Universität Rostock Traditio et Innovatio

JADEHOCHSCHULE Wilhelmshaven Oldenburg Elsfleth

Parametric Model Order Reduction of Miniaturized Energy Harvesting Modules

Dissertation to obtain the academic degree of

Doctor of Engineering (Dr.-Ing.)

from the Faculty of Computer Science and Electrical Engineering at the University of Rostock

in cooperation with Jade University of Applied Sciences

submitted by

Chengdong Yuan, born on the 10th of April, 1992, in Shanghai, China

Rostock, April 27, 2024

https://doi.org/10.18453/rosdok_id00004752

The work presented is funded by the 'Jade2Pro' Ph.D. program at Jade University of Applied Sciences. It is carried out under the supervision of Prof. Dr.-Ing. Tamara Bechtold at the Department of Engineering, Jade University of Applied Sciences, and Prof. Dr.-Ing. Dennis Hohlfeld at the Institute of Electronic Appliances and Circuits, University of Rostock.

Date of submission: April 27, 2024
Date of defense: November 22, 2024

Reviewers:

Prof. Dr.-Ing. Tamara Bechtold
Chair of Mechatronic Systems
Department of Engineering
Jade University of Applied Sciences

Prof. Dr.-Ing. Dennis Hohlfeld
Chair of Micro- and Nanotechnology of Electronic Systems
Institute of Electronic Appliances and Circuits
University of Rostock

Prof. Dr.-Ing.habil. Jan Mehner
Chair of Microsystems and Medical Engineering
Faculty of Electrical Engineering and Information Technology
Chemnitz University of Technology

Table of Contents

Abstract

Energy harvesting is a technique that carries the potential to provide a lifetime power supply to wireless sensors and has attracted widespread attention in the last two decades. An energy harvester collects various types of ambient energy, e.g., solar, thermal, or vibration, and transforms them into electrical energy. With the development of numerical modeling techniques, performing the design and optimization processes of the energy harvesters in finite element method (FEM)-based software is state-of-the-art. Simulating these finite element (FE) models reduces the cost of fabricating prototypes, thereby speeding up the development cycle.

Conventionally, simulations of multi-physics energy harvesters are performed on a discrete FE grid whose size can easily exceed 100,000 degrees of freedom (DOF), i.e., using ordinary differential equations (ODEs), due to the complex structure of the harvester. Moreover, in the design optimization process, in order to investigate the impact of different design parameters on the performance of the energy harvesters, parametric studies are computed on the large-scale FE models. Both lead to an increased requirement for modern computers to have the capacity to handle large-scale engineering models. In addition, real-time simulation of an energy harvester within a system-level model is prohibitive if full models are used directly. Hence, reducing the model size and performing parametric studies at the reduced-order model (ROM) level is a cornerstone of efficient energy harvester modeling and simulation.

This thesis presents the application of mathematical model order reduction (MOR) and parametric model order reduction (pMOR) methods for generating accurate compact models of thermoelectric, piezoelectric, and electromagnetic energy harvesters. Differing from conventional non-parametric MOR approaches, e.g., Arnoldi algorithm, multivariate moment matching-, matrix interpolation (MI)-, and algebraic parameterization (AP)-based pMOR methods are used to construct a parametric reduced-order model (pROM), which enables the preservation of the predefined parameters, e.g., boundary conditions, material properties, or geometrical parameters, in symbolic form within a pROM and their adjustment within system-level simulations for design optimizations.

Furthermore, in the case study of the thermoelectric energy harvester model, the methodology of using the pROM results is introduced to optimize the substructure model. For stable ROM generation of piezoelectric energy harvester models, stability-preserving MOR methods are developed and combined with the MI-based pMOR method to construct stable pROMs. Finally, computational efficiency and the performance of both MI- and AP-based pMOR methods are investigated and compared. The results obtained in this thesis lead to the successful validation and adaptation of these pMOR methods for more realistic industrial FE energy harvester models. Appropriate scripts are developed to create pROMs automatically to handle the geometrical parameters.

Zusammenfassung

Energy Harvesting ist eine Technik, die das Potenzial hat, drahtlose Sensoren über ihre gesamte Lebensdauer mit Strom zu versorgen. Sie hat in den letzten zwei Jahrzehnten viel Aufmerksamkeit auf sich gezogen. Ein Energy Harvester sammelt verschiedene Arten von Umgebungsenergie, z.B. Licht-, Wärme- oder Vibrationsenergie, und wandelt sie in elektrische Energie um. Mit der Entwicklung numerischer Modellierungstechniken ist die Durchführung der Entwurfs- und Optimierungsprozesse von Energy Harvestern in Finite-Elemente-Methode-basierter Software Stand der Technikgeworden. Die Simulation der Finite-Elemente-Modelle reduziert die Kosten für die Herstellung von Prototypen und beschleunigt so den Entwicklungszyklus.

Konventionell werden die Simulationen von FE-Modellen auf einem diskreten Gitter durchgeführt, dessen Größe aufgrund der komplexen Struktur leicht 100.000 Freiheitsgrad bzw. gewöhnliche Differentialgleichungen überschreiten kann. Darüber hinaus werden im Rahmen der Designoptimierung parametrische Studien an großen FE-Modellen durchgeführt, um die Auswirkungen verschiedener Designparameter auf die Leistung des Energy Harvesters zu untersuchen. Beides führt zu einem erhöhten Bedarf an modernen Computern, die große technische Modelle verarbeiten können. Zudem ist die Echtzeitsimulation des Energy Harvesters zusammen mit einer Schaltung auf Systemebene rechnerisch aufwendig, wenn vollständige Modelle direkt verwendet werden. Die Verkleinerung des Modells und die Durchführung parametrischer Studien von Modellen reduzierter Ordnung ist daher ein Meilenstein für eine effiziente Modellierung und Simulation von Energy Harvestern.

In dieser Arbeit wird die Anwendung von Methoden der mathematischen Modellordnungsreduktion (MOR) und der parametrischen Modellordnungsreduktion (pMOR) zur Generierung präziser kompakter Modelle von thermoelektrischen, piezoelektrischen und elektromagnetischen Energy Harvestern vorgestellt. Im Unterschied zu konventionellen nicht-parametrischen MOR-Ansätzen, z.B. dem Arnoldi-Algorithmus, werden multivariante Moment-Matching-, Matrix-Interpolations- (MI) und algebraische Parametrisierung (AP) basierte pMOR-Methoden verwendet, um ein parametrisches Modell reduzierter Ordnung (pROM) zu konstruieren, Dieses Vorgehen ermöglicht, die vordefinierten Parameter, z.B. Randbedingungen, Materialeigenschaften oder geometrische Parameter, in symbolischer Form innerhalb eines pROM zu erhalten und sie innerhalb von Simulationen auf Systemebene für Design-Optimierungen zu ersetzen.

Darüber hinaus stellen wir in der Fallstudie des thermoelektrischen Energy-Harvester-Modells eine Methodik vor, pROM-Ergebnisse zur weiteren Optimierung des FE-Substrukturmodells zu verwenden. Für die stabile ROM-Generierung der piezoelektrischen Energy-Harvester-Modelle werden stabilitätserhaltende MOR-Methoden entwickelt und mit der MI-basierten pMOR-Methode kombiniert, um stabile pROMs zu konstruieren. Schließlich

werden die Berechnungseffizienz und die Leistung der MI- und AP-basierten pMOR-Methoden untersucht und verglichen. Die in dieser Arbeit erzielten Ergebnisse ermöglichen eine erfolgreiche Validierung und Anpassung dieser pMOR-Methoden an realistische FE-Modelle von Energy Harvestern. Entsprechende Skripte werden entwickelt, um automatisch pROMs zur Behandlung der geometrischen Parameter zu erstellen.

Abbreviations

AP	algebraic parameterization
APDL	ANSYS parametric design language
BC	boundary condition
DOF	degrees of freedom
ECE	equivalent circuit extraction
EMF	electromotive force
FE	finite element
FEM	finite element method
FOM	full-order model
MIMO	multiple-input and multiple-output
MI	matrix interpolation
MOR	model order reduction
ODE	ordinary differential equation
PDE	partial differential equation
pMOR	parametric model order reduction
pROM	parametric reduced-order model
ROM	reduced-order model
SISO	single-input and single-output
SOAR	second order Arnoldi reduction
SVD	singular value decomposition
TEG	thermoelectric generator

List of Figures

List of Tables

Part I

Preliminaries

1 Introduction

This thesis presents the research and application of mathematical model order reduction (MOR) and parametric model order reduction (pMOR) methods for the generation of accurate compact models of thermoelectric, piezoelectric, and electromagnetic energy harvesters. The generated reduced-order model (ROM) and parametric reduced-order model (pROM) can be further utilized within a system-level simulation or used for design optimization.

1.1 Energy Harvesting Techniques

Since the onset of the 'Fourth Industrial Revolution', known as 'Industry 4.0', distributed embedded (micro)systems have been widely implemented in fabrication and logistics, construction and infrastructure, and the environmental, agricultural, and medical fields. These distributed embedded systems collect environmental data via sensors and perform data processing. Furthermore, the wireless communication among these systems has led to the revolution called the 'Internet of Things' (IoT) [1–3]. This is a concept that describes the rapid change in technology and industries as a result of growing interconnectivity and intelligent automation.

The IoT enables machine-to-machine communication using a large number of wireless devices to connect the machines digitally by exchanging sensor data. As a result, the entire manufacturing system is integrated with increasing automation, communication, and self-monitoring and even requiring no human involvement. However, the key challenge for embedded systems is to find a sustainable power supply since manual battery recharging or replacement for multiple embedded systems is not feasible and is too expensive when the distributed systems are not connected to the power grid. Therefore, various energy harvesting techniques have emerged as potential strategies to provide a lifetime power supply to wireless electronic systems and have attracted widespread attention in the last two decades [4, 5]. The energy-autonomous embedded systems developed, which operate self-sufficiently by converting ambient energy, e.g., solar, thermal, or vibration energy, into electrical energy through energy harvesting, can enable new applications (e.g. smart homes), product advantages, and cost reduction.

Nowadays, energy-autonomous systems have become useful in the health and medical fields within the growing societal percentage of elderly people, e.g., recent German demographics show that 22% of the total population are older than 65 years [6], which leads to a large demand for medical implants for therapies such as bone and cartilage regeneration, deep brain stimulation to treat movement disorders, and fixing abnormal heart rates with pacemakers [7]. Nevertheless, the limited lifetime of the batteries in these medical implants is their main drawback, as the necessitated replacement of the drained batteries every 5 to 10 years is accompanied by traumatic surgery. Moreover, their electro-chemical

composition and relatively large size of the batteries have encouraged engineers to develop miniaturized energy harvesters for self-powered long-life implantable medical devices.

In order to create power sources from renewable energy sources, there are various types of energy harvesters. For instance, solar cells use the photovoltaic effect to convert sunlight into electrical power [8, 9]. Piezoelectric, electrostatic, and electromagnetic energy harvesters convert vibration into electrical energy [10–12]. A thermoelectric energy harvester uses the Seebeck effect to generate direct current whenever a temperature gradient is present across the device [13]. In this thesis, piezoelectric, thermoelectric, and electromagnetic energy harvester are presented as case studies as shown in Figure 1.

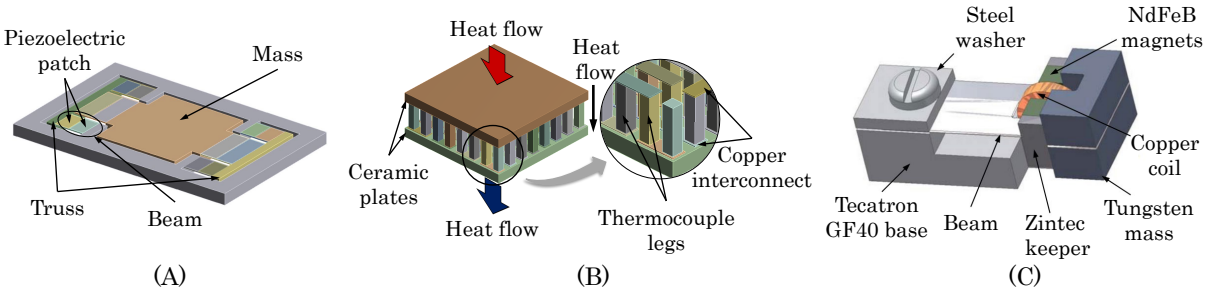


Figure 1: (A) Piezoelectric energy harvester proposed in [14]. (B) Thermoelectric energy harvester proposed in [15]. (C) Electromagnetic energy harvester proposed in [16].

1.2 Model Order Reduction

Nowadays, numerical simulation has become state-of-the-art in enhancing the design and optimization process of the energy harvester. The main advantage of using numerical models is that a software-based simulation reduces the necessity of fabricating prototypes and thereby speeds up the design and optimization cycle. In this work, the above three types of energy harvester models are implemented in the finite element method (FEM)-based software ANSYS Mechanical [17, 18].

The basic framework of the software is that, based on the laws of physics, the behavior of the device can be described by a partial differential equation (PDE) system which is to be solved numerically. After the finite element (FE)-based spatial discretization, the original model can be approximated by an ordinary differential equation (ODE) system, whose size can easily range between 10^6-10^9 degrees of freedom (DOF). However, the capacity of modern workstation computers can hardly perform the demanding and complex simulations of detailed models with refined mesh. Such models contain a lot of elements, requiring a high computational cost to solve the FE model. Therefore, mathematical MOR methods, e.g., Krylov subspace projection methods, are introduced in [19–22] to improve simulation efficiency. As illustrated in Figure 2, the goal of the MOR methods is to generate a highly compact but accurate ROM to replace the original large-scale full-order model (FOM) and enable the simulation of the model at system level.

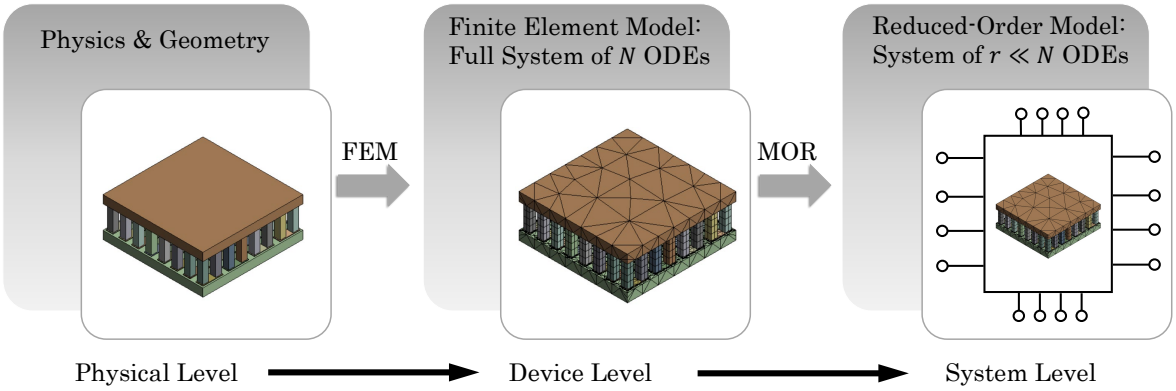


Figure 2: Motivation for model order reduction of large-scale finite element models.

Additionally, for the design optimization of the energy harvester models, studies of the parameters, e.g., material properties, boundary conditions, and geometry, are conducted. Although it is possible to construct a compact model via MOR for fast simulations, the ROM still has to be generated repeatedly for each new value of the parameters in the system. Therefore, pMOR methods are developed for the reduction of parameterized systems [23, 24]. The pROM is constructed such that all parameters can be preserved at the reduced model level as shown in Figure 3. In this way, parametric studies, which play an important role in practical applications, can be performed efficiently with acceptable accuracy.

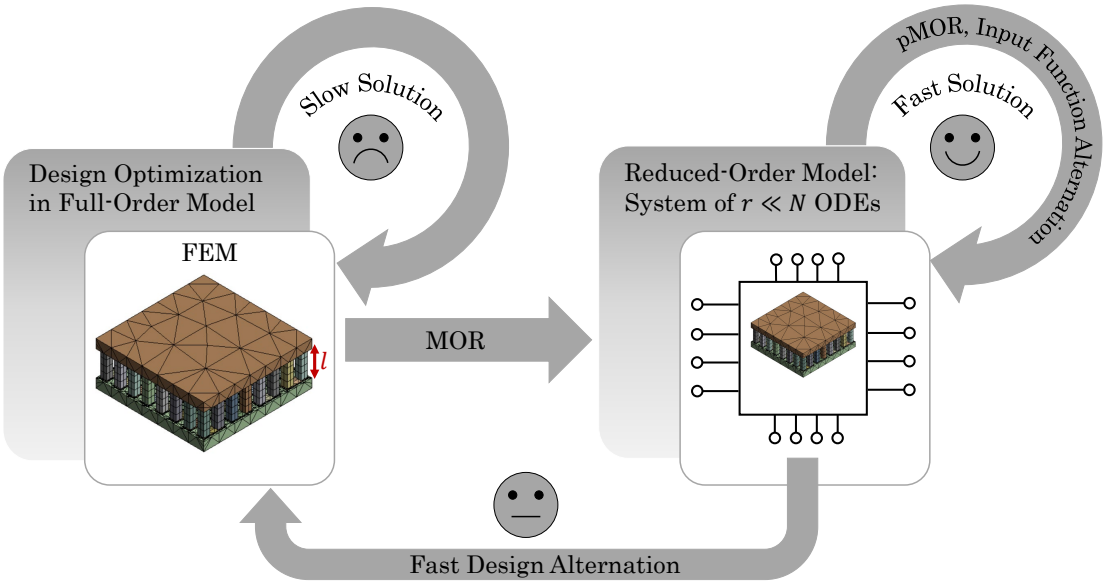


Figure 3: Application of MOR and pMOR for efficient parametric studies.

In addition, on the basis of the ROMs and pROMs, it is possible to extend the simulation from the device level to the system level, where the power management circuits of energy harvesters can be modelled and analyzed in transient simulations.

1.3 Thesis Overview

In chapter 2, the commonly used Krylov subspace-based MOR methods for the reduction of linear non-parametric first- and second-order ODE systems are presented. Such systems can be utilized to describe those energy harvester FE models which contain linear-elastic materials, constant boundary conditions, and fixed geometry. All MOR methods are based on the projection of the original large-scale system onto a lower-dimensional Krylov subspace, whose basis is orthogonalized via the Arnoldi algorithm.

In chapter 3 of this thesis, three pMOR methods are presented and studied, i.e., multivariate moment matching, matrix interpolation (MI), and algebraic parameterization (AP), which are based on the Krylov methods introduced in chapter 2. Compared to conventional MOR methods, the pMOR methods enable the generation of pROMs, which preserve the parameters in symbolic form and thus avoid the repeated generation of the ROM when altering the parameter value.

Given its rapid development in recent years, the commercially available industrial software 'Model Reduction inside ANSYS' (MORiA) [25, 26] will be introduced in chapter 4. This software enables the extraction of the full-order ODE system matrices from the linearized model in ANSYS Mechanical and their reduction via the algorithms introduced in chapter 2. Furthermore, the multivariate moment matching-based pMOR method introduced in chapter 3 is also implemented in 'Model Reduction inside ANSYS' for automatic generation of a linearized pROM. In this thesis, the development of MORiA is advanced in the direction of MI-, and AP-based pMOR methods. Our current implementation is based on Python (2.7).

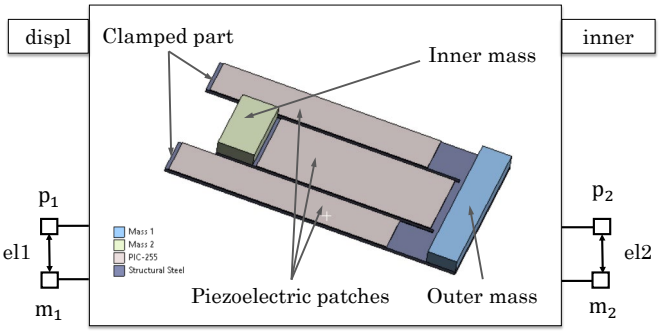
In chapter 5, three energy harvester devices are used as case studies for MOR and pMOR methods. Piezoelectric and electromagnetic energy harvesters are commonly utilized in vibration energy harvesting, which converts vibrations into electrical energy. The thermoelectric energy harvester provides the electrical power from the temperature difference across the device. All the devices are implemented in the FEM-based software ANSYS Mechanical. Moreover, three simplified versions of these three energy harvesters are introduced.

Chapter 6 presents the numerical results of MOR and pMOR for the piezoelectric energy harvester model. Its reduction via conventional MOR process leads to a stability issue in the ROM. Hence, more stable compact modeling approaches are suggested and further combined with the MI-based pMOR methods for generating stable pROMs. Chapters 7 and 8 present the numerical results of pMOR for the thermoelectric and electromagnetic energy harvester models. In addition, the results of system-level simulations of the pROMs are also presented in these chapters.

At the end of each chapter, a summary is given. The conclusions of this thesis and an outlook for future research work are given in chapter 9.

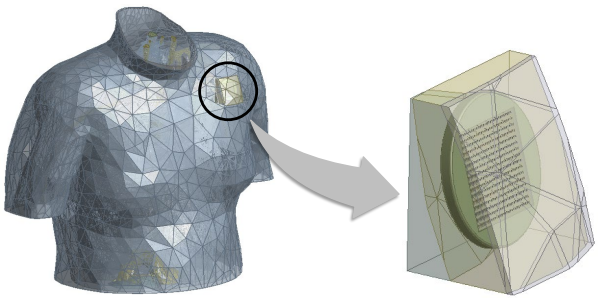
1.4 Major Results

STABLE MOR OF PIEZOELECTRIC ENERGY HARVESTER



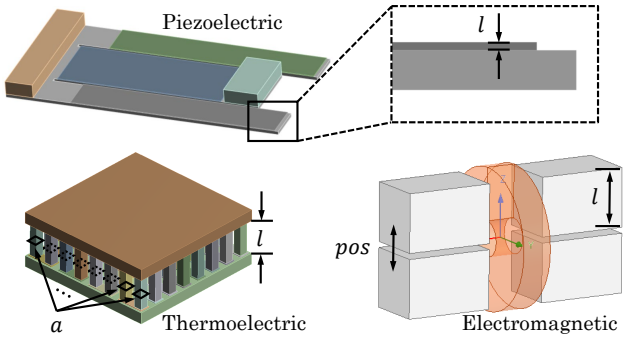
Based on the framework of the conventional Krylov subspace-based MOR, the compact model of a piezoelectric energy harvester device can be generated. However, according to the case studies presented in the previous research and this thesis, the stability of the reduced model cannot be preserved for this multi-physics model. Three stable MOR approaches, i.e., *MOR after Schur*, *Schur after MOR*, and *multi-physics structure preserving MOR* methods, have proven successful in obtaining stable reduced piezoelectric energy harvester models. In this work, a new method called *MOR after implicit Schur* is proposed. It is developed as an improved version of the method *MOR after Schur*. These four methods are validated on two different piezoelectric energy harvester devices and their computational efficiency is compared.

MOR OF A BIOHEAT HUMAN TISSUE MODEL AND COMBINATION OF pMOR WITH THE SUBMODELING TECHNIQUE



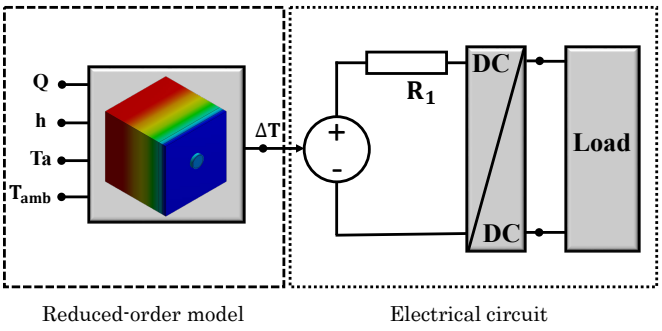
In this work, the thermoelectric energy harvester model is implanted in the fat layer of a realistic human torso model, where the maximum temperature difference is observed. Pennes' bioheat equation is used to characterize heat transfer in human tissue and the temperature-dependent blood perfusion effect is considered. Since conventional Krylov subspace-based MOR methods are designed for linear systems, a method is proposed to implement the blood perfusion heat generation as a 'convection-type' boundary condition. In addition, linearization strategies are implemented to approximate the nonlinear radiation and evaporation heat loss effects at the skin surface, which describe the continuous energy transfer from the human body to the environment. Furthermore, the impact of environmental boundary conditions upon a thermoelectric energy harvester is investigated efficiently, based on the method of combining the pROM of a human body with the submodeling technique.

APPLICATION OF pMOR METHODS TO GEOMETRICALLY PARAMETERIZED ENERGY HARVESTERS



The proposed pMOR methods are applied to three energy harvester models. The first one is a piezoelectric energy harvester. The thickness of the piezoelectric patches is chosen as the geometrical parameter. The MI-based pMOR method is further combined with stable MOR approaches to generate a stable pROM. The second example is a thermoelectric energy harvester. The height and cross-sectional area of the thermocouple legs are selected as the geometrical parameters for optimizing the power output of the device. The third example is an electromagnetic energy harvester model, where the position and the height of the magnets are parameterized. With these models, it is demonstrated that the pMOR methods work with models in which the parameters enter the system matrices in a nonlinear way, i.e., cannot be factored before the system matrices.

CO-SIMULATION OF THE ROM AND THE pROM WITH ELECTRICAL CIRCUITRY



By using efficient but highly accurate ROMs and pROMs, the computational time for transient simulation is decreased by several orders of magnitude, and a system-level simulation, that is the co-simulation of the device and its driving circuitry can be constructed. For the piezoelectric energy harvester, a multiphysics ROM is considered as a power supply for the electrical circuit at the system level. For the thermoelectric energy harvester, a purely thermal FE model is considered and reduced. The voltage output is then calculated via the Seebeck equation in the system-level circuit. This simplifies the reduction process, as it is limited to a single-domain model. For the electromagnetic energy harvester model, the equivalent circuit extraction technique is deployed to export a compact model from a parametric FE solution. This process can be accelerated efficiently via the pMOR approaches. With these case studies, it is demonstrated that the ROM and pROM enable system-level simulations incorporating several interacting components.

2 Model Order Reduction Using Krylov Subspaces

Numerical simulation techniques enable time and cost savings during the design and optimization process of energy harvester devices. However, the three-dimensional (3D) energy harvester finite element (FE) models usually result in high-dimensional ordinary differential equations (ODEs), causing a high computational cost. To speed up the simulations, the Krylov subspace-based model order reduction (MOR) methods are proposed and proven to be the best choice for the reduction of high-order linear dynamical systems [21, 27]. The main goal of this section is to review the reduction of linear systems on the basis of the corresponding Krylov subspaces.

2.1 Moments of the Transfer Function

Let's consider a dynamical first-order multiple-input and multiple-output (MIMO) ODE system of form

$$\Sigma_N \begin{cases} E \cdot \dot{x}(t) = A \cdot x(t) + B \cdot u(t) \\ y(t) = C \cdot x(t) \end{cases} , \tag{2.1}$$

where $E, A \in \mathbb{R}^{N \times N}$ are given system matrices and $x \in \mathbb{R}^N$ is the unknown state vector. N is the dimension of the system. $B \in \mathbb{R}^{N \times m}$ is the input distribution array and $u \in \mathbb{R}^m$ is the input vector. $C \in \mathbb{R}^{p \times N}$ is the gathering matrix, which is used to specify the p observer-interested outputs within the output vector $y \in \mathbb{R}^p$. For single-input and single-output (SISO) systems, $m = p = 1$, the matrices B and C are vectors and noted as b and c^T. A first-order system represented in Equation (2.1) together with appropriate initial and boundary conditions can be used to describe, for example, a spatially discretized transient thermal model. In the transient thermal model, matrices E and A are the heat capacity and the heat conductivity matrices, respectively. The state vector x represents the unknown nodal temperatures.

The transfer function of the system (2.1) with input $u(t)$ and output $y(t)$ in the Laplace domain reads

$$\begin{aligned} G(s) &= C(sE - A)^{-1}B \\ &= C(sE - A + s_0E - s_0E)^{-1}B \\ &= C[-(A - s_0E) + (s - s_0)E]^{-1}B \\ &= -C[I - \underbrace{(A - s_0E)^{-1}E(s - s_0)}_{T}]^{-1}(A - s_0E)^{-1}B \, . \end{aligned} \tag{2.2}$$

According to the Neumann series $\sum_{i=0}^{\infty} T^i = (I - T)^{-1}$, where T is a bounded operator on

the normed space, the transfer function in Equation (2.2) can be rewritten as

$$G(s) = \sum_{i=0}^{\infty} -C[(A - s_0E)^{-1}E]^i(A - s_0E)^{-1}B(s - s_0)^i \ , \tag{2.3}$$

where the moments $m_i \in \mathbb{R}^{p\times m}$ are defined as the coefficients around $s_0 \neq 0$,

$$m_i = -C[(A - s_0E)^{-1}E]^i(A - s_0E)^{-1}B \ , \ i = 0, 1, 2, \cdots \ . \tag{2.4}$$

As the moments reflect system behavior around the expansion point s_0, choosing small values of s_0 will result in a reduced-order model (ROM) with a good approximation of slow dynamics, while large values of s_0 will construct a ROM that is accurate at high frequencies. In addition, to achieve a good approximation of the ROM at steady state, $s_0 = 0$ is selected and the moments are calculated as follows:

$$m_i = -C(A^{-1}E)^iA^{-1}B \ , \ i = 0, 1, 2, \cdots \ . \tag{2.5}$$

It should be noted here that it is also possible to match the moments about different frequency domains simultaneously. This is required to get a better approximation of either a wider or a more specific frequency band as the case may be.

2.2 Krylov Subspace

The kth block Krylov subspace is defined as

$$\mathcal{K}_k(\tilde{A}, \tilde{B}) = span\{\tilde{B}, \tilde{A}\tilde{B}, \tilde{A}^2\tilde{B}, \cdots, \tilde{A}^{k-1}\tilde{B}\} \ , \tag{2.6}$$

where $\tilde{A} \in \mathbb{R}^{N\times N}$. The columns in $\tilde{B} \in \mathbb{R}^{N\times m}$ are called starting vectors. In this work, in order to reduce the system in Equation (2.1), the one-sided Krylov method is chosen because it is numerically more stable. The projection matrix $V \in \mathbb{R}^{N\times r}$ is constructed on the basis of the input Krylov subspace defined as follows:

$$colspan\{V\} = \mathcal{K}_r\{\underbrace{(A - s_0E)^{-1}E}_{\tilde{A}}, \underbrace{(A - s_0E)^{-1}B}_{\tilde{B}}\} \ , \tag{2.7}$$

where $\tilde{A}, \tilde{B}$ are the moments as presented in Equation (2.4). Notice that the main goal of the projection-based MOR method is to generate a highly accurate and compact model and the basic idea is projecting the original high-dimensional system in Equation (2.1) onto a low-dimensional subspace. Thereby, the original full-scale state vector $x \in \mathbb{R}^N$ can be replaced by the following projection equation:

$$x = V \cdot z + \varepsilon \ , \tag{2.8}$$

where $z \in \mathbb{R}^r$ is the reduced state vector, $r \ll N$, and $\varepsilon \in \mathbb{R}^N$ is the approximation error, which is to be minimized according to some norm. The Galerkin condition for the orthogonal projection imposes $\varepsilon \perp V$. After inserting Equation (2.8) into Equation (2.1) and multiplying it by the transpose of the projection matrix V^T from the left to construct an orthogonal projection onto the subspace V, a first-order ROM is obtained as follows:

$$\Sigma_r \begin{cases} \underbrace{V^T E V}_{E_r} \cdot \dot{z}(t) = \underbrace{V^T A V}_{A_r} \cdot z(t) + \underbrace{V^T B}_{B_r} \cdot u(t) \\ y(t) = \underbrace{CV}_{C_r} \cdot z(t) \end{cases} , \tag{2.9}$$

where $E_r, A_r \in \mathbb{R}^{r \times r}$, $B_r \in \mathbb{R}^{r \times m}$, and $C_r \in \mathbb{R}^{p \times r}$ are the reduced system matrices.

2.3 Arnoldi Iteration

This research first considers a single-input system, where the input matrix B is a vector denoted by b. Then the Krylov subspace in Equation (2.7) is written as

$$colspan\{V\} = \mathcal{K}_r\{\underbrace{(A - s_0E)^{-1}E}_{\tilde{A}}, \underbrace{(A - s_0E)^{-1}b}_{\tilde{b}}\} . \tag{2.10}$$

Equation (2.10) defines a low dimensional subspace spanned by the vectors in the Krylov subspace. However, computing the Krylov vectors directly is conventionally unstable, i.e., the rounding error will make it hard to distinguish between $\tilde{A}^k\tilde{b}$ and $\tilde{A}^{k+1}\tilde{b}$ for large k. The remedy is to construct orthogonal basis vectors $\{q_1, q_2, \cdots, q_r\}$ such that

$$colspan\{V\} = \mathcal{K}_r(\tilde{A}, \tilde{b}) = span\{q_1, q_2, \cdots, q_r\} . \tag{2.11}$$

The iteration to compute these orthonormal vectors is called the Arnoldi iteration [28].

Algorithm 1 Arnoldi iteration

Input: Matrix $\tilde{A}$, start vector $\tilde{b}$, dimension of the ROM r.

Set $\hat{q}_1 = \tilde{b}$.
for $i = 1, 2, \cdots, r$ **do** :
    Compute $h_{i,i-1} = \|\hat{q}_i\|$. If $h_{i,i-1} = 0$, then stop.
    Set $q_i = \frac{\hat{q}_i}{h_{i,i-1}}$.
    Set $\hat{q}_{i+1} = \tilde{A}q_i$.
    for $j = 1, \cdots, i$ **do** :
        Set $h_{j,i} = {q_j}^T \hat{q}_{i+1}$.
        $\hat{q}_{i+1} = \hat{q}_{i+1} - q_j \cdot h_{j,i}$.
    end for
end for

Output: Orthonormal vectors $q_1, q_2, \cdots, q_r$.

As shown in Algorithm 1, the norm of the newly added orthogonal vector $\hat{q}_i$ is checked in line 3. The algorithm is stopped if it is zero or close to a numerically small value. Then the newly added orthogonal vector is normalized in line 4 and the next vector $\hat{q}_{i+1}$ is found in line 5. In line 8, the next vector $\hat{q}_{i+1}$ is updated as the vector orthogonal to the previously generated orthogonal vector q_j. It could be understood that each step of the algorithm adds a new orthonormal vector to the projection basis; in other words, the process is iterative. The Arnoldi iteration algorithm calculates $r+1$ orthogonal vectors and the first r vectors are normalized into a basis of the Krylov subspace $\mathcal{K}_r(\tilde{A}, \tilde{b})$.

2.3.1 Block Arnoldi

The Arnoldi iteration can also be generalized to the case of multiple input vectors. Let's consider the block Krylov subspace in Equation (2.6), where the input matrix $\tilde{B} = [\ \tilde{b}_1\ \tilde{b}_2\ \cdots\ \tilde{b}_m\ ]$ contains m columns. Then the block Krylov reads

$$\mathcal{K}_k(\tilde{A}, \tilde{B}) = span\{\tilde{b}_1, \cdots, \tilde{b}_m, \tilde{A}\tilde{b}_1, \cdots, \tilde{A}\tilde{b}_m, \cdots, \tilde{A}^{k-1}\tilde{b}_1, \cdots, \tilde{A}^{k-1}\tilde{b}_m\}\ , \tag{2.12}$$

and the block Arnoldi iteration from [29] is deployed to construct the bases.

Algorithm 2 Block Arnoldi iteration

Input: Matrix $\tilde{A}$, start vectors in $\tilde{B}$, number of inputs m, dimension of the ROM r.

1: **for** $i = 1, 2, \cdots, m$ **do** :
2: Set $\hat{q}_i = \tilde{b}_i$.
3: **end for**
4: Set $m_c = m$.
5: **for** $n = 1, 2, \cdots, r$ **do** :
6: **if** $\|\hat{q}_n\| \leq dtol$ (deflation criterion) **then** :
7: Set $m_c = m_c - 1$.
8: **If** $m_c = 0$, set $n = n - 1$ and stop.
9: Set $\hat{q}_i = \hat{q}_{i+1}$ for $i = n, n+1, \cdots, n+m_c-1$.
10: Return to line 6.
11: **end if**
12: $t_{n,n-m_c} = \|\hat{q}_n\|$ and $q_n = \frac{\hat{q}_n}{t_{n,n-m_c}}$.
13: Compute $\hat{q}_{n+m_c} = \tilde{A}q_n$.
14: **for** $i = 1, 2, \cdots, n$ **do** :
15: Set $t_{i,n} = q_i^T \hat{q}_{n+m_c}$.
16: $\hat{q}_{n+m_c} = \hat{q}_{n+m_c} - q_i t_{i,n}$.
17: **end for**
18: **for** $i = n - m_c + 1, n - m_c + 2, \cdots, n-1$ **do** :
19: $t_{n,i} = q_n^T \hat{q}_{i+m_c}$.
20: $\hat{q}_{i+m_c} = \hat{q}_{i+m_c} - q_n t_{n,i}$.
21: **end for**
22: **end for**

Output: Orthonormal vectors $q_1, q_2, \cdots, q_r$.

In Algorithm 2, total $r + m_c$ orthogonal vectors of the block Krylov subspace are calculated, but only the first r vectors are normalized. Similar to the Arnoldi iteration in Algorithm 1, in each step of the iteration, a new orthonormal vector is added to the projection basis. The norm of the newly calculated orthogonal vector is checked in line 6 to avoid a relatively small orthogonal vector. The next vector is computed in line 13 and orthogonalized in the next two loops starting from line 14. If the newly calculated vector $\|\hat{q}_n\|$ is deflated, the indices of all remaining candidate vectors are shifted by -1 and the block Krylov subspace is exhausted when $m_c = 0$.

2.3.2 Superposition-Based Arnoldi Reduction Method

Another alternative approach to reducing a system with multiple inputs is to use superposition. The basic idea of such a method is that a m-input system is split into m single-input subsystems and the reduction of these subsystems is performed independently [30]. For example, in a first-order system with $m = 2$ inputs, the reduced system can be obtained via superposition Arnoldi as follows:

$$\Sigma_r \begin{cases} \begin{bmatrix} V_1^T E V_1 & 0 \\ 0 & V_2^T E V_2 \end{bmatrix} \begin{bmatrix} \dot{z}_1(t) \\ \dot{z}_2(t) \end{bmatrix} = \begin{bmatrix} V_1^T A V_1 & 0 \\ 0 & V_2^T A V_2 \end{bmatrix} \begin{bmatrix} z_1(t) \\ z_2(t) \end{bmatrix} + \begin{bmatrix} V_1^T b_1 & 0 \\ 0 & V_2^T b_2 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} \\ y(t) = \begin{bmatrix} CV_1 & CV_2 \end{bmatrix} \begin{bmatrix} z_1(t) \\ z_2(t) \end{bmatrix} \end{cases}, \quad (2.13)$$

where $V_1, V_2 \in \mathbb{R}^{N \times r}$ are the projection matrices for the single-input subsystems. They are constructed independently via the Arnoldi iteration in Algorithm 1 with start vectors $\tilde{b}_1, \tilde{b}_2 \in \mathbb{R}^{N \times 1}$, respectively. The two sets of the reduced systems are combined together by means of the output matrix.

The advantage of using the superposition Arnoldi method is that, compared to the dense matrices generated via the block Arnoldi algorithm, it generates a ROM with sparse system matrices. In practical applications, the number of inputs can be significant. Thereby, if using the superposition Arnoldi algorithm, the dimension of the reduced model will be a product of the number of DOF per input by the number of inputs ($r \times m$) and could reach 1000 or more. For example, let's consider a system with 100 inputs and the number of DOF per input needed in the ROM is 10. The dimension of the ROM gained by superposition Arnoldi is 10×100=1000 DOF. If using the block Arnoldi algorithm to construct a ROM of the same dimensions, each dense reduced system matrix contains nearly 1000×1000 nonzero values, resulting in high computational cost. However, the reduced matrices generated via superposition Arnoldi are in a block diagonal form with 100 blocks each of dimension 10. The number of nonzero values in each reduced system matrix is then one hundred times smaller and this speeds up the ROM simulation.

2.4 Moment Matching

Comparing Equations (2.1) and (2.9), it is observed that the ROM has the same form as FOM but the dimensions of the system matrices are much smaller. The transfer function of the ROM can also be developed into a series around an expansion point s_0 and reads

$$G_r(s) = \sum_{i=0}^{\infty} -C_r[(A_r - s_0E_r)^{-1}E_r]^i(A_r - s_0E_r)^{-1}B_r(s - s_0)^i \ , \tag{2.14}$$

where the moments of the reduced system $m_i^{(r)} \in \mathbb{R}^{p\times m}$ are defined as

$$m_i^{(r)} = -C_r[(A_r - s_0E_r)^{-1}E_r]^i(A_r - s_0E_r)^{-1}B_r \ , \ i = 0, 1, 2, \cdots \ . \tag{2.15}$$

The approximation is related to the fact that

$$m_i = m_i^{(r)} \ , \ i = 0, 1, 2, \cdots, r-1 \ , \tag{2.16}$$

which means the first r moments of the reduced system are equal to those of the original system; in other words, $G_r(s)$ is a Padé-type approximation of $G(s)$ [31].

Proof. (This proof is given in [29]) From the Equations (2.4) and (2.7), it is clear that for the moments m_i of the original system ($i = 0, 1, \cdots, r-1$), the term $Q = [(A - s_0E)^{-1}E]^i(A - s_0E)^{-1}B \in \mathcal{K}_r$. Since the columns of V are the orthonormal basis of the Krylov subspace $\mathcal{K}_r$, then the columns of Q can be represented as a linear combination of the columns of V. Therefore, there exists a matrix P, which satisfies the following equation for an arbitrary $i = q$, $q \in [0, r-1]$,

$$[(A - s_0E)^{-1}E]^q(A - s_0E)^{-1}B = V \cdot P \ , \tag{2.17}$$

$$E^q(A - s_0E)^{-1}B = (A - s_0E)^qV \cdot P \ . \tag{2.18}$$

By multiplying V^T from the left on both sides of Equation (2.18) and inserting $VV^T = I$,

$$\underbrace{V^TE^qV}_{{E_r}^q}\underbrace{V^T(A - s_0E)^{-1}V}_{(A_r - s_0E_r)^{-1}}\underbrace{V^TB}_{B_r} = \underbrace{V^T(A - s_0E)^qV}_{(A_r - s_0E_r)^q} \cdot P \ ,$$

$$(A_r - s_0E_r)^{-q}{E_r}^q(A_r - s_0E_r)^{-1}B_r = P \ . \tag{2.19}$$

Taking Equations (2.17) and (2.19) into the qth moment of m_i,

$$\begin{aligned} m_q &= -C \cdot V \cdot P \\ &= -C_r(A_r - s_0E_r)^{-q}{E_r}^q(A_r - s_0E_r)^{-1}B_r = m_q^{(r)} \ . \end{aligned} \tag{2.20}$$

It is proved that the first r moments of the full and reduced systems match. □

2.5 Model Order Reduction of Second-Order Systems

Similarly to the first-order system, a dynamic second-order MIMO ODE system of the following form is considered:

$$\Sigma_N \begin{cases} M \cdot \ddot{x}(t) + E \cdot \dot{x}(t) + K \cdot x(t) = B \cdot u(t) \\ y(t) = C \cdot x(t) \end{cases} , \tag{2.21}$$

where $M, E, K \in \mathbb{R}^{N \times N}$, $x \in \mathbb{R}^N$ are given system matrices and the unknown state vector. Such a second-order system in matrix form is usually used to represent structural dynamics with damping effects. Thereby, matrices M, E, K are the mass, damping, and stiffness matrices, respectively. Unknown state vector x contains the displacements or rotational DOF in each node. To construct a second-order ROM, the projection Equation (2.8) is inserted into Equation (2.21) and it is multiplied by V^T from the left to construct an orthogonal projection onto the subspace V as follows:

$$\Sigma_r \begin{cases} \underbrace{V^T M V}_{M_r} \cdot \ddot{z}(t) + \underbrace{V^T E V}_{E_r} \cdot \dot{z}(t) + \underbrace{V^T K V}_{K_r} \cdot z(t) = \underbrace{V^T B}_{B_r} \cdot u(t) \\ y(t) = \underbrace{C V}_{C_r} \cdot z(t) \end{cases} , \tag{2.22}$$

where $M_r, E_r, K_r \in \mathbb{R}^{r \times r}$, $B_r \in \mathbb{R}^{r \times m}$, and $C_r \in \mathbb{R}^{p \times r}$ are the reduced system matrices. It can be observed in both first- and second-order systems that, although the ROMs in Equations (2.9) and (2.22) have the same structure as the FOMs in Equations (2.1) and (2.21), the dimensions of the reduced system matrices and the state vectors are much smaller, $r \ll N$. The numbers of inputs and outputs in the systems are preserved. Similar to the first-order systems, the main goal here is to construct a projection matrix V.

The starting point is the transfer function of (2.21), which is defined without expansion point ($s_0 = 0$) as follows:

$$G(s) = C(s^2 M + sE + K)^{-1} B , \tag{2.23}$$

and can be rewritten around expansion point $s_0 \neq 0$ as follows:

$$G(s) = C[(s - s_0)^2 M + (s - s_0)\tilde{E} + \tilde{K}]^{-1} B , \tag{2.24}$$

with $\tilde{E}$, $\tilde{K}$ defined as follows:

$$\tilde{E} = 2s_0 M + E, \quad \tilde{K} = s_0^2 M + s_0 E + K . \tag{2.25}$$

According to the research work in [32–35], there exist three ways to construct the projection matrix for the second-order systems based on the Krylov subspace.

1) Transforming second-order to first-order system. In the first case, the transfer function in Equation (2.23) is rewritten into the linear form in variable s [32], as follows:

$$G(s) = \hat{C}(sL - G)^{-1}\hat{B} \ , \tag{2.26}$$

where

$$L = \begin{bmatrix} E & M \\ -I & 0 \end{bmatrix} , \ G = - \begin{bmatrix} K & 0 \\ 0 & I \end{bmatrix} , \ \hat{B} = \begin{bmatrix} B \\ 0 \end{bmatrix} , \ \hat{C} = \begin{bmatrix} C & 0 \end{bmatrix} \ . \tag{2.27}$$

Based on the matrices in Equation (2.27), the second-order system in Equation (2.21) can be transformed into the first-order one of the form

$$\Sigma_N \begin{cases} L \cdot \dot{\hat{x}}(t) = G \cdot \hat{x}(t) + \hat{B} \cdot u(t) \\ y(t) = \hat{C} \cdot \hat{x}(t) \end{cases} , \ \hat{x} = \begin{bmatrix} x \\ \dot{x} \end{bmatrix} \ . \tag{2.28}$$

As introduced in section 2.2, the Krylov subspace constructed for the first-order system in Equation (2.28) can be defined as follows:

$$colspan\{V\} = \mathcal{K}_r\{(G - s_0 L)^{-1}L, (G - s_0 L)^{-1}\hat{B}\} \ . \tag{2.29}$$

Thereby, the block Arnoldi iteration introduced in section 2.3.1 can be utilized to construct the orthonormal vectors for the projection matrix V.

2) Second-order Krylov subspace and second order Arnoldi reduction (SOAR). In the second case, a second-order Krylov subspace is constructed [32–34]. For a second-order system with a single input, its transfer function is transformed into first-order as presented in Equation (2.26), and it can be rewritten as follows:

$$G(s) = \hat{C}(sL - G)^{-1}\hat{b} = -\hat{C}(I - sH)^{-1}\hat{b}_0 \ , \tag{2.30}$$

where $H = G^{-1}L$ and $\hat{b}_0 = G^{-1}\hat{b}$. Based on the definition of the Krylov subspace in Equation (2.7), the standard rth order Krylov subspace can be defined as follows:

$$\mathcal{K}_r\{H, \hat{b}_0\} = span\{\hat{b}_0, H\hat{b}_0, H^2\hat{b}_0, ..., H^{r-1}\hat{b}_0\} \ . \tag{2.31}$$

Considering the matrices L and G defined in Equation (2.27), matrices H and $\hat{b}_0$ are expressed as follows:

$$H = G^{-1}L = \begin{bmatrix} -K^{-1}E & -K^{-1}M \\ I & 0 \end{bmatrix} = \begin{bmatrix} A_1 & B_1 \\ I & 0 \end{bmatrix} , \ \hat{B}_0 = \begin{bmatrix} -K^{-1}b \\ 0 \end{bmatrix} , \tag{2.32}$$

where $A_1 = -K^{-1}E$ and $B_1 = -K^{-1}M$. Let

$$\begin{aligned} r_0 &= -K^{-1}b \ , \\ r_1 &= A_1 r_0 \ , \\ r_j &= A_1 r_{j-1} + B_1 r_{j-2} \ for \ j \geq 2 \ . \end{aligned}$$

The jth term in the Krylov subspace in Equation (2.31) can be derived as follows:

$$\begin{bmatrix} r_j \\ r_{j-1} \end{bmatrix} = H^j \hat{b}_0 \ for \ j \geq 1 \ . \tag{2.33}$$

Formally, a rth second-order Krylov subspace with matrices A_1, B_1, and the starting vector r_0 is defined as follows:

$$colspan\{V\} = \mathcal{K}_r\{A_1, B_1; r_0\} = span\{r_0, r_1, r_2, ..., r_{r-1}\} \ . \tag{2.34}$$

In the case when Equation (2.30) is expanded around expansion point s_0, the matrices M and K in Equation (2.32) should be substituted by the matrices $\tilde{E}$ and $\tilde{K}$ defined in Equation (2.25). Finally, the SOAR algorithm from [32, 33] can be implemented to orthogonalize the vectors in Equation (2.34).

Algorithm 3 Second-Order Arnoldi Reduction (SOAR) Algorithm

Input: Matrices A_1, B_1, start vector r_0, dimension of the ROM r.

Compute $q_1 = \frac{r_0}{\|r_0\|}$.
Set $f = 0$.
for $j = 1, 2, \cdots, r$ **do** :
    Set $v = A_1 q_j + B_1 f$.
    for $i = 1, 2, \cdots, j$ **do** :
        $t_{i,j} = q_i^T v$.
        $v = v - q_i t_{i,j}$.
    end for
    Compute $t_{j+1,j} = \|v\|$
    if $t_{j+1,j} \neq 0$ **then**
        $q_{j+1} = \frac{v}{t_{j+1,j}}$
        $f = Q_j \hat{T}_j (2 : j+1, 1 : j)^{-1} e_j$
    else
        Reset $t_{j+1,j} = 1$.
        $q_{j+1} = 0$.
        $f = Q_j \hat{T}_j (2 : j+1, 1 : j)^{-1} e_j$
        Save f
        If f belongs to the subspace spanned by previously saved f, **then** stop.
    end if
end for

Output: Orthonormal vectors $q_1, q_2, \cdots, q_r$.

In Algorithm 3, matrix Q_j denotes a $N \times j$ matrix with column vectors $\{q_1, q_2, \cdots, q_j\}$. T_j denotes the $j \times j$ upper Hessenberg matrix with nonzero entries $t_{i,j}$ as defined in the algorithm. $\hat{T}_j = \begin{bmatrix} T_j \\ e_j^T t_{j+1,j} \end{bmatrix}$ is a $(j+1) \times j$ upper Hessenberg-type matrix. It should be noted here that, for a multiple-input second-order system, the SOAR Algorithm 3 can be combined with the superposition-based Arnoldi reduction method. In addition, the version of the block SOAR algorithm with m starting vectors in the input matrix $B \in \mathbb{R}^{N \times m}$ can be found in [34].

3) Proportionally damped and undamped second-order system. The third case considers a proportionally damped second-order system, where the damping matrix E can be represented as a linear combination of the mass and stiffness matrices

$$E = \alpha M + \beta K \ , \tag{2.35}$$

where α, β are the Rayleigh damping constants. When $\alpha = \beta = 0$, it is called an undamped system. As it is empirically shown in [36, 37] and mathematically proved in [35], instead of constructing the second-order Krylov subspace in Equation (2.34), one can use the matrices M, K and B to find the projection basis as follows:

$$colspan\{V\} = \mathcal{K}_r\{-K^{-1}M, -K^{-1}B\} \ . \tag{2.36}$$

Substituting matrix K by $\tilde{K}$ defined in Equation (2.25) and ignoring the damping matrix E, the projection basis is obtained when $s_0 \neq 0$ as follows:

$$colspan\{V\} = \mathcal{K}_r\{-(\hat{s}_0 M + K)^{-1}M, -(\hat{s}_0 M + K)^{-1}B\} \ , \tag{2.37}$$

where $\hat{s}_0 = s_0^2$. Additionally, the Krylov subspaces in Equations (2.36) and (2.37) can also be used for an undamped second-order system, where $E = 0$. In this way, only Arnoldi or block Arnoldi iterations are required to orthogonalize the vectors in the Krylov subspace defined in Equations (2.36) and (2.37).

It has been experimentally demonstrated in [38] and [39] that, compared to the modal superposition method [40–42], the Krylov subspace-based MOR methods lead to an impressive reduction of overall computational time and generate more accurate ROMs from second-order structural-dynamic systems. In addition, through the comparison of these three methods as mentioned above, the method of ignoring the damping matrix is strongly recommended for reducing a proportionally damped system or an undamped system. For the reduction of other second-order systems, constructing the second-order Krylov subspace and using the SOAR algorithm gives more accurate ROMs than transforming second-order systems to first-order systems.

Chapter 2 Summary

For MOR of a linear first-order system, the Krylov subspace-based MOR method has been highly developed and widely implemented in the last two decades.

- For a single-input system, the Arnoldi iteration algorithm is utilized to construct the projection matrix based on the Krylov subspace.
- For a multiple-input system, block Arnoldi or superposition-based Arnoldi reduction method can be deployed to generate the ROM.

For MOR of a linear second-order system, there are three different strategies.

- Transform a second-order system into first-order and then use the MOR approaches for reduction of the first-order system. (Not recommended, less accurate).
- Construct a second-order Krylov subspace and use SOAR method for reduction of a single-input second-order system. Either combining SOAR with the superposition-based Arnoldi reduction method or using the block SOAR method for reduction of the multiple-input second-order system. (Recommended).
- For proportionally damped or undamped second-order system, construct the second-order Krylov subspace while ignoring the damping matrix and then obtain the projection matrix via Arnoldi or block Arnoldi algorithm. (Recommended, most accurate for the case of proportional damping).

3 Parametric Model Order Reduction

As introduced in section 2, the MOR methods can be used to generate a surrogate from the original FOM in order to speed up the design and optimization process. However, when there is any change in the FOM, the ROM must be generated again. Although the time for the generation of the ROM is usually much faster than running the simulation of the FOM, it is still necessary to find a way that only needs to build the ROM once and the change of the parameter can be processed at the reduced model level. As illustrated in Figure 4, compared to performing the convectional parametric studies in an original FOM or repeatedly generated ROMs, deploying a pROM to do the parametric studies is more efficient and elegant.

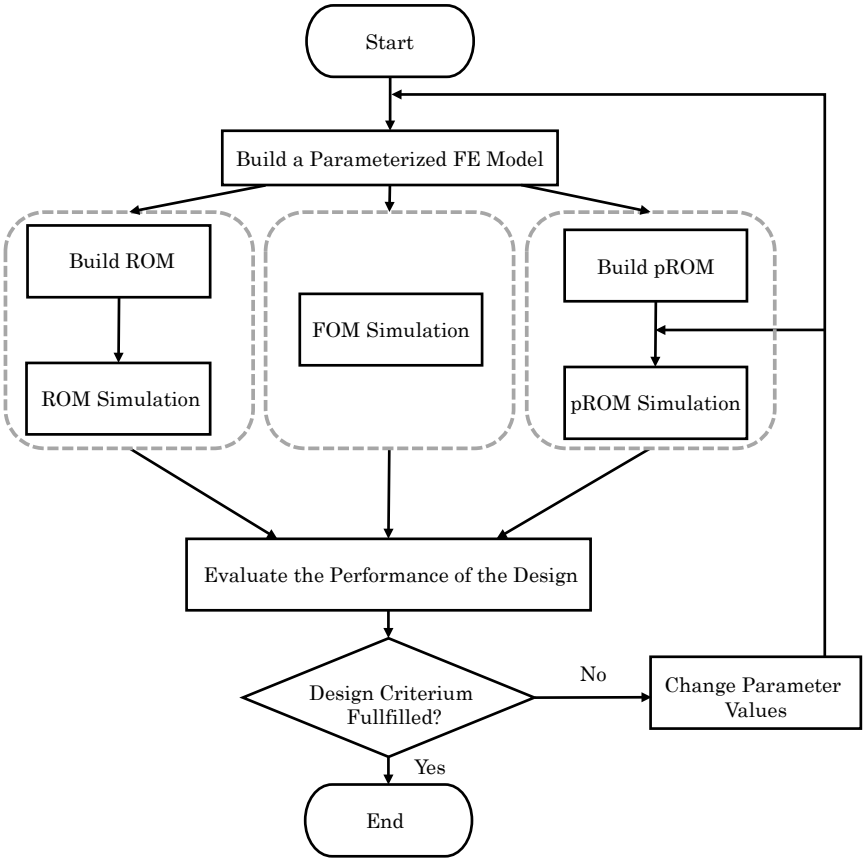


Figure 4: The use of ROMs and pROMs for parametric studies to speed up the design optimization process.

Therefore, this section will mainly present the basic framework of three pMOR methods, i.e., multivariate moment matching-, matrix interpolation (MI)-, and algebraic parameterization (AP)-based pMOR methods. The multivariate moment matching-based pMOR method is intended for reducing the model with the parameter, e.g., material parameter, which can be extracted in front of the system matrices. The MI- and AP-based pMOR methods are proposed for the geometry parameter, which is the case most often in the design optimization process.

3.1 Multivariate Moment Matching

Let's consider a parametric dynamic first-order MIMO system as follows:

$$\Sigma_N \begin{cases} E \cdot \dot{x}(t) = \underbrace{(A_0 + h \cdot A_1)}_{A(h)} \cdot x(t) + B \cdot u \\ y(t) = C \cdot x(t) \end{cases} , \tag{3.1}$$

where $E \in \mathbb{R}^{N \times N}$ is the system matrix, $B \in \mathbb{R}^{N \times m}$ and $C \in \mathbb{R}^{p \times N}$ are the input and output matrices, which are not influenced by the parameter h. $A(h) \in \mathbb{R}^{N \times N}$ is a parameter-dependent system matrix, which can be presented as a combination of two parameter-independent system matrices $A_0, A_1 \in \mathbb{R}^{N \times N}$. Parameter h is thereby extracted out of matrix A. Similar to the conventional MOR introduced in section 2, a pROM can be obtained via projecting Equation (3.1) onto a parameter-independent low-dimensional subspace $V \in R^{N \times r}$ with $r << N$, which should be constructed to capture the dynamics of the state vector $x(t) \approx V \cdot z(t)$. The pROM is then written as follows:

$$\Sigma_r \begin{cases} \underbrace{V^T E V}_{E_r} \cdot \dot{z}(t) = \overbrace{(\underbrace{V^T A_0 V}_{A_{0r}} + h \cdot \underbrace{V^T A_1 V}_{A_{1r}})}^{A_r(h)} \cdot z(t) + \underbrace{V^T B}_{B_r} \cdot u \\ y(t) = \underbrace{CV}_{C_r} \cdot z(t) \end{cases} , \tag{3.2}$$

where $E_r, A_{0r}, A_{1r} \in \mathbb{R}^{r \times r}$ are the reduced system matrices. $B_r \in \mathbb{R}^{r \times m}$ and $C_r \in \mathbb{R}^{p \times r}$ are the reduced input and output matrices, respectively. After the Laplace transformation, the transfer functions of full and reduced parametric systems in Equations (3.1) and (3.2) are written as follows:

$$G(s,h) = C[sE - (A_0 + h \cdot A_1)]^{-1} B \, , \tag{3.3}$$

$$G_r(s,h) = C_r[sE_r - (A_{0r} + h \cdot A_1 r)]^{-1} B_r \, , \tag{3.4}$$

where s is the Laplace variable. The Taylor expansion of Equations (3.3) and (3.4) around the point (s_0, h_0) reads

$$\begin{aligned} G(s,h) = {} & \textcircled{G(s_0,h_0)} + \textcircled{\frac{\partial G(s_0,h_0)}{\partial s}} \cdot (s - s_0) + \textcircled{\frac{\partial G(s_0,h_0)}{\partial h}} \cdot (h - h_0) \\ & + \textcircled{\frac{1}{2!} \frac{\partial^2 G(s_0,h_0)}{\partial s^2}} \cdot (s - s_0)^2 + \frac{\partial^2 G(s_0,h_0)}{\partial s \partial h} \cdot (s - s_0)(h - h_0) \\ & + \textcircled{\frac{1}{2!} \frac{\partial^2 G(s_0,h_0)}{\partial h^2}} \cdot (h - h_0)^2 + \dots \, , \end{aligned} \tag{3.5}$$

$$G_r(s,h) = \underbrace{G_r(s_0,h_0)} + \underbrace{\frac{\partial G_r(s_0,h_0)}{\partial s}} \cdot (s-s_0) + \underbrace{\frac{\partial G_r(s_0,h_0)}{\partial h}} \cdot (h-h_0)$$
$$+ \underbrace{\frac{1}{2!}\frac{\partial^2 G_r(s_0,h_0)}{\partial s^2}} \cdot (s-s_0)^2 + \frac{\partial^2 G_r(s_0,h_0)}{\partial s \partial h} \cdot (s-s_0)(h-h_0)$$
$$+ \underbrace{\frac{1}{2!}\frac{\partial^2 G_r(s_0,h_0)}{\partial h^2}} \cdot (h-h_0)^2 + \dots \,. \tag{3.6}$$

According to the multivariate moment matching-based pMOR method suggested in [43], the mixed moments in Equations (3.5) and (3.6) are neglected and the projection subspace V is constructed in such a way as to match only the first r circled non-mixed moments with respect to s and h separately. It is demonstrated in [44] that this indeed works well if the parameters are not physically correlated. Therefore two disjoint Krylov subspaces are computed where one parameter is kept constant, while the Krylov subspace is generated for another parameter and vice versa.

- Fix $h = h_0$ and rewrite Equation (3.3) with parameter s at the expansion point s_0.

$$\begin{aligned} H(s,h_0) &= C[sE - \underbrace{(A_0 + h_0 \cdot A_1)}_{A(h_0)}]^{-1}B \,, \\ &= C[sE - A(h_0) + s_0E - s_0E]^{-1}B \,, \\ &= C\{(s-s_0)E - [A(h_0) - s_0E]\}^{-1}B \,, \\ &= -C\{I - (s-s_0)[A(h_0) - s_0E]^{-1}E\}^{-1}[A(h_0) - s_0E]^{-1}B \,, \\ &= \sum_{i=0}^{\infty} \underbrace{-C\{[A(h_0) - s_0E]^{-1}E\}^i[A(h_0) - s_0E]^{-1}B}_{m_i^s, \quad i=0,1,\dots,r_1-1} (s-s_0)^i \,, \end{aligned} \tag{3.7}$$

  where $m_i^s,\ i = 0, 1, \cdots, r_1 - 1$ are the first r_1 moments of the transfer function $G(s, h_0)$. Then the projection matrix for the parameter s is constructed on the basis of the Krylov subspace

$$colspan\{V_s\} = \mathcal{K}_{r_1}\{(A(h_0) - s_0E)^{-1}E, (A(h_0) - s_0E)^{-1}B\} \,. \tag{3.8}$$

- Fix $s = s_0$ and rewrite Equation (3.3) with parameter h at the expansion point h_0.

$$\begin{aligned} H(s_0,h) &= C[s_0E - (A_0 + h \cdot A_1)]^{-1}B \,, \\ &= C[s_0E - (A_0 + h_0 \cdot A_1 + h \cdot A_1 - h_0 \cdot A_1)]^{-1}B \,, \\ &= C\{[s_0E - A(h_0)] - (h-h_0)A_1\}^{-1}B \,, \\ &= \sum_{j=0}^{\infty} \underbrace{C\{-[A(h_0) - s_0E]^{-1}A_1\}^j\{-[A(h_0) - s_0E]^{-1}\}B}_{m_i^h, \quad i=0,1,\dots,r_2-1} (h-h_0)^j \,. \end{aligned} \tag{3.9}$$

where $m_i^h,\ i = 0, 1, \cdots, r_2 - 1$ are the first r_2 moments of the transfer function $G(s_0, h)$. Then the projection matrix for parameter h is constructed on the basis of the Krylov subspace

$$colspan\{V_h\} = \mathcal{K}_{r_2}\{-[A(h_0) - s_0E]^{-1}A_1, -[A(h_0) - s_0E]^{-1}B\} \ . \tag{3.10}$$

The final global parameter-independent projection matrix V is constructed by merging the orthonormalized bases of V_s and V_h.

$$colspan\{V\} = colspan\{V_s, V_h\} \ , \tag{3.11}$$

where the overall dimension of the reduced system is $r_1 + r_2$. It should also be noted that the linearly dependent vectors in V_s and V_h are deflated during the merging process.

In addition, it is obvious to note that when h is fixed at h_0, the first r_1 moments of parameter s between the full and reduced systems are matching as proved in section 2.4. Similarly, it can be proved that in this parametric system, the first r_2 moments of parameter h are matching between the parametric full and reduced systems.

Proof. From the Equations (3.9), (3.10), and (3.11), it is known that for the moments m_i^h of the full system, the ith term $Q = [-(A(h_0) - s_0E)^{-1}A_1]^i\{-[A(h_0) - s_0E]\}^{-1}B \in \mathcal{K}_{r_2}$. Since the columns in V are the orthonormal basis of the Krylov subspace $\mathcal{K}_{r_2}$, then the columns in Q can be presented as a linear combination of the columns in V. Therefore, there exists a matrix P which satisfies the following equation with an arbitrary $i = q$, $q \in [0, r_2 - 1]$,

$$\{-[A(h_0) - s_0E]^{-1}A_1\}^q\{-[A(h_0) - s_0E]^{-1}\}B = V \cdot P \ , \tag{3.12}$$

$$A_1{}^q\{-[A(h_0) - s_0E]^{-1}\}B = \{-[A(h_0) - s_0E]\}^qV \cdot P \ . \tag{3.13}$$

Then multiply V^T on both sides of Equation (3.13) and insert $VV^T = I$ in between.

$$\underbrace{V^TA_1{}^qV}_{A_{1r}{}^q}\underbrace{V^T\{-[A(h_0) - s_0E]^{-1}\}V}_{-[A_r(h_0)-s_0E_r]^{-1}}\underbrace{V^TB}_{B_r} = \underbrace{V^T\{-[A(h_0) - s_0E]\}^qV}_{\{-[A_r(h_0)-s_0E_r]\}^q} \cdot P \ ,$$

$$\{-[A_r(h_0) - s_0E_r]\}^{-q}A_{1r}{}^q\{-[A_r(h_0) - s_0E_r]^{-1}\}B_r = P \ . \tag{3.14}$$

Taking Equations (3.12) and (3.14) into the qth moment of m_i^h,

$$\begin{aligned} m_q^h &= C \cdot V \cdot P \\ &= C_r\{-[A_r(h_0) - s_0E_r]\}^{-q}A_{1r}{}^q\{-[A_r(h_0) - s_0E_r]^{-1}\}B_r \\ &= m_q^{h(r)} \ . \end{aligned} \tag{3.15}$$

It is proved that the first r_2 moments from full and reduced systems are matching. □

In addition, the parametric system in Equation (3.1) can be generalized to

$$\Sigma_N \begin{cases} E \cdot \dot{x}(t) = \underbrace{(A_0 + \sum_i^k h_i \cdot A_i)}_{A(h_i)} \cdot x(t) + B \cdot u \\ y(t) = C \cdot x(t) \end{cases} . \tag{3.16}$$

The projection matrices for each parameter h_i are constructed as follows:

$$colspan\{V_{h_i}\} = \mathcal{K}_{r_2}\{-[A(h_{i0}) - s_0 E]^{-1} A_i, -[A(h_{i0}) - s_0 E]^{-1} B\} . \tag{3.17}$$

Thereby, the generalized global projection matrix reads

$$colspan\{V\} = colspan\{V_s, V_{h_i}\} . \tag{3.18}$$

3.2 Matrix Interpolation

Let's consider a first-order model with a geometrical parameter l. The system matrices depend nonlinearly on the geometrical parameter, which cannot usually be extracted from the matrices. The spatial discretization leads to the following FE model:

$$\Sigma_N \begin{cases} E(l)\ \dot{x}(t) = A(l)\ x(t) + B(l)\ u \\ y(t) = C(l)\ x(t) \end{cases} , \tag{3.19}$$

where $E(l), A(l) \in \mathbb{R}^{N \times N}$, $B(l) \in \mathbb{R}^{N \times m}$, and $C(l) \in \mathbb{R}^{p \times N}$ are the geometrical parameter-dependent matrices. To generate the pROM of (3.19), the MI-based pMOR method should be applied. As suggested in [45], the general framework for pMOR by matrix interpolation contains the following steps:

Step 1: k values are chosen for the geometrical parameter $l = \{l_1, l_2, \cdots, l_k\}$ and the corresponding models $\{\Sigma_1, \Sigma_2, \cdots, \Sigma_k\}$ are reduced. These local ROMs are generated via MOR methods, e.g., Block Arnoldi algorithm introduced in section 2.3.1, and stored in a database $DB = \{\Sigma_{r,1}, \Sigma_{r,2}, \cdots, \Sigma_{r,k}\}$.

$$\Sigma_{r,i} \begin{cases} \underbrace{V_i^T E_i V_i}_{E_{r,i}}\ \dot{z}_i(t) = \underbrace{V_i^T A_i V_i}_{A_{r,i}}\ z_i(t) + \underbrace{V_i^T B_i}_{B_{r,i}}\ u \\ y = \underbrace{C_i V_i}_{C_{r,i}}\ z_i(t) \end{cases} , \tag{3.20}$$

where $E_{r,i}, A_{r,i} \in \mathbb{R}^{r \times r}$, $B_{r,i} \in \mathbb{R}^{r \times m}$ and $C_{r,i} \in \mathbb{R}^{p \times r}$ are the reduced system matrices at discrete parameter values $l_i, i \in [1, k]$ and $z_i \in \mathbb{R}^r$ are the corresponding reduced order state vectors. $V_i \in \mathbb{R}^{N \times r}$ with $r \ll N$ are the low-dimensional subspaces.

Step 2: All the local ROMs are transformed into another set of generalized coordinates $z_i^* = \mathbf{T}_i \; z_i$ as follows:

$$\Sigma_{r,i}^* \begin{cases} \underbrace{\mathbf{M}_i E_{r,i} \mathbf{T}_i^{-1}}_{E_{r,i}^*} \; \dot{z}_i^*(t) = \underbrace{\mathbf{M}_i A_{r,i} \mathbf{T}_i^{-1}}_{A_{r,i}^*} \; z_i^*(t) + \underbrace{\mathbf{M}_i B_{r,i}}_{B_{r,i}^*} \; u \\ y = \underbrace{C_{r,i} \mathbf{T}_i^{-1}}_{C_{r,i}^*} \; z_i^*(t) \end{cases} , \tag{3.21}$$

where $\mathbf{T}_i, \mathbf{M}_i \in \mathbb{R}^{r \times r}$ are the regular matrices defined as

$$\mathbf{T}_i = \mathbf{R}^T \; V_i \; , \tag{3.22}$$

$$\mathbf{M}_i = (V_i^T \; \mathbf{R})^{-1} \; . \tag{3.23}$$

$\mathbf{R}$ is chosen to comprise the most important dynamics of all local models, via performing the singular value decomposition (SVD) on a matrix pool, which contains all the local projection matrices.

$$\mathbf{U} \Sigma \mathbf{N}^T = \mathrm{SVD}(\underbrace{[V_1, V_2, \cdots, V_k]}_{V_{all}}) \; . \tag{3.24}$$

$\mathbf{R}$ contains the first r columns of $\mathbf{U}$,

$$\mathbf{R} = \mathbf{U}(:, 1:r) \; . \tag{3.25}$$

Step 3: Global parametric ROM is obtained through the weighted interpolation based on the local system matrices.

$$\Sigma_r^* \begin{cases} E_r^* \; \dot{z}^*(t) = A_r^* \; z^*(t) + B_r^* \; u \\ y(t) = C_r^* \; z^*(t) \end{cases} , \tag{3.26}$$

where

$$E_r^* = \sum_{i=1}^{k} w_i(l) \; E_{r,i}^* \; , \qquad A_r^* = \sum_{i=1}^{k} w_i(l) \; A_{r,i}^* \; ,$$
$$B_r^* = \sum_{i=1}^{k} w_i(l) \; B_{r,i}^* \; , \qquad C_r^* = \sum_{i=1}^{k} w_i(l) \; C_{r,i}^* \; ,$$

and $\sum_{i=1}^{k} w_i(l) = 1$ are the weights. For example, the Lagrange interpolation strategy [46] can be utilized to calculate the pROM with the geometrical parameter l as follows:

$$\Sigma(l)_r^* = \sum_{i=1}^{k} \frac{w(l)}{(l - l_i) w'(l_i)} \cdot \Sigma_{r,i}^* \; , \tag{3.27}$$

where the weights $w(l)$ and $w'(l_i)$ are calculated as

$$\begin{cases} w(l) = (l - l_1)(l - l_2) \cdots (l - l_k) \\ w'(l_i) = (l_i - l_1) \cdots (l_i - l_{i-1})(l_i - l_{i+1}) \cdots (l_i - l_k) \end{cases} . \tag{3.28}$$

When $k = 2$ there are only two local ROM in the database DB, the Lagrange interpolation is equivalent to linear interpolation and Equation (3.27) is rewritten as

$$\Sigma(l)_r^* = \frac{l - l_2}{l_1 - l_2} \cdot \Sigma_{r,1}^* + \frac{l - l_1}{l_2 - l_1} \cdot \Sigma_{r,2}^* . \tag{3.29}$$

It should be noted here that an important prerequisite for the 'classic' MI-based pMOR method introduced above is that all local FE models in Equation (3.20) should have the same size, which implies an equal number of rows in each local projection matrix V_i. Otherwise, the SVD in Equation (3.24) is in-executable. However, when e.g. using an industrial FE simulator and having a complex geometry change, it is difficult and also not advisable to keep the size of the model constant when varying geometrical parameters. Hence, $V_i \in \mathbb{R}^{N_i \times r}$ for $i \in [1, k]$ are of different numbers of rows. To solve this problem, the authors in [47,48] suggested constructing a new matrix pool by padding zero elements in appropriate rows of smaller local projection matrices V_i as follows:

$$\widetilde{V}_{all} = \begin{bmatrix} \widetilde{V}_{11} & \widetilde{V}_{21} & \cdots & \widetilde{V}_{k1} \\ 0 & \widetilde{V}_{22} & \cdots & \widetilde{V}_{k2} \\ 0 & 0 & \cdots & \widetilde{V}_{k3} \\ \vdots & \vdots & \cdots & \vdots \\ 0 & 0 & \cdots & \vdots \\ 0 & 0 & \cdots & \widetilde{V}_{kk} \end{bmatrix} \in \mathbb{R}^{N_k \times (k \cdot r)} , \tag{3.30}$$

where $V_1 = \widetilde{V}_{11} \in \mathbb{R}^{N_1 \times r}$, $V_2 = \begin{bmatrix} \widetilde{V}_{21} \\ \widetilde{V}_{22} \end{bmatrix} \in \mathbb{R}^{N_2 \times r}$ and so on are the local projection matrices of their original dimension. In Equation (3.30) it is assumed that $N_1 < N_2 < \cdots < N_k$. Note that, padding the zero elements will not influence the existing directions of the vectors. Then the 'classic' MI-based pMOR method can be performed.

However, it is found in this work that varying the values of the geometrical parameter in the models built in industrial software, e.g., ANSYS Mechanical, will not only change the mesh topology but also the node ordering, which influences the accuracy of the interpolated pROM. Therefore, this work suggests a modified MI method to minimize the error caused by the node ordering problem. As displayed schematically in Figure 5, only two nearby local ROMs $\{\Sigma_{r,L}, \Sigma_{r,R}\}$ with $l = \{l_L, l_R\}$ are used for interpolation.

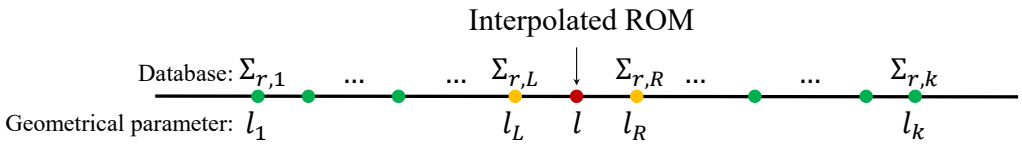


Figure 5: Strategy of using only the two nearby local ROMs for matrix interpolation.

Thereby, the matrix pool $\widetilde{V}_{all}$ in Equation (3.24) is simplified to

$$\widetilde{V}_{all} = \begin{bmatrix} V_L & V_R \end{bmatrix} , \tag{3.31}$$

where V_L is chosen from the local model $\Sigma_{r,L}$ left to the ROM to be interpolated and V_R is chosen from $\Sigma_{r,R}$ to the right. Once the geometrical parameter l is modified, V_L and V_R should also be re-selected from the database DB. The transformation matrices $\mathbf{T}$ and $\mathbf{M}$ are defined as

$$\mathbf{T} = \widetilde{\mathbf{R}}^T \cdot V_L , \tag{3.32}$$

$$\mathbf{M} = (V_L^T \cdot \widetilde{\mathbf{R}})^{-1} , \tag{3.33}$$

where V_L can also be replaced by V_R. $\widetilde{\mathbf{R}}$ is chosen to comprise the most important dynamics of both nearby local ROMs via performing the SVD on the matrix pool $\widetilde{V}_{all}$. These two local ROMs $\{\Sigma_{r,L}, \Sigma_{r,R}\}$ are then transformed via the obtained transformation matrices $\mathbf{T}$ and $\mathbf{M}$. Finally, the parametric ROM is obtained by using the linear interpolation method presented in Equation (3.29). The approach proposed above is demonstrated in section 7.3. It is less accurate than the MI-based pMOR method from [45], but it is necessary to conquer the problem of node ordering in ANSYS Mechanical.

Additionally, the MI-based pMOR method can also be applied to the second-order superposed reduced systems. The framework contains the following steps:

Step 1. k local ROMs with parameter $l_i = \{l_1, l_2, l_3, \cdots, l_k\}$ are constructed by using the superposition SOAR algorithm as follows:

$$\Sigma_{i,r} \begin{cases} \underbrace{\begin{bmatrix} V_{i,1}^T M_i V_{i,1} & 0 \\ 0 & V_{i,2}^T M_i V_{i,2} \end{bmatrix}}_{M_{i,r}} \underbrace{\begin{bmatrix} \ddot{z}_{i,1} \\ \ddot{z}_{i,2} \end{bmatrix}}_{\ddot{z}_i} + \underbrace{\begin{bmatrix} V_{i,1}^T E_i V_{i,1} & 0 \\ 0 & V_{i,2}^T E_i V_{i,2} \end{bmatrix}}_{E_{i,r}} \underbrace{\begin{bmatrix} \dot{z}_{i,1} \\ \dot{z}_{i,2} \end{bmatrix}}_{\dot{z}_i} \\ + \underbrace{\begin{bmatrix} V_{i,1}^T K_i V_{i,1} & 0 \\ 0 & V_{i,2}^T K_i V_{i,2} \end{bmatrix}}_{K_{i,r}} \underbrace{\begin{bmatrix} z_{i,1} \\ z_{i,2} \end{bmatrix}}_{z_i} = \underbrace{\begin{bmatrix} V_{i,1}^T b_{i,1} & 0 \\ 0 & V_{i,2}^T b_{i,2} \end{bmatrix}}_{B_{i,r}} \underbrace{\begin{bmatrix} u_1 \\ u_2 \end{bmatrix}}_{u} \\ y = \underbrace{\begin{bmatrix} C_i V_{i,1} & C_i V_{i,2} \end{bmatrix}}_{C_{i,r}} \underbrace{\begin{bmatrix} z_{i,1} \\ z_{i,2} \end{bmatrix}}_{z_i} \end{cases} , \tag{3.34}$$

where $V_{i,1}$ and $V_{i,2}$ are the local projection matrices generated by orthogonalizing the second-order Krylov subspaces with start vectors $b_{i,1}$ and $b_{i,2}$, respectively. $x_{i,1} \approx V_{i,1} z_{i,1}$, $x_{i,2} \approx V_{i,2} z_{i,2}$.

Step 2: Local transformation matrices are constructed for each subsystem

$$\begin{aligned} \mathbf{T}_{i,1} &= \mathbf{R}_1^T V_{i,1};, \\ \mathbf{M}_{i,1} &= (V_{i,1}^T \mathbf{R}_1)^{-1} , \end{aligned} \tag{3.35}$$

$$\begin{aligned} \mathbf{T}_{i,2} &= \mathbf{R}_2^T V_{i,2} , \\ \mathbf{M}_{i,2} &= (V_{i,2}^T \mathbf{R}_2)^{-1} , \end{aligned} \tag{3.36}$$

where $\mathbf{T}_{i,1}, \mathbf{M}_{i,1}, \mathbf{T}_{i,2}, \mathbf{M}_{i,2} \in \mathbb{R}^{r \times r}$. $\mathbf{R}_1$ and $\mathbf{R}_2$ are obtained via performing the singular value decomposition (SVD) on a matrix pool, which contains all the local projection matrices of each subsystem

$$\begin{aligned} \mathbf{U}_1 \Sigma_1 \mathbf{N}_1^T &= \mathrm{SVD}([V_{1,1}, V_{2,1}, ..., V_{k,1}]) , \\ \mathbf{U}_2 \Sigma_2 \mathbf{N}_2^T &= \mathrm{SVD}([V_{1,2}, V_{2,2}, ..., V_{k,2}]) . \end{aligned} \tag{3.37}$$

$\mathbf{R}_1$ and $\mathbf{R}_2$ contain the first r columns of $\mathbf{U}_1$ and $\mathbf{U}_2$.

$$\begin{aligned} \mathbf{R}_1 &= \mathbf{U}_1(:, 1:r) , \\ \mathbf{R}_2 &= \mathbf{U}_2(:, 1:r) . \end{aligned} \tag{3.38}$$

Then, all local ROMs are transformed as follows:

$$\Sigma_{i,r}^* \begin{cases} \underbrace{\begin{bmatrix} \mathbf{M}_{i,1} M_{i,r,1} \mathbf{T}_{i,1}^{-1} & 0 \\ 0 & \mathbf{M}_{i,2} M_{i,r,2} \mathbf{T}_{i,2}^{-1} \end{bmatrix}}_{M_{i,r}^*} \underbrace{\begin{bmatrix} \ddot{z}_{i,1}^* \\ \ddot{z}_{i,2}^* \end{bmatrix}}_{\ddot{z}_i^*} + \underbrace{\begin{bmatrix} \mathbf{M}_{i,1} E_{i,r,1} \mathbf{T}_{i,1}^{-1} & 0 \\ 0 & \mathbf{M}_{i,2} E_{i,r,2} \mathbf{T}_{i,2}^{-1} \end{bmatrix}}_{E_{i,r}^*} \underbrace{\begin{bmatrix} \dot{z}_{i,1}^* \\ \dot{z}_{i,2}^* \end{bmatrix}}_{\dot{z}_i^*} \\ + \underbrace{\begin{bmatrix} \mathbf{M}_{i,1} K_{i,r,1} \mathbf{T}_{i,1}^{-1} & 0 \\ 0 & \mathbf{M}_{i,2} K_{i,r,2} \mathbf{T}_{i,2}^{-1} \end{bmatrix}}_{K_{i,r}^*} \underbrace{\begin{bmatrix} z_{i,1}^* \\ z_{i,2}^* \end{bmatrix}}_{z_i^*} = \underbrace{\begin{bmatrix} \mathbf{M}_{i,1} b_{i,r,1} & 0 \\ 0 & \mathbf{M}_{i,2} b_{i,r,2} \end{bmatrix}}_{B_{i,r}^*} \underbrace{\begin{bmatrix} u_1 \\ u_2 \end{bmatrix}}_{u} \\ y = \underbrace{\begin{bmatrix} C_{i,r,1} \mathbf{T}_{i,1}^{-1} & C_{i,r,2} \mathbf{T}_{i,2}^{-1} \end{bmatrix}}_{C_{i,r}^*} \underbrace{\begin{bmatrix} z_{i,1}^* \\ z_{i,2}^* \end{bmatrix}}_{z_i^*} \end{cases} . \tag{3.39}$$

Step 3: Parametric ROM is finally obtained through the weighted interpolation based on the transformed local ROMs

$$\Sigma_r^* \begin{cases} M_r^* \ddot{z}^* + E_r^* \dot{z}^* + K_r^* z^* = B_r^* u \\ y = C_r^* z^* \end{cases} , \tag{3.40}$$

where

$$M_r^* = \sum_{i=1}^{k} w_i(l) M_{i,r}^* \,, \quad E_r^* = \sum_{i=1}^{k} w_i(l) E_{i,r}^* \,, \quad K_r^* = \sum_{i=1}^{k} w_i(l) K_{i,r}^* \,,$$

$$B_r^* = \sum_{i=1}^{k} w_i(l) B_{i,r}^* \,, \qquad C_r^* = \sum_{i=1}^{k} w_i(l) C_{i,r}^* \,,$$

with $\sum_{i=1}^{k} w_i(l) = 1$ being the weights, which can be calculated via Lagrange interpolation in Equation (3.28).

3.3 Algebraic Parameterization

Another pMOR method applicable to geometrically parameterized FE models is suggested in [49, 50]. The authors introduced an approach for the algebraic parameterization of FE models with varying geometrical parameters. In this section, this method will be introduced and its limitations indicated. In addition, an improved workflow of their method will be suggested in this work.

Similarly to MI-based pMOR method, the AP-based pMOR method is applicable only if the mesh topology, i.e., the matrix structure, remains unchanged. Varying the geometrical parameter could be achieved by scaling the size of the elements as shown in Figure 6.

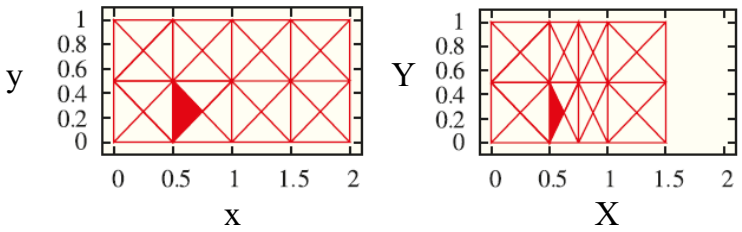


Figure 6: Varying the geometrical parameter while preserving the mesh topology via scaling the elements. This figure is adapted from [49].

The basic framework of the AP-based pMOR method contains the following steps:

Step 1: Consider the static form of the geometrical parameter-dependent first-order system and extend it to a parametric form thus:

$$\Sigma_N \begin{cases} 0 = \underbrace{(\frac{1}{\alpha} \cdot A_{\frac{1}{\alpha}} + A_1 + \alpha \cdot A_\alpha)}_{A(\frac{1}{\alpha},\alpha)} \cdot x + B \cdot u \\ y = C \cdot x \end{cases} , \tag{3.41}$$

where α is the scaling factor for the geometrical parameter l, e.g., the height of the magnets. The parameter-independent system matrices $A_{\frac{1}{\alpha}}$, A_1, and $A_\alpha \in \mathbb{R}^{N \times N}$ can be

computed through the following numerical scheme:

$$\begin{bmatrix} A_{\alpha_1,i,j} \\ A_{\alpha_2,i,j} \\ A_{\alpha_3,i,j} \end{bmatrix} = \begin{bmatrix} \frac{1}{\alpha_1} & 1 & \alpha_1 \\ \frac{1}{\alpha_2} & 1 & \alpha_2 \\ \frac{1}{\alpha_3} & 1 & \alpha_3 \end{bmatrix} \begin{bmatrix} A_{\frac{1}{\alpha},i,j} \\ A_{1,i,j} \\ A_{\alpha,i,j} \end{bmatrix} , \tag{3.42}$$

where $A_{\alpha_1}, A_{\alpha_2}, A_{\alpha_3}$ are the system matrices snapshot with $l = \{l_1, l_2, l_3\}$. The scaling factors are then identified as follows $\alpha_1 = \frac{l_1}{l_1}, \alpha_2 = \frac{l_2}{l_1}, \alpha_3 = \frac{l_3}{l_1}$.

Step 2: Equation (3.42) is constructed and solved for each matrix entry, that is to say N^2 times. N is the dimension of the system matrix. The solution of $A_{\frac{1}{\alpha},i,j}$, $A_{1,i,j}$, and $A_{\alpha,i,j}$ assembles the desired parameter-independent matrices.

Step 3: On the basis of Equation (3.41), the multivariate moment-matching-based pMOR method introduced in section 2.4 can be applied to generate the pROM thus:

$$\Sigma_r \begin{cases} 0 = \underbrace{V^T A(\frac{1}{\alpha}, \alpha) V}_{A_r(\frac{1}{\alpha},\alpha)} \cdot z + \underbrace{V^T B}_{B_r} \cdot u \\ y = \underbrace{CV}_{C_r} \cdot z \end{cases} , \tag{3.43}$$

where $A_r \in \mathbb{R}^{r \times r}$, $B_r \in \mathbb{R}^{r \times m}$, and $C_r \in \mathbb{R}^{p \times r}$ are the reduced system matrices. The global projection matrix $V \in \mathbb{R}^{N \times r}$ is obtained by merging two local projection matrices of parameter $\frac{1}{\alpha}$ and α. For the parametric static model, the local projection matrices can be constructed by orthogonalizing the Krylov subspaces of each parameter

$$colspan\{V_{\frac{1}{\alpha}}\} = \mathcal{K}_{r_1}\{-A(\frac{1}{\alpha_0}, \alpha_0)^{-1} A_{\frac{1}{\alpha}}, -A(\frac{1}{\alpha_0}, \alpha_0)^{-1} B\} , \tag{3.44}$$

$$colspan\{V_\alpha\} = \mathcal{K}_{r_2}\{-A(\frac{1}{\alpha_0}, \alpha_0)^{-1} A_\alpha, -A(\frac{1}{\alpha_0}, \alpha_0)^{-1} B\} , \tag{3.45}$$

$$colspan\{V\} = colspan\{V_{\frac{1}{\alpha}}, V_\alpha\} , \tag{3.46}$$

where $\frac{1}{\alpha_0}$ and α_0 are the fixed expansion points with respect to each parameter. Usually, $\alpha_0 = \alpha_1$ can be selected. Then matrix $A(\frac{1}{\alpha_1}, \alpha_1)$ is the snapshotted matrix at $l = l_1$.

It is worth noting that in Step 2, the authors in [49,50] didn't mention any efficient way to solve Equation (3.42) N^2 times when the dimension N of the FE model is significantly large. In this work, a new workflow is introduced to calculate these parameter-independent system matrices through rewriting the Equation (3.42) in a matrix form as follows:

$$\begin{bmatrix} A_{\alpha_1} \\ A_{\alpha_2} \\ A_{\alpha_3} \end{bmatrix} = \begin{bmatrix} \frac{1}{\alpha_1} & 1 & \alpha_1 \\ \frac{1}{\alpha_2} & 1 & \alpha_2 \\ \frac{1}{\alpha_3} & 1 & \alpha_3 \end{bmatrix} \begin{bmatrix} A_{\frac{1}{\alpha}} \\ A_1 \\ A_\alpha \end{bmatrix} . \tag{3.47}$$

Therefore, Equation (3.47) can be solved symbolically and the parameter-independent matrices are then expressed analytically as weighted sums of the snapshot matrices

$$\begin{cases} K_{\frac{1}{\alpha}} = s_{11}K_{\alpha_1} + s_{12}K_{\alpha_2} + s_{13}K_{\alpha_3} \\ K_1 = s_{21}K_{\alpha_1} + s_{22}K_{\alpha_2} + s_{23}K_{\alpha_3} \\ K_\alpha = s_{31}K_{\alpha_1} + s_{32}K_{\alpha_2} + s_{33}K_{\alpha_3} \end{cases}, \tag{3.48}$$

where $s_{i,j}(\alpha_1, \alpha_2, \alpha_3)$, $i, j = 1, 2, 3$, are the coefficients calculated based on the scaling factors. In this way, Equation (3.47) needs to be solved only once and the computational cost for the matrix-scalar multiplication and matrix summation is low due to the fact that only $q \ll N^2$ nonzero elements from the sparse system matrices are calculated. The comparison of the computational complexity of the original and improved methods is presented in Table 1.

Table 1: Comparison of the computational complexity of the methods from [49] and this work.

Method	**Steps**	**Comput.Complex.**
Method from [49]	Solve Equation (3.42) N^2 times	$\mathcal{O}$ (27×N^2)
This work	1. Solve Equation (3.47) once 2. Calculate the coefficients $s_{i,j}$ 3. Compute matrices in Equation (3.48)	$\mathcal{O}$ (15×q)

Chapter 3 Summary

The multivariate moment matching-based pMOR method is designed for the reduction of the parametric system, where the parameter-dependent system matrices can be transformed into the state of affine parameter dependence.

MI-based pMOR method is deployed to reduce the parametric system with geometrical parameters. The pROM is obtained via interpolating the local ROMs.

- The mesh topology of the local models should be retained. Otherwise, a modified method is suggested to minimize the error caused by the mesh.

AP-based pMOR method extracts of the geometrical parameter output of the diffusion matrix and then enables the implementation of the multivariate moment matching-based pMOR method.

- The mesh topology of the model has to be kept the same while changing the value of the geometrical parameter.
- In this work, the algebraic parameterization process is modified to reduce the generation time of the pROM.

4 Model Reduction Inside ANSYS

This chapter introduces the software 'Model Reduction inside ANSYS' [26] (previously named 'MOR for ANSYS' or 'mor4ansys'), which is used principally in this work to perform the Krylov subspace-based model order reduction (MOR) methods (see chapter 2) on the finite element (FE) models from ANSYS Mechanical [17,18]. The software enables users from either academia or industry to read the full-scale system matrices of the original system from ANSYS Mechanical and write out the reduced system matrices either in Market Matrix format or in other formats that can be imported directly into the system-level simulator, e.g., ANSYS Twin Builder [51].

Additionally, in practical engineering applications, the models are usually parameterized for design optimization, which is also the main topic focused upon in this work. As mentioned in chapter 3, although the computational cost of generating a reduced-order model (ROM) via MOR is faster than running the full-scale model simulation, one still has to repeat the MOR process when the model is changed in the optimization process. A more elegant solution is to use the parametric model order reduction (pMOR) method to construct a parametric reduced-order model (pROM). However, 'Model Reduction inside ANSYS' only provides the functions of the multivariate moment matching-based pMOR method introduced in section 3.1 to build a pROM of the model, in which the parameters can be preserved out of the system matrices.

'Model Reduction inside ANSYS' is written in C++ and Figure 7 shows its block scheme. The software integrates MOR and pMOR algorithms in ANSYS Mechanical. As a result, it enables its users to read the full system matrices from the ANSYS model and write out the reduced model files automatically.

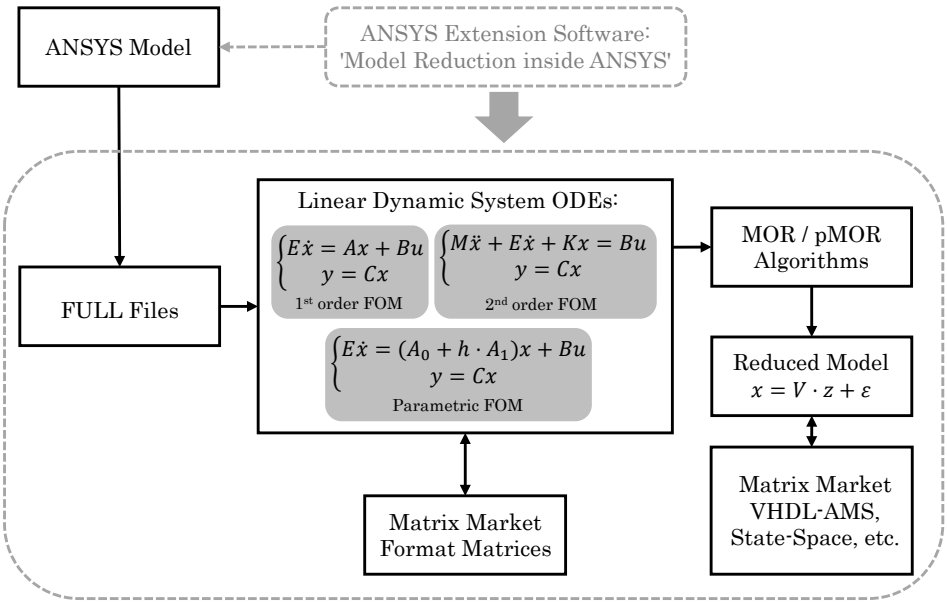


Figure 7: 'Model Reduction inside ANSYS' block scheme.

Obtaining system matrices from ANSYS Mechanical is a crucial function in 'Model Reduction inside ANSYS'. It offers more flexibility which means that the full system matrices obtained in Matrix Market can also be implemented in other model reduction software, e.g., sssMOR [52] and pyMOR [53]. In this work, the functions of pMOR methods introduced in sections 3.2 and 3.3 are further developed in Python and MATLAB based on the full system matrices obtained via 'Model Reduction inside ANSYS'. These functions are applied and validated in the case studies presented in chapter 5 of this thesis. In this chapter, the framework of MOR and pMOR that 'Model Reduction inside ANSYS' uses for automatic ROM and pROM generation will be presented.

4.1 Model Order Reduction

'Model reduction inside ANSYS' starts by obtaining system matrices from ANSYS in harmonic analysis. It uses the ANSYS parametric design language (APDL) command '`WRFULL`' to write the FULL files[1], maintaining all the original system matrices, the load vector, the ordering, the Dirichlet and equation constraints. More importantly, the APDL command '`WRFULL`' allows users to write the FULL files without a real solution phase.

Note that harmonic analysis as such is not important. It is just a way of generating the full system matrices required for MOR. Let's first consider a single-input linear system of equations in first-order to solve in harmonic response analysis as follows:

$$(i\omega E + A)x(\omega) = F \ . \tag{4.1}$$

In this case, ANSYS Mechanical APDL writes out a single complex matrix $(i\omega E + A)$, where matrix A is the real part and matrix ωE is the complex part. It is possible to obtain the matrix E while setting the frequency ω equals to one. Thus, the following APDL code is employed to prepare the FULL files for 'Model Reduction inside ANSYS' to write the full system matrices.

```
/assign,full,f_mat,full          ! The FULL file is named as 'f_mat.full'
/solu
allse
antype,harmic                    ! Solution type harmonic analysis
eqslv,sparse
harfrq,1/2/3.141592653589793     ! Frequency w is set as 1 Hz
nsubst,1,1,1
wrfull,1
solve
```

Then the command line tool in 'Model Reduction inside ANSYS' is used to generate the reduced model. A '`runmor.cmd`' file is created with the command line as follows:

[1]Remove setting 'Distributed Solution' in ANSYS Mechanical before writing the FULL files.

```
mor_for_ansys f_mat.full -b -N 30 -C output.txt -o mor > mor.out
```

`mor_for_ansys`: invoke the file '`mor_for_ansys.exe`'; its path is saved in the environment variables.

`-b`: use the block Arnoldi algorithm.

`-N`: specify the dimension of the ROM.

`-C`: read the information of the output nodes in '`output.txt`'.

`-o`: specify the name of the ROM files as '`mor.*`'. It is also the default name.

`> mor.out`: save the information of reduction process in the file '`mor.out`'.

When using option '`-b`', '`-N`' specifies the total dimension of the ROM. Number 30 is usually used because it fits well for many different engineering systems. Otherwise, one can use the error indicator proposed in [54] to choose an optimal order of the ROM. In '`output.txt`', the outputs are saved in the format 'Name DOF Node_number'.

For reduction of a single-input second-order system, the following linear system of equations to solve in harmonic response analysis is considered:

$$(-\omega^2 M + i\omega E + K)x(\omega) = F\ . \tag{4.2}$$

In this case, the APDL commands below are required to write two FULL files. As APDL command '`WRFULL`' avoids a real solution of the model, the frequency used in the harmonic analysis is merely a technique to obtain the correct system matrices.

```
/assign,full,file0,full     ! The first FULL file is named as 'file0.full'
/solu
allse
antype,harmic
eqslv,sparse
harfrq,0                    ! Frequency w1 is set as 0 Hz
nsubst,1,1,1
wrfull,1
solve
/assign,full,file1e10,full ! The second file is named as 'file1e10.full'
/solu
allse
antype,harmic
eqslv,sparse
harfrq,1e10/2/3.141592653589793 ! Frequency w2 is set as 1e10 Hz
nsubst,1,1,1
wrfull,1
solve
```

The ANSYS Mechanical APDL writes the complex matrix $m_\omega = (-\omega^2 M + i\omega E + K)$, where $m_\omega.re = (-\omega^2 M + K)$ is the real part and $m_\omega.im = \omega E$ is the complex part. The first FULL file '`file0.full`' is obtained with the frequency $\omega_1 = 0$ in the harmonic analysis. Therefore, it writes a real matrix as

$$m_{\omega_1}.re = K \ . \tag{4.3}$$

The second FULL file '`file1e10.full`' is obtained with a high frequency $\omega_2 = 1e10$. Thereby, it writes the complex matrix $m_{\omega_2} = (-{\omega_2}^2 M + i\omega_2 E + K)$ and the matrices E is calculated from the complex part as follows:

$$\frac{m_{\omega_2}.im}{\omega_2} = E \ . \tag{4.4}$$

As matrix K is obtained from Equation (4.3), then the real part of the complex matrix $m_{\omega_2}.re = (-{\omega_2}^2 M + K)$ can be used to calculate matrix M as follows:

$$\frac{K - m_{\omega_2}.re}{{\omega_2}^2} = M \ . \tag{4.5}$$

To generate the ROM, the '`runmor.cmd`' file is created with the command line.

```
mor_for_ansys file0.full file1e10.full -2 -N 30 -C output.txt > mor.out
```

`-2`: use second-order Arnoldi reduction (SOAR) algorithm for second-order system.

When using option '`-2`', '`-N`' specifies the dimension per input in case of multiple inputs. Additionally, as shown in Figure 8, 'Model Reduction inside ANSYS' also employs the other two MOR methods introduced in section 2.5. The default algorithm is used (without any command) for proportionally damped systems, where the damping matrix can be ignored during the MOR process. Option '`-1`' is deployed to transform the second-order system into first-order system first before the MOR process.

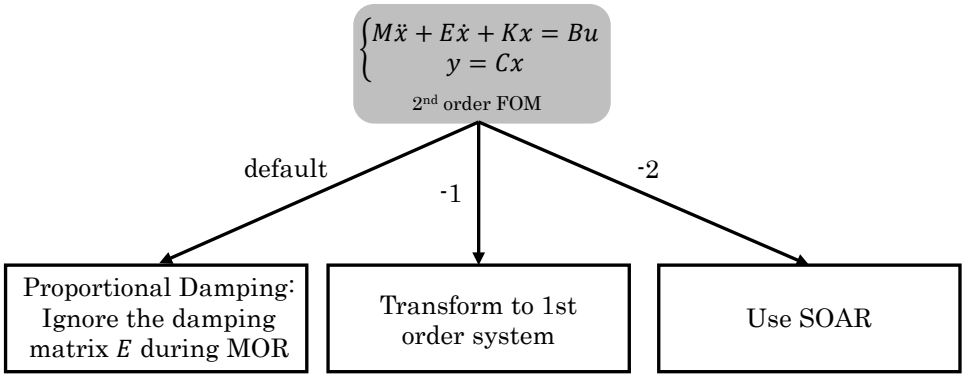


Figure 8: Model reduction scheme for second-order systems.

For reduction of the multiple-input systems, multiple FULL files should be generated as ANSYS Mechanical only writes one load in one FULL file. Therefore, for both multiple-input first- and second-order systems, the APDL scripts mentioned above should be used repeatedly for each single input which means that one should activate only one input in the model at a time and suppress all the other inputs in ANSYS Mechanical. The corresponding FULL file names '`f_mat.full`' and '`file0.full`' should be renamed for each input. The following '`runmor.cmd`' file is used to generate the second-order ROM:

```
mor_for_ansys file0.full file1e10.full -m file1.full -2 -N 15 -C ···
```

`-m`: read an additional input from '`file1.full`'.

There are two inputs in this case and '`-N 15`' is specified as the dimension of each input and thereby the dimension of the ROM is 30. In the case of many inputs, the time to write the FULL files might be much more than the time for MOR. To speed up the time of writing FULL files, one could select only elements and nodes related to the input before writing the FULL file down and then using option '`-mf`' to read these FULL files. For the further options in 'Model Reduction inside ANSYS', please see Appendix D.

The system matrix files generated via 'Model Reduction inside ANSYS' are presented in Figure 9. '`full_system`.∗' files are the original full-scale system matrices and '`mor`.∗' are the reduced system matrices, which are all saved in 'Matrix Market' format. '∗.`B.names`' and '∗.`C.names`' files contain the names of the inputs and outputs. These reduced system matrices can also be transformed into other formats for other software, e.g., state space model in MATLAB and the VHDL-AMS model in ANSYS Twin Builder.

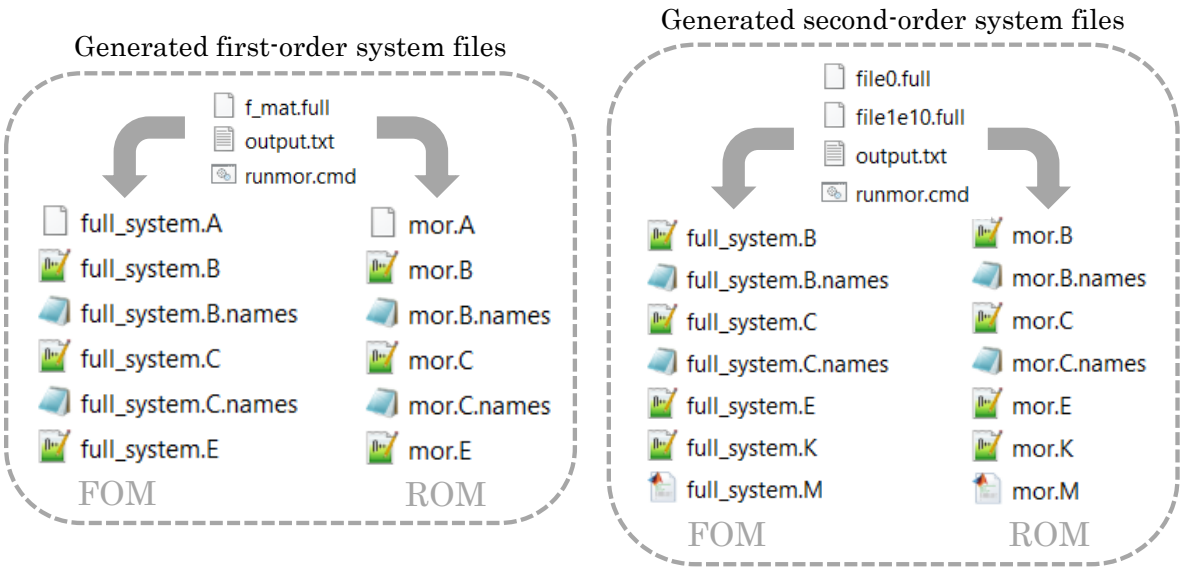


Figure 9: System matrix files generated via 'Model Reduction inside ANSYS' based on the FULL files extracted from the first- and second-order systems in ANSYS Mechanical. The output node information is saved in '`output.txt`' and '`runmor.cmd`' contains the command line for reduction.

4.2 Parametric Model Order Reduction

'Model Reduction inside ANSYS' also enables the reduction of the parametric model, in which the parameters can be extracted out of the system matrices as presented in Equation (3.1). In this case, let's consider a linear thermal model where the isotropic thermal conductivity is preserved as a parameter. The parametric system reads

$$\Sigma_N \begin{cases} E \cdot \dot{T}(t) = \underbrace{(A_0 + \kappa \cdot A_1)}_{A(\kappa)} \cdot T(t) + B \cdot u \\ y(t) = C \cdot T(t) \end{cases} . \tag{4.6}$$

It should be noted here that for technical reasons when using software 'Model Reduction inside ANSYS' to extract the system matrices from ANSYS Mechanical, matrix A_0 is obtained with fixed $\kappa = \kappa_0$. The matrix A_1 is then calculated through the equation

$$A_1 = \frac{A(\kappa_1) - A(\kappa_0)}{\kappa_1 - \kappa_0} , \tag{4.7}$$

where κ_1 should be a different value compared to κ_0. Then the model in Equation (4.6) is rewritten as follows

$$\Sigma_N \begin{cases} E \cdot \dot{T}(t) = \underbrace{[A_0(\kappa_0) + (\kappa - \kappa_0) \cdot A_1]}_{A(\kappa)} \cdot T(t) + B \cdot u \\ y(t) = C \cdot T(t) \end{cases} . \tag{4.8}$$

In order to generate the FULL files prepared for pMOR, the APDL commands below are used. The original thermal conductivity in the model is set as 0.16 W/m/K.

```
*create,mat1,txt ! Create 'mat.txt' file
cond 100 10      ! Name, parameter difference, number of Arnoldi vectors
*end
/assign,full,f_mat ! The FULL file is named as 'f_mat.full'
/solu
allsel
antype,harmic
eqslv,sparse
harfrq,1/2/3.141592653589793 ! Frequency w is set as 1 Hz
nsubst,1,1,1
wrfull,1
solve
/assign,full,cond,full ! The second FULL file is named as 'cond.full'
/prep7
cmsel,s,dummy ! Geometry with component name 'dummy' is selected.
```

```
MP,KXX,1000,100.16 ! Change the thermal conductivity to 100.16 W/m/K
MP,DENS,1000,7700
MP,C,1000,90
emodif,all,mat,1000
allsel
fini
/solu
allsel
antype,harmic
eqslv,sparse
harfrq,0           ! Frequency w is set as 0 Hz
nsubst,1
wrfull,1
solve
```

'`mat1.txt`' file is created to save the information of the parameter. '`cond`' is the name of the parameter. '`100`' is the difference of the parameter calculated as $\kappa_1 - \kappa_0$ and '`10`' is the number of Arnoldi vectors used for constructing the projection matrix of parameter κ as presented in Equation (3.10). Two FULL files '`f_mat.full`' and '`cond.full`' are generated when the thermal conductivity in the material is set as $\kappa_0 = 0.16$ W/m/K and $\kappa_1 = 100.16$ W/m/K, respectively. Thereby, matrix $A(\kappa_0) = A_0(\kappa_0)$ is written from '`f_mat.full`' and matrix $A(\kappa_1)$ is written from '`cond.full`'. The parameter-independent matrix A_1 is then calculated via Equation (4.7) with $\kappa_1 - \kappa_0 = 100$. The parametric system matrix files generated are presented in Figure 10.

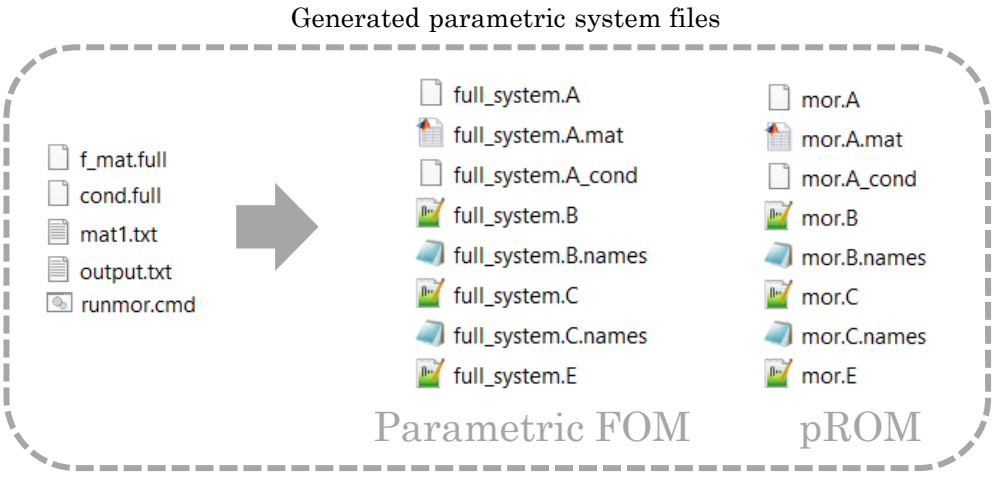


Figure 10: Parametric system matrix files generated via 'Model Reduction inside ANSYS' based on the FULL files extracted from ANSYS Mechanical, the output node information file '`output.txt`', and the parameter information file '`mat1.txt`'. The command line in '`runmor.cmd`' is used for the generation of the pROM. '`*.A`' and '`*.A_cond`' files to represent the full and reduced matrices $A_0\kappa_0$ and A_1 in Equation (4.8). '`*.A.mat`' files contain the name of the parameter.

To generate the pROM, the '`runmor.cmd`' file with the following command line is used:

```
mor_for_ansys f_mat.full -P mat1.txt -b -N 20 -C output.txt > mor.out
```

`-P`: setting for pMOR of A matrix.

Note that in the pMOR case, option '`-N`' defines the total number of Arnoldi vectors generated for the projection matrix of parameter s as presented in Equation (3.8). The dimension of the projection matrix of parameter κ is defined in '`mat1.txt`' and the global projection matrix used for pMOR is obtained via Equation (3.11). Therefore, if there is no deflation while merging these two local projection matrices; the final dimension of the pROM generated via the command line above is 30.

Chapter 4 Summary

This chapter mainly introduces the software 'Model Reduction inside ANSYS'. This industrial software enables the extraction of the full system matrices in Matrix Market format from ANSYS Mechanical. On the basis of the full system matrices, the ROMs can be generated automatically.

- The ROM generation of the first-order system is performed via Arnoldi or block Arnoldi iteration algorithms suggested in section 2.3.
- For second-order systems, the software implements the three different approaches introduced in section 2.5 for MOR.

The reduced system matrices of the ROM are saved in Matrix Market format and are further transformed into other formats which can be implemented in system-level simulation tools, e.g., state-space model in MATLAB and VHDL-AMS model in ANSYS Twin Builder.

For the generation of the pROM, only the multivariate moment matching-based pMOR method introduced in section 3.1 is implemented in 'Model Reduction inside ANSYS'. For the other two pMOR methods introduced in sections 3.2 and 3.3, the automation scripts presented in Appendices B and C are developed.

Part II

Case Studies

5 Modeling and Simulation of Energy Harvesters

This chapter presents the piezoelectric, thermoelectric, and electromagnetic energy harvesters to be used as case studies for the validation of the parametric model order reduction (pMOR) methods introduced in chapter 3. Moreover, three simplified versions of these three devices are constructed in this work to speed up the validation time. All these energy harvesters are implemented in ANSYS [17,18,55] and descriptions of these models are given in sections 5.1 - 5.3. To apply the model order reduction (MOR) and pMOR methods as mentioned in chapters 2 and 3, the material properties used in these cases are all linear-elastic. In addition, in section 5.2, the linearization strategies are implemented to build a linear human tissue bioheat thermal model.

5.1 Piezoelectric Energy Harvester

The piezoelectric energy harvester converts environmental vibrations into electrical energy, which can be used as a lifetime power supply as a substitute for batteries in wireless devices. Such vibration-based harvesters are mostly designed with spring-mass-damper systems and are eminently suitable for sensors installed in difficult or dirty environments. The maximum power output from such devices is obtained when the frequency of the resonator coincides with the environmental vibration frequency. In this work, two different coupled mass-spring resonators are implemented in commercial finite element method (FEM)-based software ANSYS Mechanical [17].

As depicted in Figure 11, the micro-fabricated piezoelectric energy harvester is initially proposed in [14] and studied by the authors further in [56]. It contains one inner and two outer mass elements. These are connected by beams. The piezoelectric patches are installed on the connection beams.

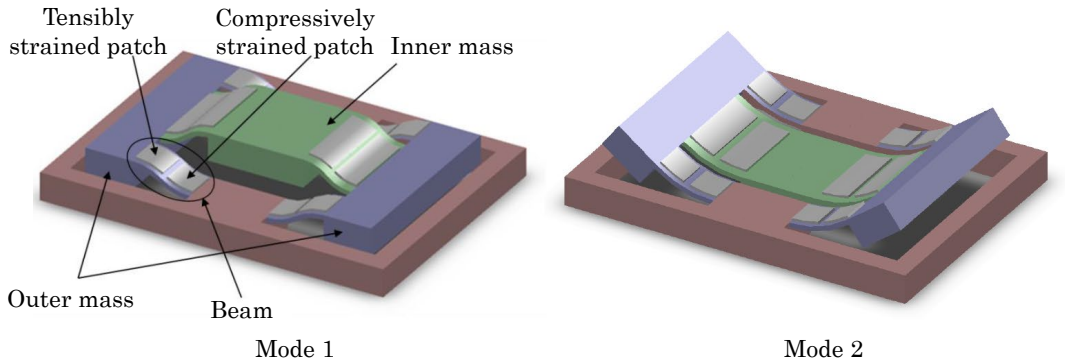


Figure 11: Schematic of the micro-fabricated piezoelectric energy harvester proposed in [14]. Device dimensions are $14\times9\times0.7$ mm^3. The beam thickness is 50 µm. The piezoelectric patch thickness is 5 µm. This figure is adapted from [56].

The flexible beam sections, the rigid outer and inner mass sections are made of sil-

icon. The piezoelectric patches are composed of metal parallel plate capacitors, where platinum (Pt) is used as the electrode material on top, while aluminum (Al) is used as the bottom electrode material. The piezoelectric thin film material in between is Aluminum nitride (AlN). Compared to the commonly used lead zirconate titanate (PZT), AlN is favored due to its more significant voltage generation and comparable power delivery performance. The mechanical material properties are specified in Table A1 and the piezoelectric properties of the thin film layer are given in Table A2.

This structure contains two coupled mechanical resonators and is intended to produce mechanical resonance in out-of-plane motion at two different frequencies. Therefore, the harvester can resonate at a minimum of two frequencies (see Figure 12). In the first mode shape at 1,966 Hz, the mass and the trusses move in phase while the mass moves up and down without tilting. Mass and truss segments act as clamped-guided beams. The piezoelectric patches undergo tensile and compressive strain types caused by the beam deflection. The second mode shape is at 3,756 Hz. The beam segments act as clamped-free beams and the piezoelectric patches undergo identical strain.

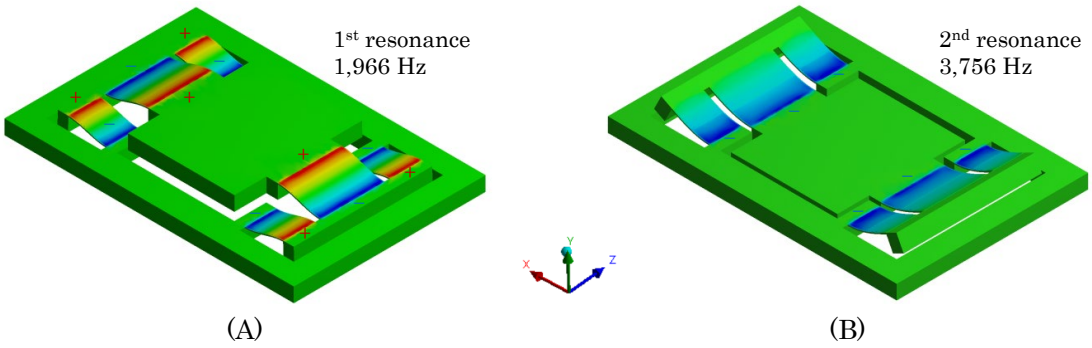


Figure 12: Plots of the normal elastic strain in the x-direction. (A) First mode shape at 1,966 Hz. The two piezoelectric patches on each beam experience tensile and compressive strain types while inner and outer mass segments move in phase (clamped-guided beam). (B) Second mode shape at 3,756 Hz. Both piezoelectric patches on each beam experience identical strain types while beam segments act as clamped-free. These figures are adapted from [56].

The second device is the frequency-tunable piezoelectric energy harvester proposed by S. Bouhedma [57,58], which is assembled of a folded beam in a clamp-free configuration as shown in Figure 13. This structure is designed due to the fact that the ambient vibrations in practical cases vary as to dominant frequency and consequently lead to inefficient power generation, caused by the mismatch between the vibration and harvester frequency. The (outer) tip mass is connected to the clamped part via two identical outer beams. An inner beam extends from the outer tip mass and it is in between the two outer beams toward the clamped part. It carries another (inner) tip mass on one side. The two masses have identical weights of 5 g. Masses and beams are all made of stainless steel. The piezoelectric patches made with PIC-225 (supplied by PI Ceramic GmbH) are integrated on the top surfaces of the inner and outer beams. They consist of thin cupronickel films

as top and bottom electrode material and a ceramic piezoelectric layer in between. The mechanical material properties are specified in Table A3 and the piezoelectric properties of the ceramic layer are given in Table A4.

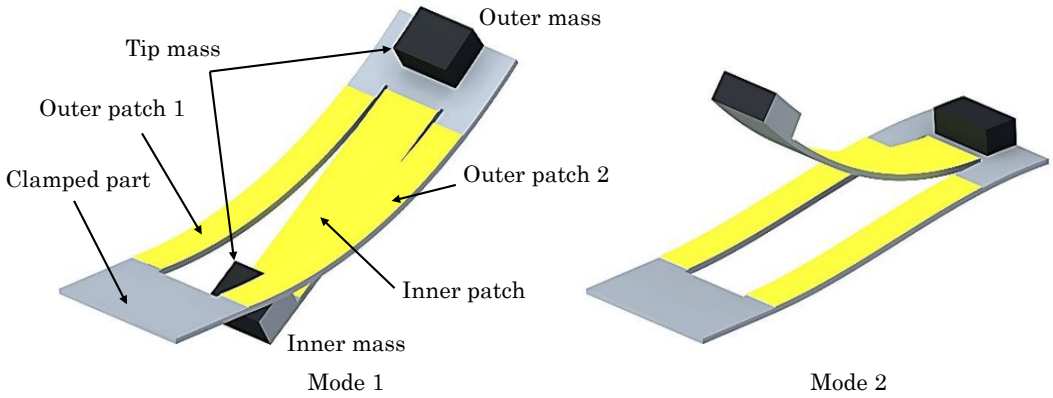


Figure 13: Schematic of the frequency-tunable piezoelectric energy harvester proposed in [57,58]. Beam dimensions are 100×40×1 mm^3. The dimensions of the tip mass are 10×18×4 mm^3. The piezoelectric patch thickness is 0.2 mm. This figure is adapted from [58, 59].

The flexible parts of the structure have two deformation mode shapes depending on the resonance frequency as shown in Figure 14. Either the outer or inner beam experiences substantial deformation, which leads to a polarization of the piezoelectric patches and generates a voltage across the patch electrodes.

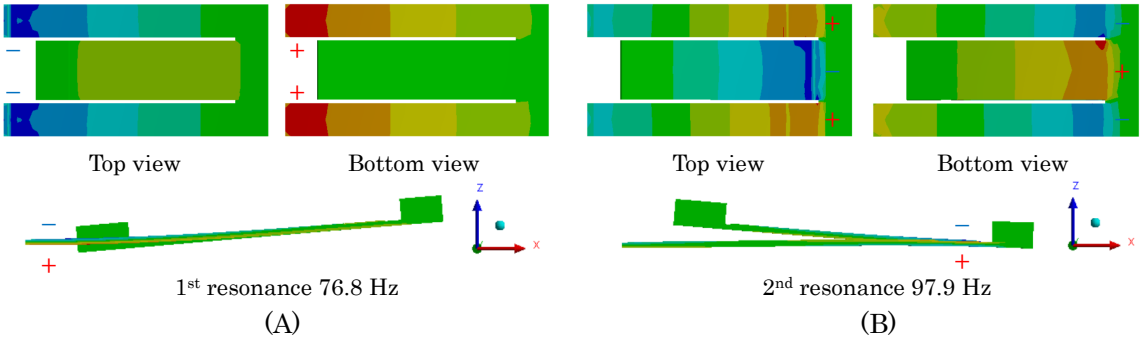


Figure 14: Descriptive scheme of the frequency-tunable piezoelectric energy harvester in two mode shapes corresponding respectively to the outer and the inner part of the harvester. Plots of the normal elastic strain in the x-direction. (A) The First mode shape at 76.8 Hz. (B) The second mode shape at 97.9 Hz.

In contrast to the micro-fabricated piezoelectric energy harvester, the resonant frequency of this device is tunable while external magnets are placed near the tip mass [57]. These magnets generate force which amplifies the restoring force of the beam, thereby hardening or softening the stiffness of the structure and leading to desirable frequency shifts.

Both models are implemented in the FEM-based software ANSYS Mechanical. The piezoelectric layer is modeled by using the solid element type with displacement and potential degrees of freedom (DOF) [60]. For other geometrical parts, an element with three translational DOF in each node is used. According to the stress-charge formulation in [61], the constitutive piezoelectric equations are described as follows

$$\begin{cases} T &= c \cdot S - e \cdot E \\ D &= e^T \cdot S + \epsilon \cdot E \end{cases}, \tag{5.1}$$

where T is the stress tensor in $\mathrm{N/m^2}$, S is the strain tensor in m/m, E is the electric field in V/m, D is the electrical charge displacement in $\mathrm{C/m^2}$, c is the elastic stiffness tensor in $\mathrm{N/m^2}$, e is the piezoelectric coupling tensor in $\mathrm{C/m^2}$, and ϵ is the electric permittivity of the material in F/m. In this work, mechanical damping and the assumption of no-loss piezoelectric material are considered. Given the finite element (FE) discretization of Equation (5.1) based on the variational principle [62], the governing equation of the piezoelectric energy harvester can be written as the following second-order system:

$$\Sigma_N : \begin{cases} \underbrace{\begin{bmatrix} M_{11} & 0 \\ 0 & 0 \end{bmatrix}}_{M} \underbrace{\begin{bmatrix} \ddot{d} \\ \ddot{v} \end{bmatrix}}_{\ddot{x}} + \underbrace{\begin{bmatrix} E_{11} & 0 \\ 0 & 0 \end{bmatrix}}_{E} \underbrace{\begin{bmatrix} \dot{d} \\ \dot{v} \end{bmatrix}}_{\dot{x}} + \underbrace{\begin{bmatrix} K_{11} & K_{12} \\ K_{21} & K_{22} \end{bmatrix}}_{K} \underbrace{\begin{bmatrix} d \\ v \end{bmatrix}}_{x} = \underbrace{\begin{bmatrix} B_1 \\ B_2 \end{bmatrix}}_{B} u \\ y = \underbrace{\begin{bmatrix} C_1 & C_2 \end{bmatrix}}_{C} \underbrace{\begin{bmatrix} d \\ v \end{bmatrix}}_{x} \end{cases}, \tag{5.2}$$

where in the state vector $x \in \mathbb{R}^N$, $d \in \mathbb{R}^{k_1}$ and $v \in \mathbb{R}^{k_2}$ presents the nodal displacement and nodal electrical potentials, respectively. $k_1 + k_2 = N$, where N is the dimension of the full-scale model. $M, E \in \mathbb{R}^{N \times N}$ are the mass and damping matrices, which only contain the submatrices $M_{11}, E_{11} \in \mathbb{R}^{k_1 \times k_1}$ from the mechanical domain. In the piezoelectric stiffness matrix $K \in \mathbb{R}^{N \times N}$, $K_{11} \in \mathbb{R}^{k_1 \times k_1}$ is the structural stiffness submatrix and $K_{22} \in \mathbb{R}^{k_2 \times k_2}$ is the dielectric conductivity submatrix. $K_{12} \in \mathbb{R}^{k_1 \times k_2}$ and $K_{21} \in \mathbb{R}^{k_2 \times k_1}$ are the piezoelectric coupling submatrices. This satisfies the condition that matrix K_{12} is symmetric to matrix K_{21}, which means $K_{12} = K_{21}^T$. Matrix K_{22} is negative semi-definite ($K_{22} \preceq 0$) due to the energy exchange between the mechanical and electrical domains. M_{11} and K_{11} are both symmetrical and positive definite ($M_{11} = M_{11}^T \succ 0$, $K_{11} = K_{11}^T \succ 0$). $E_{11} = \alpha M_{11} + \beta K_{11}$ is the damping matrix, where $\alpha, \beta \in \mathbb{R}$ are the Rayleigh damping constants. The input distribution matrix and the gathering matrix are denoted as $B \in \mathbb{R}^{N \times m}$ and $C \in \mathbb{R}^{p \times N}$ with m inputs and p outputs. Finally, $u \in \mathbb{R}^m$ is the input vector and $y \in \mathbb{R}^p$ is the user-defined output vector.

It should be noted that the original full system matrices extracted from ANSYS Mechanical are not saved in the form shown in Equation (5.2). In the state vector, the states

represent the DOF of each node, and the nodes are ordered according to the node number. Moreover, the DOF of the nodes from the Dirichlet boundary condition (BC) are excluded from the state vector. To obtain the full system matrices of the piezoelectric model within the block structure as shown in Equation (5.2), the dimension of the electrical state v is identified via calculating the sum of the elements in each row of matrix M or matrix E. The number of zero rows is denoted as k_2, which presents the number of nodal electrical potentials, and the rest of k_1 states present the nodal displacement results. Thereby, all the system matrices can be partitioned according to the indices found in these zero rows.

Additionally, except for these two differently structured energy harvesters, a single-beam piezoelectric energy harvester model (see Figure 15) is also prepared in this work. It uses the material properties in Table A3 and Table A4 for the tip mass, beam, and piezoelectric patch. This model is simulated in harmonic analysis and excited by a displacement input. The coupling voltage result from the piezoelectric patch is set as the output.

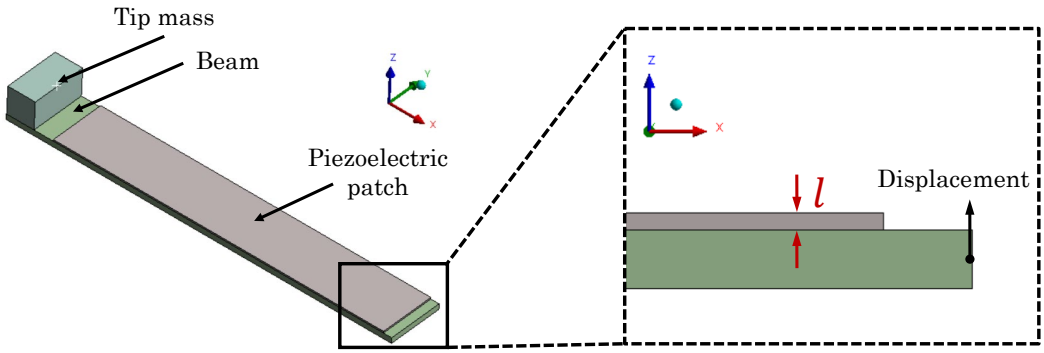


Figure 15: A single beam piezoelectric energy harvester model. The dimensions of the beam and the tip mass are $70\times10\times1$ mm^3 and $5\times10\times5$ mm^3, respectively. A $60\times10\times0.29$ mm^3 piezoelectric patch is attached to the top surface of the beam. A displacement is applied as the excitation on the side surface of the beam in a positive z-direction. The thickness of the patch is set as a geometrical parameter l in the parametric studies.

5.2 Thermoelectric Energy Harvester

A thermoelectric energy harvester, also known as a thermoelectric generator (TEG), can harvest thermal energy from the environment if there is a temperature gradient across the device. As a result, it can be utilized as a supplemental power source to prolong the life span of the batteries or a self-sufficient power supply of electrically active devices, e.g., wireless sensors or implantable medical devices. It is displayed in Figure 16 that, a TEG normally contains two ceramic plates and multiple thermocouples, each consisting of p-type and n-type bismuth telluride legs. The thermocouples are electrically connected in series through copper interconnects and thermally connected in parallel between two ceramic plates.

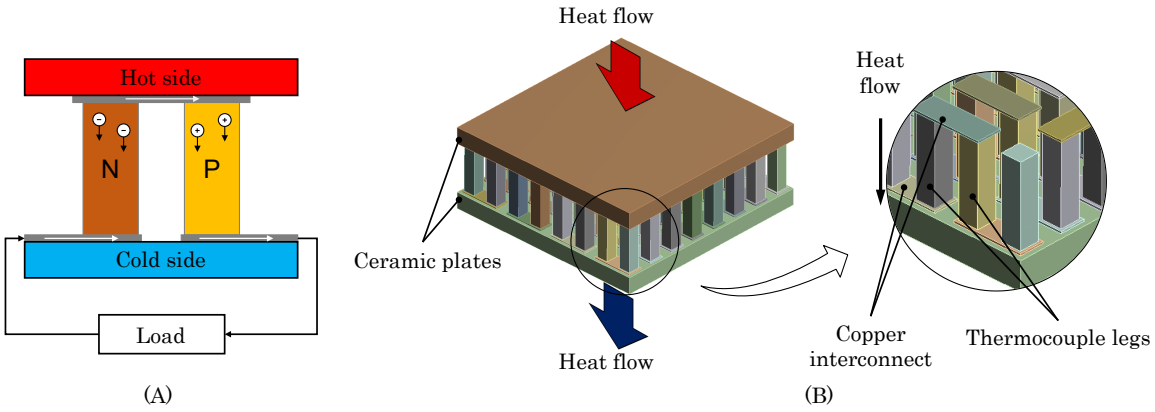


Figure 16: (A) Schematic of the thermocouple composed of p- and n-type legs connected electrically in series and thermally in parallel. (B) A thermoelectric module built in ANSYS Mechanical. Figure (B) is adapted from [63].

A temperature difference across the device will provide a voltage through the Seebeck effect as follows:

$$V_{out} = \frac{n}{2} \cdot \Delta T \cdot (\alpha_p - \alpha_n) \ , \tag{5.3}$$

where ΔT is the temperature difference across the device, n is the number of thermocouple legs and α_p, and α_n are the Seebeck coefficients of the p- and n-type thermocouple legs. It can be observed from Equation (5.3) that, if the Seebeck coefficients are temperature-independent, the Seebeck voltage is proportional to the temperature difference and the number of thermocouple legs. Then the power output of the TEG can be calculated by the following equation:

$$P = {V_{out}}^2 \cdot \frac{R_{load}}{(R_{TEG} + R_{load})^2} \ , \tag{5.4}$$

where R_{load}, R_{TEG} are the external load resistance and internal series resistance of the TEG. The maximum power output is obtained when $R_{load} = R_{TEG}$ and hence, Equation (5.4) can be written as

$$P_{max} = \frac{{V_{out}}^2}{4 \cdot R_{TEG}} \ . \tag{5.5}$$

Biomedical engineers use the TEG as a power supply in implantable medical devices by harvesting the thermal energy of the human body. In [64], the authors designed a model of a square-shaped TEG and integrated it into a simplified human tissue model. Pennes' bioheat equation [65] is used to describe the heat transfer in the tissue layers and it is solved to determine the temperature difference across the TEG. Years later, vivo experiments are conducted in [66] to demonstrate the potential of using an implantable TEG as a continuous power supply. Based on the work in [64], a simplified cuboid human body model (see Figure 17 (A)) is implemented with muscle, fat, and skin layers. The TEG is embedded into the fat layer to utilize the high temperature difference observed in Figure 17 (B). It is worth noting here that, in this case study, instead of constructing

a thermoelectric multiphysics coupled-domain model, a TEG-incorporated purely human tissue thermal model is implemented in ANSYS Mechanical. The temperature difference across the TEG is obtained and the Seebeck voltage and power output can be calculated via Equations (5.3) and (5.4).

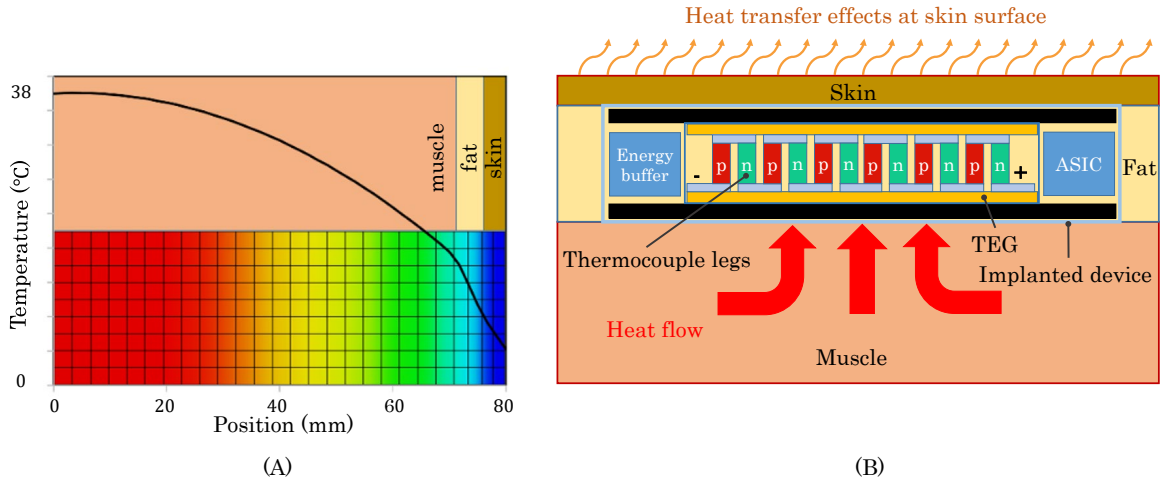


Figure 17: (A) Temperature profile inside human body tissue with no TEG present. A high change in temperature can be observed across the fat layer. (B) Assembling setup of a TEG-integrated electrically active implanted device positioned in the fat layer of the human tissue. The TEG is further connected to an energy buffer and application-specific integrated circuit (ASIC). Figure (A) is adapted from [15] and Figure (B) is adapted from [67].

The time transient heat transfer within human tissue is described by Pennes' bioheat equation, which accounts for heat conduction, metabolic heat generation, and perfusion effects as follows:

$$\rho c \frac{\partial T}{\partial t} = \nabla(\kappa \nabla T) + \underbrace{\rho_b c_b \omega (T_a - T)}_{Q_b(T)} + Q_m \ , \tag{5.6}$$

where T is the unknown temperature distribution on the human tissue and TEG. ρ, c, and κ are the density, specific heat, and thermal conductivity of different tissues, respectively. $Q_b(T)$ and Q_m are the blood perfusion and metabolic heat generation rates considered in different tissue layers. ρ_b, c_b are the density and specific heat capacity of the blood and ω is the blood perfusion rate in different body tissue layers. T_a is the blood temperature and it is usually assumed to be a constant 37 °C.

Meanwhile, for the human tissue model, the heat generated inside is balanced instantaneously by the various external heat transfer effects through the skin to the environment. For example, the convection, radiation, and evaporation effects are considered the main heat loss [68] and they are modeled as follows:

$$q_{skin} = \underbrace{h_c(T - T_{amb})}_{q_{conv}} + \underbrace{\sigma\epsilon(T^4 - {T_{amb}}^4)}_{q_{rad}} + \underbrace{h_e(P_{skin} - \phi P_{sa})}_{q_{eva}} \ , \tag{5.7}$$

where q_{conv} is the convection heat flux and q_{rad} and q_{eva} are the radiation and evaporation

heat fluxes, which are normal to the boundary skin surface. Firstly, in the convection effect, q_{conv}, T_{amb} is the temperature of the ambient air and h_c is the heat transfer coefficient (also known as film coefficient), which is defined as an air velocity-dependent parameter [68] in unit W/m^2/K as follows:

$$h_c = \begin{cases} 3.1 & \text{for} \quad 0 < v_{air} \leq 0.2 \\ 8.3 \cdot {v_{air}}^{0.6} & \text{for} \quad 0.2 < v_{air} < 4.0 \end{cases}, \tag{5.8}$$

where v_{air} is the air velocity in m/s. Secondly, the thermal process of radiation, q_{rad} is described by the Stefan-Boltzmann equation with the Stefan-Boltzmann constant σ and emissivity ϵ. Finally, the evaporation effect at the skin surface is described by q_{eva}, where h_e is the evaporation heat transfer coefficient, P_{skin} is the saturated vapor pressure at the skin surface, P_{sa} is the saturated vapor pressure and ϕ is the relative humidity. According to the Lewis relation [69], which shows that the evaporation coefficient could be presented in terms of the convection coefficient and Antoine's equation [68]. The evaporation heat loss q_{eva} in Equation 5.7 could be expressed with skin wetness w_{skin} as follows

$$q_{eva} = 1.65 h_c w_{skin} \left[\exp(18.956 - \frac{4030.18}{T_{skin} + 235}) - \phi \exp(18.956 - \frac{4030.18}{T_{amb} + 235}) \right] . \tag{5.9}$$

In this work, Pennes' bioheat equation and external heat transfer effects are implemented in either a simplified cuboid human tissue model or a realistic human torso model as shown in Figures 18 and 19. The TEG is located within the fat layer, where the maximum temperature gradient exists. Electrothermal material properties of TEG are given in Table A6 and A7, respectively. Realistic thermal data and physiologically correct material parameters from Table A5 are assigned to the various tissue sections.

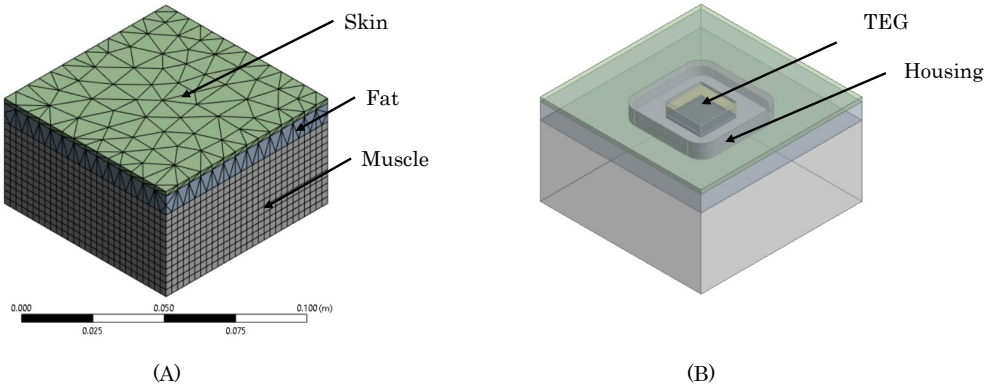


Figure 18: (A) Mesh topology of the simplified human tissue model. The lateral area of the tissue is 80×80 mm^2 and the thickness of the muscle, fat, and skin layers are 35, 8, and 2 mm, respectively. (B) The TEG is embedded in the fat layer. The electrothermal behavior of the TEG is represented by a cuboid with lateral dimensions 17.5×17.5 mm^2 and a height of 3.8 mm. It is enclosed between two copper layers of 0.1 mm thickness and two ceramic plates of 1.4 mm thickness. A hollow housing with outer dimensions of 40×40×6.8 mm^3 encases the TEG.

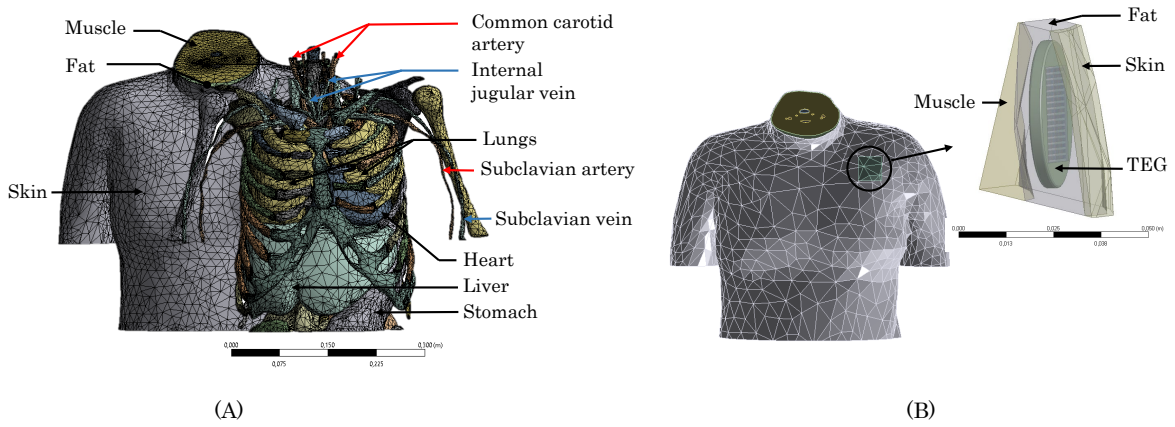


Figure 19: (A) Schematic of an upper-human torso model with solid internal organs, skeleton, main vessels, muscle, fat, and skin layers based on segmented magnetic resonance imaging (MRI) data [70]. (B) A TEG is placed in the fat layer of the chest region. These figures are adapted from [67].

However, the conventional MOR and pMOR methods are designed for linear systems. The human tissue model is presented by Pennes' bioheat equation as shown in Equation (5.6), where the blood perfusion heat generation rate is a resulting temperature-dependent input. Moreover, the nonlinear radiation and evaporation heat transfer effects at the skin surface are applied to balance the internal heat conduction in the human tissue. Therefore, linearization strategies are introduced in this work to construct a linear thermal model of human tissue.

According to the analogy between the blood perfusion heat generation rate Q_b in Equation (5.6) and the convection boundary condition q_{conv} in Equation (5.7), the blood perfusion heat generation rate Q_b could be treated as a 'convection-type' effect [71]. This temperature-dependent blood perfusion heat generation input could be transformed as a convective heat transfer boundary condition as follows:

$$q_b = -\rho_b c_b \omega (T_a - T) \ , \tag{5.10}$$

where the blood perfusion rate coefficient $\rho_b c_b \omega$ in Equation (5.10) is analogous to the film coefficient in the convection boundary condition. It is set as negative to let the heat flow into the human tissue as a heat generation input (See Appendix E).

The radiation effect in Equation (5.7), which describes the energy transfer between the human body and the environment by means of electromagnetic waves, is linearized if the temperature difference $\Delta T = T - T_{amb}$ is small [72]. The linear relationship of radiation effect could be obtained by expanding q_{rad} as Taylor series around the ambient temperature T_{amb} as follows:

$$q_{rad} = \underbrace{4\sigma\epsilon {T_{amb}}^3}_{h_{rad}} (T - T_{amb}) \ . \tag{5.11}$$

Comparing Equation (5.11) to the convection boundary condition q_{conv}, a radiation heat transfer coefficient $h_{rad} = 4\sigma\epsilon{T_{amb}}^3$ is obtained. Then the radiation effect is further integrated into the convection boundary condition through the sum of convection and radiation heat transfer coefficients.

The last nonlinearity occurs in the evaporation heat transfer effect in Equation (5.9), where no linear relationship could be found in this heat transfer effect. Therefore, the evaporation heat transfer effect is approximated by using the average heat flux generated from the evaporation heat transfer effect on each node of the skin surface. After the FE spatial discretization, N_{skin} nodes are generated at the skin surface and the heat fluxes from each node at k time steps in a transient analysis are collected and used to construct a snapshot matrix $X \in \mathbb{R}^{N_{skin} \times k}$,

$$X = [q_1 \quad q_2 \quad q_3 \quad \cdots \quad q_k] \,, \tag{5.12}$$

where the vector q_i $(i = 1, 2, 3 \ \cdots \ k)$ represents the vector of heat fluxes, in which the elements are the heat flux from each node of the skin surface at each time step. The average heat flux on each node is then obtained by calculating the weighted average of these snapshot heat flux vectors

$$\bar{q}_{eva} = \sum_{j=1}^{N_{skin}} (\sum_{i=1}^{k} w_i q_{i,j})/N_{skin} \,, \tag{5.13}$$

where w_i denotes the weights used to generate the average heat flux on each node of the skin surface. They are defined based on the ratio between the heat flux in each time step and the heat flux in the total time.

In addition, it should be noted that the linear thermal material properties are used in the TEG model in this work. As a result, combined with these linearization strategies introduced above, a linear thermal FE model of a human tissue model incorporating the TEG is presented as follows:

$$\Sigma_N \begin{cases} E \cdot \dot{T}(t) = A \cdot T(t) + B \cdot \underbrace{\begin{bmatrix} Q_m \\ T_a \\ \bar{q}_{eva} \end{bmatrix}}_{u} \\ y(t) = C \cdot T(t) \end{cases} , \tag{5.14}$$

where $A, E \in \mathbb{R}^{N \times N}$ are the global heat conductivity and heat capacity matrices. N is the dimension of the system. The blood perfusion and radiation heat transfer effects in Equation (5.10) and Equation (5.11) are applied as convection effects and integrated into the global heat conductivity matrix A. $B \in \mathbb{R}^{N \times m}$ is the input distribution array and u is

the input vector. Here the input vector contains the constant metabolic heat generation rate Q_m, blood temperature T_a, and the approximated average heat flux input $\bar{q}_{eva}$ on each node at the skin surface. $C \in \mathbb{R}^{p \times N}$ is the user defined output matrix. m and p are the number of inputs and user-defined outputs.

Based on the thermoelectric generator introduced in this section, a thermal cuboid model (see Figure 20) is implemented to imitate the thermal behavior of a thermocouple leg. It is simulated by means of a heat generation input and a temperature boundary condition. The temperature result from a selected node is set as the output.

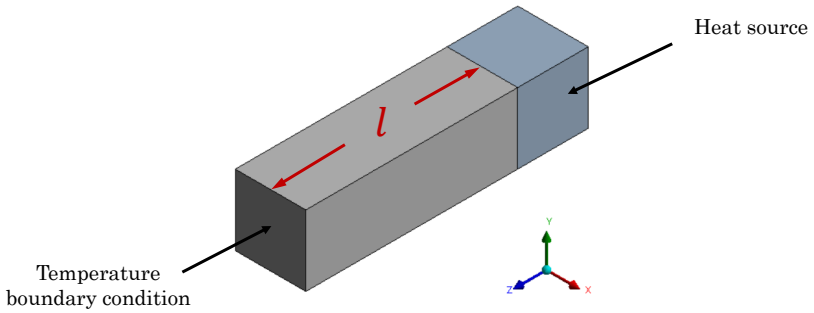


Figure 20: A thermal beam model with dimensions 4×1×1 mm^3. The heat source is applied on the 1×1×1 mm^3 cubic geometry and a temperature boundary condition is applied on the opposite surface. The length of the beam l is set as a geometrical parameter.

5.3 Electromagnetic Energy Harvester

An electromagnetic energy harvester transforms vibrational mechanical energy from the environment into electrical energy. In this work, an electromagnetic energy harvester suggested in [16] is reproduced in the FE software ANSYS Maxwell 3D [55]. Figure 21 illustrates the structure of the model.

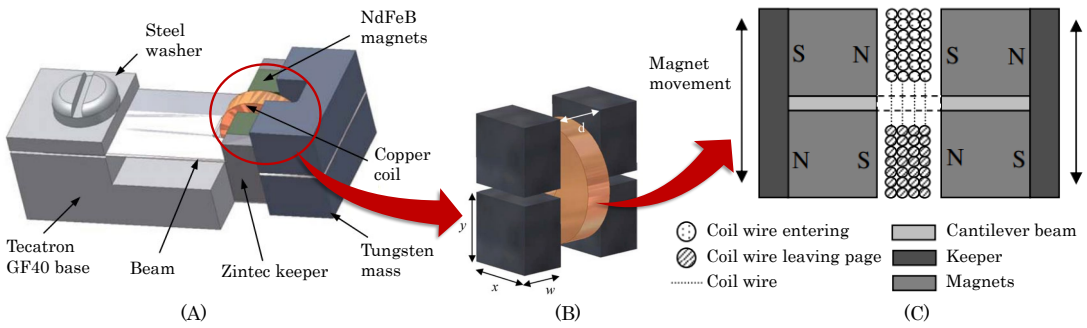


Figure 21: (A) A drawing of the assembled electromagnetic energy harvester introduced in [16]. (B) The schematic of the magnets and the coil. (C) Cross-section through the four-magnet arrangement. These figures are adapted from [16, 73].

Four high-energy density sintered rare earth neodymium iron boron (NdFeB) magnets are arranged along a copper coil. The characteristics of the materials used in the model are presented in Table A8. Two magnets are attached to the top and bottom sides of a 0.1 mm thick cantilever beam on either side of the coil. The distance d between the magnets on two sides of the coil is 1 mm. The magnets are $1{\times}1{\times}1.5$ mm^3 in size and polarized along the edge $w = 1.5$ mm. The coil in the middle of the structure has an outside radius of 1.2 mm, an inside radius of 0.3 mm, and a thickness of 0.5 mm. It is configured with 600 turns of 25 µm diameter copper wire. With this arrangement, a concentrated flux gradient through the stationary coil is produced as the magnets oscillate together with the cantilever. The changing magnetic field through the coil induces a voltage in the coil, which is utilized as a power supply.

The structure of four magnets and the coil are implemented in ANSYS Maxwell 3D as shown in Figure 22. The two magnets on each coil side are grouped and moved in a $9{\times}4.8{\times}8.2$ mm^3 motion band in the z-direction between -0.57 and 0.57 mm. The initial resting position is at -0.57 mm. In order to establish the harmonic motion of the magnets, a time-dependent force is applied to each group of magnets.

$$F(t) = m\,\omega^2\,x_0\cos(\omega\,t)\,, \tag{5.15}$$

where $m = 22.2$ µg being the mass of two magnets, the excitation frequency is $\omega = 2\pi f$ with $f = 60$ Hz and the designated oscillation amplitude is $x_0 = 0.57$ mm.

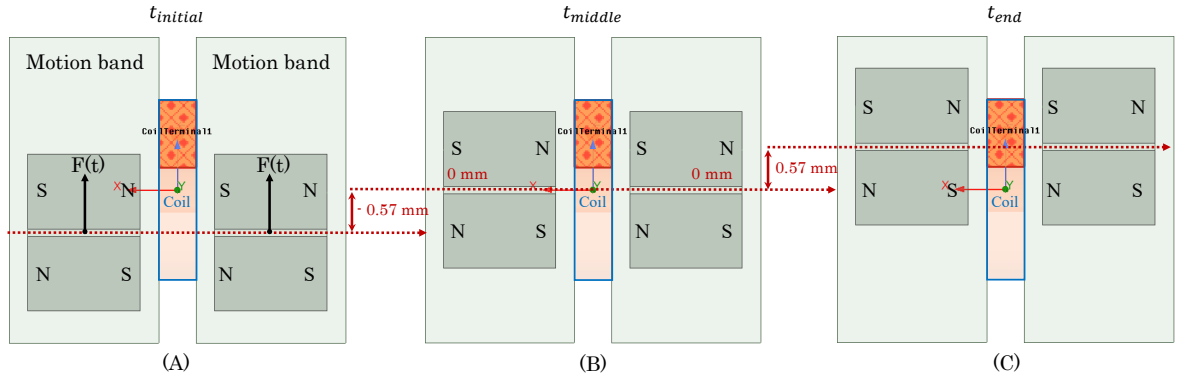


Figure 22: The side view of the positions of the magnets in the z-direction with time. (A) The initial resting position of the magnets is at -0.57 mm. (B) The center position of the magnets is at 0 mm. (C) The top position of the magnets is at 0.57 mm. This figure is adapted from [73].

The model is solved with solution type 'Transient' as shown in Figure 23 (A). The oscillation of the magnets is modeled via the motion setups, which are defined in two motion bands around the magnets. The cross-sectional area of the coil is defined as a terminal, which is connected to an external circuit. For usual electromagnetic coil applications in ANSYS Maxwell 3D, a current/voltage excitation is added to the circuit. In this case study, the external circuit is composed of a coil inductor 'L_coil', a coil

internal resistor 'R_coil' = 100 Ω, and a large load resistor 'R_load' = 10 GΩ. In this way, the open circuit voltage induced in the coil can be observed. There is no current/voltage excitation defined in the external circuit. Instead, the induced voltage in the coil (see Figure 23 (B)) calculated from ANSYS Maxwell 3D is used as the power supply. The maximum voltage output during transient analysis is 65.4 mV, which is close to the simulation finding of 64 mV from [16] with a relative error of 2.2 %.

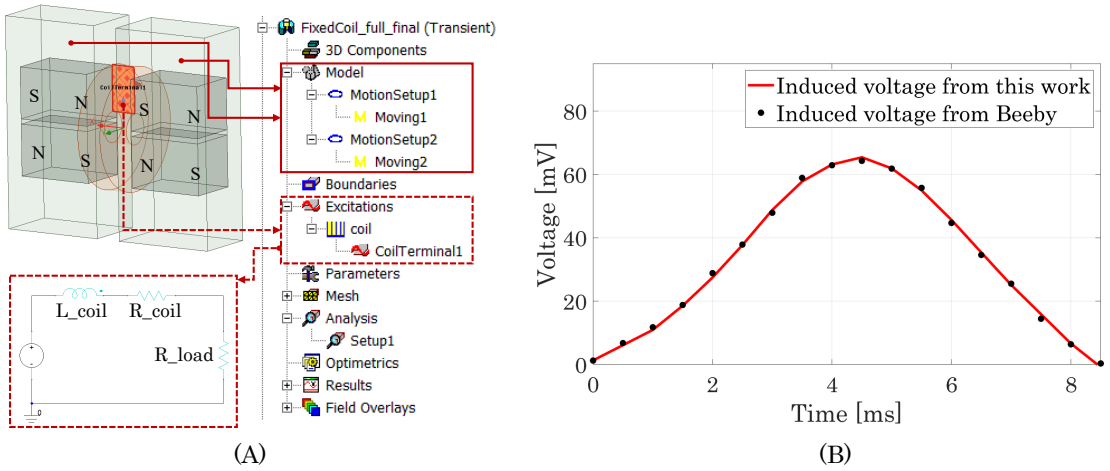


Figure 23: (A) Magnetic model setups in ANSYS Maxwell 3D. The position of the coil is fixed and two pairs of magnets are moved in the motion bands in a transient analysis. (B) Comparison of the open voltage results obtained from [16] and from ANSYS Maxwell 3D transient analysis in this work. These figures are adapted from [73].

For efficient design optimization of the device, applying pMOR to the original FE model is necessary. However, to the best of our knowledge, there is no way to obtain the system matrices from ANSYS Maxwell 3D. Therefore, the model is implemented in ANSYS Mechanical APDL [18], where the system matrices can be obtained via the software 'Model Reduction inside ANSYS' introduced in chapter 4.

In contrast to the model implemented in ANSYS Maxwell 3D, where a transient scheme is deployed to simulate the motion of the magnets, solution type 'Magnetostatic' is used in ANSYS Mechanical APDL and the positions of the magnets are parameterized to represent the oscillation of the magnets already used in the transient analysis. By using Faraday's law of induction [74], the induced voltage output in the coil is calculated via the magnetic flux rate change obtained through the coil

$$EMF = -N_{coil} \cdot \frac{\Delta\, \Phi}{\Delta\, t}\,, \tag{5.16}$$

where N_{coil} is the number of turns in the coil, $\Delta\ \Phi$ is the change of the magnetic flux through the coil in each time step $\Delta\ t$. The negative sign in the equation indicates that the induced electromotive force (EMF) opposes the change of the magnetic flux.

In this case study, the magnetostatic model is simulated without current input to the

coil. Therefore, the system can be represented by the subset of Maxwell's equations [75]

$$\nabla \times \mathrm{H} = \mathrm{J} \,, \tag{5.17}$$

$$\nabla \cdot \mathrm{B} = 0 \,, \tag{5.18}$$

where H is the magnetic field intensity, B is the magnetic flux density. J = 0 is the source current density applied. When permanent magnets are considered, a magnetization vector field M is introduced and it relates to the magnetic field intensity H and magnetic flux density B as follows:

$$\mathrm{B} = \mu_0(\mathrm{H} + \mathrm{M}) \,, \tag{5.19}$$

where μ_0 is the permeability of free space. In ANSYS Mechanical APDL, Equation (5.19) can be solved by using either

- Magnetic scalar potential: the magnetic field intensity H can be expressed as a negative gradient of the scalar potential ϕ,

  $$\mathrm{H} = -\nabla\phi \,. \tag{5.20}$$

  In Equation (5.19), replacing H by Equation (5.20) and putting Equation (5.19) into Equation (5.18), the following is obtained:

  $$0 = -\nabla \cdot \mu_0 \nabla\phi + \nabla \cdot \mu_0 \mathrm{M} \,. \tag{5.21}$$

- Magnetic vector potential: the magnetic flux density B is defined as a curl of the vector potential A,

  $$\mathrm{B} = \nabla \times \mathrm{A} \,. \tag{5.22}$$

  Replace B in Equation (5.19) by Equation (5.22) and put it into Equation (5.17), it is then written as follows:

  $$0 = -\nabla \times \frac{1}{\mu_0}(\nabla \times \mathrm{A}) + \nabla \times \mathrm{M} \,. \tag{5.23}$$

After FE discretization, both Equations (5.21) and (5.23) can be written in the matrix form as follows:

$$\Sigma_N : \begin{cases} 0 = K \cdot x + B \cdot u \\ y = C \cdot x \end{cases} \,, \tag{5.24}$$

where $K \in \mathbb{R}^{N \times N}$ is the coefficient matrix with dimension N. $B \in \mathbb{R}^{N \times m}$ is the input matrix with m inputs and $C \in \mathbb{R}^{p \times N}$ is the output matrix, and there are p outputs. x is the unknown state vector which contains the magnetic scalar potential ϕ or vector potential A. y is the output vector and u is the input vector, which is constructed by the

coercive force properties of the permanent magnets and defines the polarization direction.

In a simplified case study, a single magnet two-dimensional (2D) model is also implemented in ANSYS Mechanical APDL (see Figure 24) to validate the pMOR methods for geometrical parameters. It is simulated in magnetostatic analysis in the same way as the 3D electromagnetic model introduced in this section. The magnetic vector potential resulting from a selected node in the air domain is set as the output.

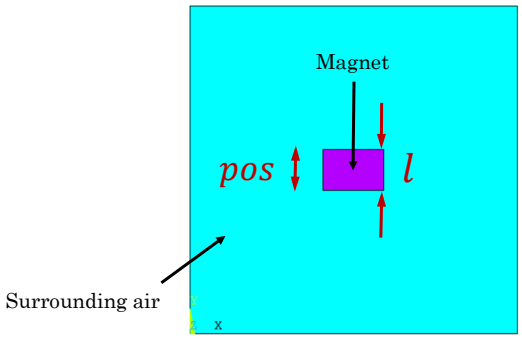


Figure 24: A single magnet 2D model with dimensions 1.5×1 mm^2 is surrounded by an air domain with dimension 8×8 mm^2. The height of the magnet l and the position of the magnet *pos* are set as geometrical parameters.

Chapter 5 Summary

This chapter introduces three energy harvester devices as case studies.

- A micro-fabricated piezoelectric energy harvester and a frequency-tunable piezoelectric energy harvester are implemented. In addition, a single beam piezoelectric energy harvester is also prepared for efficient validation of the pMOR methods. All models are constructed in multiphysics piezoelectric domain.

- A thermoelectric energy harvester as the case study is embedded in the fat layer of either a simplified human tissue model or a realistic human torso model. The heat transfer within human tissue is described by Pennes' bioheat equation. Convection, radiation, and evaporation effects are considered heat loss effects at the skin surface. Nonlinear effects are all linearized. In addition, a thermal beam model with a geometrical parameter is prepared for fast validation of the pMOR methods.

- An electromagnetic energy harvester is first simulated in ANSYS Maxwell 3D in 'Transient' and then implemented in ANSYS Mechanical APDL in 'Magnetostatic'. This is because there is no way to obtain full system matrices from ANSYS Maxwell 3D. A single 2D magnet model is prepared in ANSYS Mechanical APDL as a simplified model for validation of the pMOR methods.

Part III

Numerical Results

6 Parametric Model Order Reduction of Piezoelectric Energy Harvester

This chapter presents the stability issues of the reduced-order models (ROMs) which are generated from the multiphysics piezoelectric energy harvesters via conventional model order reduction (MOR) process, in section 6.1. For stable compact modeling of piezoelectric energy harvesters, stable MOR approaches are developed and introduced in section 6.2. Moreover, these stable MOR methods are combined with the matrix interpolation (MI)-based parametric model order reduction (pMOR) method described in section 3.2 to generate a stable parametric reduced-order model (pROM) of the piezoelectric energy harvester.

6.1 Stability of Reduced-Order Model

To generate a compact model of a second-order piezoelectric system, the second order Arnoldi reduction (SOAR) method can be used [32–34]. A survey of MOR methods for differential-algebraic equations can be found in [76]. Based on Equation (5.2), a projection matrix $V \in \mathbb{R}^{N\times r}$ is constructed by choosing the orthonormal basis of a second order input Krylov subspace via SOAR around expansion point $s_0 = 0$, as follows:

$$colspan\{V\} = \mathcal{K}_r\{-K^{-1}E, -K^{-1}M; -K^{-1}B\} \; . \tag{6.1}$$

Through projection, the reduced system of 5.2) is obtained as follows:

$$\Sigma_r : \begin{cases} \underbrace{V^T M V}_{M_r}\; \ddot{z} + \underbrace{V^T E V}_{E_r}\; \dot{z} + \underbrace{V^T K V}_{K_r}\; z = \underbrace{V^T B}_{B_r}\; u \\ y = \underbrace{C V}_{C_r}\; z \end{cases} , \tag{6.2}$$

where the full-scale state vector $x \in \mathbb{R}^N$ is projected onto a lower-dimensional subspace, $x \approx V\; z$, and $z \in \mathbb{R}^r$ is the reduced state vector. M_r, D_r and $K_r \in \mathbb{R}^{r\times r}$ are the reduced mass, damping, and stiffness matrices, respectively. $B_r \in \mathbb{R}^{r\times m}$ and $C_r \in \mathbb{R}^{p\times r}$ are the reduced input and output matrices. The dimension of the reduced system r is much smaller than the original full-order model (FOM) $r \ll N$ and hence the ROM enables efficient design optimization and system-level simulation.

Applying MOR to the piezoelectric devices described in section 5.1 is considered in this work. The ROMs for system-level simulations are depicted in Figure 25. The outer frame of the micro-fabricated piezoelectric energy harvester as well as the clamped part of the frequency-tunable piezoelectric energy harvester is the location where the displacement excitation (**displ**) is applied. For the micro-fabricated piezoelectric energy harvester

model, the nodal displacements (**center**, **south**, and **north**) of the mass and the trusses are computed. In the frequency-tunable energy harvester model, the nodal displacement results (**inner** and **outer**) from the inner and outer mass tips are set as the outputs. Moreover, both models contain two electrical ports (**el1** and **el2**).

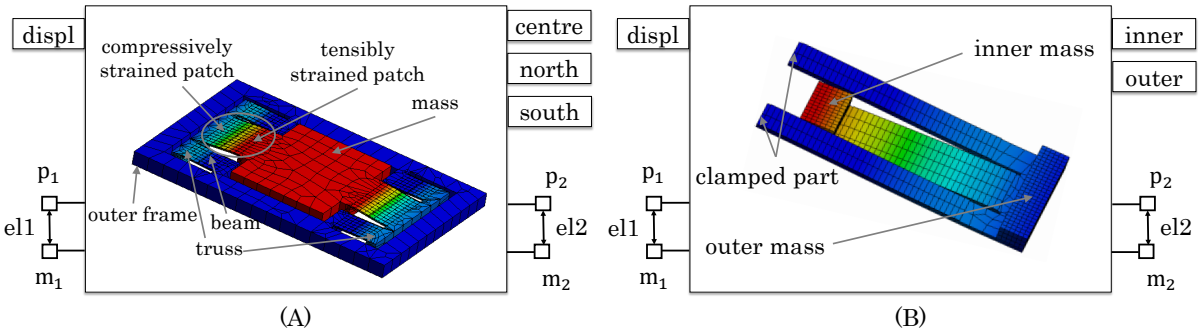


Figure 25: (A) Reduced micro-fabricated piezoelectric energy harvester model. (B) Reduced frequency-tunable piezoelectric energy harvester. In both case studies, **displ** is the displacement excitation input. **centre**, **north**, **south**, **inner**, and **outer** are the displacement output pins. **el1** and **el2** are the electrical ports. These figures are adapted from [59].

However, the research studies in [56, 77–79] showed the instability in the electrical response of the reduced piezoelectric model, which is also observed in the reduced piezoelectric model in this work. Figure 26 shows the diverging transient voltage outputs of the two reduced piezoelectric energy harvester models from Figure 25. Hence, both case studies show that the stability of the original system cannot be preserved via MOR and the ROM presented in Equation (6.2) cannot be used as a surrogate to replace and approximate the original large-dimensional model in the system-level simulations.

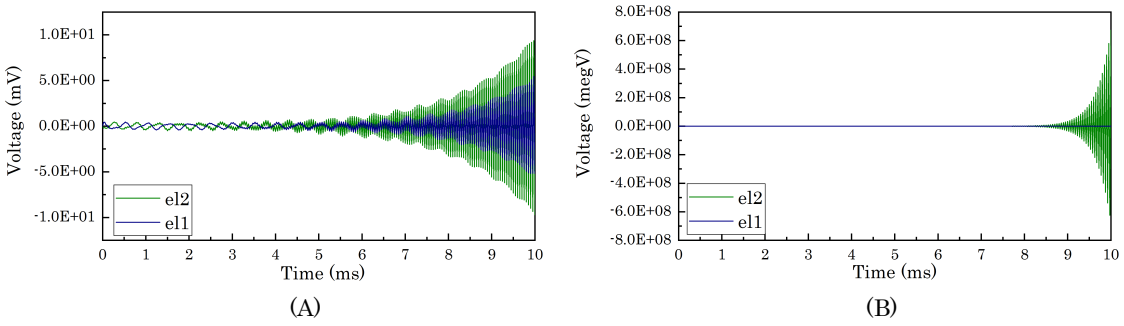


Figure 26: Transient responses at electrical ports **el1** and **el2** from reduced micro-fabricated piezoelectric energy harvester model (A) and reduced frequency-tunable piezoelectric energy harvester model (B) show the instability. These figures are adapted from [59].

According to the lemma from [34], the ROM is stable if the system matrices from the FOM fulfills the following properties:

$$M = M^T \succeq 0 \ , \ E + E^T \succeq 0 \ , \ K = K^T \succ 0 \ . \tag{6.3}$$

This means the stiffness matrix K requires to be symmetric and positive definite, that is $x^T K x > 0$ for all $x \in \mathbb{R}^N$. It is explained clearly in [80] that, in terms of mechanics, $Kx = F$ represents the force and thereby $x^T K x = x^T F$ represents a product of displacement and force, which is energy. When $x^T F$ is positive, the system receives energy or dissipates the energy when $x^T F$ is negative. In the piezoelectric energy harvester model, the stiffness matrix K given by Equation (5.2) incorporates mechanical and electrical domains. Therefore, due to the energy exchange between the mechanical and electrical domains, $x^T F$ changes its sign and the ROMs arising from piezoelectric systems are not guaranteed to be stable.

6.2 Stable Model Order Reduction

To generate stable compact models from the piezoelectric energy harvester devices, this section first reviews the stabilization approaches introduced in [56]. The basic working frames of the three methods, *MOR after Schur*, *Schur after MOR*, and *multiphysics structure-preserving MOR*, are briefly reviewed. By way of continuing their work, the method *MOR after implicit Schur* is developed and presented in this thesis. It generates the same ROM as the method *MOR after Schur*, but with much less computational effort.

6.2.1 MOR after Schur

The first method is noted as *MOR after Schur*, which means that, before reducing the piezoelectric energy harvester model, a Schur transformation (also called a Schur complement) is performed. Starting from the block structure presented in Equation (5.2), the purely algebraic (electrical) states are eliminated before MOR, as follows:

$$\begin{aligned} K_{21}\, d + K_{22}\, v &= B_2\, u\ , \\ \implies v &= K_{22}^{-1} B_2\, u - K_{22}^{-1} K_{21}\, d\ . \end{aligned} \tag{6.4}$$

Equation (6.4) is inserted into Equation (5.2) and the resulting Schur transformed system is obtained, as follows:

$$\Sigma_{schur}: \begin{cases} M_{11}\, \ddot{d} + E_{11}\, \dot{d} + \underbrace{(K_{11} - K_{12}K_{22}^{-1}K_{21})}_{K_S}\, d = \underbrace{(B_1 - K_{12}K_{22}^{-1}B_2)}_{B_S}\, u \\ y = \underbrace{(C_1 - C_2K_{22}^{-1}K_{21})}_{C_S}\, d + \underbrace{(C_2K_{22}^{-1}B_2)}_{D_S}\, u \end{cases} . \tag{6.5}$$

The system (6.5) can be reduced by the standard SOAR method, and a stable and accurate compact model can be generated, as proven by the authors in [58]. Note that $D_s = 0$ in Equation (6.5) is the result of the Schur complement of the algebraic part. It would be different from zero only if the inputs and outputs were identical and both in the

electrical domain, which is not the case in our model. For other real system models, D_s can usually be omitted [81]. Thereby, the reduced model of Equation (6.5) resulting from the projection can then be written as

$$\Sigma_{schur,r} : \begin{cases} V^T M_{11} V \ \ddot{d}_r + V^T E_{11} V \ \dot{d}_r + V^T K_S V \ d_r = V^T B_S \ u \\ y = C_S V \ d_r \end{cases} . \tag{6.6}$$

The great disadvantage of the method *MOR after Schur* is that the sparsity of the stiffness matrix K is lost through Schur transformation, as displayed in Figure 27. Consequently, the matrix factorization necessary for the MOR process (see Appendix D), results in a rise in computational costs. Table 2 shows that doubling the original system size, after the Schur transformation, results in an increase in the memory required by a factor of 6.3, and an increase in the time for MOR by a factor of 81. Note that this approach might be prohibitive for systems with more than 10^5 degrees of freedom (DOF), even using high-performance computing resources.

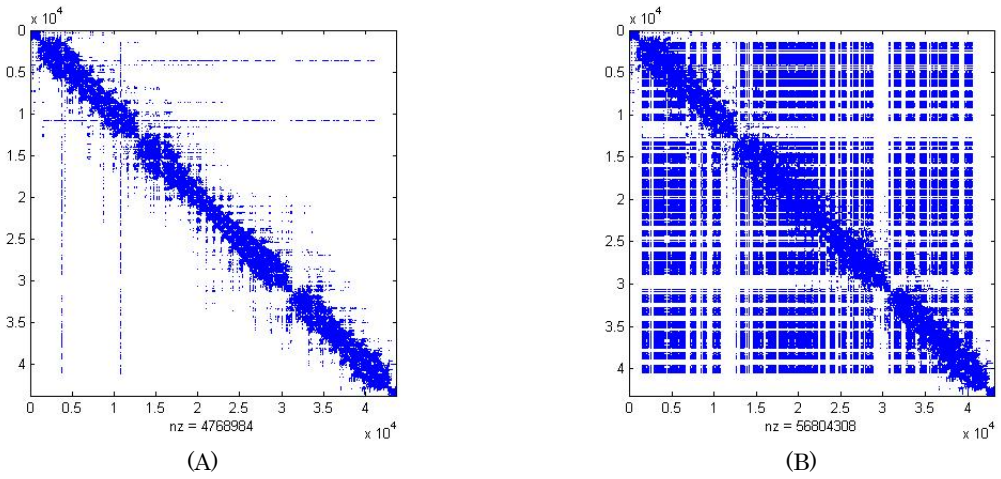


Figure 27: (A) For a piezoelectric system with 43,748 DOF, the number of nonzero elements in the initial system stiffness matrix is nz = 4,768,984. (B) After the Schur transformation, the number of elements in the stiffness matrix is increased to nz = 56,804,308. These figures are adopted from [56].

Table 2: Comparison of the computational time and required RAM derived from the same piezoelectric energy harvester model (all calculations are made when using a 3.1 GHz processor with 8 GB RAM). This table is adopted from [56].

System size	**43,748 DOF**	**90,663 DOF**
Original K matrix required memory	74 MB	168 MB
K matrix after Schur transformation required memory	944 MB	5.81 GB
MOR time of the initial system	51 s	130 s
MOR time of the system after Schur transformation	60 s	81 min

6.2.2 Schur after MOR

In the second method, called *Schur after MOR*, the reduced system in Equation (6.2) is approximated by differential-algebraic equations (DAEs) before being Schur transformed. This step is also called 'quasi-Schur transformation' in [82]. The general framework of this method is that the reduced mass matrix from Equation (6.2) can be factorized by means of eigenvalue decomposition

$$M_r = PDP^T \,, \tag{6.7}$$

where P is the matrix whose columns are eigenvectors of M_r and D is a sorted (largest to smallest) diagonal matrix of corresponding eigenvalues of M_r. Then the reduced system in Equation (6.2) is transformed into eigenbasis of its mass matrix via $z = P\,q$,

$$\widehat{\Sigma_r} : \begin{cases} D\,\ddot{q} + \underbrace{P^T E_r P}_{\widehat{E_r}}\,\dot{q} + \underbrace{P^T K_r P}_{\widehat{K_r}}\,q = \underbrace{P^T B_r}_{\widehat{B_r}}\,u \\ y = \underbrace{C_r P}_{\widehat{C_r}}\,q \end{cases} . \tag{6.8}$$

D is a sorted diagonal eigenvalue matrix of M_r, where $D_{ii} \geq D_{jj}$ holds for all $i < j$. Among these eigenvalues, those several orders of magnitude smaller are found and set $D_{ii} = 0$ for all $i \geq I$, with $I \in [1, r]$ such that $D_{(I-1)(I-1)} \gg D_{II}$. Therefore, matrix D is partitioned into blocks and approximated as follows:

$$D = \begin{bmatrix} D_1 & 0 \\ 0 & D_2 \end{bmatrix} \approx \begin{bmatrix} D_1 & 0 \\ 0 & 0 \end{bmatrix} = \widetilde{D} \,, \tag{6.9}$$

where $D_1 \in \mathbb{R}^{I\times I}$, $D_2 \in \mathbb{R}^{(r-I)\times(r-I)}$. D_2 presents the part with small eigenvalues and is ignored. Similarly to matrix $\widetilde{D}$, the corresponding columns and rows in matrix $\widehat{E_r}$ are rounded to zero and all the other system matrices are partitioned in accordance with the block structure of matrix $\widetilde{D}$

$$\widehat{E_r} = \begin{bmatrix} \widehat{E_{r11}} & \widehat{E_{r12}} \\ \widehat{E_{r21}} & \widehat{E_{r22}} \end{bmatrix} \approx \begin{bmatrix} \widehat{E_{r11}} & 0 \\ 0 & 0 \end{bmatrix} = \widetilde{E_r} \,, \tag{6.10}$$

$$\widehat{K_r} = \begin{bmatrix} \widehat{K_{r11}} & \widehat{K_{r12}} \\ \widehat{K_{r21}} & \widehat{K_{r22}} \end{bmatrix} , \; \widehat{B_r} = \begin{bmatrix} \widehat{B_{r1}} \\ \widehat{B_{r2}} \end{bmatrix} , \; \widehat{C_r} = \begin{bmatrix} \widehat{C_{r1}} & \widehat{C_{r2}} \end{bmatrix} . \tag{6.11}$$

where $\widehat{E_{r11}}, \widehat{K_{r11}} \in \mathbb{R}^{I\times I}$, $\widehat{E_{r22}}, \widehat{K_{r22}} \in \mathbb{R}^{(r-I)\times(r-I)}$, $\widehat{E_{r12}}, \widehat{K_{r12}} \in \mathbb{R}^{I\times(r-I)}$, $\widehat{E_{r21}}, \widehat{K_{r21}} \in \mathbb{R}^{(r-I)\times I}$, $\widehat{B_{r1}} \in \mathbb{R}^{I\times m}$, $\widehat{B_{r2}} \in \mathbb{R}^{(r-I)\times m}$, $\widehat{C_{r1}} \in \mathbb{R}^{p\times I}$ and $\widehat{C_{r2}} \in \mathbb{R}^{p\times(r-I)}$. Then, a reduced system with the same block structure as the full-order piezoelectric system (5.2) is written

as follows:

$$\widehat{\Sigma_r} \approx: \begin{cases} \underbrace{\begin{bmatrix} D_1 & 0 \\ 0 & 0 \end{bmatrix}}_{\widetilde{D}} \underbrace{\begin{bmatrix} \ddot{q}_1 \\ \ddot{q}_2 \end{bmatrix}}_{\ddot{q}} + \underbrace{\begin{bmatrix} \widehat{E_{r11}} & 0 \\ 0 & 0 \end{bmatrix}}_{\widetilde{E_r}} \underbrace{\begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \end{bmatrix}}_{\dot{q}} + \underbrace{\begin{bmatrix} \widehat{K_{r11}} & \widehat{K_{r12}} \\ \widehat{K_{r21}} & \widehat{K_{r22}} \end{bmatrix}}_{\widehat{K_r}} \underbrace{\begin{bmatrix} q_1 \\ q_2 \end{bmatrix}}_{q} = \underbrace{\begin{bmatrix} \widehat{B_{r1}} \\ \widehat{B_{r2}} \end{bmatrix}}_{\widehat{B_r}} u \\ y = \underbrace{\begin{bmatrix} \widehat{C_{r1}} & \widehat{C_{r2}} \end{bmatrix}}_{\widehat{C_r}} \underbrace{\begin{bmatrix} q_1 \\ q_2 \end{bmatrix}}_{q} \end{cases} . \quad (6.12)$$

Finally, the reduced system in Equation (6.12) is Schur-transformed to generate a stable reduced model as follows:

$$\widehat{\Sigma_r}_{,schur} \begin{cases} D_1\, \ddot{q}_1 + \widehat{E_{r11}}\, \dot{q}_1 + \underbrace{(\widehat{K_{r11}} - \widehat{K_{r12}}\widehat{K_{r22}}^{-1}\widehat{K_{r21}})}_{\widehat{K}_S}\, q_1 = \underbrace{(\widehat{B_{r1}} - \widehat{K_{r12}}\widehat{K_{r22}}^{-1}\widehat{B_{r2}})}_{\widehat{B}_S}\, u \\ y = \underbrace{(\widehat{C_{r1}} - \widehat{C_{r2}}\widehat{K_{r22}}^{-1}\widehat{K_{r21}})}_{\widehat{C}_S}\, q_1 + \underbrace{(\widehat{C_{r2}}\widehat{K_{r22}}^{-1}\widehat{B_{r2}})}_{\widehat{D}_S}\, u \end{cases} . \quad (6.13)$$

It is mathematically proven in [82] that the *Schur after MOR* method can actually stabilize the reduced piezoelectric system. Compared to the method *MOR after Schur*, the method *Schur after MOR* avoids performing the Schur transformation on the full-order piezoelectric stiffness matrix K and avoids generating a matrix K_s with a large number of elements, as illustrated in Figure 27, because the Schur transformation is applied in the reduced space. Note that, in the software 'Model Reduction inside ANSYS', the Schur transformation of the reduced system is performed based on the eigenbasis of matrix K_r. The index I is obtained according to the number of negative eigenvalues in K_r. This approach is equivalent and easier to implement.

6.2.3 Multiphysics Structure-Preserving MOR

The *multiphysics structure-preserving MOR* method is originally proposed in [56]. It is designed based on the approach *SPRIM* introduced by Freund [83]. *SPRIM* is initially applied to passive electrical circuits for preserving the block structure of the system matrices. By analogy, taking advantage of the system matrices' structure in Equation (5.2), the block structure of the piezoelectric system, originating from the mechanical and electrical domains, can remain separated while performing MOR. Compared with the dense reduced system matrices obtained from the methods *MOR after Schur* and *Schur after MOR*, *multiphysics structure-preserving MOR* preserves the particular block structure in the reduced system matrices from the original full-scale system. The basic idea of this method is partitioning the local projection matrix $V \in \mathbb{R}^{N \times r}$ into mechanical and

electrical parts according to the block sizes of the system matrices in Equation (5.2).

$$V = \begin{bmatrix} V_m \\ V_e \end{bmatrix} \rightarrow \widetilde{V} = \begin{bmatrix} \widetilde{V}_m & 0 \\ 0 & \widetilde{V}_e \end{bmatrix} , \tag{6.14}$$

where $\widetilde{V} \in \mathbb{R}^{N\times(r_1+r_2)}$, $\widetilde{V}_m \in \mathbb{R}^{k_1\times r_1}$, $\widetilde{V}_e \in \mathbb{R}^{k_2\times r_2}$. As mentioned in section 5.1, the size of the electrical part k_2 is obtained by calculating the sum of the elements in each row of the mass matrix M and finding the zero rows where $\Sigma_{j=1}^{N} M_{ij} = 0$. Differently to *SPRIM*, the number of Arnoldi basis vectors in the reconstructed projection matrix $\widetilde{V}$ is not just twice as many vectors in V because the columns of the projection matrix $\widetilde{V}$ are linearly independent due to the multi-physical nature of the original model. Corresponding to the ratio between the number of mechanical and electrical states, $\widetilde{V}_m \in \mathbb{R}^{k_1\times r_1}$ is chosen as the projection matrix for the mechanical domain, and $\widetilde{V}_e \in \mathbb{R}^{k_2\times r_2}$ is chosen as the projection matrix for the electrical domain, where $r_2 < r_1$ and $r_1 + r_2 = r$. Then, the piezoelectric system in Equation (5.2) can be reduced blockwise through the special feature of the projection matrix $\widetilde{V}$. The structure preserved ROM can be obtained as follows:

$$\widetilde{\Sigma}_r \begin{cases} \underbrace{\begin{bmatrix} \widetilde{V}_m^T M_{11} \widetilde{V}_m & 0 \\ 0 & 0 \end{bmatrix}}_{\widetilde{M}_r} \begin{bmatrix} \ddot{d}_r \\ \ddot{v}_r \end{bmatrix} + \underbrace{\begin{bmatrix} \widetilde{V}_m^T E_{11} \widetilde{V}_m & 0 \\ 0 & 0 \end{bmatrix}}_{\widetilde{E}_r} \begin{bmatrix} \dot{d}_r \\ \dot{v}_r \end{bmatrix} + \underbrace{\begin{bmatrix} \widetilde{V}_m^T K_{11} \widetilde{V}_m & \widetilde{V}_m^T K_{12} \widetilde{V}_e \\ \widetilde{V}_e^T K_{21} \widetilde{V}_m & \widetilde{V}_e^T K_{22} \widetilde{V}_e \end{bmatrix}}_{\widetilde{K}_r} \begin{bmatrix} d_r \\ v_r \end{bmatrix} = \underbrace{\begin{bmatrix} \widetilde{V}_m^T B_1 \\ \widetilde{V}_e^T B_2 \end{bmatrix}}_{\widetilde{B}_r} u \\ y = \underbrace{\begin{bmatrix} C_1 \widetilde{V}_m & C_2 \widetilde{V}_e \end{bmatrix}}_{\widetilde{C}_r} \begin{bmatrix} d_r \\ v_r \end{bmatrix} \end{cases} . \tag{6.15}$$

This method allows for preservation of the block structure of the piezoelectric system during the MOR process and shows no instabilities in the reduced model afterward. The stability of the reduced piezoelectric model obtained via the *multiphysics structure preserving MOR* method is guaranteed and mathematically proven in [59].

6.2.4 MOR after Implicit Schur

As mentioned in section 6.2.1, the main disadvantage of the *MOR after Schur* method is that the sparsity of the piezoelectric stiffness matrix K is lost through Schur transformation and the transformed stiffness matrix K_s is fulfilled with elements. This leads to an undesired computational cost for the MOR process. In this section, an improved stable MOR approach *MOR after implicit Schur*, which preserves the stability of the reduced piezoelectric system without explicit Schur complement, will be introduced. The mathematical proof of this method will also be given in this section.

The method *MOR after implicit Schur* presented in this work is oriented on the work in [81], where the authors show that an alternative realization for the first-order semi-explicit system can be established by projecting the input matrix onto the right deflating subspace

corresponding to the finite eigenvalues. The alternative realization, which only differs from the original system in the input matrix B, could tackle the numerical inefficiency caused by the Schur complement. Therefore, this implicit Schur complement is introduced into our second-order semi-explicit piezoelectric system in order to preserve the original structure of the stiffness matrix, as presented in Equation 6.16. The Schur complement is performed implicitly through projection during the MOR process, as stated in the following result.

Definition 6.1 (Alternative Realization)**.** *The alternative realization of the semi-explicit second-order piezoelectric system in Equation (5.2) is given by*

$$\Sigma_{implicit\_schur}: \begin{cases} \underbrace{\begin{bmatrix} M_{11} & 0 \\ 0 & 0 \end{bmatrix}}_{M} \begin{bmatrix} \ddot{d} \\ \ddot{v} \end{bmatrix} + \underbrace{\begin{bmatrix} E_{11} & 0 \\ 0 & 0 \end{bmatrix}}_{E} \begin{bmatrix} \dot{d} \\ \dot{v} \end{bmatrix} + \underbrace{\begin{bmatrix} K_{11} & K_{12} \\ K_{21} & K_{22} \end{bmatrix}}_{K} \begin{bmatrix} d \\ v \end{bmatrix} = \underbrace{\begin{bmatrix} B_s \\ 0 \end{bmatrix}}_{\tilde{B}} u \\ y = \underbrace{\begin{bmatrix} C_1 & C_2 \end{bmatrix}}_{C} \begin{bmatrix} d \\ v \end{bmatrix} + D_s u \end{cases}, \tag{6.16}$$

where in the input matrix $\tilde{B}$, $B_s = B_1 - K_{11}{K_{22}}^{-1}B_2$. In addition, the feed-through matrix D_s is added, which is normally equal to zero, as explained in Equation (6.5).

This sparse alternative realization can be used to compute quantities of interest with respect to the Schur complement from Equation (6.5), as can be seen by comparing the projection matrix $\overline{V} \in \mathbb{R}^{N \times r}$ for the alternative realization in Equation (6.16) with the second-order Krylov-subspace $V \in \mathbb{R}^{k_1 \times r}$ in Equation (6.6). Note that Equation (6.2) and Equation (6.16) share the same transfer function, hence preserving the moment matching property. Similar expressions are given in [76, 84].

Lemma 6.1. *Let $\overline{V}$ be a basis for the second-order input Krylov subspace corresponding to the alternative realization in Equation (6.16) and consider the partition with $V_1 \in \mathbb{R}^{k_1 \times r}$.*

$$\overline{V} = \begin{bmatrix} V_1 \\ V_2 \end{bmatrix} = \begin{bmatrix} V_1 \\ -{K_{22}}^{-1}K_{21}V_1 \end{bmatrix}, \tag{6.17}$$

Then $V_1 = V$ is a basis for the second-order Krylov subspace from Equation (6.5).

Proof. Consider the projection matrix V constructed from system (6.5)

$$colspan\{V\} = \mathcal{K}\{-{K_s}^{-1}E_{11}, -{K_s}^{-1}M_{11}; -{K_s}^{-1}B_s\} = span\{r_0, r_1, r_2, ..., r_{q-1}\}\ , \tag{6.18}$$

$$r_0 = -{K_s}^{-1}B_s\ , \tag{6.19}$$

$$r_1 = -{K_s}^{-1}E_{11}\ r_0\ , \tag{6.20}$$

$$r_j = -{K_s}^{-1}E_{11}\ r_{j-1} - {K_s}^{-1}M_{11}\ r_{j-2}\ ,\ j \geq 2\ . \tag{6.21}$$

Also consider the projection matrix $\overline{V}$ constructed from the system (6.16)

$$colspan\{\overline{V}\} = \mathcal{K}\{-K^{-1}E, -K^{-1}M; -K^{-1}\widetilde{B}\} = span\{\overline{r}_0, \overline{r}_1, \overline{r}_2, ..., \overline{r}_{q-1}\} \ , \tag{6.22}$$

$$\overline{r}_0 = -K^{-1}\widetilde{B} \ , \tag{6.23}$$

$$\overline{r}_1 = -K^{-1}E\,\overline{r}_0 \ , \tag{6.24}$$

$$\overline{r}_j = -K^{-1}E\,\overline{r}_{j-1} - K^{-1}M\,\overline{r}_{j-2} \ , \ j \geq 2 \ . \tag{6.25}$$

1) When $j = 0$, partition $\overline{r}_0$ according to the block structure of matrices M, E, K.

$$\overline{r}_0 = \begin{bmatrix} \overline{r}_{01} \\ \overline{r}_{02} \end{bmatrix} = -K^{-1}\widetilde{B} = -\begin{bmatrix} K_{11} & K_{12} \\ K_{21} & K_{22} \end{bmatrix}^{-1} \begin{bmatrix} B_s \\ 0 \end{bmatrix} ,$$

$$\begin{bmatrix} K_{11} & K_{12} \\ K_{21} & K_{22} \end{bmatrix} \begin{bmatrix} \overline{r}_{01} \\ \overline{r}_{02} \end{bmatrix} = -\begin{bmatrix} B_s \\ 0 \end{bmatrix} ,$$

$$\begin{bmatrix} \underbrace{(K_{11} - K_{12}{K_{22}}^{-1}K_{21})}_{K_s}\overline{r}_{01} \\ \overline{r}_{02} \end{bmatrix} = \begin{bmatrix} -B_s \\ -{K_{22}}^{-1}K_{21}\overline{r}_{01} \end{bmatrix} ,$$

$$\implies \overline{r}_0 = \begin{bmatrix} \overline{r}_{01} \\ \overline{r}_{02} \end{bmatrix} = \begin{bmatrix} -{K_s}^{-1}B_s \\ -{K_{22}}^{-1}K_{21}\overline{r}_{01} \end{bmatrix} . \tag{6.26}$$

2) When $j = 1$, partition $\overline{r}_1$ as follows:

$$\overline{r}_1 = \begin{bmatrix} \overline{r}_{11} \\ \overline{r}_{12} \end{bmatrix} = -K^{-1}E \cdot \overline{r}_0 = -\begin{bmatrix} K_{11} & K_{12} \\ K_{21} & K_{22} \end{bmatrix}^{-1} \begin{bmatrix} E_{11} & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \overline{r}_{01} \\ \overline{r}_{02} \end{bmatrix} ,$$

$$\begin{bmatrix} K_{11} & K_{12} \\ K_{21} & K_{22} \end{bmatrix} \begin{bmatrix} \overline{r}_{11} \\ \overline{r}_{12} \end{bmatrix} = -\begin{bmatrix} E_{11} & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \overline{r}_{01} \\ \overline{r}_{02} \end{bmatrix} ,$$

$$\begin{bmatrix} \underbrace{(K_{11} - K_{12}{K_{22}}^{-1}K_{21})}_{K_s}\overline{r}_{11} \\ \overline{r}_{12} \end{bmatrix} = \begin{bmatrix} -E_{11}\overline{r}_{01} \\ -{K_{22}}^{-1}K_{21}\overline{r}_{11} \end{bmatrix} ,$$

$$\implies \overline{r}_1 = \begin{bmatrix} \overline{r}_{11} \\ \overline{r}_{12} \end{bmatrix} = \begin{bmatrix} -{K_s}^{-1}E_{11}\overline{r}_{01} \\ -{K_{22}}^{-1}K_{21}\overline{r}_{11} \end{bmatrix} . \tag{6.27}$$

3) For the jth vector $\overline{r}_j$, $j \geq 2$,

$$\overline{r}_j = \begin{bmatrix} \overline{r}_{j1} \\ \overline{r}_{j2} \end{bmatrix} = -K^{-1}E \cdot \overline{r}_{j-1} - K^{-1}M \cdot \overline{r}_{j-2}$$

$$= - \begin{bmatrix} K_{11} & K_{12} \\ K_{21} & K_{22} \end{bmatrix}^{-1} \begin{bmatrix} E_{11} & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \overline{r}_{(j-1)1} \\ \overline{r}_{(j-1)2} \end{bmatrix} - \begin{bmatrix} K_{11} & K_{12} \\ K_{21} & K_{22} \end{bmatrix}^{-1} \begin{bmatrix} M_{11} & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \overline{r}_{(j-2)1} \\ \overline{r}_{(j-2)2} \end{bmatrix},$$

$$\begin{bmatrix} K_{11} & K_{12} \\ K_{21} & K_{22} \end{bmatrix} \begin{bmatrix} \overline{r}_{j1} \\ \overline{r}_{j2} \end{bmatrix} = - \begin{bmatrix} E_{11} & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \overline{r}_{(j-1)1} \\ \overline{r}_{(j-1)2} \end{bmatrix} - \begin{bmatrix} M_{11} & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \overline{r}_{(j-2)1} \\ \overline{r}_{(j-2)2} \end{bmatrix},$$

$$\begin{bmatrix} \underbrace{(K_{11} - K_{12}{K_{22}}^{-1}K_{21})}_{K_s} \overline{r}_{j1} \\ \overline{r}_{j2} \end{bmatrix} = \begin{bmatrix} -E_{11}\overline{r}_{(j-1)1} - M_{11}\overline{r}_{(j-2)1} \\ -{K_{22}}^{-1}K_{21}\overline{r}_{j1} \end{bmatrix},$$

$$\Longrightarrow \overline{r}_j = \begin{bmatrix} \overline{r}_{j1} \\ \overline{r}_{j2} \end{bmatrix} = \begin{bmatrix} -{K_s}^{-1}E_{11}\overline{r}_{(j-1)1} - {K_s}^{-1}M_{11}\overline{r}_{(j-2)1} \\ -{K_{22}}^{-1}K_{21}\overline{r}_{j1} \end{bmatrix}. \tag{6.28}$$

Comparing Equation (6.19) to Equation (6.26), Equation (6.20) to Equation (6.27), and Equation (6.21) to Equation (6.28), it is obvious that $r_0 = \overline{r}_{01}$, $r_1 = \overline{r}_{11}$, and $r_j = \overline{r}_{j1}$, which shows that in Equation (6.17) $V_1 = V$ and $V_2 = -{K_{22}}^{-1}K_{21}V_1$.

□

Finally, it is obvious that the projection of the alternative realization (6.16) onto a second-order input Krylov subspace effectively yields the same ROM as obtained through *MOR after Schur* whereas the sparse structure of the stiffness matrix K is unchanged.

Theorem 6.2. *'MOR after Schur' and 'MOR after implicit Schur' yield the same ROMs.*

Proof. The proof can be conducted by applying the projection matrix $\overline{V}$ as defined by Lemma 6.1 to the system matrices in Equation (6.16) as follows:

$$M_r = \overline{V}^T M \, \overline{V} = {V_1}^T M_{11} V_1 \,, \tag{6.29}$$

$$E_r = \overline{V}^T E \, \overline{V} = {V_1}^T E_{11} V_1 \,, \tag{6.30}$$

$$K_r = \overline{V}^T K \, \overline{V} = \underbrace{{V_1}^T K_{11} V_1 + {V_1}^T K_{12} V_2}_{{V_1}^T K_s V_1} + \underbrace{{V_2}^T K_{21} V_1 + {V_2}^T K_{22} V_2}_{=0} \,, \tag{6.31}$$

$$\widetilde{B}_r = \overline{V}^T \widetilde{B} = {V_1}^T B_s \,, \tag{6.32}$$

$$C_r = C \, \overline{V} = C_s V_1 \,. \tag{6.33}$$

By Lemma 6.1, $V = V_1$, which leads to the same reduced model as shown in Equation (6.6).

□

6.2.5 Comparison of Stable MOR Approaches

In this section, both the micro-fabricated and the frequency-tunable energy harvester models introduced in section 5.1 are considered to verify the proposed method *MOR after implicit Schur*. Furthermore, all these stable MOR approaches reviewed in sections 6.2.1 - 6.2.3 are subsequently applied to the piezoelectric energy harvester models to stabilize the reduced models. A comparison of the computational time of these four stabilization approaches is given in this section.

As described in section 6.1, both energy harvester devices are excited by a displacement input (**displ**). In both mechanical and electrical domains, displacement outputs at the selected nodes (**centre**, **south**, and **north**; **inner** and **outer**) and the voltage outputs from the electrical ports (**el1** and **el2**) are observed to verify the stability of the system. The projection matrix V is constructed through the SOAR method at the expansion point $s_0 = 0$, which indicates a good approximation at low frequencies and steady state. The respective frequency responses of the voltage outputs from the electrical ports **el1** and **el2** are further compared as shown in Figure 28, which indicate that the ROMs obtained through these stable MOR approaches are accurate.

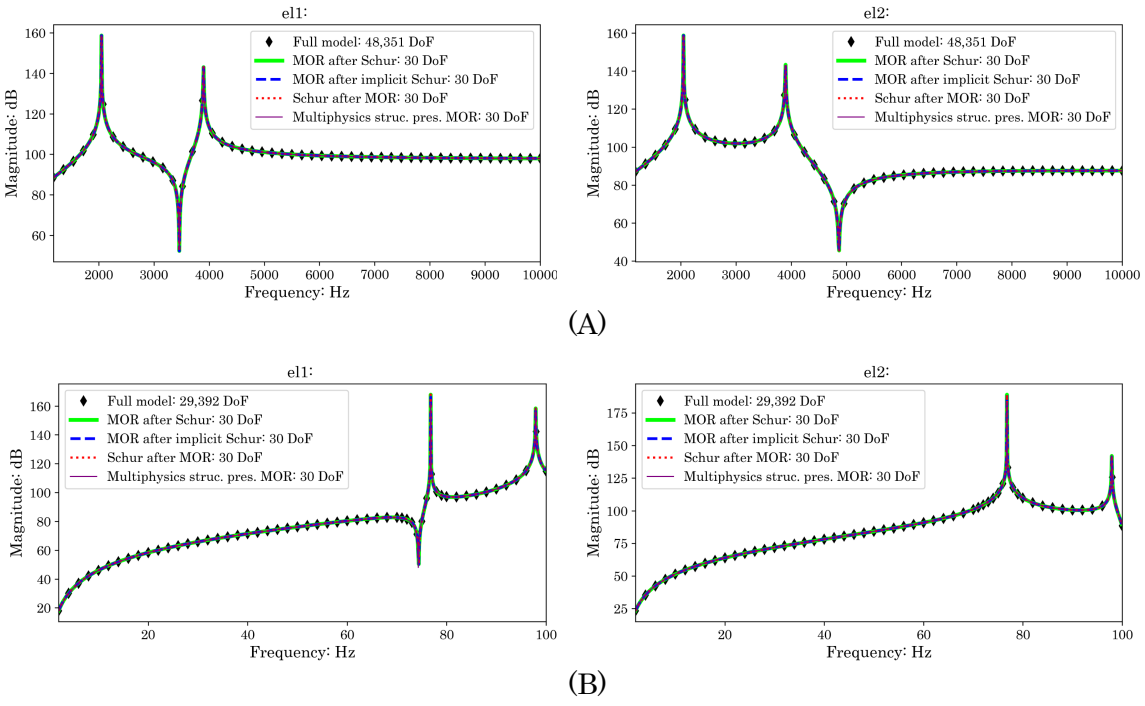


Figure 28: Frequency responses of the voltage outputs from the ports **el1** and **el2** in the (A) micro-fabricated and (B) frequency-tunable reduced models. These figures are adapted from [59].

It is illustrated in Figure 28 (A) that the original full-order micro-fabricated model is reduced from 48,351 DOF to 30 DOF through *MOR after Schur* and *MOR after implicit Schur* methods. With *Schur after MOR* method, the reduced model obtained via SOAR

(32 DOF) is first projected onto the eigenspace of the reduced stiffness matrix K_r. The indices of two negative eigenvalues in matrix K_r's diagonal eigenvalue matrix are found and used to partition all the other system matrices, which are already transformed based on the eigenspace of matrix K_r. After that, the ROM has a matrix structure analogous to the original system and the Schur transformation is applied. The final dimension of the ROM obtained through *Schur after MOR* is then 30 DOF. In the *multiphysics structure-preserving MOR* method, the dimension of the reduced model is $r = 30$ DOF. As known from Equation (6.14), the projection matrix V is separated into mechanical and electrical parts and reconstructed as $\widetilde{V}$. Owing to the fact that the number of mechanical DOF is much larger than the number of electrical DOF in the state vector, fewer Arnoldi basis vectors are required in the electrical projection matrix $\widetilde{V}_e$ compared to the mechanical projection matrix $\widetilde{V}_m$. According to the experimental results, $r_1 = 28$ and $r_2 = 2$ are sufficient for obtaining an accurate reduced model. The total number of columns in the projection matrix $\widetilde{V}$ remains as $r = r_1 + r_2 = 30$ DOF for comparison to other methods. Figure 28 (B) shows the comparison of the frequency responses of **el1** and **el2** of the full (29,392 DOF) and reduced frequency-tunable models. Both cases show that these four MOR approaches stabilize the reduced models of the piezoelectric energy harvesters.

Furthermore, the computational times of these four stabilization approaches are shown in Table 3. It is observed that *MOR after Schur* requires the highest computational effort. This is because the Schur transformation of the full-scale system changes stiffness matrix K's original sparse structure and increases its number of nonzero elements significantly, as shown in Figure 27. Consequently, the computational time for processing the matrix files is longer. In comparison, the other three methods incur a low cost in processing matrices. *MOR after implicit Schur* avoids the Schur transformation in the full-order stiffness matrix and preserves its sparse structure. *Schur after MOR* performs the Schur transformation on the reduced system matrices and *multiphysics structure-preserving MOR* preserves the block structure of the original full system matrices during the MOR.

Table 3: Comparison of the computational time of stable MOR approaches applied to the micro-fabricated piezoelectric energy harvester model with full-order dimension 48,351 DOF (Intel Core(TM) i5-7600 3.5 GHz, 32 GB RAM).

Method	**Step**	**Time (s)**	**Total (s)**
MOR after Schur	Schur transformation	73.3	402.3
	MOR	329	
MOR after implicit Schur	Implicit Schur transformation	25.8	74.4
	MOR	48.6	
Schur after MOR	MOR	48.2	48.7
	Quasi-Schur transformation	0.4	
Multiphysics structure-preserving MOR	Partitioning projection matrix	25.3	93.1
	MOR	67.8	

The most efficient method for stable compact modeling is *Schur after MOR*, which takes only 0.4 s for quasi-Schur transformation based on these reduced system matrices. *MOR after implicit Schur* and *multiphysics structure-preserving MOR* methods are efficient as well, but still expend some computational effort in transforming a full-order input matrix and generating the partitioned projection matrix.

6.3 Stable Parametric Model Order Reduction

In the design process, the impact of the thickness of piezoelectric patches on power output is important. pMOR bears the potential to construct a pROM, which preserves this parameter in symbolic form. For this purpose, the authors in [45] suggested an MI-based pMOR, which is reviewed in section 3.2. The basic idea of this method is reducing several local FOMs with fixed values of the geometrical parameter in the parameter space and calculating the pROMs in-between by interpolating the system matrices of the local ROMs. For this reason, the pROMs generated are only accurate in a limited parameter space. In recent years, the MI-based pMOR method has proven successful for single-physical domain models [45, 85, 86]. However, similarly to the non-parametric ROMs described in section 6.1, pROMs may become unstable when multi-physical domains are involved. Unstable local ROMs will lead to unstable pROMs after the interpolation. Therefore, in this section, a framework will be presented where these four stabilization approaches, as was suggested in the previous section 6.2, are combined with the MI-based pMOR method for ensuring a stable pROM of the piezoelectric energy harvester model.

6.3.1 MOR after (Implicit) Schur Combined with MI-based pMOR

In order to obtain a stable pROM, the stable MOR methods *MOR after (implicit) Schur* introduced in sections 6.2.1 and 6.2.4 are combined with the MI-based pMOR method. As shown in Algorithm 4 and Figure 29, the basic idea is to use *MOR after (implicit) Schur* methods to generate the stable local ROMs first and then do the interpolation. This method will be referred as '*MOR after (implicit) Schur* + MI'.

Algorithm 4 *MOR after (implicit) Schur* + MI

Input: local full systems $\Sigma = \{\Sigma_1, \Sigma_2, ..., \Sigma_k\}$, $\Sigma_i = \{M_i, E_i, K_i, B_i, C_i\}$, $i = 1, 2, ..., k$.

1: **for** $i = 1, 2, ..., k$ **do**

2: Transform full system Σ_i into $\Sigma_{schur,i}$ or $\Sigma_{implicit,i}$ through Schur/implicit Schur transformation.

3: Construct the local projection matrix V_i and the local ROM $\Sigma_{i,r}$.

4: Compute the transformation matrices $\mathbf{T}_i$, $\mathbf{M}_i$ and transform the local ROM $\Sigma^*_{i,r}$.

5: **end for**

6: Interpolating the local ROMs $\Sigma^*_r = \sum_{i=1}^{k} w_i \Sigma^*_{i,r}$.

Output: stable pROM Σ^*_r.

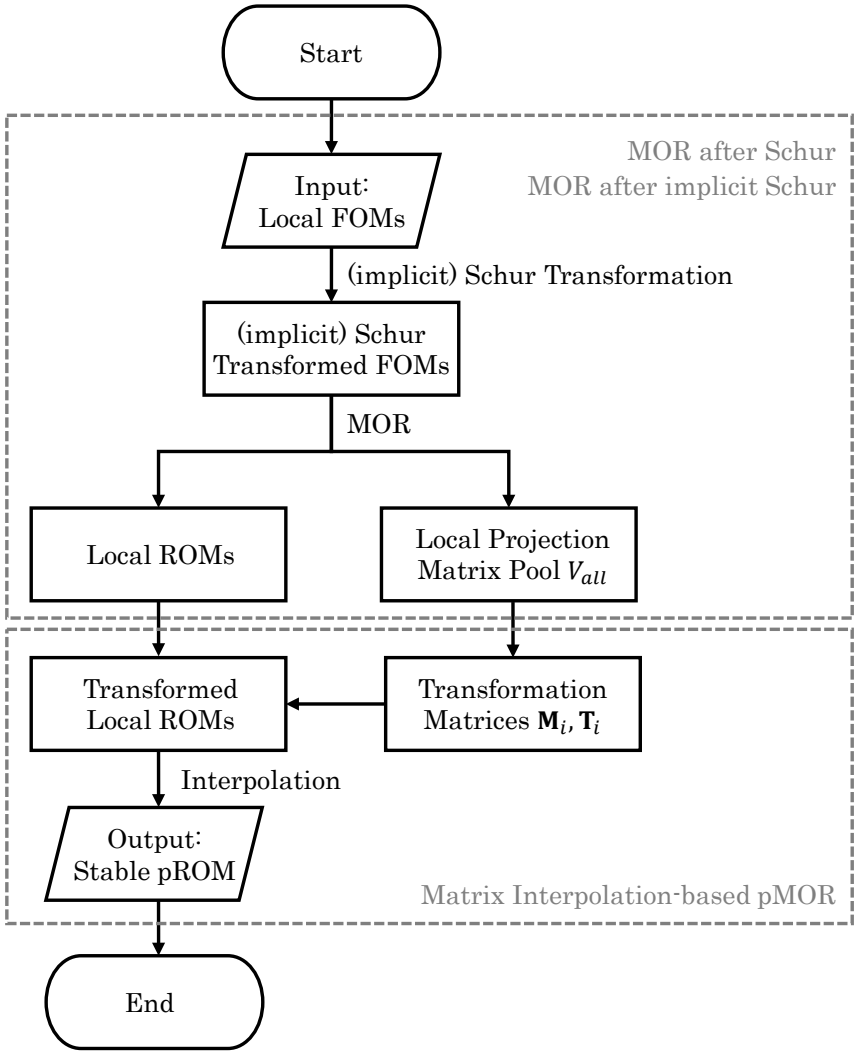


Figure 29: The workflow chart of combining *MOR after (implicit) Schur* with MI-based pMOR.

6.3.2 Schur after MOR Combined with MI-based pMOR

In this case, the method *Schur after MOR* is combined with the MI-based pMOR method. The algorithm is present in Algorithm 5 and defined as '*Schur after MOR* + MI'. As illustrated in the workflow chart in Figure 30, the Schur complement is applied on the pROM, which is obtained through MI-based pMOR.

Algorithm 5 *Schur after MOR* + MI

Input: local full systems $\Sigma = \{\Sigma_1, \Sigma_2, ..., \Sigma_k\}$, $\Sigma_i = \{M_i, E_i, K_i, B_i, C_i\}$, $i = 1, 2, ..., k$.

1: **for** $i = 1, 2, ..., k$ **do**
2: Construct the local projection matrix V_i and the local ROM $\Sigma_{i,r}$.
3: Compute the transformation matrices $\mathbf{T}_i$, $\mathbf{M}_i$ and transform the local ROM $\Sigma^*_{i,r}$.
4: **end for**
5: Interpolating the local ROMs $\Sigma^*_r = \sum_{i=1}^{k} w_i \Sigma^*_{i,r}$.
6: Apply Schur transformation on the interpolated pROM Σ^*_r.

Output: stable pROM Σ^*_r.

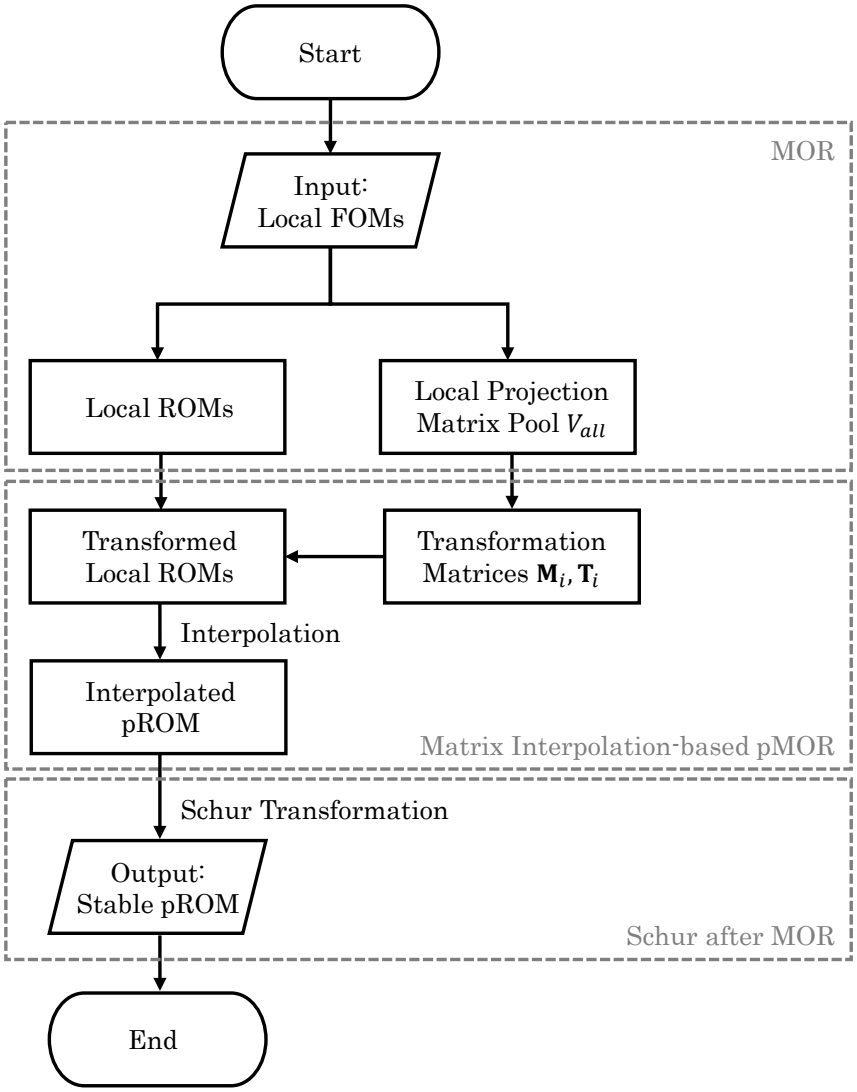


Figure 30: The workflow chart of combining *Schur after MOR* with MI-based pMOR.

6.3.3 Multiphysics Structure-Preserving MOR with MI-based pMOR

In this case, the stable local ROMs are generated by using the method *multiphysics structure-preserving MOR* introduced in section 6.2.3. The basic idea of this method is partitioning the local projection matrices V_i into mechanical and electrical parts

$$V_i = \begin{bmatrix} V_{i,m} \\ V_{i,e} \end{bmatrix} \rightarrow \widetilde{V}_i = \begin{bmatrix} \widetilde{V}_{i,m} & 0 \\ 0 & \widetilde{V}_{i,e} \end{bmatrix} , \tag{6.34}$$

where $V_i \in \mathbb{R}^{N \times r}$ are the local projection matrices constructed when generating the local ROMs. $\widetilde{V}_{i,m} \in \mathbb{R}^{k_1 \times r_1}$, $\widetilde{V}_{i,e} \in \mathbb{R}^{k_2 \times r_2}$ are partitioned according to the block sizes of the system matrices in Equation (5.2). In this case study, $r_1 = r_2 = r$ is chosen and all the

local models are then reduced blockwise as follows:

$$\widetilde{\Sigma}_{i,r}\begin{cases}\underbrace{\begin{bmatrix}\widetilde{V}_{i,m}^T M_{i,11}\widetilde{V}_{i,m} & 0\\ 0 & 0\end{bmatrix}}_{\widetilde{M}_{i,r}}\underbrace{\begin{bmatrix}\ddot{d}_{i,r}\\ \ddot{v}_{i,r}\end{bmatrix}}_{\ddot{x}_{i,r}}+\underbrace{\begin{bmatrix}\widetilde{V}_{i,m}^T E_{i,11}\widetilde{V}_{i,m} & 0\\ 0 & 0\end{bmatrix}}_{\widetilde{E}_{i,r}}\underbrace{\begin{bmatrix}\dot{d}_{i,r}\\ \dot{v}_{i,r}\end{bmatrix}}_{\dot{x}_{i,r}}\\ +\underbrace{\begin{bmatrix}\widetilde{V}_{i,m}^T K_{i,11}\widetilde{V}_{i,m} & \widetilde{V}_{i,m}^T K_{i,12}\widetilde{V}_{i,e}\\ \widetilde{V}_{i,e}^T K_{i,21}\widetilde{V}_{i,m} & \widetilde{V}_{i,e}^T K_{i,22}\widetilde{V}_{i,e}\end{bmatrix}}_{\widetilde{K}_{i,r}}\underbrace{\begin{bmatrix}d_{i,r}\\ v_{i,r}\end{bmatrix}}_{x_{i,r}}=\underbrace{\begin{bmatrix}\widetilde{V}_{i,m}^T B_{i,1}\\ \widetilde{V}_{i,e}^T B_{i,2}\end{bmatrix}}_{\widetilde{B}_{i,r}}u\\ y=\underbrace{\begin{bmatrix}C_{i,1}\widetilde{V}_{i,m} & C_{i,2}\widetilde{V}_{i,e}\end{bmatrix}}_{\widetilde{C}_{i,r}}\underbrace{\begin{bmatrix}d_{i,r}\\ v_{i,r}\end{bmatrix}}_{x_{i,r}}\end{cases}. \tag{6.35}$$

Afterwards, the transformation matrices $\mathbf{T}_i, \mathbf{M}_i \in \mathbb{R}^{r\times r}$ are constructed, based on the local projection matrices $V_i \in \mathbb{R}^{N\times r}$ via Equations (3.22) and (3.23).

The dimensions of the reduced matrices in Equation (6.35) are $\widetilde{M}_{i,r}, \widetilde{E}_{i,r}, \widetilde{K}_{i,r} \in \mathbb{R}^{2r\times 2r}$, $\widetilde{B}_{i,r} \in \mathbb{R}^{2r\times m}$ and $\widetilde{C}_{i,r} \in \mathbb{R}^{p\times 2r}$. The blockwise transformation matrices $\widetilde{\mathbf{T}}_i^{-1}, \widetilde{\mathbf{M}}_i \in \mathbb{R}^{2r\times 2r}$ are then constructed as follows:

$$\widetilde{\mathbf{T}}_i^{-1}=\begin{bmatrix}\mathbf{T}_i^{-1} & 0\\ 0 & \mathbf{T}_i^{-1}\end{bmatrix},\quad \widetilde{\mathbf{M}}_i=\begin{bmatrix}\mathbf{M}_i & 0\\ 0 & \mathbf{M}_i\end{bmatrix}. \tag{6.36}$$

The local ROMs in Equation (6.35) are then transformed blockwise with Equation (6.36)

$$\widetilde{\Sigma}_{i,r}^*\begin{cases}\underbrace{\widetilde{\mathbf{M}}_i\begin{bmatrix}\widetilde{M}_{i,r,11} & 0\\ 0 & 0\end{bmatrix}\widetilde{\mathbf{T}}_i^{-1}\begin{bmatrix}\ddot{d}_{i,r}^*\\ \ddot{v}_{i,r}^*\end{bmatrix}}_{\widetilde{M}_{i,r}^*\ddot{x}_{i,r}^*}+\underbrace{\widetilde{\mathbf{M}}_i\begin{bmatrix}\widetilde{E}_{i,r,11} & 0\\ 0 & 0\end{bmatrix}\widetilde{\mathbf{T}}_i^{-1}\begin{bmatrix}\dot{d}_{i,r}^*\\ \dot{v}_{i,r}^*\end{bmatrix}}_{\widetilde{E}_{i,r}^*\dot{x}_{i,r}^*}\\ \underbrace{+\widetilde{\mathbf{M}}_i\begin{bmatrix}\widetilde{K}_{i,r,11} & \widetilde{K}_{i,r,12}\\ \widetilde{K}_{i,r,21} & \widetilde{K}_{i,r,22}\end{bmatrix}\widetilde{\mathbf{T}}_i^{-1}\begin{bmatrix}d_{i,r}^*\\ v_{i,r}^*\end{bmatrix}}_{\widetilde{K}_{i,r}^* x_{i,r}^*}=\underbrace{\widetilde{\mathbf{M}}_i\begin{bmatrix}\widetilde{B}_{i,r,1}\\ \widetilde{B}_{i,r,2}\end{bmatrix}}_{\widetilde{B}_{i,r}^*}u\\ y=\underbrace{\begin{bmatrix}\widetilde{C}_{i,r,1} & \widetilde{C}_{i,r,2}\end{bmatrix}\widetilde{\mathbf{T}}_i^{-1}}_{\widetilde{C}_{i,r}^*}\underbrace{\begin{bmatrix}d_{i,r}^*\\ v_{i,r}^*\end{bmatrix}}_{x_{i,r}^*}\end{cases}. \tag{6.37}$$

It should be noted here that in the *multiphysics structure-preserving MOR* method, although the local projection matrices are partitioned into mechanical and electrical parts (see $\widetilde{V}_i$ in Equation (6.34)), the blockwise transformation matrices $\widetilde{\mathbf{T}}_i^{-1}, \widetilde{\mathbf{M}}_i$ are still constructed based on the transformation matrices $\mathbf{T}_i$ and $\mathbf{M}_i$, which are generated by using the local projection matrices V_i but not the partitioned local projection matrices $\widetilde{V}_{i,m}, \widetilde{V}_{i,e}$.

This could be explained if the blockwise transformation matrices $\widetilde{\mathbf{T}}_i^{-1}, \widetilde{\mathbf{M}}_i$ are constructed as follows:

$$\widetilde{\mathbf{T}}_i^{-1} = \begin{bmatrix} \mathbf{T}_{i,m}^{-1} & 0 \\ 0 & \mathbf{T}_{i,e}^{-1} \end{bmatrix}, \quad \widetilde{\mathbf{M}}_i = \begin{bmatrix} \mathbf{M}_{i,m} & 0 \\ 0 & \mathbf{M}_{i,e} \end{bmatrix}. \tag{6.38}$$

where the transformation matrices $\mathbf{T}_{i,m}$, $\mathbf{T}_{i,e}$, and $\mathbf{M}_{i,m}$, $\mathbf{M}_{i,e}$ are constructed through Equations (3.22) and (3.23) on the basis of the partitioned local projection matrices $V_{i,m}$ and $V_{i,e}$. Then the transformed local ROMs in Equation (6.37) can be rewritten as follows:

$$\widetilde{\Sigma}_{i,r}^* \begin{cases} \underbrace{\begin{bmatrix} \mathbf{M}_{i,m} M_{i,r,11} \mathbf{T}_{i,m}^{-1} & 0 \\ 0 & 0 \end{bmatrix}}_{\widetilde{M}_{i,r}^*} \underbrace{\begin{bmatrix} \ddot{d}_{i,r} \\ \ddot{v}_{i,r} \end{bmatrix}}_{\ddot{x}_{i,r}^*} + \underbrace{\begin{bmatrix} \mathbf{M}_{i,m} E_{i,r,11} \mathbf{T}_{i,m}^{-1} & 0 \\ 0 & 0 \end{bmatrix}}_{\widetilde{E}_{i,r}^*} \underbrace{\begin{bmatrix} \dot{d}_{i,r} \\ \dot{v}_{i,r} \end{bmatrix}}_{\dot{x}_{i,r}^*} \\ + \underbrace{\begin{bmatrix} \mathbf{M}_{i,m} K_{i,r,11} \mathbf{T}_{i,m}^{-1} & \boxed{\mathbf{M}_{i,m} K_{i,r,12} \mathbf{T}_{i,e}^{-1}} \\ \boxed{\mathbf{M}_{i,e} K_{i,r,21} \mathbf{T}_{i,m}^{-1}} & \mathbf{M}_{i,e} K_{i,r,22} \mathbf{T}_{i,e}^{-1} \end{bmatrix}}_{\widetilde{K}_{i,r}^*} \underbrace{\begin{bmatrix} d_{i,r} \\ v_{i,r} \end{bmatrix}}_{x_{i,r}^*} = \underbrace{\begin{bmatrix} \mathbf{M}_{i,m} B_{i,r,1} \\ \mathbf{M}_{i,e} B_{i,r,2} \end{bmatrix}}_{\widetilde{B}_{i,r}^*} u \\ y = \underbrace{\begin{bmatrix} C_{i,r,1} \mathbf{T}_{i,m} & C_{i,r,2} \mathbf{T}_{i,e} \end{bmatrix}}_{\widetilde{C}_{i,r}^*} \underbrace{\begin{bmatrix} d_{i,r} \\ v_{i,r} \end{bmatrix}}_{x_{i,r}^*} \end{cases}. \tag{6.39}$$

It is obvious that in the boxed terms in Equation (6.39), the coupling matrices $K_{i,r,12}$ and $K_{i,r,21}$ are transformed into different bases and cannot be used for interpolation. Therefore, the interpolation method can only be deployed based on the transformed local ROMs in Equation (6.37), where all the submatrices are transformed into a generalized basis. This algorithm is named as '*multiphysics structure-preserving MOR* + MI' as shown in Algorithm 6 and the workflow chart is presented in Figure 31.

Algorithm 6 *multiphysics structure-preserving MOR* + MI

Input: local full systems $\Sigma = \{\Sigma_1, \Sigma_2, ..., \Sigma_k\}$, $\Sigma_i = \{M_i, E_i, K_i, B_i, C_i\}$, $i = 1, 2, ..., k$.

1: **for** $i = 1, 2, ..., k$ **do**
2: Partition full system matrices according to different physical domains.
3: Construct the local projection matrix V_i.
4: Construct the blockwise local projection matrix $\widetilde{V}_i = \begin{bmatrix} \widetilde{V}_{i,m} & 0 \\ 0 & \widetilde{V}_{i,e} \end{bmatrix}$.
5: Generate the local ROM $\Sigma_{i,r}$ through blockwise reduction by using $\widetilde{V}_i$.
6: Construct the blockwise transformation matrices $\widetilde{\mathbf{T}}_i^{-1}$, $\widetilde{\mathbf{M}}_i$ on the basis of the local projection matrix V_i and transform the local ROM $\Sigma_{i,r}^*$.
7: **end for**
8: Interpolating the local ROMs $\Sigma_r^* = \sum_{i=1}^k w_i \Sigma_{i,r}^*$.

Output: stable pROM Σ_r^*

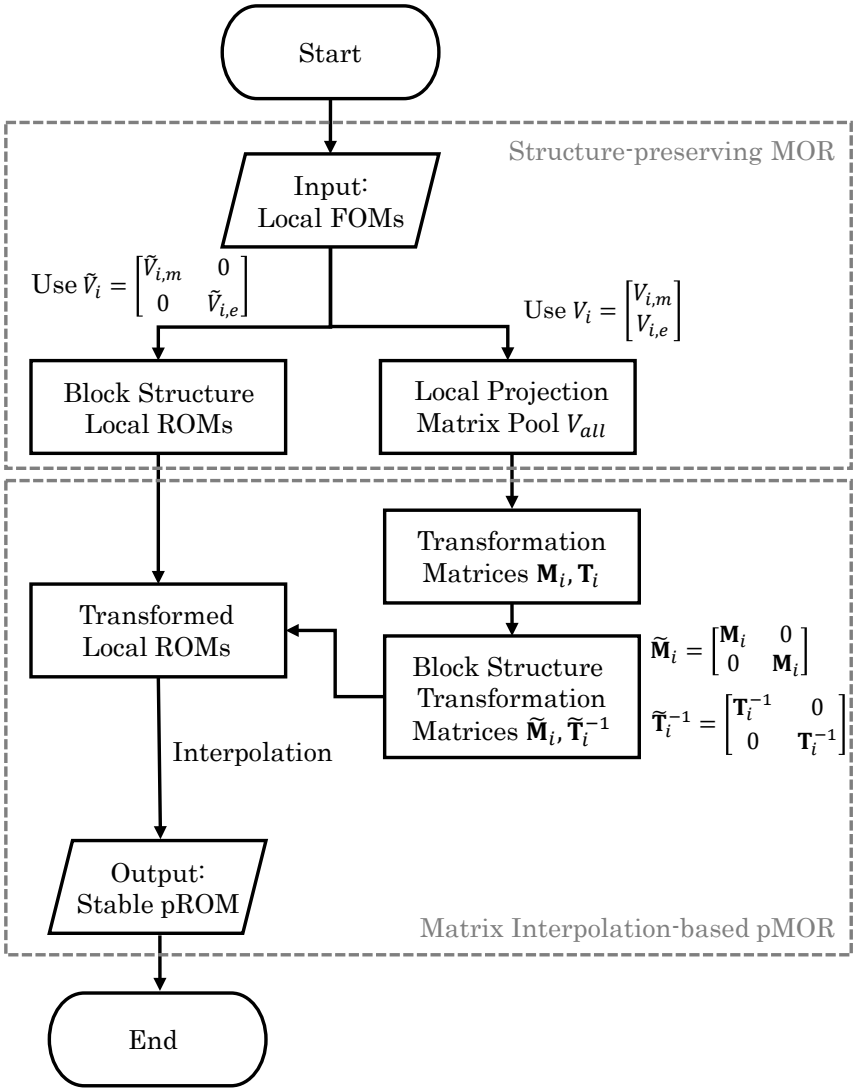


Figure 31: The workflow chart of combining *multiphysics structure-preserving MOR* with MI-based pMOR.

6.3.4 Comparison of Stable pMOR Approaches

In this section, the simple piezoelectric beam model and the frequency-tunable piezoelectric energy harvester model introduced in section 5.1 are both deployed to validate the stable pMOR algorithms presented in section 6.3. In these devices, the thickness of the piezoelectric patches is set as a geometrical parameter, which is defined as l. In order to retain the mesh topology, the parameter l is varied by using the ANSYS parametric design language (APDL) command '`NMODIF`' to shift the nodes on the top surfaces of the piezoelectric patches. Two local models are sampled at $l = 0.29$ mm and $l = 0.31$ mm. The parametric ROM is evaluated at $l = 0.3$ mm via combining stable MOR methods with MI-based pMOR.

For the single piezoelectric beam model, the projection matrices of the local models are constructed with one expansion point $s_0 = 2\pi f_0$ selected near the peak as shown in

Figure 32, where $f_0 = 147.6$ Hz. The full model with 14,466 DOF is reduced to 30 DOF in the pROMs, which are obtained from '*MOR after (implicit) Schur* + MI' methods. In the '*multiphysics structure-preserving MOR* + MI' method, the dimension of the reduced model is also 30 DOF due to the fact that the number of columns generated in the local projection matrices $\widetilde{V}_{i,m}$ and $\widetilde{V}_{i,e}$ is set as 15 for each physical domain. For '*Schur after MOR* + MI', the dimension of the interpolated pROM is 29 DOF due to the fact that the Schur complement has been applied on the reduced model. It is observed in Figure 32 that both displacement and voltage outputs calculated from the interpolated pROMs at 0.3 mm matches the reference results calculated from the FOM.

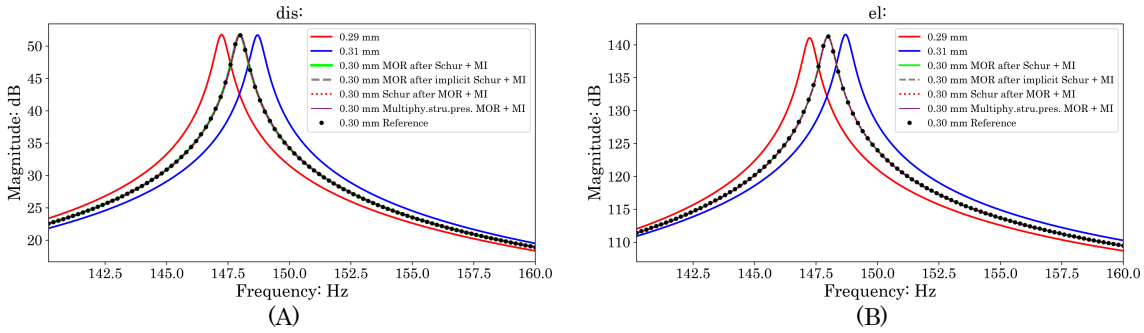


Figure 32: (A) Comparison of the harmonic response of directional displacement output **dis** from the mass tip and (B) comparison of the harmonic response of electrical port **el1** from the piezoelectric patch to the displacement excitation **displ** computed from the pROMs constructed via four different stable pMOR methods. Red and blue curves are the results from the local FOMs when $l = \{0.29, 0.31\}$ mm. The black dotted curve is the reference result obtained from the FOM when $l = 0.3$ mm.

In the frequency-tunable piezoelectric energy harvester model, which has a more complicated geometry compared to the single beam model, it is found that the accuracy of the interpolated ROM is highly sensitive to the selection of the expansion points used for the reduction of the local models. In this case study, multiple expansion points are selected according to the frequencies of the peaks and valleys as shown in Figure 33, Figure 34, and Figure 35. Twelve expansion points $s_0 = 2\pi f_0$, $f_0 = \{76, 76.4, 77, 78.6, 79.2, 79.6, 81.2, 81.4, 81.6, 81.8, 82, 82.2\}$ Hz are utilized to calculate the projection matrices. Five vectors are generated around each expansion point.

The full-scale piezoelectric energy harvester model is built with 9,054 DOF. The pROMs with 180 DOF are obtained through methods '*MOR after Schur* + MI' and '*MOR after implicit Schur* + MI'. For '*Schur after MOR* + MI', the dimension of the interpolated pROM is 178 DOF due to the fact that the Schur complement has been applied to the ROM. For '*multiphysics structure-preserving MOR* + MI', it is special that the dimension of the projection matrices $\widetilde{V}_{i,m}$ and $\widetilde{V}_{i,e}$ in Equation (6.14) has to be the same (in this case $r_1 = r_2 = r$), as singular value decomposition (SVD) is performed on the matrix pool of local projection matrices V_i as presented in Equation (3.24). In addition, each local projection matrix V_i is generated by orthogonalizing the second-order

Krylov subspace, which uses b_1 from the input matrix B as the starting vector. The final dimension of the interpolated ROM is 120 DOF.

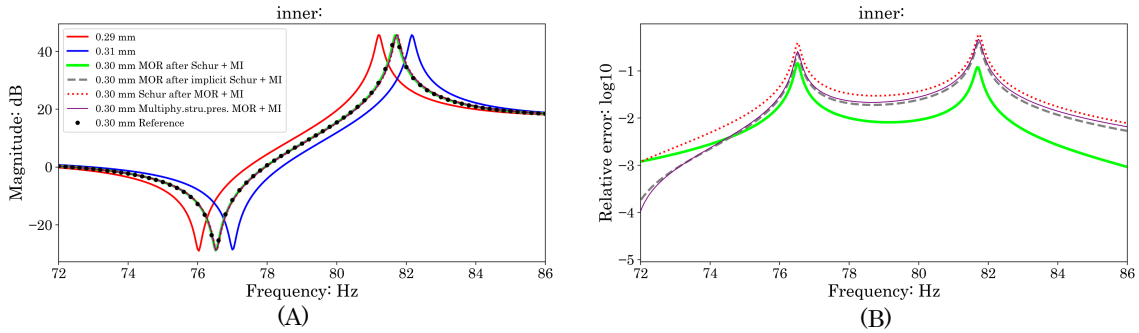


Figure 33: (A) Comparison of the harmonic response of directional displacement output from the **inner** mass tip to the displacement excitation **displ**. These results are computed from the pROMs generated via four different stable pMOR methods. (B) Relative errors are calculated between full and reduced models. These figures are adapted from [87].

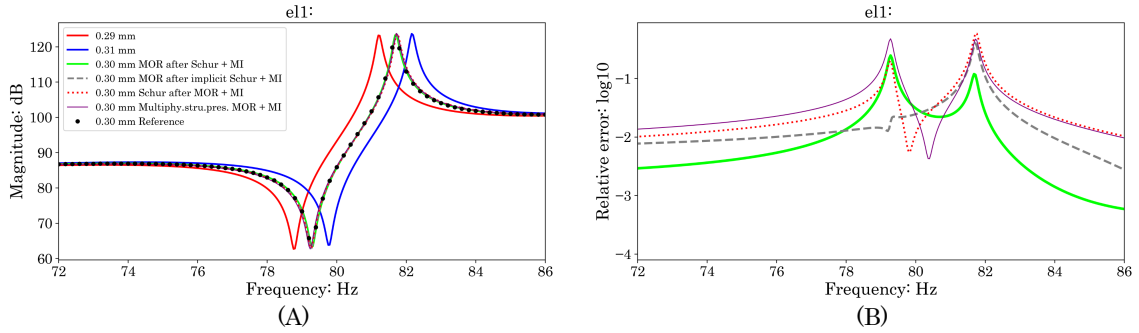


Figure 34: (A) Comparison of harmonic response of voltage from the electrical port **el1** to the displacement excitation **displ**. (B) Relative errors are calculated between full and reduced models. These figures are adapted from [87].

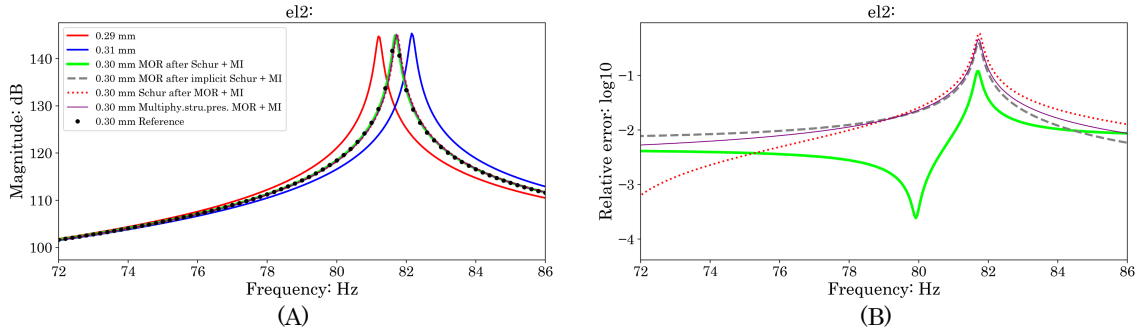


Figure 35: (A) Comparison of harmonic response of voltage from the electrical port **el2** to the displacement excitation **displ**. (B) Relative errors are calculated between full and reduced models. These figures are adapted from [87].

The respective frequency responses of displacement and voltage outputs from **inner**, **el1**, and **el2** are compared as shown in Fig. 33, Fig. 34, and Fig. 35. It is observed that the results from the interpolated pROMs match well with the result obtained from the FOM at $l = 0.3$ mm. Relatively large error occurs at the points near the peaks and valleys of the frequency responses.

6.4 Device-Circuit Co-Simulation at System Level

As is mentioned in section 5.1, the piezoelectric energy harvester transforms environmental vibrations into electrical energy. As this is the case, it can be used as a power supply for electrical devices. An electrical circuit is an inherent part of any energy harvesting system. The modeling of such systems is supported by MOR, as this technique provides compact but accurate ROMs from FE models.

In this section, device-circuit co-simulation using the simple bridge rectifier circuit and the ROM of the frequency-tunable piezoelectric energy harvester (see Figure 36) is demonstrated. The ROM is obtained via the method *MOR after implicit Schur* and the system simulation circuit is employed in the software ANSYS Twin Builder. A sinusoidal displacement excitation is applied to the input **displ**. The displacement amplitude is 4.3 µm at a frequency of 76.8 Hz. This signal represents the environmental vibration. One electrical port **el2** is connected to a bridge rectifier, with the other port **el1** being shunted by a resistor. A storage capacitor of 10 µF collects the electrical energy while a load resistor of 100 kΩ is connected in parallel to illustrate power dissipation over time.

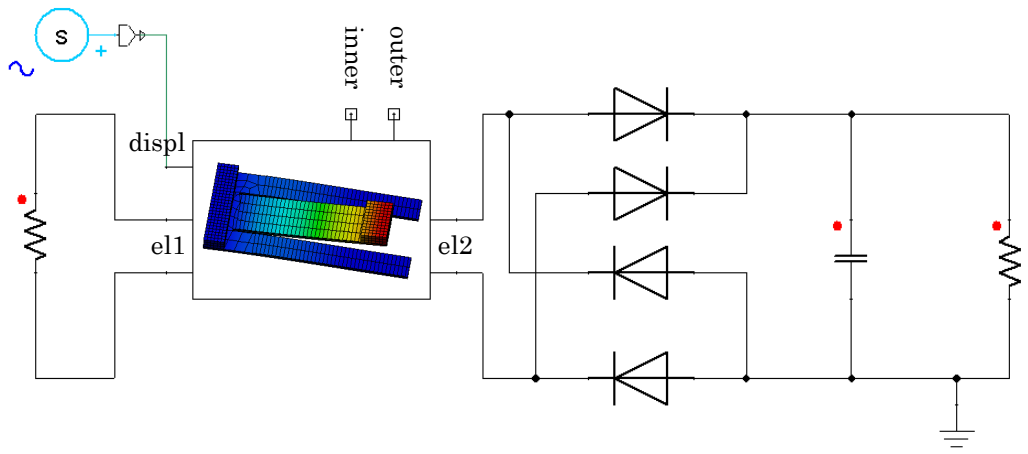


Figure 36: ROM connected to a shunt resistor (port **el1**) and bridge rectification circuit with energy storage capacitor and load resistor (port **el2**). This figure is adapted from [58].

The voltage across the load resistor is shown in Figure 37. This voltage output is rectified to a constant value of 1.5 V. This system-level simulation technique is also capable of modeling more sophisticated power management circuits with the ROM.

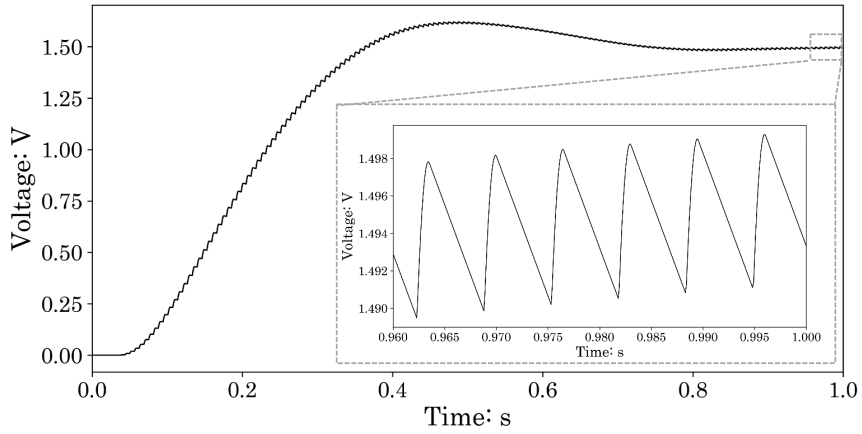


Figure 37: Voltage on the load resistor. The inner plot shows the voltage between 0.96 and 1 s. This figure is adapted from [58].

Chapter 6 Summary

This chapter presents the instability of the ROMs generated from multiphysics piezoelectric energy harvester models. Unstable transient responses are observed at the electrical ports of the ROMs. As a consequence, three existing stable MOR approaches, which are designed based on the block structure of the piezoelectric system matrices, are reviewed.

- *MOR after Schur* method performs the Schur transformation on the full system matrices first and then applies the MOR. The disadvantage of this method is that the number of nonzero elements in the stiffness matrix K is significantly increased after the Schur transformation.

- *Schur after MOR* method introduces a way to perform the Schur transformation on the ROM instead. Compared to other stable MOR methods, this is the most efficient method of constructing a stable piezoelectric ROM.

- *Multiphysics structure-preserving MOR* preserves the block structure of the original piezoelectric system in the ROM via constructing separate projection matrices for mechanical and electrical domains. The Schur transformation is actually performed implicitly during the MOR process.

Finally, the *MOR after implicit Schur* method is suggested in this work. This new method applies the Schur transformation to the input matrix only and preserves the original sparse stiffness matrix K. The scripts developed for these four stable MOR approaches are presented in Appendix F.

Furthermore, the combination of stable MOR approaches with the MI-based pMOR method is proposed. The algebraic parameterization-based pMOR method introduced in section 3.3 is inapplicable in this case because it changes the original structure of the stiffness matrix.

7 Parametric Model Order Reduction of Thermoelectric Energy Harvester

This chapter presents the application of parametric model order reduction (pMOR) methods on a miniaturized thermoelectric generator (TEG). A parametric reduced-order model (pROM) is constructed for efficient parametric studies. Sections 7.1 and 7.4 evaluate the impact of environmental conditions on the TEG with film coefficient and ambient temperature as the parameters in the convection boundary condition (BC). In section 7.2, an optimization strategy is proposed for TEG based on replacing the detailed model with a surrogate model. In this way, the detailed geometrical array configuration of the thermocouples is modified by parameterizing the equivalent thermal conductivity of the surrogate model. In these two case studies, multivariate moment-matching-based pMOR methods can be implemented to generate the pROM. For the other geometrical parameters, e.g., the height of the thermocouples, matrix-interpolation (MI)- and algebraic parameterization (AP)-based pMOR methods are applied and adapted for pROM generation.

7.1 Convection Boundary Condition Parametrization

For efficient simulation of a linear thermal model, the model order reduction (MOR) methods introduced in section 2.3 can be implemented to generate an accurate reduced-order model (ROM). However, in the case study where the environmental conditions are changed, the MOR process has to be repeated to generate a new ROM with updated BCs. Therefore, generating a ROM that preserves the parameters in ROM-level is essential. According to the previous research [44, 88], it is possible to construct a pROM with parameterized film coefficient and ambient temperature in the convection BC through the multivariate moment matching-based pMOR method mentioned in section 3.1. The TEG model in Figure 38 is chosen as the case study in this section.

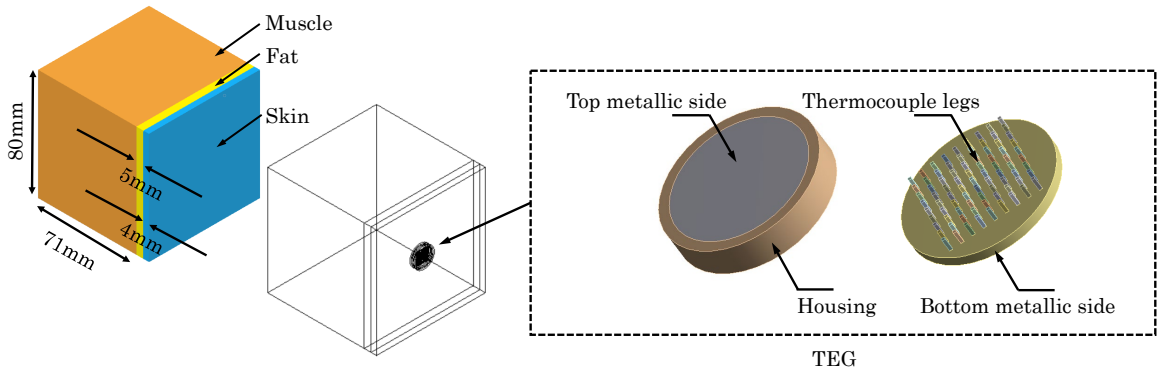


Figure 38: A cylindrical TEG incorporated in simplified human tissue model.

The TEG consists of two metallic discs (diameter 13 mm, height 0.9 mm) and an array of 9×9 thermocouple legs (height 1.2 mm) in between (thermal conductivity $\kappa = 1.35$ W/m/K, Seebeck coefficient $\alpha_{p,n} = \pm 200$ µV/K, electrical resistivity $\rho_{el} = 10$ µΩ·m). It is enclosed by a polymer housing with low thermal conductivity ($\kappa = 0.25$ W/m/K). A leg cross-section of 275×275 µm^2 results in maximum power delivery, as demonstrated in [15]. A cuboid human tissue with a dimension of 80 mm is composed of muscle, fat, and skin layers with thickness 71, 5, and 4 mm, respectively (as proposed in [64]). Thermal material properties of each tissue type are given in Table A5.

As mentioned in section 5.2, the heat transfer in human tissue can be described by Pennes' bioheat equation as presented in Equation (5.6), which considers heat conduction, metabolic heat generation, and blood perfusion. In this case, a constant metabolic heat generation rate is assumed across the muscle tissue[2]. Then in Equation (5.6), the governing heat-transfer partial differential equation is rewritten as follows:

$$\nabla(\kappa \nabla T) + Q - \rho c \frac{\partial T}{\partial t} = 0 \,, \tag{7.1}$$

where T is the temperature distribution of interest. ρ, c and κ are the density, specific heat capacity, and thermal conductivity properties of each tissue layer specified in Table A5. $Q = 800$ W/m^3 is a constant metabolic heat generation across the muscle tissue. The body core temperature $T = 37$ °C is accounted for through a Dirichlet BC at the back of the muscle layer. The heat removal from the skin surface is modeled by the convection BC as follows:

$$q_\perp = h \cdot (T(t) - T_{amb}) \,, \tag{7.2}$$

where $q_\perp$ is the heat flux normal to the boundary selected at the top skin surface. T_{amb} is the ambient temperature and h is the film coefficient in W/m^2/K. In this case study, the aim is to investigate the environmental influence on the temperature difference across the thermocouple legs by varying the film coefficient and ambient temperature. After the spatial discretization of the governing Equation (7.1) with convection BC in Equation (7.2), a parameterized ODE system similar to Equation (3.1) reads

$$\Sigma_N \begin{cases} E \cdot \dot{T}(t) = \underbrace{(A_0 + h \cdot A_1)}_{A(h)} \cdot T(t) + B \cdot \underbrace{\begin{bmatrix} Q \\ T_a \\ h \cdot T_{amb} \end{bmatrix}}_{u} \\ y(t) = C \cdot T(t) \end{cases} , \tag{7.3}$$

where $A_0, A_1 \in \mathbb{R}^{N \times N}$ are the parameter independent thermal conductivity matrices.

[2]This is the first TEG model implemented in this thesis for MOR. For this, the effect of perfusion, the heat sources in fat and skin layers are all ignored to simplify the model. Starting from the model in section 7.2, all these effects are considered.

Thus the two parameters h and T_{amb} from the convection BC. h is the film coefficient, which can be extracted in front of the thermal conductivity matrix. T_{amb} is the ambient temperature utilized as an input in the input vector.

As is explained in section 4.2, in order to extract the system matrices from ANSYS Mechanical, matrices A_0 and B are obtained when fixing $h = h_0$. The matrix A_1 is then calculated through the equation $A_1 = \frac{A(h_1)-A(h_0)}{h_1-h_0}$, where h_1 should be a different value compared to h_0. In addition, due to the fact that the ambient temperature input is multiplied by the film coefficient, the model in Equation (7.3) is then rewritten as follows:

$$\Sigma_N \begin{cases} E \cdot \dot{T}(t) = \underbrace{[A_0(h_0) + (h - h_0) \cdot A_1]}_{A(h)} \cdot T(t) + B(h_0) \cdot \underbrace{\begin{bmatrix} Q \\ T_a \\ \frac{h}{h_0} \cdot T_{amb} \end{bmatrix}}_{u} \\ y(t) = C \cdot T(t) \end{cases} . \tag{7.4}$$

According to the multivariate moment matching-based pMOR method described in section 2.4, a projection matrix $V \in \mathbb{R}^{N \times r}$ is obtained via merging two local projection matrices $V_s \in \mathbb{R}^{N \times r_1}$, $V_h \in \mathbb{R}^{N \times r_2}$ constructed based on each parameter s and h. Here, the pROM is obtained as follows:

$$\Sigma_r \begin{cases} \underbrace{V^T E V}_{E_r} \dot{z}(t) = \overbrace{[\underbrace{V^T A_0(h_0) V}_{A_{0r}} + (h - h_0) \cdot \underbrace{V^T A_1 V}_{A_{1r}}]}^{A_r(h)} z(t) + \underbrace{V^T B(h_0)}_{B_r} \cdot \underbrace{\begin{bmatrix} Q_m \\ T_a \\ \frac{h}{h_0} \cdot T_{amb} \end{bmatrix}}_{u} \\ y = \underbrace{CV}_{C_r} z(t) \end{cases} , \tag{7.5}$$

where $E_r, A_{0r}, A_{1r} \in \mathbb{R}^{r \times r}$, $B_r \in \mathbb{R}^{r \times m}$ and $C_r \in \mathbb{R}^{p \times r}$. The dimension of the pROM is $r = r_1 + r_2 \ll N$. $h_0 = 10$ W/m^2/K is the fixed film coefficient value used when extracting the system matrices from ANSYS Mechanical. The temperature results from the specific nodes selected on the top and bottom surfaces of the thermocouple legs are defined as the outputs. Differing from pMOR of a transient thermal system in [44], in this case study, observation of the static temperature distribution is of particular interest. The temperature difference across the thermocouple legs for different different values of h and T_{amb} are investigated. The time-derivative terms in Equations (7.4) and (7.5)[3] are negligible. Thereby, $V = V_h$ is obtained while setting $s_0 = 0$ in the Krylov subspace as presented in Equation (3.10). This speeds up the construction time of the pROM as only a single Krylov subspace needs to be constructed.

[3]This means no projection matrix is constructed based on the Krylov subspace of parameter s. When using the command line tool in 'Model Reduction inside ANSYS' for pMOR, the option '`-N 0`' is set.

By using this pROM, one only needs to modify the values of these parameters in the pROM and no re-generation of the ROM is required. This pROM is efficient for investigating the impact of environmental parameters on the temperature distribution inside human tissue via parametric study. In order to verify the accuracy of the pROM generated , stationary analyses of the FOM in Equation (7.3) are performed with film coefficient and ambient temperature as parameters within a specific range. The air velocity (v_{air}) dependent film coefficient (h) is calculated based on Equation (5.8). v_{air} is selected at {0.2, 0.8, 1.4, 2.6, 3.2, 3.8} m/s and the corresponding h is calculated as {3.1, 7.3, 10.2, 14.7, 16.7, 18.5} W/m^2/K. The ambient temperature T_{amb} is selected at {10, 14, 18, 22, 26, 30} °C. Finally, 36 different combinations of the parameters h and T_{amb} are obtained. It is illustrated in Figure 39 that different environmental conditions affect the temperature distribution inside the tissue and hence, the temperature difference across the TEG. As expected, a combination of a high film coefficient of $h = 18.49$ W/m^2/K and a low ambient temperature of $T_{amb} = 10$ °C leads to a maximum temperature difference of 2.18 K. A corresponding Seebeck voltage output, which is calculated via Equation (5.3), amounts to $V_{out} = 70.63$ mV for this model.

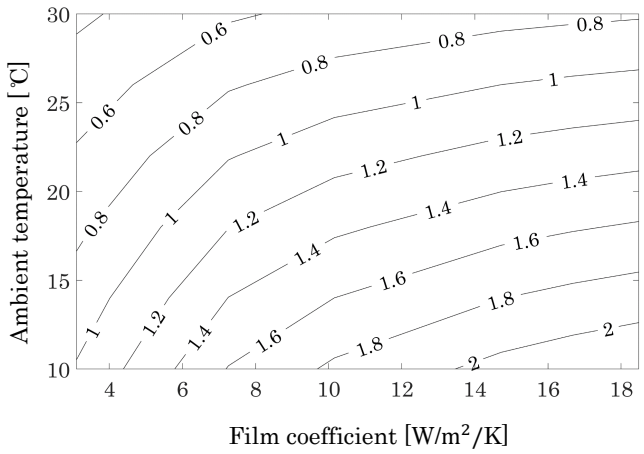


Figure 39: Temperature difference (K) across the TEG as a function of film coefficient and ambient temperature. Parametric solutions are obtained from the FOM. This figure is adapted from [89].

Furthermore, the pROM in Equation (7.5) is evaluated within stationary simulations over the same parameter space. The stationary full-size model is reduced from dimension $N = 108{,}292$ degrees of freedom (DOF) to $r_2 = 4$ DOF, that is only four moments around $h_0 = 10$ W/m^2/K have been matched. Figure 40 (A) compares the temperature difference across the TEG, as obtained from the full and the reduced models. The corresponding relative error is illustrated in Figure 40 (B). The maximum relative error between the full and the reduced model amounts to 3.3×10^{-3} %, which indicates that the pROM is an excellent substitute for the original large-scale model in parametric simulation. The error vanishes at $h = 10$ W/m^2/K, as $h_0 = 10$ is chosen for computing the projection matrix

V_h. Furthermore, as T_{amb} is a part of the model input in Equations (7.3) and (7.4) and hence, is not taken into account during the pMOR process, as it does not impact the accuracy of the ROM.

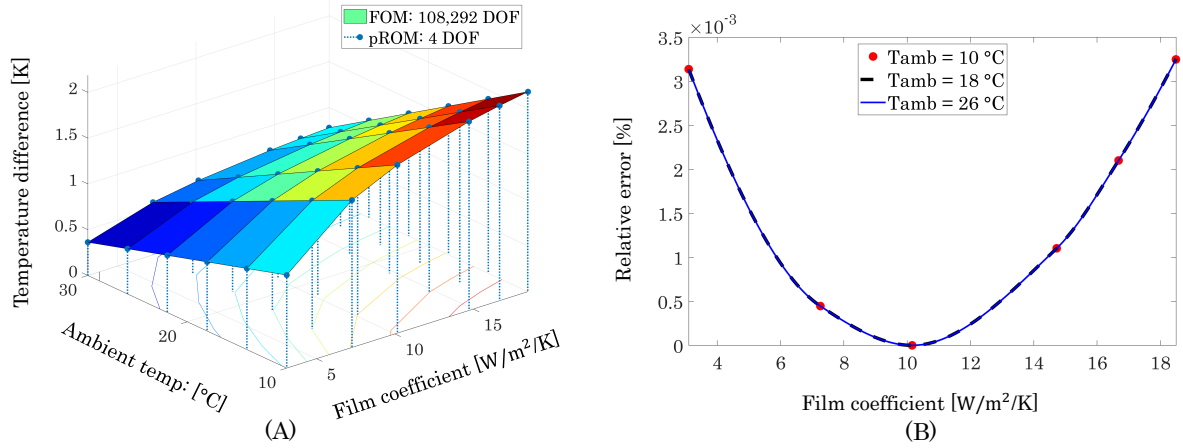


Figure 40: (A) Comparison of temperature difference across the TEG as obtained from the FOM and pROM. (B) The relative error in temperature difference as a function of film coefficient and ambient temperature. These figures are adapted from [89].

Lastly, the computational effort is presented in Table 4. The computational time for a single stationary solution of each parametrized full model is 9.1 s, which makes 327.6 s for the parametric studies within a 6×6 parameter space. To create a pROM via constructing the projection matrix $V = V_h$ based on a single Krylov subspace in Equation (3.10), it requires 8.1 s. However, when the pROM is available, its integration only takes 0.7 ms, which makes 25.2 ms for the complete parameter space. Hence, an acceleration of several orders of magnitude has been achieved through the pMOR.

Table 4: Computational time for pMOR and performing the parametric studies within a 6×6 parameter space (Intel®Core i5-7600 3.5 GHz, 32 GB RAM).

	FOM (108,292 DOF)	**pROM (4 DOF)**
Reduction time	n/a	8.1 s
Time for 36 static simulations	327.6 s	25.2 ms

7.2 Material Property Parametrization

According to TEG's maximum power output Equation (5.5), choosing the proper geometry for the TEG is an optimization problem. The electrical power output of TEG will be improved by increasing the cross-sectional area of the thermocouple legs, as this leads to the lower electrical resistance of the thermopile. However, the thermal resistance of the thermopile will decrease simultaneously, which leads to a drop in temperature difference across the TEG. Accordingly, voltage and power output will decrease. The temperature

field can not be considered stable in a human tissue environment. Heat flux and temperature gradients are altered in the presence of a TEG. Furthermore, according to the Seebeck voltage calculated in Equation (5.3), the number of thermocouples also determines the output voltage of the TEG. In this section, in order to identify the optimal size and the total number of thermocouples to balance these opposing effects, a surrogate TEG is utilized to replace the detailed structure as shown in Figure 41.

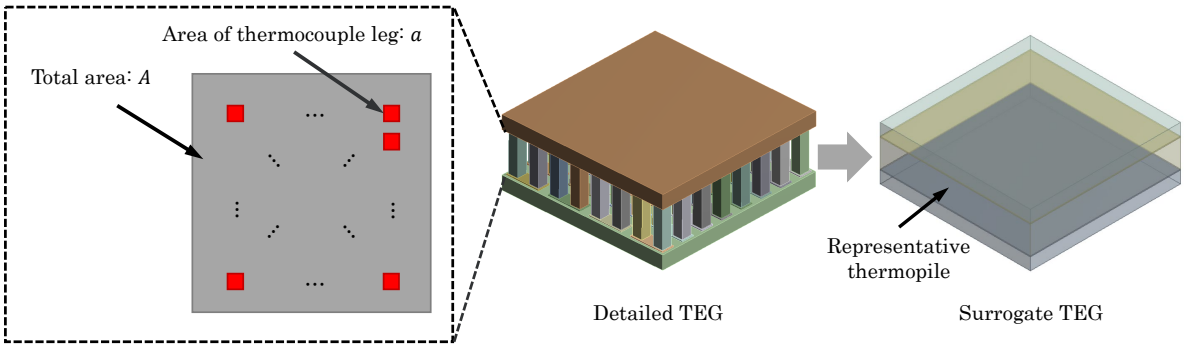


Figure 41: Detailed TEG with thermocouple legs in between the ceramic plates is replaced by a surrogate TEG with a block structure-representative thermopile.

The surrogate TEG features a footprint of A and represents n thermocouple legs, each with a cross-sectional area a. Hence, a fill factor η of the thermopile can be defined as

$$\eta = \frac{n \cdot a}{A} \,. \tag{7.6}$$

In this way, the thermal effect of varying the geometry of the thermocouple legs and their number can be represented via the fill factor. The effective thermal conductivity κ' of the surrogate thermopile amounts to

$$\kappa' = \eta \cdot \kappa \,, \tag{7.7}$$

where κ is the thermal conductivity of bismuth telluride. On the basis of Equation (7.6) and ignoring the negligible electrical resistance of the copper interconnects, the electrical series resistance of the TEG is calculated as

$$R_{TEG} = \frac{n^2 \cdot \rho_{el} \cdot l}{\eta \cdot A} \,, \tag{7.8}$$

where ρ_{el} and l are the electrical resistivity and height of the thermopile, respectively. Substituting (5.3) and (7.8) into Equation (5.5), the maximum power output of the TEG is computed as

$$P_{max} = \eta \cdot \frac{[\Delta T \cdot (\alpha_p - \alpha_n)]^2 \cdot A}{16 \cdot \rho_{el} \cdot l} \,. \tag{7.9}$$

In this case study, the material properties of bismuth telluride are assumed to be temperature-independent. Hence, the maximum power output of the TEG is dependent on the temperature difference across the TEG, the fill factor, and the height of the

thermopile, but not on the number of thermocouple legs. It should be noted that the temperature difference across the device is a function of the TEG's thermal resistance, which is impacted by the fill factor. This effect is considered in the bioheat model presented in Figure 18. The linear thermal FE model is presented in Equation (5.14), where convection, radiation, and evaporation heat transfer effects are all considered.

As mentioned in section 7.1, the film coefficient and ambient temperature in the convection BC can be set as the parameters for investigating the impact of environmental thermal conditions upon the TEG. Without the need to regenerate the ROM, the film coefficient and ambient temperature can be varied at the level of ROM by using pMOR methodology. In this section, the effective thermal conductivity κ' of the surrogate thermopile is preserved as a fill factor-dependent parameter within the FOM to find the optimum power output of a TEG. Additionally, static thermal analysis of the tissue model containing the TEG is performed to calculate the temperature difference across the TEG. Then the parameterized steady-state FE linear thermal model is written as

$$\Sigma_N \begin{cases} 0 = \underbrace{(A_0 + \kappa' \cdot A_1 + h \cdot A_2)}_{A(\kappa', h)} \cdot T + B \cdot \underbrace{\begin{bmatrix} Q_m \\ T_a \\ h \cdot T_{amb} \\ \bar{q}_{eva} \end{bmatrix}}_{u} \\ y(t) = C \cdot T \end{cases} , \tag{7.10}$$

where $A_0,\ A_1,\ A_2 \in \mathbb{R}^{N \times N}$ are the parameter-independent parts of heat conductivity matrix. The overall heat transfer coefficient is defined in Equation (7.19). Similarly to Equation (7.20), Equation (7.10) can be reduced as follows:

$$\Sigma_r \begin{cases} 0 = \overbrace{(\underbrace{V^T A_0 V}_{A_{0r}} + \kappa' \cdot \underbrace{V^T A_1 V}_{A_{1r}} + h \cdot \underbrace{V^T A_2 V}_{A_{2r}})}^{A_r(\kappa', h)} z + \underbrace{V^T B}_{B_r} \cdot u \\ y = \underbrace{CV}_{C_r} z \end{cases} , \tag{7.11}$$

where the projection matrix V is constructed by merging two physically independent local projection matrices, which are defined as the orthonormal bases of the following disjoint Krylov subspaces:

$$colspan\{V_{\kappa'}\} = \mathcal{K}_{r_1}\{[-A(\kappa'_0, h_0)]^{-1} A_1, [-A(\kappa'_0, h_0)]^{-1} B\} , \tag{7.12}$$

$$colspan\{V_h\} = \mathcal{K}_{r_2}\{[-A(\kappa'_0, h_0)]^{-1} A_2, [-A(\kappa'_0, h_0)]^{-1} B\} , \tag{7.13}$$

$$colspan\{V\} = colspan\{V_{\kappa'}, V_h\} , \tag{7.14}$$

where κ_0' and h_0 are the fixed expansion points with respect to each parameter. The local Krylov subspaces are generated for one parameter while keeping the other parameter constant. The dimensions of the local and the merged projection matrices are $V_{\kappa'} \in \mathbb{R}^{N \times r_1}$, $V_h \in \mathbb{R}^{N \times r_2}$, and $V \in \mathbb{R}^{N \times r}$, $r = r_1 + r_2$. Similarly to the proof given in section 3.1, it can be proved that the first r_1 moments of parameter κ' and the first r_2 moments of parameter h correspond between the pROM and the full model.

In order to investigate the impact of the fill factor on the TEG's power output, the environmental thermal conditions are set as a constant. The ambient temperature is chosen to be $T_{amb} = 21$ °C with a film coefficient of $h = 8.58$ W/m^2/K. Metabolic heat generation rates (Q_m) are applied as 498.5, 279.8, and 841.6 W/m^3 in muscle, fat, and skin layers, respectively. $\bar{q}_{eva} = -13.64$ W/m^2 is applied as the average heat flux boundary condition of the evaporation effect at the skin surface. Simulations of the full-scale model and the pROM are performed for different fill factors to obtain the temperature difference across the TEG (fill factor $\eta = \{1, 1.4, 1.9, 2.7, 3.7, 5.2, 7.2, 10, 13.9, 19.3, 26.9, 37.2, 51.8, 71.9, 100\}$ %). The maximum power delivered is computed according to Equation (7.9) and is shown in Figure 42.

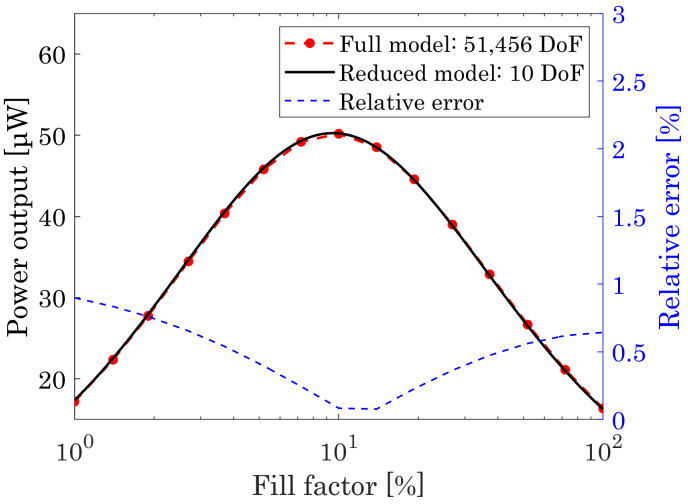


Figure 42: Comparison of the power output from the FOM and pROM with $T_{amb} = 21$ °C and $h = 8.58$ W/m^2/K. This figure is adapted from [90, 91].

The optimum fill factor is around 10 %, resulting in a power generation of 50.25 µW in the case of load resistance matching. This fulfills the power requirements of a cardiac pacemaker (30 - 100 µW). It should be noted that a DC/DC converter is necessary to boost the relatively low voltage level of the TEG to a usable value. Furthermore, the results obtained via the full and reduced models match well, with a maximum relative error of 0.9 %. Thus, the reduced model can be considered a highly accurate approximation of the original FE model. The reduction in model dimensions from $N = 51,456$ DOF to $r = 10$ DOF speeds up the parametric studies by a factor of 67,500, as shown in Table 5.

Table 5: Computational time for executing the pMOR and performing the parametric studies with 15 different fill factors (Intel®Core i5-7600 3.5 GHz, 32 GB RAM).

	Full model (51,456 DOF)	**pROM (10 DOF)**
Reduction time	n/a	6.5 s
Parametric simulation time	135 s	2 ms

On the basis of the pROM, the impact of environmental thermal conditions on the values of the fill factor and the performance of a TEG is studied. The film coefficient and the ambient temperature are varied. The values of the environmental air velocity are selected at $v_{air} = \{0.2, 0.95, 1.7, 2.7, 3.7\}$ m/s. Then the corresponding values of the film coefficient $h_c = \{3.1, 8.05, 11.41, 15.06\ 18.2\}$ W/m^2/K are calculated according to Equation (5.8). Firstly, the ambient temperature is set to 21 °C and the film coefficient h_{rad} is calculated according to Equation (5.11) as $h_{rad} = 5.48$ W/m^2/K. The values of the total film coefficient $h = h_c + h_{rad} = \{8.58, 13.53, 16.89, 20.54, 23.68\}$ W/m^2/K are obtained and the power output is calculated as a function of different fill factors (see Figure 43 (A)). The data confirms that a larger film coefficient will increase the power output of the TEG. Furthermore, it is found that the value of the optimum fill factor will vary between 9.5 and 11.1% depending on the film coefficient.

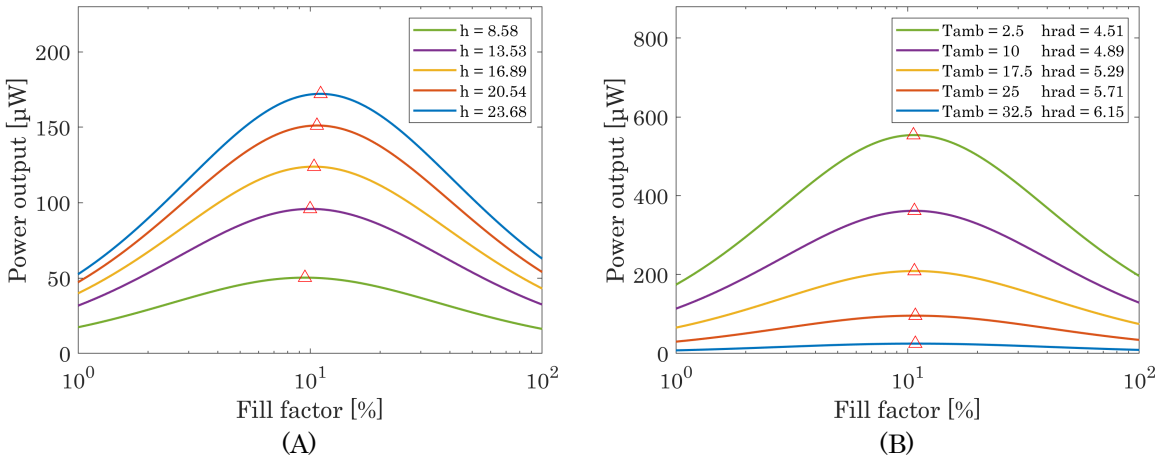


Figure 43: (A) Power delivery of TEG with parameterized fill factor and heat transfer coefficient. (B) Maximum power of TEG with parameterized fill factor and ambient temperature. These figures are adapted from [90, 91].

It is shown in Figure 43 (B) that when fixing the air velocity at $v_{air} = 2.7$ m/s, the film coefficient h_c is fixed at 15.06 W/m^2/K. The ambient temperature is changed in the range $T_{amb} = \{2.5, 10, 17.5, 25, 32.5\ \}$ °C and the corresponding values of the film coefficient $h_{rad} = \{4.51, 4.89, 5.29, 5.71, 6.15\}$ W/m^2/K are calculated. The value of the optimum fill factor is found to remain constant at 10.7 %. Finally, it is concluded that a combination of high film coefficient and low ambient temperature leads to maximum power output in the TEG. The optimum fill factor increases with the film coefficient.

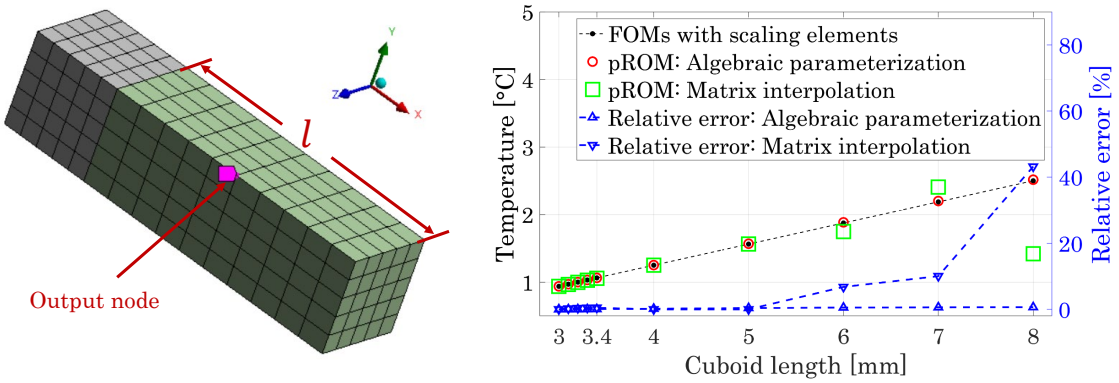
7.3 Geometry Parametrization

In contrast to the design optimization strategy used for finding the optimum cross-sectional area and number of thermocouple legs, the height of the thermocouple legs cannot be preserved as a parameter in the reduced model. The reason is that the system matrices are dependent on this geometrical parameter. It means that after the spatial discretization via FEM, the system equation of the thermal model is written as follows:

$$\Sigma_N \begin{cases} E(l) \cdot \dot{T}(t) = A(l) \cdot T(t) + B(l) \cdot u \\ y(t) = C(l) \cdot T(t) \end{cases}, \tag{7.15}$$

where l is the thermopile height and $E(l), A(l) \in \mathbb{R}^{N \times N}$, $B(l) \in \mathbb{R}^{N \times m}$, and $C(l) \in \mathbb{R}^{p \times N}$ are the geometrical parameter-dependent matrices. To generate the pROM from Equation (7.15), MI- and AP-based pMOR methods as mentioned in section 3.2 can be used.

At the beginning of this case study, these two methods are firstly applied to a thermal cuboid model (see Figure 44 (A)), which is used to imitate the thermocouple leg in a TEG, as introduced in section 5.2 for comparison. The length of the thermal cuboid is parameterized between 3 and 8 mm by using the ANSYS parametric design language (APDL) command '`NMODIF`'[4].



(A) (B)

Figure 44: (A) An output node is selected in the middle of the thermal cuboid for comparison of the results obtained from FOMs and pROMs. (B) Comparison of the temperature results from the pROMs obtained via MI- and AP-based pMOR methods.

For both MI- and AP-based pMOR methods, the local ROMs and the snapshot matrices are sampled at $l = \{3{,}4{,}5\}$ mm for a fair comparison. The pROMs are generated at $l =$

[4]It is found that changing the geometry in the ANSYS geometry editor 'DesignModeler' might also change the node ordering in the mesh, which has an impact on the accuracy of the generated pROM. Therefore, the command '`NMODIF`' is used to preserve the mesh topology and node ordering while parameterizing the geometrical parameter. This modifies the coordinate locations of the nodes in the mesh directly. Commands '`VLSCALE`' and '`ARSCALE`' can also be used to adjust the nodes in volume and area.

{3.1, 3.2, 3.3, 3.4, 6, 7, 8} mm. It can be observed clearly from Figure 44 (B) that both methods generate reliable pROMs. The pROM generated via AP-based pMOR method gives more accurate results even when the parameter deviates from the snapshot samples. In comparison, the MI-based pMOR method constructs precise pROM only close to the local ROMs.

The computational cost for generating the pROMs is presented in Table 6. Compared to an AP-based pMOR method, the MI-based pMOR method is still more efficient at generating the pROM even though the MOR process for the generation of the local ROMs is performed three times. This is because the full-scale parameter-independent matrices need to be constructed and written in Matrix Market format.

Table 6: Comparison of the computational time for generating the pROMs from the thermal beam model and performing parametric studies (Intel®Core i5-7600 3.5 GHz, 32 GB RAM).

	FOM	**pROM: matrix interpolation**	**pROM: algebraic parameterization**
System dimension	1980 DOF	30 DOF	30 DOF
Extracting full system matrices	n/a	3×0.2 s	3×0.2 s
Generate the pROM	n/a	0.4 s	2.3 s
Parametric simulation	10×4 s	10×0.3 ms	10×19 ms
Total time	40 s	1 s	3.1 s

Afterwards, the TEG model, which is incorporated in a simplified human tissue model, as presented in section 7.2, is utilized as the case study to investigate the impact of thermopile height on a TEG. The fill factor is fixed at $\eta = 10$ %. $h = 8.58$ W/m^2/K and $T_{amb} = 21$ °C are set in the thermal BC. The height of the thermopile is parameterized between 3 and 4 mm, $l = \{3.0, 3.2, 3.4, 3.6, 3.8, 4.0\}$ mm. With different values of the geometrical parameter, the size of the full-scale local models implemented in ANSYS Mechanical is changed as shown in Table 7.

Table 7: Size of full-scale and reduced local models with different heights of the thermopile. The geometry of the model is parameterized in the ANSYS geometry editor 'DesignModeler'.

Height l (mm)	**Full model (N_i DOF)**	**Reduced local model (r DOF)**
3.0	51,476	10
3.2	51,452	10
3.4	51,467	10
3.6	51,427	10
3.8	51,456	10
4.0	51,388	10

In accordance with the method suggested in [47, 48], the local models are reduced to 10 DOF, and the corresponding projection matrices are padded with zero elements as

indicated in Equation (3.30). Then all the pROMs are transformed onto a generalized coordinate and interpolated at $l = \{3.1, 3.3, 3.5, 3.7, 3.9\}$ mm. The power output results from both full-scale and reduced models are shown in Figure 45. For a fair comparison, both methods use linear interpolation, i.e., only two nearby local ROMs are used for the interpolation. It is observed that increasing the height of the thermopile has a positive impact on the TEG power output. Furthermore, it is found that the modified MI-based pMOR method suggested in this work generates the more accurate interpolated pROM. The pROMs speed up the parametric studies by a factor of 27,500 as shown in Table 8.

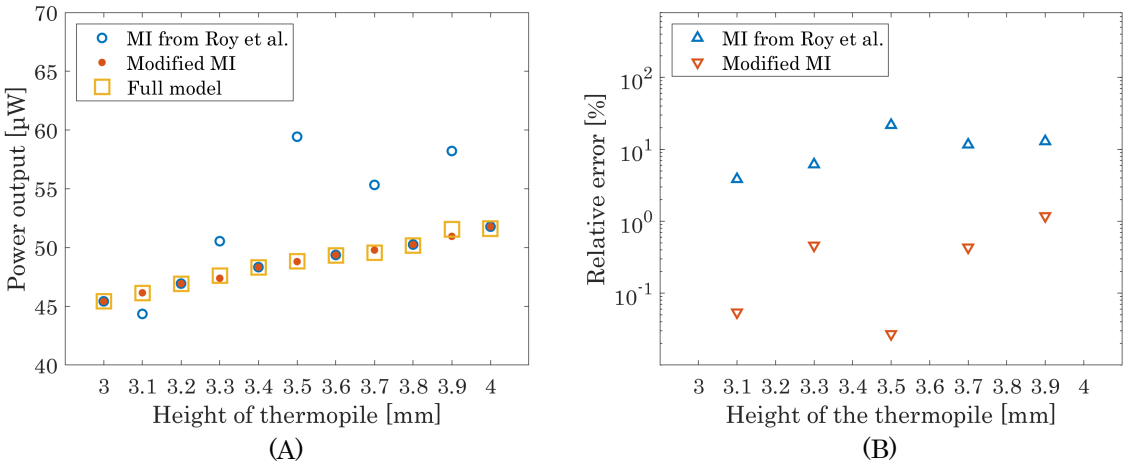


Figure 45: (A) Power output of a TEG with parameterized height of thermopile. Comparison of the results from the pROMs obtained via the MI-based pMOR method suggested in [47, 48] and the modified MI-based pMOR method suggested in section 3.2. (B) Relative errors between FOMs and pROMs. These figures are adapted from [91].

Table 8: Computational time for executing the MI-based pMOR and performing the parametric studies with 11 different heights of thermopile (Intel®Core i5-7600 3.5 GHz, 32 GB RAM).

Computational time	**Full model**	**pROM**
Local ROMs generation	n/a	71.8 s
Interpolating ROMs	n/a	5.7 s
Simulation time	110 s	4 ms

Alternatively, the size of the local FOMs and the node ordering can be retained while changing the height of thermopile by using the APDL command '`NMODIF`', which is verified in the thermal cuboid model. As presented in Figure 46, the original height of the thermopile is 3 mm and the selected nodes in the bottom ceramic plate, bottom copper layer, and surrogate thermopile are moved in the negative y-axis direction. Table 9 shows the mesh sizes of the local FOMs and the ROMs. As shown in Figure 47, the pROMs are interpolated at $l = \{3.1, 3.3, 3.5, 3.7, 3.9\}$ mm via both modified and 'classic' MI-based pMOR methods. It is demonstrated that the 'classic' MI-based pMOR method works well if the mesh topology and node ordering of the local models can be controlled.

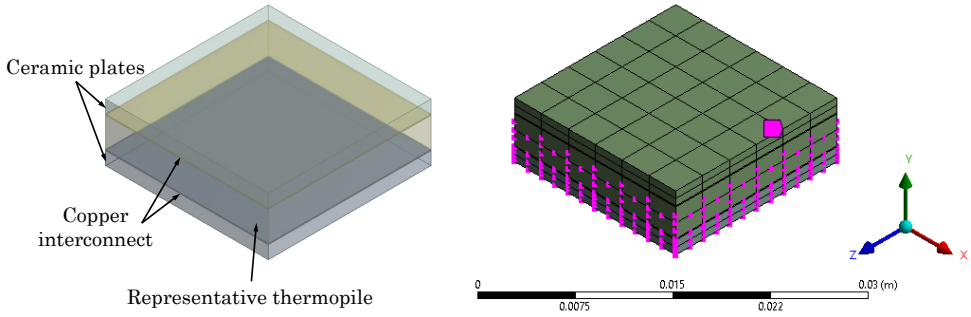


Figure 46: Location-modified nodes in surrogate TEG by using the APDL command '`NMODIF`'. These figures are adapted from [86].

Table 9: Size of full and reduced local models with different heights of the thermopile. The geometry of the model is parameterized in the ANSYS geometry editor 'DesignModeler' or by using the APDL command '`NMODIF`'.

Height l **(mm)**	**Full model DesignModeler** (N_i **DOF)**	**Full model NMODIF** (N **DOF)**	**Reduced local model** (r **DOF)**
3.0	51.476	51.476	10
3.2	51.452	51.476	10
3.4	51.467	51.476	10
3.6	51.427	51.476	10
3.8	51.456	51.476	10
4.0	51.388	51.476	10

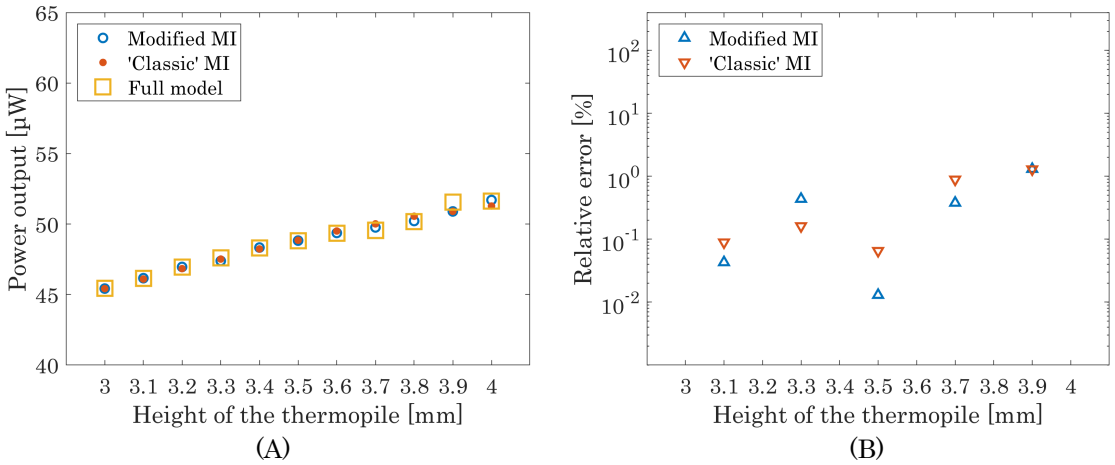


Figure 47: (A) Power output of a TEG with parameterized height of thermopile. Comparison of the results from the pROMs obtained via the 'classic' and modified MI-based pMOR method. (B) Relative errors between full-scale models and pROMs. These figures are adapted from [86].

It is observed in the time comparison Table 10 that the 'classic' MI-based pMOR method requires less time to interpolate the pROMs. This is because, in the modified

MI-based pMOR method, the singular value decomposition (SVD) needs to be performed repeatedly on the matrix pool in Equation (3.31), where the left and right local ROMs are updated for interpolation of the new pROM. In the 'classic' MI-based pMOR method, the SVD process needs to be done only once.

Table 10: Computational time for executing modified and 'classic' MI-based pMOR methods and performing the parametric studies with 11 different heights of thermopile (Intel®Core i5-7600 3.5 GHz, 32 GB RAM).

Computational time	**Full model**	**Modified MI**	**'Classic' MI**
Local ROMs generation	n/a	75.1 s	75.5 s
Interpolating ROMs	n/a	6 s	3.1 s
Simulation time	110 s	6.3 ms	6.8 ms

Except for the MI-based pMOR method, the AP-based pMOR method introduced in section 3.3 is also applicable to the TEG model. In this case study, the static thermal analysis calculates the temperature difference across the TEG. Therefore, after the algebraic parameterization, a static geometrical parameter-independent FE model reads

$$\Sigma_N \begin{cases} 0 = \underbrace{(\frac{1}{\alpha} \cdot A_{\frac{1}{\alpha}} + A_1 + \alpha \cdot A_\alpha)}_{A(\frac{1}{\alpha},\alpha)} \cdot T + B \cdot u \\ y = C \cdot T \end{cases}, \tag{7.16}$$

where α is the scaling factor for the height of the thermopile. The parameter-independent system matrices $A_{\frac{1}{\alpha}}$, A_1, and $A_\alpha \in \mathbb{R}^{N\times N}$ are computed through the adapted numerical scheme in Equation (3.47). The snapshot matrices are sampled at $l = \{3.0, 3.1, 3.2\}$ mm. On the basis of Equation (7.16), a multivariate moment matching-based pMOR method can be applied to generate a pROM

$$\Sigma_r \begin{cases} 0 = \underbrace{(\frac{1}{\alpha} \cdot V^T A_{\frac{1}{\alpha}} V + V^T A_1 V + \alpha \cdot V^T A_\alpha V)}_{A_r(\frac{1}{\alpha},\alpha)} \cdot z + \underbrace{V^T B}_{B_r} \cdot u \\ y = \underbrace{CV}_{C_r} \cdot z \end{cases}, \tag{7.17}$$

where $V \in \mathbb{R}^{N\times r}$ is a global projection matrix. It is constructed via merging the local projection matrices presented in Equations (3.44) and (3.45), where $\alpha_0 = \alpha_1 = 1$.

As is presented in Figure 48, the maximum relative error of the temperature results between the FOM and pROM is only 0.4 %. When $l = \{3.0, 3.1, 3.2\}$ mm, as expected the relative error is less than 0.05 % and the relative error in the pROM increases while the height of the thermopile deviates from the snapshot models. The computational time for applying the AP-based pMOR method on the TEG model is shown in Table 11. The dimension of the full-scale model is N = 19,731 DOF, which means that the numerical

scheme in Equation (3.42) needs to be solved 389,312,361 times via the original method from [49]. By using the adapted numerical scheme introduced in section 3.3 of this work, one needs to solve Equation (3.47) just once to obtain the coefficients $s_{i,j}$. Afterward, one needs to perform the linear combination of the snapshot matrices, which contains only q nonzero elements. The speed-up factor is 1,786.

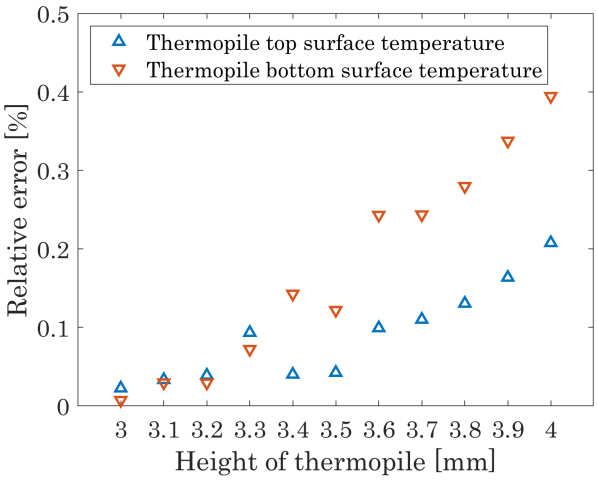


Figure 48: Relative error of the temperature results between full model and pROM with parameterized height of thermopile. The pROM is constructed based on the local ROMs with $l = \{3.0, 3.1, 3.2\}$ mm.

Table 11: Computational time for generating parameter-independent matrices, $N = 19{,}731$ DOF (Intel®Core Processor (Broadwell, IBRS) 3.00 GHz, 64 GB RAM).

Method	**Steps**	**Time**
Method from [49]	Solve Equation (3.42) N^2 times	50,943.5 s
This work	1. Solve Equation (3.47) once 2. Calculate the coefficients $s_{i,j}$ 3. Compute the matrices in Equation (3.48)	28.5 s

7.4 Combining pMOR with the Submodeling Technique

In order to obtain realistic human tissue temperature distribution results, a human upper-torso model consisting of realistic geometry of the solid internal organs, skeleton, and main vessels, as well as muscle, fat, and skin layers, is constructed based on segmented magnetic resonance imaging (MRI) data [70] as presented in Figure 19 (A). The model is implemented in ANSYS Mechanical, where realistic thermal data and physiologically correct material parameters obtained from [92] as shown in Table A5, are assigned to the various tissue sections. Subsequently, as shown in Figure 49, the TEG model is placed in the fat layer within the chest region of the human torso model.

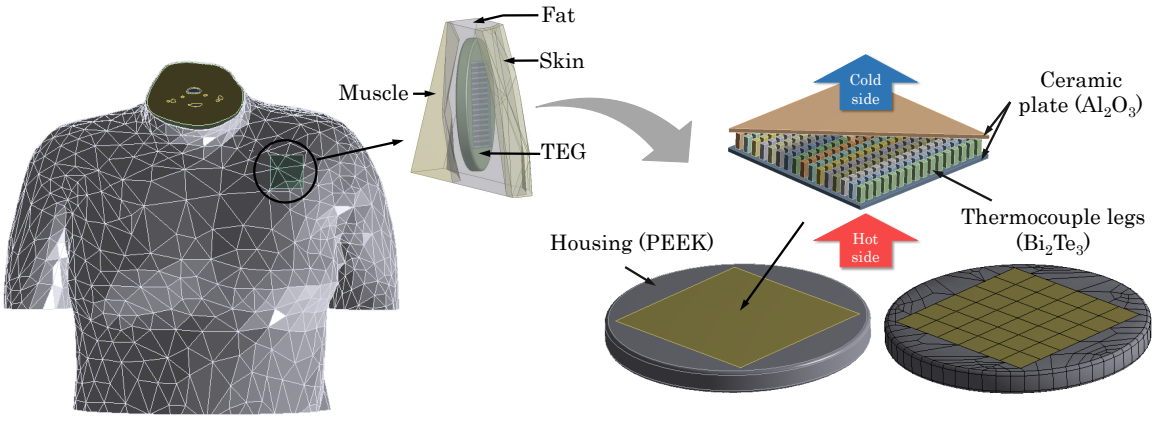


Figure 49: A TEG with an array of 16×16 thermocouple legs and a disk-shaped housing. It is placed in the fat layer of the chest region. This figure is adapted from [67].

In this case study, the geometry of the TEG model is constructed based on a commercially available TEG. An array of 16×16 p-type (hole transporting) and n-type (electron transporting) bismuth telluride (Bi_2Te_3) thermocouples with a height of 2.27 mm and a cross-section of 0.8×0.8 mm^2 are located between upper and lower 24.6× 24.6×0.565 mm^3 ceramic square plates made of aluminum oxide (Al_2O_3) excluding the copper interconnections. The overall height of the TEG is 3.4 mm. Further, the TEG is surrounded by a disk-shaped housing made of Polyetheretherketone(PEEK) with the same height. After spatial discretization, the FE model of TEG contains 127,307 nodes in total. The material properties used for each part in the TEG are shown in Table A7. To characterize the internal heat transfer in human tissue, the Pennes' bioheat Equation (5.6) is used. The blood perfusion and metabolic heat generation rates are considered in muscle, fat, and skin layers while no heat by blood perfusion and metabolism in other internal organs is considered. Furthermore, instead of using a central blood pool model proposed in [93,94], the temperature of arterial blood T_a is assumed constant at 37 °C when it travels from the heart pool to other body parts. This assumption ignores the temperature variations in arterial blood and obviously overestimates the effects of blood perfusion in tissue. However, given a constant arterial blood temperature, it is possible to apply the blood perfusion effect to the model. In the future, if this assumption limits the accuracy of the blood perfusion source term, one can still overcome it by adjusting the arterial blood temperature or the blood perfusion rate in the Pennes' bioheat equation as suggested in [95]. In addition, the heat generated inside the human torso model is balanced simultaneously by the various external heat transfer effects. The convection, radiation, and evaporation effects between the skin surface and the environment described in Equation (5.7) are applied as the main heat losses. It is shown in [96] that the contributions of heat losses for each effect are calculated as 29 %, 38.1 % and 24.2 %, respectively. The remaining heat losses from other sources are ignored in this model. No BCs are imposed at the truncated neck, extremities, and lower torso, but only at the skin surface.

It should be noted that the usual design optimization process of finding the maximum temperature difference in a TEG is shown in Figure 50 (A). For each new value of the parameter, the temperature distribution result on the TEG is obtained after the solution of the original full-scale FE model.

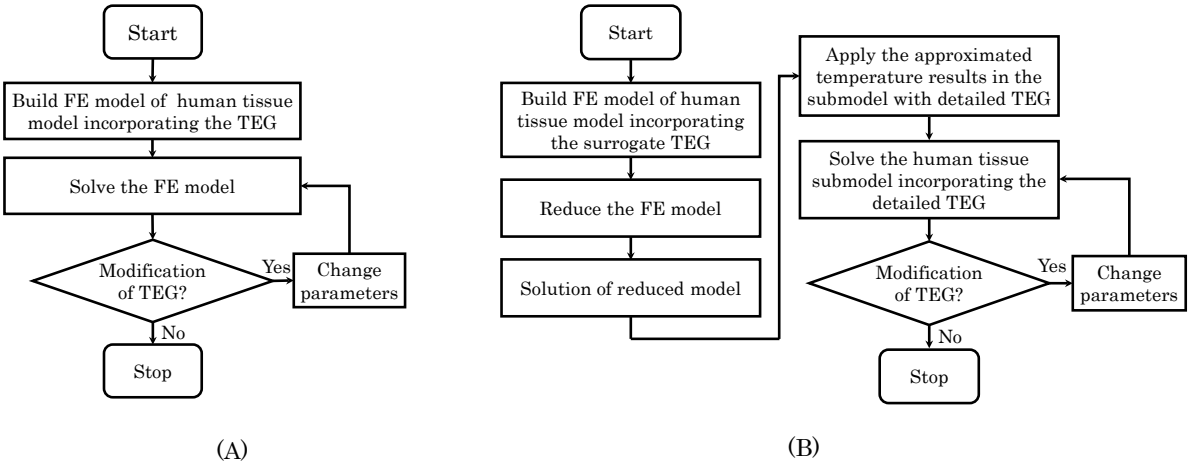


Figure 50: Flow charts for design optimization of TEG. (A) Usual design process. (B) Optimized design process with MOR and submodeling techniques. These figures are adapted from [67].

However, due to the complex geometry of the realistic human torso model, a large-dimensional FE model is generated (with 1,045,923 nodes in total) and the simulations performed on such a model are time-consuming. Therefore, in this section, a new method, which combines the pMOR and submodeling techniques (see Figure 50 (B)), is presented for speeding up the whole simulation and optimization process. With this new method, simulations of a TEG can be handled in a relatively small human tissue submodel, whose dimension is much smaller (with 132,826 nodes in the human tissue submodel incorporating the TEG) than the original human torso model.

The submodeling technique has already been developed and applied in ANSYS Mechanical [97]. The idea of the submodeling technique is to separate the simulation of a complex model into global and submodel parts. The submodel is a given region of interest, where you can obtain more satisfactory results. In this case study, a surrogate TEG is implemented in the human torso global model (see Figure 51 (A)). The structure of the thermocouple legs is replaced with a representative thermopile, where fewer elements are generated after spatial discretization. Compared to 127,307 nodes in a detailed TEG, a surrogate TEG contains only 2,694 nodes. Then, the detailed TEG is placed in a submodel, which contains the human tissue surrounding the detailed TEG (see Figure 51 (B)). The temperature distribution results calculated from the human torso global model are used as temperature cut boundaries at the cut surfaces of the submodel. By using the submodeling technique, it is possible to perform efficient simulations of the TEG for design optimization. Given that fewer elements are generated in the submodel, less computational effort is required during the simulations.

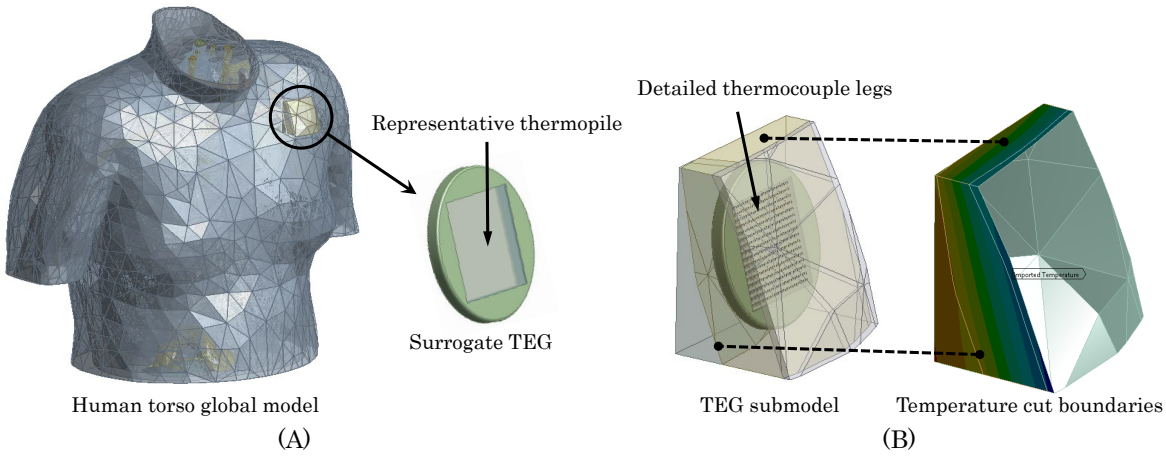


Figure 51: (A) Human torso global model incorporating the surrogate TEG with representative thermopile inside. (B) Human torso submodel incorporating the detailed TEG. Applying temperature cut boundaries, which are calculated from the global model, on the cut surfaces of the human tissue submodel. These figures are adapted from [67].

Although the submodeling methodology speeds up simulations by means of simulating the TEG in a relatively small submodel, a full-scale human torso model incorporating the surrogate TEG simulation is still indispensable for obtaining the same temperature cut boundaries as those used in the submodel. Whenever any modification to the TEG is required, the whole human torso model needs to be simulated once again. Moreover, if the environmental conditions surrounding the human body are changed, one needs to generate different reduced models reflecting different environmental conditions. Therefore, it is essential to generate a pROM which preserves the parameters in the environmental conditions. As verified in section 7.1, the parametric form of the full-scale linear thermal tissue model in Equation (5.14) can be written as follows:

$$\Sigma_N \begin{cases} E \cdot \dot{T}(t) = \underbrace{[A_0(h_0) + (h - h_0) \cdot A_1]}_{A(h)} \cdot T(t) + B(h_0) \cdot \underbrace{\begin{bmatrix} Q_m \\ T_a \\ \frac{h}{h_0} \cdot T_{amb} \\ \bar{q}_{eva} \end{bmatrix}}_{u} \\ y(t) = C \cdot T(t) \end{cases} , \tag{7.18}$$

where h is the updated film coefficient. According to the linearized radiation heat transfer effect in Equation (5.11), the radiation coefficient h_{rad} is added to the convection heat transfer coefficient h_c, and h reads

$$h = h_c + h_{rad} \, . \tag{7.19}$$

Then the parametric studies of Equation (7.18) are carried out with the updated film

coefficient h. On the basis of the pMOR method introduced in section 3.1 and validated in section 7.1, the pROM of Equation (7.18) is then obtained as follows:

$$\Sigma_r \begin{cases} \underbrace{V^T E V}_{E_r} \dot{z}(t) = \overbrace{[\underbrace{V^T A_0(h_0) V}_{A_{0r}} + (h - h_0) \cdot \underbrace{V^T A_1 V}_{A_{1r}}]}^{A_r(h)} z(t) + \underbrace{V^T B(h_0)}_{B_r} \cdot \underbrace{\begin{bmatrix} Q_m \\ T_a \\ \frac{h}{h_0} \cdot T_{amb} \\ \bar{q}_{eva} \end{bmatrix}}_{u} \\ y = \underbrace{CV}_{C_r} z(t) \end{cases} . \tag{7.20}$$

The reduced state $z \in \mathbb{R}^r$ given by the pROM can be projected back to its full size through the projection Equation (2.8), and the full-scale temperature distribution results are implemented as the temperature cut boundaries in the human tissue submodel incorporating the detailed TEG, as presented in Figure 51 (B).

In the full-scale human tissue model incorporating the surrogate TEG, the temperature distribution results are obtained in two steps, as presented in Table 12. Firstly, an initial thermal state is obtained via the steady-state thermal simulation with the air velocity $v_{air} = 1.25$ m/s and the ambient temperature $T_{amb} = 15$ °C. The values of the convection film coefficient h_c and radiation coefficient h_{rad} are then calculated depending on the air velocity v_{air} and the ambient temperature T_{amb}, respectively. On the basis of this initial state, a transient thermal simulation is carried out in the second step with $v_{air} = 0.38$ m/s. Furthermore, in order to verify the accuracy of the human torso pROM, the air velocity and the ambient temperature are again changed to $v_{air} = 0.75$ m/s and $T_{amb} = 25$ °C.

Table 12: Parameter values used in the environmental boundary conditions.

Environmental variables	$\boldsymbol{v_{air}}$ **(m/s)**	$\boldsymbol{h_c}^{\dagger}$ **(W/m²/K)**	$\boldsymbol{T_{amb}}$ **(°C)**	$\boldsymbol{h_{rad}}^{\ddagger}$ **(W/m²/K)**
Steady-state thermal: initial state	1.25	9.49	15	5.16
Transient thermal: case 1	0.38	4.64	15	5.16
Transient thermal: case 2	0.75	6.98	25	5.71

† Calculated according to Equation (5.8).
‡ Calculated according to Equation (5.11).

Given the use of a surrogate TEG to replace the detailed TEG model in the human tissue model, it is essential to investigate the influence of the surrogate TEG on the temperature distribution result in human tissue. This being the case, the thermal simulation results of the human torso model with detailed and surrogate TEG are compared, as shown in Figure 52. It is observed that the maximum temperature relative error between the nodes in the paths selected in the detailed and surrogate TEG implanted submodels

is 2.8 %. This result shows that the surrogate TEG will not influence the temperature distribution results to excess, and they can be used as the temperature cut boundaries in the submodel.

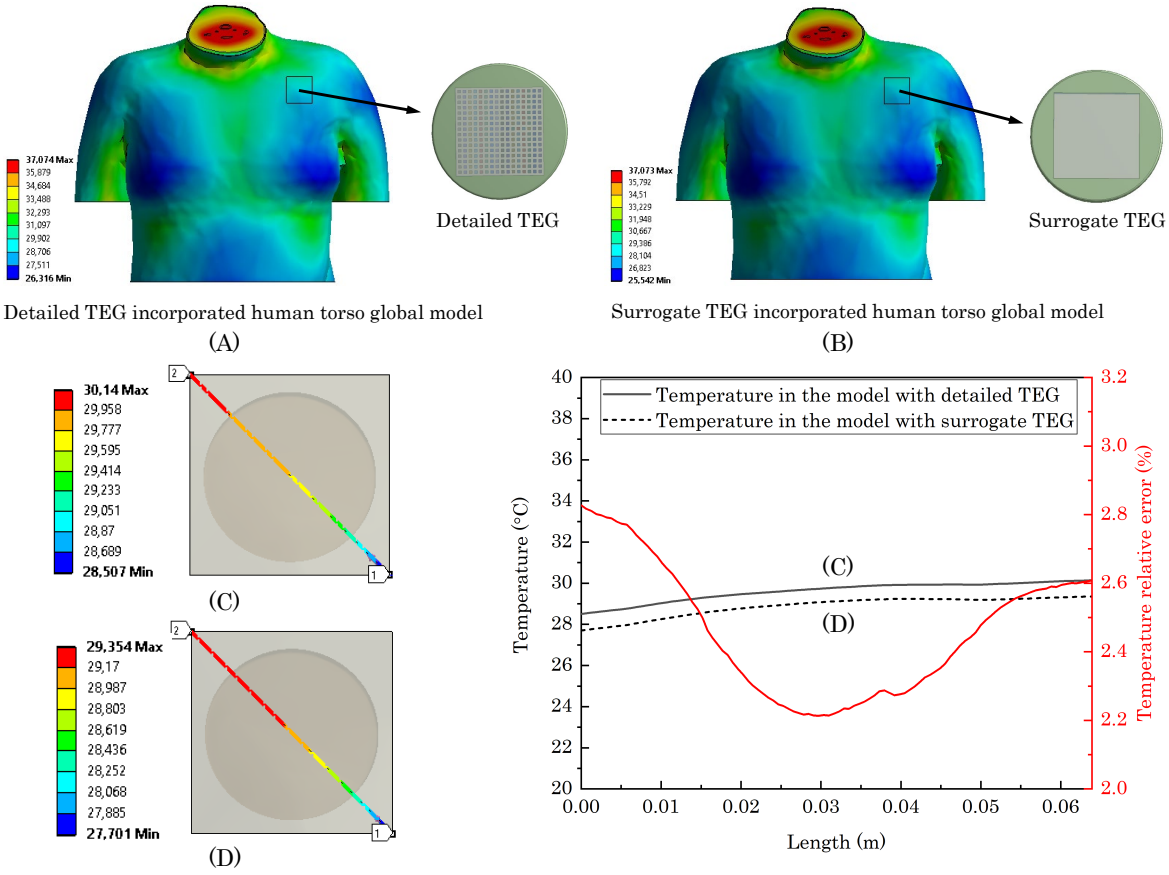


Figure 52: Comparison of the temperature results in the path selected at the skin surface of the submodel. (A) Torso model incorporating the detailed TEG. (B) Torso model incorporating the surrogate TEG. (C) Temperature results in the path selected at the skin surface of the human torso model incorporating the detailed TEG. (D) Temperature results in the path selected at the skin surface of the human torso model incorporating the surrogate TEG. These figures are adapted from [67].

After that, on the basis of the pROM as presented in Equation (7.20), the accuracy of the pROM is verified as shown in Figure 53. Parametric transient thermal simulations are performed based on the pROM by choosing the different environment variables as shown in Table 12. In these two transient thermal simulations, different average heat fluxes $\bar{q}_{eva}$ are calculated by collecting the heat flux snapshots at all k time steps in a total 8000 s simulation time ($k = 14$, $t = \{80, 135, 190, 355, 851, 1651, 2451, 3251, 4051, 4851, 5651, 6451, 7251, 8000\}$ s). $r_1 = 31$ moments around the parameter s is enough and $r_2 = 5$ additional moments around the film coefficient parameter h is required to give the accurate results in the pROM. It is found that the norm of the first 5 column vectors in projection matrices V_s and V_h are the same. A deflation is thus applied when merging these two projection matrices. In other cases, the accuracy of the pROM can be improved by increasing the values of r_1 and r_2. The temperature results selected from

three nodes in the muscle, fat, and skin layers are compared. The maximum relative error between the full and reduced models is 0.8 %, which shows that the pROM is an excellent approximation of the full-scale human torso thermal model.

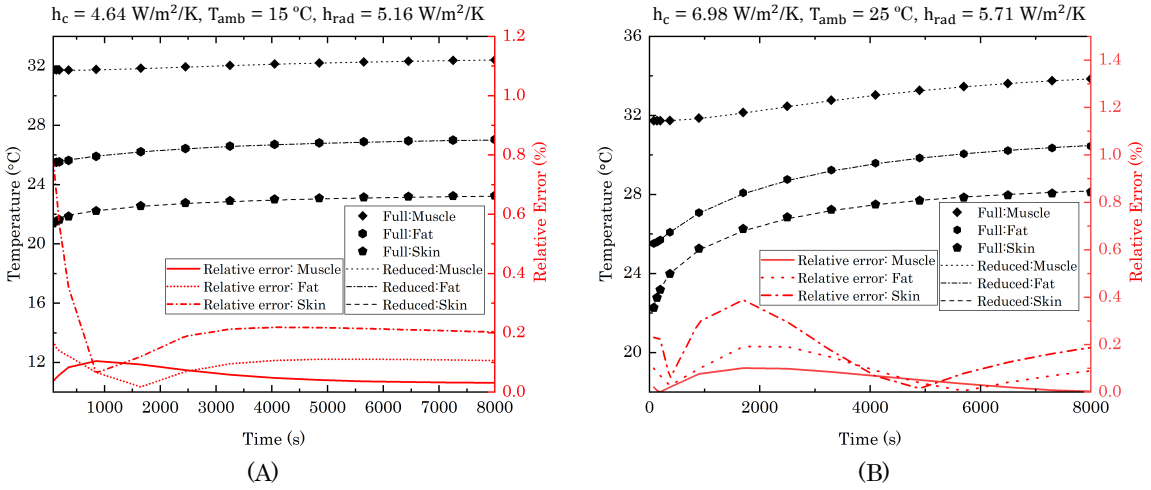


Figure 53: Comparison of the temperature results between full and parametric reduced human torso models (921,336 DOF vs. 31 DOF) in the environmental boundary conditions presented in Table 12. (A) Transient thermal simulation with convection film coefficient $h_c = 4.64$ W/m^2/K, ambient temperature $T_{amb} = 15$ °C, and radiation coefficient $h_{rad} = 5.16$ W/m^2/K. (B) Transient thermal simulation with convection film coefficient $h_c = 6.98$ W/m^2/K, ambient temperature $T_{amb} = 25$ °C, and radiation coefficient $h_{rad} = 5.71$ W/m^2/K. These figures are adapted from [67].

Subsequently, the accurate temperature results obtained from the pROM are used to approximate the full-scale temperature distribution result on the human torso model by projecting the reduced temperature state vector back to full size according to Equation (2.8). These full-scale temperature results are later used as the temperature cut boundaries in the detailed TEG implanted human tissue submodel. In the human torso submodel incorporating the detailed TEG, the convection, radiation, evaporation effects at the skin surface and the blood perfusion and metabolic heat generation rates in the muscle, fat, and skin layers are all still applied. To verify the accuracy of the simulation in the submodel, the temperature results are compared between these two detailed TEG models: one is simulated in the global human torso model and the other is simulated in the submodel (see Figure 54). The maximum absolute temperature error between the nodes in the selected path in the TEG model is 0.52 K, which is 1.6 % as its relative error. This result shows that, through the pMOR and submodeling techniques, the accuracy of temperature distribution results in the TEG model is guaranteed.

Finally, the simulation time used for generating the temperature results on the TEG with and without pMOR and submodeling techniques is compared. It is presented in Table 13 that, for the global human torso model incorporating the detailed TEG, the total simulation time is 1,515 s. Compared to the total computational time in the submodel (669 s), the computational effort is improved by 2.3 times. In addition, there is no need to repeat the steps of pMOR and reprojection if the environmental condition is not changed.

The optimization process of the TEG can be handled with the submodel in just 182 s, which is 8.3 times faster than the simulation in the full-scale torso model.

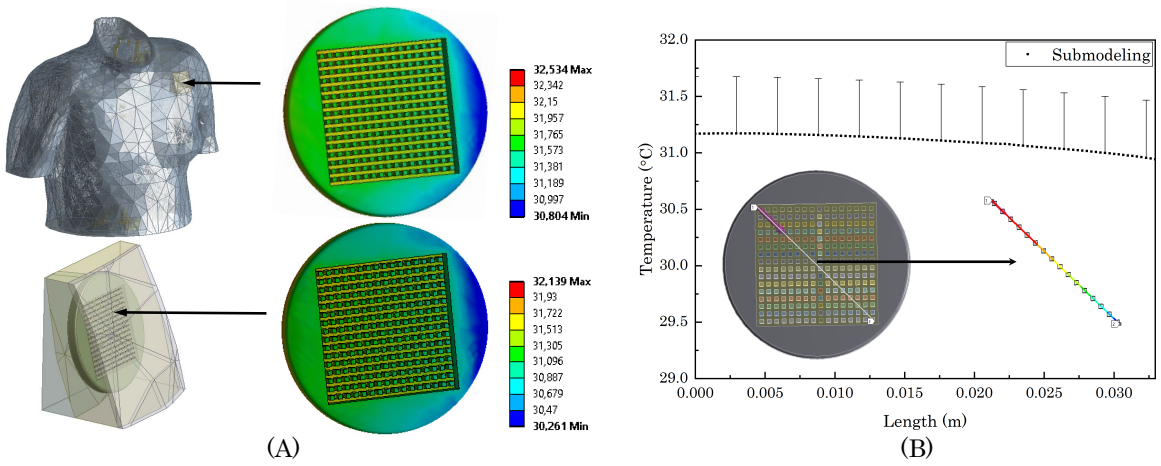


Figure 54: Comparison of the temperature results on the detailed TEG model. (A) Detailed TEG simulated separately in the global human torso model and submodel. (B) Nodal temperature results (total 100 nodes) and their error bars (referring to the results) in the selected path on top surfaces of thermocouple legs simulated in the submodel. These figures are adapted from [67].

Table 13: Computational time comparison between the thermal simulations of detailed TEG in submodel and global torso model (Intel®Xeon E5-2687W v4 3.00 GHz, 324 GB RAM).

Computational time	**pMOR**	**Reprojection**	**Simulation in submodel**	**Total**
Detailed TEG in submodel	432 s	54.5 s	182 s	669 s
Detailed TEG in global model	n/a	n/a	n/a	1515 s

7.5 System-Level Simulation with Peltier Effect

As introduced in section 5.2, the TEG can be utilized as a power source supplemental to the electrically active devices. In the previous sections on TEGs, for the efficiency and applicability of MOR methods, all material properties are defined as temperature-independent. The model is implemented in the thermal domain only due to the fact that the open-circuit Seebeck voltage output and the corresponding power output from the TEG can be calculated in Equations (5.3) and (5.4), respectively. However, when connecting the TEG to the electrical circuit, one has to consider the Peltier effect in the model, which is the reverse of the Seebeck effect. It presents the generation of heat in the thermocouple legs by an electrical current. The Thomson effect could be neglected as the Seebeck coefficient is considered to be temperature-independent. As is proved in [98], it is mandatory to include the Peltier effect in the model as it decreases the power output by almost one-third. Therefore, in this section, the aim is to present the

analysis of a TEG thermal model, which considers the Peltier effect when using effective thermal conductivity in the material of thermopile. Furthermore, a pROM is constructed to speed up the simulation and the Peltier effect could also be considered via system-level simulation.

In this case study, similarly to the TEG introduced in section 7.2, a surrogate TEG model with a representative thermopile ($20\times20\times4$ mm^3) is implemented. By using the fill factor η defined in Equation (7.6), the changes of geometry in the detailed TEG model during parametric analysis are avoided. It helps to facilitate the parameterization of the array configuration and thermopile cross-section. In consequence, as presented in [98], the representative thermopile deploys the equivalent material properties including equivalent Seebeck coefficient α', electrical resistivity ρ'_{el}, and thermal conductivity κ' as follows:[5]

$$\alpha' = n \cdot \frac{(\alpha_p - \alpha_n)}{2} \,, \tag{7.21}$$

$$\kappa' = \eta \cdot \frac{(\kappa_p + \kappa_n)}{2} \,, \tag{7.22}$$

$$\rho' = \frac{n^2}{\eta} \cdot \frac{(\rho_{el,p} + \rho_{el,n})}{2} \,, \tag{7.23}$$

where n is the number of thermocouple legs. α_p, α_n, κ_p, κ_n, and $\rho_{el,p}, \rho_{el,n}$ are the Seebeck coefficient, thermal conductivity, and the electrical resistivity of p- and n-type thermoelectric materials, respectively.

In addition, the TEG is integrated with a housing inside the human tissue (cross-section of 100×100 mm^2). The thermoelectric model, which also implements a lumped resistor as a resistive load, is developed with 92,427 DOF. However, the thermoelectric TEG model involves nonlinearity because the current generated has an impact on the temperature distribution in the TEG and vice versa, in which case MOR is not applicable. Hence, a solely thermal TEG model with 84,093 DOF is constructed, which implements the Peltier heat by means of an effective thermal conductivity κ_e expressed as follows:

$$\kappa_e = \kappa'(1 + \frac{ZT'}{1+m}) \,, \tag{7.24}$$

where ZT' is the equivalent figure of merit of the thermopile and m is the resistance coefficient between load resistance (R_{load}) and internal resistance of TEG (R_{TEG}). ZT' and m are defined as follows:

$$ZT' = \frac{\alpha'^2 \cdot T_{avg}}{\kappa' \rho'} \,, \tag{7.25}$$

$$m = \frac{R_{load}}{R_{TEG}} \,, \tag{7.26}$$

where T_{avg} is the average temperature at the top (T_{top}) and bottom (T_{bot}) surfaces of the

[5]The Equations from 7.21 to 7.28 are derived by Yongchen Rao, the first author in [98].

representative thermopile. R_{TEG} is the resistance of the total thermocouple legs. They are calculated as follows:

$$T_{avg} = \frac{1}{2} \cdot (T_{top} + T_{bot}) \,, \tag{7.27}$$

$$R_{TEG} = \frac{n}{2} \cdot \frac{(\rho_{el,p} + \rho_{el,n}) \cdot l}{a} = \frac{n^2}{2\eta} \cdot \frac{(\rho_{el,p} + \rho_{el,n}) \cdot l}{A} \,, \tag{7.28}$$

where A is the cross-sectional area of the ceramic plate and l is the height of the representative thermopile. a is the cross-sectional area of a single thermocouple leg, which can be derived from Equation (7.6). As a consequence, substituting Equations (7.22) and (7.25) in Equation (7.24), the effective thermal conductivity κ_e can be rewritten as follows:

$$\kappa_e = \eta \cdot \underbrace{\frac{\kappa_p + \kappa_n}{2}}_{\kappa_{ini}} + \underbrace{\frac{\alpha'^2 \cdot T_{avg}}{\rho' \cdot (1+m)}}_{\kappa_{peltier}} \tag{7.29}$$

where κ_{ini} is the original thermal conductivity of the thermoelectric material bismuth telluride and $\kappa_{peltier}$ is the additional effective thermal conductivity caused by the Peltier effect when the current flows through the TEG. It can be observed in Equation (7.29) that the effective thermal conductivity κ_e and average temperature T_{avg} depend on each other. Therefore, the model is solved iteratively. In this case, starting with κ_e evaluated at $T_{avg} = 27$ K, the sequence κ_e is updated in the model until convergence is obtained. As presented in [98], the thermal TEG model with effective thermal conductivity is proven as accurate as the thermoelectric model. Another advantage of using the thermal TEG model is that the dimensions of the thermal model are less than the thermoelectric model due to the fact that only a thermal domain is considered during the simulation.

It is also obvious in Equation (7.29) that if one can generate a pROM of the TEG with thermal conductivity as the parameter, the iterative simulation can be sped up significantly. In this case study, a pROM of the thermal TEG, which is similar to the pROM presented in Equation (7.11), is generated. The average temperature results from the top and bottom surfaces of the representative thermopile are set as the outputs. Based on the pROM, the electrical load circuit is constructed as illustrated in Figure 55. The value of the additional effective thermal conductivity caused by the Peltier effect ($\kappa_{peltier}$) is updated with the temperature outputs from the pROM.

The parametric simulations of the electrical load circuit are performed by choosing different parameter values of fill factor ($\eta = \{1, 1.4, 1.9, 2.7, 3.7, 5.2, 7.2, 10, 13.9, 19.3, 26.9, 37.2, 51.8, 71.9, 100\}$ %) and load resistance ($R_{load} = \{0.1, 0.15, 0.21, 0.3, 0.5, 0.7, 1, 1.5, 2.2, 3.2, 4.8, 7.1, 10.2, 15.1, 22.4\}$ Ω). The inputs $\bar{q}_{eva} = -12$ W/m^2, $T_a = 37$ °C, $h = 8.58$ W/m^2/K, and $T_{amb} = 21$ °C are applied. $Q_m = \{498.5, 279.8, 841.6\}$ W/m^3 are used as the metabolic heat generation rates in muscle, fat, and skin layers. Initial thermal conductivity value $\kappa_{ini} = 1.6$ W/m/K is used. The power outputs of TEG obtained from both full and reduced thermal models are compared in Figure 56. It is observed that the

results obtained from the pROM match excellently with the results from the full-scale thermal model. The relative error caused by the pROM ranges from -1.4 % to 1.5 % and the average absolute relative error is 0.9 %, which means that the pROM of the thermal TEG model is a good approximation of the original full-scale system.

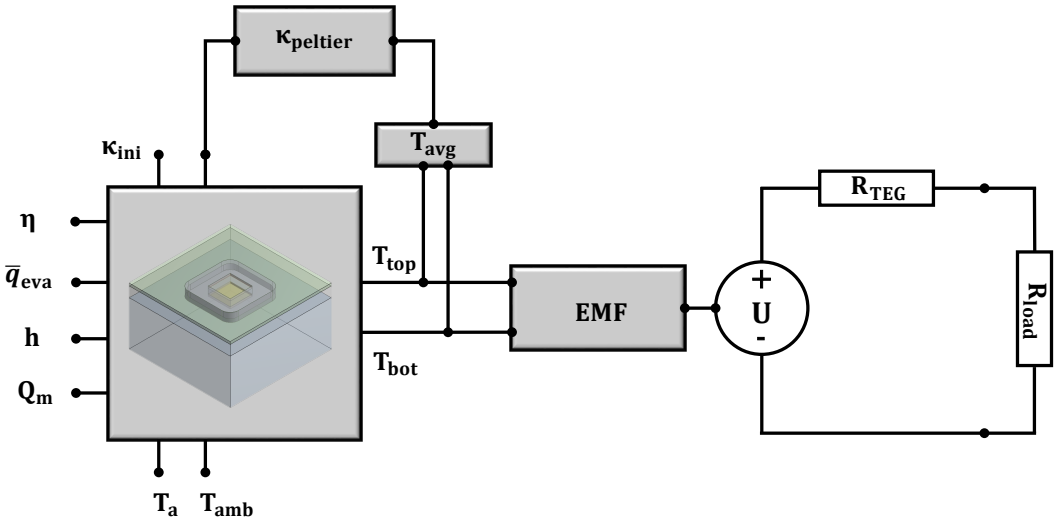


Figure 55: Electrical load circuit constructed based on the pROM of the TEG thermal model. The additional effective thermal conductivity $\kappa_{peltier}$ is calculated based on the temperature outputs from the pROM. This figure is adapted from [98].

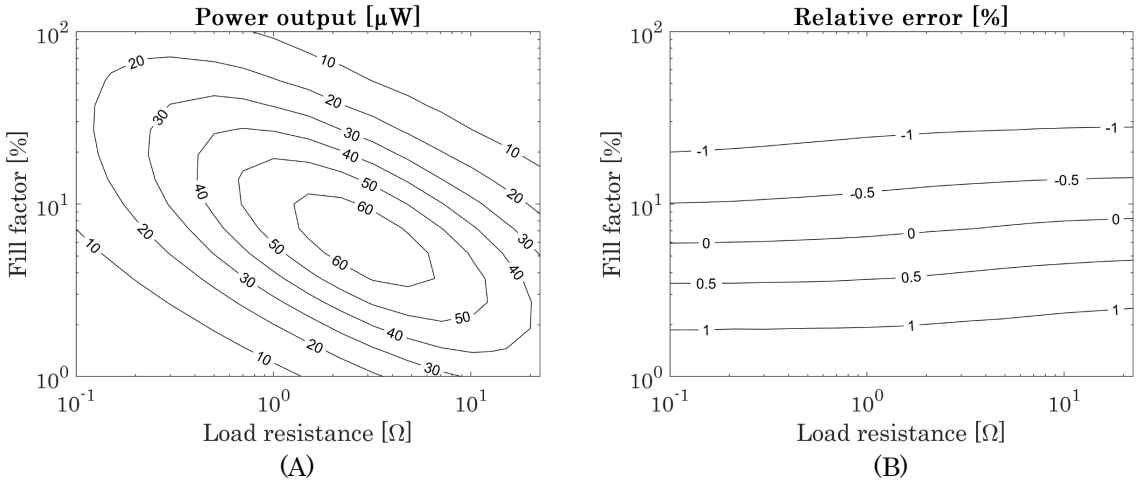


Figure 56: (A) Power output results from the thermal pROM. (B) Relative error of the power output results between the full thermal model and its pROM. These figures are adapted from [98].

The computational time for the parametric simulations of the full and reduced TEG thermal models are shown in Table 14. Based on the pROM, the integration of 225 groups of parametric simulations takes only a few seconds. Compared to the parameterized full-scale TEG thermoelectric and thermal models, the pROM provides a significant speed-up of the computational time. Therefore, it is concluded that the small dimension pROM is highly accurate and efficient for electronic circuit simulation at the system level.

Table 14: Computational time of 225 groups of parametric simulations between full and reduced TEG models (Intel®Xeon Processor E5-4650 2.7 GHz, 128 GB RAM).

Model	Full thermoelectric	Full thermal	Thermal pROM
System dimension	92,427 DOF	84,093 DOF	10 DOF
Time for pMOR	n/a	n/a	19.5 s
Parametric analysis	7,875 s	5625 s	1.3 s

Chapter 7 Summary

This chapter presents the application of pMOR methods to a miniaturized TEG model, which is embedded in a simplified cuboid human tissue model and a realistic human torso model. Thermal analysis is utilized and the material properties are all temperature-independent for efficiency and applicability of MOR and pMOR methods.

- The fill coefficient and the ambient temperature in the convection BC are defined as the parameters to investigate the impact of environmental conditions on TEGs. In this case, the film coefficient can be represented symbolically in front of the thermal conductivity matrix and the ambient temperature can be set as an input. The pROM is generated via the multivariate moment matching-based pMOR method.

- The fill factor is introduced to represent the thermal effect of varying the geometry of the thermocouple legs in TEGs. The pROM with thermal conductivity as the parameter is constructed via the Multivariate moment matching-based pMOR.

- For the TEG model with a geometrical parameter, e.g., the height of the thermopile, MI- and AP-based pROMs are both applied to construct a highly accurate and compact pROM. The comparison of these two methods shows that when using the local ROMs and snapshot matrices sampled at the same locations, the pROM generated via AP-based pMOR is more accurate. However, MI-based pMOR takes less time to construct the pROM.

- The framework of combining pMOR with the submodeling technique is introduced. A human torso global model within a representative TEG is parametrically reduced and the results from the pROM are employed as the cut-boundaries of a submodel, which is a detailed TEG model surrounded by a small part of the human tissue.

The analysis of a TEG thermoelectric model, which implements a lumped resistor as a resistive load and considers the Peltier effect in the thermopile, is also presented. For the efficiency and applicability of MOR and pMOR methods, a thermal model implements the thermoelectric effects by means of an effective thermal conductivity. Then the pROM is constructed to speed up the simulation and the Peltier effect can be considered via system-level simulation.

8 Parametric Model Order Reduction of Electromagnetic Energy Harvester

In this chapter, a compact model from the electromagnetic full-order model (FOM) described in section 5.3 is first presented. In section 8.1, the compact model is simulated efficiently and demonstrated in the system-level simulation. Furthermore, in sections 8.2 and 8.3, to reduce the generation time of the compact model, the matrix interpolation (MI)- and algebraic parameterization (AP)-based parametric model order reduction (pMOR) methods described in sections 3.2 and 3.3 are utilized. A highly accurate parametric reduced-order model (pROM), which enables variation of the parameters at the reduced-order model (ROM)-level for parametric studies, is constructed. The results and computational efficiency of these two pMOR methods are compared.

8.1 Compact Model for System-Level Simulation

For the electromagnetic energy harvester model presented in section 5.3, the transient simulation of the three-dimensional (3D) model in ANSYS Maxwell 3D is time-consuming. It takes around 135 minutes to perform the transient simulation throughout 8.5 ms with a time step of 0.5 ms (Intel®Core Processor (Broadwell, IBRS) 3.0 GHz, 64 GB RAM). Therefore, to reduce the computational cost, a compact model must be generated for efficient system-level simulations. In this section, the compact model of the original full-scale finite element (FE) electromagnetic model, which is generated via the technique presented in Appendix G, will be introduced for fast simulations at the system level.

The compact model is constructed based on a look-up table as presented in Figure 57. In this look-up table, the positions of the magnets and the current in the coil are used as the inputs. The magnetic flux through the coil is the output, which is obtained from a parametric solution of the model. The interpolation method is deployed to calculate the magnetic flux outputs from those input values not listed in the look-up table. The compact model is then imported into the software ANSYS Twin Builder and is connected to both electrical and mechanical circuits for real-time system-level simulations. As shown in Figure 57, the dynamics of the harvester are imitated by lumped mechanical elements. $m = 22.2$ µg is the mass component, which presents the inertia of two magnets on one side of the coil. It is excited with $f = 60$ Hz as an excitation frequency. Position source, damping, and spring components determine the excitation applied to the magnets. In this case study, the spring rate and the excitation amplitude are calculated as follows:

$$k = \omega^2 m = (2\pi f)^2 m = 3.16 \text{ N/m} \tag{8.1}$$

$$y_0 = \frac{a_0}{\omega^2} = \frac{a_0}{(2\pi f)^2} = 4.15 \text{ µm} \tag{8.2}$$

where $a_0 = 0.59$ m/s^2 is the acceleration amplitude. The damping coefficient is chosen to be 6.8×10^{-5} Ns/m to yield the designated oscillation amplitude of 570 µm. Thus, the quality factor of the harvester model becomes 137.3.

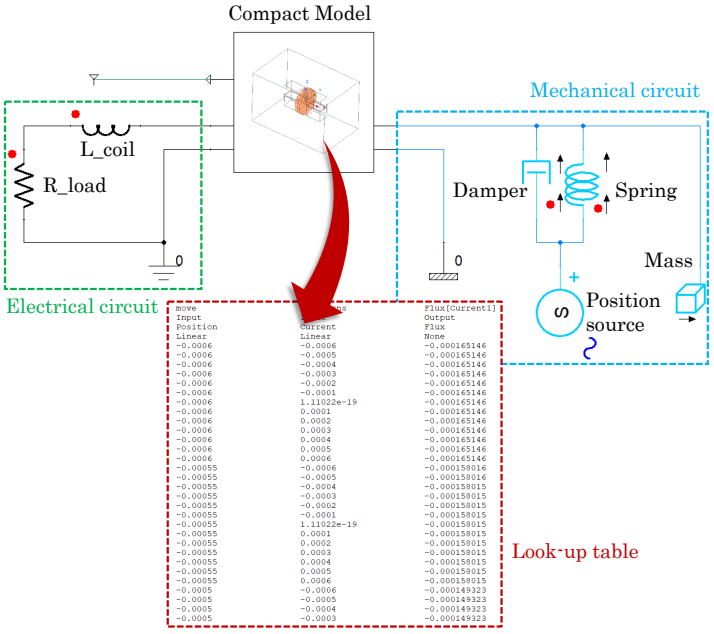
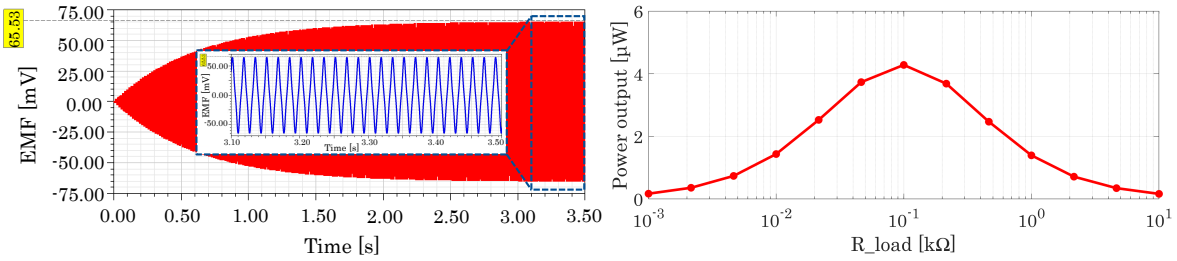


Figure 57: The compact model is constructed on the basis of the look-up table of the parametric simulation results. It is further connected to both electrical and mechanical circuits at the system level. This figure is adapted from [99].

In addition, an electrical circuit, which is composed of a load resistor and a coil inductor, is also connected to the compact model. The resistance of the load resistor is defined as 10 GΩ, which sets an open circuit condition. The resulting voltage induced in the coil simulated from the compact model is shown in Figure 58 (A). The maximum voltage output is obtained at around 65.5 mV, which is close to the expected value of 64 mV as presented in [16]. Then the parametric studies of the load resistance can be performed efficiently on the basis of the compact model. As shown in Figure 58 (B), the maximum power output of 4.28 µW is obtained when the load resistance matches the internal resistance of the coil.

It can also be found in Table 15 that the computational time for the simulation of the compact model is much less than for the finite element model. It takes only 19 s to perform the simulation of the compact model with a simulation period of 3.5 s and time step 1 µs, while it took around 135 minutes to simulate the full-scale FE model with a simulation period of 8.5 ms and time step 0.5 ms. Although the compact model enables efficient system-level simulation, generation of the compact model requires 244 minutes, as shown in Table 15. This effort stems from the necessity to solve 325 parameter sets of the full-scale FE electromagnetic energy harvester model. In this case study, the position of the magnets is parameterized between -0.6 and 0.6 mm with a step size of 0.05 mm. The coil current is parameterized between -0.6 and 0.6 mA in steps of 0.1 mA.



(A) (B)

Figure 58: (A) The outer plot shows the open circuit voltage obtained from the compact model with a simulation period of 3.5 s. The inner plot shows the voltage from 3.1 to 3.5 s. (B) Power dissipation in the load resistor. These figures are adapted from [99].

Table 15: Comparison of the computational time between finite element model and compact model (Intel®Core Processor (Broadwell, IBRS) 3.0 GHz, 64 GB RAM).

Model	Model Generation	Period	Time Step	Comput. Time
Finite element	n/a	8.5 ms	0.5 ms	135 min
Compact model	244 min†	3.5 s	1 µs	19 s

† 325 groups of parametric solutions.

For compact model generation and design optimization of the device, efficient parametric studies of the original FE model with geometrical parameters are necessary. Therefore, in the following sections 8.2 and 8.3, pMOR methods are suggested to reduce the computational effort of parametric studies. More specifically, pMOR methods will enable us to parameterize the geometrical parameters of the electromagnetic energy harvester model at the reduced-order model level. These parametric solutions calculated from the pROM can be saved in the look-up table, which can be used to construct the compact model introduced in this section for system-level simulations. Moreover, the pROM can be transformed into a system-level model in other formats, e.g., a VHDL-AMS model, with conservative ports. This could be done and needs to be validated in the future.

8.2 Position Parametrization

According to the technique introduced in Appendix G, a compact model can be generated from parametric solutions, where the position of the magnet is set as the geometrical parameter. In this subsection, the results from the pROM constructed through the MI-based pMOR method will be shown.

A simplified two-dimensional (2D) permanent magnet model built in ANSYS Mechanical APDL is used (see Figure 59). The magnet is 1.5×1 mm^2 in size. The air region around the magnet is separated into four parts to generate a regular mesh and connected using the ANSYS parametric design language (APDL) command '`CPINTF`'. When the magnet is at the initial position of 0 mm, both the top and bottom air regions are in

size 8×3.5 mm^2. In order to implement the MI-based pMOR method, the position of the magnet is changed while preserving the mesh topology via scaling the elements in the top and bottom regions. APDL command '`ARSCALE`' is used to scale the elements.

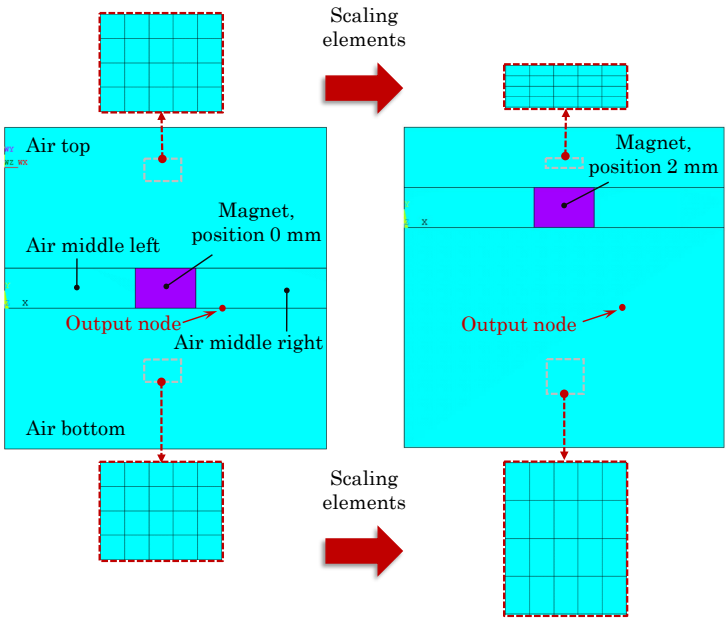


Figure 59: Preserving the mesh topology of the 2D permanent magnet model while changing the position of the magnet via scaling the mesh in the top and bottom air regions. An output node is selected from a fixed position in the bottom air region. This figure is adapted from [99].

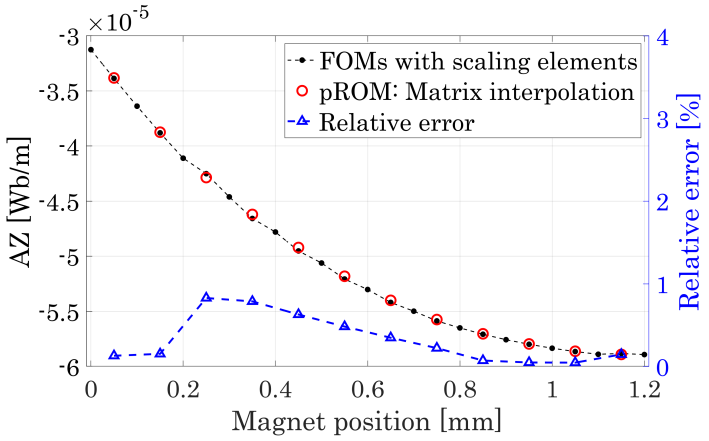


Figure 60: Simulation results of the pROM obtained from MI-based pMOR method. The local ROMs are constructed at $l = \{0, 0.1, 0.2, \cdots, 1.2\}$ mm and the pROMs are interpolated at $l = \{0.05, 0.15, 0.25, \cdots, 1.15\}$ mm. This figure is adapted from [99].

In this case study, the model in Equation (5.23) is solved and the output is the magnetic vector potential (AZ) from a selected node. The local ROMs are constructed via parameterizing the position of the magnet between 0 and 1.2 mm with step 0.1 mm. The results at the positions in between are calculated from the pROM. Figure 60 shows that

the pROM produces reliable results with the maximum relative error of 0.8 %.

8.3 Size Parametrization

The previous section 8.2 presents how to preserve the topology of the mesh via scaling the elements and using the MI-based pMOR method to construct the pROM from a 2D permanent magnet model with parameterized position of the magnet. It is demonstrated in this subsection how to derive a pROM through MI-based pMOR method while parameterizing the size of the magnet. The position of the magnet is then fixed and the height of the magnet is defined as the geometrical parameter instead. In line with section 8.2, the mesh topology of the model is fully controlled by scaling the elements in the magnet and the two middle air regions as shown in Figure 61.

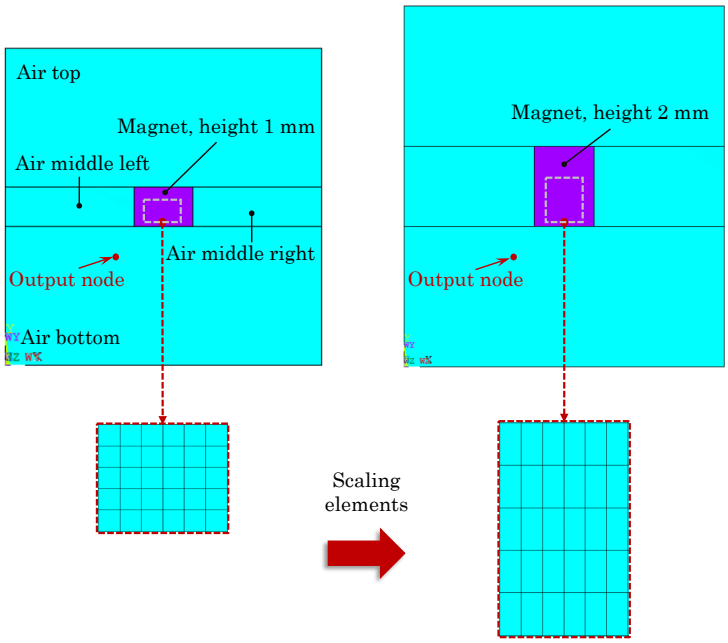


Figure 61: Preserving the mesh topology of the 2D permanent magnet model while changing its height via scaling the mesh in the magnet and the two air regions beside the magnet. An output node is selected from a fixed position in the bottom air region. This figure is adapted from [99].

The height values of the magnet $l = \{1,\ 1.1,\ 1.2\}$ mm are selected initially for constructing the local ROMs. As shown in Figure 62, the pROM constructed by the MI-based pMOR method is more accurate when the pROM is interpolated with the geometrical parameter between 1 and 1.2 mm. Furthermore, instead of using 3 local ROMs, the accuracy of the interpolated pROM between 1.2 and 1.5 mm can be improved significantly when the pROM is interpolated with 6 local ROMs selected at $l = \{1,\ 1.1,\ 1.2,\ 1.3,\ 1.4,\ 1.5\}$ mm. The average relative error is reduced from 0.7 % to $1.6{\times}10^{-4}$ %. These results indicate that for the MI-based pMOR method, the accuracy of the pROM is highly reliant on the local ROMs and decays as the geometrical parameter deviates.

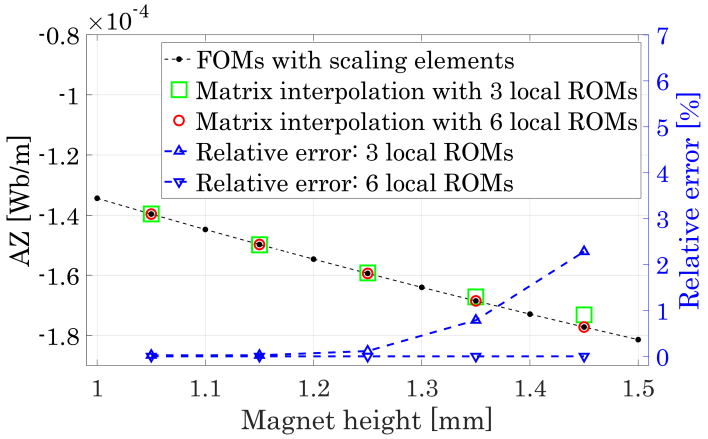


Figure 62: Comparison of the magnetic vector potential results between FOMs and pROMs. The pROMs are obtained via matrix interpolation with 3 and 6 local ROMs, respectively. This figure is adapted from [99].

In this case study, the computational efficiencies are also compared and the limitations of the MI- and AP-based pMOR methods are investigated based on both the 2D magnet model and the 3D electromagnetic energy harvester model, while the size of the magnet is set as the geometrical parameter. Firstly, both methods are implemented on the 2D magnet model. So far the following setups for the generation of the pROMs have been discussed:

- For implementing the MI-based pMOR method, the height of the magnet is selected at $l = \{1.0, 1.1, 1.2\}$ mm. These FOMs are reduced and then used as local ROMs for interpolation.

- For implementing the AP-based pMOR method, three snapshot system matrices $K(l)$ are taken at $l = \{1.0, 1.1, 1.2\}$ mm to construct the geometrical parameter-independent matrices $K_{\frac{1}{\alpha}}$, K_1, and K_α.

For a fair comparison, the same height values are selected for constructing both the local ROMs and the snapshot matrices. The results from the selected output node are obtained from the pROMs within $l = \{1.05, 1.15, 1.25, 1.3, 1.35, 1.4, 1.45, 1.5\}$ mm. Comparison of the results is presented in Figure 63. It is observed that both pROMs produce reliable results. The results from the pROM constructed by the AP-based pMOR method are more precise. The average relative error is $6.68{\times}10^{-4}$ % compared to 1.1 % obtained from the pROM as obtained by the MI-based pMOR method. The computational cost for the generation of the pROMs is presented in Table 16. It is observed that using the MI-based pMOR method is more efficient than using the AP-based pMOR method to construct the pROM. Although the MI-based pMOR method requires constructing the local ROMs, it still takes less time to generate the pROM, since the AP-based pMOR method needs to construct and store the geometrical parameter-independent system matrices at full-scale size.

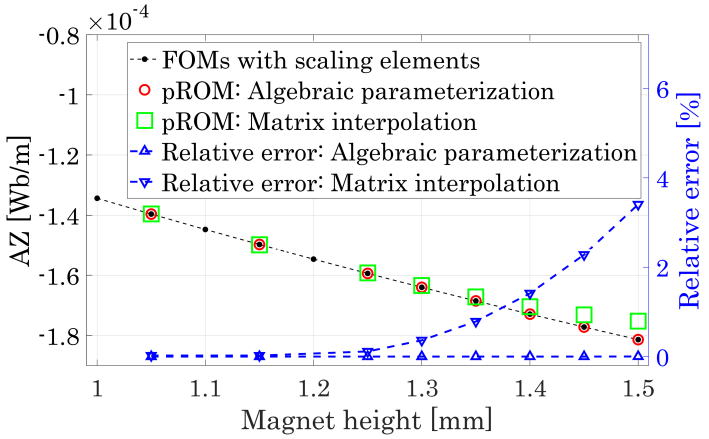


Figure 63: Comparison of the magnetic vector potential results between FOMs and pROMs. The pROMs are obtained via MI- and AP-based pMOR methods, respectively. This figure is adapted from [99].

Table 16: Comparison of the computational time for generating the pROMs from the 2D magnet model and performing parametric studies (Intel®Core i5-7600 3.5 GHz, 32 GB RAM).

	FOM	**pROM: matrix interpolation**	**pROM: algebraic parameterization**
System dimension	75,354 DOF	1 DOF	39 DOF
Extract full system matrices	n/a	3×1.2 s	3×1.2 s
Generate the pROM	n/a	5.7 s	30.3 s
Parametric simulation	8×18.6 s	8×1.8 ms	8×2.9 ms
Total time	148.8 s	9.3 s	34 s

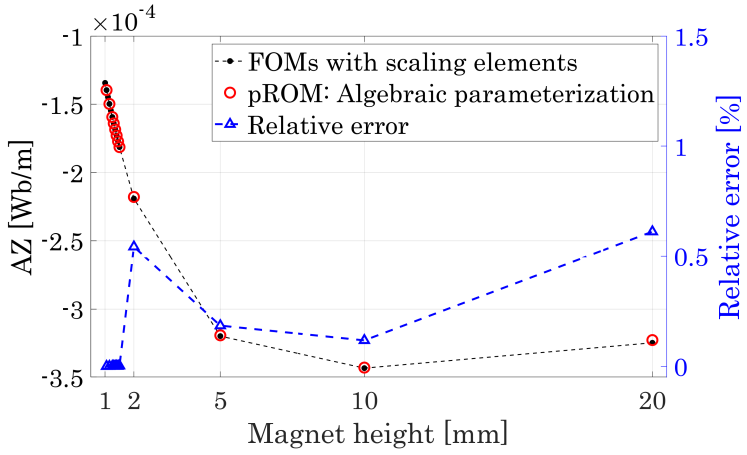


Figure 64: Simulation results of the pROM obtained from AP-based pMOR method in a parameter range between 1 and 20 mm. This figure is adapted from [99].

In addition, investigating the accuracy of the pROMs within a large parameter range is also of interest. As shown in Figure 64, the pROM generated from the AP-based pMOR method gives accurate results even when the parameter value is selected between 1 and

20 mm. The maximum relative error is 0.6 %. This is because, through the algebraic parameterization method, the geometrical parameter l is extracted from the system matrix $K(l)$ during the discretization process of the finite element method (FEM). However, it should be noted that the FOM results used for comparison in Figures 62, 63, and 64 are obtained by scaling the mesh elements. This method degenerates the mesh and influences the accuracy of the simulation results, as shown in Figure 65. The relative error is smaller than 5% only when scaling the height of the magnet from 1 to 5 mm. One has to take this into account in order to ensure reliable solutions.

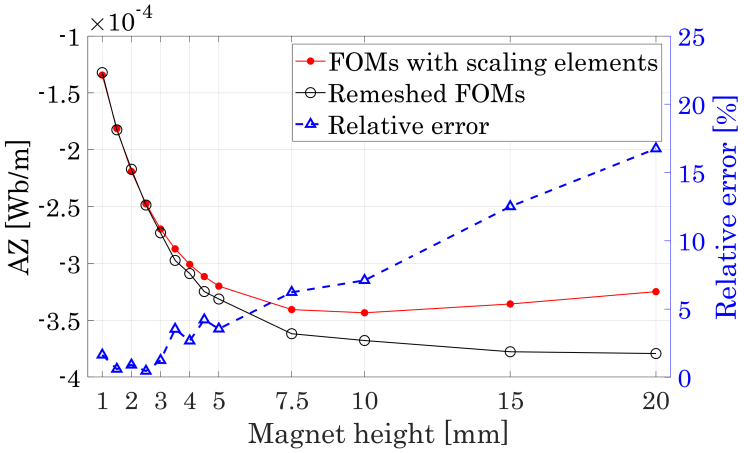


Figure 65: Simulation results from the FOM with and without scaled mesh. Scaling the elements degenerates the mesh and decreases the accuracy of the FOM. This figure is adapted from [99].

As presented in sections 8.2 and 8.3, the 2D permanent magnet model implemented in ANSYS Mechanical APDL allows one to change the geometry of the model while preserving the mesh topology via scaling the elements. Therefore, the 3D electromagnetic energy harvester model with a more complex geometry is also implemented in ANSYS Mechanical APDL (see Figure 66). The main structure of the electromagnetic energy harvester contains a coil and two groups of magnets as introduced in section 5.3. They are surrounded by a $5\times5\times7.5$ mm^3 air region, which is separated into smaller blocks in order to generate the box elements in the mesh. These box elements are prepared for convenient element scaling. The height of the magnets and their surrounding air block are scaled by using the APDL command '`VLSCALE`'. Then, the separated parts are connected by using the APDL command '`CEINTF`'. The accuracy of the simulation resulting from the FOM with scaling elements is validated in Figure 67. The height of the magnets is changed from 1 to 1.6 mm and the average relative error is 0.9 %. Then, both MI- and AP-based pMOR methods are implemented on the 3D electromagnetic energy harvester model, respectively. The computational cost of generating the pROMs is shown in Table 17. It is observed that the results from $l = \{1.05, 1.15, 1.25, 1.35, 1.45, 1.55\}$ mm are calculated from the pROMs. For the AP-based pMOR method, it takes a considerable time to generate the geometrical parameter-independent matrix with dimension 188,765 DOF.

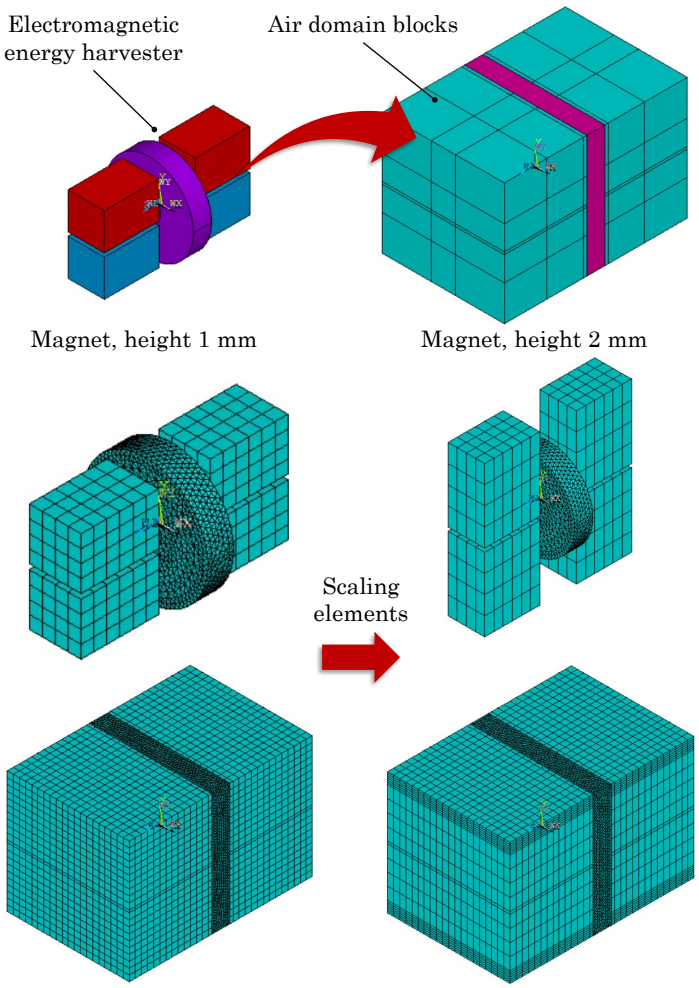


Figure 66: Electromagnetic energy harvester model built in ANSYS Mechanical APDL. The height of the magnets is parameterized while preserving the mesh topology via scaling the elements. This figure is adapted from [99].

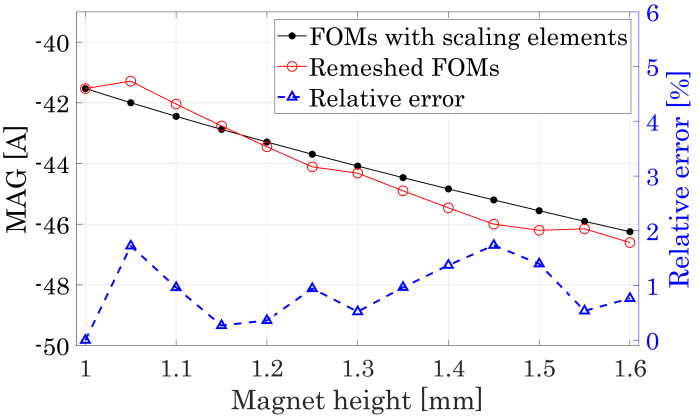


Figure 67: Changing the height of the magnets from 1 to 1.6 mm via scaling the elements. Comparison of the results between the FOMs of the 3D electromagnetic energy harvester model with scaling elements and the remeshed FOMs. This figure is adapted from [99].

Table 17: Comparison of the computational time for generating the pROMs from the 3D electromagnetic energy harvester model via MI- and AP-based pMOR methods and performing parametric studies (Intel®Core i5-7600 3.5 GHz, 32 GB RAM).

	FOM	**pROM: matrix interpolation**†	**pROM: algebraic parameterization**‡
System dimension	188,765 DOF	2 DOF	38 DOF
Extracting system matrices	n/a	4×5.9 s	3×5.9 s
Generate the pROM	n/a	77.3 s	405.2 s
Parametric simulation	3×89 s	1.9 ms	Unsolvable, singular matrix $K_r(\frac{1}{\alpha}, \alpha)$
Total time	267 s	101 s	422.9 s

† Local ROMs are constructed at $l = \{1.0, 1.1, 1.2, 1.3\}$ mm.
‡ Snapshot matrices are constructed at $l = \{1.0, 1.1, 1.2\}$ mm.

In addition, it is found that the pROM generated from this method is unsolvable in this case study because the generated reduced matrix $K_r(\frac{1}{\alpha}, \alpha)$ is singular. The pROM can only be solved when it is obtained from the MI-based pMOR method, and comparison of the magnetic scalar potential (MAG) results between the full and the reduced models are presented in Figure 68.

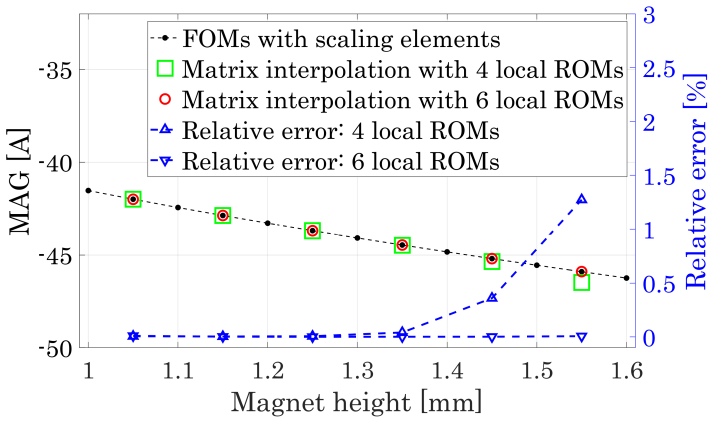


Figure 68: Comparison of the results between the 3D electromagnetic energy harvester FOM and the pROM obtained via MI-based pMOR method. The interpolated pROMs are generated based on 4 local ROMs at {1.0, 1.1, 1.2, 1.3} mm and 6 local ROMs at {1.0, 1.1, ..., 1.6} mm, respectively. This figure is adapted from [99].

Chapter 8 Summary

In this chapter, the electromagnetic energy harvester model is implemented in the FEM-based software ANSYS Maxwell 3D. Through the ECE technique, a look-up table-based compact model is generated to replace the original FOM for simulations at the system level. The look-up table is constructed based on the parametric solutions of the FOM.

pMOR methods are utilized to speed up the simulation time of parametric solutions.

- The position and the size of the magnets are parameterized, respectively.
- MI- and AP-based pMOR methods are implemented to generate the geometrical parameter-independent pROMs.
- Through the test case of a 2D single magnet model, it is found that when sharing the same number of local ROMs and snapshot system matrices, the MI-based pMOR method is more efficient, but the AP-based pMOR method gives a more accurate pROM, even in a large parameter range.
- When applying AP-based pMOR methods to a 3D electromagnetic energy harvester model, it is found that the reduced system matrix $K_r(\frac{1}{\alpha}, \alpha)$ generated is singular and the pROM is unsolvable. Further research is required on this topic.

Part IV

Conclusions and Outlook

9 Conclusions and Outlook

In this chapter, the main achievements of this work are summarized and its conclusions are discussed in section 9.1. Potential future research topics about parametric model order reduction of miniaturized energy harvesting modules are proposed in section 9.2.

9.1 Conclusions

This thesis has presented the modeling and simulation of piezoelectric, thermoelectric, and electromagnetic energy harvesting devices using finite element method (FEM)-based software ANSYS. Prior to fabricating prototypes, finite element (FE) simulation is highly recommended, as it significantly reduces the cost and improves the efficiency of the development cycle. Based on the supplement tool 'Model Reduction inside ANSYS', the Krylov subspace-based model order reduction (MOR) methods were applied for the reduction of large-scale linear models, which further sped up the simulation of the FE models. The main contribution of this thesis is further development of algorithms available within 'Model Reduction inside ANSYS' towards stable parametric model order reduction (pMOR) for multi-physical FE models with parameters in boundary conditions, material, and geometry.

The Krylov subspace-based MOR methods are successful for single-domain linear models. However, applying the conventional MOR methods to piezoelectric models leads to an unstable reduced-order model (ROM). Currently, there exist three stable MOR approaches, *MOR after Schur*, *Schur after MOR*, and *multiphysics structure-preserving MOR*. In this work, a further method *MOR after implicit Schur* is suggested, which is an improvement of *MOR after Schur*. These four stable MOR approaches have been demonstrated on different piezoelectric energy harvesters.

Through comparison of these four methods, it is observed that *MOR after Schur* leads to the highest computational cost due to the large number of entries in the full-order stiffness matrix obtained after the Schur transformation. *MOR after implicit Schur* only performs the Schur transformation of the input matrix and avoids the transformation of the full-order stiffness matrix. Hence, it preserves the sparse structure of the original full-order stiffness matrix, which to speeds up the reduction. A *multiphysics structure-preserving MOR* method preserves the block structure of all original full system matrices during the MOR process. Both *MOR after implicit Schur* and *multiphysics structure-preserving MOR* methods require less computational time than *MOR after Schur*. The most efficient method for stable compact modeling of a piezoelectric system is *Schur after MOR*, which performs the Schur transformation on the ROM instead. It is validated that all these stable MOR methods will generate accurate and stable piezoelectric ROMs.

As main contribution, various parametric model order reduction (pMOR) methods, e.g., multivariate moment matching-, matrix interpolation (MI)-, and algebraic parame-

terization (AP)-based pMOR methods, were researched further and implemented successfully for presented energy harvester devices to generate parametric reduced-order models (pROM) with parameterized boundary conditions (BCs), material properties, or geometry. Comparisons of the pMOR methods introduced in this thesis are presented in Table 18. It has been posited in the case studies of the thermoelectric generator (TEG) models that, the film coefficient in the convection BC and the thermal conductivity of the thermocouple legs can be extracted in front of the system matrices. The multivariate moment matching-based pMOR method was utilized to generate the pROM, which enables efficient parametric studies by changing the value of the parameters at the reduced model level. It should be noted that the MI-based pMOR method can be deployed to generate the pROM with these two parameters as well. However, in this work, we did not test it. The MI- and AP-based pMOR methods are the methods of choice for geometrically parameterized models. The MI-based pMOR method generates the pROM via interpolating the local ROMs with fixed values of the geometrical parameter. The AP-based pMOR method constructs a geometrical parameter-independent FOM. Based on that, the multivariate moment matching-based pMOR is implemented to generate the pROM.

Table 18: Comparison table of the pMOR methods for different kinds of parameters.

Parameter Type	**Multivariate Moment Matching**	**MI-based pMOR**	**AP-based pMOR**
Boundary condition	Applicable	Applicable[†]	n/a
Material property	Applicable	Applicable[†]	n/a
Geometrical parameter	Not directly applicable[‡]	Applicable	Applicable

[†] The MI-based pMOR method is applicable for the parameters in boundary conditions and material properties. However, it is more convenient to use the multivariate moment matching method in these cases.

[‡] On the basis of the geometrical parameter-independent FOM obtained via the AP-based pMOR method, the multivariate moment matching-based pMOR method is implemented to generate the pROM.

As presented in Table 19, both MI- and AP-based pMOR methods are applicable to the thermoelectric and electromagnetic energy harvesters. However, for the piezoelectric energy harvester, only the MI-based pMOR method is applicable. The framework of combining the four stable MOR approaches with the MI-based pMOR method has been presented for generating a stable piezoelectric pROM, in which the thickness of the piezoelectric patches is set as the geometrical parameter. The AP-based pMOR method is not applicable because the stable MOR approaches will change the original structure of the full-order system matrices, and then it would be pointless to perform the numerical scheme of constructing the geometrical parameter-independent matrices via the AP-based pMOR method.

Table 19: Comparison table of the pMOR methods for the geometrical parameters in different energy harvester devices.

Energy Harvester Type	MI-based pMOR	AP-based pMOR
Piezoelectric energy harvester	Applicable	n/a
Thermoelectric energy harvester	Applicable	Applicable
Electromagnetic energy harvester	Applicable	Applicable

For the thermoelectric and electromagnetic energy harvester models, it is found that the MI-based pMOR method is more efficient than the AP-based pMOR method when the numbers of local ROMs and snapshot system matrices are equal. This is because, for the AP-based pMOR method, the geometrical parameter is extracted from the system matrices on the basis of the full-order snapshot matrices. Although the method has been introduced to speed up the construction progress, it is still more time-consuming than the MI-based pMOR method. However, the pROM generated via the AP-based pMOR method is more accurate for a large parameter range while the pROM constructed via MI-based pMOR method is accurate only in the vicinity of local ROMs. Lastly, the advantages and disadvantages of the three pMOR methods are listed in Table 20.

Table 20: Advantages and disadvantages of the pMOR methods introduced in this work.

Methods	Advantage	Disadvantage
Multivariate moment matching	Simple implementation, Computationally efficient	Works only if the parameter can be extracted from the full matrix
MI-based pMOR	Nonlinear parameters, Computationally efficient	Accurate pROM only in vicinity of local ROMs
AP-based pMOR	Accurate pROM for large parameter range	High computational cost

9.2 Outlook

In the future, the MOR and pMOR methods will be implemented in other even more complex models to find the limitations of these methods. As section 8.3 implies, compared to the MI-based pMOR method, the computational cost of the AP-based pMOR method is significantly increased once it is implemented in the large-scale 3D electromagnetic energy harvester model. In addition, the reduced stiffness matrix generated via the AP-based pMOR method is singular and thus the pROM is unsolvable. These issues cannot be observed if these pMOR methods are applied to the simplified 2D permanent magnet model.

Another future research topic is the integration of the ROMs and pROMs in FE analysis. In section 7.4, it has been shown that the results from the pROM of the human

torso global model can be imported into the TEG FE submodel. However, this is only a one-way coupling, which means there is no bi-directional interaction between the global model and the submodel. Repeated imports are required once the results in the pROM are updated. To solve this problem, the super-element technique, which enables the connection of the ROMs and pROMs to the FE models, can be used. Recently, the Krylov-based super-element method in [100] demonstrates the possibility.

Finally, the MOR and pMOR methods described in this thesis can definitely be applied to reduce other components built in FEM-based software. In the future, the ROMs and pROMs can be implemented not only as virtual prototyping tools for efficient product design optimization but can also be utilized to construct virtual replicas of the entire system. The preliminary work on this subject is presented in sections 6.4, 7.5, and 8.1. Such a virtual system, the so-called digital twin [101,102], can provide transparent, readily accessible information, empowering predictive simulations and scenario analyses. The ROMs and pROMs require moderate computing resources and will offer a solution to the computational bottleneck of simulating extensive integrated systems and enable real-time monitoring, data filtering, and transmission. As far as I can see, such a topic is of high interest to multiple industrial and academic problems.

Appendices

A Material Properties

Table A1: Material properties of each part in the micro-fabricated piezoelectric energy harvester.

Material	**Density (kg/m^3)**	**Young's Modulus (GPa)**	**Poisson's ratio**
Silicon	2330	170	0.3
Aluminum nitride	2330	n/a	n/a

Table A2: Material parameters of AlN used in the micro-fabricated piezoelectric energy harvester.

	Dielectric Coefficients	**Piezoelectric Constants (N/V/m)**	**Elasticity Coefficients (10^{10} N/m^2)**
ϵ_{11}	1730	n/a	n/a
ϵ_{33}	1700	n/a	n/a
e_{31}	n/a	-5.4	n/a
e_{33}	n/a	15.8	n/a
e_{15}	n/a	12.3	n/a
${C_{11}}^E$	n/a	n/a	12.10
${C_{12}}^E$	n/a	n/a	7.52
${C_{13}}^E$	n/a	n/a	7.54
${C_{22}}^E$	n/a	n/a	11.10
${C_{21}}^E$	n/a	n/a	7.52
${C_{33}}^E$	n/a	n/a	12.10
${C_{44}}^E$	n/a	n/a	2.11
${C_{55}}^E$	n/a	n/a	2.11
${C_{66}}^E$	n/a	n/a	2.26

Table A3: Material properties of each part in the frequency-tunable piezoelectric energy harvester.

Material	**Density (kg/m^3)**	**Young's Modulus (GPa)**	**Poisson's ratio**
Stainless steel	7900	200	0.30
Ceramic film	7800	64	0.36

Table A4: Material parameters of PIC-255 used in the frequency-tunable piezoelectric energy harvester.

	Dielectric Coefficients	**Piezoelectric Constants (N/V/m)**	**Elasticity Coefficients (10^{10} N/m^2)**
ϵ_{11}	1649	n/a	n/a
ϵ_{33}	1750	n/a	n/a
e_{31}	n/a	-7.15	n/a
e_{33}	n/a	13.80	n/a
e_{15}	n/a	11.90	n/a
C_{11}^E	n/a	n/a	12.30
C_{12}^E	n/a	n/a	7.67
C_{13}^E	n/a	n/a	7.03
C_{22}^E	n/a	n/a	12.30
C_{21}^E	n/a	n/a	7.03
C_{33}^E	n/a	n/a	9.71
C_{44}^E	n/a	n/a	2.23
C_{55}^E	n/a	n/a	2.23
C_{66}^E	n/a	n/a	2.32

Table A5: Thermal and physical properties of human tissues from [92].

Tissue	**Density (kg/m^3)**	**Specific Heat (J/kg/K)**	**Thermal Conductivity (W/m/K)**	**Perfusion Rate (1/s)**	**Metabolic Heat Rate (W/m^3)**
Muscle	1090	3421	0.49	3.37×10^{-4}	498.52
Fat	911	2348	0.21	3.01×10^{-4}	279.80
Skin	1109	3391	0.37	9.06×10^{-4}	841.57
Blood	1050	3617	0.52	n/a	n/a
Bone (Cortical)	1908	1313	0.32	n/a	n/a
Kidney	1066	3763	0.53	n/a	n/a
Cartilage	1100	3568	0.49	n/a	n/a
Heart muscle	1081	3686	0.56	n/a	n/a
Liver	1079	3540	0.52	n/a	n/a
Lung	394	3886	0.39	n/a	n/a
Stomach	1088	3690	0.53	n/a	n/a
Trachea	1080	3568	0.49	n/a	n/a
Spinal cord	1075	3630	0.51	n/a	n/a

Table A6: Electrothermal material properties of TEG.

Components	**Material Type**	**Seebeck Coefficient (µV/K)**	**Thermal Conductivity (W/m/K)**	**Electrical Resistivity (Ω·m)**
p-type thermocouple legs	Bi_2Te_3	162	1.60	1.10×10^{-5}
n-type thermocouple legs	Bi_2Te_3	-162	1.60	1.10×10^{-5}
Ceramic plates	Al_2O_3	n/a	25	10^{14}
Housing	PEEK	n/a	0.25	3.15×10^{13}

Table A7: Material properties of the commercially available TEG implemented in section 7.4.

Components	**Material Type**	**Density (kg/m³)**	**Thermal Conductivity (W/m/K)**	**Specific Heat (J/kg/K)**
p-type thermocouple legs	Bi_2Te_3	7700	1.58 to 1.52†	90
n-type thermocouple legs	Bi_2Te_3	7700	1.62 to 1.58†	90
Ceramic plates	Al_2O_3	3720	25	880
Housing	PEEK	1300	0.25	1500

† Temperature-dependent thermal conductivity property in the temperature range between 25 and 37.5 °C.

Table A8: Material properties of copper and NdFeB.

Material Type	**Relative Permeability**	**Bulk Conductivity (S/m)**	**Magnetic Coercivity (A/m)**	**Mass Density (kg/m³)**
NdFeB (Magnets)	1.099779	6.25×10^{5}	8.90×10^{5}	7400
Copper (Coil)	0.999991	5.80×10^{7}	n/a	8933

B Python Scripts for Matrix Interpolation-based pMOR

This section presents the Python functions programmed in this work for the implementation of the matrix-interpolation (MI)-based parametric model order reduction (pMOR) method introduced in section 3.2.

```
""" MI functions in Python.
Requirements:
    Python 2.7 - http://www.python.org
    SciPy and NumPy for linear algebra - http://www.scipy.org
Variable '__all__' contains all function names.
"""
import numpy
__all__ = [
          'ConstructMatrixPool',
          'FilllocalProj',
          'CommonBasis1',
          'CommonBasis2',
          'TransformFirstOrderLocalSys',
          'TransformSecondOrderLocalSys',
          'CalculateWeights',
          'FirstOrderMI',
          'SecondOrderMI',
          'FirstOrderLagrangeMI']

def ConstructMatrixPool(V):
    """This function is used to append local projection
   matrices in a matrix pool.
        Argument:
            V: list of local projection matrices.
        Returns:
            V_pool: matrix pool constains all local projection
    matrices.
            n: number of local systems.
    """
    # Number of local systems
    n =len(V)
    V_pool=V[0]
    for i in range(n):
        if i<n-1:
            V_pool=numpy.append(V_pool,V[i+1],axis=1)
```

```
34      return V_pool,n
35
36  def FilllocalProj(V):
37      """This function is used to add zero elements in the
    local projection matrices.
38          Arguments:
39              V: the list of local projection matrices.
40          Returns:
41              V: the list of local projection matrices filled
    with zero elements.
42      """
43      # Number of local projection matrices
44      n =len(V)
45      # Dimension of the local reduced system
46      q=int(V[0].shape[1])
47      # Find the number of rows in local projection matrices
48      dim=numpy.zeros(n)
49      for i in range(n):
50          dim[i]=int(V[i].shape[0])
51      # Find the maximum number of rows in local projection
    matrices
52      dim_max=int(numpy.amax(dim))
53      # Equalizing the projection matrices
54      for i in range(n):
55          if dim[i]<dim_max:
56              dim_empty=int(dim_max-dim[i])
57              V_empty=numpy.zeros([dim_empty,q])
58              V[i]=numpy.append(V[i],V_empty,axis=0)
59      return V
```

Function '`ConstructMatrixPool`' is employed to construct the matrix pool V_{all} as shown in Equation (3.24) if the mesh topologies of the local models are the same. Otherwise, if the mesh topologies of the local models are changed with different values of the geometrical parameter, functions '`FilllocalProj`' and '`ConstructMatrixPool`' are implemented to create a matrix pool as presented in Equation (3.30). The local projection matrices are firstly filled with zero elements and then the matrix pool is constructed [47, 48].

```
64  def CommonBasis1(V_pool,V,n):
65      """This function is used to calculate the common basis
    for local systems. Matrix Mi and Ti are calculated from
```

```
    each local system.
66          Arguments:
67              V_pool: projection matrix pool.
68              V: the list of local projection matrices.
69              n: number of local systems.
70          Returns:
71              T: the list of matrices Ti.
72              M: the list of matrices Mi.
73              q: the dimension of the reduced model.
74      """
75      # SVD decomposition
76      u,s,vh=numpy.linalg.svd(V_pool,full_matrices=False)
77      # Build R by taking first q colums of u
78      q=int(V[0].shape[1])
79      Rv=u[:,0:q]
80      # Calculate different R,T,M matrices
81      R=[]
82      T=[]
83      M=[]
84      for i in range(n):
85          R.append(0)
86          T.append(0)
87          M.append(0)
88      for i in range(n):
89          R[i]=Rv[0:len(V[i]),:]
90      for i in range(n):
91          T[i]=numpy.dot(R[i].transpose(),V[i])
92      for i in range(n):
93          M[i]=numpy.linalg.inv(numpy.dot(V[i].transpose(),R[i
    ]))
94      return T,M,q
95
96  def CommonBasis2(V_pool,V,n):
97      """This function is used to calculate the common basis
    for local systems. Matrix M and T are calculated from one
    local system.
98          Arguments:
99              V_pool: projection matrix pool.
100             V: the list of local projection matrices.
101             n: number of local systems.
```

```
102         Returns:
103             T: matrix T calculated from the first projection
    matrix V[0].
104             M: matrix M calculated from the first projection
    matrix V[0].
105             + Any local projection matrices in V can be used,
    but you could only chose one of them. +
106             q: the dimension of the reduced model
107      """
108      # SVD decomposition
109      u,s,vh=numpy.linalg.svd(V_pool,full_matrices=False)
110      # Build R by taking first q colums of u
111      q=int(V[0].shape[1])
112      Rv=u[:,0:q]
113      # Calculate different R,T,M matrices
114      R=[]
115      T=[]
116      M=[]
117      for i in range(n):
118          R.append(0)
119          T.append(0)
120          M.append(0)
121      for i in range(n):
122          R[i]=Rv[0:len(V[i]),:]
123      for i in range(n):
124          T[i]=numpy.dot(R[i].transpose(),V[0])
125      for i in range(n):
126          M[i]=numpy.linalg.inv(numpy.dot(V[0].transpose(),R[i
    ]))
127      return T,M,q
```

Function '`CommonBasis1`' is implemented to construct the transformation matrices as presented in Equations (3.22) and (3.23), which are suggested in the 'classic' MI-based pMOR method [45]. Function '`CommonBasis2`' is used to construct the transformation matrices as shown in Equations (3.32) and (3.33), which are suggested in this work.

```
134 def TransformFirstOrderLocalSys(loc_matE,loc_matA,loc_matB,
    loc_matC,T,M,n,loc_init=None):
135      """This function is used to transform the first order
    local ROMs into another set of generalized coordinates.
136          Arguments:
```

```
137             loc_matE: list of matrices E in local ROMs.
138             loc_matA: list of matrices A in local ROMs.
139             loc_matB: list of matrices B in local ROMs.
140             loc_matC: list of matrices C in local ROMs.
141             T: list of matrices T.
142             M: list of matrices M.
143             n: number of local systems.
144             loc_init: initial state of the local system.
145         Returns:
146             matE,matA,matB,matC,init or matE,matA,matB,matC if
     no initial state.
147     """
148     # Transform local ROMs to another set of generalized
    coordinates
149     matE=[]
150     matA=[]
151     matB=[]
152     matC=[]
153     for i in range(n):
154         matE.append(0)
155         matA.append(0)
156         matB.append(0)
157         matC.append(0)
158     for i in range(n):
159         matE[i]=numpy.dot(M[i],numpy.dot(loc_matE[i],numpy.
    linalg.inv(T[i])))
160         matA[i]=numpy.dot(M[i],numpy.dot(loc_matA[i],numpy.
    linalg.inv(T[i])))
161         matB[i]=numpy.dot(M[i],loc_matB[i])
162         matC[i]=numpy.dot(loc_matC[i],numpy.linalg.inv(T[i]))
163     # When initial state exists
164     if loc_init is not None:
165         init=[]
166         for i in range(n):
167             init.append(0)
168         for i in range(n):
169             init[i]=numpy.dot(T[i],loc_init[i])
170         return matE,matA,matB,matC,init
171     else:
172         return matE,matA,matB,matC
```

```
173
174 def TransformSecondOrderLocalSys(loc_matM,loc_matE,loc_matK,
    loc_matB,loc_matC,T,M,n,loc_init=None):
175     """This function is used to transform the second order
    local ROMs into another set of generalized coordinates.
176         Arguments:
177             loc_matM: list of matrices M in local ROMs.
178             loc_matE: list of matrices E in local ROMs.
179             loc_matK: list of matrices K in local ROMs.
180             loc_matB: list of matrices B in local ROMs.
181             loc_matC: list of matrices C in local ROMs.
182             T: list of matrices T.
183             M: list of matrices M.
184             n: number of local systems.
185             loc_init: initial state of the local system.
186         Returns:
187             matM,matE,matK,matB,matC,init or matM,matE,matK,
    matB,matC if no initial state.
188     """
189     # Transform local ROMs to another set of generalized
    coordinates
190     matM=[]
191     matE=[]
192     matK=[]
193     matB=[]
194     matC=[]
195     for i in range(n):
196         matM.append(0)
197         matE.append(0)
198         matK.append(0)
199         matB.append(0)
200         matC.append(0)
201     for i in range(n):
202         matM[i]=numpy.dot(M[i],numpy.dot(loc_matM[i],numpy.
    linalg.inv(T[i])))
203         matE[i]=numpy.dot(M[i],numpy.dot(loc_matE[i],numpy.
    linalg.inv(T[i])))
204         matK[i]=numpy.dot(M[i],numpy.dot(loc_matK[i],numpy.
    linalg.inv(T[i])))
205         matB[i]=numpy.dot(M[i],loc_matB[i])
```

```
206         matC[i]=numpy.dot(loc_matC[i],numpy.linalg.inv(T[i]))
207     # When initial state exists
208     if loc_init is not None:
209         init=[]
210         for i in range(n):
211             init.append(0)
212         for i in range(n):
213             init[i]=numpy.dot(T[i],loc_init[i])
214         return matM,matE,matK,matB,matC,init
215     else:
216         return matM,matE,matK,matB,matC
```

The function 'TransformFirstOrderLocalSys' transforms the first-order local models as presented in Equation (3.21). Similarly, a second-order system can be transformed via function 'TransformSecondOrderLocalSys'.

```
217 def CalculateWeights(P,p):
218     """This algorithm is used to calculate the weights of
    only two nearby sample points.
219         Argument:
220             P: parameter values of the local systems, saved in
     an array. Normally, the sample points are chosen evenly.
221             p: interpolated point.
222         Return:
223             w: the weights, saved in an array. Sum(w) = 1.
224     """
225     # The first two paramters in P are used to calculate the
    distance
226     sample_dis = abs(P[0]-P[1])
227     # Number of local models
228     n=int(len(P))
229     # Calculate the distance of desired parameter from local
    ROMs
230     dis=numpy.zeros(n)
231     for i in range(n):
232         dis[i]=P[i]-p
233     # Calculate the weights of the parameters
234     w=numpy.zeros(n)
235     for i in range(n):
236         if abs(dis[i])>sample_dis:
237             # The only two parameters near to the desired
```

```
    parameter p is accounted
238             w[i]= 0
239         else:
240             w[i]=(sample_dis-abs(dis[i]))/sample_dis
241     return w
242
243 def FirstOrderMI(matE,matA,matB,matC,n,w,init=None):
244     """This function is used to do the matrix interpolation
    of the local ROMs (1st order).
245         Arguments:
246             matE,matA,matB,matC: matrices of local first order
     ROMs transformed in the common basis.
247             n: number of local ROMs.
248             w: weights of the local ROMs.
249             init: initial states of the local ROMs in the
    common basis. Default is None.
250         Returns:
251             matE_bar,matA_bar,matB_bar,matC_bar,init_bar:
    interpolated system
252     """
253     # Define the initial state of the weighted ROMs
254     matE_bar = matE[0]*w[0]
255     matA_bar = matA[0]*w[0]
256     matB_bar = matB[0]*w[0]
257     matC_bar = matC[0]*w[0]
258      # Interpolate the local ROMs
259     for i in range(n):
260         if i<n-1:
261             matE_bar = matE_bar+matE[i+1]*w[i+1]
262             matA_bar = matA_bar+matA[i+1]*w[i+1]
263             matB_bar = matB_bar+matB[i+1]*w[i+1]
264             matC_bar = matC_bar+matC[i+1]*w[i+1]
265     # When initial state exists
266     if init is not None:
267         init_bar = init[0]*w[0]
268         for i in range(n):
269             if i<n-1:
270                 init_bar = init_bar+init[i+1]*w[i+1]
271         return matE_bar,matA_bar,matB_bar,matC_bar,init_bar
272     else:
```

```
273         return matE_bar,matA_bar,matB_bar,matC_bar
274 
275 def SecondOrderMI(matM,matE,matK,matB,matC,n,w,init=None):
276     """This function is used to do the matrix interpolation
    of the local ROMs (2nd order).
277         Arguments:
278             matM,matE,matK,matB,matC: matrices of local second
     order ROMs transformed in the common basis.
279             n: number of local ROMs.
280             w: weights of the local ROMs.
281             init: initial states of the local ROMs in the
    common basis. Default is None.
282         Returns:
283             matE_bar,matA_bar,matB_bar,matC_bar,init_bar:
    interpolated system.
284     """
285     # Define the initial state of the weighted ROMs
286     matM_bar = matM[0]*w[0]
287     matE_bar = matE[0]*w[0]
288     matK_bar = matK[0]*w[0]
289     matB_bar = matB[0]*w[0]
290     matC_bar = matC[0]*w[0]
291      # Interpolate the local ROMs
292     for i in range(n):
293         if i<n-1:
294             matM_bar = matM_bar + matM[i+1]*w[i+1]
295             matE_bar = matE_bar + matE[i+1]*w[i+1]
296             matK_bar = matK_bar + matK[i+1]*w[i+1]
297             matB_bar = matB_bar + matB[i+1]*w[i+1]
298             matC_bar = matC_bar + matC[i+1]*w[i+1]
299     # When initial state exists
300     if init is not None:
301         init_bar = init[0]*w[0]
302         for i in range(n):
303             if i<n-1:
304                 init_bar = init_bar+init[i+1]*w[i+1]
305         return matM_bar,matE_bar,matK_bar,matB_bar,matC_bar,
    init_bar
306     else:
307         return matM_bar,matE_bar,matK_bar,matB_bar,matC_bar
```

```
308
309 def FirstOrderLagrangeMI(matE,matA,matB,matC,P,p,n,init=None):
310     """This function is used to do the matrix interpolation of the local ROMs (1st order) via Lagrange interpolation method.
311         Arguments:
312             matE,matA,matB,matC: matrices of local first order ROMs transformed in the common basis.
313             P: values of the parameter for local ROMs.
314             p: the values of the interpolated position. The interpolated ROM is calculated at this position.
315             n: number of local ROMs.
316             init: initial states of the local ROMs in the common basis. Default is None.
317         Returns:
318             matE_bar,matA_bar,matB_bar,matC_bar,init_bar: interpolated system.
319     """
320     # Define the numerator for each local ROM, w[p]=(p-p0)(p-p1)...(p-pn)
321     w_n = 1
322     for i in range(len(P)):
323         w_n = w_n*(p-P[i])
324     # Define the domenator for each local ROM, w'[pi]=(pi-p0)...(pi-p_i-1)(pi-p_i+1)...(pi-pn)
325     w_d = []
326     for i in range(len(P)):
327         w_d_temp = 1
328         for j in range(len(P)):
329             if i != j:
330                 w_d_temp = w_d_temp*(P[i] - P[j])
331         w_d.append(w_d_temp)
332     # Calculate the weight for each local ROM
333     w = []
334     for i in range(len(P)):
335         w_temp = w_n/((p-P[i])*w_d[i])
336         w.append(w_temp)
337     # Define the initial state of the weighted ROMs
338     matE_bar = matE[0]*w[0]
```

```
339     matA_bar = matA[0]*w[0]
340     matB_bar = matB[0]*w[0]
341     matC_bar = matC[0]*w[0]
342      # Interpolate the local ROMs
343     for i in range(n):
344         if i<n-1:
345             matE_bar = matE_bar+matE[i+1]*w[i+1]
346             matA_bar = matA_bar+matA[i+1]*w[i+1]
347             matB_bar = matB_bar+matB[i+1]*w[i+1]
348             matC_bar = matC_bar+matC[i+1]*w[i+1]
349     # When initial state exists
350     if init is not None:
351         init_bar = init[0]*w[0]
352         for i in range(n):
353             if i<n-1:
354                 init_bar = init_bar+init[i+1]*w[i+1]
355         return matE_bar,matA_bar,matB_bar,matC_bar,init_bar
356     else:
357         return matE_bar,matA_bar,matB_bar,matC_bar
```

The function '`CalculateWeights`' calculates the weights used for linear interpolation as presented in Equation (3.29). Based on the calculated weights, functions '`FirstOrderMI`' and '`SecondOrderMI`' are used to interpolate the first- and second-order systems, respectively. The Lagrange interpolation as shown in Equation (3.27) is deployed via function '`FirstOrderLagrangeMI`'.

C Python Scripts for Algebraic Parameterization-based pMOR

This section presents the Python functions programmed in this work for the implementation of the algebraic parameterization (AP)-based parametric model order reduction (pMOR) method introduced in section 3.3.

```
"""AP functions in Python
Requirements:
    Python 2.7 - http://www.python.org
    SciPy and NumPy for linear algebra - http://www.scipy.org
"""
from scipy import linalg
import scipy.io
import scipy.sparse
import numpy

def writeMatrix(name,mat,Format):
     """Write the matrix in coo format
     """
     f=open(name,'w')
     if isinstance(mat,type(numpy.array(1))):
          mat=scipy.sparse.coo_matrix(mat)
     if Format=='MatrixMarket':
          scipy.io.mmwrite(f,mat,precision=17)
     else:
          numpy.savetxt(f,mat,fmt='%.17e')
     f.close()

def AlgebraicParameterization(L1,L2,L3,mat1,mat2,mat3,name='full_system'):
    """ This function is used to generate the geometrical parameter independent diffusion matrices.
        Inputs:
            - L1,L2,L3: value of the geometrical parameter.
            - mat1,mat2,mat3: diffusion matrices snapshoted at corresponding parameter.
            - name: name of the full matrix files.
        Returns:
            - A_1/alpha: matrix multiplied by scalar 1/alpha.
            - A: matrix multiplied by scalar 1.
            - A_alpha: matrix multiplied by scalar alpha.
```

```
33            + The parameter-independent diffusion matrices
  are written in MatrixMarket format. +
34            - alpha1,alpha2,alpha3: scaling factors
35     """
36     if name=='':
37         name='full_system'
38     # Define the scaling factors
39     alpha1=L1/L1
40     alpha2=L2/L1
41     alpha3=L3/L1
42     # Define the inverse of the scaling factors
43     beta1=1/alpha1
44     beta2=1/alpha2
45     beta3=1/alpha3
46     # Get the dimension of the full system matrices
47     N=mat1.shape[0]
48     # Coefficients
49     s11=((alpha1*alpha2*alpha3)/((alpha1-alpha2)*(alpha1-
  alpha3)))
50     s12=(-(alpha1*alpha2*alpha3)/((alpha1-alpha2)*(alpha2-
  alpha3)))
51     s13=((alpha1*alpha2*alpha3)/((alpha1-alpha3)*(alpha2-
  alpha3)))
52     s21=(-(alpha1*alpha2**2-alpha1*alpha3**2)/((alpha1-alpha2
  )*(alpha1-alpha3)*(alpha2-alpha3)))
53     s22=((alpha2*alpha1**2-alpha2*alpha3**2)/((alpha1-alpha2)
  *(alpha1-alpha3)*(alpha2-alpha3)))
54     s23=(-(alpha3*alpha1**2-alpha3*alpha2**2)/((alpha1-alpha2
  )*(alpha1-alpha3)*(alpha2-alpha3)))
55     s31=((alpha1*alpha2-alpha1*alpha3)/((alpha1-alpha2)*(
  alpha1-alpha3)*(alpha2-alpha3)))
56     s32=(-(alpha1*alpha2-alpha2*alpha3)/((alpha1-alpha2)*(
  alpha1-alpha3)*(alpha2-alpha3)))
57     s33=((alpha1*alpha3-alpha2*alpha3)/((alpha1-alpha2)*(
  alpha1-alpha3)*(alpha2-alpha3)))
58     # Compute the geometrical parameter-independent diffusion
   matrices
59     A_1_alpha=s11*mat1+s12*mat2+s13*mat3
60     A_1=s21*mat1+s22*mat2+s23*mat3
61     A_alpha=s31*mat1+s32*mat2+s33*mat3
```

```
62      # Write the matrices in MatrixMarket format
63      writeMatrix(name+'.A_1_alpha',A_1_alpha,'MatrixMarket')
64      writeMatrix(name+'.A_1',A_1,'MatrixMarket')
65      writeMatrix(name+'.A_alpha',A_alpha,'MatrixMarket')
66      return alpha1,alpha2,alpha3,A_1_alpha,A_1,A_alpha
```

Function '`AlgebraicParameterization`' calculates the parameter-independent matrices presented in Equation (3.48). These three matrices '`A_1_alpha`', '`A_1`', and '`A_alpha`' are written in Matrix Market format. The three snapshot matrices '`mat1`', '`mat2`', '`mat3`' and the corresponding geometrical parameters '`L1`', '`L2`', '`L3`' are implemented as inputs. The coefficients $s_{11} \cdots s_{33}$ in Equation (3.48) are calculated based on the numerical scheme in Equation (3.47). In the following MATLAB script, the 'Symbolic Math Toolbox' is used to solve Equation (3.47) symbolically and the coefficients are collected in '`C_S1`', '`C_S2`', and '`C_S3`'.

```
1  % Calculate AP coefficients in MATLAB
2
3  % Setup
4  clear A1 A2 A3 S1 S2 S3 alpha1 alpha2 alpha3
5  syms A1 A2 A3 S1 S2 S3 alpha1 alpha2 alpha3
6
7  % Solve linear system for S1, S2 and S3
8  eqns=[1/alpha1*S1+S2+alpha1*S3==A1,
9        1/alpha2*S1+S2+alpha2*S3==A2,
10       1/alpha3*S1+S2+alpha3*S3==A3];
11 S=solve(eqns,[S1 S2 S3]);
12 sol=[S.S1;S.S2;S.S3]
13
14 % Collect coefficients
15 C_S1 = collect(S.S1,[A1 A2 A3])
16 C_S2 = collect(S.S2,[A1 A2 A3])
17 C_S3 = collect(S.S3,[A1 A2 A3])
```

D Options of 'Model Reduction inside ANSYS'

General command line format used for model order reduction.

```
mor_for_ansys -A full1 [full2] [-m full3 ···] [-options]
mor_for_ansys -M base_name [-options]
```

`-A`: read matrices from ANSYS FULL files (default).

`-M`: read matrices in the Matrix Market format.

Note that option '`-A`' is applied in default. Option '`-M`' enables the reduction of the system matrices obtained from other software.

Other common `[-options]`:

`-x`: expansion point (default is zero).

Use '`-x number`' to specify another expansion point. Use '`-x exp.txt`' if several expansion points need to be specified. The file '`exp.txt`' contains the following lines.

```
expansion_point_1 number_of_vectors
expansion_point_2 number_of_vectors
                  ⋮
expansion_point_n number_of_vectors
```

As is introduced in sections 2.1 and 2.5, expansion point[6] $s_0 = 2\pi f_0$ is recommended for the first-order system and the second-order system when using SOAR. For reduction of the proportionally damped second-order system, $\hat{s}_0 = -(2\pi f_0)^2$ is recommended. It should be noted here that the option '`-N`' is ignored when one uses a file to define the expansion points.

`-w`: follow with '`file_name`'. Write the original full system matrices to files (not to write in default).

`-t`: tolerance to deflate a new vector.

In default, the tolerance used in block Arnoldi and SOAR algorithms is `1e-12`. It can be changed by using '`-t number`'.

`-S`: write the complete projection matrix.

Note that the ordering in the projection matrix saved with '`-S`' is the same as in the original full system matrices obtained with option '`-w`'. The projection matrix is saved with the name '`state.proj.1.V`'. When using SOAR for the multiple-input second-order system, the superposition method is used, and several projection matrices for every single input are saved with numbers '`1,2,3,···`'

[6] Although s_0 would be an imaginary number in principle, real expansion points are used throughout this work. It is shown in [103] that the positive real number $|s_0|$ can be used as an expansion point as well.

`-0`: follow with '`init_state.txt`'. Use an inhomogeneous initial state.

Some other `[-options]` might be useful:

`-l`: evaluate the stationary solution.
`-v`: print information from solvers.
`-p`: follow with parameter '`out-of-core`' or '`ldlt`'.

Usually, 'Model Reduction inside ANSYS' automatically detects whether the matrix is symmetric or asymmetric. Afterward, the MUMPS solver offers three factorization strategies depending on the matrix.

- Positive definite matrix: Cholesky decomposition (used as the default for symmetric matrix).
- General symmetric matrix: $L^T DL$ decomposition (use option '`-p ldlt`').
- Asymmetric matrix: LU decomposition.

It is necessary to manually specify '`-p ldlt`' when the matrix is symmetric but indefinite. In addition, the system matrices read from the FULL files are saved in the RAM. Specify '`-p out-of-core`' if the size of the matrix exceeds the capacity of the RAM. However, using '`-p out-of-core`' is rather slow because the full-scale matrices will be saved on the local disc. Therefore, it is recommended to add the extra RAM instead.

In addition, in the practical applications of the MOR algorithms introduced in chapter 2, the main issue is that in each step of an iterative Krylov subspace algorithm, a matrix-vector product needs to be computed. For example, in the Arnoldi algorithm introduced in section 2.3, the next orthogonal vector $\hat{q}_{i+1}$ is calculated as follows:

$$\hat{q}_{i+1} = (A - s_0 E)^{-1} E \cdot q_i \ , \tag{.1}$$

where q_i is the orthonormal vector computed in the previous iteration. It takes a high computational cost of computing $(A - s_0 E)^{-1}$ explicitly due to the fact that the system matrices are usually high-dimensional and sparse. The only feasible solution is to solve the following linear equation in each step:

$$(A - s_0 E) \cdot \hat{q}_{i+1} = E \cdot q_i \ . \tag{.2}$$

In 'Model Reduction inside ANSYS', the open-source MUMPS [104] direct solver is used. The strategy is that first the matrix $(A - s_0 E)$ is factorized, e.g., LU decomposition. Then the cheap cost back substitution is used for computing vector $\hat{q}_{i+1}$. It is important that the factorization process could be done out of the iteration process and needs to be done only once. In addition, the factorization time can be significantly reduced by matrix reordering when METIS [105] is employed.

E APDL Scripts for Blood Perfusion

This section presents the APDL commands used for implementing the blood perfusion heat generation rate in the tissue layer as a 'convection-type' effect, which is introduced in section 5.2.

```
/com,***** MATERIAL PROPERTIES OF BLOOD *****
/prep7                          ! Enters the model creation processor
rho = 1049.75                   ! Density kg/m^3
c = 3617                        ! Specific heat J/kg/K

/com,***** 'CONVECTION-TYPE' EFFECT *****
omega_muscle = 0.000337         ! Perfusion rate 1/s
volume_muscle = 2.24e-004       ! Muscle tissue volume
*get,n_max,node,0,num,max       ! Get the maximum node number
nbc = n_max+1                   ! Set the number of an external node
N,nbc,15e-002,15e-002,15e-002 ! Create an external node
d,nbc,temp,37                   ! Set the temperature at the external node

cmsel,s,muscle                  ! Select the component 'muscle'
nsle,s                          ! Select the nodes in 'muscle'
*get,nn,node,,count             ! Get number of nodes in 'muscle'
*get,etnext,etyp,0,num,max
etnext = etnext + 1
et,etnext,34                    ! Change the element type to LINK34
r,etnext,1
hfeff = rho*c*omega_muscle*volume_muscle/nn ! Film coefficient
mp,hf,etnext,hfeff
type,etnext
real,etnext
mat,etnext

ni = ndnext(0)
*do,i,1,nn                      ! Create LINK34 for each node in 'muscle'
 e,ni,nbc
 ni = ndnext(ni)
*enddo
allsel
fini
/solu
```

The logic of this script is shown in Figure 69. In this case study, the aim is to implement the blood perfusion heat generation rate in the muscle tissue. An external node '`nbc`' is created out of the original model. The element LINK34, which has the ability to convect heat between its nodes, is created between the external node and each node in the muscle tissue layer via a loop. According to Equation (5.10), '`hfeff`' is the average film coefficient per node calculated based on the blood perfusion rate coefficient $\rho_b c_b \omega$. T(I) represents the temperature from the muscle tissue node and T(J) is the constant blood temperature at 37 °C. T(J) > T(I) makes the heat flow negative, which presents the heat flow into the tissue layer. This script can also be applied to other tissue layers and one needs to update the component name, perfusion rate, and tissue volume (noted as red text in the above APDL script).

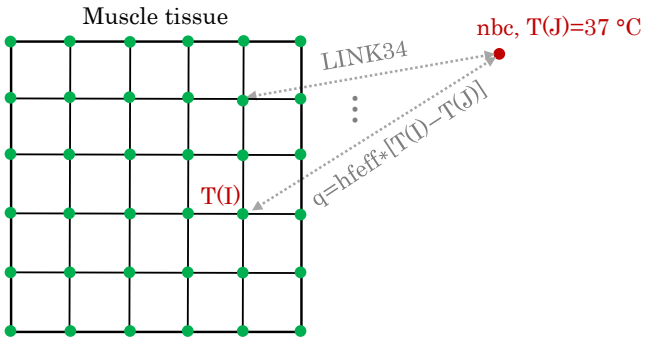


Figure 69: The logic of implementing the 'convection-type' blood perfusion heat generation rate via element LINK34.

F MATLAB Scripts for Stable MOR of Piezoelectric Models

This section mainly presents the MATLAB scripts used for implementing the stable MOR approaches introduced in section 6.2. MATLAB functions 'mmread' and 'mmwrite' are deployed to read and write the Matrix Market matrices. In this work, the function 'mmwrite' is modified to be 'mmwrite2', which requires less computational time to write the sparse and symmetric full-scale matrices.

MOR after Schur: the following script is utilized to generate the Schur-transformed full system presented in Equation (6.5). Afterward, the reduced model is generated via the command line from 'Model Reduction inside ANSYS'.

```
% Names of the system matrix files
basename='full_system';
outname='full_schur';

%%%%% Read matrix E %%%%%
[rawE,rows,cols,entries,rep,field,symm]=mmread([basename,'.E']);
% Dimension of the full system
systemDim=rows;
% Find the zero/nonzero colomns in matrix E
where_are_zeros=uint32(find(sum(abs(rawE))==0));
where_are_not_zeros=uint32(1:systemDim);
where_are_not_zeros(where_are_zeros)=[];
% Dimension of the new system after Schur complement
newsystemDim=systemDim-length(where_are_zeros);
% Remove colomns/rows in matrix E
E1=rawE(where_are_not_zeros,where_are_not_zeros);
% Write the new matrix E
mmwrite2([outname,'.e'],E1,'',field,17,symm);

%%%%% Read matrix M %%%%%
[rawM,rows,cols,entries,rep,field,symm]=mmread([basename,'.M']);
% Remove colomns/rows in matrix M
M1=rawM(where_are_not_zeros,where_are_not_zeros);
% Write the new matrix M
mmwrite2([outname,'.m'],M1,'',field,17,symm);

%%%%% Read matrix K %%%%%
```

```
[rawK,rows,cols,entries,rep,field,symm]=mmread([basename,'.K'
   ]);
% Construct submatrices in matrix K
K11=rawK(where_are_not_zeros,where_are_not_zeros);
K22=rawK(where_are_zeros,where_are_zeros);
K12=rawK(where_are_not_zeros,where_are_zeros);
K21=rawK(where_are_zeros,where_are_not_zeros);
% Construct the new matrix K via Schur complement
TempK=K22\K21;
K1=K11-K12*TempK;
% Write the new matrix K
mmwrite2([outname,'.k'],K1,'',field,17,symm);

%%%%% Read matrix C %%%%%
rawC=mmread([basename,'.C']);
% Construct the new matrix C via Schur complement
C11=rawC(:,where_are_not_zeros);
C22=rawC(:,where_are_zeros);
C1=C11-C22*(TempK);
% Write the new matrix C
mmwrite2([outname,'.c'],C1);
% File 'full_system.C.names' are copied and renamed
copyfile([basename,'.C.names'],[outname,'.C.names'])

%%%%% Read matrix B %%%%%
rawB=mmread([basename,'.B']);
% Construct the new matrix B via Schur complement
B11=rawB(where_are_not_zeros,:);
B22=rawB(where_are_zeros,:);
TempBK=K22\B22;
B1=B11-K12*TempBK;
% Write the new matrix B
mmwrite2([outname,'.b'],B1);
% File 'full_system.B.names' are copied and renamed
copyfile([basename,'.B.names'],[outname,'.B.names'])

%%%%% Get new matrix D %%%%%
D1=C22*TempBK;
% Write the new matrix D
mmwrite2([outname,'.d'],D1);
```

Schur after MOR: the reduced model is first reduced via 'Model Reduction inside ANSYS' and then the script below is utilized to generate the Schur-transformed reduced system presented in Equation (6.13). It should be noted here that, for different models, the value of the threshold in line 26 should be changed according to the eigenvalues obtained from matrix M.

```
1  % Names of the system matrix files
2  basename='mor';
3  outname='schur_mor';
4
5  % Read reduced matrices
6  M=mmread([basename,'.M']);
7  E=mmread([basename,'.E']);
8  K=mmread([basename,'.K']);
9  B=mmread([basename,'.B']);
10 C=mmread([basename,'.C']);
11 % Dimension of the reduced system
12 systemDim=size(M);
13
14 % Eigevalues of the matrix M
15 [evM,eigM]=eig(M);
16 evalM=diag(eigM);
17
18 % Transform to the basis of matrix M eigenvectors
19 eM=eigM;
20 eE=evM'*E*evM;
21 eK=evM'*K*evM;
22 eB=evM'*B;
23 eC=C*evM;
24
25 % Find small eigenvalues of M
26 ze=find(evalM<1e-16);
27 nze=1:systemDim;
28 nze(ze)=[];
29
30 % Implement Schur complement
31 K11=eK(nze,nze);
32 K22=eK(ze,ze);
33 K12=eK(nze,ze);
34 K21=eK(ze,nze);
```

```
35 sM=eM(nze,nze);
36 sE=eE(nze,nze);
37 sK=K11-K12*(K22\K21);
38 sB=eB(nze,:)-K12*(K22\eB(ze,:));
39 sC=eC(:,nze)-eC(:,ze)*(K22\K21);
40 sD=eC(:,ze)*(K22\eB(ze,:));
41
42 %Save new reduced system matrices
43 mmwrite([outname,'.M'],sparse(sM),'','real',17);
44 mmwrite([outname,'.E'],sparse(sE),'','real',17);
45 mmwrite([outname,'.K'],sparse(sK),'','real',17);
46 mmwrite([outname,'.B'],sparse(sB),'','real',17);
47 mmwrite([outname,'.C'],sparse(sC),'','real',17);
48 mmwrite([outname,'.D'],sparse(sD),'','real',17);
49 copyfile([basename,'.B.names'],[outname,'.B.names'])
50 copyfile([basename,'.C.names'],[outname,'.C.names'])
```

Multiphysics structure-preserving MOR: the following scripts are utilized to generate the block structure preserved reduced system as described in section 6.2.3. The work flow contains the following steps:
Step 1: Write the full system matrices from FULL files. The following command line is used to save the full system matrices in files '`full_system.*`'. In this case, '`-N`' is set as 0 and the reduction will not be performed.

```
mor_for_ansys file0.full file1e10.full -N 0 -C output.txt -w full_system
```

Step 2: Reorder the full system matrices. As mentioned in section 5.1, the full system matrices written from the FULL files are not saved in the block structure as presented in Equation (5.2). All the system matrices need to be reordered via the script below. Except for the reordered full system matrices, the ordering indices of the block structure matrices are saved in the file '`reordering_vectors.mat`' and an identity output matrix named as '`full_reordereddummy.c`' is created. The identity output matrix is used for constructing the projection matrix in the next step.

```
1 % Names of the system matrix files
2 basename='full_system';
3 outname='full_reordered';
4
5 %%%%% Read matrix E %%%%%
6 [rawE,rows,cols,entries,rep,field,symm]=mmread([basename,'.E'
    ]);
7 % Dimension of the full system
```

```
systemDim=rows;
% Find the zero/nonzero colomns in matrix E
where_are_zeros=uint32(find(sum(abs(rawE))==0));
where_are_not_zeros=uint32(1:systemDim);
where_are_not_zeros(where_are_zeros)=[];
% Save the ordering indices in a vector
reord_vector=[where_are_not_zeros,where_are_zeros];
% Reorder the matrix E
E=rawE(reord_vector,reord_vector);
% Write the reordered matrix E
mmwrite2([outname,'.e'],E,'',field,17,symm);

%%%%% Read matrix M %%%%%
[rawM,rows,cols,entries,rep,field,symm]=mmread([basename,'.M'
   ]);
% Reorder the matrix M
M=rawM(reord_vector,reord_vector);
% Write the reordered matrix M
mmwrite2([outname,'.m'],M,'',field,17,symm);

%%%%% Read matrix K %%%%%
[rawK,rows,cols,entries,rep,field,symm]=mmread([basename,'.K'
   ]);
% Reorder the matrix K
K=rawK(reord_vector,reord_vector);
% Write the reordered matrix K
mmwrite2([outname,'.k'],K,'',field,17,symm);

%%%%% Read matrix B %%%%%
rawB=mmread([basename,'.B']);
% Reorder the matrix B
B=rawB(reord_vector,:);
% Write the reordered matrix B
mmwrite([outname,'.b'],B); % Save to file
% File 'full_system.B.names' are copied and renamed
copyfile([basename,'.B.names'],[outname,'.B.names'])

%%%%% Read matrix C %%%%%
rawC=mmread([basename,'.C']);
% Reorder the matrix C
```

```
C=rawC(:,reord_vector);
% Write the reordered matrix C
mmwrite([outname,'.c'],C);
% File 'full_system.C.names' are copied and renamed
copyfile([basename,'.C.names'],[outname,'.C.names'])
% Create an identity output matrix
mmwrite([outname,'dummy.c'],speye(systemDim));

% Save the ordering indices in file 'reordering_vectors.mat'
save('reordering_vectors.mat','where_are_not_zeros','
   where_are_zeros')
```

Step 3: Construct the projection matrix. According to Equations (2.9) and (2.22), it is observed clearly that when the output matrix C is an identity matrix, the reduced output matrix $C_r = CV$ is equivalent to the projection matrix V. Therefore, before running the following command line, the output matrix '`full_reordereddummy.c`' is renamed to replace the original output matrix '`full_reordered.c`'. The generated file '`mor.C`' is the projection matrix, which will be used in the next step.

```
mor_for_ansys -M full_reordered -2 -N 30 -o mor > mor.out
```

Step 4: Perform the structure-preserving MOR. The reordered full system matrices '`full_reordered.*`', the projection matrix '`mor.C`', and the ordering indices from the file '`reordering_vectors.mat`' are utilized via the following script. Except for the reduced system matrices presented in Equation (6.15), projection matrices for mechanical and electrical domains are saved in files '`mor_SP_mech.V`' and '`mor_SP_el.V`'.

```
% Names of the reordered system matrix files
basename='full_reordered';
% Name of projection matrix file
pr_vectors='mor.C';
% Names of reduced system matrix files
out_mor='mor_SP';

% Read the projection matrix
V_proj=mmread(pr_vectors);
% Load the reordering indices
load reordering_vectors
% Get the number of mechanical states
n_m=size(where_are_not_zeros,2);
% Get the number of electrical states
n_e=size(where_are_zeros,2);
```

```
% Set the number of mechanical moments
n_r_mech=28
% Set the number of electrical moments
n_r_el=2
% Reconstruct the projection matrix
V_proj_sprim=[V_proj(1:n_m,1:n_r_mech),zeros(n_m,n_r_el);
   zeros(n_e,n_r_mech),V_proj(n_m+1:end,1:n_r_el)];
% Save the projection matrix of the mechanical part
V_proj_mech=V_proj(1:n_m,1:n_r_mech);
mmwrite([out_mor,'_mech.V'],V_proj_mech,'',field,17);
% Save the projection matrix of the electrical part
V_proj_el=V_proj(n_m+1:end,1:n_r_el);
mmwrite([out_mor,'_el.V'],V_proj_el,'',field,17);

%%%%% Read reordered matrix M %%%%%
rawM=mmread([basename,'.M']);
% Reduce matrix M
M=V_proj_sprim'*rawM*V_proj_sprim;
% Get the dimension of the reduced matrix
systemDim=size(M);
% Write the reduced matrix M
field='real';
mmwrite([out_mor ,'.m'],M,'',field,17);

%%%%% Read reordered matrix E %%%%%
rawE=mmread([basename,'.E']);
% Reduce matrix E
E=V_proj_sprim'*rawE*V_proj_sprim;
% Write the reduced matrix E
mmwrite([out_mor ,'.e'],E,'',field,17);

%%%%% Read reordered matrix K %%%%%
rawK=mmread([basename,'.K']);
% Reduce matrix K
K=V_proj_sprim'*rawK*V_proj_sprim;
% Write the reduced matrix K
mmwrite([out_mor ,'.k'],K,'',field,17);

%%%%% Read reordered matrix B %%%%%
rawB = mmread([basename,'.B']);
```

```
% Reduce matrix B
B = V_proj_sprim'*rawB;
% Write the reduced matrix B
mmwrite([out_mor,'.b'],B,'',field,17);
% File 'full_reordered.B.names' are copied and renamed
copyfile([basename,'.B.names'],[out_mor,'.B.names'])

%%%%% Read reordered matrix C %%%%%
rawC=mmread([basename,'.C']);
% Reduce matrix C
C=rawC*V_proj_sprim;
% Write the reduced matrix C
mmwrite([out_mor,'.c'],C,'',field,17);
% File 'full_reordered.C.names' are copied and renamed
copyfile([basename,'.C.names'],[out_mor,'.C.names'])
```

MOR after implicit Schur: the following script is utilized to generate the implicitly Schur-transformed system presented in Equation (6.16). Afterward, the reduced model is generated via the command line from 'Model Reduction inside ANSYS'.

```
% Names of the system matrix files
basename='full_system';
outname='full_implicit';

%%%%% Read matrix E %%%%%
[rawE,rows,cols,entries,rep,field,symm]=mmread([basename,'.E'
   ]);
% Dimension of the full system
systemDim=rows;
% Find the zero/nonzero colomns in matrix E
where_are_zeros=uint32(find(sum(abs(rawE))==0));
where_are_not_zeros=uint32(1:systemDim);
where_are_not_zeros(where_are_zeros)=[];
% Reorder the matrix E
E11=rawE(where_are_not_zeros,where_are_not_zeros);
E22=rawE(where_are_zeros,where_are_zeros);
E12=rawE(where_are_not_zeros,where_are_zeros);
E21=rawE(where_are_zeros,where_are_not_zeros);
E=[E11,E12;E21,E22];
% Write the new matrix E
mmwrite2([outname,'.e'],E,'',field,17,symm);
```

```
21
22 %%%%% Read matrix M %%%%%
23 [rawM,rows,cols,entries,rep,field,symm]=mmread([basename,'.M'
      ]);
24 % Reorder the matrix M
25 M11=rawM(where_are_not_zeros,where_are_not_zeros);
26 M22=rawM(where_are_zeros,where_are_zeros);
27 M12=rawM(where_are_not_zeros,where_are_zeros);
28 M21=rawM(where_are_zeros,where_are_not_zeros);
29 M=[M11,M12;M21,M22];
30 % Write the new matrix M
31 mmwrite2([outname,'.m'],M,'',field,17,symm);
32
33 %%%%% Read matrix K %%%%%
34 [rawK,rows,cols,entries,rep,field,symm]=mmread([basename,'.K'
      ]);
35 % Reorder the matrix K
36 K11=rawK(where_are_not_zeros,where_are_not_zeros);
37 K22=rawK(where_are_zeros,where_are_zeros);
38 K12=rawK(where_are_not_zeros,where_are_zeros);
39 K21=rawK(where_are_zeros,where_are_not_zeros);
40 K=[K11,K12;K21,K22];
41 % Write the new matrix K
42 mmwrite2([outname,'.k'],K,'',field,17,symm);
43
44 %%%%% Read matrix C %%%%%
45 rawC=mmread([basename,'.C']);
46 % Reorder the matrix C
47 C11=rawC(:,where_are_not_zeros);
48 C22=rawC(:,where_are_zeros);
49 C=[C11,C22];
50 % Write the new matrix C
51 mmwrite2([outname,'.c'],C);
52 % File 'full_system.C.names' are copied and renamed
53 copyfile([basename,'.C.names'],[outname,'.C.names'])
54
55 %%%%% Read matrix B %%%%%
56 rawB=mmread([basename,'.B']);
57 % Reorder the matrix B
58 B11=rawB(where_are_not_zeros,:);
```

```
59 B22=rawB(where_are_zeros,:);
60 % Construct the new matrix B via implicit Schur complement
61 TempBK=K22\B22;
62 smallB=B11-K12*TempBK;
63 zeroB=zeros(length(where_are_zeros),size(rawB,2));
64 B=[smallB;zeroB];
65 % Write the new matrix B
66 mmwrite2([outname,'.b'],B);
67 % File 'full_system.B.names' are copied and renamed
68 copyfile([basename,'.B.names'],[outname,'.B.names'])
69
70 %%%%% Get new matrix D %%%%%
71 D = C22*TempBK;
72 % Write the new matrix D
73 mmwrite2([outname,'.d'],D);
```

G Compact Modeling Technique for Electromagnetic Energy Harvesters

This section presents the equivalent circuit extraction (equivalent circuit extraction (ECE)) technique [106] for generating the compact model of an electromagnetic energy harvester model in ANSYS Maxwell 3D. The compact model constructed via ECE can only be exported based on a parametric solution when the solution type is 'Magnetostatic' or 'Electrostatic', whereas in section 5.3 a 'Transient' scheme is implemented. To set up a 'Magnetostatic' solution type, the positions of the magnets are parametrized to represent the oscillation of the magnets. In addition, the induced voltage cannot be obtained directly from the simulation with solution type 'Magnetostatic'. By using Equation (5.16), the induced voltage output is calculated via the magnetic flux rate change obtained through the coil. The workflow of exporting the compact model from the electromagnetic energy harvester model described in section 5.3 is as follows:

Step 1: Rebuild the structure of the magnets and coil in ANSYS Maxwell 3D with solution type 'Magnetostatic'.

Step 2: Instead of implementing the motion setups in the motion bands, the function 'Move' is implemented to move the magnets as shown in Figure 70. A variable '`move`' is defined and used in the 'Move Vector'.

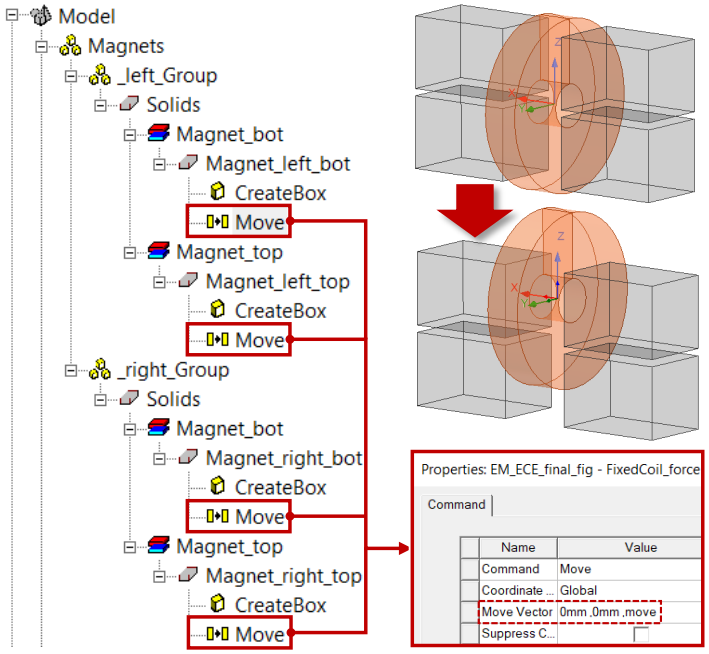


Figure 70: Magnet position parameterization setup. The position of the magnets is parameterized via the 'Move' function, where a parameter '`move`' is defined to change the position of the magnets along the coil. This figure is adapted from [73].

Step 3: In the 'Excitation', select the cross-section of the coil and assign a 'Current' valued as a variable '`amp_turns`' (see Figure 71). The coil excitation type is selected as

'Stranded'. Although there is no current input in this case, this variable is still required for generating the compact model.

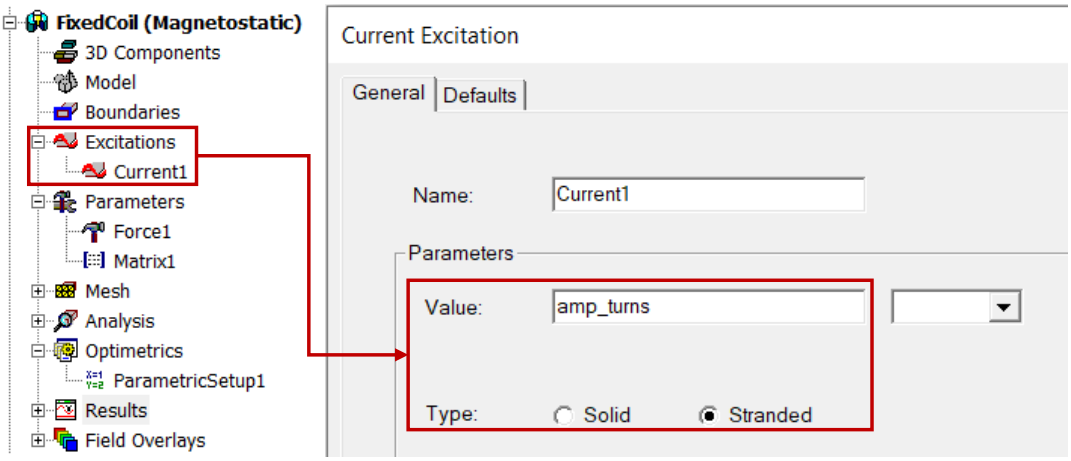


Figure 71: The excitation setup in the coil. A parameter '`amp_turns`' is defined as the current input in the coil and the parameter type is selected as 'Stranded'. The number of turns in the coil will be set in the next step. This figure is adapted from [73].

Step 4: Two parameters, 'Force' and 'Matrix', are defined in the model setup (see Figure 72). 'Force' is assigned to the geometry of the coil with type 'Lorentz'. The parameter 'Matrix' includes the current input in the coil and the number of turns is 600.

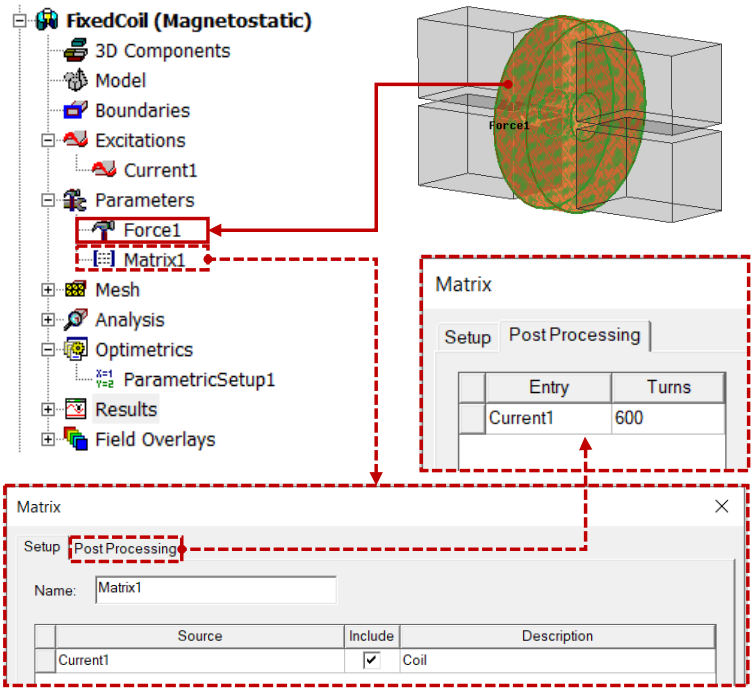


Figure 72: Essential setup of the parameters for generating the compact model. 'Force' is assigned to the geometry of the coil with type 'Lorentz'. The number of turns in the coil is defined as 600 in 'Matrix'. This figure is adapted from [73].

Step 5: Insert a 'ParametricSetup' under the tag of 'Optimetrics' (see Figure 73). Parameterize the variable '`move`' between -0.6 and 0.6 mm with step 0.05 mm and the variable '`amp_turns`' is parameterized between -0.6 and 0.6 mA with step 0.1 mA.

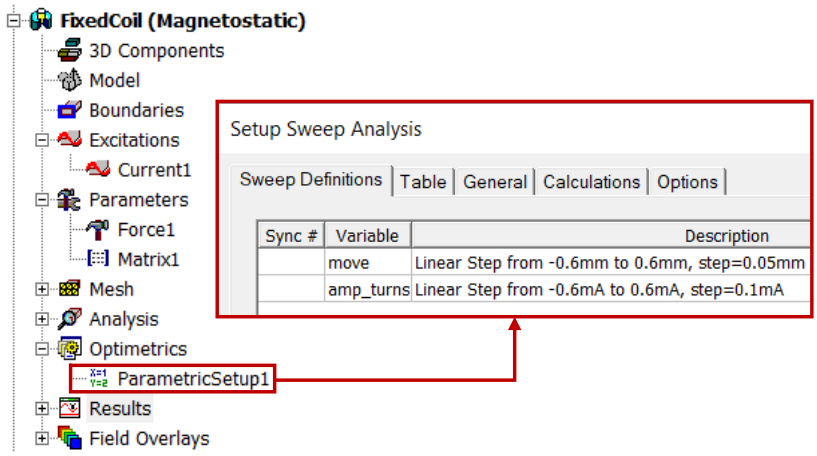


Figure 73: Parametric setup of the variables. Parameterizing the variable '`move`' and the variable '`amp_turns`' in specific ranges. This figure is adapted from [73].

Step 6: Analyze the 'ParametricSetup' and export the equivalent circuit from the parametric solution. As shown in Figure 74, the model type is selected as 'Linear Motion'. The 'ParametricSetup', 'Matrix', and 'Force' defined in the previous steps are selected. Force is set up in the component z. Current variables represent 'Amperes'. Finally, the internal resistance of the coil is defined as 100 Ω when exporting the compact model, which will be saved in the *sml*-format.

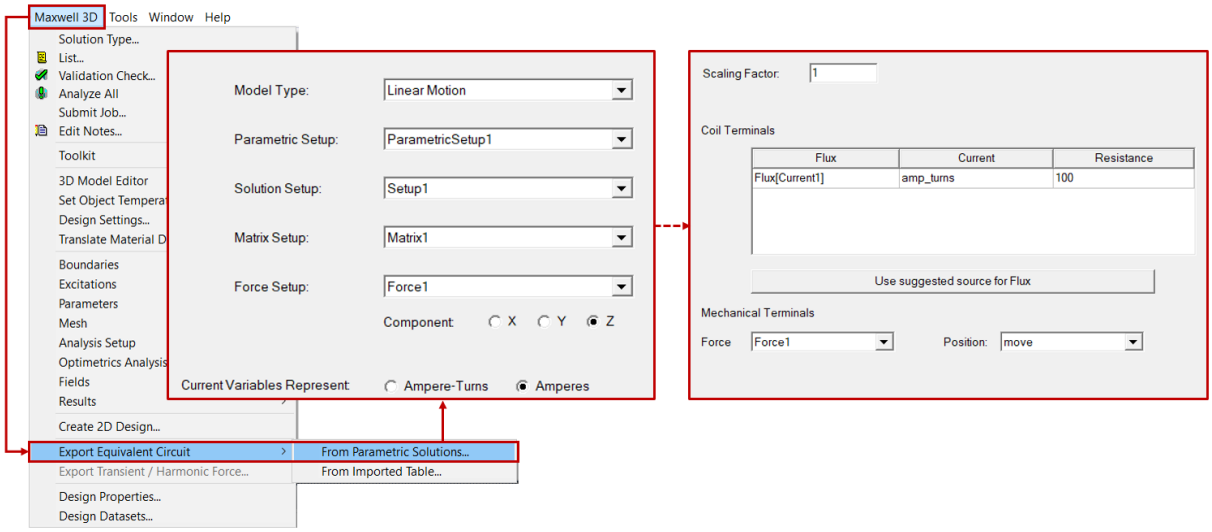


Figure 74: General setup for exporting the compact model. Model type: 'Linear Motion'. Component: 'Z'. Current variables represent: 'Amperes'. Resistance: 100 Ω.

References

[1] C. Bai, P. Dallasega, G. Orzes, and J. Sarkis, "Industry 4.0 technologies assessment: A sustainability perspective," *International Journal of Production Economics*, vol. 229, p. 107776, 2020. https://doi.org/10.1016/j.ijpe.2020.107776.

[2] L. S. Dalenogare, G. B. Benitez, N. F. Ayala, and A. G. Frank, "The expected contribution of industry 4.0 technologies for industrial performance," *International Journal of Production Economics*, vol. 204, pp. 383–394, 2018. https://doi.org/10.1016/j.ijpe.2018.08.019.

[3] J. M. Müller, D. Kiel, and K. Voigt, "What drives the implementation of industry 4.0? The role of opportunities and challenges in the context of sustainability," *Sustainability*, vol. 10, no. 1, p. 247, 2018. https://doi.org/10.3390/su10010247.

[4] N. S. Hudak and G. G. Amatucci, "Small-scale energy harvesting through thermoelectric, vibration, and radiofrequency power conversion," *Journal of Applied Physics*, vol. 103, no. 10, p. 101301, 2008. https://doi.org/10.1063/1.2918987.

[5] S. Priya and D. J. Inman, eds., *Energy harvesting technologies*. New York, USA: Springer, 2009. https://doi.org/10.1007/978-0-387-76464-1.

[6] S. Bundesamt, "Coordinated population projection for german," 2023. `https://service.destatis.de/bevoelkerungspyramide/#!y=2023&a=20,65&g` Last accessed 24 August 2023.

[7] V. K. Khanna, *Implantable medical electronics*. Cham, CHE: Springer International Publishing, 2016. https://doi.org/10.1007/978-3-319-25448-7.

[8] F. K. Shaikh and S. Zeadally, "Energy harvesting in wireless sensor networks: A comprehensive review," *Renewable and Sustainable Energy Reviews*, vol. 55, pp. 1041–1054, 2016. https://doi.org/10.1016/j.rser.2015.11.010.

[9] A. J. Williams, M. Torquato, I. M. Cameron, A. A. Fahmy, and J. Sienz, "Survey of energy harvesting technologies for wireless sensor networks," *IEEE Access*, vol. 9, pp. 77493–77510, 2021. DOI: 10.1109/ACCESS.2021.3083697.

[10] A. Toprak and O. Tigli, "Piezoelectric energy harvesting: state-of-the-art and challenges," *Applied Physics Reviews*, vol. 1, no. 3, p. 031104, 2014. https://doi.org/10.1063/1.4896166.

[11] F. U. Khan and M. U. Qadir, "State-of-the-art in vibration-based electrostatic energy harvesting," *Journal of Micromechanics and Microengineering*, vol. 26, no. 10, p. 103001, 2016. DOI: 10.1088/0960-1317/26/10/103001.

[12] Y. Tan, Y. Dong, and X. Wang, "Review of MEMS electromagnetic vibration energy harvester," *Journal of Microelectromechanical Systems*, vol. 26, no. 1, pp. 1–16, 2017. DOI: 10.1109/JMEMS.2016.2611677.

[13] N. Jaziri, A. Boughamoura, J. Müller, B. Mezghani, F. Tounsi, and M. Ismail, "A comprehensive review of thermoelectric generators: technologies and common applications," *Energy Reports*, vol. 6, pp. 264–287, 2020. https://doi.org/10.1016/j.egyr.2019.12.011.

[14] Z. Wang, S. Matova, R. Elfrink, M. Jambunathan, C. de Nooijer, R. van Schaijk, and R. J. M. Vullers, "A piezoelectric vibration harvester based on clamped-guided beams," in *IEEE 25th International Conference on Micro Electro Mechanical Systems (MEMS)*, (Paris, France), pp. 1201–1204, 2012. DOI: 10.1109/MEMSYS.2012.6170379.

[15] O. S. Jadhav, C. Yuan, D. Hohlfeld, and T. Bechtold, "Design of a thermoelectric generator for electrical active implants," in *MikroSystemTechnik (MST) Congress*, (Berlin, Germany), pp. 402–405, 2017. https://ieeexplore.ieee.org/document/8278689.

[16] S. P. Beeby, R. N. Torah, M. J. Tudor, P. Glynne-Jones, T. O'Donnell, C. R. Saha, and S. Roy, "A micro electromagnetic generator for vibration energy harvesting," *Journal of Micromechanics and Microengineering*, vol. 17, no. 7, pp. 1257–1265, 2007. https://dx.doi.org/10.1088/0960-1317/17/7/007.

[17] ANSYS®, "Ansys Mechanical." Release 2023 R1, Ansys, Inc.

[18] ANSYS®, "Ansys Mechanical APDL." Release 2023 R1, Ansys, Inc.

[19] S. Gugercin and A. C. Antoulas, "A comparative study of 7 algorithms for model reduction," in *Proceedings of the 39th IEEE Conference on Decision and Control*, vol. 3, (Sydney, Australia), pp. 2367–2372, 2000. DOI: 10.1109/CDC.2000.914153.

[20] A. C. Antoulas, D. C. Sorensen, and S. Gugercin, "A survey of model reduction methods for large-scale systems," *Contemporary Mathematics*, vol. 280, pp. 193–219, 2001. https://doi.org/10.1090/conm/280.

[21] E. B. Rudnyi and J. G. Korvink, "Review: Automatic model reduction for transient simulation of MEMS-based devices," *Sensors Update*, vol. 11, no. 1, pp. 3–33, 2002. https://doi.org/10.1002/seup.200211105.

[22] A. C. Antoulas, *Approximation of large-scale dynamical systems*. Advances in design and control, Philadelphia, USA: Society for Industrial and Applied Mathematics, 2005. https://doi.org/10.1137/1.9780898718713.

[23] P. Benner, S. Gugercin, and K. Willcox, "A survey of projection-based model reduction methods for parametric dynamical systems," *SIAM Review*, vol. 57, no. 4, pp. 483–531, 2015. https://doi.org/10.1137/130932715.

[24] U. Baur, P. Benner, B. Haasdonk, C. Himpe, I. Martini, and M. Ohlberger, "Comparison of methods for parametric model order reduction of time-dependent problems," in *Model reduction and approximation: theory and algorithm* (P. Benner, M. Ohlberger, A. Cohen, and K. Willcox, eds.), pp. 377–407, Philadelphia, USA: Society for Industrial and Applied Mathematics, 2017. https://doi.org/10.1137/1.9781611974829.

[25] E. B. Rudnyi and J. G. Korvink, "Model order reduction for large scale engineering models developed in ansys," in *Applied Parallel Computing. State of the Art in Scientific Computing.* (J. Dongarra, K. Madsen, and J. Waśniewski, eds.), pp. 349–356, Springer, Berlin, Heidelberg, 2006. https://doi.org/10.1007/11558958_41.

[26] E. B. Rudnyi, "Mor for ANSYS," in *System-level modelling of MEMS* (T. Bechtold, G. Schrag, and L. Feng, eds.), Advanced Micro and Nanosystems, pp. 425–438, Weinheim, Germany: Wiley VCH Verlag GmbH & Co. KGaA, 2013. https://doi.org/10.1002/9783527647132.ch18.

[27] H. A. van der Vorst, "Krylov subspace iteration," *Computing in Science & Engineering*, vol. 2, no. 1, pp. 32–37, 2000. DOI: 10.1109/5992.814655.

[28] W. E. Arnoldi, "The principle of minimized iteration in the solution of the matrix eigenvalue problem," *Quarterly of Applied Mathematics*, vol. 9, pp. 17–29, 1951. https://doi.org/10.1090/qam/42792.

[29] R. W. Freund, "Krylov-subspace methods for reduced-order modeling in circuit simulation," *Journal of Computational and Applied Mathematics*, vol. 123, no. 1, pp. 395–421, 2000. https://doi.org/10.1016/S0377-0427(00)00396-4.

[30] P. Benner, L. Feng, and E. B. Rudnyi, "Using the superposition property for model reduction of linear systems with a large number of inputs," in *Proceedings of the 18th International Symposium on Mathematical Theory of Networks & Systems*, 2008. https://scholar.lib.vt.edu/MTNS/Papers/028.pdf.

[31] C. Brezinski, *Padé-Type Approximation and General Orthogonal Polynomials*. Basel, CHE: Birkhäuser Basel, 1980. https://doi.org/10.1007/978-3-0348-6558-6.

[32] Z. Bai and Y. Su, "Dimension reduction of large-scale second-order dynamical system via a second-order Arnoldi method," *SIAM Journal on Scientific Computing*, vol. 26, no. 5, pp. 1692–1709, 2005. https://doi.org/10.1137/040605552.

[33] Z. Bai and Y. Su, "SOAR: A second-order Arnoldi method for the solution of the quadratic eigenvalue problem," *SIAM Journal on Matrix Analysis and Applications*, vol. 26, no. 3, pp. 640–659, 2005. https://doi.org/10.1137/S0895479803438523.

[34] S. Salimbahrami, *Structure preserving order reduction of large scale second order models*. PhD thesis, Technical University of Munich, 2005. https://mediatum.ub.tum.de/doc/601950/00000941.pdf.

[35] R. Eid, B. Salimbahrami, B. Lohmann, E. B. Rudnyi, and J. G. Korvink, "Parametric order reduction of proportionally damped second order systems," *Sensors and Materials*, vol. 19, no. 3, pp. 149–164, 2007. https://sensors.myu-group.co.jp/sm_pdf/SM672.pdf.

[36] E. B. Rudnyi, J. Lienemann, A. Greiner, and J. G. Korvink, "mor4ansys: Generating compact models directly from ANSYS models," in *Technical Proceedings of the 2004 Nanotechnology Conference and Trade Show*, vol. 2, (Boston, Massachusetts, USA), pp. 279–282, 2004. https://briefs.techconnect.org/papers/mor4ansys-generating-compact-models-directly-from-ansys-models/.

[37] J. S. Han, E. B. Rudnyi, and J. G. Korvink, "Efficient optimization of transient dynamic problems in MEMS devices using model order reduction," *Journal of Micromechanics and Microengineering*, vol. 15, no. 4, pp. 822–832, 2005. https://dx.doi.org/10.1088/0960-1317/15/4/021.

[38] M. Schwarzer, E. Barti, and T. Bein, "Model order reduction of dynamical structural simulation models of electric motors using Krylov subspaces," in *Proceedings of International Conference on Noise and Vibration Engineering*, (Leuven, Belgium), pp. 1473–1486, 2014. https://past.isma-isaac.be/downloads/isma2014/papers/isma2014_0566.pdf.

[39] C. Yuan, *Model order reduction and system simulation of a machine tool for real-time compensation of thermally induced deformations*. Master thesis, University of Rostock, 2016.

[40] J. W. Tedesco, W. G. McDougal, and C. A. Ross, *Structural Dynamics: Theory and Applications*. Boston, USA: Addison Wesley Longman, 1999. .

[41] J. H. Ginsberg, *Mechanical and Structural Vibrations: Theory and Applications*. New York, USA: Wiley, 2001. .

[42] J. Mehner, "Modal-superposition-based nonlinear model order reduction for MEMS gyroscopes," in *System-level modelling of MEMS* (T. Bechtold, G. Schrag, and L. Feng, eds.), Advanced Micro and Nanosystems, pp. 291–309, Weinheim, Germany: Wiley VCH Verlag GmbH & Co. KGaA, 2013. https://doi.org/10.1002/9783527647132.ch12.

[43] P. K. Gunupudi and M. Nakhla, "Multi-dimensional model reduction of VLSI interconnects," in *Proceedings of IEEE 2000 Custom Integrated Circuits Conference*, (Orlando, USA), pp. 499–502, 2000. DOI: 10.1109/CICC.2000.852717.

[44] T. Bechtold, D. Hohlfeld, E. B. Rudnyi, and M. Günther, "Efficient extraction of thin-film thermal parameters from numerical models via parametric model order reduction," *Journal of Micromechanics and Microengineering*, vol. 20, no. 4, p. 045030, 2010. https://dx.doi.org/10.1088/0960-1317/20/4/045030.

[45] H. Panzer, J. Mohring, R. Eid, and B. Lohmann, "Parametric model order reduction by matrix interpolation," *Automatisierungstechnik*, vol. 58, no. 8, pp. 475–484, 2010. https://doi.org/10.1524/auto.2010.0863.

[46] F. B. Hildebrand, *Introduction to Numerical Analysis*. New York, USA: Dover Publications, Inc., 2 ed., 1974. https://doi.org/10.1007/978-0-387-21738-3.

[47] A. Roy and M. Nabi, "Efficient simulation of electro-thermal micro-gripper using pMOR," in *Indian Control Conference (ICC)*, 2018.
DOI: 10.1109/INDIANCC.2018.8307978.

[48] A. Roy, M. Nabi, and N. Rahman, "Finite element compatible matrix interpolation for parametric model order reduction of electrothermal microgripper," *Journal of Computational Design and Engineering*, vol. 8, no. 6, pp. 1622–1635, 2021.
https://doi.org/10.1093/jcde/qwab066.

[49] C. Moosmann, *ParaMOR - Model Order Reduction for parameterized MEMS applications*. PhD thesis, University of Freiburg, 2007.
https://freidok.uni-freiburg.de/data/3971.

[50] U. Baur, P. Benner, A. Greiner, J. G. Korvink, J. Lienemann, and C. Moosmann, "Parameter preserving model order reduction for MEMS applications," *Mathematical and Computer Modelling of Dynamical Systems: Methods, Tools and Applications in Engineering and Related Sciences*, vol. 17, no. 4, pp. 297–317, 2011.
https://doi.org/10.1080/13873954.2011.547658.

[51] ANSYS®, "Electronics Desktop, Twin Builder." Release 2023 R1, Ansys, Inc.

[52] A. Castagnotto, M. C. Varona, L. Jeschek, and B. Lohmann, "sss & sssMOR: Analysis and reduction of large-scale dynamic systems in MATLAB," *Automatisierungstechnik*, vol. 65, pp. 134–150, 2017. https://doi.org/10.1515/auto-2016-0137.

[53] R. Milk, S. Rave, and F. Schindler, "pyMOR - Generic algorithms and interfaces for model order reduction," *SIAM Journal on Scientific Computing*, vol. 38, pp. S194–S216, 2016. https://doi.org/10.1137/15M1026614.

[54] T. Bechtold, E. B. Rudnyi, and J. G. Korvink, "Error indicators for fully automatic extraction of heat-transfer macromodels for MEMS," *Journal of Micromechanics and Microengineering*, vol. 15, no. 3, pp. 430–440, 2004. https://dx.doi.org/10.1088/0960-1317/15/3/002.

[55] ANSYS®, "Electronics Desktop, Maxwell 3D." Release 2023 R1, Ansys, Inc.

[56] M. Kudryavtsev, E. B. Rudnyi, J. G. Korvink, D. Hohlfeld, and T. Bechtold, "Computationally efficient and stable order reduction methods for a large-scale model of MEMS piezoelectric energy harvester," *Microelectronics Reliability*, vol. 55, no. 5, pp. 747–757, 2015. https://doi.org/10.1016/j.microrel.2015.02.003.

[57] S. Bouhedma, Y. Zheng, and D. Hohlfeld, "Multiphysics modeling and simulation of a dual frequency energy harvester," in *Proceedings of the 32nd European Conference on Modeling and Simulation (ECMS)*, (Wilhelmshaven, Germany), pp. 386–390, 2018. https://doi.org/10.7148/2018-0386.

[58] S. Hu, C. Yuan, A. Castagnotto, B. Lohmann, S. Bouhedma, D. Hohlfeld, and T. Bechtold, "Stable reduced order modeling of piezoelectric energy harvesting modules using implicit Schur complement," *Microelectronics Reliability*, vol. 85, pp. 148–155, 2018. https://doi.org/10.1016/j.microrel.2018.03.026.

[59] C. Yuan, S. Hu, and T. Bechtold, "Stable compact modeling of piezoelectric energy harvester devices," *COMPEL-The International Journal for Computation and Mathematics in Electrical and Electronic Engineering*, vol. 39, no. 2, pp. 467–480, 2020. https://doi.org/10.1108/COMPEL-07-2019-0305.

[60] ANSYS®, "Piezo and MEMS v19.3." `https://catalog.ansys.com/product/5b3bc6857a2f9a5c90d32e9e/piezo-and-mems` Last accessed 09 July 2023.

[61] "IEEE standard on piezoelectricity," *ANSI/IEEE Std 176-1987*, 1988. DOI: 10.1109/IEEESTD.1988.79638.

[62] H. Allik and T. J. R. Hughes, "Finite element method for piezoelectric vibration," *International Journal for Numerical Methods in Engineering*, vol. 2, no. 2, pp. 151–157, 1970. https://doi.org/10.1002/nme.1620020202.

[63] U. Verma, J. Bernhardt, and D. Hohlfeld, "Modeling and simulation of bioheat powered subcutaneous thermoelectric generator," in *Proceedings of the 32nd European Conference on Modelling and Simulation (ECMS)*, (Wilhelmshaven, Germany), pp. 381–385, 2018. https://doi.org/10.7148/2018-0381.

[64] Y. Yang, X. Wei, and J. Liu, "Suitability of a thermoelectric power generator for implantable medical electronic devices," *Journal of Physics D: Applied Sciences*,

vol. 40, no. 18, pp. 5790–5800, 2007.
https://dx.doi.org/10.1088/0022-3727/40/18/042.

[65] H. H. Pennes, "Analysis of tissue and arterial blood temperatures in the resting human forearm," *in Journal of Applied Physiology*, vol. 1, no. 2, pp. 93–122, 1948. https://doi.org/10.1152/jappl.1998.85.1.5.

[66] Y. Yang, G. D. Xu, and J. Liu, "Prototype of an implantable thermoelectric generator for permanent power supply to body inside a medical device," *Journal of Medical Devices*, vol. 8, no. 1, p. 014507, 2014. https://doi.org/10.1115/1.4025619.

[67] C. Yuan, S. Kreß, G. Sadashivaiah, E. B. Rudnyi, D. Hohlfeld, and T. Bechtold, "Towards efficient design optimization of a miniaturized thermoelectric generator for electrically active implants via model order reduction and submodeling technique," *International Journal for Numerical Methods in Biomedical Engineering*, vol. 36, no. 4, 2020. https://doi.org/10.1002/cnm.3311.

[68] K. C. Parsons, *Human thermal environments (2nd ed.)*. London, U.K.: Taylor & Francis Inc, 2003. https://doi.org/10.1201/b16750.

[69] W. K. Lewis, "The evaporation of a liquid into a gas," *Transactions of the American Society of Mechanical Engineers*, vol. 44, pp. 325–332, 1922.
https://doi.org/10.1115/1.4058175.

[70] S. N. Makarov, G. M. Noetscher, J. Yanamadala, M. W. Piazza, S. Louie, A. Prokop, A. Nazarian, and A. Nummenmaa, "Virtual human models for electromagnetic studies and their applications," *IEEE Reviews in Biomedical Engineering*, vol. 10, pp. 95–121, 2017. VHP-Female V.2.2. DOI: 10.1109/RBME.2017.2722420.

[71] C. Yuan, G. Sadashivaiah, T. Bechtold, and E. B. Rudnyi, "Efficient design optimization of a thermoelectric generator by a combination of model order reduction and thermal submodeling techniques," in *Proceedings of the 33rd European Conference on Modeling and Simulation (ECMS)*, (Caserta, Italy), pp. 290–295, 2019. https://doi.org/10.7148/2019-0290.

[72] E. Marín, "Linear relationships in heat transfer," *Latin-American Journal of Physics Education*, vol. 3, no. 2, pp. 243–245, 2009. http://lajpe.org/may09/09_Marin.pdf.

[73] C. Yuan, D. Hohlfeld, and T. Bechtold, "Towards system-level simulation of an electromagnetic energy harvester model via equivalent circuit extraction from ANSYS Maxwell 3D," in *24th International Conference on Thermal, Mechanical and Multi-Physics Simulation and Experiments in Microelectronics and Microsystems (EuroSimE)*, (Graz, Austria), 2023. DOI: 10.1109/EuroSimE56861.2023.10100816.

[74] P. M. Whelan and M. J. Hodgeson, *Essential Principles of Physics (2nd ed.)*. John Murray, 1978. .

[75] ANSYS®, "Mechanical APDL, Release 2023 R1, Theory Reference, Chapter 5: Electromagnetics.." ANSYS, Inc.

[76] P. Benner and T. Stykel, "Model order reduction for differential-algebraic equations: a survey," in *Surveys in Differential-Algebraic Equations IV* (A. Ilchmann and T. Reis, eds.), pp. 107–160, Springer, Cham, 2017. https://doi.org/10.1007/978-3-319-46618-7_3.

[77] C. Chen, C. Kuo, and Y. Yang, "Generating passive compact models for piezoelectric devices," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 30, no. 3, pp. 464–467, 2011. DOI: 10.1109/TCAD.2010.2090750.

[78] T. Aftab, D. Hohlfeld, E. Rudnyi, T. Bechtold, and J. Korvink, "New modelling approach for micro energy harvesting systems based on model order reduction enabling truly system-level simulation," in *Proceedings of the 23rd Micromechanics and Microsystems Europe Workshop*, (Ilmenau, Germany), pp. 9–12, 2012.

[79] M. Kurch, *Entwicklung einer simulationsumgebung für die auslegung piezoelektrischer energy harvester*. PhD thesis, Technische Universität Darmstadt, 2013. http://tuprints.ulb.tu-darmstadt.de/4232.

[80] A. Schütz and T. Bechtold, "Performance comparison for stable compact modelling of piezoelectric microactuator," in *23rd International Conference on Thermal, Mechanical and Multi-Physics Simulation and Experiments in Microelectronics and Microsystems (EuroSimE)*, (St Julian, Malta), 2022. DOI: 10.1109/EuroSimE54907.2022.9758853.

[81] A. Castagnotto, H. K. Panzer, K. D. Reinsch, and B. Lohmann, "Stability-preserving adaptive model order reduction of DAEs by Krylov-subspace methods," *arXiv:1508.07227*, 2015. https://doi.org/10.48550/arXiv.1508.07227.

[82] S. Hu, C. Yuan, and T. Bechtold, "Quasi-schur transformation for the stable compact modeling of piezoelectric energy harvester devices," in *Scientific Computing in Electrical Engineering* (G. Nicosia and V. Romano, eds.), pp. 267–276, Springer, Cham, 2020. https://doi.org/10.1007/978-3-030-44101-2_25.

[83] R. W. Freund, "SPRIM: structure-preserving reduced order interconnect macromodeling," in *Proceedings of the International Conference on Computer-Aided Design*, (San Jose, CA, USA), pp. 80–87, 2004. DOI: 10.1109/ICCAD.2004.1382547.

[84] J. Rommes and N. Martins, "Exploiting structure in large-scale electrical circuit and power system problems," *Linear Algebra and its Applications*, vol. 431, no. 3, pp. 318–333, 2009. https://doi.org/10.1016/j.laa.2008.12.027.

[85] M. Fischer and P. Eberhard, "Application of parametric model reduction with matrix interpolation for simulation of moving loads in elastic multibody systems," *Advances in Computational Mathematics*, vol. 41, no. 5, pp. 1049–1072, 2015. https://doi.org/10.1007/s10444-014-9379-7.

[86] C. Yuan and T. Bechtold, "Geometrically parametrized reduced order model of a miniaturized thermoelectric generator for electrically active implants," in *22nd International Conference on Thermal, Mechanical and Multi-Physics Simulation and Experiments in Microelectronics and Microsystems (EuroSimE)*, 2021. DOI: 10.1109/EuroSimE52062.2021.9410842.

[87] C. Yuan, S. Hu, and T. Bechtold, "Stable parametric model order reduction of piezoelectric energy harvester by matrix interpolation," *IFAC-PapersOnLine*, vol. 55, no. 20, pp. 217–222, 2022. https://doi.org/10.1016/j.ifacol.2022.09.098.

[88] E. B. Rudnyi, C. Moosmann, A. Greiner, T. Bechtold, and J. G. Korvink, "Parameter preserving model reduction for MEMS system-level simulation and design," in *Proceedings of 5th MathMOD*, (Vienna, Austria), 2006. http://modelreduction.rudnyi.ru/doc/papers/rudnyi06mathmod.pdf.

[89] C. Yuan, O. S. Jadhav, E. B. Rudnyi, D. Hohlfeld, and T. Bechtold, "Parametric model order reduction of a thermoelectric generator for electrically active implants," in *19th International Conference on Thermal, Mechanical and Multi-Physics Simulation and Experiments in Microelectronics and Microsystems (EuroSimE)*, (Toulouse, France), 2018. DOI: 10.1109/EuroSimE.2018.8369946.

[90] C. Yuan, D. Hohlfeld, and T. Bechtold, "Design optimization of a miniaturized thermoelectric generator," in *21st International Conference on Thermal, Mechanical and Multi-Physics Simulation and Experiments in Microelectronics and Microsystems (EuroSimE)*, 2020. DOI: 10.1109/EuroSimE48426.2020.9152746.

[91] C. Yuan, D. Hohlfeld, and T. Bechtold, "Design optimization of a miniaturized thermoelectric generator via parametric model order reduction," *Microelectronics Reliability*, vol. 119, p. 114075, 2021. https://doi.org/10.1016/j.microrel.2021.114075.

[92] P. A. Hasgall, F. D. Gennaro, C. Baumgartner, E. Neufeld, B. Lloyd, M. C. Gosselin, D. Payne, A. Klingenböck, and N. Kuster, *IT'IS Database for thermal and electromagnetic parameters of biological tissues*. IT'IS Foundation, Zurich, Switzerland, version 4.0 ed., 2018. https://itis.swiss/virtual-population/tissue-properties/overview/.

[93] D. Fiala, K. J. Lomas, and M. Stohrer, "A computer model of human thermoregulation for a wide range of environmental conditions: the passive system," *Journal of Applied Physiology*, vol. 87, no. 5, pp. 1957–1972, 1999. https://doi.org/10.1152/jappl.1999.87.5.1957.

[94] J. Laszczyk, A. Maczko, W. Walas, and A. J. Nowak, "Determining of the neonatal thermal model parameters using inverse thermal analysis," in *6th European Conference on Computational Fluid Dynamics*, (Barcelona, Spain), pp. 5920–5928, 2014. https://congress.cimne.com/iacm-eccomas2014/frontal/Ebook.asp.

[95] L. Zhu, *Heat transfer applications in biological systems*, ch. 2, pp. 33–67. Biomedical Engineering & Design Handbook, Volume 1: Bioengineering Fundamentals, New York, USA: The McGraw-Hill Companies, Inc., 2009. https://bioheat.umbc.edu/wp-content/uploads/sites/363/2015/07/Chapter2_2007.pdf.

[96] S. Murakami, S. Kato, and J. Zeng, "Combined simulation of airflow, radiation and moisture transport for heat release from a human body," *Building and Environment*, vol. 35, pp. 489–500, 2000. https://doi.org/10.1016/S0360-1323(99)00033-5.

[97] ANSYS®, "Mechanical APDL, Release 2023 R1, Advanced Analysis Guide, 7-Submodeling." ANSYS, Inc.

[98] Y. Rao, C. Yuan, G. Sadashivaiah, D. Hohlfeld, and T. Bechtold, "Efficient design optimization of a miniaturized thermoelectric generator for electrically active implants based on parametric model order reduction," *International Journal for Numerical Methods in Biomedical Engineering*, vol. 37, no. 10, 2021. https://doi.org/10.1002/cnm.3517.

[99] C. Yuan, A. Schütz, D. Hohlfeld, and T. Bechtold, "Towards system-level simulation of a miniature electromagnetic energy harvester model," *Electronics*, vol. 12, no. 15, 2023. https://doi.org/10.3390/electronics12153252.

[100] C. B. Umunnakwe, I. Zawra, E. B. Rudnyi, M. Niessner, and T. Bechtold, "Thermo-mechanical super-element of a packaged-chip model for re-integrating reduced state-space models into finite element environment," in *24th International Conference on Thermal, Mechanical and Multi-Physics Simulation and Experiments in Microelectronics and Microsystems (EuroSimE)*, (Graz, Austria), 2023. DOI: 10.1109/EuroSimE56861.2023.10100846.

[101] J. Lee, B. Bagheri, and H.-A. Kao, "A cyber-physical systems architecture for industry 4.0-based manufacturing systems," *Manufacturing Letters*, vol. 3, pp. 18–23, 2015. https://doi.org/10.1016/j.mfglet.2014.12.001.

[102] Z. Wang, R. Gupta, K. Han, H. Wang, A. Ganlath, N. Ammar, and P. Tiwari, "Mobility digital twin: Concept, architecture, case study, and future challenges," *IEEE Internet of Things Journal*, vol. 9, no. 18, pp. 17452–17467, 2022. DOI: 10.1109/JIOT.2022.3156028.

[103] E. J. Grimme, *Krylov projection methods for model reduction*. PhD thesis, University of Illinois at Urbana-Champain, 1997. https://hdl.handle.net/2142/81180.

[104] P. R. Amestoy, I. S. Duff, and J. Y. L'Excellent, "Multifrontal parallel distributed symmetric and unsymmetric solvers," *Computer Methods in Applied Mechanics and Engineering*, vol. 184, no. 2, pp. 501–520, 2000. https://doi.org/10.1016/S0045-7825(99)00242-X.

[105] G. Karypis and V. Kumar, "A fast and high-quality multilevel scheme for partitioning irregular graphs," *SIAM Journal on Scientific Computing*, vol. 20, no. 1, pp. 359–392, 1998. https://doi.org/10.1137/S1064827595287997.

[106] ANSYS®, "Electronics Desktop, Maxwell 3D, Release 2022 R1, Maxwell Help, 19-Exporting Equivalent Circuit Data." ANSYS, Inc.

Publication List

The scientific results of this thesis have been partly published in the following journal papers and conference proceedings.

Journal Publications

C. Yuan, A. Schütz, D. Hohlfeld, and T. Bechtold, "Towards system-level simulation of a miniature electromagnetic energy harvester model," *Electronics*, vol. 12, no. 15, 2023. https://doi.org/10.3390/electronics12153252

Y. Rao, **C. Yuan**, G. Sadashivaiah, D. Hohlfeld, and T. Bechtold, "Efficient design optimization of a miniaturized thermoelectric generator for electrically active implants based on parametric model order reduction," *International Journal for Numerical Methods in Biomedical Engineering*, vol. 37, no. 10, 2021. https://doi.org/10.1002/cnm.3517

C. Yuan, D. Hohlfeld, and T. Bechtold, "Design optimization of a miniaturized thermoelectric generator via parametric model order reduction," *Microelectronics Reliability*, vol. 119, p. 114075, 2021. https://doi.org/10.1016/j.microrel.2021.114075

C. Yuan, S. Hu, and T. Bechtold, "Stable compact modeling of piezoelectric energy harvester devices," *COMPEL-The International Journal for Computation and Mathematics in Electrical and Electronic Engineering*, vol. 39 no. 2, pp. 467-480, 2020. https://doi.org/10.1108/COMPEL-07-2019-0305

C. Yuan, S. Kreß, G. Sadashivaiah, E. B. Rudnyi, D. Hohlfeld, and T. Bechtold, "Towards efficient design optimization of a miniaturized thermoelectric generator for electrically active implants via model order reduction and submodeling technique," *International Journal for Numerical Methods in Biomedical Engineering*, vol. 36, no. 4, 2020. https://doi.org/10.1002/cnm.3311

S. Bouhedma, Y. Rao, A. Schütz, **C. Yuan**, S. Hu, L. Fred, T. Bechtold, and D. Hohlfeld, "System-level model and simulation of a frequency-tunable vibration energy harvester," *Micromachines*, vol. 11, no. 1, 2020. https://doi.org/10.3390/mi11010091

S. Hu, **C. Yuan**, A. Castagnotto, B. Lohmann, S. Bouhedma, D. Hohlfeld, and T. Bechtold, "Stable Reduced Order Modelling of Piezoelectric Energy Harvesting Modules Using Implicit Schur Complement," *Microelectronics Reliability*, vol. 85, pp. 148-155, 2018. https://doi.org/10.1016/j.microrel.2018.03.026

Conference Publications

C. Yuan, A. Schütz, D. Hohlfeld, and T. Bechtold, "Matrix interpolation-based parametric model order reduction of a miniaturized electromagnetic energy harvester model," *22nd International Conference on Micro and Nanotechnology for Power Generation and Energy Conversion Applications (PowerMEMS)*, (Abu Dhabi, United Arab Emirates), 2023. DOI: 10.1109/PowerMEMS59329.2023.10417160

C. Yuan, D. Hohlfeld, and T. Bechtold, "Towards system-level simulation of an electromagnetic energy harvester model via equivalent circuit extraction from ANSYS Maxwell 3D," *24th International Conference on Thermal, Mechanical and Multi-Physics Simulation and Experiments in Microelectronics and Microsystems (EuroSimE)*, (Graz, Austria), 2023. DOI: 10.1109/EuroSimE56861.2023.10100816

C. Yuan, S. Hu, and T. Bechtold, "Stable parametric model order reduction of piezoelectric energy harvester by matrix interpolation," *IFAC-PapersOnLine*, vol. 55, no. 20, pp.217-222, 2022. https://doi.org/10.1016/j.ifacol.2022.09.098

A. Roy, G. Sadashivaiah, **C. Yuan**, M. Nabi, and T. Bechtold, "Modeling of thermoelectric generator via parametric model order reduction based on modified matrix interpolation," *In: M. van Beurden, N. Budko, W. Schilders (eds) Scientific Computing in Electrical Engineering. Mathematics in Industry*, vol. 36, pp. 275-283, Springer, Cham, 2021. https://doi.org/10.1007/978-3-030-84238-3_27

G. Sadashivaiah, **C. Yuan**, and T. Bechtold, "Nonlinear model order reduction of a thermal human torso model," *In: M. van Beurden, N. Budko, W. Schilders (eds) Scientific Computing in Electrical Engineering. Mathematics in Industry*, vol. 36, pp. 285-292, Springer, Cham, 2021. https://doi.org/10.1007/978-3-030-84238-3_28

C. Yuan and T. Bechtold, "Geometrically parametrized reduced order model of a miniaturized thermoelectric generator for electrically active implants," *22nd International Conference on Thermal, Mechanical and Multi-Physics Simulation and Experiments in Microelectronics and Microsystems (EuroSimE)*, 2021.
DOI: 10.1109/EuroSimE52062.2021.9410842

Y. Rao, S. Bouhedma, **C. Yuan**, T. Bechtold, and D. Hohlfeld, "Towards a system-level model of a tunable dual-frequency piezoelectric energy harvester," *21st International Conference on Thermal, Mechanical and Multi-Physics Simulation and Experiments in Microelectronics and Microsystems (EuroSimE)*, 2020.
DOI: 10.1109/EuroSimE48426.2020.9152721

C. Yuan, D. Hohlfeld, and T. Bechtold, "Design optimization of a miniaturized thermoelectric generator," *21st International Conference on Thermal, Mechanical and Multi-Physics Simulation and Experiments in Microelectronics and Microsystems (EuroSimE)*, 2020. DOI: 10.1109/EuroSimE48426.2020.9152746

Y. Rao, S. Kreß, **C. Yuan**, T. Bechtold, and D. Hohlfeld, "Thermoelectric energy harvesting in medical implants," *GMM-Workshop Energieautonome Sensorsysteme*, 2020.

S. Hu, **C. Yuan**, and T. Bechtold, "Quasi-Schur transformation for the stable compact modeling of piezoelectric energy harvester devices," in *Scientific Computing in Electrical Engineering*, (G. Nicosia and V. Romano, eds.), pp. 267-276, Springer, Cham, 2020. https://doi.org/10.1007/978-3-030-44101-2_25

C. Yuan, G. Sadashivaiah, E. B. Rudnyi, and T. Bechtold, "A combination of POD-based model order reduction and thermal submodeling for miniaturized thermoelectric generator," in *MikroSystemTechnik (MST) Congress*, (Berlin, Germany), pp. 738-741, 2019. https://ieeexplore.ieee.org/document/9012802

C. Yuan, G. Sadashivaiah, E. B. Rudnyi, and T. Bechtold, "Efficient design optimization of a thermoelectric generator by a combination of model order reduction and thermal submodeling techniques," in *Proceedings of the 33rd European Conference on Modeling and Simulation (ECMS)*, (Caserta, Italy), pp.290-295, 2019. https://doi.org/10.7148/2019-0290

O. S. Jadhav, **C. Yuan**, E. B. Rudnyi, D. Hohlfeld, and T. Bechtold, "Nonlinear model order reduction of thermoelectric generator for electrically active implants," *International Journal of Bioelectromagnetism*, vol. 20, no. 1, pp. 5-7, 2018. http://www.ijbem.org/volume20/number1/5-7.pdf

C. Yuan, O. S. Jadhav, E. B. Rudnyi, D. Hohlfeld, and T. Bechtold, "Parametric model order reduction of a thermoelectric generator for electrically active implants," *19th International Conference on Thermal, Mechanical and Multiphysics Simulation and Experiments in Microelectronics and Microsystems (EuroSimE)*, (Toulouse, France), 2018. DOI: 10.1109/EuroSimE.2018.8369946

C. Yuan, S. Hu, A. Castagnotto, B. Lohmann, and T. Bechtold, "Implicit Schur complement for model order reduction of second order piezoelectric energy harvester model," *MathMOD 2018 Extended Abstract Volume, ARGESIM Report 55, 9th Vienna Conference on Mathematical Modelling*, (Vienna, Austria), 2018. DOI: 10.11128/arep.55.a55255

O. S. Jadhav, **C. Yuan**, D. Hohlfeld, and T. Bechtold, "Design of a thermoelectric generator for electrical active implants," in *MikroSystemTechnik (MST) Congress* , (Berlin, Germany), pp. 402-405, 2017. https://ieeexplore.ieee.org/document/8278689