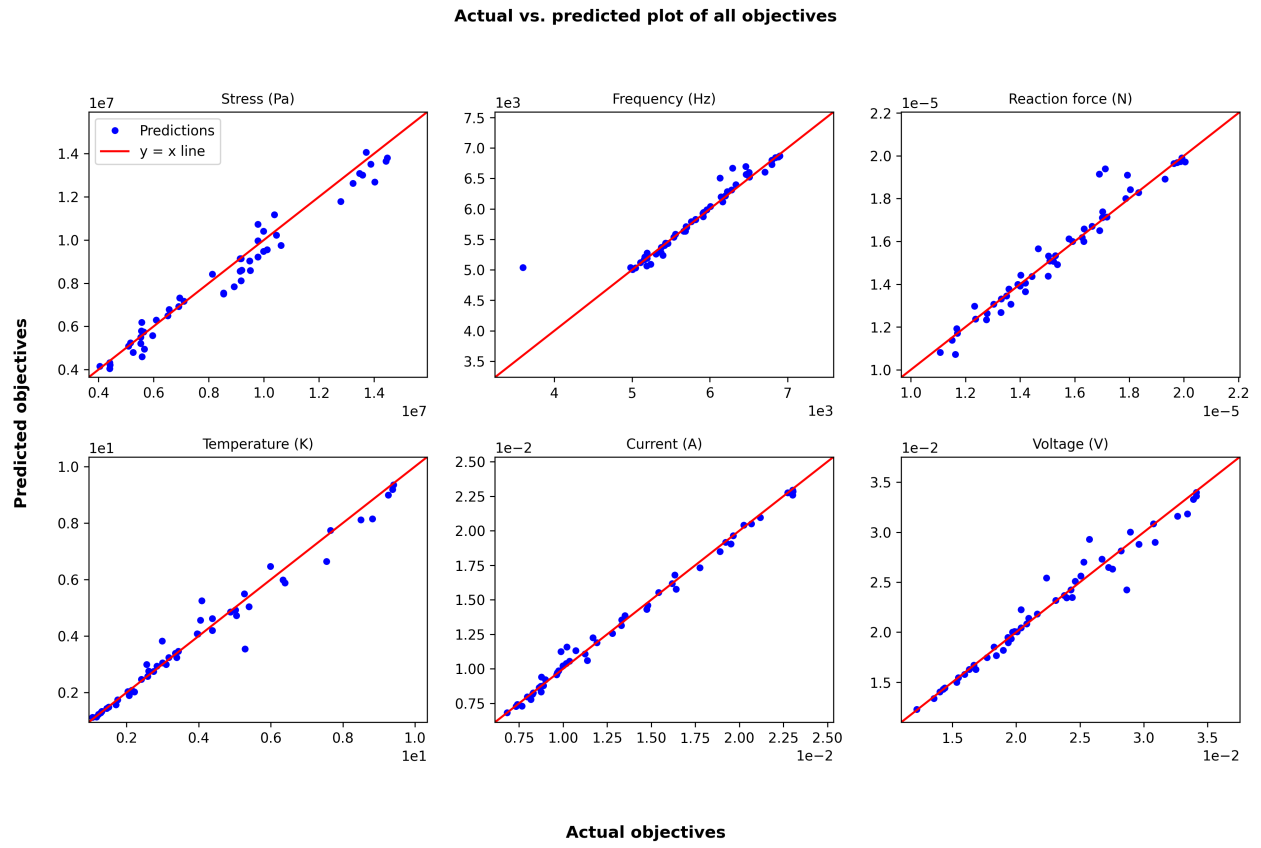


Figure B.15: Actual vs predicted plot for the Lorentz force actuator with springs with a training size of 20: After Adaptive (A-SAMOO) Optimization.

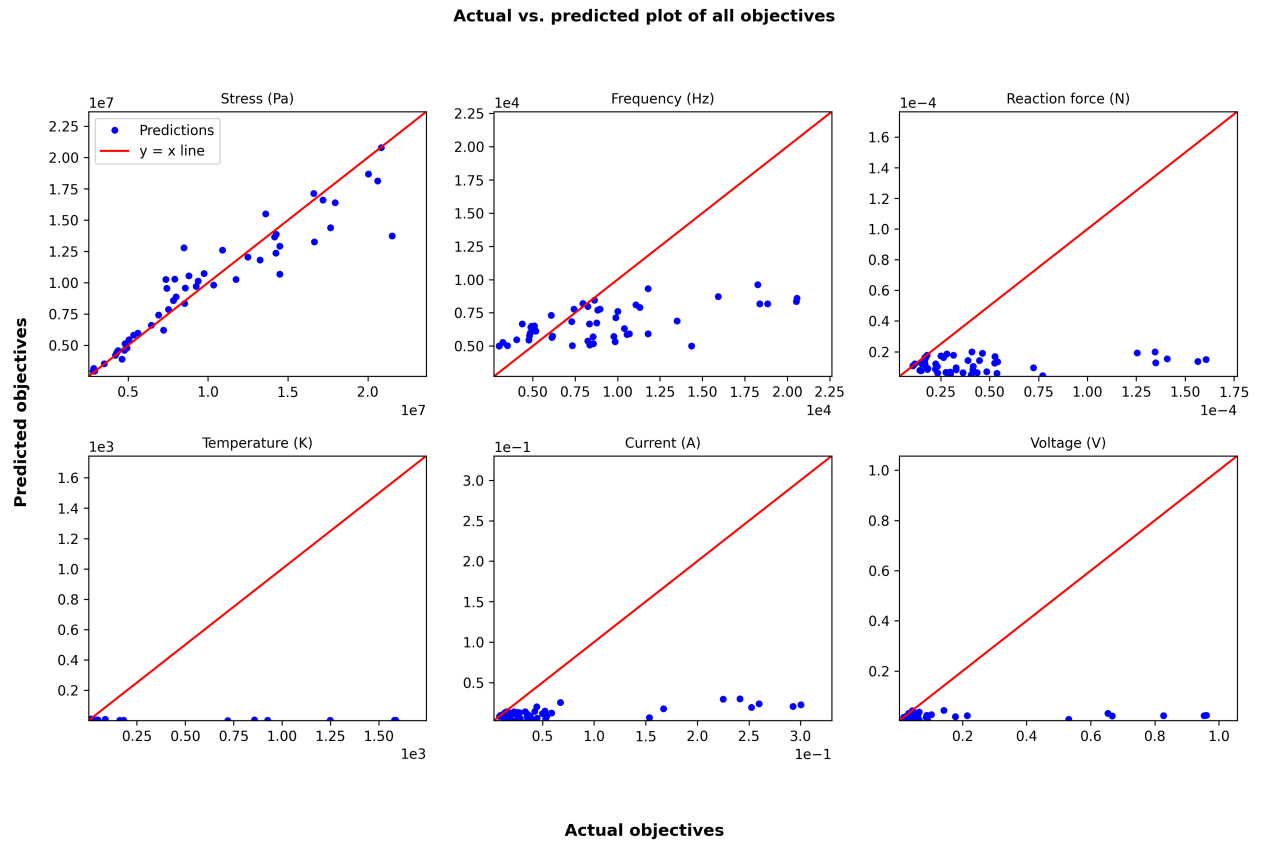


Figure B.16: Actual vs predicted plot for the Lorentz force actuator with springs with a training size of 20: After Regular (SAMOO) Optimization.

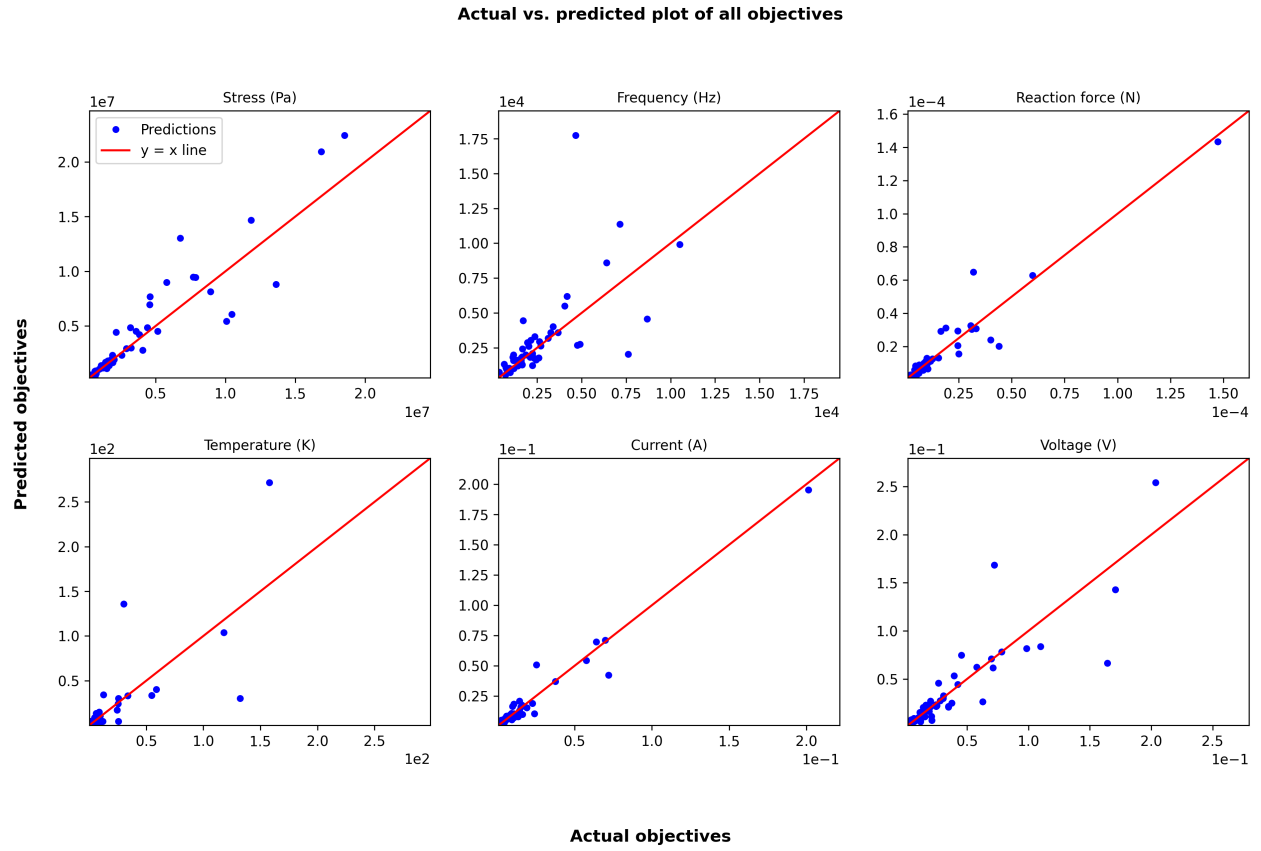


Figure B.17: Actual vs predicted plot for the Lorentz force actuator with springs with a training size of 50.

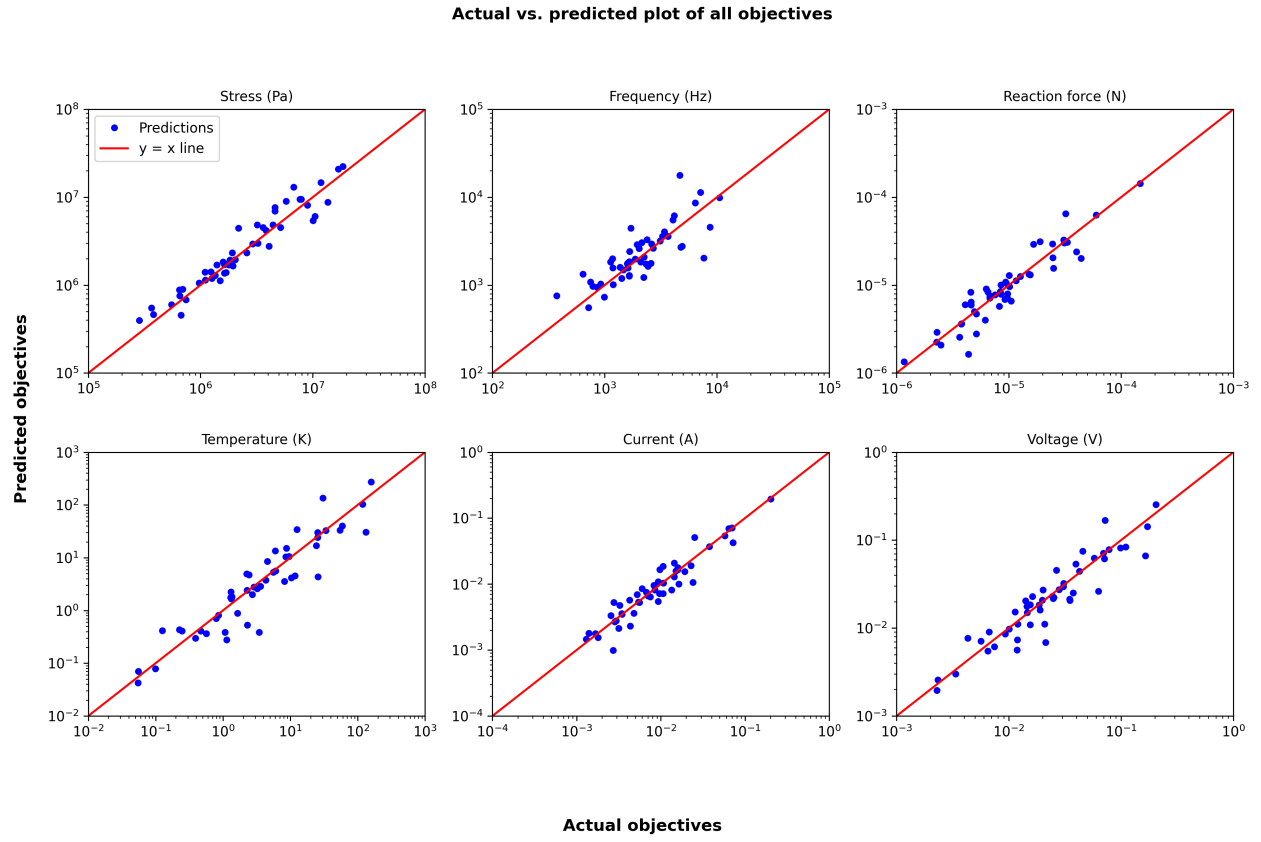


Figure B.18: Actual vs predicted plot (log scaled) for the Lorentz force actuator with springs with a training size of 50.

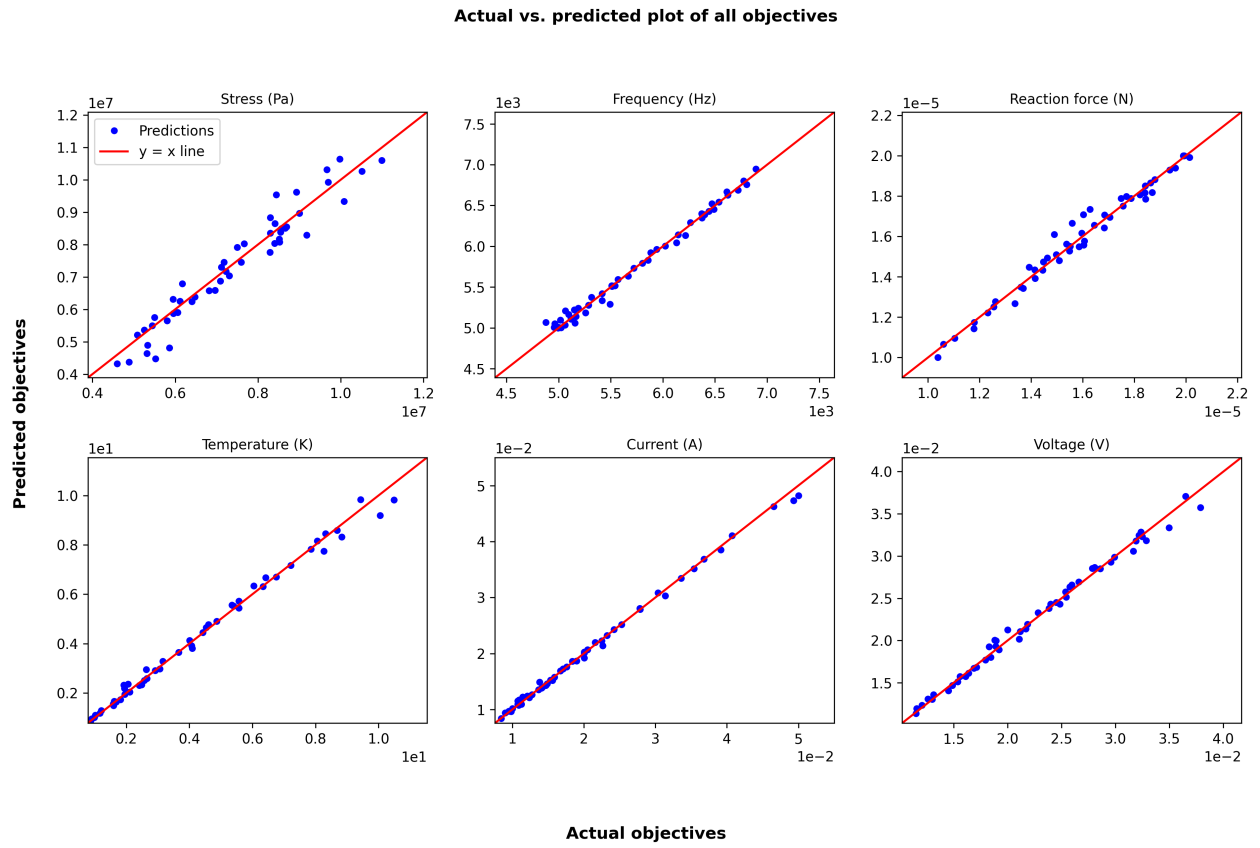


Figure B.19: Actual vs predicted plot for the Lorentz force actuator with springs with a training size of 50: After Adaptive (A-SAMOO) Optimization.

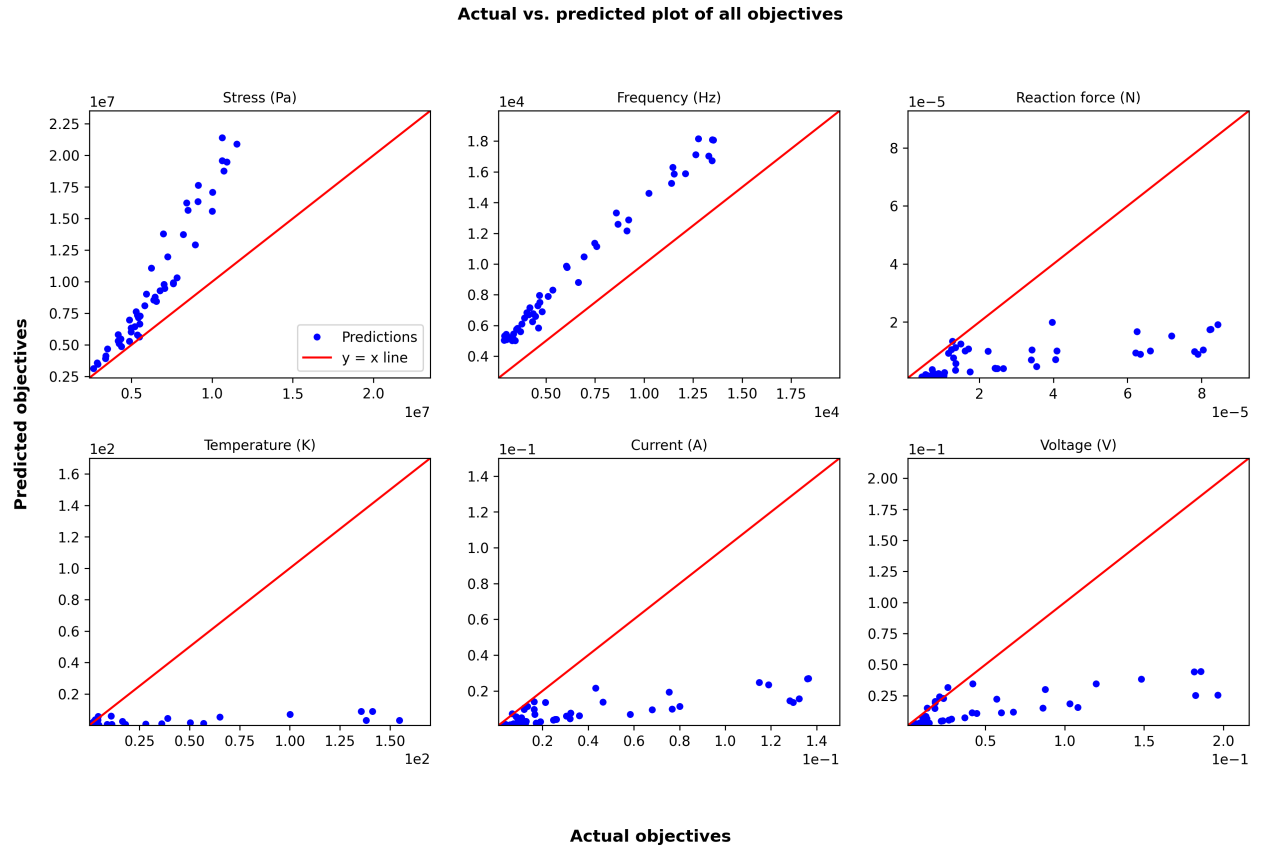


Figure B.20: Actual vs predicted plot for the Lorentz force actuator with springs with a training size of 50: After Regular (SAMOO) Optimization.

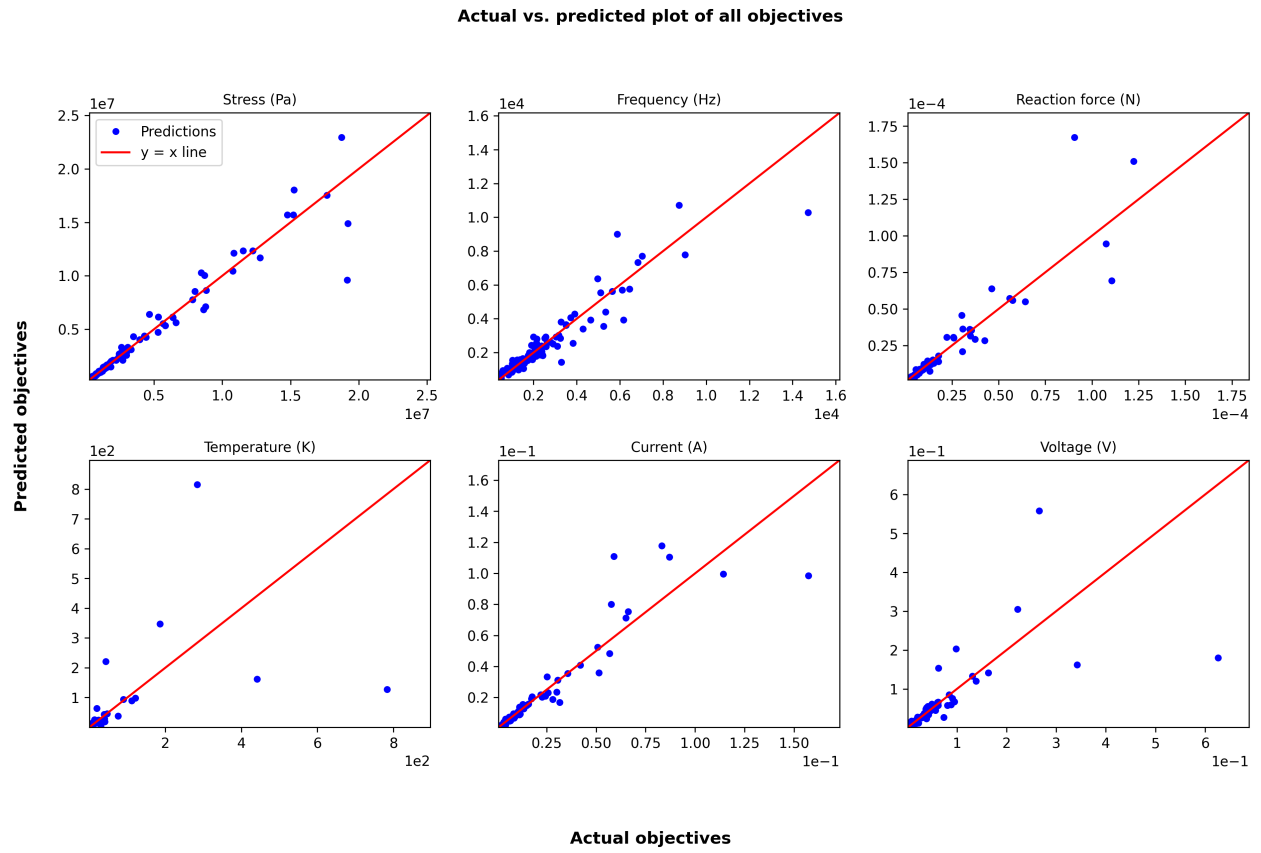


Figure B.21: Actual vs predicted plot for the Lorentz force actuator with springs with a training size of 100.

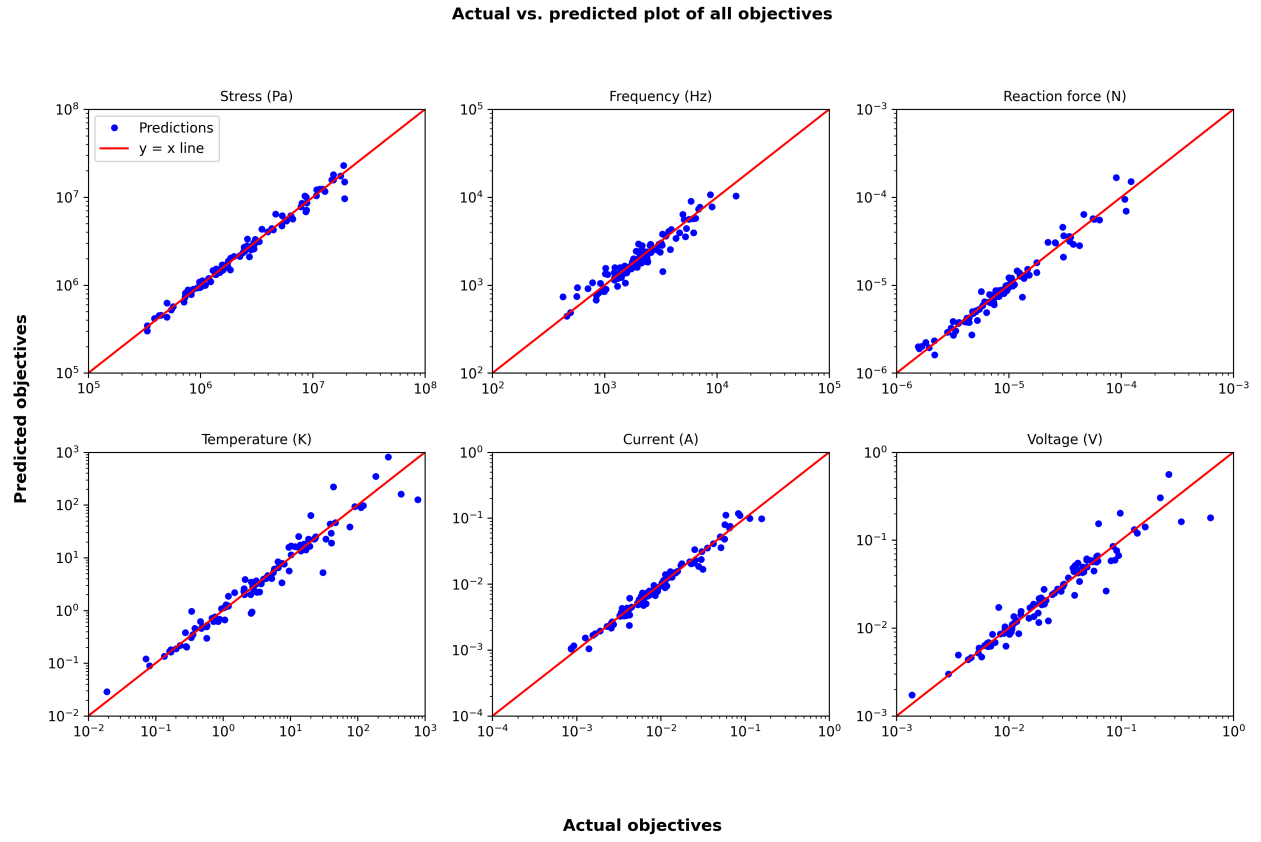


Figure B.22: Actual vs predicted plot (log scaled) for the Lorentz force actuator with springs with a training size of 100.

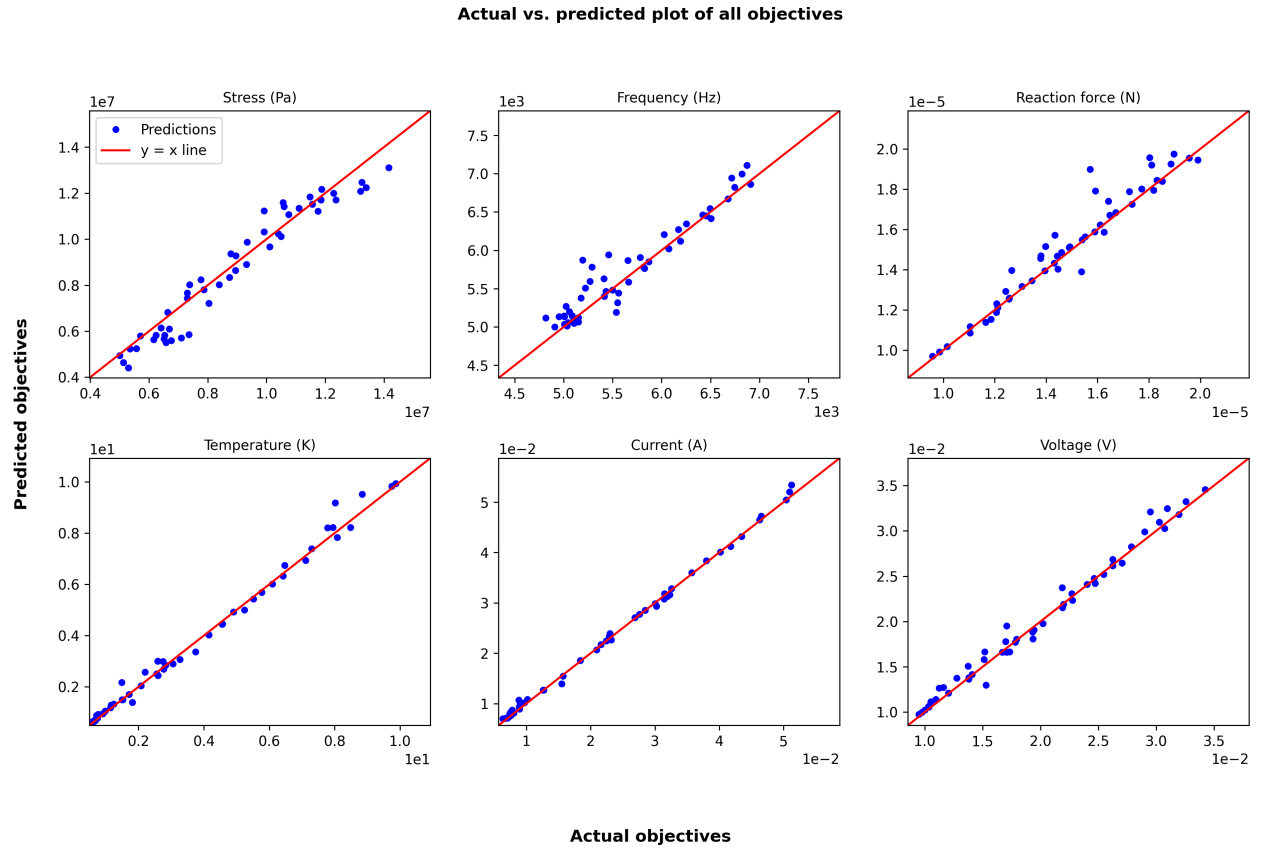


Figure B.23: Actual vs predicted plot for the Lorentz force actuator with springs with a training size of 100: After Adaptive (A-SAMOO) Optimization.

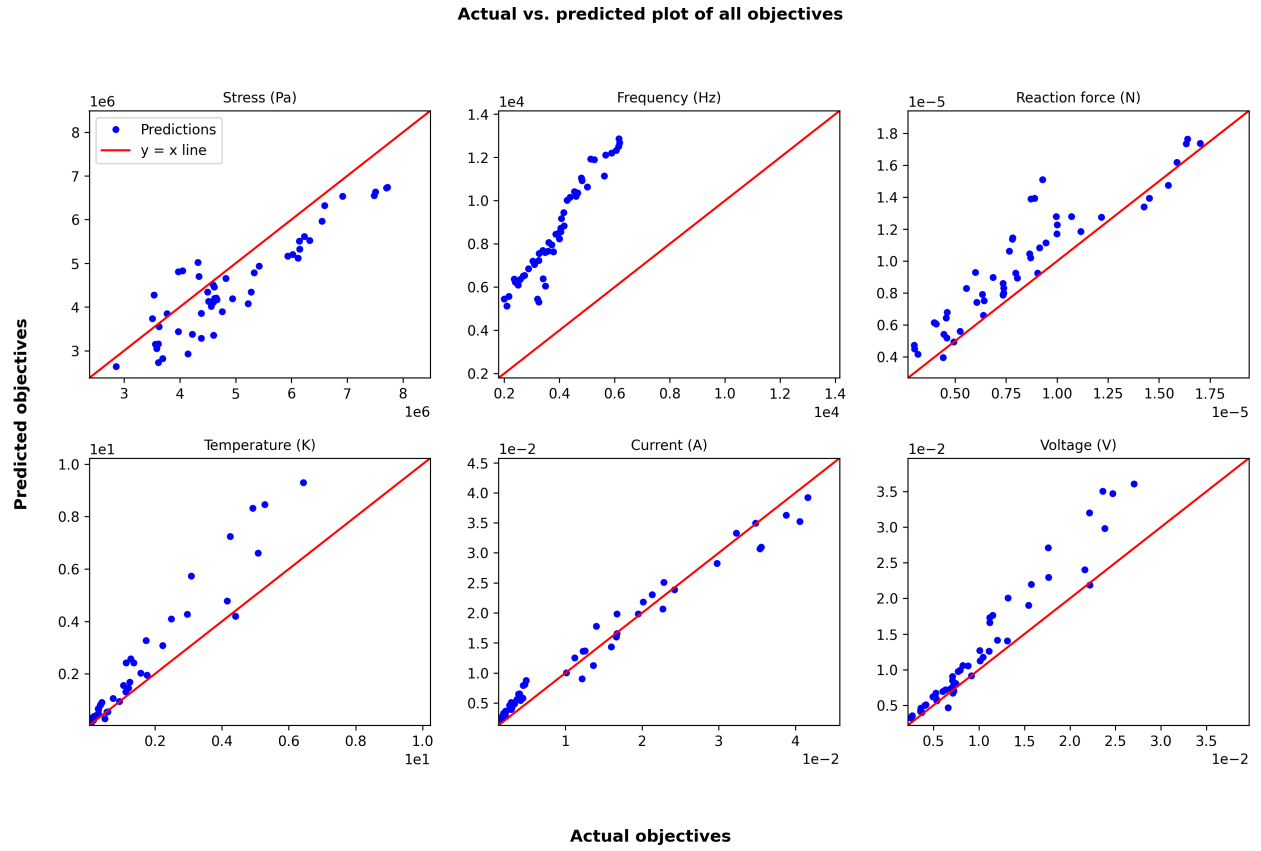


Figure B.24: Actual vs predicted plot for the Lorentz force actuator with springs with a training size of 100: After Regular (SAMOO) Optimization.

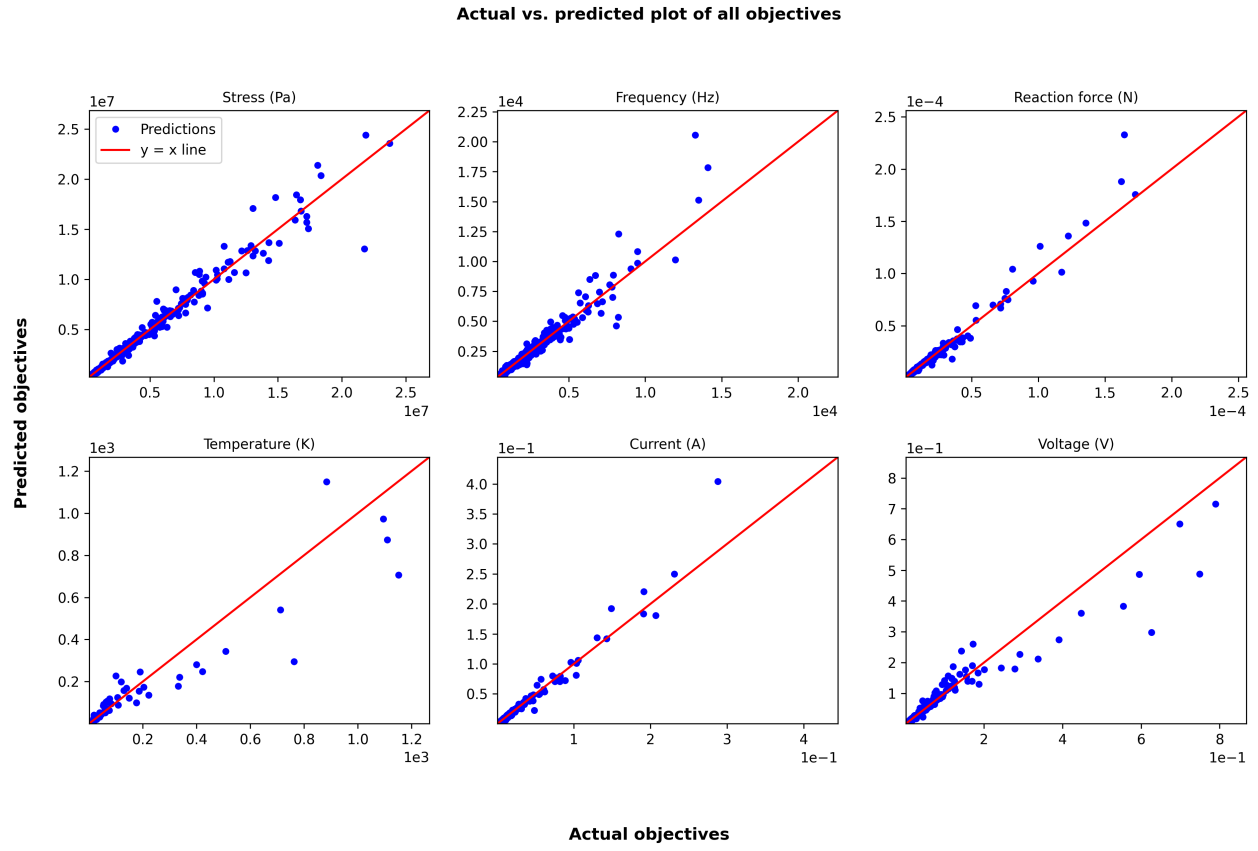


Figure B.25: Actual vs predicted plot for the Lorentz force actuator with springs with a training size of 400.

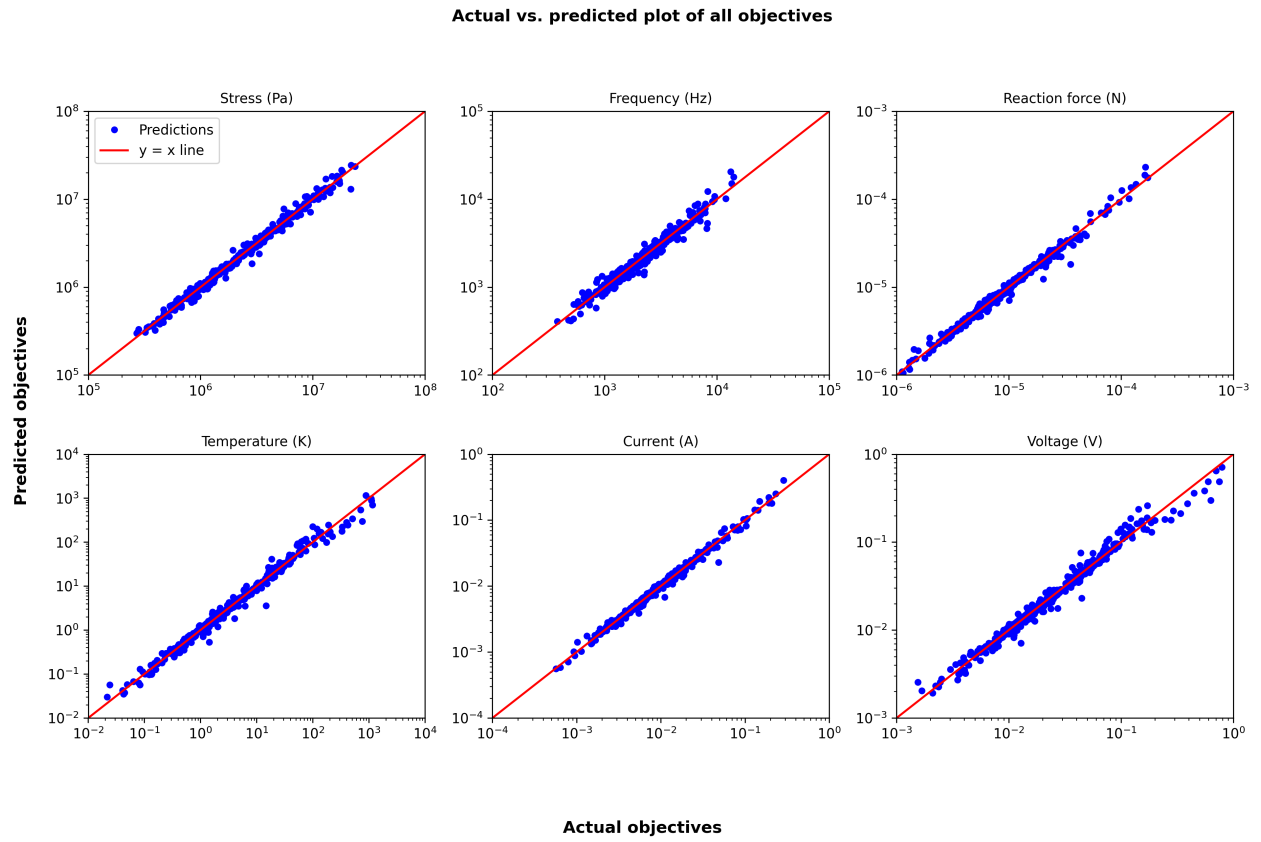


Figure B.26: Actual vs predicted plot (log scaled) for the Lorentz force actuator with springs with a training size of 400.

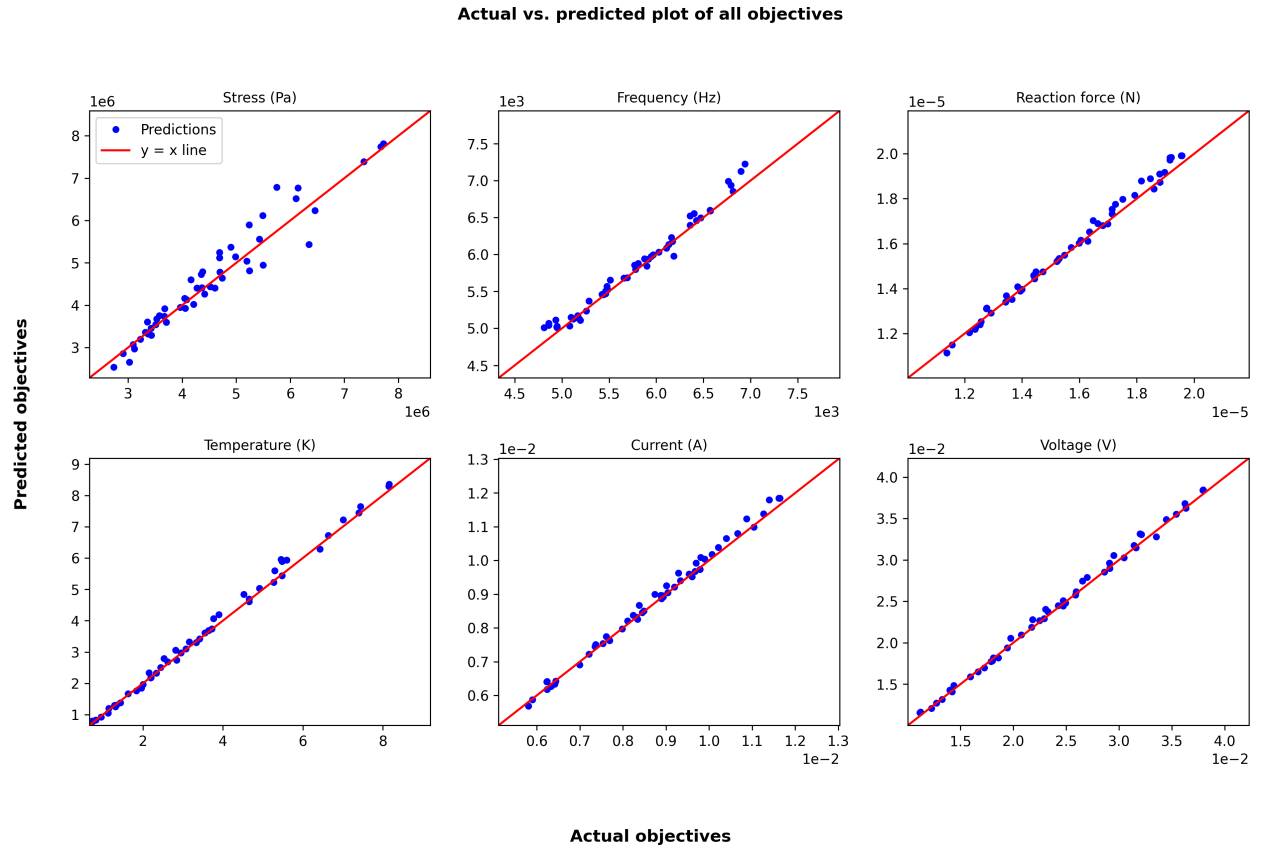


Figure B.27: Actual vs predicted plot for the Lorentz force actuator with springs with a training size of 400: After Adaptive (A-SAMOO) Optimization.

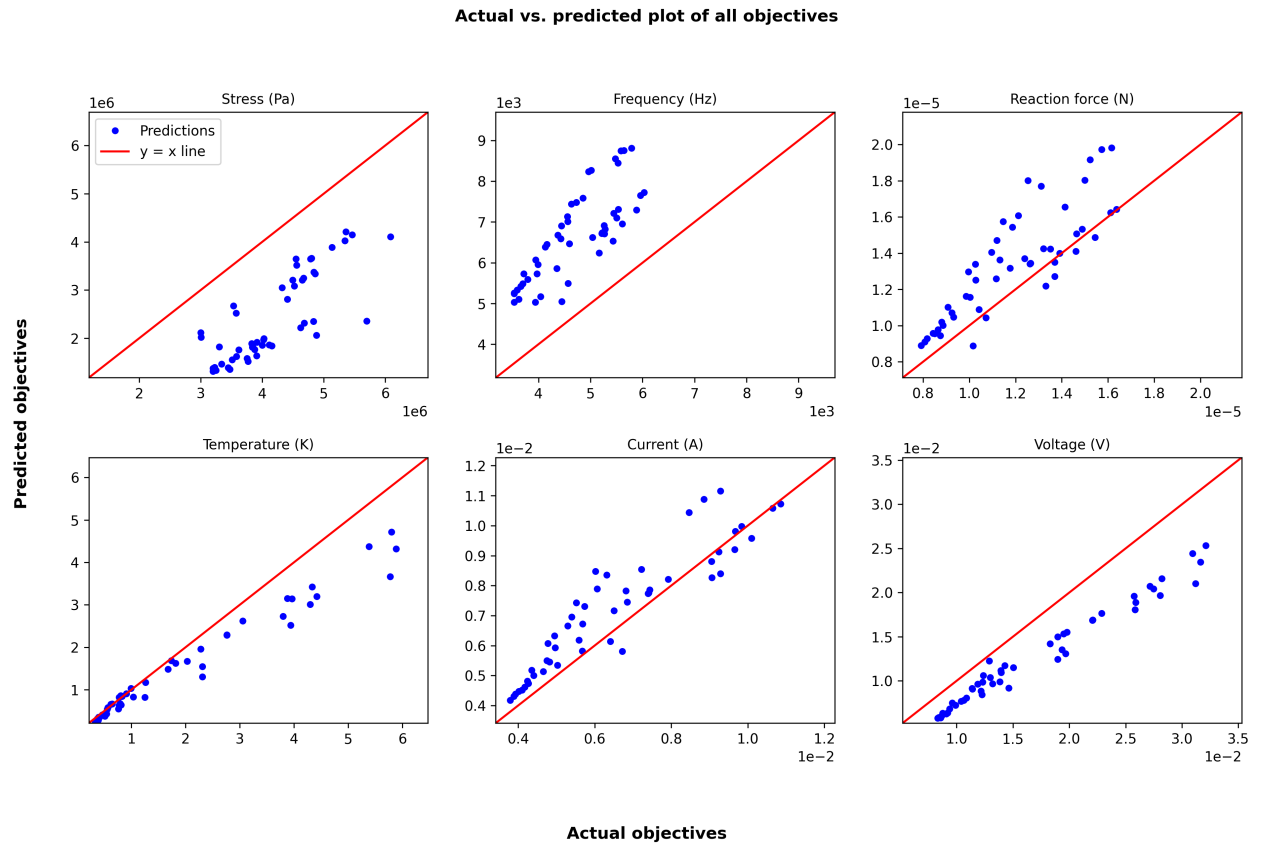


Figure B.28: Actual vs predicted plot for the Lorentz force actuator with springs with a training size of 400: After Regular (SAMOO) Optimization.

B.3 | Thermal Actuator

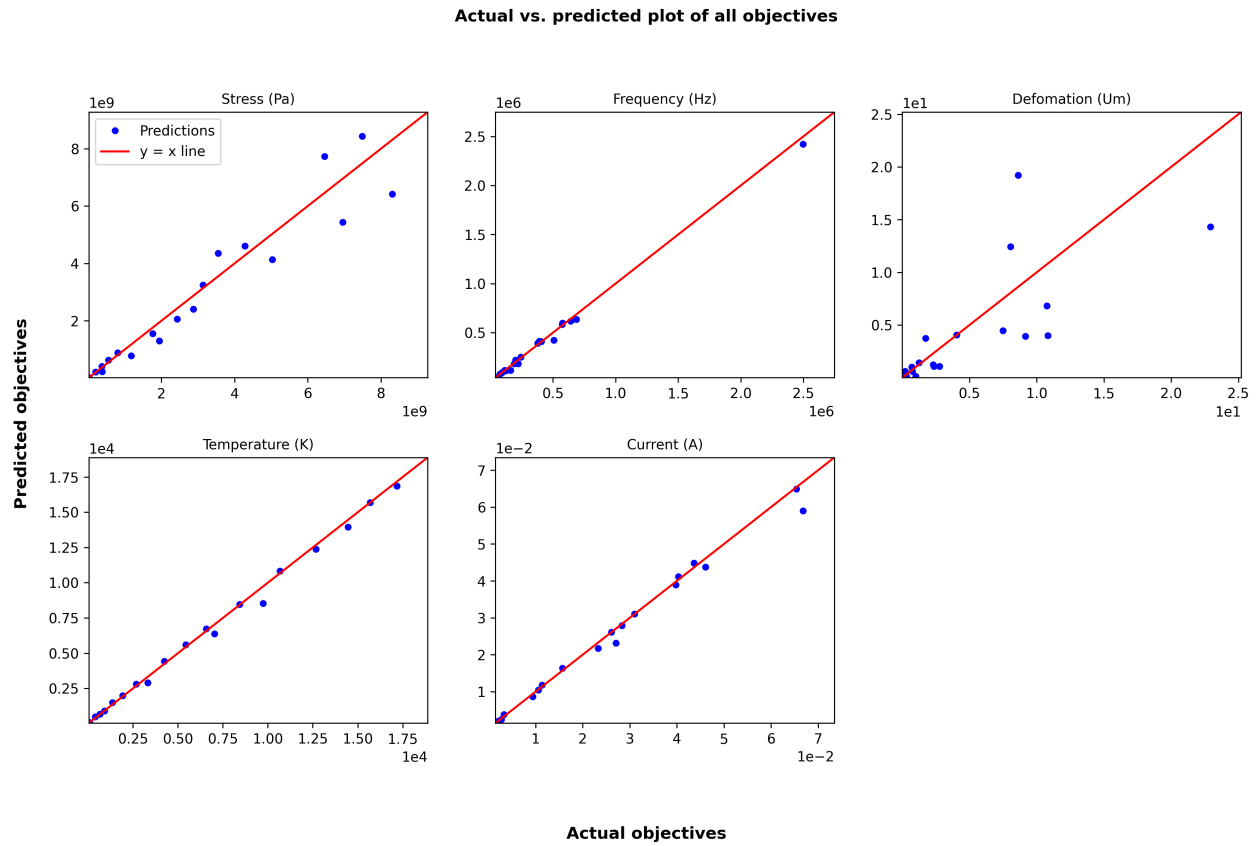


Figure B.29: Actual vs predicted plot for the thermal actuator with a training size of 20.

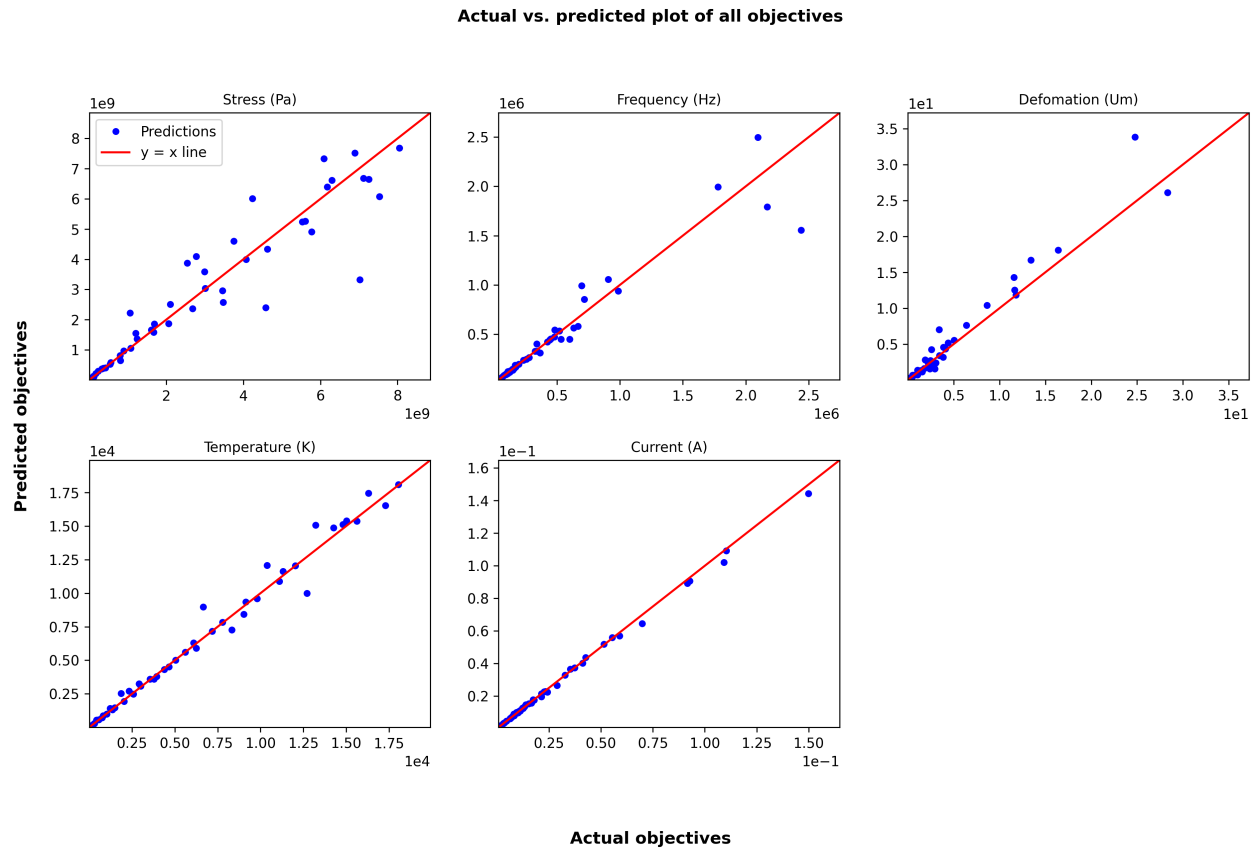


Figure B.30: Actual vs predicted plot for the thermal actuator with a training size of 50.

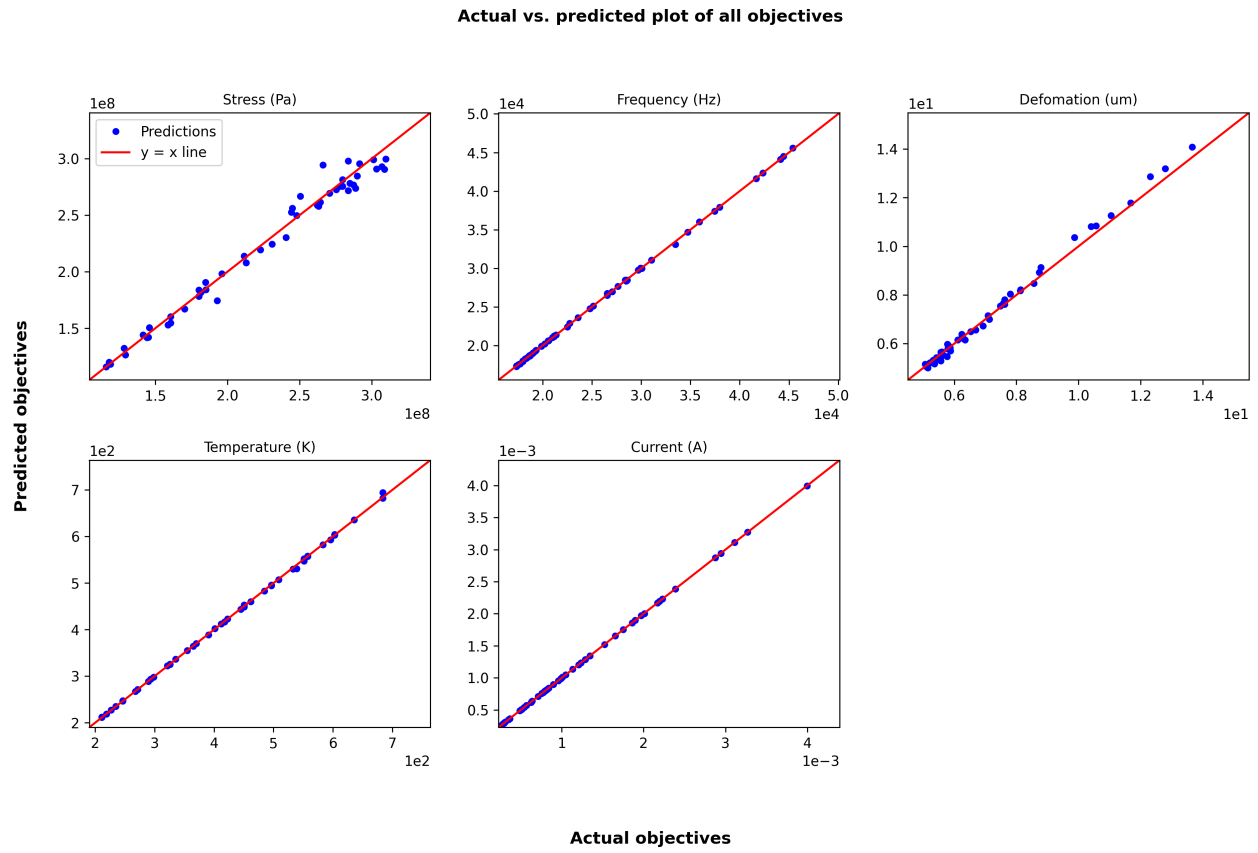


Figure B.31: Actual vs predicted plot for the thermal actuator with a training size of 50: After Adaptive (A-SAMOO) Optimization.

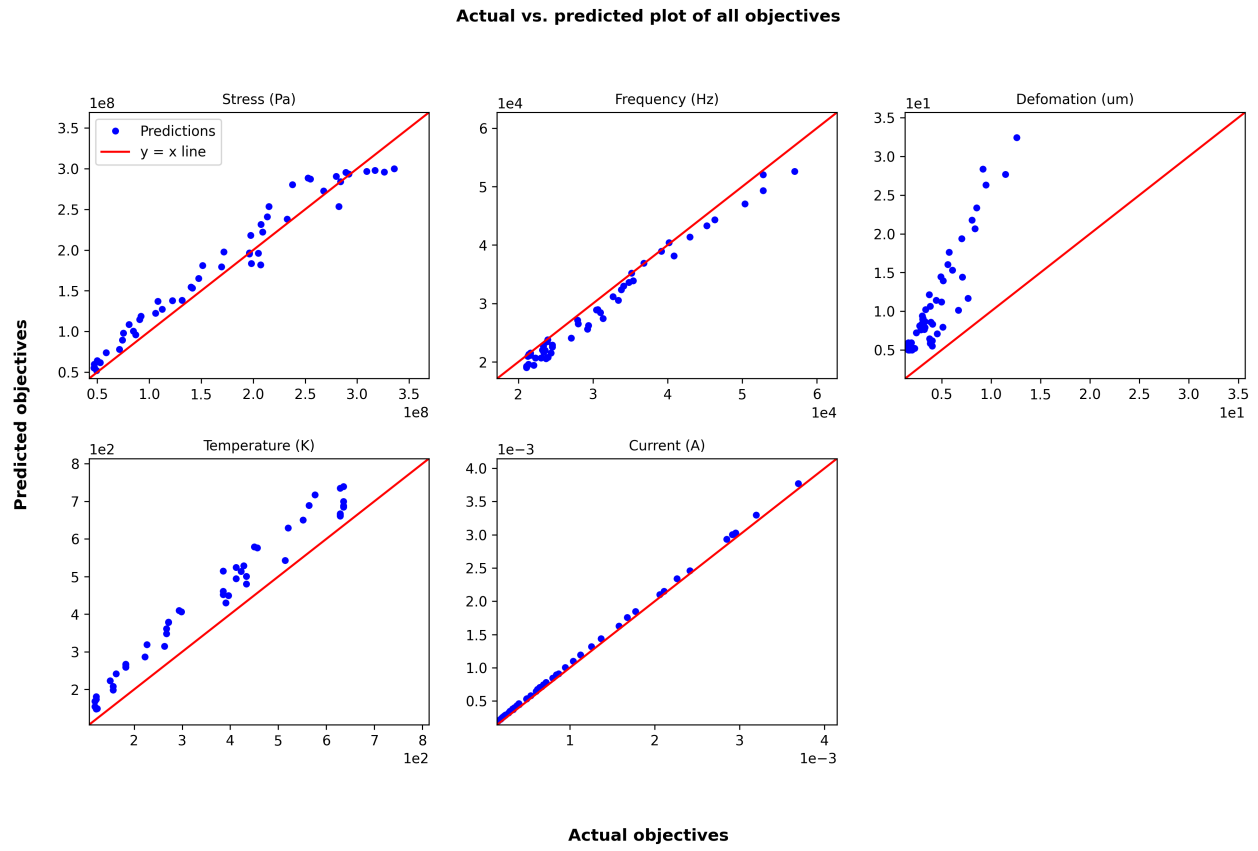


Figure B.32: Actual vs predicted plot for the thermal actuator with a training size of 50: After Regular (SAMOO) Optimization.

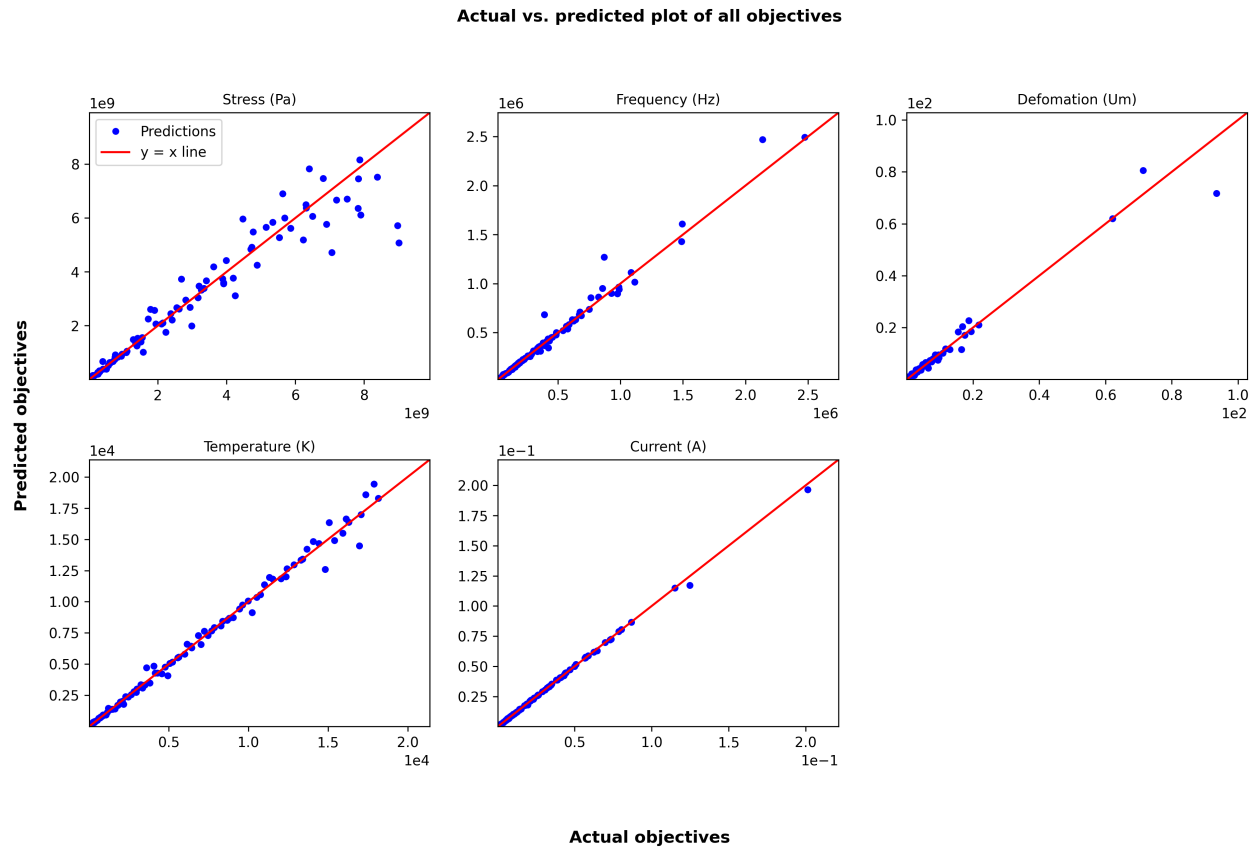


Figure B.33: Actual vs predicted plot for the thermal actuator with a training size of 100.

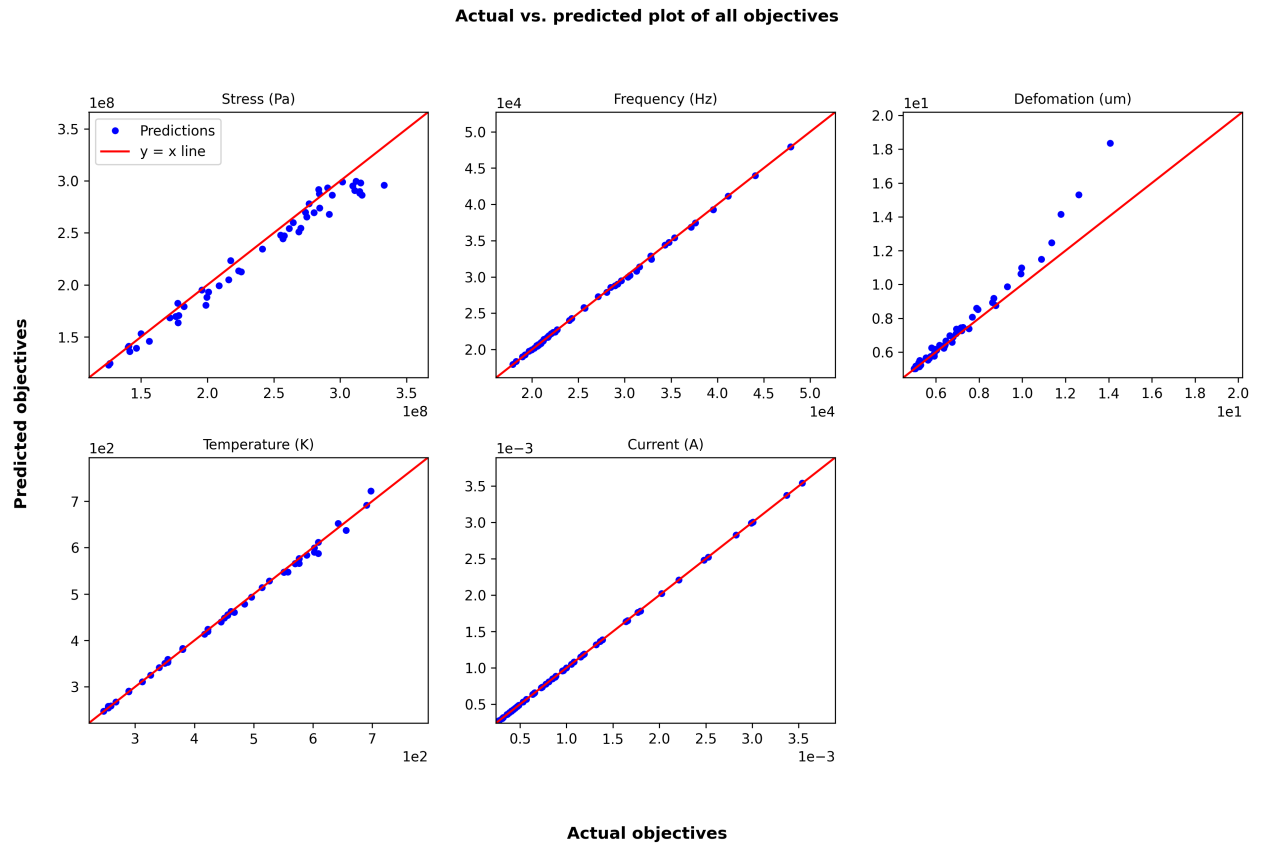


Figure B.34: Actual vs predicted plot for the thermal actuator with a training size of 100: After Adaptive (A-SAMOO) Optimization.

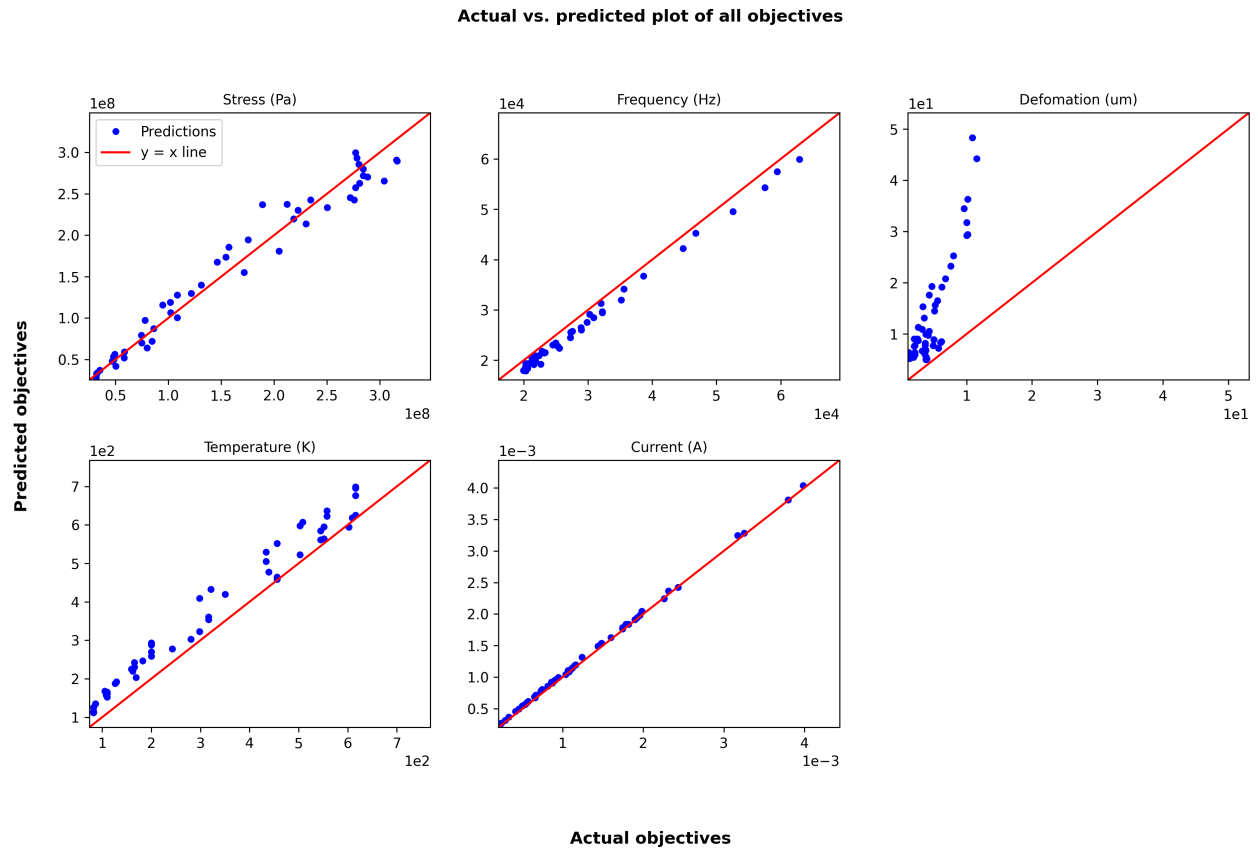


Figure B.35: Actual vs predicted plot for the thermal actuator with a training size of 100: After Regular (SAMOO) Optimization.

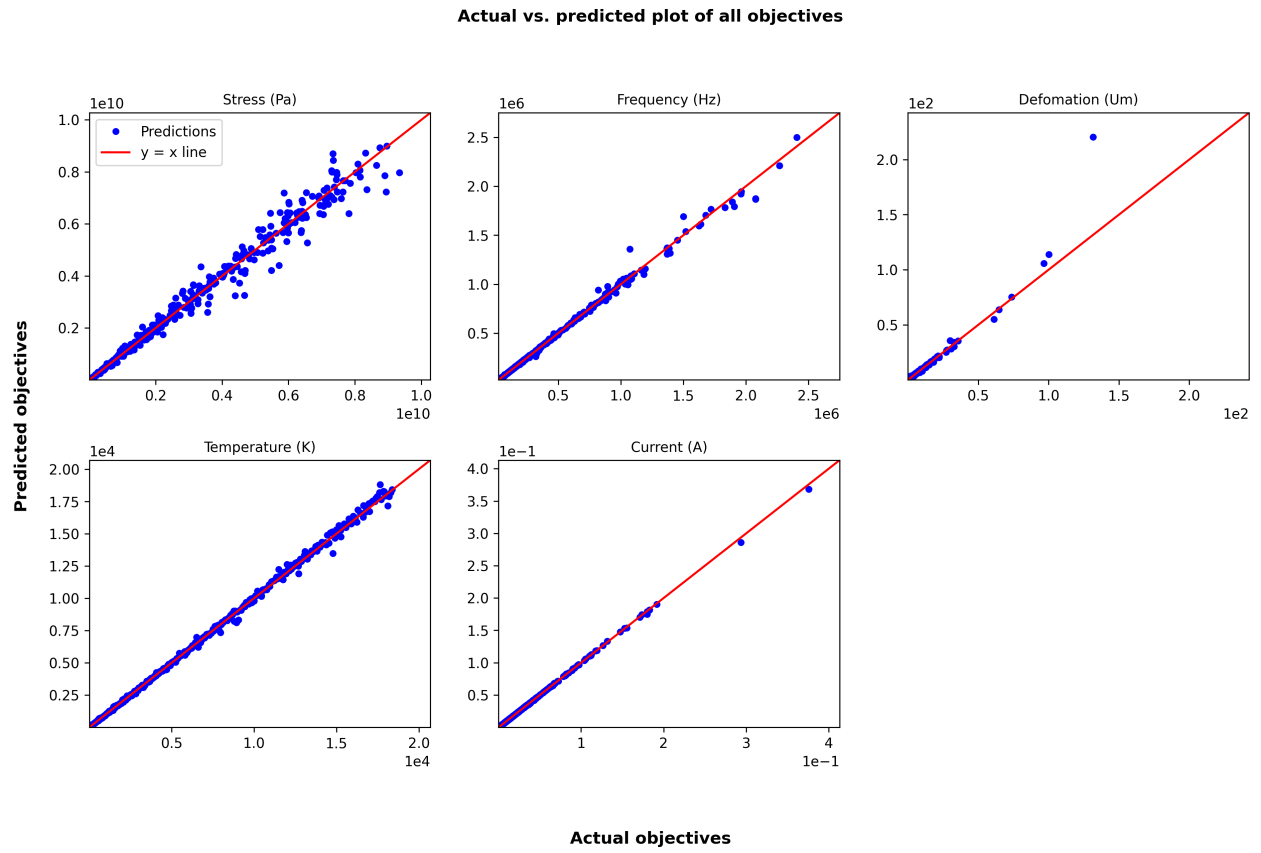


Figure B.36: Actual vs predicted plot for the thermal actuator with a training size of 400.

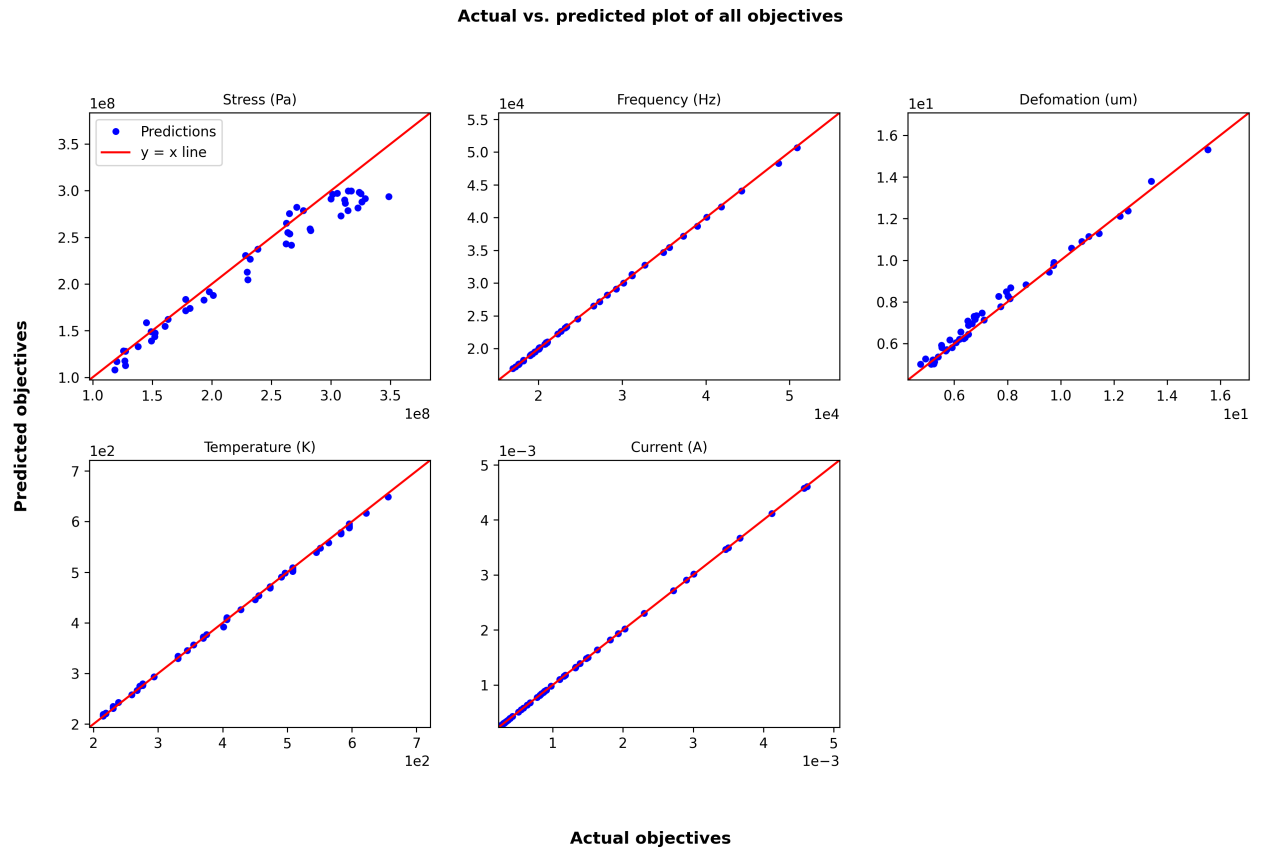


Figure B.37: Actual vs predicted plot for the thermal actuator with a training size of 400: After Adaptive (A-SAMOO) Optimization.

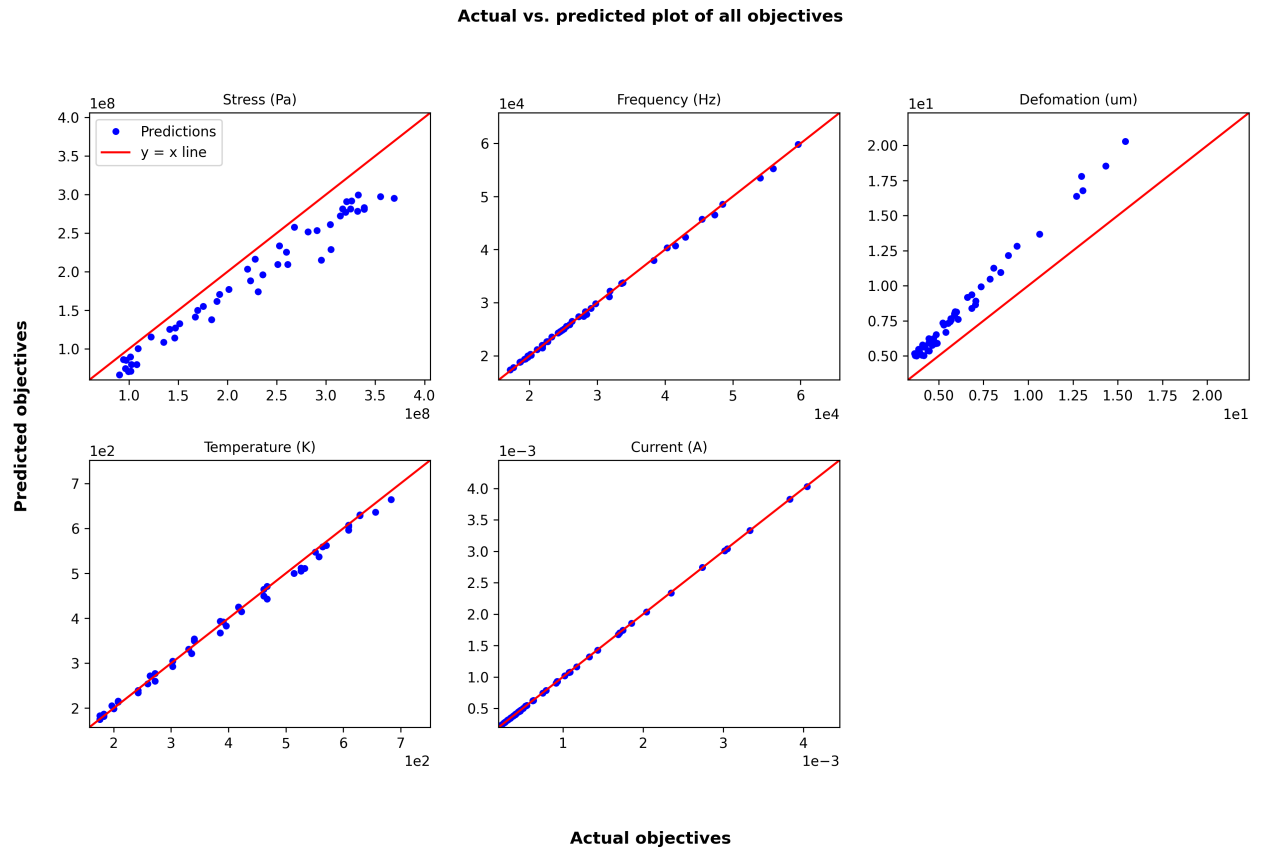


Figure B.38: Actual vs predicted plot for the thermal actuator with a training size of 400: After Regular (SAMOO) Optimization.

References

- [1] F. A. Mohd Ghazali, M. N. Hasan, T. Rehman, M. Nafea, M. S. Mohamed Ali, and K. Takahata, "MEMS actuators for biomedical applications: A review," *Journal of Micromechanics and Microengineering*, vol. 30, no. 7, p. 073001, Jul. 2020.
- [2] C. Chircov and A. M. Grumezescu, "Microelectromechanical Systems (MEMS) for Biomedical Applications," *Micromachines*, vol. 13, no. 2, p. 164, Jan. 2022.
- [3] W. Honda, S. Harada, T. Arie, S. Akita, and K. Takei, "Wearable, Human-Interactive, Health-Monitoring, Wireless Devices Fabricated by Macroscale Printing Techniques," *Advanced Functional Materials*, vol. 24, no. 22, pp. 3299–3304, Jun. 2014.
- [4] S. T. S. Holmstrom, U. Baran, and H. Urey, "MEMS Laser Scanners: A Review," *Journal of Microelectromechanical Systems*, vol. 23, no. 2, pp. 259–275, Apr. 2014.
- [5] S. Stassi, V. Cauda, G. Canavese, and C. Pirri, "Flexible Tactile Sensing Based on Piezoresistive Composites: A Review," *Sensors*, vol. 14, no. 3, pp. 5296–5332, Mar. 2014.
- [6] H. Fourati, N. Manamanni, L. Afilal, and Y. Handrich, "Complementary Observer for Body Segments Motion Capturing by Inertial and Magnetic Sensors," *IEEE/ASME Transactions on Mechatronics*, vol. 19, no. 1, pp. 149–157, Feb. 2014.

- [7] M. M. Atia, S. Liu, H. Nematallah, T. B. Karamat, and A. Noureldin, "Integrated Indoor Navigation System for Ground Vehicles With Automatic 3-D Alignment and Position Initialization," *IEEE Transactions on Vehicular Technology*, vol. 64, no. 4, pp. 1279–1292, Apr. 2015.
- [8] A. Kasturi, V. Milanovic, B. H. Atwood, and J. Yang, "UAV-borne lidar with MEMS mirror-based scanning capability," in *SPIE Defense + Security*, M. D. Turner and G. W. Kamerman, Eds., Baltimore, Maryland, United States, May 2016, p. 98320M.
- [9] Y. Li, J. Georgy, X. Niu, Q. Li, and N. El-Sheimy, "Autonomous Calibration of MEMS Gyros in Consumer Portable Devices," *IEEE Sensors Journal*, vol. 15, no. 7, pp. 4062–4072, Jul. 2015.
- [10] D. Gavel, R. Kupke, D. Dillon, A. Norton, C. Ratliff, J. Cabak, A. Phillips, C. Rockosi, R. McGurk, S. Srinath, M. Peck, W. Deich, K. Lanclos, J. Gates, M. Saylor, J. Ward, and T. Pfister, "ShaneAO: Wide science spectrum adaptive optics system for the Lick Observatory," in *SPIE Astronomical Telescopes + Instrumentation*, E. Marchetti, L. M. Close, and J.-P. Véran, Eds., Montréal, Quebec, Canada, Jul. 2014, p. 914805.
- [11] R. Haindl, W. Trasischker, B. Baumann, M. Pircher, and C. Hitzenberger, "Three-beam Doppler optical coherence tomography using a facet prism telescope and MEMS mirror for improved transversal resolution," *Journal of Modern Optics*, vol. 62, no. 21, pp. 1781–1788, Dec. 2015.
- [12] A. S. Algamili, M. H. M. Khir, J. O. Dennis, A. Y. Ahmed, S. S. Alabsi, S. S. Ba Hashwan, and M. M. Junaid, "A Review of Actuation and Sensing Mechanisms in MEMS-Based Sensor Devices," *Nanoscale Research Letters*, vol. 16, no. 1, p. 16, Jan. 2021.
- [13] P. D. Barba and S. Wiak, "Evolutionary Computing and Optimal Design of MEMS," *IEEE/ASME Transactions on Mechatronics*, vol. 20, no. 4, pp. 1660–1667, Aug. 2015.
- [14] C. Wang, X. Song, W. Fang, F. Chen, I. Zeimpekis, Y. Wang, A. Quan, J. Bai, H. Liu, G. Schropfer, C. Welham, and M. Kraft, "Design of freeform geometries in a MEMS accelerometer with a mechanical

- motion preamplifier based on a genetic algorithm,” *Microsystems & Nanoengineering*, vol. 6, no. 1, pp. 1–15, Nov. 2020.
- [15] M. Farnsworth, E. Benkhelifa, A. Tiwari, and M. Zhu, “A Novel Approach to Multi-level Evolutionary Design Optimization of a MEMS Device,” in *Evolvable Systems: From Biology to Hardware*, ser. Lecture Notes in Computer Science, G. Tempesti, A. M. Tyrrell, and J. F. Miller, Eds. Berlin, Heidelberg: Springer, 2010, pp. 322–334.
- [16] R. Guo, F. Sui, W. Yue, Z. Wang, S. Pala, K. Li, R. Xu, and L. Lin, “Deep learning for non-parameterized MEMS structural design,” *Microsystems & Nanoengineering*, vol. 8, no. 1, pp. 1–10, Aug. 2022.
- [17] D. Giannini, G. Bonaccorsi, and F. Braghin, “Size optimization of MEMS gyroscopes using substructuring,” *European Journal of Mechanics - A/Solids*, vol. 84, p. 104045, Nov. 2020.
- [18] Y. Jin, “Surrogate-assisted evolutionary computation: Recent advances and future challenges,” *Swarm and Evolutionary Computation*, vol. 1, no. 2, pp. 61–70, Jun. 2011.
- [19] T. Chugh, C. Sun, H. Wang, and Y. Jin, “Surrogate-Assisted Evolutionary Optimization of Large Problems,” in *High-Performance Simulation-Based Optimization*, T. Bartz-Beielstein, B. Filipič, P. Korošec, and E.-G. Talbi, Eds. Cham: Springer International Publishing, 2020, vol. 833, pp. 165–187.
- [20] R.-L. Chereches, P. Di Barba, and S. Wiak, “Non-linear inverse problems and optimal design of MEMS,” *Compel*, vol. 34, no. 3, pp. 608–623, 2015.
- [21] B. Liu and A. Nikolaeva, “Efficient global optimization of MEMS based on surrogate model assisted evolutionary algorithm,” in *2016 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, Mar. 2016, pp. 555–558.
- [22] R. Bajwa and M. K. Yapici, “Machine Learning-Based Modeling and Generic Design Optimization Methodology for Radio-Frequency Microelectromechanical Devices,” *Sensors*, vol. 23, no. 8, p. 4001, Jan. 2023.

- [23] J. Giehler, L. Chamard, C. Ebert, J. Weiss, A. Giani, and P. Combette, "Surrogate-based parameter exploration for the design of MEMS calorimetric wall shear stress sensors by means of CFD," *Sensors and Actuators A: Physical*, vol. 347, p. 113949, Nov. 2022.
- [24] M. O. Akinsolu, B. Liu, P. I. Lazaridis, K. K. Mistry, M. E. Mognaschi, P. D. Barba, and Z. D. Zaharis, "Efficient Design Optimization of High-Performance MEMS Based on a Surrogate-Assisted Self-Adaptive Differential Evolution," *IEEE Access*, vol. 8, pp. 80 256–80 268, 2020.
- [25] B. Liu, V. Grout, and A. Nikolaeva, "Efficient Global Optimization of Actuator Based on a Surrogate Model Assisted Hybrid Algorithm," *IEEE Transactions on Industrial Electronics*, vol. 65, no. 7, pp. 5712–5721, Jul. 2018.
- [26] M. O. Akinsolu, B. Liu, V. Grout, P. I. Lazaridis, M. E. Mognaschi, and P. D. Barba, "A Parallel Surrogate Model Assisted Evolutionary Algorithm for Electromagnetic Design Optimization," *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 3, no. 2, pp. 93–105, Apr. 2019.
- [27] S. Saghir, M. M. Saleem, A. Hamza, K. Riaz, S. Iqbal, and R. I. Shakoar, "A Systematic Design Optimization Approach for Multiphysics MEMS Devices Based on Combined Computer Experiments and Gaussian Process Modelling," *Sensors*, vol. 21, no. 21, p. 7242, Jan. 2021.
- [28] P. Di Barba, M. E. Mognaschi, and S. Wiak, "CNN-Based Surrogate Models of the Electrostatic Field for a MEMS Motor: A Bi-Objective Optimal Shape Design," *Electronics*, vol. 11, no. 23, p. 3877, Jan. 2022.
- [29] K. Deb, "Multi-objective Optimisation Using Evolutionary Algorithms: An Introduction," in *Multi-Objective Evolutionary Optimisation for Product Design and Manufacturing*, L. Wang, A. H. C. Ng, and K. Deb, Eds. London: Springer, 2011, pp. 3–34.
- [30] Y. Cui, Z. Geng, Q. Zhu, and Y. Han, "Review: Multi-objective optimization methods and application in energy saving," *Energy*, vol. 125, pp. 681–704, Apr. 2017.
- [31] A.-T. Nguyen, S. Reiter, and P. Rigo, "A review on simulation-based optimization methods applied to building performance analysis," *Applied Energy*, vol. 113, pp. 1043–1058, Jan. 2014.

- [32] G. Lei, J. Zhu, Y. Guo, C. Liu, and B. Ma, "A Review of Design Optimization Methods for Electrical Machines," *Energies*, vol. 10, no. 12, p. 1962, Nov. 2017.
- [33] R. Yondo, E. Andrés, and E. Valero, "A review on design of experiments and surrogate models in aircraft real-time and many-query aerodynamic analyses," *Progress in Aerospace Sciences*, vol. 96, pp. 23–61, Jan. 2018.
- [34] B. Liu, D. Zhao, P. Reynaert, and G. G. E. Gielen, "GASPAD: A General and Efficient mm-Wave Integrated Circuit Synthesis Method Based on Surrogate Model Assisted Evolutionary Algorithm," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 33, no. 2, pp. 169–182, Feb. 2014.
- [35] R. Roy, S. Hinduja, and R. Teti, "Recent advances in engineering design optimisation: Challenges and future trends," *CIRP Annals*, vol. 57, no. 2, pp. 697–715, 2008.
- [36] E. Minisci, M. Vasile, J. Periaux, N. R. Gauger, K. C. Giannakoglou, and D. Quagliarella, Eds., *Advances in Evolutionary and Deterministic Methods for Design, Optimization and Control in Engineering and Sciences*, ser. Computational Methods in Applied Sciences. Cham: Springer International Publishing, 2019, vol. 48.
- [37] T. Chugh, K. Sindhya, J. Hakanen, and K. Miettinen, "A survey on handling computationally expensive multiobjective optimization problems with evolutionary algorithms," *Soft Computing*, vol. 23, no. 9, pp. 3137–3166, May 2019.
- [38] A. Slowik and H. Kwasnicka, "Evolutionary algorithms and their applications to engineering problems," *Neural Computing and Applications*, vol. 32, no. 16, pp. 12 363–12 379, Aug. 2020.
- [39] P. Wang, Q. Lu, and Z. Fan, "Evolutionary design optimization of MEMS: A review of its history and state-of-the-art," *Cluster Computing*, vol. 22, no. 4, pp. 9105–9111, Jul. 2019.

- [40] Z. Fan, J. Liu, T. Sorensen, and P. Wang, "Improved Differential Evolution Based on Stochastic Ranking for Robust Layout Synthesis of MEMS Components," *IEEE Transactions on Industrial Electronics*, vol. 56, no. 4, pp. 937–948, Apr. 2009.
- [41] J. K. Coultate, C. H. Fox, S. McWilliam, and A. R. Malvern, "Application of optimal and robust design methods to a MEMS accelerometer," *Sensors and Actuators A: Physical*, vol. 142, no. 1, pp. 88–96, Mar. 2008.
- [42] S. Koziel and A. Bekasiewicz, "Multi-objective optimization of expensive electromagnetic simulation models," *Applied Soft Computing*, vol. 47, pp. 332–342, Oct. 2016.
- [43] C. B. Erickson, B. E. Ankenman, and S. M. Sanchez, "Comparison of Gaussian process modeling software," *European Journal of Operational Research*, vol. 266, no. 1, pp. 179–192, Apr. 2018.
- [44] R. B. Christianson, R. M. Pollyea, and R. B. Gramacy, "Traditional kriging versus modern Gaussian processes for large-scale mining data," *Statistical Analysis and Data Mining: The ASA Data Science Journal*, vol. 16, no. 5, pp. 488–506, Oct. 2023.
- [45] T. Cui, D. Pagendam, and M. Gilfedder, "Gaussian process machine learning and Kriging for groundwater salinity interpolation," *Environmental Modelling & Software*, vol. 144, p. 105170, Oct. 2021.
- [46] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, Apr. 2002.
- [47] J. Sacks, W. J. Welch, T. J. Mitchell, and H. P. Wynn, "Design and Analysis of Computer Experiments," *Statistical Science*, vol. 4, no. 4, Nov. 1989.
- [48] R. Storn and K. Price, "Differential Evolution – A Simple and Efficient Heuristic for global Optimization over Continuous Spaces," *Journal of Global Optimization*, vol. 11, no. 4, pp. 341–359, Dec. 1997.
- [49] R. Chereches, P. Di Barba, V. Topa, M. Purcar, and S. Wiak, "Optimal shape design of electrostatic microactuators: A multiobjective formulation," *International Journal of Applied Electromagnetics and Mechanics*, vol. 43, no. 1-2, pp. 65–76, Aug. 2013.

- [50] E. Del Castillo, D. C. Montgomery, and D. R. McCarville, “Modified Desirability Functions for Multiple Response Optimization,” *Journal of Quality Technology*, vol. 28, no. 3, pp. 337–345, Jul. 1996.
- [51] L. Breiman, J. Friedman, R. A. Olshen, and C. J. Stone, *Classification and Regression Trees*. New York: Chapman and Hall/CRC, Oct. 2017.
- [52] B. Park, E. Afsharipour, D. Chrusch, C. Shafai, D. Andersen, and G. Burley, “A low voltage and large stroke Lorentz force continuous deformable polymer mirror for wavefront control,” *Sensors and Actuators A: Physical*, vol. 280, pp. 197–204, Sep. 2018.
- [53] O. Brand, I. Dufour, S. Heinrich, and F. Josse, *Resonant MEMS: Fundamentals, Implementation, and Application*, 1st ed. John Wiley & Sons, Ltd, Jun. 2015.
- [54] H. Hussein, A. Tahhan, P. L. Moal, G. Bourbon, Y. Haddab, and P. Lutz, “Dynamic electro-thermo-mechanical modelling of a U-shaped electro-thermal actuator,” *Journal of Micromechanics and Microengineering*, vol. 26, no. 2, p. 025010, Jan. 2016.
- [55] A. I. J. Forrester, A. Söbester, and A. J. Keane, *Engineering Design via Surrogate Modelling: A Practical Guide*, 1st ed. Wiley, Jul. 2008.
- [56] 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*, vol. 42, no. 1, pp. 55–61, Feb. 2000.
- [57] M. E. Johnson, L. M. Moore, and D. Ylvisaker, “Minimax and maximin distance designs,” *Journal of Statistical Planning and Inference*, vol. 26, no. 2, pp. 131–148, Oct. 1990.
- [58] K. Fang, R.-z. Li, and A. Sudjianto, *Design and Modeling for Computer Experiments*, ser. Computer Science and Data Analysis Series. Boca Raton, FL: Chapman & Hall/CRC, 2006.
- [59] “Skopt.sampler.Lhs — scikit-optimize 0.8.1 documentation,” <https://scikit-optimize.github.io/stable/modules/generated/skopt.sampler.Lhs.html>.

- [60] "Installation - MPh 1.2.4," <https://mph.readthedocs.io/en/1.2/installation.html#licenses>.
- [61] Y. Kuang, Q.-A. Huang, and N. K. S. Lee, "Numerical simulation of a polysilicon thermal flexure actuator," *Microsystem Technologies*, vol. 8, no. 1, pp. 17–21, Mar. 2002.
- [62] R. Hickey, M. Kujath, and T. Hubbard, "Heat transfer analysis and optimization of two-beam microelectromechanical thermal actuators," *Journal of Vacuum Science & Technology A: Vacuum, Surfaces, and Films*, vol. 20, no. 3, p. 971, May 2002.
- [63] O. Ozsun, B. E. Alaca, A. D. Yalcinkaya, M. Yilmaz, M. Zervas, and Y. Leblebici, "On heat transfer at microscale with implications for microactuator design," *Journal of Micromechanics and Microengineering*, vol. 19, no. 4, p. 045020, Mar. 2009.
- [64] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and É. Duchesnay, "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research*, vol. 12, no. 85, pp. 2825–2830, 2011.
- [65] P. Saves, R. Lafage, N. Bartoli, Y. Diouane, J. Bussemaker, T. Lefebvre, J. T. Hwang, J. Morlier, and J. R. Martins, "SMT 2.0: A Surrogate Modeling Toolbox with a focus on hierarchical and mixed variables Gaussian processes," *Advances in Engineering Software*, vol. 188, p. 103571, Feb. 2024.
- [66] A. E. Hoerl and R. W. Kennard, "Ridge Regression: Biased Estimation for Nonorthogonal Problems," *Technometrics*, vol. 12, no. 1, pp. 55–67, Feb. 1970.
- [67] F. Santosa and W. W. Symes, "Linear Inversion of Band-Limited Reflection Seismograms," *SIAM Journal on Scientific and Statistical Computing*, vol. 7, no. 4, pp. 1307–1330, Oct. 1986.
- [68] R. Tibshirani, "Regression Shrinkage and Selection Via the Lasso," *Journal of the Royal Statistical Society Series B: Statistical Methodology*, vol. 58, no. 1, pp. 267–288, Jan. 1996.

- [69] B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani, "Least angle regression," *The Annals of Statistics*, vol. 32, no. 2, pp. 407–499, Apr. 2004.
- [70] M. A. Fischler and R. C. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Commun. ACM*, vol. 24, no. 6, pp. 381–395, Jun. 1981.
- [71] M. A. Bouhlef, N. Bartoli, A. Otsmane, and J. Morlier, "Improving kriging surrogates of high-dimensional design models by Partial Least Squares dimension reduction," *Structural and Multidisciplinary Optimization*, vol. 53, no. 5, pp. 935–952, May 2016.
- [72] M. A. Bouhlef and J. R. R. A. Martins, "Gradient-enhanced kriging for high-dimensional problems," *Engineering with Computers*, vol. 35, no. 1, pp. 157–173, Jan. 2019.
- [73] E. Schulz, M. Speekenbrink, and A. Krause, "A tutorial on Gaussian process regression: Modelling, exploring, and exploiting functions," *Journal of Mathematical Psychology*, vol. 85, pp. 1–16, Aug. 2018.
- [74] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning*, ser. Adaptive Computation and Machine Learning. Cambridge, Mass: MIT Press, 2006.
- [75] J. Quiñonero-Candela and C. E. Rasmussen, "A Unifying View of Sparse Approximate Gaussian Process Regression," *Journal of Machine Learning Research*, vol. 6, no. 65, pp. 1939–1959, 2005.
- [76] H. Ma, Y. Zhang, S. Sun, T. Liu, and Y. Shan, "A comprehensive survey on NSGA-II for multi-objective optimization and applications," *Artificial Intelligence Review*, vol. 56, no. 12, pp. 15 217–15 270, Dec. 2023.
- [77] M. Hort and F. Sarro, "The Effect of Offspring Population Size on NSGA-II: A Preliminary Study."
- [78] J. Blank and K. Deb, "Pymoo: Multi-Objective Optimization in Python," *IEEE Access*, vol. 8, pp. 89 497–89 509, 2020.
- [79] Z. He and G. G. Yen, "Comparison of visualization approaches in many-objective optimization," in *2017 IEEE Congress on Evolutionary Computation (CEC)*, Jun. 2017, pp. 357–363.

- [80] T. Tušar and B. Filipič, “Visualization of Pareto Front Approximations in Evolutionary Multiobjective Optimization: A Critical Review and the Prosecution Method,” *IEEE Transactions on Evolutionary Computation*, vol. 19, no. 2, pp. 225–245, Apr. 2015.
- [81] M. Li and X. Yao, “Quality Evaluation of Solution Sets in Multiobjective Optimisation: A Survey,” *ACM Computing Surveys*, vol. 52, no. 2, pp. 1–38, Mar. 2020.
- [82] K. Deb, K. Miettinen, and S. Chaudhuri, “Toward an Estimation of Nadir Objective Vector Using a Hybrid of Evolutionary and Local Search Approaches,” *IEEE Transactions on Evolutionary Computation*, vol. 14, no. 6, pp. 821–841, Dec. 2010.
- [83] M. Li, T. Chen, and X. Yao, “How to Evaluate Solutions in Pareto-based Search-Based Software Engineering? A Critical Review and Methodological Guidance,” *IEEE Transactions on Software Engineering*, vol. 48, no. 5, pp. 1771–1799, May 2022.
- [84] K. Deb, K. Sindhya, and T. Okabe, “Self-adaptive simulated binary crossover for real-parameter optimization,” in *Proceedings of the 9th Annual Conference on Genetic and Evolutionary Computation*, ser. GECCO ’07. New York, NY, USA: Association for Computing Machinery, Jul. 2007, pp. 1187–1194.
- [85] N. Paryab, H. Jahed, and A. Khajepour, “Creep and Fatigue Failure in Single- and Double Hot Arm MEMS Thermal Actuators,” *Journal of Failure Analysis and Prevention*, vol. 9, no. 2, pp. 159–170, Apr. 2009.
- [86] A. Nazari, A. Aghajani, P. Buhr, B. Park, M. Belov, Y. Wang, and C. Shafai, “Surrogate-Assisted Multi-Objective Design Optimization of a Lorentz Force Actuator,” in *2024 IEEE Canadian Conference on Electrical and Computer Engineering (CCECE)*, Aug. 2024, pp. 648–653.