CONSISTENCY ANALYSIS BETWEEN LUMPED AND DISTRIBUTED PARAMETER MODELS

by

Randi Wang

A dissertation submitted in partial fulfillment of
the requirements for the degree of

Doctor of Philosophy

(Mechanical Engineering)

at the

UNIVERSITY OF WISCONSIN–MADISON

2021

Date of final oral examination: 01/28/2021

The dissertation is approved by the following members of the Final Oral Committee:

Vadim Shapiro, Professor, Mechanical Engineering
Krishnan Suresh, Professor, Mechanical Engineering
Tom N. Krupenkin, Professor, Mechanical Engineering
Matthew S. Allen, Professor, Engineering Physics
Dan Negrut, Professor, Mechanical Engineering
Morad Behandish, Senior Research Scientist, Palo Alto Research Center

ACKNOWLEDGMENTS

Not all the students who want to become a PhD candidate could have such an opportunity; not all the PhD candidate who wants to get guidance and inspiration from the leading authority could have such a lucky chance; not all the PhD candidate who want to gain working experience during the PhD study could have such good fortune. My advisor, Dr. Vadim Shapiro helped me realize all my dreams about a PhD trip. Still clearly remember those days when I was struggling with my first journal paper writing, it was my advisor, who sat beside me and instructed me sentence by sentence. Still clearly remember those days when I did not know how to formulate a good PhD topic due to my lack of practical experience. It was my advisor, who found me an internship to let me detect existing practical problems independently. My advisor guided me with his patience and love, which was carved into my mind and would be staying there forever. The PhD study has an end, but the effects my advisor has on me will not.

Another person who has a deep influence on me is Morad Behandish, my PhD co-advisor. During my internship at PARC, I worked with him, learned from him and, was profoundly inspired by his passion and spirit on daily work, which is a kind of endless energy to handle everything and take fate into his own hands. His positiveness towards life encouraged and pushed me to overcome the obstacles from my work and daily life one after another. With his encouragement and help, I went through those hard times and well prepared for challenges in the future.

I really appreciate my parents for their vision and support to let me pursue my dreams in a foreign country. So many nights, it was my parents who accompanied me, comforted me, and listened to my happiness and sorrow over the phone. I would always cherish my friendship with my colleagues at the Spatial Automation Lab. Thank Mikola Lysenko, Goldy Kumar, Xingchen Liu, Brian McCarthy, Yaqi Zhang, Xin Liu, and Parag Pathak for many useful technical discussions and their colorful life stories. Last, it is so fortunate to have Professors Krishnan Suresh, Tom N.Krupenkin, Matthew S.Allen, and Dan Negrut on my committee. Their constructive suggestions and guidance are important for me to complete this thesis.

CONTENTS

LIST OF FIGURES

CONSISTENCY ANALYSIS BETWEEN LUMPED AND DISTRIBUTED PARAMETER MODELS

Randi Wang

Under the supervision of Professor Vadim Shapiro
At the University of Wisconsin-Madison

The system-level design represented by lumped parameter models (LPM) usually comes first for the high-value engineering design innovation at the functional level, followed by a geometric design represented by the distributed parameter models (DPM). Nevertheless, the non-unique mapping from lumped parameters to distributed shape and material properties leads to an ill-posed design problem. Before solving this problem, it is critical to have a well-defined concept of the "consistency" between LPM and DPM and find a systematic way to check the consistency.

The simulations of LPM and DPM start from different model specifications whose correspondence is difficult to be established but is indispensable for comparing simulation results. Simulating these two models usually results in solutions that have different state dimensions hence cannot be directly compared. The only reliable way nowadays to compare model solutions is a posterior testing through point-to-point comparison of the solved variables, however, it is unfavorable due to the high computational cost for large-scale models, unstable and non-convergent simulation solutions, etc. In addition, lumped modeling languages such as Modelica, Simulink, etc. differ in syntax and informal semantics, which sets a barrier to find a unified way for model consistency analysis.

We propose a general model consistency analysis framework to establish the correspondence of model specifications and solutions between LPM and DPM, which is independent of any modeling languages and tools, numerical methods and supports different types of physical models. The common semantics of the lumped parameter system is proposed and it can be in principle extended to spatially distributed systems. A simulation-free scheme is proposed to compare LPM and

spatially-discretized DPM, where only the model specifications are used to provide a priori guarantees of the simulations. The scheme supports any spatial discretization methods in principle. In particular, a model order reduction technique that a priori guarantees the accuracy, stability, and convergence is adapted by the scheme to resolve the high time cost issue caused by large model scales. We demonstrate the validity, time efficacy, and generality of the scheme by applying it to analyze the model consistency of different single- and multi-physics problems.

1 INTRODUCTION

1.1 Motivation

Lumped and distributed parameter models are commonly used to model physical systems and are key to engineering design. Determining *consistency* between underlying models of the system-level design (e.g. Modelica models [38]) and the geometric design (e.g. Computer-Aided Design (CAD) models [86]) is critical for the integration of these two design activities [39, 46, 113]. System-level design concentrates designers on functional aspects of the system, in which geometric aspects of the system are abstracted by a discrete set of lumped parameters such as spring stiffness, thermal conduction and electrical resistance, etc. This abstraction enables approximating the spatially continuous and distributed behavior. Lumped parameter models (LPMs) can capture such approximation by encapsulating the spatially distributed nature of engineering systems into a network of lumped components. The behavior of LPMs is governed by a system of ordinary differential equations (ODEs) whose state variables are functions of time [69]. In contrast, distributed parameter models (DPMs) explicitly account for the geometric and material properties of engineering systems, whose behavior is governed by a system of partial differential equations (PDEs) where state variables are functions of both time and space [82]. These PDEs can be approximated by a typically large system of ODEs using numerical discretization methods [14], whose solution could be prohibitive due to large time/memory requirements, especially for assemblies consisted of many geometric parts. System-level design often comes first for the high-value design innovations at a conceptual and functional level. To enable such innovations, designers have to search extremely high-dimensional design spaces, which is challenging enough even in a finite-dimensional state space with lumped parameters. Adding spatial degrees of freedom (e.g., arbitrary part shapes and material distributions) on top of that would make the problem completely intractable. Once the system-level design is obtained through searching the more manageable lumped parameter design space, the next step is the geometric design. However,

since the mapping from lumped parameters to distributed shape and material properties is not unique, we have an ill-posed design problem. Before we look for approaches to solve this problem, it is critical to have a well-defined notion of the concept of "consistency" between LPM and DPM. Informally, it represents a test to check if we succeeded in realizing a geometric design based on a system-level design. To the best of our knowledge, there is a lack of a formal framework to systematically establish the consistency between LPMs underlying the system-level design and DPMs underlying the geometric design. We propose to fill this gap in this thesis.

Figure 1.1 shows a process of the geometric design of an automobile based on its system-level design. The process starts from a system model built in Modelica [38], which consists of connected lumped components such as springs, electrical resistors, thermal resistors, etc. In the geometric design based on this system model, lumped components will be realized by geometric parts to satisfy the functional requirements. As shown in the upper left of the figure, the mechanical stiffness/-damping and thermal conductivity of automobile components are modeled by springs/dampers and conductors in the system model, regardless of their precise geometric realizations. For each component above, two options of geometric parts are provided to realize the component. For each selected geometric part, designers need to check if it can behave as intended by its corresponding lumped component (① in the figure). This check is usually conducted by manually adding initial and boundary conditions to each selected geometric part, simulating it then comparing the simulation result with that of the corresponding lumped component. Nevertheless, in such manual checks, different engineers usually have different interpretations of the lumped source terms (e.g. forces, voltage sources) and initial conditions, which leads to different translated initial and boundary conditions added to the geometric part, hence resulting in different DPM simulation results and comparison results. For example, suppose there is a lumped force added to the lumped spring-damper component (① in the figure). The force could be translated into pressure multiplied by a boundary area of the geometric absorber. However, the decision of which portion of the geometric boundary to add the pressure and

which spatial distribution (uniform, sine, and cosine, etc.) that the pressure should follow differ from person to person. *In addition, the component-by-component replacement may not guarantee consistency at the system level*. For instance, the geometric sub-assembly of the front suspension system at the upper right of the figure is obtained by component-by-component replacement from part of the system model, however, their behaviors are usually not consistent. There are several common reasons for such inconsistent behaviors, such as inappropriate mating conditions used in assembling the geometric parts, new auxiliary parts added for assembling geometric parts, and unignorable new contact problem between two connected geometric parts, etc. Therefore, the consistency between the geometric assembly and the system model cannot be naturally derived from the consistency at the component level. The geometric design process will end with finding at least one geometric design whose behavior is consistent with the system design behavior (② in the figure).

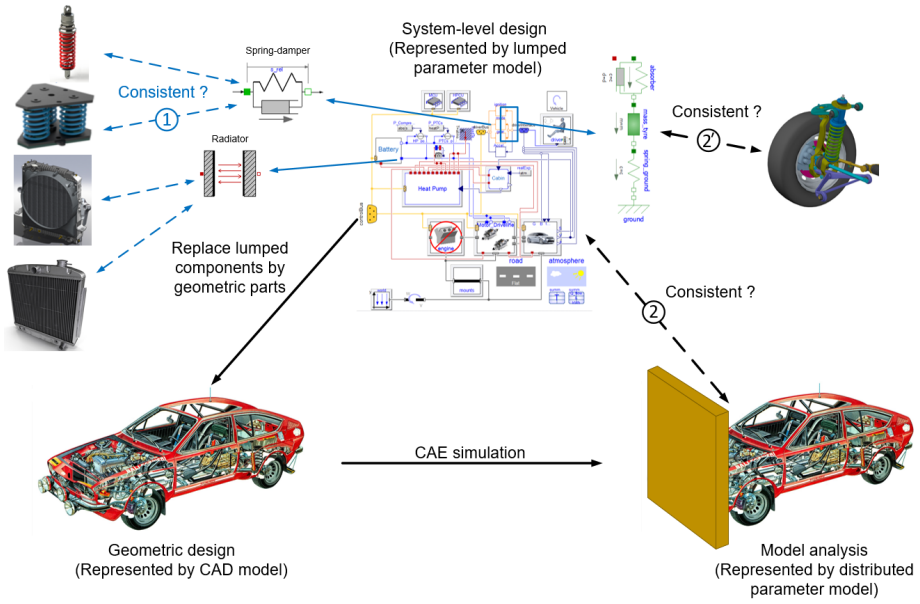


Figure 1.1: Consistency analysis in the process of system-to-geometric design

There are several challenges in checking the consistency between system and geometric designs. The first challenge is that lumped modeling languages (linear

graphs [104], bond graphs [69], Modelica [38], Simulink [33], etc.) differ in syntax, informal semantics, and in the methods by which the governing equations are generated and simulated, which has become a barrier to develop a unified framework for systematically establishing the consistency between LPMs and DPMs created by different languages and simulation tools. For example, in Simulink, components exchange numeric information uni-directionally and are not subject to conservation laws; by contrast, the energy flow between components is bi-directional in other languages such as linear graph, bond graph, and Modelica, satisfying conservation laws. Parallel and serial junctions in a bond graph do not specify the ordering of branches in the junctions, which implies that every bond graph in fact corresponds to a family of (dynamically) equivalent graph-based models in other languages. Many languages generate state equations using efforts and flows as variables, but their integral forms may also be used, for example in bond graphs. Furthermore, the challenge is compounded by the difference of distributed parameter modeling tools in geometric representations (e.g. boundary representation and function or implicit representation, etc.), spatial discretizations and numerical schemes used for computations (finite difference [80, 82], finite element [14], and finite volume [82] methods, etc.). The first goal of this thesis is to establish formal correspondence between concepts and constructs in distinct lumped modeling languages and tools to pave the way for developing a unified framework for systematically establishing consistency between LPMs and DPMs created in different languages and simulation tools.

The simulations of LPM and DPM start from different pre-known model specifications, for instance, the initial conditions and lumped sources of LPM and the initial and boundary conditions of DPM. A correct correspondence of specifications is the prerequisite for comparing simulation results afterward, which is a necessary condition for determining model consistency. The second challenge is systematically establishing the correspondence of specifications between LPM and DPM. For example, DPMs have boundary conditions but they do not exist in LPMs since there is no spatial boundary in LPMs. How to systematically map the boundary conditions of DPMs to physical quantities of the LPM is nontrivial. Figure 1.2 shows

such an example, where an LPM and a DPM of a suspension system are given. The DPM has a Dirichlet boundary condition of displacement $u = u_0$ at boundary $\partial\Omega_2$ and a Neumann boundary condition pressure $p = p_0$ at boundary $\partial\Omega_1$. How to systematically map these two types of boundary conditions to physical quantities of the LPM needs to be explored. In addition, the simulation of LPMs and DPMs usually lead to different solutions (e.g. different number of states) that are not directly comparable. After simulation, for example, the unknown displacement field "u =?" of the DPM in Figure 1.2 is usually represented by thousands of states but the number of the solved displacement states of LPM is only three. How to map and compare these two sets of states in a time-efficient way is nontrivial. The second goal of this thesis is to systematically establish the correspondence for both specifications and solutions for a wide variety of physical systems.

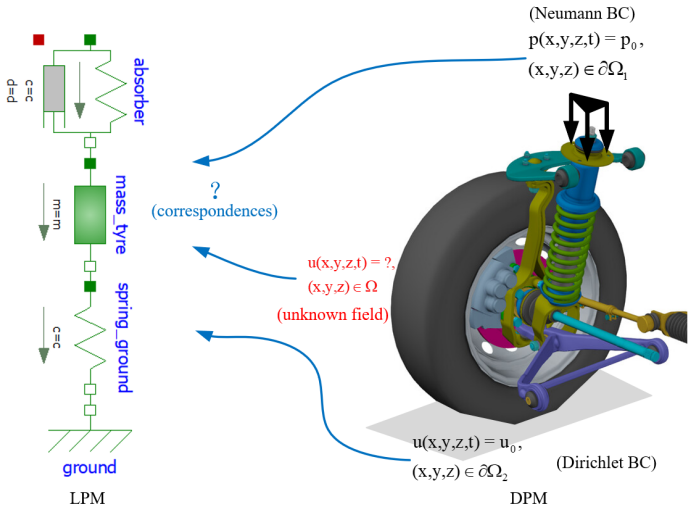


Figure 1.2: The correspondence of specifications between LPM and DPM is non-trivial

The third challenge is that the only reliable way nowadays to compare model solutions is a posterior testing by point-to-point comparison of lumped and distributed model "variables" after simulating both, which is not favorable for several reasons such as the low computational efficiency for large-scale models, unstable and nonconvergent simulation solutions, etc. The third goal of this thesis is to

present a simulation-free scheme for comparing model solutions such that it can a priori compute difference between two model solutions solely from comparing lumped and distributed model “parameters", without resorting to simulations. Admittedly, it is generally not possible to have a scheme that could provide *strong* a priori guarantees of consistency between *any* types of LPM and DPM without simulating both models and directly comparing the results. Nonetheless, *it is important to understand to what extent such a priori analyses are possible and quantify their limitations.*

1.2 Approach

In the present study, a general framework (shown in Figure 1.3) is proposed to check the consistency between system and geometric designs, which supports establishing the correspondence of model specifications and solutions between LPM and DPM created in different languages and simulation tools.

In the framework, to establish the formal correspondence between concepts and constructs in distinct lumped modeling languages and tools, we use tools from algebraic topology and well-known classification of physical theories developed over the years by Tonti [111, 112], Roth [102], Branin [26], Kron [72], and others. Specifically, Tonti diagrams classify a variety of physical theories in terms of physical laws (topological and constitutive) and could be used to describe LPMs [112]. Such diagrams are independent of any specific linguistic constructs, implementation assumptions, or numerical simulation schemes, and hence could become common semantic description for models built by Modelica [38], Simulink [33], linear graphs [104] and bond graphs [92], etc. Importantly, the classification generalizes to higher-dimensional physical models, suggesting that the proposed algebraic topological semantics can be extended to include spatially-distributed models represented by three-dimensional solid models [107], PDEs, and finite element methods [14].

In the framework, we propose to establish the correspondence of model specifications and solutions between LPMs and DPMs at three levels, namely, the correspondence at (1) model structure level (2) physical quantity type level (3)

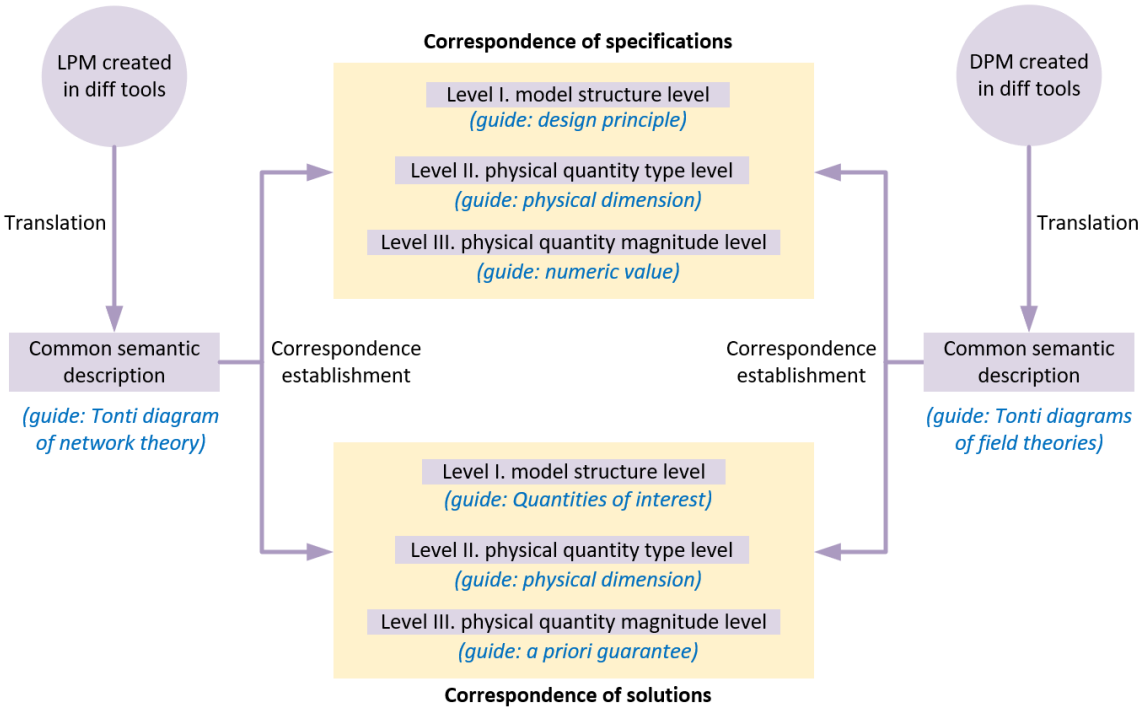


Figure 1.3: A framework to check the consistency between system and geometric designs

physical quantity magnitude level. Specifically, (1) accounts for the condition that the correspondence of physical quantities must be established between the lumped component and its own geometric realization. For example, in the design process, if the lumped spring (connected to the ground) in Figure 1.2 is realized by the tire in the DPM, then the displacement of the lumped spring can only correspond to the displacement field of the tire, not some other geometric parts (e.g. the absorber). This restriction guarantees the unification between the correspondence of physical quantities and the correspondence of model structures in the system and geometric designs. Notably, this restriction opens up an opportunity to automatically add initial and boundary conditions to the geometric design, provided the system design and design principle (e.g. component-to-component realization). (2) and (3) account for the condition that any two corresponding physical quantities of LPM and DPM must have the same physical dimension and magnitude after a certain

functional mapping. For example, the Neumann boundary condition of the DPM in Figure 1.2 is a pressure p, which is generally corresponded to a lumped force source F though integral $\int_S dS$, a linear functional, where S represents the pressure area. In particular, the relationship between any two physical variables associated with the same time and space elements in the Tonti diagram of LPM and the Tonti diagram of DPM of the same physical type can be used to define and search such functional mappings, which provides a solid foundation for automatically adding appropriate initial and boundary conditions to the geometric design of different physical systems. For example, the DPM model in Figure 1.2 can be described by the Tonti diagram of elastodynamics [112], where the Neumann boundary condition (i.e., imposed tractions on the boundary) is associated with the dual time and surface elements. By contrast, the lumped mass-spring-damper model in Figure 1.2 can be described by the Tonti diagram of transnational mechanical system [112], where the physical quantity associated with the dual time and surface elements is the lumped force source. The difference of the physical dimension between the traction and the lumped force source is L^2 so it is natural to relate them with a Reimann integral $\int_S dS$.

We propose a simulation-free scheme to check if a given LPM M_l and a given spatially-discretized DPM M_d have similar simulation solution values. In the scheme, spatial discretization techniques are required to project infinite-dimensional state space of PDEs of the DPM to the finite-dimensional subspace of ODEs or DAEs, and model order reduction (MOR) techniques are adapted to accelerate the computation process and hence the procedure has a high time-efficiency for the large-scale model. The function of the scheme is to compute a priori error bound ε between the given LPM M_l and spatially-discretized DPM M_d that are represented in terms of their transfer function matrices obtained from a Laplace transform of the ODEs/DAEs. However, computing such error bound will be slow if the model scale is too large (e.g. in terms of the common $\mathcal{H}_2$ norm). To reduce the time cost, we will first use a MOR method that can provide a priori error bound to reduce the large-scale[1] semi-discretized DPM M_d to a small-scale surrogate LPM

[1]In this thesis, the given LPM M_l is supposed to be small-scale, otherwise, we need one more

M_r. Suppose the obtained a priori error bound is ε_1. Then we compute the error ε_2 between transfer functions of the surrogate LPM M_r and the given LPM M_l, which is usually rapid because of the small model scale. After that, the a priori error ε between transfer functions of the given LPM M_l and the semi-discretized DPM M_d satisfies $\varepsilon \leqslant \varepsilon_1+\varepsilon_2$, which is obtained from the triangular inequality.

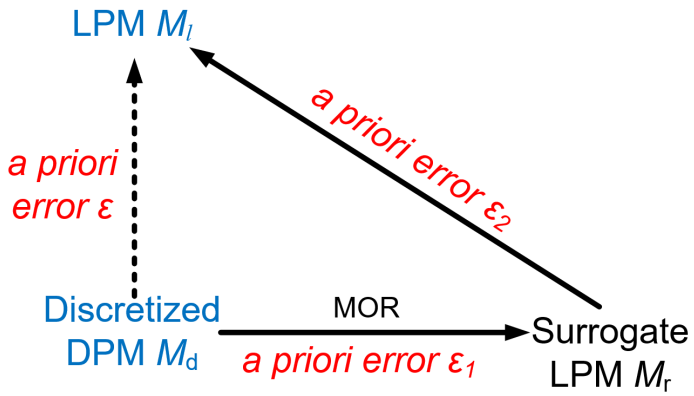


Figure 1.4: Compute a priori error bound by using triangular inequality

In particular, examples of spatial discretization techniques that can be applied to the DPM are finite difference [80, 82], finite element [14], finite volume [82], spectral basis [55], isogeometric analysis [65], and other methods that fall under Galerkin projections using conforming or non-conforming mesh and meshfree methods (e.g. particle-based methods) [105, 67, 78]. We illustrate our scheme with an embodiment that uses finite element analysis (FEA); however, the MOR method to be used afterward is not limited nor affected by the choice of numerical methods. The adapted MOR method to illustrate the scheme is CUmulative REduction (CURE) scheme [88], where many model reduction algorithms could be embedded, such as iterative rational Krylov algorithm (IRKA)[60], stability-preserving, adaptive rational Krylov (SPARK) algorithm [88], etc. The CURE scheme can seamlessly, systematically, automatically, and rapidly generates a family of different orders of ROMs by applying rigorous upscaling to the full-order models (FOMs). Importantly, it provides relatively tight a priori error bounds for all generated ROMs

step to reduce its order as we do to the semi-discretized DPM M_d.

and therefore opens the way for the procedure to fast evaluate the a priori error between the surrogate LPM M_r and spatially-discretized DPM M_d.

1.3 Contributions

The thesis introduces a framework for model consistency determination between LPMs and DPMs. It has four major contributions as follows:

1. Unified semantics for lumped parameter systems, which provides a basis for interoperability in systems modeling. Such semantics provides a single neutral format that can be translated to/from models in different lumped parameter modeling languages such as Modelica, Simulink, bond graphs, and linear graphs. Importantly, the study paves the way to go beyond LPM to DPM, meaning that the proposed formal semantic model could in principle be extended to spatially distributed models, which would eliminate the need for customized consistency analysis between LPMs and DPMs created in different modeling languages and tools.

2. A framework for establishing the correspondence of model specifications and solutions between LPMs and DPMs, which is independent of any modeling languages and tools, numerical methods, and supports a variety of different types of physical models. The framework provides a solid theoretical basis for developing an automatic model consistency analysis tool.

3. A scheme to compare LPM and spatially-discretized DPMs before simulations, where only model specifications are used to provide upfront guarantees of the simulations. The scheme can in principle be used alongside any spatial discretization methods. Particularly, a MOR technique that a priori guarantees accuracy, stability, and convergence is adapted to address the low time efficiency problem caused by large model scales.

4. We show the application of this framework to determine the consistency between LPMs and DPMs of different single- and multiple-physics systems, which lays a solid foundation for automation of the system-to-geometric design.

1.4 Outline

The rest of this thesis is organized as follows. Chapter 2 is a detailed survey of previous efforts related to the consistency between lumped and distributed parameter models. In Section 2.1, we introduce the significance of the LPM and the DPM in physical modeling and simulations, the difference of their solution forms, and the commonly-used physical modeling languages and tools for lumped and distributed parameter models. After that, previous works on the system-to-geometric design are investigated in Section 2.2. Results of these works contribute to the motivation of this thesis because none of them proposed a standard or systematic approach to evaluate the validity of the geometric design with respect to the given system design. Nevertheless, the comparison of solutions between LPMs and DPMs can be found in many published papers. In Section 2.3, we summarize and compare different strategies adapted in these papers and point out the reason why they cannot be directly used for solving the model consistency problem formulated in this thesis.

Chapter 3 is an illustration of the general framework for model consistency analysis. In Section 3.1, we define a few critical terminologies to scope our proposed model consistency problem, namely the LPM, the DPM, initial condition correspondence, boundary condition correspondence, field correspondence, the behavior of LPM, the behavior of DPM, and model consistency, followed by formulating the problem using these definitions. The framework to systematically check the consistency between a given LPM and a given DPM is introduced in Section 3.2, where we first propose the strategy of using common semantics of physical models to avoid expensive customized model consistency analysis of LPMs and DPMs created in different physical modeling languages and tools, then introduce how to systematically establish the correspondence of model specifications and solutions. After that, we propose a simulation-free scheme in Section 3.2.3 to compare LPMs and semi-discretized DPMs.

The main results of the thesis are contained in Chapters 4 and 5. The common semantics for lumped parameter systems is introduced in Chapter 4. The distinct

semantics issue of lumped parameter modeling languages and tools is introduced in Section 4.1. We summarize the well-known algebraic topological model of physical systems in Section 4.2; this model serves as the basis for Tonti diagrams that classify physical variables, laws, and theories [112]. Single-domain lumped parameters systems are examined in detail in Section 4.3. Because all such models are isomorphic in terms of their physical variables, laws, and Tonti diagrams, we only discuss the models of electrical systems – with the understanding that the discussion also applies to lumped parameter models in other physical domains. In particular, we show that paths over the corresponding Tonti diagram correspond to all possible ways to generate state equations for an electrical network, including well-known node-potential and mesh-current methods that date to Maxwell [81]. While the discussion and the semantics proposed in Section 4.3 apply to *any one* single-domain lumped-parameter system model, we consider the more general multi-domain systems in Section 4.4 (that may combine electrical, mechanical, hydraulic, thermal, and other single-domain subsystems). We show that all such systems may be represented by an algebraic topological model in terms of generalized state variables associated with Tonti diagrams.

In Chapter 5, we introduce the approaches underlying the proposed simulation-free scheme to compare a given LPM and a given DPM, followed by applying the scheme to different mechanical and thermal problems. Spatial discretization methods that can be used to generate the numerical equation of the DPM are first introduced in Section 5.1.1, followed by a brief comparison of the highlighted properties of different commonly-used MOR approaches in Section 5.1.2, among which the CURE scheme proposed in [88] is selected to accelerate generating the a priori guarantee between the given LPM and DPM. The CURE scheme has many rigorous properties, such as ensuring a priori error guarantees, stability, convergence, and automatic model order decision. In Appendix A.1 mentioned in this Section, we introduce how this CURE scheme is gradually evolved from the classical Krylov subspace MOR approach, whose MOR principle is to enforce the first several Taylor expansion terms of the transfer functions to be the same between FOM and ROM. Nevertheless, the classical Krylov subspace MOR approach cannot

provide a priori error guarantees and automatically select the expansion points. Many advanced rigorous MOR methods are proposed based on the same principle afterward yet have more rigorous pragmatic properties. The CURE scheme is one of them.

In Section 5.2, our proposed framework is applied to check the model consistency for several pairs of given LPMs and DPMs from mechanical and thermal domains, where both single and multiple domains problems are included. We show that if the order (i.e., the number of variables in the governing equation) of the discretized DPM is larger than a certain value, comparing two model solutions using our scheme will be faster than using the numerical simulation, and a high comparison accuracy is guaranteed. In our examples, this value is around several thousand but slightly different problem by problem. Importantly, the improvement of the time efficiency accelerates with respect to the increase of the discretized DPM order. At the end of each example, we use simulation results to show the validity and tightness of the obtained a priori error bound between the given LPM and DPM. We conclude in Chapter 6 with consequences of the proposed semantic unification for lumped parameter models, the framework for model consistency analysis, and open issues.

2.1 Lumped and distributed parameter system modeling

Informally, the significance of LPMs can be attributed to two main factors. First, LPMs are the simplest models of system behavior (usually, but not always dynamic) that encapsulate and abstract away all detailed geometric information. This allows modeling behavior of systems without dealing with a detailed embodiment of such systems – either because these details are too complex to simulate or not known (for example in early design stages). Secondly, the LPM provides a common ODE based framework to model many different engineering systems (mechanical, thermal, electrical, fluid, etc.) without dealing with their differences when it comes to spatial/geometric modeling. Paynter's work [92] originally focused on methodology for use of lumped parameters models for design and analysis of engineering systems and culminated in the language of bond graphs. He described a systematic method of converting a general system to a simplified LPM by a process of "reticulation" (creating a network model through discretization of a spatially-distributed system) and developed a symbolic language of *bond graphs* for describing such models. A major conceptual and practical significance of bond graphs is that they model all lumped parameter systems in terms of a small number of abstract generalized variables: efforts, flows, their derivatives, integrals, and constitutive relationships. This allowed bond graphs to represent LPMs of complex physically heterogeneous systems that are coupled using the abstract transformer and gyrator elements and can be easily translated into a system of ordinary differential equations.

Arguably, all LPMs and languages are based on concepts similar to those used in bond graphs and are now a standard textbook material [69, 104]. LPMs abstract a spatially and temporally distributed system by a network of components. The purpose of the network is to model the flow of energy through the system's components as a function of time; the detailed geometry of the system is accounted

for by integral properties of the components. Formally, the LPM is described in terms of differential-algebraic equations with variables that are specific to physical phenomena modeled by the network. However, all such models are isomorphic based on the well-known "analogies" between different physical theories [69]. In each physical domain, the power is a product of two generalized variables that are classified as across or through (in the case of Modelica and linear graph models), or effort and flow (in case of bond graphs). The network includes abstract sources of these variables, and transforms the energy by storing or dissipating in three types of elements that are commonly described by analogy with electrical networks as generalized resistors, inductances, and capacitors. In addition, the energy may be transformed between different physical domains using the transformer and gyrator components in the network. More general components with complex behavior may be incorporated into the network, but all such components may be modeled as a composition of these basic components listed above [92]. The formal reason for these analogies is the common underlying algebraic topological structure that we will discuss in Section 4.2.

The differences between various modeling languages stem from their approach to describing the basic components, their connectivity in the network, and the method by which the governing system of equations is generated. Thus, the bond graph uses 0 and 1 junctions to respectively represent parallel and serial connections of components; the linear graph uses directed edges to connect components (nodes), while Modelica/Simulink uses connectors to link library components. The behavior of components is described in terms of state variables that are related by constitutive relations: bond graph associates effort, flow, and constitutive relationships to 1-ports; linear graph associates across and through variables to directed edges, while Modelica and Simulink associate these variables with blocks. The governing equation of the LPM is a system of ODEs (or more generally DAEs) that are generated from such descriptions by methods that differ in the choice of state variables and generation algorithms and all such equations describe the identical dynamical behavior of the modeled system that is determined by initial conditions.

By contrast, DPMs are developed for describing both spatial and temporal be-

haviors of physical systems [77]. Such models are commonly used in modeling a number of important physical phenomena such as deformation of flexible structure [12], thermal conduction [91], fluid dynamics [23], electromagnetism [47], and their combinations. The behavior of such models is usually described by the solution of a system of PDEs, with the consideration of initial and boundary conditions of interest, that are subjected to numeral analysis [80]. A critical step of the numerical analysis is reformulating the equations to the discrete representation, a system of ODEs that can be solved by modern computers. Such discretization is generally accompanied by loss of information about the original continuous equations. FEM [14], FVM [40], and FDM [82] are three commonly-used numerical methods for solving PDEs [6] which differ in the ways of discretization of PDEs. Specifically, FEM converts the PDEs to the variational form and restricts it to finite-dimensional subspaces. FVM converts the PDEs to integral form and expresses the conservation law in terms of integral variables on the control volume. FDM approximates the partial differential operators by linear combinations of function values at the grid points. ANSYS [2], Comsol [84], and Abaqus [1] are commonly-used commercial software for distributed parameter modeling and analysis, where different numerical methods are adopted. Being an important step of the product design and analysis, distributed parameter modeling and simulation is critical for predicting the performance of geometric designs, which could dramatically reduce the product development cost due to prototyping [110].

2.2 System to geometric design

The system-to-geometric design represents a design procedure where a system-level design is firstly built using LPMs, followed by a geometric design at the level of details of an engineering drawing. As remarked by Ulrich [113], this design strategy can reduce the complexity and clarify the procedure of solving the design problem. Most approaches developed for the system-to-geometric design are based on the one-to-one correspondence between the lumped component in the LPM and the geometric part in the repository of solid models. Particularly,

bond graphs are commonly adapted in these approaches to build the system-level design. For example, Finger and Rinderle proposed a component database that contains the correspondence between ports of bond graphs and geometric parts for a limited class of mechanical designs [46]. Engelson developed an integrated environment that combines geometric design tools and system modeling tools to assist engineers to construct and verify large, moving rigid-body assemblies [39]. However, these two approaches may result in unrealistic design due to ignoring functional-sharing. Ulrich's work is an exception [113], where a single geometric part simultaneously implements several functions. His approach is characterized as a design-and-debug process, where geometric parts are first selected based on the component-to-component realization using the lumped components in LPM and then reconfigure the selected geometric parts for function sharing. However, the approach is only valid for single-input and single-output systems. By contrast, the approach proposed by Prabhu and Taylor in [97] provides a basis for the synthesis of multiple-input and multiple-output systems whose specifications containing spatial orientations and positions, in which the augmented bond graphs are used by introducing new 2-ports representing orientation and position transformers, however, functional-sharing is not considered in this approach. Functional-sharing is critical for realizing realistic design and product evolution. Potential ways to deal with function-sharing include but are not limited to the methods introduced in [56, 57, 71], where parts having relative motions can be combined into monolithic compliant mechanisms. However, none of the above works of the system-to-geometric design introduce how to establish the correspondence between specifications of the system and geometric designs in the design process and none of them introduce an approach to evaluate the validity of the geometric design.

2.3 Comparison of simulation solutions between different physical models

The comparison of model specifications and solutions between LPM and DPM can be found in two research areas. The first one is the model conversion which refers to the process of converting the spatially-discretized DPM to the LPM, in which the dimension of the state space before and after the conversion is not changed. The second is the MOR [3, 20, 94, 87] ,whose goal is to simplify the large-scale LPM or spatially-discretized DPM to a small-scale system of ODEs that have a smaller dimension of state space. Previous works related to the model conversion and reduction and how model simulation solutions are compared before and after the conversion and reduction will be respectively reviewed below.

Model conversion techniques are widely used in computer graphics where the deformable objects such as skin, cloth fabrics, soft tissue, etc. can be automatically converted to lumped mass-spring model to realize fast deformable object simulation owing to the simplicity and computational efficiency of LPMs [83, 73, 13, 66]. For example, Gelder [51] derived a lumped-spring element consisted of three lumped springs linked end to end in a series connection from the geometric angle and length information of the two-dimensional linear triangular element. Authors in [16, 15] extend Gelder's work to rectangular elements. They use the linear rectangular finite element as the reference to generate the lumped mass-spring element by using the energy method and the least action principle [45]. The simulation results of the converted lumped-spring model and the original finite element model were compared in terms of each pair of the corresponding nodal displacement and lumped mass displacement. Researchers in [85, 75] proposed to convert the finite element model of 3-dimensional deformable objects to the corresponding lumped mass-spring models by minimizing the difference of the stiffness matrices of these two models. The difference of model behaviors before and after the conversion is formulated as the spatial variation rate of the difference between the nodal displacement in the finite element model and the corresponding lumped mass displacement. All the converted lumped mass-spring models in the literature have

the same number of displacement states as the given finite element model. The strategy for comparing model simulation results adapted in model conversion techniques cannot be directly used to solve the model solution comparison problem between the system-level design and geometric design that is proposed in this thesis because the number of state variables of the system-level design is usually much smaller than that of the spatially-discretized geometric model. There is no one-to-one correspondence between these two sets of state variables as we have in the model conversion problems.

In contrast to model conversion, the comparison of different dimensional behaviors exist in MOR, which is usually used in reducing the number of states of several classes of mathematical models of large-scale dynamic systems such as first-order linear time-invariant (LTI) systems [3, 9, 21, 52, 58, 59], second-order LTI systems [106, 32, 5, 20], port-Hamiltonian (LTI) systems [94, 54], parametric (LTI) systems [42, 87, 76], and nonlinear (time-invariant) systems [34, 50, 101], etc., aiming to find a good approximation of the full order model (FOM) that is numerically efficient and stable and simultaneously preserves certain system properties, for instance, preserving the number of inputs and outputs for first-order LTI systems, the definiteness of constitutive matrices (e.g. mass, spring and damping matrices) for second-order LTI systems, the port-Hamiltonian structure for port-Hamiltonian LTI systems, and parameter dependency for parametric LTI systems. Large-scale dynamic systems generally come from spatially discretized DPMs cite spatial discretization methods [82, 14, 40] or large-scale LPMs consisted of a large number of lumped components [5]. Because of different used principles, which method can give us the best approximation is case by case. For example, the balanced truncation method [32, 99, 109] removes the states that have both weak controllability and observability while the rational Krylov subspace method [9, 10, 48, 59] matches several most significant terms of the Tayler series expansion of the ROM transfer function, expanded around carefully selected frequencies, to those of the FOM. So, on one hand, if we use different MOR methods for accelerating a posterior model solution comparison, then the answer for the solution similarity may swing between yes and no; on the other hand, the strategy for model solution comparison before

and after MOR cannot be directly used for the model solution comparison problem between the system-level and geometric designs proposed in this thesis. The reason is that the correspondence between state variables before and after model reduction is automatically derived by the approach. However, such correspondence in our proposed model consistency analysis problem is not pre-known and has to be determined by users, which varies with respect to different design principles and model application scenarios.

3 GENERAL FRAMEWORK FOR MODEL CONSISTENCY ANALYSIS

In this chapter, we will introduce the framework for model consistency analysis between lumped and distributed parameter models. The challenges underlying the model consistency analysis were briefly introduced in the motivation section, but critical terminologies such as LPM, DPM, and model behaviors, etc. still have not well-defined. Definitions of these terminologies are necessary to propose the model consistency problem. Although such definitions can be found in many literature, those definitions are either too broad or narrow for the topic of this thesis so they will be re-defined in the context of model consistency analysis. Below, we will first introduce the definition of these terminologies, then formulate the problem in terms of these definitions, and after that propose the framework for solving the problem.

3.1 Definitions and problem formulation

Before formulating the problem of model consistency analysis between lumped and distributed parameter models, we will introduce several related definitions, namely *LPM*, *DPM*, *initial and boundary conditions correspondence*, *field correspondence*, *behavior of LPM*, and *behavior of DPM*.

• *DPM* is a mathematical model of the well-posed initial and boundary value problem, whose specification is (i) the PDE, (ii) the region of space-time on which the PDE is required to be satisfied, and (iii) initial and boundary conditions to be met [74]. The PDE of the DPM can be symbolically represented as follows:

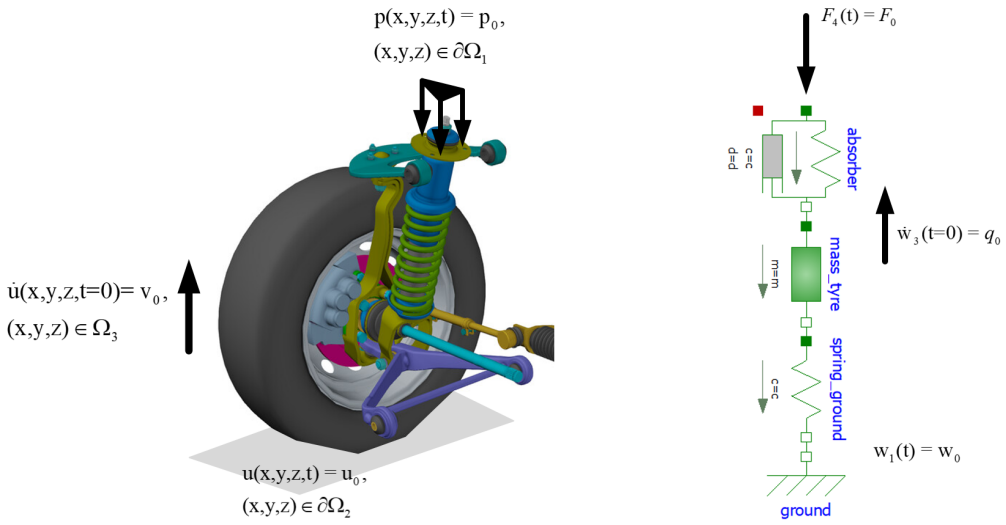
$$F\left(x, y, z, t, \partial_x u, ..., \partial_t u, \partial^2_{xx} u, \partial^2_{xy} u, ..., \partial^n_{x...t} u; \boldsymbol{\mu}\right) = 0, \tag{3.1}$$

where u is a dependent variable that depends on the independent space variables x, y, z and time variables t. F involves derivatives of u with respect to at least one of the independent variables. $\boldsymbol{\mu}$ is a set of time and/or space-dependent parameters that generally have certain physical meanings such as mass density, fluid viscosity,

and thermal conductivity, etc. The solution of interest of $F = 0$ is generally required in some region of space and time and there are boundary conditions (i,e, Dirichlet, Neumann and Robin conditions [74]) and initial conditions (e.g. $u(x, y, z, 0) = 0$) to be satisfied. For example, Figure 3.1a shows a DPM of the automobile suspension system, whose governing equation is the Eq.3.2, expressed by components with respect to the rectangular Cartesian coordinate system $(Oxyz)$.

$$\mu u_{i,jj} + (\lambda + \mu) u_{i,ij} + f = \rho \ddot{u}_i, \tag{3.2}$$

where the material is assumed to be isotropic and homogeneous. u_i represents the displacement and the comma represents the partial differentiation with respect to the coordinates. ρ is the density of material, λ is Lamé's first parameter, μ is the shear modulus, and f is the body force. The geometric domain is where the equation to be satisfied in space. The displacement $u = u_0$ and pressure $p = p_0$ are respectively Dirichlet and Neumann boundary conditions and the velocity $\dot{u} = v_0$ is the initial condition.



(a) DPM (b) LPM

Figure 3.1: Examples of DPM and LPM of a suspension system

• *LPM* is a mathematical model of the well-posed initial value problem, which can be used to model the lumped parameter system consisted of a network of connected lumped components such as lumped springs, electrical resistors, and thermal conduction, etc. The specification of the LPM is (i) a system of ODEs, (ii) the region of time on which the ODEs are required to be satisfied, and (iii) initial conditions. The ODEs of the lumped parameter model can be symbolically represented as follows:

$$\mathbf{G}\left(t, \mathbf{w}, \mathbf{w}_t, ...; \theta\right) = \mathbf{0}, \tag{3.3}$$

where $\mathbf{w}$ is a vector of a finite number of state variables that depend on the time variables t. θ is a set of time-dependent parameters that generally have certain physical meanings such as lumped spring stiffness, voltage source, hydraulic capacitance, and thermal resistance, etc. The solution of interest of $\mathbf{G} = \mathbf{0}$ is generally required in some time intervals of interest hence there will be some initial conditions (e.g. $\mathbf{w}\left(0\right) = \mathbf{0}$) to be satisfied. For example, Figure 3.1b shows an LPM of the automobile suspension system, whose governing equation is as follows:

$$\mathbf{m}\ddot{\mathbf{w}} + \mathbf{d}\dot{\mathbf{w}} + \mathbf{k}\mathbf{w} = \mathbf{F}, \tag{3.4}$$

where $\mathbf{w}$ is a $n \times 1$ vector of displacement (with respect to reference) where n is the number of states; $\mathbf{F}$ is a $n \times 1$ vector of lumped force source and $\mathbf{F} = [0, 0, 0, F_0]^T$ in this example; $\mathbf{m}$, $\mathbf{d}$ and $\mathbf{k}$ are $n \times n$ lumped mass, damping and stiffness matrices, respectively; w_1=w_0 is the displacement reference; $t \in (0, +\infty)$ is the time domain where the ODEs to be satisfied; $\dot{w}_3 = q_0$ is the initial condition representing the initial velocity of the lumped mass.

Below we will introduce definitions of the correspondence of the initial and boundary conditions and the field of interest between a pair of given LPM and DPM, where they are supposed to model the same physical system.

• *Initial conditions correspondence* between the LPM and the DPM is a functional T_I that maps the initial conditions of DPM to the initial conditions of LPM.

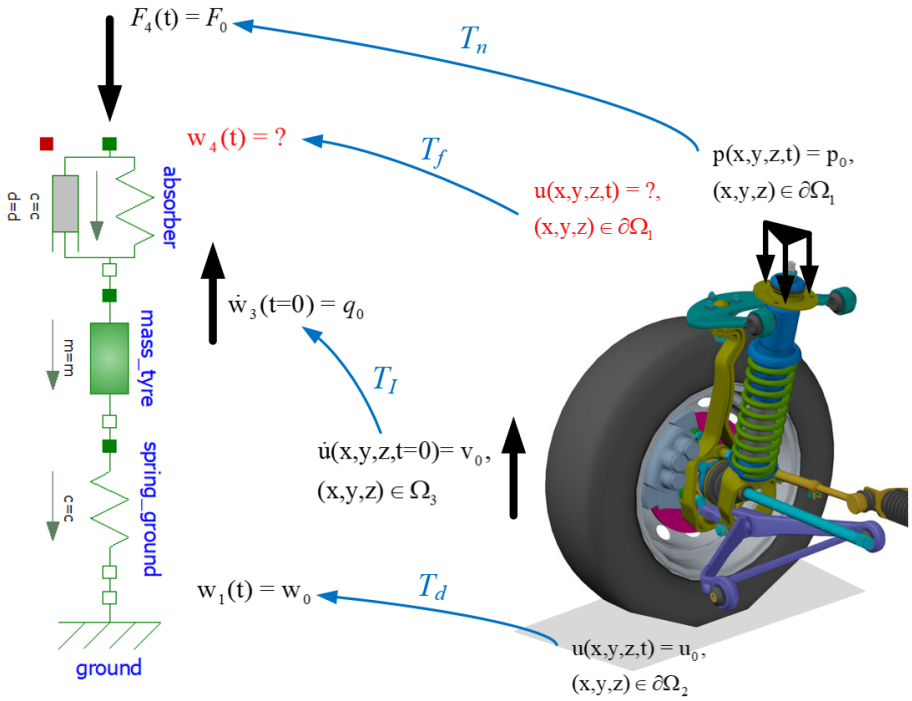


Figure 3.2: The correspondences of initial conditions T_I, boundary conditions T_B, and fields T_f between LPM and DPM of the suspension system

$$T_I : u^{(i)}(x, y, z, 0) \rightarrow \mathbf{w}^{(i)}(0), \tag{3.5}$$

where i represents the order of time derivative. An example of T_I is shown in Figure 3.8, where it maps the initial velocity of the tire barycenter to the initial velocity of the lumped mass $\dot{u} \rightarrow \dot{w}_3$.

• *Boundary conditions correspondence* between the LPM and the DPM is a functional set T_B consisted of two functionals: T_d that maps the Dirichlet boundary conditions of DPM to state variables $\mathbf{w}(t)$ and T_n that maps the Neumann boundary conditions of DPM to lumped source parameters $\theta_s(t) \in \theta$ of the LPM, for example, lumped force source in mechanical mass-spring-damper model and lumped voltage source in electrical resistor-capacitor-inductor model. The form of these two functionals are given as follows:

$$T_d : u(\bar{x}, \bar{y}, \bar{z}, t) \to \mathbf{w}(t), \qquad (\bar{x}, \bar{y}, \bar{z}) \in \partial\Omega \tag{3.6}$$

$$T_n : \frac{\partial u(\tilde{x}, \tilde{y}, \tilde{z}, t)}{\partial n} \to \theta_s(t), \qquad (\tilde{x}, \tilde{y}, \tilde{z}) \in \partial\Omega, \tag{3.7}$$

where $\partial\Omega$ represents the boundary of the spatial domain Ω. The examples of T_d and T_n are shown in Figure 3.8. Specifically, T_d maps the displacement of contact surface of the DPM between tire and ground to the displacement of lumped spring connecting to the ground in LPM $u(\bar{x}, \bar{y}, \bar{z}, t) \to w_1(t)$ and T_n maps the pressure on the top surface of absorber to the lumped force added to the lumped spring-damper component in LPM $p(\tilde{x}, \tilde{y}, \tilde{z}, t) \to F_4(t)$.

• *Field correspondence* between LPM and DPM is a functional T_f that maps the field of DPM to state variables of LPM.

$$T_f : u(x, y, z, t) \to \mathbf{w}(t) \tag{3.8}$$

An example of T_f is given in Figure 3.8, where T_f maps the unknown displacement of the top surface of absorber in the DPM to the absolute displacement of lumped spring-damper component in the LPM $u(\hat{x}, \hat{y}, \hat{z}, t) \to w_4(t)$.

• *Behavior of DPM* B_d is the exact solution $u = \bar{u}(x, y, z, t)$ obtained by solving the DPM, i.e., the PDE (Eq.3.1) that satisfies the region of space-time and initial and boundary conditions.

• *Behavior of LPM* B_l is the exact solution $\mathbf{w} = \bar{\mathbf{w}}(t)$ obtained by solving the LPM, i.e., the ODEs (Eq.3.3) that satisfy the region of time and initial conditions.

• A LPM and a DPM have *consistency* if the norm[1] of difference between the behavior of DPM B_d after mapping T_f and the behavior of LPM B_l is less than a positive real number $\epsilon > 0$, providing T_I and T_B. Symbolically,

$$\|T_f(u) - \mathbf{w}(t)\| \leqslant \varepsilon \tag{3.9}$$

[1]The norm could be any type of vector norms. The commonly-used norms are L^1, L^2, and L^∞.

Since the exact solution of most DPMs cannot be obtained, the spatial discretization schemes such as FEM [14], FVM [40], and FDM [82], etc. are usually used to generate a system of ODEs whose solution is an approximation of the exact solution. The spatial discretization resolution of the DPM to generate a system of ODEs is usually determined by the user based on practical experience. Generally, the finer discretization, the better approximation. However, there is always a trade-off between the computational efficiency and the accuracy of the numerical result. We suppose herein the spatial discretization resolution selected by the user is sufficiently fine to obtain a good approximation u' of the exact solution u. In other words, the error between u and u' caused by spatial discretization is supposed to be small and has little effect on the model consistency determination result[2]. Based on this supposition and all the above definitions, we formulate the problem of model consistency determination between the LPM and the DPM as follows.

Problem formulation: given a LPM M_l, a DPM M_d, the correspondences T_I, T_B, and T_f between M_l and M_d, and the spatial discretization of M_d, determine if $\|\Gamma_f(u') - \mathbf{w}(t)\|$ is smaller than a positive real number $\varepsilon' > 0$, such that the LPM M_l and the DPM M_d have consistency.

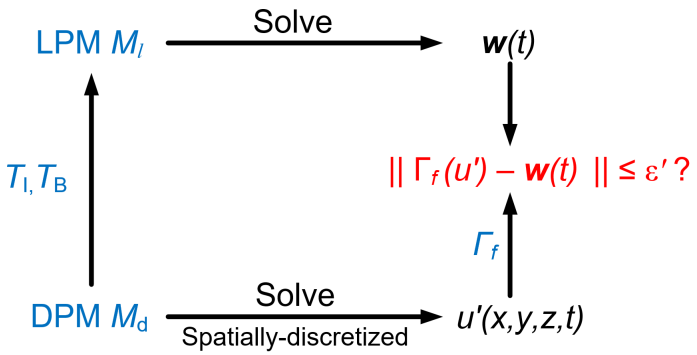


Figure 3.3: Problem formulation

The problem formulation can be described by a diagram shown in Figure 3.3, where the blue parts are pre-known and the red part is what to estimate. The

[2]In practice, the error between the exact and the numerical solution that is generated from a fine spatial discretization could be covered by moderately loosening the user-defined error threshold.

behavior of DPM is discretized in space and Γ_f is the dicrete form (in space) of T_f. If the norm $\|\Gamma_f(u') - \mathbf{w}(t)\|$ is less than a user-defined real number $\varepsilon' > 0$, then M_l and M_d have consistency, otherwise, they are inconsistent. Notably, if any of the provided correspondences T_I and T_B is incorrect (i.e., the values or physical dimensions before and after the mapping are not the same), M_l and M_d will be instantly determined to be inconsistent. Below we will introduce a simulation-free scheme to compute the a priori upper bound of $\Gamma_f(u') - \mathbf{w}(t)$ in terms of the L^∞ norm.

3.2 Framework for model consistency analysis

We aim to develop a general framework for model consistency analysis, where two technical challenges need to be solved. The first challenge is that the framework must be general, meaning that it can be used to check the consistency between lumped and distributed parameter models of different physical problems, supporting different physical modeling languages and tools. The second challenge is how to systematically establish the correspondence of specifications and solutions between lumped and distributed parameter models. In this section, we will introduce the approaches to solve these two challenges.

3.2.1 Unified semantic model of lumped parameter systems

To eliminate the need for a customized framework for the consistency analysis between DPMs and LPMs that are created in different modeling languages, we aim to propose a unified semantic model of lumped parameter systems such that we only need to develop the consistency analysis framework for the unified semantic model and it would naturally work for the other lumped parameter models. Lumped modeling languages such as Modelica [38], Simulink [33], bond graphs [69], and linear graphs [104] differ in syntax, informal semantics, and in the methods by which governing equations are generated and simulated. The unified semantic model we proposed for lumped parameter systems using standard notions from

algebraic topology. In particular, Tonti diagrams classify all physical theories in terms of topological and constitutive laws and could be used to describe different types of lumped parameter systems, supporting the algorithmic generation of all possible forms of the governing equations. This semantic model is independent of specific implementation assumptions, coordinates, linguistic constructs, or numerical simulation schemes. Notably, our proposed semantics of lumped parameter systems can be extended to include DPMs. Chapter 4 is our study results of such a common semantic model. Figures 3.4 and 3.5 illustrate the relation between the common semantic LPM and DPM represented by Tonti diagrams and the LPM and DPM created by different modeling languages and tools, for instance, Modelica, Simulink, linear graphs, and bond graphs for LPMs and Comsol, Abaqus, ANSYS and Nastran for DPMs. All the other physical models can be in principle translated from/to Tonti diagrams of certain types.

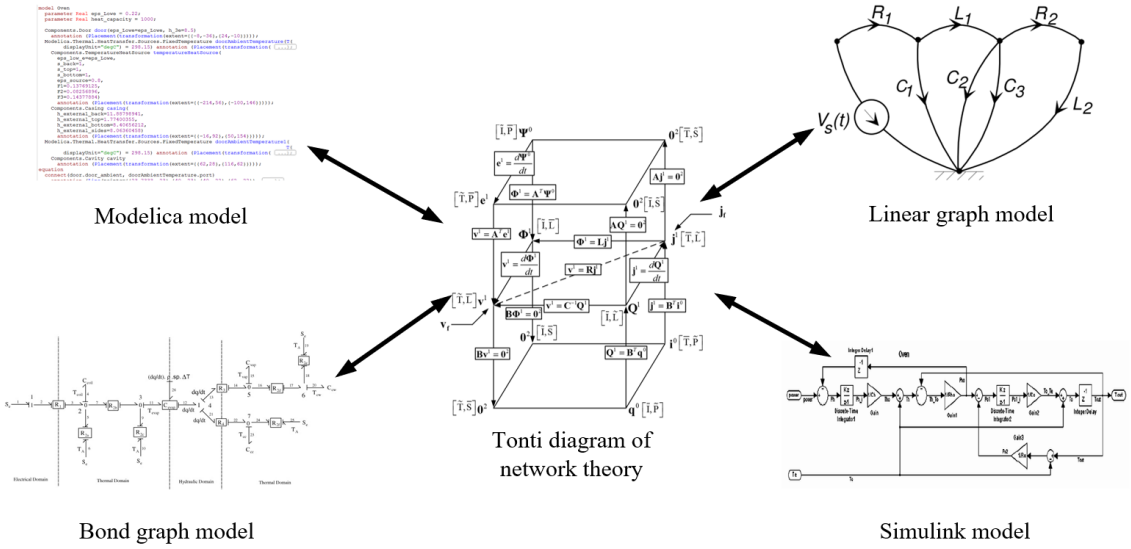


Figure 3.4: Common semantic LPM described by the Tonti diagram of network theory

Recall the LPM and the DPM examples of a suspension system in the problem formulation above. The LPM is a lumped mass-spring-damper model and the DPM

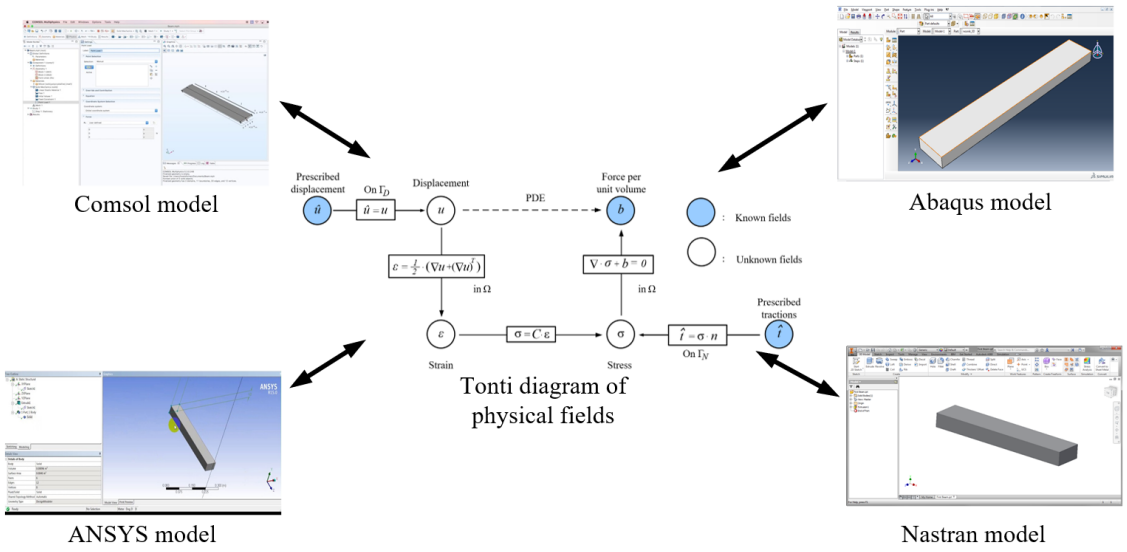


Figure 3.5: Common semantic DPM represented by Tonti diagrams of field theories

is an elastodynamic model, whose common semantic models described by the Tonti diagram are respectively shown in Figures 3.6 and 3.7, where symbols in circles represent the physical variables, the boxes on the vertical arrows represent topological/differential operators in time and space, and the other boxes represent the initial/boundary conditions or constitutive relations. The constitutive parameters in two Tonti diagrams are g (gravity acceleration), ρ (mass density), C (stiffness tensor), H(damping tensor), $\mathbf{M}$ (lumped mass matrix), $\mathbf{K}$ (lumped stiffness matrix) and $\mathbf{R}$ (lumped damping matrix). Importantly, by following paths over these two Tonti diagrams, the governing equations of these two models can be automatically generated, details of which can be found in Sections 4.3.

3.2.2 Establishing correspondence of model specifications and solutions

The correspondence of specifications and solutions between LPM and DPM of the same physical system will be established at three levels, which are the correspondences at (1) model structure level (2) physical quantity type level (3) physical

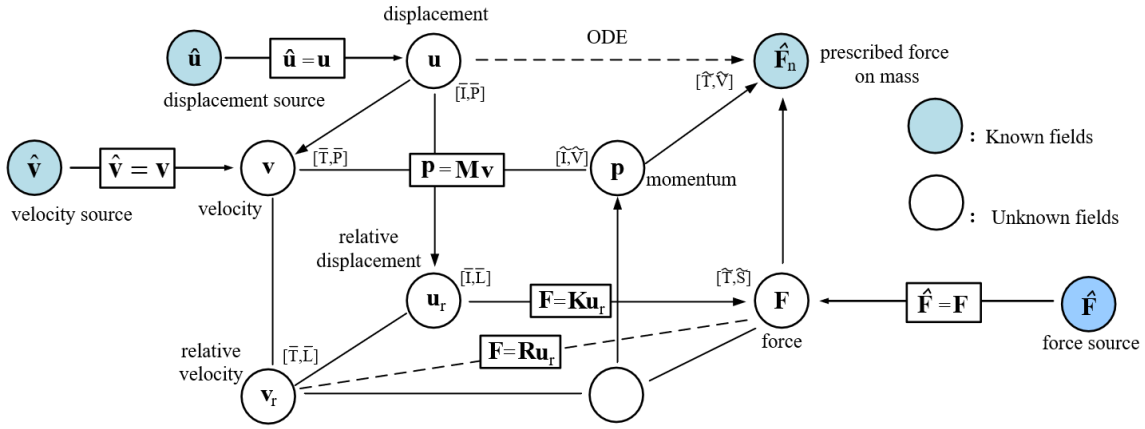


Figure 3.6: Semantic model of lumped mass-spring-damper models

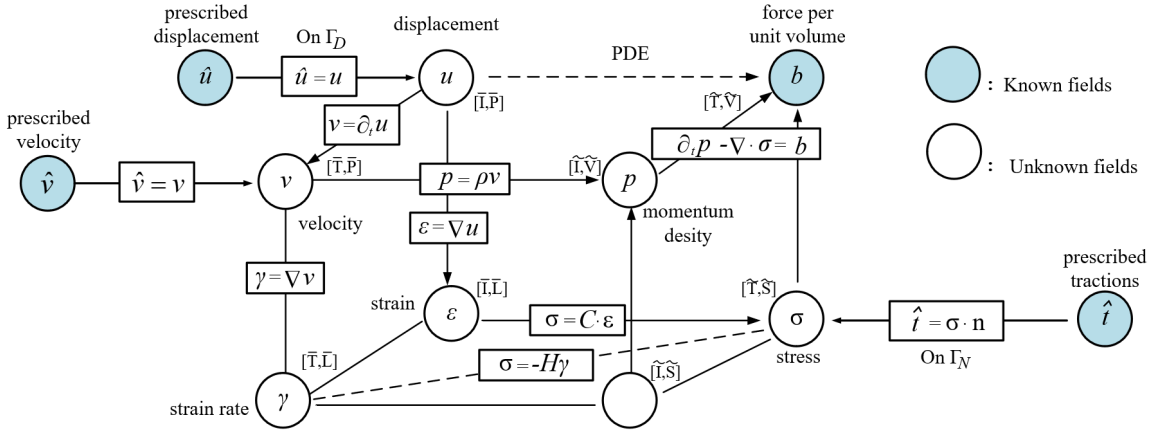


Figure 3.7: Semantic model of elastodynamic models

quantity magnitude level. We will introduce these three levels of correspondences one by one below.

(1) Correspondence at model structure level

As mentioned in the motivation, the system-to-geometric design is based on the component-by-component realization design principle, which means for each lumped component, we find a geometric part to realize it. Therefore, the correspondence T_I, T_B, and T_f between M_l and M_d must be built according to such realization. Specifically, suppose we have a system-level design, if the initial condi-

tion/lumped source/state is on a certain lumped component p of the LPM, then it must be mapped from the initial/boundary condition/field on the geometric part G_p of the DPM that corresponds to p via the functional $T_I/T_B/T_f$, as shown in the example of suspension system in Figure 3.8, copied below. Moreover, the particular spatial region of G_p to add initial/boundary condition/field is supposed to be determined by the function of G_p, i.e., how G_p would be used. At the model structure level, building correspondence T_I, T_B and T_f could be automated as long as the spatial region of G_p to add initial/boundary condition/field are pre-set on every geometric part in the library based on their functions. In the examples of this thesis, they are supposed to be given. For example, in Figure 3.8, to correspond to the lumped force $F_4(t) = F_0$ added to the lumped absorber (spring-damper) component, a pressure $p(x, y, z, t) = p_0$ is added to the top surface of the geometric part of the absorber.

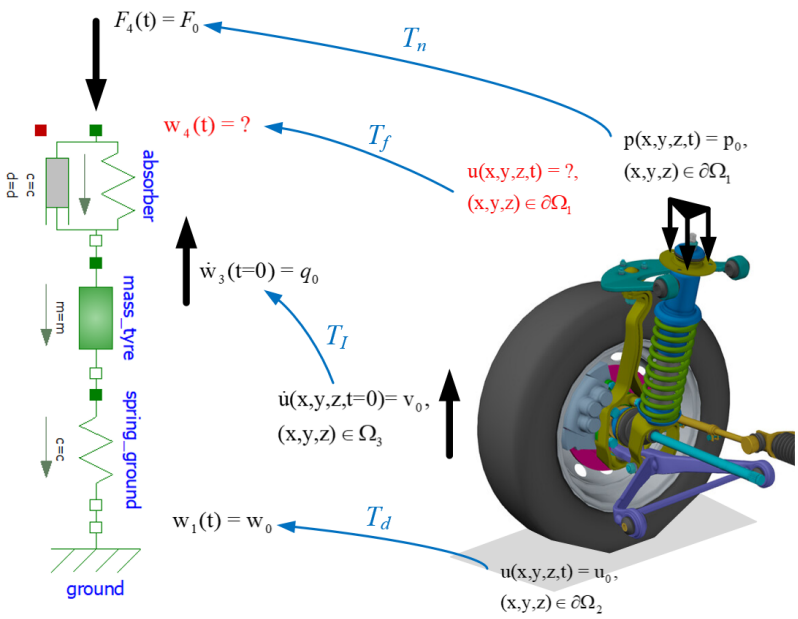


Figure 3.8: The correspondences of initial conditions T_I, boundary conditions T_B, and fields T_f between LPM and DPM of the suspension system at the model structure level

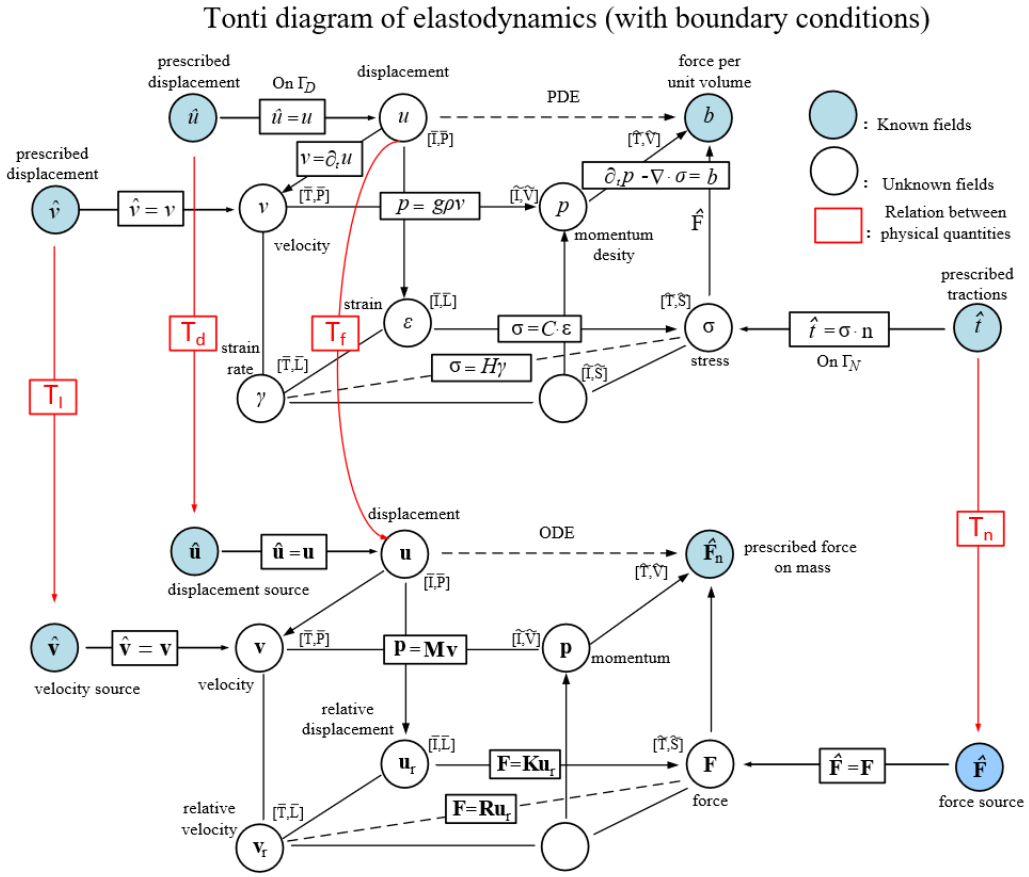


Figure 3.9: The correspondences of initial conditions T_I, boundary conditions T_B, and fields T_f between LPM and DPM of any mechanical systems at the physical quantity type level

(2) Correspondence at physical quantity type level

In Figure 3.8, how do we know the physical quantity added to the geometric part of the absorber that corresponds to the lumped force $F_4(t) = F_0$ should be the pressure $p(x, y, z, t)$, rather than other physical quantities, e.g. acceleration or momentum? This question seems easy to answer for people who are familiar with mechanical models. However, if the model is from other physical domains, similar questions might not be easy to answer. For example, suppose we have an

LPM of the electrical system where there is a lumped current source, what physical quantity would be used as the boundary condition added to the DPM? In fact, both questions above could be answered by using the Tonti diagrams of LPMs and DPMs. Particularly, the correspondence between physical quantities that are associated with the same time and space element[3] of Tonti diagrams describing the same physical system can be used to search such corresponding physical quantity and the functional mapping between them.

For example, the above question about the correspondence between $F_4(t)$ in M_l and $p(x, y, z, t)$ in M_d can be answered if we know the position of correspondences T_I, T_d, T_n, and T_f in relating physical quantities in the Tonti diagram of lumped mass-spring-damper models and the Tonti diagram of elastodynamic models, as shown in Figure 3.9. Since these two diagrams are describing the same physical system - a suspension system, there are correspondences between physical quantities that associated with the same space and time element. Note that T_n links the force source of the lower Tonti diagram and its counterpart, the prescribed traction (e.g. pressure) in the upper Tonti diagram, which explains the reason for using pressure as the boundary condition in DPM to correspond to the lumped source in LPM. Moreover, the commonly-used form of the T_n is the integral $\int_s dS$, where S represents the surface area where the prescribed traction acts. The form of T_n can be customized as long as it guarantees the physical dimension consistency between the functional mapping and two related physical quantities.

Importantly, unlike the sketch of the suspension system example shown in Figure 3.8, where T_* represents the correspondence of physical quantities for only one specific example, the linked Tonti diagrams in Figure 3.9 represent such correspondences between a wide class of mechanical LPMs and DPMs.

(3) Correspondence at physical quantity magnitude level

Since the LPM M_l and the DPM M_d model the same physical system, the magnitude of the physical quantities before and after the functional mapping T_I and T_B must be the same and before and after the functional mapping T_f must be similar. Take the T_n in Figure 3.8 for example, we need to guarantee $T_n p_0$ and

[3]The definition of time and space element can be found in [112]

F_0 have the same magnitude $|T_n p_0| = |F_0|$. Since the commonly-used T_n is $\int_S dS$, $\left|\int_S p_0 dS\right| = |F_0|$ must be held to establish the correspondence of F_0 and p_0 at the magnitude level. Below we will propose a simulation-free scheme to check if the magnitudes of the physical quantities before and after the functional mapping T_f are similar.

3.2.3 A priori guarantee for model solutions correspondence establishment

Suppose we are given the LPM M_l, the DPM M_d and established correspondences T_I and T_B, below we will introduce a scheme to compute the a priori error between LPM M_l and spatially-discretized DPM M_d after mapping Γ_f, where no simulation of differential equations is needed. The visualized steps of this scheme are shown in Figure 3.10.

With the given an LPM M_l and a DPM M_d, we first generate the governing equations of these two models, a system of second-order ODEs for LPM M_l and a system of PDEs for DPM M_d, which can be realized either using existing modeling languages and tools or paths over corresponding Tonti diagrams, depending on how M_l and M_d are represented. Then, applying the spatial discretization method (e.g. FEM [14]) to the PDEs to generate a system of second-order ODEs for the DPM M_d. After that, we convert the obtained ODEs of LPM M_l and DPM M_d to their equivalent state-space form by linear transformation [7], with the results shown in Eq.3.10 and 3.11.

$$\begin{aligned} \mathbf{E}_l \dot{\mathbf{x}}_l(t) &= \mathbf{A}_l \mathbf{x}_l(t) + \mathbf{B}_l \mathbf{u}_l(t) \\ \mathbf{y}_l(t) &= \mathbf{C}_l \mathbf{x}_l(t) + \mathbf{D}_l \mathbf{u}_l(t) \end{aligned} \tag{3.10}$$

$$\begin{aligned} \mathbf{E}_d \dot{\mathbf{x}}_d(t) &= \mathbf{A}_d \mathbf{x}_d(t) + \mathbf{B}_d \mathbf{u}_d(t) \\ \mathbf{y}_d(t) &= \mathbf{C}_d \mathbf{x}_d(t) + \mathbf{D}_d \mathbf{u}_d(t) \end{aligned} \tag{3.11}$$

Eq.3.10 is the state space form $(\mathbf{E}_l, \mathbf{A}_l, \mathbf{B}_l, \mathbf{C}_l, \mathbf{D}_l)$ of LPM M_l, usually having tens to hundreds of state variables; Eq.3.11 is the state space form $(\mathbf{E}_d, \mathbf{A}_d, \mathbf{B}_d, \mathbf{C}_d, \mathbf{D}_d)$

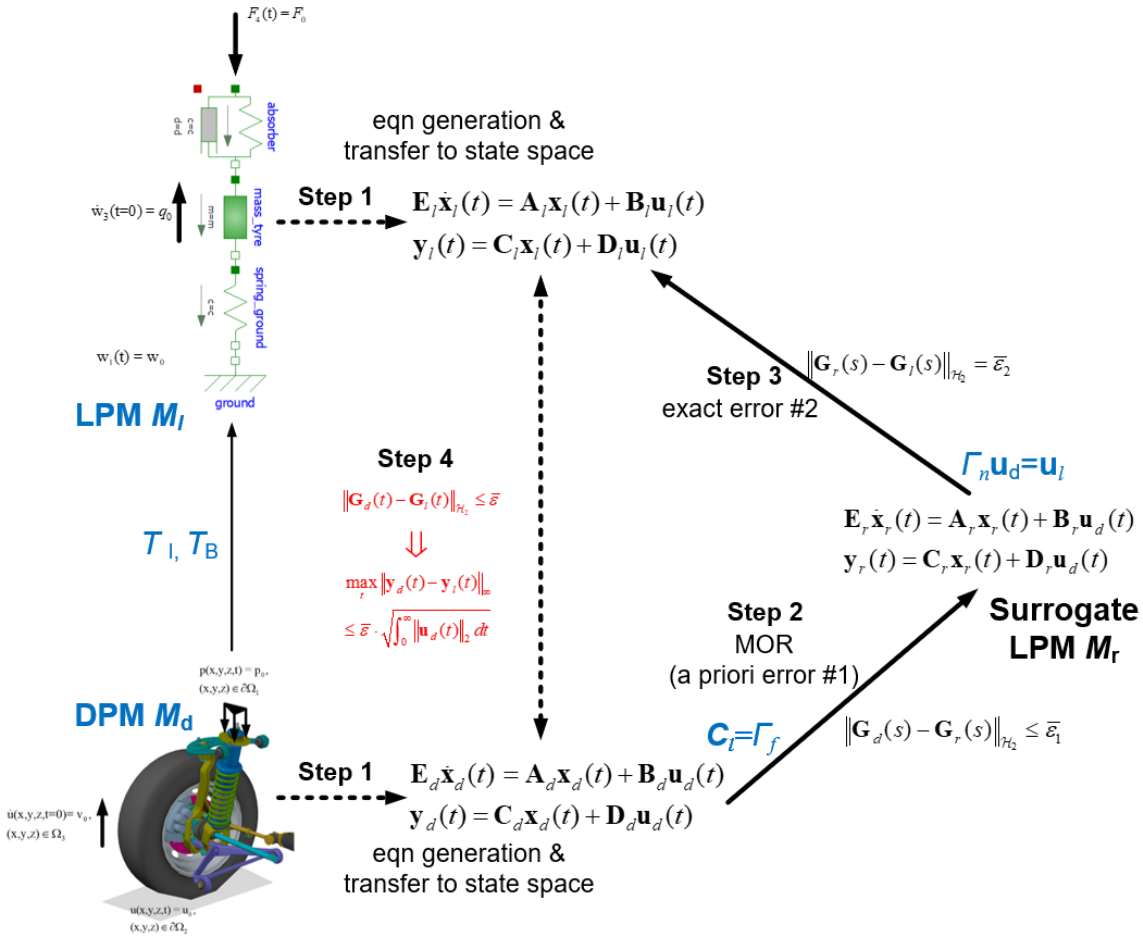


Figure 3.10: Simulation-free scheme to compute a priori error between LPM M_l and DPM M_d

of the numerical model of DPM M_d, usually comprising tens or hundreds of thousands, if not millions, of state variables. In these two equations, $\mathbf{x}(t)$ is the vector of state variables, $\mathbf{y}(t)$ is the vector of outputs of interest, $\mathbf{u}$ is the vector of inputs, $\mathbf{E}$ is the descriptor matrix, $\mathbf{A}$ is the dynamic matrix, $\mathbf{B}$ is the input matrix, $\mathbf{C}$ is the output matrix, and $\mathbf{D}$ is the feed-forward matrix. Particularly, $\mathbf{C}$ and $\mathbf{D}$ are defined by the user based on the expected outputs of interest and $\mathbf{D}$ is usually selected to be $\mathbf{0}$, which means the input will not directly affect the output without going through the model.

With the given T_B, T_I, T_f, and the spatial discretization of the DPM M_d, we have (1) $\Gamma_n \mathbf{u}_d = \mathbf{u}_l$, where Γ_n is the discrete form of Neumann boundary condition correspondence T_n ($\in T_B$) (2) $\Gamma_I \mathbf{x}_d(0) = \mathbf{x}_l(0)$, where Γ_I is the discrete form of initial conditions correspondence T_I (3) $\mathbf{C}_d = \Gamma_f$, which is the discrete form of field correspondence T_f. With $\mathbf{C}_d = \Gamma_f$, $\mathbf{y}_d$ now represents the behavior B'_d of the discretized DPM M_d after mapping Γ_f and $\mathbf{y}_l$ represents the behavior of LPM B_l. Since $\Gamma_n \mathbf{u}_d = \mathbf{u}_l$, substituting $\mathbf{u}_l$ with $\Gamma_n \mathbf{u}_d$ in Eq.3.10 generates the state space form $(\mathbf{E}_l, \mathbf{A}_l, \mathbf{B}'_l, \mathbf{C}_l, \mathbf{D}'_l)$ as follows, where $\mathbf{B}'_l = \mathbf{B}_l \Gamma_n$ and $\mathbf{D}'_l = \mathbf{D}_l \Gamma_n$. With such substitution, Eq.3.10 is changed to its equivalent form Eq.3.12. It can be observed that the Eq.3.11 of the discretized DPM M_d and Eq.3.12 of the LPM M_l have the same inputs $\mathbf{u}_d$.

$$\begin{aligned} \mathbf{E}_l \dot{\mathbf{x}}_l(t) &= \mathbf{A}_l \mathbf{x}_l(t) + \mathbf{B}'_l \mathbf{u}_d(t) \\ \mathbf{y}_l(t) &= \mathbf{C}_l \mathbf{x}_l(t) + \mathbf{D}'_l \mathbf{u}_d(t) \end{aligned} \tag{3.12}$$

Below, we will explain how to evaluate the difference between outputs $\mathbf{y}_d$ and $\mathbf{y}_l$ with respect to the infinity norm. Generally, there are two approaches to compute $\|\mathbf{y}_d(t) - \mathbf{y}_l(t)\|_\infty$. The first and also the commonly-used posterior approach is using simulation to obtain $\mathbf{y}_d(t)$ and $\mathbf{y}_l(t)$ and then find out the maximum difference in the whole time domain. Considering the model scale of the discretized DPM M_d, stability, and convergence issues, this strategy will not be considered. The second approach is generating a tight a priori error guarantee for this infinity norm without simulation. Below we will introduce the second approach.

It is well-known that the solutions of Eq.3.12 and 3.11 in the Laplace domain have the form as follows [29]:

$$\mathbf{y}_l(s) = \underbrace{\left(\mathbf{C}_l(s\mathbf{E}_l - \mathbf{A}_l)^{-1}\mathbf{B'}_l + \mathbf{D'}_l\right)}_{\mathbf{G}_l(s)} \mathbf{u}_d(s) \tag{3.13}$$

$$\mathbf{y}_d(s) = \underbrace{\left(\mathbf{C}_d(s\mathbf{E}_d - \mathbf{A}_d)^{-1}\mathbf{B}_d + \mathbf{D}_d\right)}_{\mathbf{G}_d(s)} \mathbf{u}_d(s) \tag{3.14}$$

, where s stands for frequency; $\mathbf{G}_l(s)$ and $\mathbf{G}_d(s)$ respectively denote transfer function of the LPM M_l and transfer function of the discretized DPM M_d. The transfer function is the input-output relationship between the input $\mathbf{u}$ and the output $\mathbf{y}$ in the Laplace domain.

According to [4], the maximum difference between $\mathbf{y}_d(t)$ and $\mathbf{y}_l(t)$ of two models that have the same input $\mathbf{u}(t)$ has an upper bound, which is the product of $\|\mathbf{G}_d(s) - \mathbf{G}_l(s)\|_{H_2}$ and $\sqrt{\int_0^\infty \|\mathbf{u}_d(t)\|_2 dt}$, as shown in Eq.3.15. Particularly, to satisfy this inequality equation, $\mathbf{u}_d$ must be a finite energy input (i.e., $\int_0^\infty \|\mathbf{u}_d(t)\|_2^2 \, dt < \infty$). A simple example of such input is that $\mathbf{u}_d(t)$ is a positive constant real vector at the time interval [0,1], but zero at the rest of the time domain.

$$\max_t \|\mathbf{y}_d(t) - \mathbf{y}_l(t)\|_\infty \leqslant \|\mathbf{G}_d(s) - \mathbf{G}_l(s)\|_{H_2} \cdot \sqrt{\int_0^\infty \|\mathbf{u}_d(t)\|_2 dt} \tag{3.15}$$

It can be observed from Eq.3.15 that for a specific discretized DPM M_d having a finite energy input, the term $\sqrt{\int_0^\infty \|\mathbf{u}_d(t)\|_2 dt}$ is a fixed value. As long as the value of term $\|\mathbf{G}_d(s) - \mathbf{G}_r(s)\|_{\mathcal{H}_2}$ approaches to zero, then term $\max_t \|\mathbf{y}_d(t) - \mathbf{y}_l(t)\|_\infty$ would approach to zero as well. In other words, the similarity between $\mathbf{y}_d(t)$ and $\mathbf{y}_l(t)$ with respect to the infinity norm in the time domain is determined by the similarity between $\mathbf{G}_d(s)$ and $\mathbf{G}_l(s)$ with respect to the $\mathcal{H}_2$ norm in the frequency domain.

The advantage of computing the value of $\mathcal{H}_2$ norm of a transfer function is that no numerical simulation of ODEs is needed; Instead, only a system of algebraic Lyapunov equations is required to be solved [93, 62]. However, if the model scale is large, as the $\mathbf{G}_d(s) - \mathbf{G}_l(s)$ we have, then computing its $\mathcal{H}_2$ norm, $\|\mathbf{G}_d(s) - \mathbf{G}_r(s)\|_{\mathcal{H}_2}$, is still time-consuming and requires a plenty of computational effort. Panzer proposed a MOR method called CUmulative REduction (CURE) scheme [88], which can fast compute a relatively tight H_2 upper bound of the transfer function between the FOM and its ROM. This scheme can be used to speed up computing the upper bound $\bar{\varepsilon}$ for $\|\mathbf{G}_d(s) - \mathbf{G}_l(s)\|_{H_2}$. Naturally, if $\bar{\varepsilon}$ approaches

zero, then $\|\mathbf{G}_d(s) - \mathbf{G}_l(s)\|_{H_2}$ approaches zero, and so does $\max_t \|\mathbf{y}_d(t) - \mathbf{y}_l(t)\|_\infty$. Below we will explain how to compute this upper bound $\bar{\varepsilon}$.

We will first find a small-scale[4] surrogate LPM M_r for the given discretized DPM M_d by using the CURE scheme such that their a priori $\mathcal{H}_2$ error $\bar{\varepsilon}_1$ (Eq.3.16a) is guaranteed and then we compute the *exact* $\mathcal{H}_2$ error $\bar{\varepsilon}_2$ (Eq.3.16b) between the surrogate LPM M_r and the given LPM M_l by solving a small-scale system of algebraic Lyapunov equations, which generally can be realized in a few seconds.

$$\|\mathbf{G}_d(s) - \mathbf{G}_r(s)\|_{H_2} \leqslant \bar{\varepsilon}_1 \tag{3.16a}$$

$$\|\mathbf{G}_r(s) - \mathbf{G}_l(s)\|_{H_2} = \bar{\varepsilon}_2 \tag{3.16b}$$

In Eq.3.16a and 3.16b, $\mathbf{G}_r(s)$ is the transfer function of the surrogate LPM M_l, whose state space form is shown in Eq.3.17.

$$\begin{aligned} \mathbf{E}_r\dot{\mathbf{x}}_r(t) &= \mathbf{A}_r\mathbf{x}_r(t) + \mathbf{B}_r\mathbf{u}_d(t) \\ \mathbf{y}_r(t) &= \mathbf{C}_r\mathbf{x}_r(t) + \mathbf{D}_r\mathbf{u}_d(t) \end{aligned} \tag{3.17}$$

With the obtained a priori errors $\bar{\varepsilon}_1$ and $\bar{\varepsilon}_2$, the a priori error $\bar{\varepsilon}$ of $\|\mathbf{G}_d(t) - \mathbf{G}_l(t)\|_{H_2}$ can be directly obtained by using the triangular inequality, as shown in Eq.3.18.

$$\begin{aligned} \|\mathbf{G}_d(s) - \mathbf{G}_l(s)\|_{H_2} &= \|\mathbf{G}_d(s) - \mathbf{G}_r(s) + \mathbf{G}_r(s) - \mathbf{G}_l(s)\|_{H_2} \\ &\leqslant \|\mathbf{G}_d(s) - \mathbf{G}_r(s)\|_{H_2} + \|\mathbf{G}_r(s) - \mathbf{G}_l(s)\|_{H_2} \\ &\leqslant \bar{\varepsilon}_1 + \bar{\varepsilon}_2 = \bar{\varepsilon} \end{aligned} \tag{3.18}$$

In practice, however, engineers are usually interested in restricting the relative error. The relative upper bound $\bar{\varepsilon}_{rel}$ for the difference between $\mathbf{G}_l(s)$ and $\mathbf{G}_d(s)$ with respect to $\mathcal{H}_2$ norm is as follows:

[4]small-scale means the number of variables is tens to hundreds

$$\begin{aligned}\frac{\|\mathbf{G}_d(s)-\mathbf{G}_l(s)\|_{\mathcal{H}_2}}{\|\mathbf{G}_l(s)\|_{\mathcal{H}_2}} &= \frac{\|\mathbf{G}_d(s)-\mathbf{G}_r(s)+\mathbf{G}_r(s)-\mathbf{G}_l(s)\|_{\mathcal{H}_2}}{\|\mathbf{G}_l(s)\|_{\mathcal{H}_2}} \\ &\leqslant \frac{\|\mathbf{G}_d(s)-\mathbf{G}_r(s)\|_{H_2}}{\|\mathbf{G}_l(s)\|_{\mathcal{H}_2}} + \frac{\|\mathbf{G}_r(s)-\mathbf{G}_l(s)\|_{\mathcal{H}_2}}{\|\mathbf{G}_l(s)\|_{\mathcal{H}_2}} \\ &\leqslant \frac{\bar{\varepsilon}_1+\bar{\varepsilon}_2}{\|\mathbf{G}_l(s)\|_{\mathcal{H}_2}} = \bar{\varepsilon}_{rel},\end{aligned} \tag{3.19}$$

where the transfer function $\mathbf{G}_l(s)$ of the given LPM M_l is selected as the reference because the DPM M_d is built based on the LPM M_l in the system-based geometric design. The computation of $\|\mathbf{G}_l(s)\|_{\mathcal{H}_2}$ is rapid because the scale of the LPM M_l is small.

Suppose we obtained the relative error bound $\bar{\varepsilon}_{rel}$, as long as it is less than a user-defined threshold, then the correspondence T_f at the physical quantity magnitude level between the discretized DPM M_d and the LPM is established and therefore the LPM M_l and DPM M_d are determined to have the weak consistency. Note that if users want to further know the upper bound for $\max_t \|\mathbf{y}_d(t)-\mathbf{y}_l(t)\|_\infty$, then it can be computed using Eq.3.15 and the relative error bound $\bar{\varepsilon}_{rel}$ as follows:

$$\max_t \|\mathbf{y}_d(t)-\mathbf{y}_l(t)\|_\infty \leqslant \bar{\varepsilon}\cdot\sqrt{\int_0^\infty \|\mathbf{u}_d(t)\|_2 dt} = \bar{\varepsilon}_{rel}\cdot\|\mathbf{G}_l(s)\|_{\mathcal{H}_2}\cdot\sqrt{\int_0^\infty \|\mathbf{u}_d(t)\|_2 dt} \tag{3.20}$$

The steps of the scheme are shown in Figure 3.10, where an LPM and a DPM of the suspension system are used as examples. The blue symbols are the given information, the red symbol is our aim and the rest is the intermediate results/steps. The scheme can be summarized into four steps as listed below.

Step (i) generate ODEs of the LPM M_l and PDEs of the DPM M_d by any available modeling languages and tools, then use spatial discretization schemes (e.g. FEM) to convert the PDEs of DPM M_d to a system of ODEs, and after that convert ODEs of the LPM M_l and the discretized DPM M_d to their state-space form by linear transformation.

Step (ii) reduce the order of the discretized DPM M_d by CURE scheme such that it a priori guarantees the $\mathcal{H}_2$ error $\bar{\varepsilon}_1$ between the discretized DPM M_d

and its surrogate LPM M_r.

Step (iii) solve Lyapunov equation to obtain the exact $\mathcal{H}_2$ error $\bar{\varepsilon}_2$ between the surrogate LPM M_r and the LPM M_l

Step (iv) compute the a priori $\mathcal{H}_2$ relatively error $\bar{\varepsilon}_{rel}$ between the DPM M_d and the LPM M_l using the triangular inequality and compare it with the user-defined threshold.

3.3 Discussion

In this Chapter, we formulated the problem of model consistency analysis between lumped and distributed parameter models as a process of checking if the given LPM and DPM have correct correspondences between their initial and boundary conditions and similar solutions after the field correspondence mapping. We introduced a framework for model consistency analysis to solve two major underlying issues: (1) the customized consistency analysis problem between models created in different physical modeling languages and tools (2) the establishment of the correspondences of model specifications and solutions.

The framework is proposed for developing software to realize automated system-based geometric design, provided a system-level design and a library of geometric components. Particularly, this automatic design procedure requires automatic model consistency at both component and assembly level, where engineers need to add appropriate ICs and BCs to the geometric parts candidates according to the given ICs and lumped sources in the given system-level design. How to automatically adding ICs and BCs is still an open but critical issue.

One potential solution could be that we first build a DPM library where geometric components are labeled with several common IC and BC types. Such a DPM library example is shown in Figure 3.11, where a composition of common BC and IC types is labeled to the geometric parts based on their functions. In addition, a library of Tonti diagrams needs to be built for automatically searching the common BC and IC types for different physical problems. Particularly in this library, the

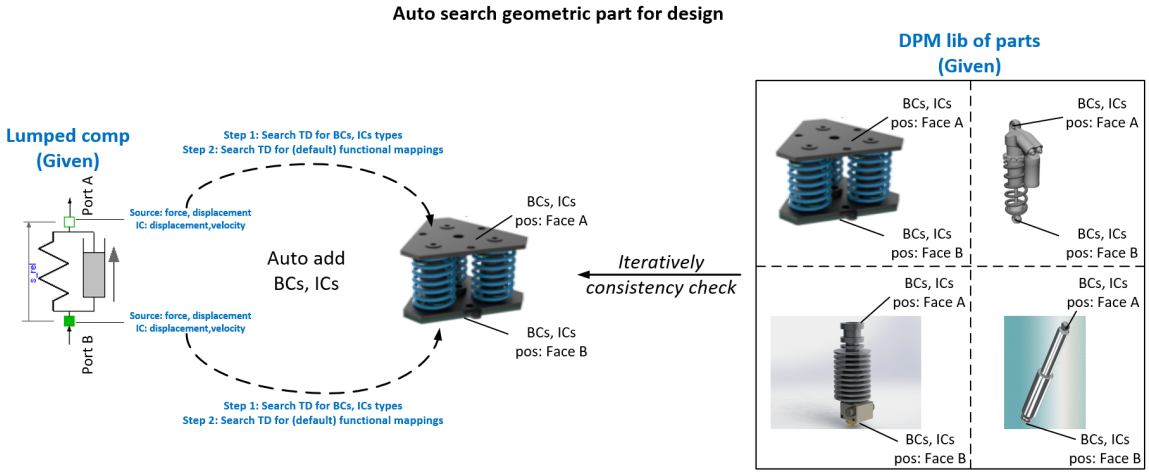


Figure 3.11: Automatic search geometric parts for design

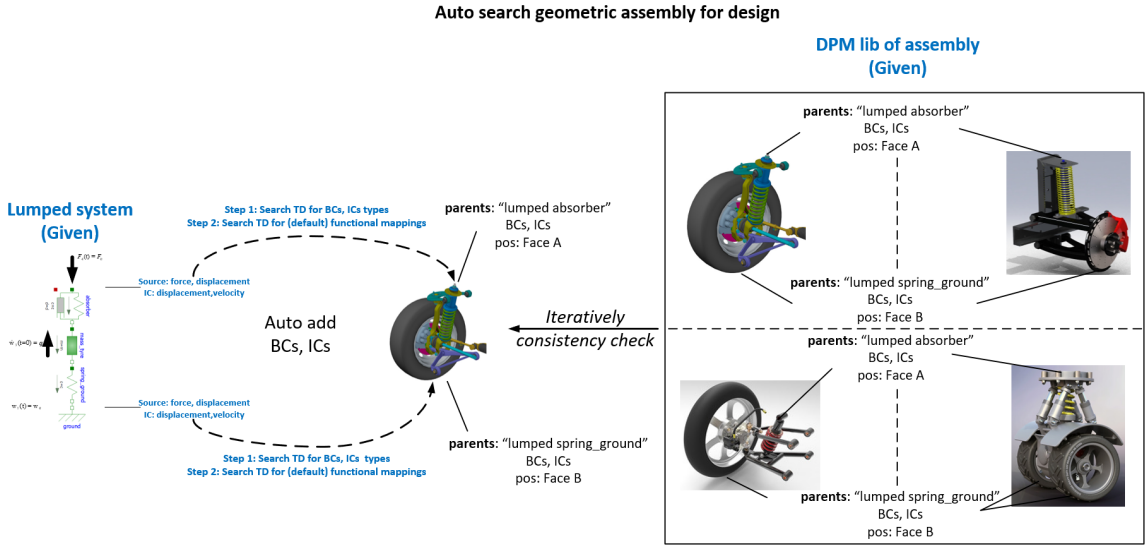


Figure 3.12: Automatic search geometric assemblies for design

default functional forms (e.g. linear integral) mapping between BCs and ICs of Tonti diagrams would be provided and users would have the authority to change thees functional form with respect to different application scenarios. With these two libraries and a given lumped component, the computer could automatically add ICs and BCs to a geometric part and iteratively conduct the model consistency

analysis for all the geometric parts candidates until finding a consistent one. Similarly, such automatic adding BCs and ICs to the geometric parts and iterative model consistency analysis procedure at the component level could be naturally extended to the system/assembly level as shown in Figure 3.12. Particularly, if the assembly is assembled from the selected parts based on the consistency analysis result, then the BCs and ICs added to the parts in the geometric assembly could be directly inherited from BCs and ICs added to geometric parts in the consistency analysis at the component level. Such inheritance will accelerate adding BCs and ICs to the geometric assembly without the need to searching the Tonti diagram library again.

4 UNIFIED SEMANTICS FOR LUMPED PARAMETER SYSTEM MODELING

This chapter introduces unified semantics for lumped parameter system modeling. The distinct semantics issue underlying different lumped parameter modeling languages and tools will be first summarized in Section 4.1, followed by an introduction of the algebraic topological model of lumped parameter systems in Section 4.2. After that, the extended and generalized Tonti diagrams of network theories and how these Tonti diagrams can be used to describe single and multiple domain lumped parameter models and generate governing equations will be illustrated in Section 4.3 ~ 4.4.

4.1 Distinct semantics issue of lumped parameter modeling languages and tools

Lumped parameter models are commonly used to describe behaviors of many engineering systems [70]. In such systems, spatially and temporally distributed physical phenomena are approximated by a finite network of abstract components that store, dissipate, or transform energy; the phenomena-specific constitutive properties of the components (e.g. generalized impedances) are estimated by domain integrals from the actual system by a process of "reticulation"[92]. Bond graphs [92], linear graphs [104], Modelica [38] and Simulink/Simscape [33] are commonly used physical modeling languages for creating, editing, and simulating lumped parameter models. These languages may differ widely in their syntax, but have similar (though not identical) semantics that specifies interconnectivity and constitutive relations of individual components in a system; the resulting model of a system created in such languages is then compiled into a system of state equations that may be numerically solved to simulate the system's dynamic response. Formally, the dynamic behavior of a lumped parameter model is described by the state equations,

a set of ordinary differential equations (ODEs) or differential algebraic equations (DAEs), whose solutions depend on the initial conditions.

Generally speaking, all modeling languages can handle the same broad class of problems but with non-trivial differences in system types, representations, state equation derivation and simulation mechanisms [68, 104, 49, 115]. For example, in Simulink, components exchange numeric information uni-directionally and are not subject to conservation laws; by contrast, the energy flow between components is bi-directional in other languages, satisfying conservation laws. While every linear graph model may be represented by a bond graph, the converse statement is not true [95]. Furthermore, parallel and serial junctions in a bond graph do not specify ordering of branches in the junctions, which implies that every bond graph in fact corresponds to a family of (dynamically) equivalent graph-based models in other languages. Many languages generate state equations using efforts and flows as variables, but their integral forms may also be used, for example in bond graphs [69]. The above and other differences in syntax and semantics of system modeling languages lead to challenges in construction of a unified model consistency determination tool. Such interoperability difficulties are only likely to increase due to ubiquitous and growing adoption of physical modeling languages by industry and standards organizations.

Conceptually, there are two possible approaches to dealing with semantic interoperability issues: "point-to-point" correspondence[1] between models created in different languages, or standardization on a single neutral format that can be translated to/from models in any such languages. The first approach is more practical but is problematic because it requires $O(n^2)$ such translators, which is not only expensive, but discourages development of new languages and simulation solutions. The second approach is similar in spirit to STEP for product models, which requires the neutral format to be formally defined and include the superset of models present in any such language.

Irrespectively of the selected approach, semantic interoperability requires establishing formal correspondence between concepts and constructs in distinct

[1]Such correspondence may take a form of direct translation or using APIs.

modeling languages. This is the main goal of this chapter. Our approach relies on tools from algebraic topology and well known classification of physical theories developed over the years by Tonti [111, 112], Roth [102], Branin [26], Kron [72], and others. Importantly, this classification generalizes to higher-dimensional physical models, suggesting that the proposed framework could be extended to include spatially-distributed models represented by three-dimensional solid models, partial differential equations and numerical methods for solving partial differential equations.

4.2 Algebraic topological models of lumped parameter systems

Algebraic topological interpretations of LPMs and various types of electrical circuits are well known in literature [26, 112] and are now a standard textbook material [11]. We now apply such an interpretation to LPMs and show that it provides a neutral standard semantics for all such systems. Superficially, all algebraic topological formulations are identical, but important semantic difference emerge in details. As a starting point, we adopt Tonti's classification [112] of physical theories in terms of their algebraic topological models; however, specific requirements of the lumped parameter system require significant extensions and modifications that we discuss below. Strictly speaking, a proper setting for physical modeling is 4D spacetime. But, most engineering models are set in space$\times$time, where space and time models are treated separately.

4.2.1 LPMs as cochains

In algebraic topological view of physics, physical properties are distributed in spacetime over finite chunks of space called p-dimensional cells, or p-cells, ($p = 0, 1, 2...$) that fit together to form a cell complex that decomposes the undelying physical space. Many choices of cells are possible; they can be open or closed, p-simplicies, p-balls, or p-manifolds; specific choices are dictated by convenience and

applications and define the type of cell complex [61]. All cells are endowed with orientation, or sense of direction, which becomes important in order to properly assign signs to physical properties associated with cells.

As we already saw, all LPMs are formulated using 2-dimensional cells complexes: 0-cells (nodes), 1-cells (edges), and 2-cells (cycles or "meshes"). These complexes are abstract in the sense that geometric coordinates or shapes of the cells are immaterial; only their connectivity carries important physical information.[2] The distribution of physical properties is described by assigning their types and quantities to the individual cells in this complex. The formal mechanism for doing so requires discretizing the property g over p-cells e^p_α as a p-*cochain* C^p, a formal sum

$$C^p = \sum_{\alpha=1}^{n_p} g_\alpha e^p_\alpha \tag{4.1}$$

For example, Figure 4.1 shows a 2-cell complex composed of 0-cells (P),1-cells (L) and 2-cells (M) and the associated coefficients (real numbers) with 0-cells. The coefficients typically represent discrete distributions of physical properties such as temperature, electric potential, tip displacement, etc. The 0-cochain corresponding to such a distribution is

$$C^0 = -2.1P_1 + 1.6P_2 + 0.8P_3 + 0P_4 + 3.0P_5$$

The relation between physical properties is governed by two types of fundamental laws: metric laws and topological laws. Metric laws usually involve measurement while topological laws relate physical properties associated with space and its boundary. Topological laws can be formulated using formal linear coboundary δ operations on cochains. Specifically, coboundary δ_p operating on a p-cochain produces a $(p+1)$-cochain by transferring and adding the coefficient of the p-cochain

[2]This is in stark contrast to spatially distributed physical phenomena modeled governed by partial differential equation where geometry of cells becomes critical.

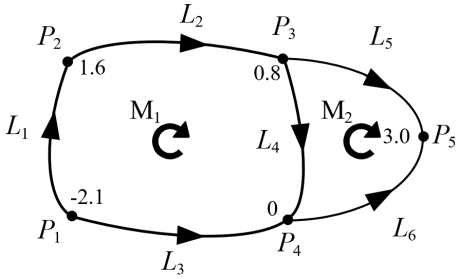


Figure 4.1: A 2-cell complex and coefficients associated with all 0-cells

to its cofaces (Eq.4.2). Formally,

$$\delta_p(C^p) = \delta_p\left(\sum_{\alpha=1}^{n_p} g_\alpha e_\alpha^p\right) = \sum_{\beta=1}^{n_{p+1}}\left(\sum_{\alpha=1}^{n_p} h_{\alpha\beta}\cdot g_\alpha\right)\cdot e_\beta^{p+1}, \tag{4.2}$$

where n_p represents the number of cells in the p-cochain. The incidence coefficient $h_{\alpha\beta} = [e_\alpha^p, e_\beta^{p+1}] \in \{0, \pm 1\}$ is determined by relative orientation of p-cell e_α^p and it cofaces $(p+1)$-cell e_β^{p+1} [112]. If e_β^{p+1} is not a coface of e_α^p, then $h_{\alpha\beta} = 0$; otherwise, if the orientations of e_α^p and e_β^{p+1} are consistent, then $h_{\alpha\beta} = +1$, otherwise, $h_{\alpha\beta} = -1$. If we denote the usual p-incidence matrix as $\mathbf{A} = [h_{\beta\alpha}]$, then the coboundary operator δ_p is commonly represented by its transpose $\mathbf{A}^T$.

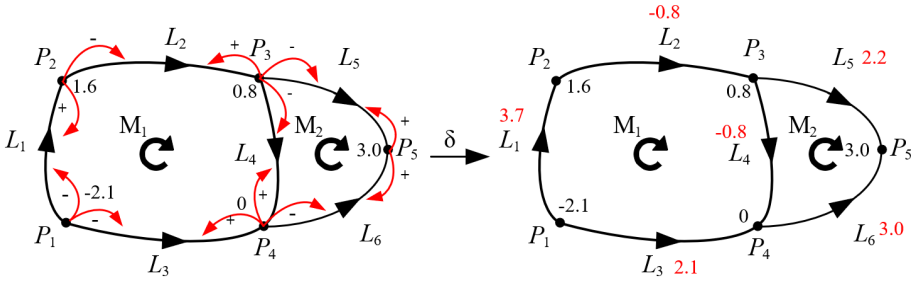


Figure 4.2: Example of the coboundary operation over a 0-cochain producing a 1-cochain

For example, suppose the coefficients associated with the 0-cells in Figure 4.2 are electric potentials, then the coboundary operation δ_0 over 0-cochain electric

potentials C^0 would produce a 1-cochain voltage drops

$$\begin{aligned} C^1 &= \delta_0(C^0) \\ &= (-(-2.1) + 1.6)L_1 + (-1.6 + 0.8)L_2 + (-(-2.1) + 0)L_3 \\ &\quad +(-0.8 + 0)L_4 + (-0.8 + 3.0)L_5 + (-0 + 3.0)L_6 \\ &= 3.7L_1 + (-0.8)L_2 + 2.1L_3 + (-0.8)L_4 + 2.2L_5 + 3.0L_6 \ , \end{aligned}$$

where the coefficient associated with 1-cell L_i is obtained by adding the two coefficients that are respectively associated with its two cofaces (0-cells), with taking into account the orientation consistency between L_i and its cofaces.

Informally, the coboundary operations capture the essence of balance, equilibrium, conservation, compatibility, and other topological laws. For cochains on finite cell complexes, coboundary operators $\delta_p, p = 1, 2, 3$ correspond to the usual vector calculus operators of gradient, curl, and divergence respectively. The vector calculus identities $\nabla \times (\nabla\phi) = 0$, and $\nabla \cdot (\nabla \times \mathbf{F}) = 0$ are simply instances of the Poincare lemma stating that $\delta_p(\delta_{p-1}()) = \mathbf{0}$, where $\mathbf{0}$ denotes a null cochain. A collection of cochains and coboundary operators on a cell complex form a cochain complex[61].

4.2.2 Physical theories as Tonti diagrams

Every physical theory is conceptualized in terms of relationships between two types of dual physical quantities that are referred to by various authors as configuration /source [112], through/across [26], or effort/flow [68]. In what follows we will adopt Tonti's convention and distinguish between configuration type variables, that are modeled as cochains on primary cell complex decomposition of space, and source variables that are modeled as cochains on the dual cell complex decomposition of the same space. This notion is illustrated in Figure 4.8, where the primal cell complex is shown in blue the dual cell complex is shown in black. (The actual geometry of cells is irrelevant for this discussion.) Each p-dimensional cell in the primal cell complex is dual to a unique $(n - p)$-dimensional cell in the dual cell complex. By duality, it also follows that the coboundary operator $\widetilde{\delta}_p$ on the dual

cell complex can be represented by the transpose of $\delta_p = \mathbf{A}^T$, or simply $\mathbf{A}$.

This conceptualization of physical quantities in terms cochains on dual cell complexes is not arbitrary: it arises from first principles based on how the postulated quantities are measured. In each case, the measurement process implies the intrinsic dimension of the associated quantity (e.g., displacements are measured at a point, currents are measured across the surface, voltage drop is measured along a path, and so on). The decision whether a particular quantity belongs to the primal or dual complex is determined by the oddness principle that requires change of sign under change of orientation of the relevant cell. The primal cells are are endowed with inner orientation, while the dual cells are oriented relative to the containing (outer) space. The reader is referred to [112] for detailed discussion of these concepts.

Figure 4.3 shows the correspondence between the primal and dual cochains of variables and naming conventions used by different authors and lumped-parameter modeling systems. We note that the adopted classification in terms of primal and dual cochains is consistent with the conventions in linear graphs [104], Modelica [38], Simscape [33] and NIST models [24], but differs slightly from that in bond graphs [70].

Physical laws (topological and metric) relate different types of variables within each physical theory. Tonti proposed a systematic method for representing these laws using a diagram that can be considered an evolved combination of the so-called Roth diagrams [103] in terms of cochain sequences and "Maxwell house" diagram to represent all topological and metric relationships in electromagnetism [25]. A simple example of such a diagram is the Tonti diagram of electrical (static) network theory is shown in (Figure 4.4). It describes the network systems that satisfy Kirchhoff Current Law (KCL) and Kirchhoff Voltage Law (KVL) using a pair of cochains complexes dual to each other [112].

The diagram consists of two vertical sequences corresponding to the primal (left) and dual (right) cochain complexes, ordered by dimension. The vertical arrows correspond to the coboundary operations, going down for primal cochains and going up for dual cochains. Formally, the two sequences are exact[3] and form

[3]The cochain sequence is exact if it satisfies $\delta_p \circ \delta_{p-1} = 0$ $(p \geqslant 1)$

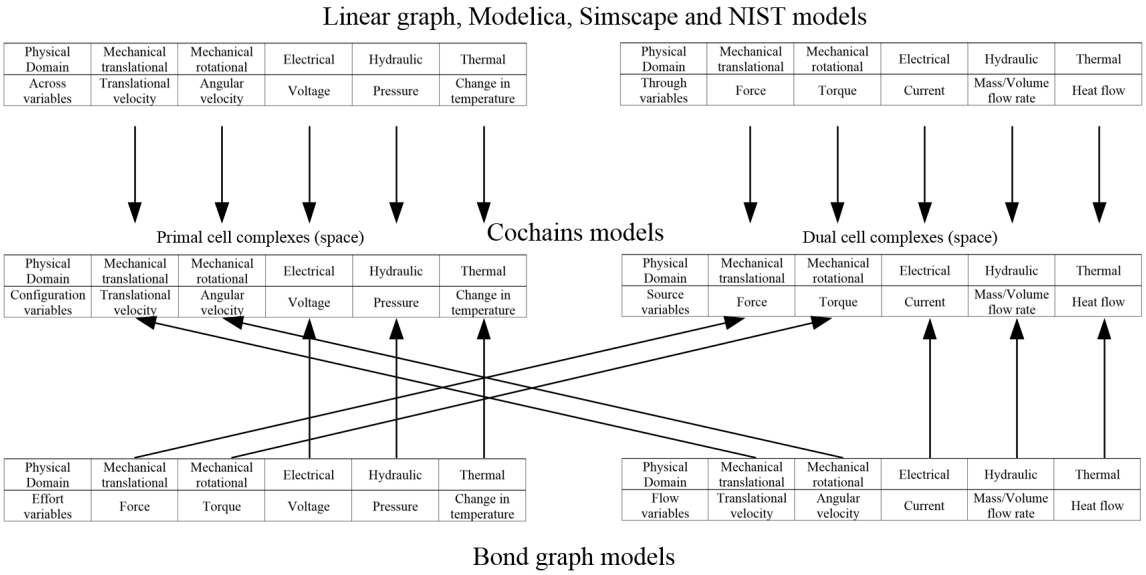


Figure 4.3: Categorization of physical variables category in different approaches to physical modeling

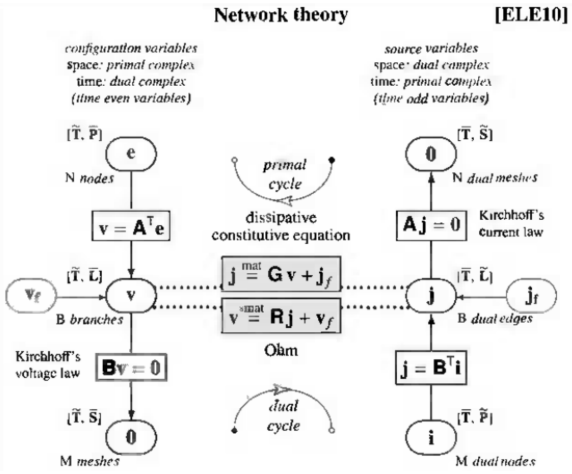


Figure 4.4: Tonti Diagram of network theory - constitutive equations are modified to account for voltage and current sources

two dual cochain complexes:

$$\text{primal}: \ \mathbf{e}^0 \xrightarrow{\delta_0} \mathbf{v}^1 \xrightarrow{\delta_1} \mathbf{0}^2 \tag{4.3}$$

$$\text{dual}: \ \mathbf{0}^2 \xleftarrow{\widetilde{\delta}_1} \mathbf{j}^1 \xleftarrow{\widetilde{\delta}_0} \mathbf{i}^0 \tag{4.4}$$

The measured relationships between dual quantities are represented by the horizontal links in the diagram. In the case of network theory, as shown in Figure 4.4, the primal variable are node potentials **e** associated with 0-cells, voltage drops **v** and sources associated with 1-cells, and voltage drops associated with 2-cells (meshes or cycles) that are identically **0** as the consequence of KVL. The cochains of adjacent dimensions satisfy topological laws expressed by the corresponding coboundary operations depicted as down-facing vertical arrows. Thus, 1-cochain of voltage drops $\mathbf{v}^1 = \mathbf{A}^\top \mathbf{e}^0$ is implied by the coboundary operation δ_0 on 0-cochain of node potentials $\mathbf{e}^0$; and KVL is just a restatement of the Poicare lemma. Similarly, the dual source (current) variables: 0-cochain **i**, 1-cochains **j**, and 2-cochain **0** are indicated in the right branch of the diagram, related by the sequence of two coboundary operations, indicated as arrows going up and expressing KCL.

The constitutive relation between 1-cochain of voltage drops **v** and 1-cochain of currents **j** satisfies Ohm's Law (or its inverse). The diagram also reveals two (dual) methods of generating the governing state equations for network models, depending on the choice of state variables. The two methods are indicated by primal and dual 'cycles' which refer to two different ways of composing topological and metric laws. For instance, in the primal cycle, 1-cochain voltage drop **v** is obtained by coboundary operation on 0-cochain node potential **e**. Using the Ohm's law, **v** is converted to 1-cochain branch currents **j** in the dual cell complex, where coboundary operation **A** on **j** equals zero. The physical meaning of the latter coboundary operation **A** is that the algebraic sum of branch currents of a dual loop equals zero, as required by KCL. Similarly, the dual cycle relies on KVL to generate the dual state equations for the same system.

Even though the diagram in Figure 4.4 describes a static phenomena, the reader will notice that the configuration and source variables are associated with time instances: primal instances for source variables and dual instances for configuration variables. This distinction becomes critical in dynamic physical models where, once again, the primary time elements (0-dimensional instances and 1-dimensional

intervals) are distinguished from the dual time elements based on the oddness principle that requires sign change under reversal of motion [112]. Strictly speaking, a proper setting for all physical models is a 4D spacetime, which we chose to represent as direct product of space and time. In other words, for each type of spatial variable, we can also consider its behavior in time which is represented by a pair of dual 1-dimensional time complexes, as shown in Figure 4.5. Here 0-cells $\overline{\mathrm{I}}_*$ and 1-cells $\overline{\mathrm{T}}_*$ represent primal time instances and intervals, while 0-cells $\widetilde{\mathrm{I}}_*$ and 1-cells $\widetilde{\mathrm{T}}_*$ represent the dual time instances and intervals, respectively.

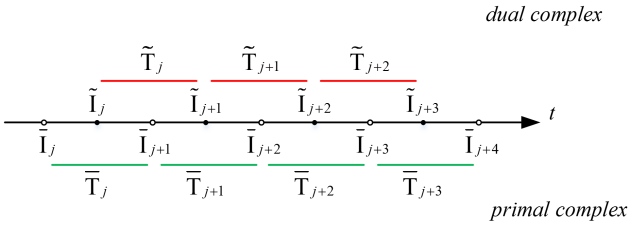


Figure 4.5: Cell complexes model of time

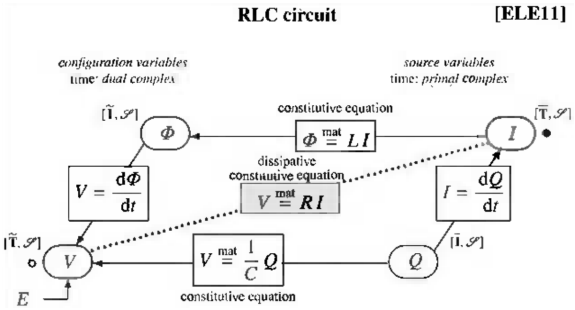
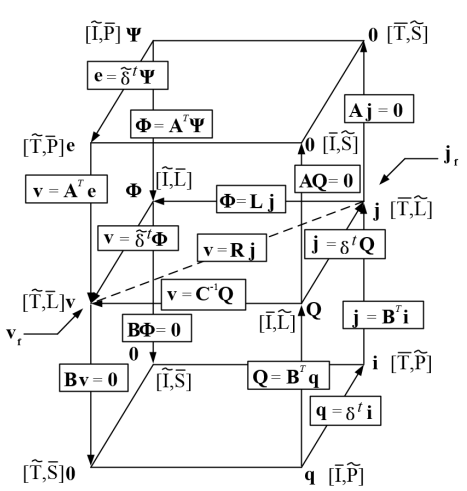


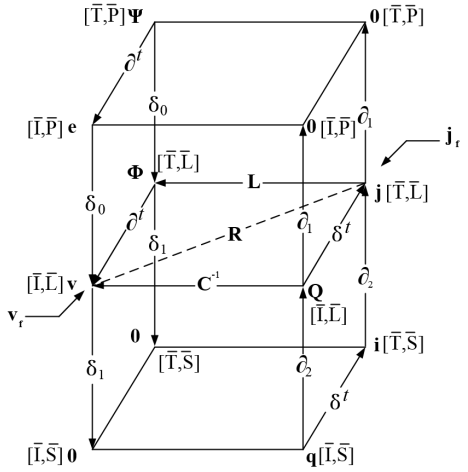
Figure 4.6: Tonti diagram of RLC circuit - only voltage sources are included

The introduction of time cell complexes in time has two consequences. First, it identifies the usual time derivative with the corresponding coboundary operator δ^t. Thus, if $\mathbf{q}$ is a 0-cochain of primal (or dual) time instances, then $\overline{\mathrm{I}}$ (respectively, $\widetilde{\mathrm{I}}$) is the 1-cochain of time intervals of $\mathbf{q}$ defined by $\delta_0^t\mathbf{q}$ (respectively $\widetilde{\delta}_0^t\mathbf{q}$). On a finite cell complex, δ_0^t become the time difference operation. Secondly, all space Tonti diagrams now acquire an additional time dimension, giving rise to horizontal

sections of the diagram. One such section is shown in Figure 4.6, which corresponds to the Tonti diagram of RLC circuit systems [112]. Here, the usual differential relations $V = d\Phi/dt$ and $I = dQ/dt$ are consequence of topological relations $\mathbf{V} = \tilde{\delta}_0^t\mathbf{\Phi}$ and $\mathbf{I} = \delta_0^t\mathbf{Q}$, respectively, where Φ is magnetic flux and Q is electric charge. Two new constitutive equations describe the capacitance relation between electric charge Q and voltage drop v and the inductance relation between magnetic flow Φ and currents j. The second time derivative is the result of composition of two first time derivatives. For instance, the second order equation differential equation $V = d\Phi/dt = Ld(I)/dt = Ld^2Q/dt^2$ in Figure 4.6 can be expressed as $\mathbf{V} = \tilde{\delta}_0^t\mathbf{\Phi} = \mathbf{L}\tilde{\delta}_0^t\mathbf{I} = \tilde{\delta}_0^t\mathbf{L}\delta_0^t\mathbf{Q}$ [112, 44]. Finally, we note that the vertical space diagram in Figure 4.4 and the horizontal time diagram in Figure 4.6 can be combined into a single three-dimensional diagram, as described in Section 4.3 and shown in Figure 4.7a.



(a) Matrix operators on dual cell complexes



(b) Topological and constitutive relationships on a single cell complex

Figure 4.7: Extended Tonti diagram for RLC network system - with voltage and current sources

In addition to physical quantities in Figure 4.4 and 4.6, the diagram in Figure 4.7a includes magnetic flux potential Ψ and mesh electric charge q that are related by coboundary operators to magnetic flux Φ and electric charge Q respectively. In

other words, the extended Tonti diagram includes two additional cochain complexes defined by the two sequences:

$$\text{primal}: \ \mathbf{\Psi}^0 \xrightarrow{\delta_0} \mathbf{\Phi}^1 \xrightarrow{\delta_1} \mathbf{0}^2 \tag{4.5}$$

$$\text{dual}: \ \mathbf{0}^2 \xleftarrow{\widetilde{\delta}_1} \mathbf{Q}^1 \xleftarrow{\widetilde{\delta}_0} \mathbf{q}^0 \tag{4.6}$$

Additional horizontal arrows indicate the corresponding time coboundary (derivative) operations. With the added constitutive, topological equations and physical variables, the extended Tonti diagram includes all possible quantities, as well as constitutive and topological laws of RLC electrical circuit systems. In the following section, we explicitly distinguish between the constitutive and topological laws as they are commonly used in practice in a single cell complex network model that is shown in Figure 4.7b.

4.2.3 Dual cochain complexes on a single cell complex

The ultimate goal of physical modeling is numerical simulation which is usually performed on a discretization of spacetime. If one were to accept that every physical theory is formulated in terms of dual cochain complexes, it would be reasonable to expect that most modeling and simulation tools are also formulated in terms of dual discretizations (one for configuration variables and the other one for the source variable) as illustrated in Figure 4.8 (on the left). In fact, such dual discretizations are often advocated in literature as more natural and numerically stable alternatives, for example, in mimetic discretization schemes [31], cell methods [112, 44], discrete exterior calculus [64] and other modeling approaches. However, the vast majority of numerical schemes appear to be based on a single discretization of space, which supports evaluation of both primal and dual cochains. For example, in most finite element, finite difference, and finite volume methods, all configuration and source variables are associated with cells (often nodes) in the underlying mesh, and their duality is hidden within the numerical scheme itself [79].

In LPMs, dual discretizations are particularly counter-intuitive, since all spatially

distributed properties have already been integrated (lumped) and only connectivity of the underlying cell complex remains visible. That connectivity often directly corresponds to the physical embedding; for example, a single electrical network carries both voltage and current information. Similarly, general network model is a single cell complex where primal and dual cochains are represented. The mapping of dual cochains on the primal cell complex is straightforward and is accomplished by mapping the dual $(n-p)$-cells to their corresponding primal p-cells, as shown in Figure 4.8. Consider how this mapping would work for the cochains in Figure 4.4. With this mapping, node potentials **e**, voltage drops **v** and sources, and voltage drops of cycles **0** are still associated with primal 0-cells (e.g. A), 1-cells (e.g. L), and 2-cells (e.g. M) respectively. However, the dual source (current) variables: mesh currents **i**, branch currents **j**, and currents merging at nodes **0** which were originally associated with dual 0-cells (e.g. a), 1-cells (e.g. l), and 2-cells (e.g. m) are now associated with primal 2-cells (e.g. M), 1-cells (e.g. L), and 0-cells (e.g. A) respectively.

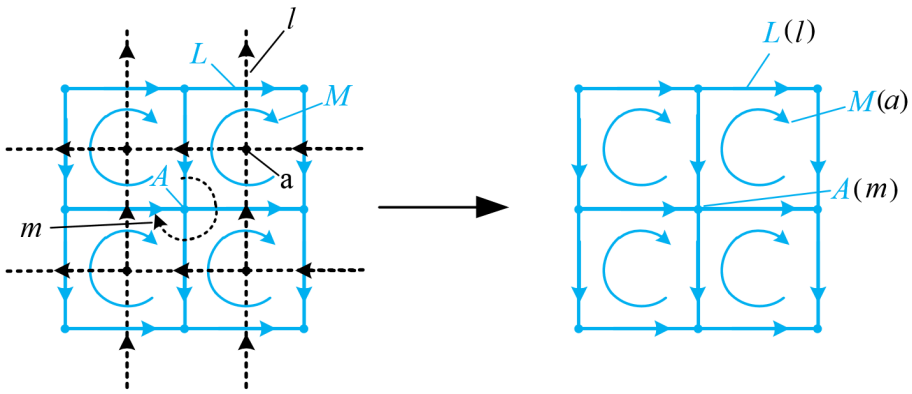


Figure 4.8: Dual cochain complexes on a single cell complex

With this mapping, all $(n-p)$-coboundary operations in dual cell complexes would become the p-boundary operations ∂ in the primal cochain complex, which

operate on a p-cochains and produce a $(p-1)$-cochains:

$$\partial_p(C^p) = \partial_p(\sum_{\alpha=1}^{n_p} g_\alpha e_\alpha^p) = \sum_{\beta=1}^{n_{p-1}} \left(\sum_{\alpha=1}^{n_p} h_{\alpha\beta} \cdot g_\alpha \right) \cdot e_\beta^{p-1}, \tag{4.7}$$

which is similar to Eq.4.2, except that the coefficients are transferred from p-cells to their $(p-1)$-faces. This implies that the coboundary operation $\widetilde{\delta}_{n-p}$ on the dual cells complex is mapped into the boundary operation ∂_p on the primal cell complex,[4] i.e.,

$$\partial_p = \widetilde{\delta}_{n-p} \tag{4.8}$$

This explains, for example, why the KCL on the primal complex is described by condition $\partial_1 \mathbf{j}^1 = \mathbf{0}$, stating that the branch currents must add up to zero at every node.

The single complex network model is summarized by a new type of Tonti diagram shown in Figure 4.7b. Here the dual (source) cochains have been mapped to the corresponding cochains on the primal cell complex to form the dual cochain complex with boundary ∂_p operators replacing the original dual $\widetilde{\delta}_{n-p}$ coboundary operators. In other words, the dual cochain sequences (4.4) and (4.6) become respectively

$$\mathbf{i}^2 \xrightarrow{\partial_2} \mathbf{j}^1 \xrightarrow{\partial_1} \mathbf{0}^0 \tag{4.9}$$

$$\mathbf{q}^2 \xrightarrow{\partial_2} \mathbf{Q}^1 \xrightarrow{\partial_1} \mathbf{0}^0 \tag{4.10}$$

In the following sections, we will assume that an algebraic topological model of a lumped parameter system is described by such single (primal) cell complex and a corresponding Tonti diagram with four cochain complexes (correspong to the four vertical 'legs' of the diagram) and all relationships between them.[5] The two primal cochain complex include: (1) 0-cochain node potentials $\mathbf{e}^0$, 1-cochain voltage drops

[4] This observation also justifies use of ∂_p operator, usually reserved for p-chains, on p-cochains

[5] Strictly speaking, use ∂_p operators give rise to chain complexes, but it should be clear that these chain complexes on the primal cell complex are isomorphic to the cochain complexes over the dual cell complex.

$\mathbf{v}^1$, 2-cochain of mesh voltages (which is $\mathbf{0}$ by KVL); and (2) 0-cochain magnetic flux potentials $\mathbf{\Psi}^0$, 1-cochain magnetic fluxes $\mathbf{\Phi}^1$, and 2-cochain of mesh magnetic fluxes and 2-cochain of mesh voltages (which is $\mathbf{0}$ by KVL). The cochains in each complex are related by the spatial coboundary operators, while the corresponding p-cochains in the two complexes are related by the time boundary operator. Similarly, the two dual cochain complexes are: (1) 0-cochain of node currents (which is $\mathbf{0}$ by KCL), 1-cochain currents $\mathbf{j}^1$, 2-cochain mesh currents $\mathbf{i}^2$; and (2) 0-cochain of node electric charges, 1-cochain electric charges $\mathbf{Q}^1$, 2-cochain mesh electric charges $\mathbf{q}^2$. The cochains in each of the dual complexes related by the spatial boundary operators, while the corresponding p-chains in the two dual complexes are related by the time coboundary operators.

4.3 Single-domain lumped parameter systems

In this section, we will show how to use Tonti diagrams to describe lumped parameter systems and introduce an automated method of generating system state equations. We will focus on LPMs of a single physical domain, exemplified by classical RLC electrical circuit systems. Application to other physical domains is immediate, since all such models are isomorphic.

4.3.1 Static systems

Classical single-domain RLC electrical circuits consist of five types of physical elements: resistors, capacitors, inductors, voltage sources and current sources. We will first consider a special case of static resistive circuits that contain only constant resistors, as well as voltage and current sources; later we will extend the approach to general dynamic electrical circuits. The algebraic topological structure of static electrical circuits relies only on the two dual cochain complexes modeled over a single cell complex. The primal cochain complex includes 0-cochain of node potentials $\mathbf{e}^0$, 1-cochain of voltage drops $\mathbf{v}^1$, and 2-cochain of mesh voltages (which is $\mathbf{0}$ by KVL), related by the coboboundry operators; the dual cochain complex

consists of 0-cochain of node currents (which is **0** by KCL), 1-cochain of branch currents $\mathbf{j}^1$, and 2-cochain mesh currents $\mathbf{i}^2$ related by the boundary operators. The topological and constitutive relations between these cochains are given by the diagram in Figure 4.4, and two methods of generating the equations are indicated by primal and dual 'cycles' respectively in the diagram. In the context of the more general model, these cycles correspond to red and blue paths in the extended Tonti diagram as shown in (Figure 4.9). Each path is defined by a sequence of the arrows in the diagram indicating composition of the corresponding physical laws.

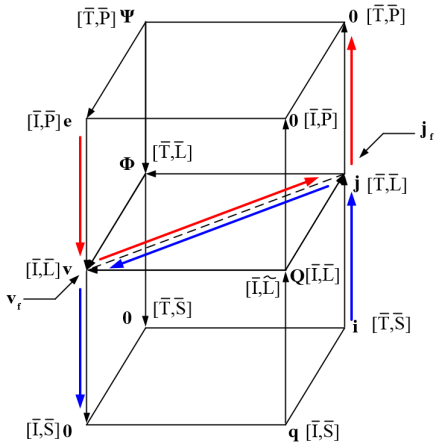


Figure 4.9: Paths corresponding to primal and dual cycles in Figure 4.4

For example, if we use the red path in Figure 4.9, then the 0-cochain $\mathbf{e}^0$ is selected as the state variable. The system state equation can be generated by composition of three steps starting with a 0-cochain $\mathbf{e}^0$ of node potentials. First, the downward red arrow indicates that the node potentials give rise to voltage drops associated with incident branches using the coboundary operator $\mathbf{v}^1 = \delta_0 \mathbf{e}^0$. The second lateral red arrow correspond to the constitutive relation $\mathbf{j}^1 = \mathbf{G}\left(\mathbf{v}^1 + \mathbf{v}_f^1\right) + \mathbf{j}_f^1$, which accounts for the contribution of 1-cochains of voltage sources $\mathbf{v}_f^1$ and source currents $\mathbf{j}_f^1$ in each branch of the electrical network. Finally, the upward vertical red arrow indicates the application of KCL $\partial_1 \mathbf{j}^1 = \mathbf{0}$ requiring that branch currents add up to zero. The composition of the three laws yields:

$$\partial_1 \left(\mathbf{G} \left(\delta_0 \mathbf{e}^0 + \mathbf{v}_f^1 \right) + \mathbf{j}_f^1 \right) = \mathbf{0} \tag{4.11}$$

Eq.4.11 is usually written in a more traditional form as

$$\partial_1 \left(\mathbf{G} \left(-\delta_0 \mathbf{e}^0 - \mathbf{v}_f^1 \right) + \mathbf{j}_f^1 \right) = \mathbf{0} \tag{4.12}$$

As explained in [112], the minus sign in front of $\delta_0 \mathbf{e}^0$ is due to the largely historical assumption that the vertices (0-cells) of the primal cell complex are oriented as sinks. The second minus sign in front of $\mathbf{v}_f^1$ signifies the fact that a (positive) voltage source should be subtracted from the (positive) voltage drop in every branch.

An alternative method for generating the system state equations follows the dual blue path in Figure 4.9. The process starts with the dual 2-cochain of mesh currents $\mathbf{i}^2$ selected as the state variable and amounts to composition of the analogous three steps: boundary operator ∂_2 applied to mesh currents in order to generate branch currents, constitutive Ohm's law $\mathbf{R}$ that relates the branch currents to voltage drops, and coboundary operator δ_1 applied to the voltage drops in accordance with KVL. Taking into account the voltage and current sources, the process results in:

$$\delta_1 \left(\mathbf{R} (\partial_2 \mathbf{i}^2 - \mathbf{j}_f^1 \right) + \mathbf{v}_f^1) = \mathbf{0} \tag{4.13}$$

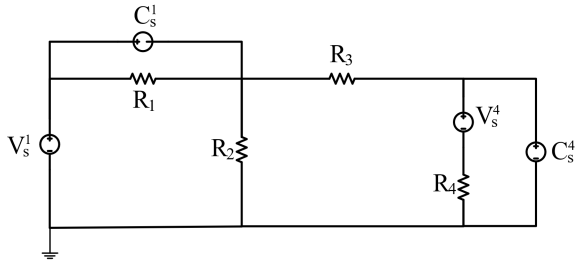
Collecting the terms with known voltage and current sources and moving them to the right hand side, the system state equations Eq.4.12 and Eq.4.13 transform to Eq.4.14 and Eq.4.15, respectively.

$$\partial_1 \mathbf{G} \delta_0 \mathbf{e}^0 = \partial_1 \left(\mathbf{j}_f^1 - \mathbf{G} \mathbf{v}_f^1 \right) \tag{4.14}$$

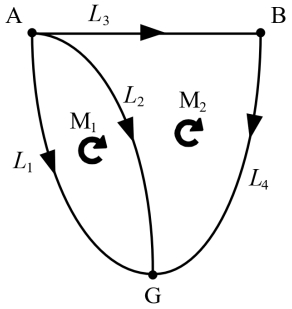
$$\delta_1 \mathbf{R} \partial_2 \mathbf{i}^2 = \delta_1 \left(-\mathbf{v}_f^1 + \mathbf{R} \mathbf{j}_f^1 \right) \tag{4.15}$$

When boundary and cobounadry operators are replaced by the corresponding incidence matrices describing a cell complex underlying a specific lumped parameter system, the state equations become systems of linear equations that can be

solved for the unknown state variables (node potentials $\mathbf{e^0}$ and mesh currents $\mathbf{i^2}$ respectively).



(a) An electrical circuit of pure resistance



(b) Topological structure

Figure 4.10: An electrical circuit of pure resistance and its topological structure

Example 1. *We will use an example of static electrical circuit in Figure 4.10a to illustrate the derivation of Eq.4.14 and Eq.4.15 in concrete setting. The shown electrical network contains four constant resistors* $R_1 = 6\Omega$, $R_2 = 3\Omega$, $R_3 = 2\Omega$, $R_4 = 4\Omega$, *two constant current sources* $C_s^1 = 3A$, $C_s^4 = 1A$ *and two constant voltage sources* $V_s^1 = 8V$, $V_s^4 = 4V$. *Topologically, the network is a 2-dimensional complex shown in Figure 4.10b and consisting of three 0-cells* (*A,B,G*), *four 1-cells* (L_1,L_2,L_3,L_4) *and two 2-cells* (M_1,M_2).

The algebraic topological model of the electrical circuit contains: primal 0-cochain node potentials ($\mathbf{e}^0 = e_A \cdot A + e_B \cdot B + e_G \cdot G$), *primal 1-cochain voltage drops* ($\mathbf{v}^1 = V_1 \cdot L_1 + V_2 \cdot L_2 + V_3 \cdot L_3 + V_4 \cdot L_4$), *dual 1-cochain of branch currents* ($\mathbf{j}^1 = j_1 \cdot L_1 + j_2 \cdot L_2 + j_3 \cdot L_3 + j_4 \cdot L_4$), *dual 2-cochain of mesh currents* ($\mathbf{i}^2 = i_1 \cdot M_1 + i_2 \cdot M_2$) *and two cochains that are always* **0**: *the dual 0-cochain of node currents and the primal 2-cochain of loop voltage drops. In order to get unique solution of the state equations, we consider 0-cell G as the reference node, implying the boundary condition of* $e_G = 0$. *Following the red path on the Tonti diagram generates equation Eq.4.14 with individual terms as follows:*

$$\partial_1 = \begin{bmatrix} -1 & -1 & -1 & 0 \\ 0 & 0 & +1 & -1 \\ +1 & +1 & 0 & +1 \end{bmatrix} \tag{4.16}$$

$$\delta_0 = \partial_1^T \tag{4.17}$$

$$\mathbf{G} = \begin{bmatrix} R_1^{-1} & 0 & 0 & 0 \\ 0 & R_2^{-1} & 0 & 0 \\ 0 & 0 & R_3^{-1} & 0 \\ 0 & 0 & 0 & R_4^{-1} \end{bmatrix} = \begin{bmatrix} 1/6 & 0 & 0 & 0 \\ 0 & 1/3 & 0 & 0 \\ 0 & 0 & 1/2 & 0 \\ 0 & 0 & 0 & 1/4 \end{bmatrix} \tag{4.18}$$

$$\mathbf{j}_f^1 = \begin{bmatrix} C_s^1 & 0 & 0 & -C_s^4 \end{bmatrix}^T = \begin{bmatrix} 3 & 0 & 0 & -1 \end{bmatrix}^T \tag{4.19}$$

$$\mathbf{v}_f^1 = \begin{bmatrix} V_s^1 & 0 & 0 & V_s^4 \end{bmatrix}^T = \begin{bmatrix} 8 & 0 & 0 & 4 \end{bmatrix}^T \tag{4.20}$$

Substituting Eq.4.16 ~ Eq.4.20 into Eq.4.14 we obtain a linear system of equations that has the solution of $\mathbf{e}^0 = [-1/2,\ 7/3,\ 0]^T$. *Once the 0-cochain node potentials* $\mathbf{e}^0$ *is known, it is easy to obtain 1-cochain voltage drops* $\mathbf{v}^1$ *and 1-cochain branch currents* $\mathbf{j}^1$ *by using the topological equation* $\mathbf{v}^1 = -\left(\delta_0 \mathbf{e}^0\right)$ *and the constitutive equation* $\mathbf{j}^1 - \mathbf{j}_f^1 = \mathbf{G}\left(\mathbf{v}^1 - \mathbf{v}_f^1\right)$.

Similarly following the blue path in the Tonti diagram, the generated system state equation Eq.4.15 would involve:

$$\delta_1 = \begin{bmatrix} -1 & +1 & 0 & 0 \\ 0 & -1 & +1 & +1 \end{bmatrix} \tag{4.21}$$

$$\partial_2 = {\delta_1}^T \tag{4.22}$$

$$\mathbf{R} = \begin{bmatrix} R_1 & 0 & 0 & 0 \\ 0 & R_2 & 0 & 0 \\ 0 & 0 & R_3 & 0 \\ 0 & 0 & 0 & R_4 \end{bmatrix} = \begin{bmatrix} 6 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 \\ 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 4 \end{bmatrix} \tag{4.23}$$

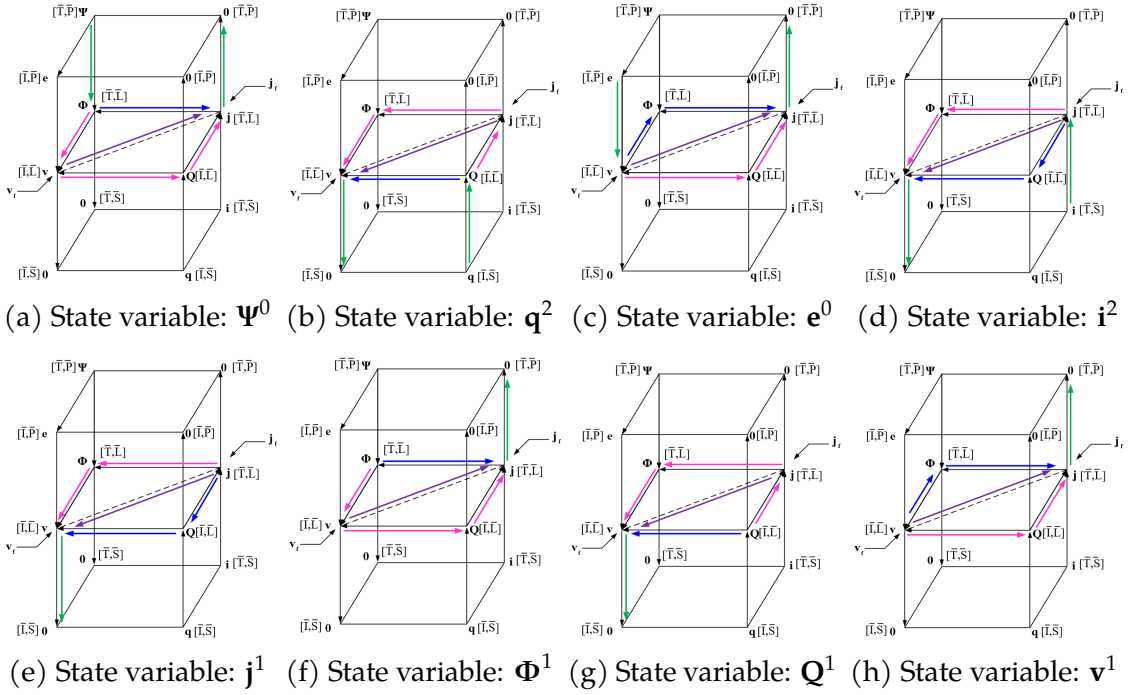
Substituting Eq.4.19 ~ Eq.4.23 into Eq.4.15, we obtain another system of linear equations that yields the solutions for the mesh currents $\mathbf{i}^2 = [\ -19/12 \quad -17/12\ ]^T$. *The 1-cochain branch currents* $\mathbf{j}^1$ *and 1-cochain voltage drops* $\mathbf{v}^1$ *are immediately obtained by applying the*

topological relation $\mathbf{j}^1 = \partial_2 \mathbf{i}^2$ *and the constitutive equation* $\mathbf{v}^1 - \mathbf{v}_f^1 = \mathbf{R}\left(\mathbf{j}^1 - \mathbf{j}_f^1\right)$.

4.3.2 Dynamic systems

In this section, we will extend the above approach to general dynamic electrical circuits. The algebraic topological structure of dynamic electrical circuits relies on cochains from all four cochain complexes modeled over a single cell complex. The topological and constitutive relations between these cochains are given by the diagram in Figure 4.7b and eight different methods of generating the state equations are indicated by paths in the diagrams shown in Figure 4.11. Just as with static systems, each path is a sequence of the arrows indicating composition of the corresponding physical laws. In contrast to static systems, the middle horizontal section of the diagram allows three alternative (pink, purple and blue) paths relating the primal 1-cochain of voltage drops $\mathbf{v}^1$ and the dual 1-cochain of branch currents $\mathbf{j}^1$ corresponding to capacitance, resistance, and inductance constitutive relationship respectively. The presence of alternative paths indicate superposition of the corresponding equations generated by each path.

For example, if we use the paths in Figure 4.11a to generate the state equations, then the 0-cochain $\mathbf{\Psi}^0$ is selected as the state variable. The system state equation can be generated by composition of five physical laws (two topological and three constitutive) starting with a 0-cochain $\mathbf{\Psi}^0$. First, coboundary operator in space δ_0 applied to potential magnetic fluxes $\mathbf{\Psi}^0$ in order to generate magnetic fluxes $\mathbf{\Phi}^1$. Now the path splits in two: the blue arrow corresponds to constitutive law $\mathbf{L}^{-1}$ (of the inductor) that relates the magnetic fluxes to branch currents of inductors; the pink arrow takes the magnetic fluxes to generate voltage drops $\mathbf{v}^1$ via the boundary operator in time ∂_1^t. From here the path splits in two again: the purple arrow corresponds to the constitutive Ohm's law $\mathbf{G}$ that relates the voltage drops to branch currents of resistors; continuing along the pink path, the constitutive capacitance law $\mathbf{C}$ relates the voltage drops to electric charges $\mathbf{Q}^1$ of capacitors is followed by the time coboundary operator δ_0^t applied to electric charges of capacitors to generate branch currents of capacitors. Note that the three paths corresponding to the three



(a) State variable: $\mathbf{\Psi}^0$ (b) State variable: $\mathbf{q}^2$ (c) State variable: $\mathbf{e}^0$ (d) State variable: $\mathbf{i}^2$

(e) State variable: $\mathbf{j}^1$ (f) State variable: $\mathbf{\Phi}^1$ (g) State variable: $\mathbf{Q}^1$ (h) State variable: $\mathbf{v}^1$

Figure 4.11: State equation generation paths on the extended Tonti diagram

constitutive laws merge into a single 1-cochain of branch currents $\mathbf{j}^1$, which is then transformed one more time by the upward green arrow corresponding to KCL $\partial_1 \mathbf{j}^1 = \mathbf{0}$. Taking into account the voltage and current sources, above processes results in state equations Eq.4.24:

$$\partial_1 \left(\delta_0^t \mathbf{C} \left(-\partial_1^t \left(\delta_0 \mathbf{\Psi}^0 \right) - \mathbf{v}_f^1 \right) + \mathbf{R}^{-1} \left(-\partial_1^t \left(\delta_0 \mathbf{\Psi}^0 \right) - \mathbf{v}_f^1 \right) + \mathbf{L}^{-1} \left(- \left(\delta_0 \mathbf{\Psi}^0 \right) - \delta_0^t \mathbf{v}_f^1 \right) + \mathbf{j}_f^1 \right) = \mathbf{0} \tag{4.24}$$

Collecting the terms with known voltage and current sources and moving them to the right hand side, the system state equations Eq.4.24 can be written in a more intuitive form as

$$\partial_1 \left(\underbrace{\delta_0^t \mathbf{C} \partial_1^t \delta_0 \mathbf{\Psi}^0}_{\text{currents of C}} + \underbrace{\mathbf{R}^{-1} \partial_1^t \delta_0 \mathbf{\Psi}^0}_{\text{currents of R}} + \underbrace{\mathbf{L}^{-1} \delta_0 \mathbf{\Psi}^0}_{\text{currents of L}} \right) = \partial_1 \left[\underbrace{\mathbf{j}_\mathbf{f}^1}_{\text{current sources}} - \underbrace{\left(\delta_0^t \mathbf{C} + \mathbf{R}^{-1} + \mathbf{L}^{-1} \delta_0^t \right) \mathbf{v}_\mathbf{f}^1}_{\substack{\text{equivalent current sources} \\ \text{generated from} \\ \text{voltage sources}}} \right] \tag{4.25}$$

Other methods for generating the system state equation follow the different paths in Figure 4.11b - Figure 4.11h. For example, in Figure 4.11b, the process starts with the dual 2-cochain of mesh charge $\mathbf{q}^2$ selected as the state variable and amounts to another composition of the five physical laws indicated by the corresponding paths. The blue, purple, and pink path corresponds to the three constitutive laws (capacitance, resistance, and inductance respectively), relating the branch electric charges to the branch voltage drops. The two green arrows correspond to the ∂_2 operator transforming mesh charges $\mathbf{q}^2$ to branch charges $\mathbf{Q}^1$ and application of KVL ($\delta_1 \mathbf{v}^1 = \mathbf{0}$). Putting it all together and taking into account the voltage and current sources, the composition procedures results in Eq.4.26, or in a more intuitive form similar to Eq.4.25, as shown in Eq.4.27.

$$\delta_1 \left(\partial_1^t \mathbf{L} \left(\delta_0^t \partial_2 \mathbf{q}^2 - \mathbf{j}_f^1 \right) + \mathbf{R} \left(\delta_0^t \partial_2 \mathbf{q}^2 - \mathbf{j}_f^1 \right) + \mathbf{C}^{-1} \left(\partial_2 \mathbf{q}^2 - \partial_1^t \mathbf{j}_f^1 \right) + \mathbf{v}_f^1 \right) = \mathbf{0} \tag{4.26}$$

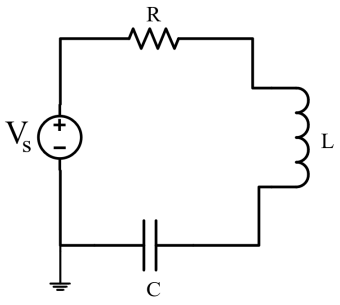
$$\delta_1 \left(\underbrace{\partial_1^t \mathbf{L} \delta_0^t \partial_2 \mathbf{q}^2}_{\text{voltage drops of L}} + \underbrace{\mathbf{R} \delta_0^t \partial_2 \mathbf{q}^2}_{\text{voltage drops of R}} + \underbrace{\mathbf{C}^{-1} \partial_2 \mathbf{q}^2}_{\text{voltage drops of C}} \right) = \delta_1 \left[\underbrace{-\mathbf{v}_\mathbf{f}^1}_{\text{voltage sources}} + \underbrace{\left(\partial_1^t \mathbf{L} + \mathbf{R} + \mathbf{C}^{-1} \partial_1^t \right) \mathbf{j}_\mathbf{f}^1}_{\substack{\text{equivalent voltage drops} \\ \text{generated from} \\ \text{current sources}}} \right] \tag{4.27}$$

It can be observed that Eq.4.25 and Eq.4.27 respectively represent the current and voltage equilibrium of the system. The generated system state equations may be viewed as algebraic with coboundary operators interpreted as finite difference operators on a finite cell complex. However, as we already observed, in lumped

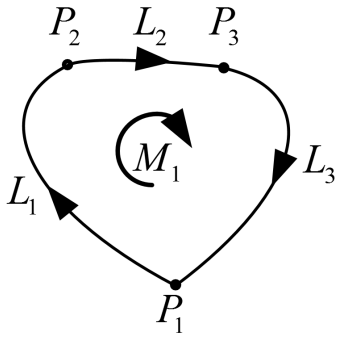
parameter systems space and time are treated separately, and discretization of time is often delayed until a particular numerical integration scheme is chosen. In this case, viewing boundary ∂_1^t and coboundary δ_0^t operations as differentiation in time syntactically transforms Eq.4.25 and Eq.4.27 to a more familiar form:

$$\partial_1 \left(\underbrace{\mathbf{C}\delta_0\ddot{\mathbf{\Psi}}^0}_{\text{currents of C}} + \underbrace{\mathbf{R}^{-1}\delta_0\dot{\mathbf{\Psi}}^0}_{\text{currents of R}} + \underbrace{\mathbf{L}^{-1}\delta_0\mathbf{\Psi}^0}_{\text{currents of L}} \right) = \partial_1 \left[\underbrace{\mathbf{j}_\mathbf{f}^1}_{\text{current sources}} - \underbrace{\left(\mathbf{C}\dot{\mathbf{v}}_\mathbf{f}^1 + \mathbf{R}^{-1}\mathbf{v}_\mathbf{f}^1 + \mathbf{L}^{-1}\int \mathbf{v}_\mathbf{f}^1 \mathrm{d}t \right)}_{\substack{\text{equivalent current sources} \\ \text{generated from} \\ \text{voltage sources}}} \right] \tag{4.28}$$

$$\delta_1 \left(\underbrace{\mathbf{L}\partial_2\ddot{\mathbf{q}}^2}_{\text{voltage drops of C}} + \underbrace{\mathbf{R}\partial_2\dot{\mathbf{q}}^2}_{\text{voltage drops of R}} + \underbrace{\mathbf{C}^{-1}\partial_2\mathbf{q}^2}_{\text{voltage drops of L}} \right) = \delta_1 \left(\underbrace{-\mathbf{v}_\mathbf{f}^1}_{\text{voltage sources}} + \underbrace{\mathbf{L}\dot{\mathbf{j}}_\mathbf{f}^1 + \mathbf{R}\mathbf{j}_\mathbf{f}^1 + \mathbf{C}^{-1}\int \mathbf{j}_\mathbf{f}^1 \mathrm{d}t}_{\substack{\text{equivalent voltage drops} \\ \text{generated from} \\ \text{current sources}}} \right) \tag{4.29}$$



(a) A simple RLC electrical circuit



(b) Topological structure

Figure 4.12: A simple RLC electrical circuit and its topological structure

Example 2. *We will use a simple example of RLC electrical circuit (Figure 4.12a) to illustrate the derivation of Eq.4.29. The electrical network contains one resistor R, one*

capacitor C, one inductor L and one voltage source V_s. *Topologically, the network is a 2-dimensional cell complex shown in Figure 4.12b and consists of three 0-cells* (P_1,P_2,P_3), *three 1-cells* (L_1,L_2,L_3) *and one 2-cell* (M_1).

The algebraic topological model of the electrical circuit contains: primal 0-cochain node potentials ($\mathbf{e}^0 = e_1 \cdot P_1 + e_2 \cdot P_2 + e_3 \cdot P_3$), *primal 0-cochain magnetic flux potentials* ($\mathbf{\Psi}^0 = \psi_1 \cdot P_1 + \psi_2 \cdot P_2 + \psi_3 \cdot P_3$), *primal 1-cochain voltage drops* ($\mathbf{v}^1 = v_1 \cdot L_1 + v_2 \cdot L_2 + v_3 \cdot L_3$), *primal 1-cochain magnetic fluxes* ($\mathbf{\Phi}^1 = \phi_1 \cdot L_1 + \phi_2 \cdot L_2 + \phi_3 \cdot L_3$), *dual 1-cochain currents* ($\mathbf{j}^1 = j_1 \cdot L_1 + j_2 \cdot L_2 + j_3 \cdot L_3$), *dual 1-cochain electric charges* ($\mathbf{Q}^1 = Q_1 \cdot L_1 + Q_2 \cdot L_2 + Q_3 \cdot L_3$), *dual 2-cochain mesh currents* ($\mathbf{i}^2 = i \cdot M_1$), *dual 2-cochain mesh electric charges* ($\mathbf{q}^2 = q \cdot M_1$), *and four cochains that are always* $\mathbf{0}$: *2-cochain loop voltage drops, 2-cochain loop magnetic fluxes, 0-cochain node currents and 0-cochain of node electric charges. Following the paths in Figure 4.11b generates Eq.4.29, with individual terms as follows:*

$$\delta_1 = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix} \tag{4.30}$$

$$\partial_2 = {\delta_1}^T \tag{4.31}$$

$$\mathbf{L} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & L & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{4.32}$$

$$\mathbf{R} = \begin{bmatrix} R & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{4.33}$$

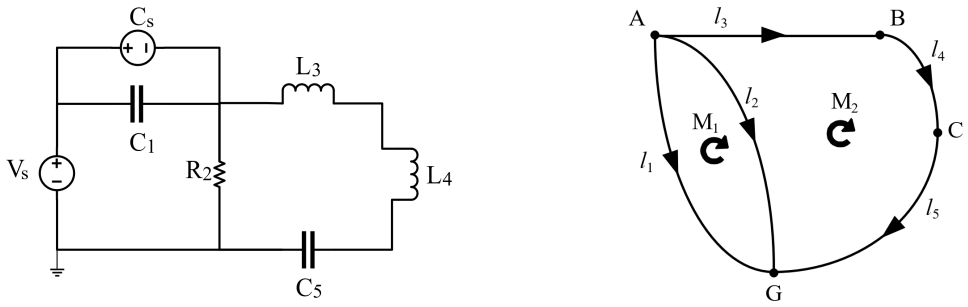
$$\mathbf{C}^{-1} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & C^{-1} \end{bmatrix} \tag{4.34}$$

$$\mathbf{v}_\mathbf{f}^1 = \begin{bmatrix} -V_s & 0 & 0 \end{bmatrix}^T \tag{4.35}$$

$$\mathbf{j}_{\mathbf{f}}^{1} = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}^{\mathsf{T}} \tag{4.36}$$

Substituting Eq.4.30 ~ Eq.4.36 into Eq.4.29, we obtain the system state equations as shown in Eq.4.37.

$$\mathrm{L}\ddot{\mathrm{q}} + \mathrm{R}\dot{\mathrm{q}} + \mathrm{C}^{-1}\mathrm{q} = \mathrm{V}_{\mathrm{s}} \tag{4.37}$$



(a) An RLC electrical circuit (b) Topological structure

Figure 4.13: An RLC electrical circuit and its topological structure

Example 3. *We will use another example of electrical circuit in Figure 4.13a to illustrate the derivation of Eq.4.28. The shown electrical circuit contains one resistor* R_2*, two capacitors* C_1*,* C_5*, two inductors* L_3*,* L_4*, one current source* C_s *and one voltage source* V_s*. Topologically, the network is a 2-dimensional cell complex shown in Figure 4.13b, and consisting of four 0-cells* (A,B,C,G)*, five 1-cells*$(\mathrm{l}_1,\mathrm{l}_2,\mathrm{l}_3,\mathrm{l}_4,\mathrm{l}_5)$ *and two 2-cells*$(\mathrm{M}_1,\mathrm{M}_2)$*.*

The algebraic topological model of the electrical circuit contains: primal 0-cochain node potentials $(\mathbf{e}^0 = \mathrm{e}_A \cdot A + \mathrm{e}_B \cdot B + \mathrm{e}_C \cdot C + \mathrm{e}_G \cdot G)$*, primal 0-cochain magnetic flux potentials* $(\mathbf{\Psi}^0 = \psi_A \cdot A + \psi_B \cdot B + \psi_C \cdot C + \psi_G \cdot G)$*, primal 1-cochain voltage drops* $(\mathbf{v}^1 = \nu_1 \cdot \mathrm{l}_1 + \nu_2 \cdot \mathrm{l}_2 + \nu_3 \cdot \mathrm{l}_3 + \nu_4 \cdot \mathrm{l}_4 + \nu_5 \cdot \mathrm{l}_5)$*, primal 1-cochain magnetic fluxes* $(\mathbf{\Phi}^1 = \phi_1 \cdot \mathrm{l}_1 + \phi_2 \cdot \mathrm{l}_2 + \phi_3 \cdot \mathrm{l}_3)$*, dual 1-cochain currents* $(\mathbf{j}^1 = \mathrm{j}_1 \cdot \mathrm{l}_1 + \mathrm{j}_2 \cdot \mathrm{l}_2 + \mathrm{j}_3 \cdot \mathrm{l}_3 + \mathrm{j}_4 \cdot \mathrm{l}_4 + \mathrm{j}_5 \cdot \mathrm{l}_5)$*, dual 1-cochain electric charges* $(\mathbf{Q}^1 = \mathrm{Q}_1 \cdot \mathrm{l}_1 + \mathrm{Q}_2 \cdot \mathrm{l}_2 + \mathrm{Q}_3 \cdot \mathrm{l}_3 + \mathrm{Q}_4 \cdot \mathrm{l}_4 + \mathrm{Q}_5 \cdot \mathrm{l}_5)$*, dual 2-cochain mesh currents* $(\mathbf{i}^2 = \mathrm{i}_1 \cdot \mathrm{M}_1 + \mathrm{i}_2 \cdot \mathrm{M}_2)$*, dual 2-cochain mesh electric charges* $(\mathbf{q}^2 = \mathrm{q}_1 \cdot \mathrm{M}_1 + \mathrm{q}_2 \cdot \mathrm{M}_2)$*, and four cochains that are always* $\mathbf{0}$*: 2-cochain loop voltage drops, 2-cochain loop magnetic fluxes, 0-cochain node currents and 0-cochain of node electric*

charges. In order to get unique solution of the state equations, we consider 0-cell G as the reference node, implying the boundary condition of $e_G = 0$*. Following the paths in Figure 4.11a, generates Eq.4.28, with individual terms as follows:*

$$\partial_1 = \begin{bmatrix} +1 & +1 & 0 & 0 & +1 \\ -1 & -1 & -1 & 0 & 0 \\ 0 & 0 & +1 & -1 & 0 \\ 0 & 0 & 0 & +1 & -1 \end{bmatrix} \tag{4.38}$$

$$\delta_0 = \partial_1^T \tag{4.39}$$

$$\mathbf{C} = \begin{bmatrix} C_1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & C_5 \end{bmatrix} \tag{4.40}$$

$$\mathbf{R}^{-1} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 1/R_2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{4.41}$$

$$\mathbf{L}^{-1} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1/L_3 & 0 & 0 \\ 0 & 0 & 0 & 1/L_4 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{4.42}$$

$$\mathbf{v}_\mathbf{f}^1 = \begin{bmatrix} V_s & 0 & 0 & 0 & 0 \end{bmatrix}^T \tag{4.43}$$

$$\mathbf{j}_\mathbf{f}^1 = \begin{bmatrix} C_s & 0 & 0 & 0 & 0 \end{bmatrix}^T \tag{4.44}$$

Substituting Eq.4.38 ~ Eq.4.44 into Eq.4.28, we obtain the system state equations Eq.4.45 as follows:

$$\begin{bmatrix} C_1+C_5 & -C_1 & 0 & -C_5 \\ -C_1 & C_1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ -C_5 & 0 & 0 & C_5 \end{bmatrix} \ddot{\boldsymbol{\Psi}}^0 + \begin{bmatrix} 1/R_2 & -1/R_2 & 0 & 0 \\ -1/R_2 & 1/R_2 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \dot{\boldsymbol{\Psi}}^0 + \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 1/L_3 & -1/L_3 & 0 \\ 0 & -1/L_3 & 1/L_3+1/L_4 & -1/L_4 \\ 0 & 0 & -1/L_4 & 1/L_4 \end{bmatrix} \boldsymbol{\Psi}^0 = \begin{bmatrix} C_s - \dot{V}_s C_1 \\ -C_s + \dot{V}_s C_1 \\ 0 \\ 0 \end{bmatrix} \tag{4.45}$$

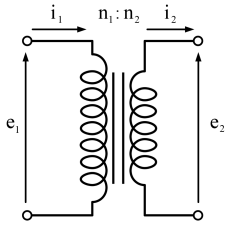
4.4 Multi-domain lumped parameter systems

4.4.1 Interactions of single-domain models

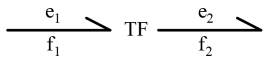
Engineering systems are usually constructed as compositions of single-domain subsystems in order to perform complex engineering tasks. Representative examples include electric motors (electro-mechanical systems), ovens (electro-thermal systems), hydraulic pumps (hydraulic-mechanical systems), etc. We will refer to such systems as multi-domain systems, where lumped-parameter behavior of each single-domain is governed by an extended Tonti diagram as described in the previous section. It should be understood that multi-domain systems subsume the special case of homogeneous systems where all single-domain systems are of the same type, e.g. all electrical or all mechanical. Composition of two single-domain systems is associated with the process of energy conversion between the two systems, also called *transduction* [69]. Specific means of transduction vary from system to system, but common examples of transducers (devices the perform transduction) include gears and levers, electrical transformers, motors, piezoelectric devices, hydraulic pumps, and so on.

Devices used to couple the same type variables of different physical domains are usually called transformers (e.g. electrical transformers), while devices coupling the dual type variables are called gyrators (e.g. electric motors). Physical transformers and gyrators always have energy loss during the energy transduction, but if the leakage is small enough then the energy loss is usually neglected during modeling. Transformers and gyrators with no energy loss are called ideal transformers and gyrators [69]. System modeling languages generally use ideal transformers and

gyrators as abstract constructs in order to avoid modeling complex structures of physical transformers and gyrators devices. For instance, bond graphs use abstract 2-port transformers and gyrators which can be used to connect any 1-ports. Figure 4.14 and Figure 4.15 show several transformer and gyrator models from different system modeling languages.



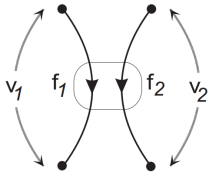
(a) Schematic ideal electrical transformer



(b) Bond graph model of transformers

```
model IdealTransformer "Ideal electrical transformer"
  extends Modelica.Electrical.Analog.Interfaces.TwoPort;
  parameter Real n=1 "Turns ratio";
equation
  v1 = n*v2;
  i2 = -n*i1;
end IdealTransformer;
```

(c) Modelica model of the ideal electrical transformer

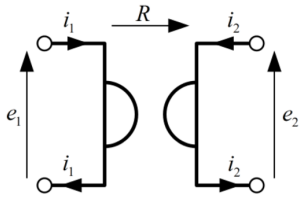


(d) Linear graph model of transformers

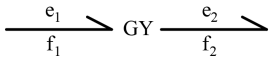
Figure 4.14: Comparison of transformers of different modeling languages

Such abstract transformers and gyrators in each language can represent hundreds of complex transformer and gyrator devices. For example, Figures 4.16 and 4.17 show two examples of multi-domain models represented by different modeling languages. Figure 4.16 is an example of a homogeneous multi-domain system, where all sub-domains are electrical domains. Figure 4.17 is an example of a heterogeneous multi-domain system, where one electrical domain and one mechanical domain are connected by a DC motor.

In our proposed combinatorial model of lumped parameter systems, physical transducer devices can be abstracted by additional relations between primal and dual variables in each of the subsystems; such relations may be governed by additional constitutive, interaction, or conservation constrains imposed on the multi-domain system [112]. For example, consider the three-domain system con-



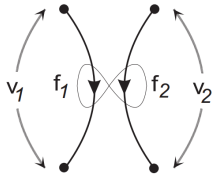
(a) Schematic ideal electrical gyrator



(b) Bond graph model of gyrators

```
model IdealGyrator "Ideal gyrator"
  extends Modelica.Electrical.Analog.Interfaces.TwoPort;
  parameter SIunits.Conductance G=1 "Gyration conductance";
equation
  i1 = G*v2;
  i2 = -G*v1;
end IdealGyrator;
```

(c) Modelica model of the ideal gyrator



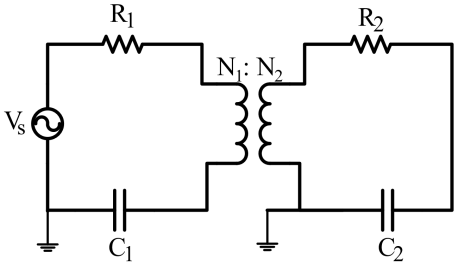
(d) Linear graph model of gyrators

Figure 4.15: Comparison of gyrators of different modeling languages

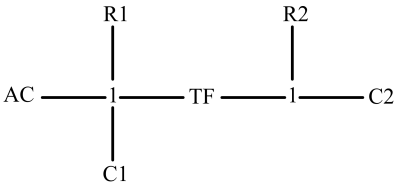
sisting of a hydraulic pump that is controlled by an electrical motor. Behaviors of the subsystems (electrical, rotational motion, and hydraulic) are described by the corresponding extended Tonti diagrams) that are further constrained as shown in Figure 4.18.

In principle, such a representation is sufficient for capturing the behavior of a multi-domain system. Each Tonti diagram corresponds to a system of ordinary differential equations that are coupled by the transducer constraints. When the constraints are algebraic, this representation corresponds to the usual system of differential algebraic equations. More complex transducer relationships may involve multiple physical variables as well as non-linear and differential constraints [108], resulting in more complex models of behaviors.

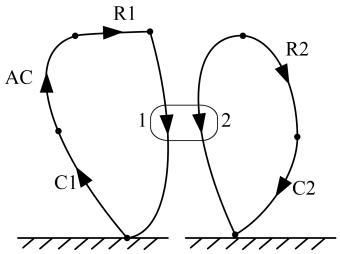
However, this representation neither recognizes nor takes advantage of the fact that all single-domain behaviors are isomorphic, which allows to treat the whole multi-domain system as a collection of four constrained cochain complexes on a *single* cell complex model. Below we will define such a model, which takes a form of a generalized Tonti diagram. We then show that, in the presence of two most common transducers: ideal transformers and gyrators, the governing equations for such a model may be generated by following the paths on the generalized Tonti



(a) A multi-domain RLC electrical circuit



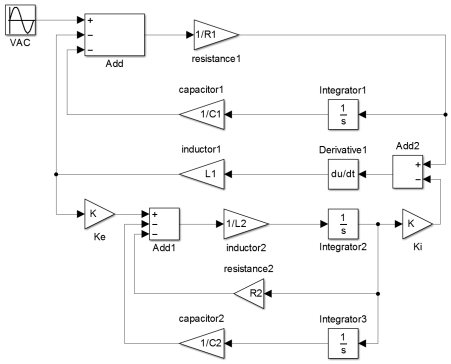
(b) Bond graph model



(c) Linear graph model

```
model homogeneous
  Modelica.Electrical.Analog.Basic.Resistor R1(R = 5);
  Modelica.Electrical.Analog.Sources.SineVoltage AC(V = 5, freqHz = 10);
  Modelica.Electrical.Analog.Basic.Capacitor C1(C = 0.01);
  Modelica.Electrical.Analog.Basic.Transformer T(L1 = 0.2, L2 = 0.1, M = 0);
  Modelica.Electrical.Analog.Basic.Resistor R2(R = 5);
  Modelica.Electrical.Analog.Basic.Capacitor C2(C = 0.001);
  Modelica.Electrical.Analog.Basic.Ground ground1;
  Modelica.Electrical.Analog.Basic.Ground ground2;
equation
  connect(ground1.p, AC.n);
  connect(ground2.p, C2.n);
  connect(C2.p, T.n2);
  connect(R2.n, C2.n);
  connect(R2.p, T.p2);
  connect(R1.n, T.p1);
  connect(AC.p, R1.p);
  connect(C1.p, AC.n);
  connect(T.n1, C1.n);
end homogeneous;
```

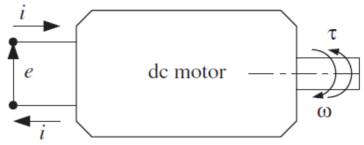
(d) Modelica model



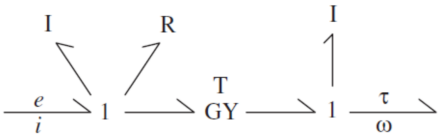
(e) Simulink model

Figure 4.16: Different models of a multi-domain RLC electrical circuit

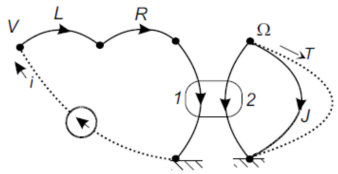
diagram. This results extends the result of Section 4.3 to multi-domain lumped parameter systems.



(a) A schematic DC motor



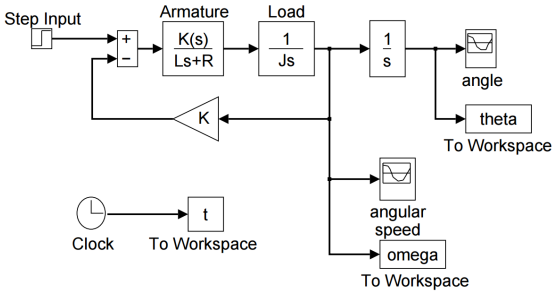
(b) Bond graph model [69]



(c) Linear graph model [104]

```
model DC_motor
  Modelica.Electrical.Analog.Sources.ConstantVoltage constantVoltage1;
  Modelica.Electrical.Analog.Basic.Inductor inductor1;
  Modelica.Electrical.Analog.Basic.Resistor resistor1;
  Modelica.Electrical.Analog.Basic.EMF emf;
  Modelica.Mechanics.Rotational.Components.Inertia inertia1;
  Modelica.Electrical.Analog.Basic.Ground ground1;
equation
  connect(emf.flange, inertia1.flange_a);
  connect(emf.n, constantVoltage1.n);
  connect(resistor1.n, emf.p);
  connect(inductor1.n, resistor1.p);
  connect(constantVoltage1.p, inductor1.p);
  connect(ground1.p, constantVoltage1.n);
end DC_motor;
```

(d) Modelica model



(e) Simulink model [8]

Figure 4.17: Different models of a DC motor

4.4.2 Generalized Tonti diagram for multi-domain systems

Since LPMs in different physical domain are isomorphic, so are their corresponding Tonti diagrams. In this sense, a single Tonti diagram describes behavior of all lumped parameter systems, provided that the variable of the same space-time type are identified and generalized.

Two most common generalizations are mechanical (generalized displacement-force model) and electrical (generalized voltage-current model). For the sake of consistency with the discussion in Section 4.3, we will adopt the generalized electrical model. For example, the electrical voltage, mechanical translational velocity and hydraulic pressure difference are all considered to be of the same type called the generalized voltage; the electrical resistors, mechanical dampers and hydraulic

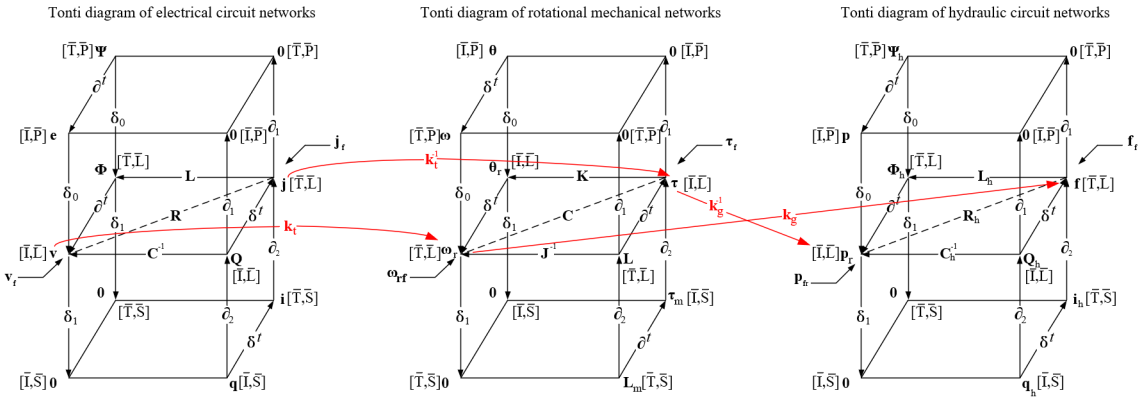


Figure 4.18: Behavior of three-domain system, a hydraulic pump driven by electrical motor, is abstracted by relations between three interacting extended Tonti diagrams.

Commonly-used physical variables in the Tonti diagram of rotational mechanical networks: θ - angle, θ_r - relative angle, ω - rotational velocity, ω_r - relative rotational velocity, $\boldsymbol{\tau}$ - torque, $\mathbf{M}$ - angular momentum, $\mathbf{K}$ - rotational stiffness, $\mathbf{D}$ - rotational damping coefficient and $\mathbf{J}$ - moment of inertia. Commonly-used physical variables in the Tonti diagram of hydraulic networks: $_h$ - potential of hydraulic flux, $\mathbf{fl}_h$ - hydraulic flux, $\mathbf{p}$ - pressure, $\mathbf{p}_r$ - pressure drop, $\mathbf{Q}_h$ - flow of volume, $\mathbf{f}$ - flow rate, $\mathbf{L}_h$ - hydraulic inductance, $\mathbf{R}_h$ - hydraulic resistance, $\mathbf{C}_h$ - hydraulic capacitance. $\mathbf{h}_t$ and $\mathbf{k}_t$ are transformer modulus. $\mathbf{k}_t$ is the ratio of relative rotational velocity of motor to voltage drop and $\mathbf{h}_t$ is the ratio of torque to current. $\mathbf{h}_g$ and $\mathbf{k}_g$ are gyrator modulus. $\mathbf{k}_g$ is the ratio of flow rate to relative rotational velocity of motor and $\mathbf{h}_g$ is the ratio of pressure drop to torque.

resistors are all identified as generalized resistors, and so on. In order to emphasize the generalized nature of all physical quantities and to distinguish them from the actual physical electrical network model, we will choose a different set of symbols. Specifically, the generalized Tonti diagram is defined by four exact cochain sequences on a single cell complex:

$$\text{primal}: \quad \mathbf{p}^0 \xrightarrow{\delta_0} \mathbf{a}^1 \xrightarrow{\delta_1} \mathbf{0}^2 \tag{4.46}$$

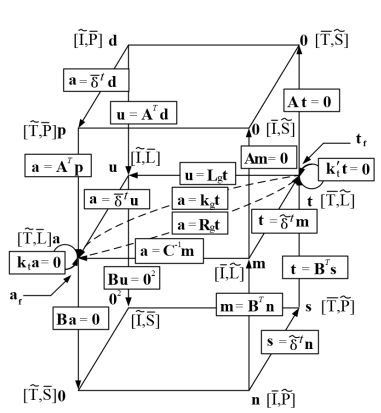
$$\mathbf{d}^0 \xrightarrow{\delta_0} \mathbf{u}^1 \xrightarrow{\delta_1} \mathbf{0}^2 \tag{4.47}$$

$$\text{dual}: \quad \mathbf{s}^2 \xrightarrow{\partial_2} \mathbf{t}^1 \xrightarrow{\partial_1} \mathbf{0}^0 \tag{4.48}$$

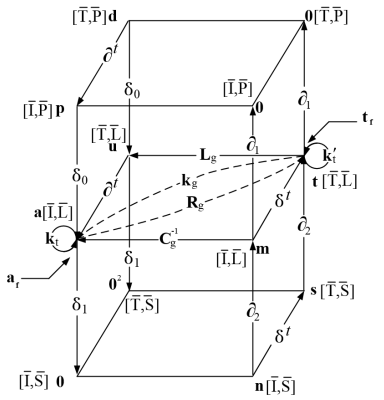
$$\mathbf{n}^2 \xrightarrow{\partial_2} \mathbf{m}^1 \xrightarrow{\partial_1} \mathbf{0}^0 \tag{4.49}$$

,where $\mathbf{p}^0$ is a 0-cochain generalized potentials, $\mathbf{a}^1$ is a 1-cochain generalized voltages, $\mathbf{d}^0$ is a 0-cochain generalized potential magnetic fluxes, $\mathbf{u}^1$ is a 1-cochain generalized magnetic fluxes, $\mathbf{s}^2$ is a 2-cochain generalized mesh currents, $\mathbf{t}^1$ is a 1-cochain generalized currents, $\mathbf{n}^2$ is 2-cochain generalized mesh charges, and $\mathbf{m}^1$ is a 1-cochain generalized electric charges. There are also four cochains that are always $\mathbf{0}$: 2-cochain of generalized mesh magnetic fluxes, 2-cochain of generalized mesh voltages, 0-cochain of generalized node currents and 0-cochain of generalized node electric charges.

With such a generalization, all the physical variables of the same space-time type are replaced by their generalized counterparts, effectively transforming model of the heterogeneous multi-domain system in an abstract (generalized) homogeneous system. The behavior of this system is governed by the generalized Tonti diagram shown in Figure 4.19. As before, the generalized primal and dual cochains are related by (generalized) constitutive relations: resistance $\mathbf{R}_g$, capacitance $\mathbf{C}_g$, and inductance $\mathbf{L}_g$.



(a) Matrix operators on dual cell complexes



(b) Topological operators on a single cell complex

Figure 4.19: Generalized extended Tonti diagram for generalized RLC network system

Furthermore, since all physical quantities are generalized, the actions of ideal transformers and gyrators can be modeled simply as additional constraints on the

cochains in the generalized Tonti diagram. Traditionally, a transformer is abstracted as a linear transformation

$$\begin{bmatrix} a_1 \\ t_1 \end{bmatrix} = \begin{bmatrix} k_t & 0 \\ 0 & 1/k_t \end{bmatrix} \begin{bmatrix} a_2 \\ t_2 \end{bmatrix} \tag{4.50}$$

where k_t is the transformer's modulus measuring the ratio between two (generalized) voltages a_1 and a_2, as well as the reciprocal ratio between the generalized currents t_1 and t_2 in order to enforce energy balance $a_1 t_1 = a_2 t_2$. It is easy to see that the same relationships may be enforced by a pair of linear constraints

$$\begin{cases} \begin{bmatrix} 1 & -k_t \end{bmatrix} \cdot \begin{bmatrix} a_1 & a_2 \end{bmatrix}^T = 0 \\ \begin{bmatrix} 1 & -k_t^{-1} \end{bmatrix} \cdot \begin{bmatrix} t_1 & t_2 \end{bmatrix}^T = 0 \end{cases} \tag{4.51}$$

Generalizing, every ideal transformer can be represented by a pair of linear constraints

$$\mathbf{k}_t \mathbf{a} = 0; \qquad \mathbf{k}_t' \mathbf{t} = 0$$

on cochains of generalized voltages $\mathbf{a}$ and currents $\mathbf{t}$. These constraints are indicated on the generalized Tonti diagram in Figure 4.19 by two cycles.

Similarly, the effect of an abstract gyrator is usually described by a linear transformation

$$\begin{bmatrix} a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} 0 & k_g \\ k_g & 0 \end{bmatrix} \begin{bmatrix} t_1 \\ t_2 \end{bmatrix} \tag{4.52}$$

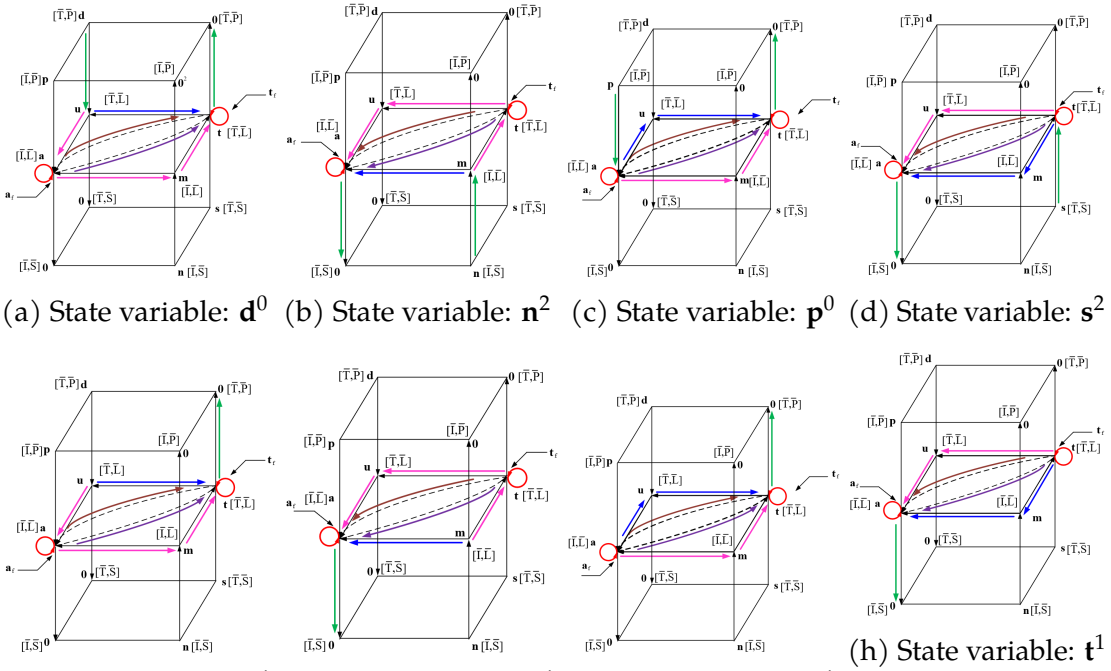
where the modulus k_g relates the dual quantities in two interacting domains: generalized voltage a_1 of the first domain is proportional to the generalized current t_2 of the second domain, and vice versa, again satisfying the ideal energy balance law. Equivalently, a generalized gyrator may be represented by a linear transformation $\mathbf{k}_g$ that relates the cochains of generalized voltages and currents, as indicated by a dotted arrow in the Tonti diagram in Figure 4.19.

4.4.3 System state equations of multi-domain systems

With all physical variables generalized, the heterogeneous multi-domain system now becomes a homogeneous multi-domain system in terms of generalized physical variables. Instead of multiple 2-cochain complexes associated with different types of physical variables, the algebraic topological model of the multi-domain system is now a set of 2-cochain complexes associated with the same (generalized) type of physical variables that are defined over a single cell complex and are constrained by abstract transformers and gyrators.

Eight different methods of generating the state equations are indicated by paths in the generalized Tonti diagram shown in Figure 4.20. Just as with the single domain systems, each path is a sequence of the arrows indicating composition of the corresponding physical laws. In contrast to the single domain diagram, the middle horizontal section of the generalized Tonti diagram allows an additional alternative path relating the primal 1-cochain of generalized voltages $\mathbf{a}^1$ and the dual 1-cochain of generalized currents $\mathbf{t}^1$ by gyrators as well as two alternative cyclic paths (shown in red), which respectively constrain the generalized voltages and currents of transformers.

For example, if we use the paths in Figure 4.20a to generate the state equations, then the 0-cochain $\mathbf{d}^0$ is selected as the state variable. The system state equation can be generated by composition of seven physical laws (two topological and five constitutive) starting with a 0-cochain $\mathbf{d}^0$. First, coboundary operator in space δ_0 applied to generalized potential magnetic fluxes $\mathbf{d}^0$ in order to generate generalized magnetic fluxes $\mathbf{u}^1$. Now the path splits into two: the blue arrow corresponds to constitutive law $\mathbf{L}_g^{-1}$ that relates generalized magnetic fluxes to generalized currents of generalized inductors; the pink arrow takes the generalized magnetic fluxes to generate generalized voltages $\mathbf{a}^1$ via the boundary operator in time ∂_1^t. From here the path splits into four: (1) the purple arrow corresponds to the constitutive law $\mathbf{R}_g^{-1}$ that relates generalized voltages to generalized currents of generalized resistors; (2) the brown arrow corresponds to the constitutive law $\mathbf{k}_g^{-1}$ that relates generalized voltages to generalized currents of gyrators; (3) the left red cyclic



(a) State variable: $\mathbf{d}^0$ (b) State variable: $\mathbf{n}^2$ (c) State variable: $\mathbf{p}^0$ (d) State variable: $\mathbf{s}^2$

(e) State variable: $\mathbf{u}^1$ (f) State variable: $\mathbf{m}^1$ (g) State variable: $\mathbf{a}^1$ (h) State variable: $\mathbf{t}^1$

Figure 4.20: State equation generation paths on the generalized extended Tonti diagram

arrow corresponds to constitutive law $\mathbf{k}_t$ that constrains the generalized voltages of generalized transformers; (4) the pink arrow takes the generalized voltages to generate generalized electric charge of generalized capacitors by using constitutive law $\mathbf{C}_g$, followed by taking the generalized electric charge to generate generalized currents of generalized capacitors via the coboundary operator in time δ_0^t. Note that these four paths and the right red cyclic path[6] corresponding to five constitutive laws merge into a single 1-cochain of generalized through variables $\mathbf{t}^1$, which is then transformed one more time by the upward green arrow corresponding to KCL $\partial_1 \mathbf{t}^1 = \mathbf{0}$. Taking into account the generalized sources, collecting the terms with known generalized sources and moving them to the right hand side, above

[6]Note that the generalized currents of transformers $\mathbf{t}_T^1$ cannot be directly obtained from the generalized voltages of transformers $\mathbf{a}_T^1$, so we treat them as additional unknown variables in the system state equation.

procedure results in the state equation Eq.4.53, with Eq.4.54 and Eq.4.55 being the constraint equations generated from two cyclic red paths.

$$\partial_1 \left(\underbrace{\delta_0^t \mathbf{C}_g \partial_1^t \delta_0 \mathbf{d}^0}_{\substack{\text{generalized currents} \\ \text{of generalized C}}} + \underbrace{\mathbf{R}_g^{-1} \partial_1^t \delta_0 \mathbf{d}^0}_{\substack{\text{generalized currents} \\ \text{of generalized R}}} + \underbrace{\mathbf{L}_g^{-1} \delta_0 \mathbf{d}^0}_{\substack{\text{generalized currents} \\ \text{of generalized L}}} + \underbrace{\mathbf{k}_g^{-1} \partial_1^t \delta_0 \mathbf{d}^0}_{\substack{\text{generalized currents} \\ \text{of gyrators}}} + \underbrace{\mathbf{t}_T^1}_{\substack{\text{generalized currents} \\ \text{of transformers}}} \right)$$

$$= \partial_1 \left[\underbrace{\mathbf{t}_f^1}_{\substack{\text{generalized currents} \\ \text{sources}}} \quad \underbrace{- \left(\delta_0^t \mathbf{C}_g + \mathbf{R}_g^{-1} + \mathbf{k}_g^{-1} + \mathbf{L}_g^{-1} \delta_0^t \right) \mathbf{a}_f^1}_{\substack{\text{equivalent generalized current} \\ \text{sources generated from} \\ \text{generalized voltage sources}}} \right] \tag{4.53}$$

$$\mathbf{k}_t \left(\partial_1^t \delta_0 \mathbf{d}^0 - \mathbf{a}_f^1 \right) = \mathbf{0} \tag{4.54}$$

$$\mathbf{k}_t' \left(\mathbf{t}^1 - \mathbf{t}_f^1 \right) = \mathbf{0} \tag{4.55}$$

Assuming that the number of state variables is N and the number of transformers is M, then the system of equations Eq.4.53 has N+$2M$ unknowns. The two transformer's constraints generate $2M$ constraint equations, while the other paths generate N state equations. As expected, the number of unknowns equals to the total number of state and constraint equations.

Other methods for generating the system state equation follow the different paths in Figure 4.20b ~ Figure 4.20h. For example, in Figure 4.20b, the process starts with the dual 2-cochain of generalized mesh electric charges $\mathbf{n}^2$ selected as the state variable and amounts to another composition of the seven physical laws indicated by the corresponding paths. The blue, purple, pink and brown path corresponds to the four generalized constitutive laws (generalized capacitance, resistance, inductance and gyrator), relating the generalized currents to generalized voltages. The two green arrows correspond to the ∂_2 operator transforming $\mathbf{n}^2$ to generalized electric charges $\mathbf{m}^1$ and application of KVL ($\delta_1 \mathbf{a}^1 = \mathbf{0}$). The two red cyclic arrows correspond

to the constitutive equations of transformers. Putting it all together and taking into account the generalized sources, the composition procedures results in Eq.4.56, with Eq.4.57 and Eq.4.58 being the constraint equations generated from two cyclic red paths.

$$\delta_1 \left(\underbrace{\partial_1^t \mathbf{L}_g \delta_0^t \partial_2 \mathbf{n}^2}_{\substack{\text{generalized voltages} \\ \text{of generalized L}}} + \underbrace{\mathbf{R}_g \delta_0^t \partial_2 \mathbf{n}^2}_{\substack{\text{generalized voltages} \\ \text{of generalized R}}} + \underbrace{\mathbf{C}_g^{-1} \partial_2 \mathbf{n}^2}_{\substack{\text{generalized voltages} \\ \text{of generalized C}}} + \underbrace{\mathbf{k}_g \delta_0^t \partial_2 \mathbf{n}^2}_{\substack{\text{generalized voltages} \\ \text{of gyrators}}} + \underbrace{\mathbf{a}_T^1}_{\substack{\text{generalized voltages} \\ \text{of transformers}}} \right)$$
$$= \delta_1 \left[\underbrace{-\mathbf{a}_f^1}_{\substack{\text{generalized voltages} \\ \text{sources}}} + \underbrace{\left(\partial_1^t \mathbf{L}_g + \mathbf{R}_g + \mathbf{k}_g + \mathbf{C}_g^{-1} \partial_1^t \right) \mathbf{t}_f^1}_{\substack{\text{equivalent generalized voltage} \\ \text{sources generated from} \\ \text{generalized current sources}}} \right] \tag{4.56}$$

$$\mathbf{k}_t' \left(\delta_0^t \partial_2 \mathbf{n}^2 - \mathbf{t}_f^1 \right) = \mathbf{0} \tag{4.57}$$

$$\mathbf{k}_t \left(\mathbf{a}^1 - \mathbf{a}_f^1 \right) = \mathbf{0} \tag{4.58}$$

As with any dynamic system, interpreting boundary ∂_1^t and coboundary δ_0^t operations as differentiation in time syntactically transforms Eq.4.53 ~ Eq.4.55 and Eq.4.56 ~ Eq.4.58 to a more familiar form:

$$\partial_1 \left(\underbrace{\mathbf{C}_g \delta_0 \ddot{\mathbf{d}}^0}_{\substack{\text{generalized currents} \\ \text{of generalized C}}} + \underbrace{\mathbf{R}_g^{-1} \delta_0 \dot{\mathbf{d}}^0}_{\substack{\text{generalized currents} \\ \text{of generalized R}}} + \underbrace{\mathbf{L}_g^{-1} \delta_0 \mathbf{d}^0}_{\substack{\text{generalized currents} \\ \text{of generalized L}}} + \underbrace{\mathbf{k}_g^{-1} \delta_0 \dot{\mathbf{d}}^0}_{\substack{\text{generalized currents} \\ \text{of gyrators}}} + \underbrace{\mathbf{t}_T^1}_{\substack{\text{generalized currents} \\ \text{of transformers}}} \right)$$

$$= \partial_1 \left[\underbrace{\mathbf{t}_f^1}_{\substack{\text{generalized current} \\ \text{sources}}} - \underbrace{\left(\mathbf{C}_g \dot{\mathbf{a}}_f^1 + \mathbf{R}_g^{-1} \mathbf{a}_f^1 + \mathbf{k}_g^{-1} \mathbf{a}_f^1 + \mathbf{L}_g^{-1} \int \mathbf{a}_f^1 \, dt \right)}_{\substack{\text{equivalent generalized current} \\ \text{sources generated from} \\ \text{generalized voltage sources}}} \right] \tag{4.59}$$

$$\mathbf{k}_t \left(\delta_0 \dot{\mathbf{d}}^0 - \mathbf{a}_f^1 \right) = \mathbf{0} \tag{4.60}$$

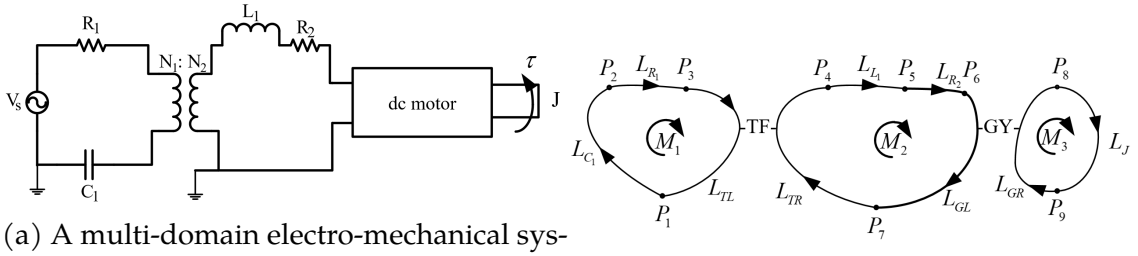
$$\mathbf{k}_t' \left(\mathbf{t}^1 - \mathbf{t}_f^1 \right) = \mathbf{0} \tag{4.61}$$

$$\delta_1 \left(\underbrace{\mathbf{L}_g \partial_2 \ddot{\mathbf{n}}^2}_{\substack{\text{generalized voltages} \\ \text{of generalized L}}} + \underbrace{\mathbf{R}_g \partial_2 \dot{\mathbf{n}}^2}_{\substack{\text{generalized voltages} \\ \text{of generalized R}}} + \underbrace{\mathbf{C}_g^{-1} \partial_2 \mathbf{n}^2}_{\substack{\text{generalized voltages} \\ \text{of generalized C}}} + \underbrace{\mathbf{k}_g \partial_2 \dot{\mathbf{n}}^2}_{\substack{\text{generalized voltages} \\ \text{of gyrators}}} + \underbrace{\mathbf{a}_T^1}_{\substack{\text{generalized voltages} \\ \text{of transformers}}} \right)$$

$$= \delta_1 \left(\underbrace{-\mathbf{a}_f^1}_{\substack{\text{generalized voltages} \\ \text{sources}}} + \underbrace{\mathbf{L}_g \dot{\mathbf{t}}_f^1 + \mathbf{R}_g \mathbf{t}_f^1 + \mathbf{k}_g \mathbf{t}_f^1 + \mathbf{C}_g^{-1} \int \mathbf{t}_f^1 \, dt}_{\substack{\text{equivalent generalized voltage} \\ \text{sources generated from} \\ \text{generalized current sources}}} \right) \tag{4.62}$$

$$\mathbf{k}_t' \left(\partial_2 \dot{\mathbf{n}}^2 - \mathbf{t}_f^1 \right) = \mathbf{0} \tag{4.63}$$

$$\mathbf{k}_t \left(\mathbf{a}^1 - \mathbf{a}_f^1 \right) = \mathbf{0} \tag{4.64}$$



(a) A multi-domain electro-mechanical system

(b) Topological structure

Figure 4.21: An electro-mechanical system and its topological structure

Example 4. *We will use an example of multi-domain electrical-mechanical system in Figure 4.21a to illustrate the derivation of Eq.4.59 and Eq.4.62 . The shown electrical-mechanical system contains two resistors* R_1, R_2, *one capacitor* C_1, *one inductor* L_1, *one voltage source* V_s, *one moment of inertia* J, *one external torque* τ, *one electrical transformer and one DC motor. The transformer between two electrical domains is an ideal electrical transformer, where the ratio of voltage drops* (*currents*) *equals the ratio* (*inverse ratio*) *of the winding numbers* N_1/N_2*; the gyrator between the electrical and the mechanical domain is an ideal DC motor, where the ratio* (*inverse ratio*) *of the voltage drop* (*current*) *and the rotational velocity* (*torque*) *is a constant number* k_g. *Topologically, the system is a 2-cell complex shown in Figure 4.21b, and consisted of nine 0-cells* (P_1,P_2,P_3,P_4,P_5, P_6,P_7,P_8,P_9), *nine 1-cells* (L_{R_1},L_{C_1},L_{TL},L_{TR}, L_{R_2},L_{L_1}, L_{GL},L_{GR},L_J) *and three 2-cells* (M_1,M_2,M_3). *We use symbol -TF- to represent the abstract transformer and a symbol -GY- to represent the abstract gyrator. These two symbols identify the cells where the energy transaction may occur.*

The algebraic topological model of the system contians: primal 0-cochain generalized potential magnetic fluxes ($\mathbf{d}^0 = d_1 \cdot P_1 + d_2 \cdot P_2 + d_3 \cdot P_3 + d_4 \cdot P_4 + d_5 \cdot P_5 + d_6 \cdot P_6 + d_7 \cdot P_7 + d_8 \cdot P_8 + d_9 \cdot P_9$), *primal 0-cochain generalized potentials* ($\mathbf{p}^0 = p_1 \cdot P_1 + p_2 \cdot P_2 + p_3 \cdot P_3 + p_4 \cdot P_4 + p_5 \cdot P_5 + p_6 \cdot P_6 + p_7 \cdot P_7 + p_8 \cdot P_8 + p_9 \cdot P_9$), *primal 1-cochain generalized magnetic fluxes* ($\mathbf{u}^1 = u_1 \cdot L_{R_1} + u_2 \cdot L_{C_1} + u_3 \cdot L_{TL} + u_4 \cdot L_{TR} + u_5 \cdot L_{R_2} + u_6 \cdot L_{L_1} + u_7 \cdot L_{GL} + u_8 \cdot L_{GR} + u_9 \cdot L_J$), *primal 1-cochain generalized voltages* ($\mathbf{a}^1 = a_1 \cdot L_{R_1} + a_2 \cdot L_{C_1} + a_{TL} \cdot L_{TL} + a_{TR} \cdot L_{TR} + a_5 \cdot L_{R_2} + a_6 \cdot L_{L_1} + a_{GL} \cdot L_{GL} + a_{GR} \cdot L_{GR} + a_9 \cdot L_J$), *dual 1-cochain generalized currents* ($\mathbf{t}^1 = t_1 \cdot L_{R_1} + t_2 \cdot L_{C_1} + t_{TL} \cdot L_{TL} + t_{TR} \cdot L_{TR} + t_5 \cdot L_{R_2} + t_6 \cdot L_{L_1} + t_{GL} \cdot L_{GL} + t_{GR} \cdot L_{GR} + t_9 \cdot L_J$), *dual 1-cochain generalized electric charges* ($\mathbf{m}^1 = m_1 \cdot L_{R_1} + m_2 \cdot L_{C_1} + m_{TL} \cdot L_{TL} + m_{TR} \cdot L_{TR} + m_5 \cdot L_{R_2} + m_6 \cdot L_{L_1} + m_{GL} \cdot L_{GL} + m_{GR} \cdot L_{GR} + m_9 \cdot L_J$), *dual 2-cochain generalized*

mesh electric charges $\mathbf{n}^2 = \mathfrak{n}_1 \cdot \mathrm{M}_1 + \mathfrak{n}_2 \cdot \mathrm{M}_2 + \mathfrak{n}_3 \cdot \mathrm{M}_3$*, dual 2-cochain generalized mesh currents* $\mathbf{s}^2 = \mathfrak{s}_1 \cdot \mathrm{M}_1 + \mathfrak{s}_2 \cdot \mathrm{M}_2 + \mathfrak{s}_3 \cdot \mathrm{M}_3$ *and four cochains that are always* $\mathbf{0}$*: 2-cochain of generalized mesh magnetic fluxes, 2-cochain of generalized mesh voltages, 0-cochain of generalized node currents and 0-cochain of generalized node electric charges. In order to obtain unique solution of the state equations, we consider 0-cells* P_1*,* P_7 *and* P_9 *as the reference node. Following the paths shown in Figure 4.20a, generates Eq.4.59 ~ Eq.4.61, with individual terms as follows:*

$$\partial_1 = \begin{bmatrix} -1 & 0 & +1 & 0 & 0 & 0 & 0 & 0 & 0 \\ +1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & +1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & +1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & +1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & +1 & -1 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 & +1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & +1 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & +1 \end{bmatrix} \tag{4.65}$$

$$\delta_0 = \partial_1^{\mathsf{T}} \tag{4.66}$$

$$\mathbf{C}_g = \begin{bmatrix} \mathrm{C}_{g_1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{4.67}$$

$$\mathbf{R}_g^{-1} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & R_{g_1}^{-1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & R_{g_2}^{-1} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{4.68}$$

$$\mathbf{L}_g^{-1} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & L_{g_1}^{-1} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & L_{g_2}^{-1} \end{bmatrix} \tag{4.69}$$

$$\mathbf{t}_T^1 = \begin{bmatrix} 0 & 0 & t_{TL} & t_{TR} & 0 & 0 & 0 & 0 & 0 \end{bmatrix}^T \tag{4.70}$$

$$\mathbf{a}_f^1 = \begin{bmatrix} 0 & -a_{f_1} & 0 & 0 & 0 & 0 & 0 & 0 & -a_{f_2} \end{bmatrix}^T \tag{4.71}$$

$$\mathbf{t}_f^1 = [\mathbf{0}]_{9\times 1} \tag{4.72}$$

$$\mathbf{k}_g^{-1} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & k_g^{-1} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & k_g^{-1} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{4.73}$$

$$\mathbf{k}_t = \begin{bmatrix} 0 & 0 & 1 & -N_1/N_2 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{4.74}$$

$$\mathbf{k'}_t = \begin{bmatrix} 0 & 0 & 1 & -N_2/N_1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{4.75}$$

Substitute Eq.4.65 ~ Eq.4.75 to Eq.4.59 ~ Eq.4.61, the generated system state equation is as follows:

$$
\begin{cases}
\begin{bmatrix}
C_{g_1} & -C_{g_1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-C_{g_1} & C_{g_1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix} \ddot{\mathbf{d}}^0 +
\begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & R_{g_1}^{-1} & -R_{g_1}^{-1} & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & -R_{g_1}^{-1} & R_{g_1}^{-1} & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & R_{g_2}^{-1} & -R_{g_2}^{-1} & -k_g^{-1} & k_g^{-1} \\
0 & 0 & 0 & 0 & 0 & -R_{g_2}^{-1} & R_{g_2}^{-1} & k_g^{-1} & -k_g^{-1} \\
0 & 0 & 0 & 0 & 0 & -k_g^{-1} & k_g^{-1} & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & k_g^{-1} & -k_g^{-1} & 0 & 0
\end{bmatrix} \dot{\mathbf{d}}^0 \\
+ \begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & L_{g_1}^{-1} & -L_{g_1}^{-1} & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & -L_{g_1}^{-1} & L_{g_1}^{-1} & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & L_{g_2}^{-1} & -L_{g_2}^{-1} \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & -L_{g_2}^{-1} & L_{g_2}^{-1}
\end{bmatrix} \mathbf{d}^0 +
\begin{bmatrix}
t_{TL} \\ 0 \\ -t_{TL} \\ t_{TR} \\ 0 \\ 0 \\ -t_{TR} \\ 0 \\ 0
\end{bmatrix} =
\begin{bmatrix}
0 \\ -R_{g_1}^{-1} a_{f_1} \\ R_{g_1}^{-1} a_{f_1} \\ 0 \\ 0 \\ 0 \\ 0 \\ -L_{g_2}^{-1} \int a_{f_2} dt \\ L_{g_2}^{-1} \int a_{f_2} dt
\end{bmatrix} \\
\left(\dot{\mathbf{d}}^0_{(3)} - \dot{\mathbf{d}}^0_{(1)}\right) \Big/ \left(\dot{\mathbf{d}}^0_{(7)} - \dot{\mathbf{d}}^0_{(4)}\right) = N_1/N_2 \\
t_{TL}/t_{TR} = N_2/N_1
\end{cases}
\tag{4.76}
$$

Similarly following the paths in Figure 4.20b, generates Eq.4.62 ~ Eq.4.64, with individual terms as follows:

$$
\mathbf{k}_g = \begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & k_g & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & k_g & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}
\tag{4.77}
$$

$$\mathbf{L}_g = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & L_{g_1} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & L_{g_2} \end{bmatrix} \tag{4.78}$$

$$\mathbf{R}_g = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & R_{g_1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & R_{g_2} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{4.79}$$

$$\mathbf{a}_T^1 = \begin{bmatrix} 0 & 0 & a_{TL} & a_{TR} & 0 & 0 & 0 & 0 & 0 \end{bmatrix}^T \tag{4.80}$$

$$\delta_1 = \begin{bmatrix} +1 & +1 & +1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & +1 & +1 & +1 & +1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & +1 & +1 \end{bmatrix} \tag{4.81}$$

$$\partial_2 = {\delta_1}^T \tag{4.82}$$

$$\mathbf{C}_g^{-1} = \begin{bmatrix} C_{g_1}^{-1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{4.83}$$

Substitute Eq.4.71, Eq.4.72, Eq.4.77 ~ Eq.4.83 to Eq.4.62 ~ Eq.4.64, the generated system state equation is as follows:

$$\begin{cases} \begin{bmatrix} 0 & 0 & 0 \\ 0 & L_{g_1} & 0 \\ 0 & 0 & L_{g_2} \end{bmatrix} \ddot{\mathbf{n}}^2 + \begin{bmatrix} R_{g_1} & 0 & 0 \\ 0 & R_{g_2} & k_g \\ 0 & k_g & 0 \end{bmatrix} \dot{\mathbf{n}}^2 + \begin{bmatrix} C_{g_1}^{-1} & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \mathbf{n}^2 + \begin{bmatrix} a_{TL} \\ a_{TR} \\ 0 \end{bmatrix} = \begin{bmatrix} a_{f_1} \\ 0 \\ a_{f_2} \end{bmatrix} \\ a_{TL}/a_{TL} = N_1/N_2 \\ \dot{\mathbf{n}}_{(1)}^2 \Big/ \dot{\mathbf{n}}_{(2)}^2 = N_2/N_1 \end{cases} \tag{4.84}$$

4.5 Discussion

In this chapter, we proposed unified semantics for lumped parameter systems modeling. The proposed semantics relies only on standard tools from algebraic topology and known results in classification of physical theories and systems. The semantics is effectively "representation free" in that it is independent of specific implementation assumptions, coordinates, linguistic constructs, or numerical simulation schemes.

We first showed that single-domain lumped parameter systems can be represented as single extended Tonti diagrams of physical network systems while multi-domain lumped parameter systems can be represented either as a collection of interacting single-domain Tonti diagrams or a single Tonti diagram of phys-

ical network systems of a generalized physical type where energy transduction is represented by additional constraints. All possible ways of generating governing equations (ODEs/DAEs) by following paths over the diagrams are shown, where LPMs of electrical circuit are used as prototypes. These equations could be either symbolic or numerical depending on either symbolically or numerically interpreting the topological and constitutive operators labeled on the edges of Tonti diagrams.

Paving the way for eliminating the need to build customized point-to-point model consistency analysis tools and achieving greater interoperability of physical modeling tools and languages were the main motivation for developing the proposed canonical semantic lumped parameter model. The existence of common semantics implies that all compliant models become fully interchangeable irrespective of specific syntax or modeling concepts adapted in a particular simulation tool. The representation of this semantics using the corresponding Tonti diagrams can serve as a formal neutral format for exchange of all such models. Furthermore, this representation supports exchange not only of the system models but also of the adapted simulation approaches in individual systems, now represented as paths in the corresponding Tonti diagrams.

Even though lumped parameter systems are intrinsically two-dimensional and devoid of geometry, both classification of physical theories and Tonti diagrams span the entire spectrum of full-dimensional physical models and behaviors. This implies that the proposed formal semantic model can be extended to all spatially distributed models whose behavior is commonly described by integral and partial differential equations. In fact, developing the semantics model for a DPM whose behavior is described by a known Tonti diagram is a straightforward task that would follow the development in Section 4.3. Different paths on such a diagram would correspond to different methods to generate governing equations; replacing symbolic operators by numerical approximations yields rich variety of standard numerical simulation methods as introduced in [79, 41, 44].

5 APPLICATION OF SIMULATION-FREE SCHEME FOR MODEL CONSISTENCY ANALYSIS

In this chapter, the simulation-free scheme is used to determine the consistency between LPMs and DPMs of several linear-time invariant (LTI) mechanical, thermal, and thermo-mehcanical systems, where the correspondences of initial conditions T_I, boundary conditions T_B, and the field of interest T_f are provided. We will first introduce the spatial discretization method to be used to the DPM M_d and the CURE scheme to generate the surrogate LPM M_r of the DPM M_d. Thereafter, implementations of the proposed scheme to several mechanical, thermal, and thermo-mechanical problems will be illustrated.

5.1 Introduction of selected methods and examples

5.1.1 Spatial Discretization

To spatially discretize the DPM M_d, we will use linear-time invariant (LTI) models and Finite Element Analysis (FEA) discretization [14] to illustrate the scheme, but virtually every spatial discretization works as long as it generates a system of LTI ODEs or DAEs. Commonly-used spatial discretization methods can be found in [14, 105, 67, 78, 65, 55, 82]

Most spatial discretization schemes work by tessellating the domain geometry into small pieces that represent the support of the basis/shape functions to approximate a spatio-temporal continuum field (representing distributed physical quantities) by interpolation. The finite basis can be viewed as a projection from the infinite-dimensional state-space of PDEs to the finite-dimensional subspace of ODEs or DAEs, in which the spatio-temporal continuum fields are replaced with a large number of temporal continuum signals (i.e., spatial variations are discretized with integral properties). The number of unknown states is generally tens or hundreds of thousands, if not millions, based on the discretization resolution.

Note that temporal discretization is not required for the sake of MOR procedure thereafter, as the MOR method we will employ to generate the surrogate LPM of the DPM are best described in terms of continuous and differentiable functions in time, which is also true for most existing MOR methods. Temporal discretization comes into play for numerical simulation (if required) of the ODE/DAE system (before or after MOR) by finite time-stepping to approximate integration with given initial conditions, i.e., inversion of time derivative statements. The resolution of the spatial discretization of the DPM M_d will be determined by users. We suppose herein the discretization resolution provided by users is fine enough to approximate the solution of the DPM if it is simulated.

5.1.2 Model Order Reduction

To obtain the surrogate LPM M_r of the spatially-discretized DPM M_d in the simulation-free scheme, the MOR needs to be adapted. There are numerous MOR techniques, reviewed in [22], that can be applied to project the finite-dimensional state-space to a much smaller principal subspace while minimizing the error. In the simulation-free scheme, we use a Krylov subspace method based on CUmulative REduction (CURE) scheme [88], which is a state-of-the-art MOR method that has numerous advantages over other MOR techniques such as balanced truncation (BT) method [32, 99, 109], rational Krylov subspace (RKS) method [9, 10, 48, 59], and iterative rational Krylov algorithm (IRKA) [60] in terms of the numerical efficiency, stability, convergence, a priori error guarantees, and automatic model order(MO) decisions. Figure 5.1 compares the different MOR methods in terms of these five properties. Specifically, the BT method reduces MO by removing states that are both hard to control and observe. To obtain such states, one has to solve full-order Lyapunov equations, which is a very time-consuming process and needs lots of computational effort for large-scale models having more than $O(10^3)$ variables [22], so it is not suitable for reducing the order of the DPM with fine resolution discretization. By comparison, the RKS method aims to match a number of most-significant Taylor series expansion terms (i.e., "moments") of the model's transfer function in the

frequency-domain, obtained from a Laplace transformation of the ODEs or DAEs [10]. This method only uses standard matrix decomposition and multiplication, which can be implemented efficiently in parallel on multi-core CPU or many-core GPU architectures. The RKS method is significantly faster than the BT method and could be used to deal with large-scale numerical models. The IRKA and CURE schemes are developed based on the same basic idea behind the RKS method, and are both numerically efficient. Such development is summarized in the **Appendix**. Notably, IRKA improves the convergence capability of RKS while CURE further ensures stability, a priori error guarantees, and automatic MO decision. We choose CURE for our implementation as it offers a modern approach optimized for the most important properties.

	Balanced Truncation (BT)	Rational Krylov subspace (RKS)	Iterative RK Algorithm	SPARK+CURE
Numerical efficiency	low	high	high	high
Maintain stability	✓	✗	✓	✓
Guarantee convergence	✓	✗	✗	✓
A priori error guarantee	✓	✗	✗	✓
Auto reduction order decision	✗	✗	✗	✓

Figure 5.1: Comparison of properties of four commonly-used MOR methods (CURE is uniquely positioned)

5.1.3 Temporal Integration

In the examples below, to show the computed a priori error bound between the DPM M_d and the surrogate LPM M_r is sufficiently tight, we will compare it to the a posterior error generated from the simulation. Specifically, we will use the

backward Euler method for numerical integration [28] to simulate the temporal response of the DPM M_d and the surrogate LPM M_r. We made this choice as the backward Euler method is usually more time-efficient than higher-order numerical integration methods, e.g., Runge-Kutta [27] and multi-step methods, e.g., Adams-Bashforth [37] and further allows us to pick larger time steps compared to the forward Euler method [28]. Nevertheless, any temporal integration method can be used as long as it is independent of the spatial discretization and MOR.

5.1.4 Mechanical, thermal and thermo-mechanical examples

We illustrate the application of our proposed simulation-free scheme with four geometric objects for DPMs (Figs. 5.2b, 5.15a, 5.15b,and 5.25b) with linear elastic material properties undergoing dynamic mechanical and/or thermal loads. The shapes are arbitrary and the scheme is applicable as long as the geometry can be tessellated (i.e., conforming mesh). The boundary conditions of these DPMs include fixed displacement at some surfaces, uniform pressure at other surfaces, fixed temperature at some surfaces, and heat flux at other surfaces, but can be anything in general. Particularly, we mainly use the hex-rod example to explain the general idea behind our proposed scheme because it has a simple shape and its closed-form response to uniform loading (i.e., the ground truth) is known and intuitively well-understood. We also briefly present our results for the other three examples (bracket, bike frame, and piston) with different boundary conditions. In general, the simulation-free scheme works for arbitrary shapes, material properties, and initial/boundary conditions. For LPMs, we will use simple lumped mass-spring-damper models, lumped thermal resistance-capacitance models, and multi-domain models consisted of these two models associated by transformers/gyrators, whose number of states is no more than ten, but any complex LPM can be used in general.

5.2 Applications

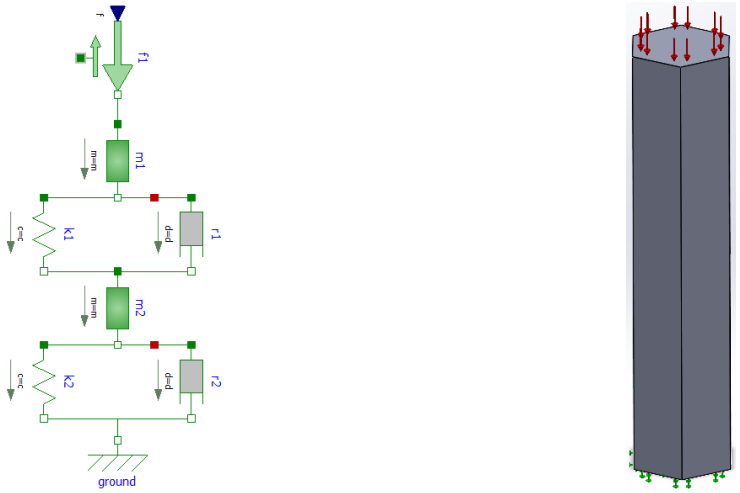
Below, we illustrate the capability of our proposed simulation-free scheme with a few results, where we compute the a priori error bound between the given LPM and DPM. In each example, a LPM M_l, a DPM M_d, and the initial and boundary conditions correspondences T_I, T_B will be given. In each example, our task is to compute the a priori error bound of corresponding behaviors of interest between the given LPM and DPM. Supposing the given correspondences are correct, the error bound is the last "checkpoint" to determine the consistency between two given models. As long as it is smaller than a sensible user-defined threshold, two models would be determined as consistent. Particularly, the results of the hex-rod model will be elaborated below, followed by a brief introduction of the results of other models.

5.2.1 Hex-rod model results

5.2.1.1 Model specifications

In this section, we will show the results of computing the a prior error bound for the correspondence T_f at the physical quantity magnitude level between a given LPM and a given DPM. The LPM is a two degrees of freedom (dof) mass-spring-damper model (Figure 5.2a), where $m_1 = 7.8284 \times 10^8 kg$, $m_2 = 1.1422 \times 10^5 kg$, $k_1 = 1.6448 \times 10^8 N/m$, $k_2 = 1.1996 \times 10^5 N/m$, $r_1 = 9.0984 \times 10^7 Ns/m$, $r_2 = 3.6580 \times 10^4 Ns/m$, and lumped force source $f_1 = 0.026N$. The lower end of k_2 and r_2 is fixed to the ground.

The geometry of the DPM is a hex rod with one end fixed to the ground (Figure 5.2b) and the other end bearing a constant unit pressure $p_0(t) = 1$ (Figure 5.4a). The correspondences of initial/boundary conditions T_I, T_B and fields T_f between the given LPM and DPM is shown in Figure 5.3, where $T_n : f_1(t) = p_0(t)S_1$; the initial velocity of top surface S_1 and middle surface S_{mid} rod is zero, so are the initial velocity of m_1 and m_2, as represented by correspondence T_I; the displacement of the lower end of rod fixed to the ground $u(S_2, t) = 0$ corresponds to (via T_d)



(a) LPM (b) DPM

Figure 5.2: The given DPM and LPM

the displacement of the end of k_2 and r_2 fixed to the ground $w_g(t) = 0$. The field correspondence of interest T_f is between the average displacement $\bar{u}(S_1, t)$ of S_1 and the displacement $w_1(t)$ of m_1. With above given correspondences T_I, T_n and T_d, our goal is to compute the a priori bound between $\bar{u}(S_1, t)$ and $w_1(t)$.

5.2.1.2 Surrogate LPM generation

As mentioned in the general framework in Section 3.2, the discretization resolution of DPM will be determined by the user and as this resolution increases and reaches a certain of fine level, computing the a priori guarantee using our proposed framework would be faster than the behavior comparison using the numerical simulation. We will show this statement in this example.

We discretize the hex rod using second-order tetrahedral finite elements with eight different mesh resolutions (only one sketch is shown in Fig. 5.4b), from which we generate eight groups of governing equations whose number of variables ranges from around 2600 to 15,000. By applying the CURE framework embedded by the SPARK algorithm, we obtain eight families of ROMs for each of the eight cases starting from the lowest model order of $k := 2$ and iteratively increasing the model

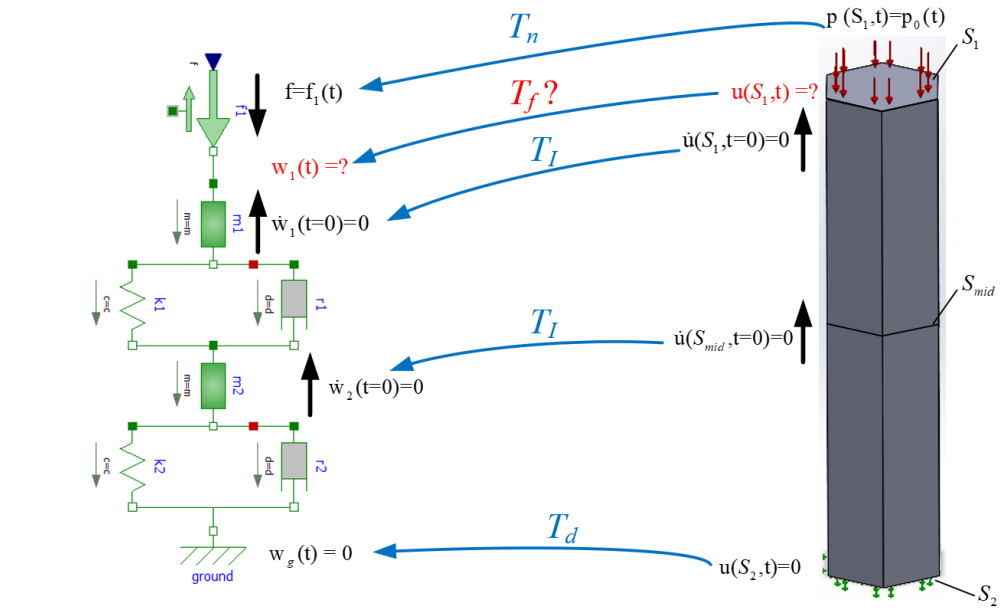
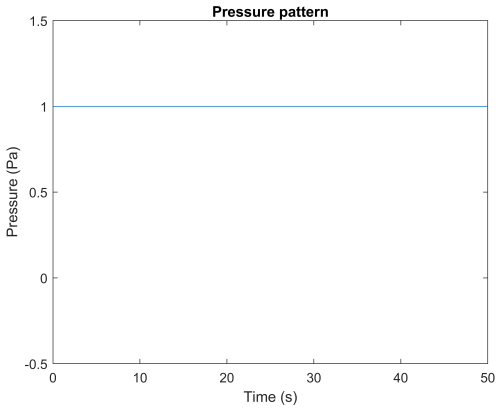
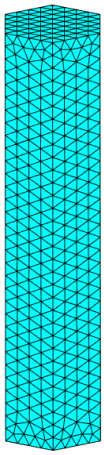


Figure 5.3: Initial, boundary, and field correspondences between LPM and DPM - rod example



(a) Input signal of pressure



(b) Mesh

Figure 5.4: Mesh and input signal of pressure

order until the relative a priori $\mathcal{H}_2$ error is less than 1%. To visualize the quality of the ROMs, we simulate the FOM and all ROMs with the backward Euler method

and the same solver[1], whose simulation results are compared in Figure 5.5, where the abscissa represents the simulation time and the ordinate represents the average displacement of the surface S_1 of the rod. Only a few ROM curves are labeled due to the limited white space. Note that the solver used is built in the Matlab "sparse state space (sss)" toolbox [63], which can be used to analyze dynamical systems with state-space dimensions $O(10^4)$ or higher. Particularly, the solver can preserve the sparsity of large-scale models and take advantage of it for computations (e.g., using sparse LU decompositions) that would otherwise be computationally expensive or even infeasible [30].

In all eight cases in Figure 5.5, as the model order of ROM increases, the simulated response of the ROM gradually approaches that of the FOM. In Figure 5.6, Figure 5.7, and Figure 5.9, we show three different measures of the error caused by the MOR process; namely, the relative $\mathcal{H}_2-$error of the transfer functions, the steady-state error (SSE), and the root-mean-squared error (RMSE). The SPARK+CURE does not provide rigorous a priori bounds on SSE and RMSE, nor does it guarantee a strictly monotonic decay of these two errors by increasing the ROM order in the cumulative reduction process, although an overall reduction pattern is commonly observed.

5.2.1.3 Error Analysis

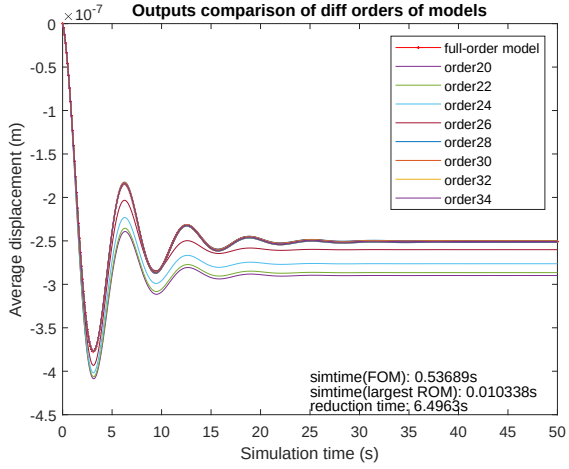
Exact Relative $\mathcal{H}_2-$Error

The exact relative $\mathcal{H}_2-$error is defined by the absolute $\mathcal{H}_2-$norm $\|\mathbf{G}-\mathbf{G}_\mathrm{r}\|_{\mathcal{H}_2}$ divided by the $\mathcal{H}_2-$norm of transfer function of FOM $\|\mathbf{G}\|_{\mathcal{H}_2}$:

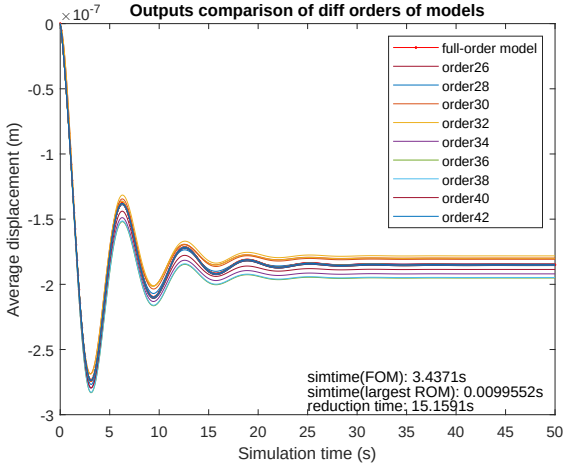
$$\text{Exact relative } \mathcal{H}_2-\text{error } = \frac{\|\mathbf{G}-\mathbf{G}_\mathrm{r}\|_{\mathcal{H}_2}}{\|\mathbf{G}\|_{\mathcal{H}_2}} \tag{5.1}$$

Figure 5.6 shows the exact relative $\mathcal{H}_2-$error for the first two cases, where the abscissa represents the ROM order and the ordinate represents logarithmic error value.

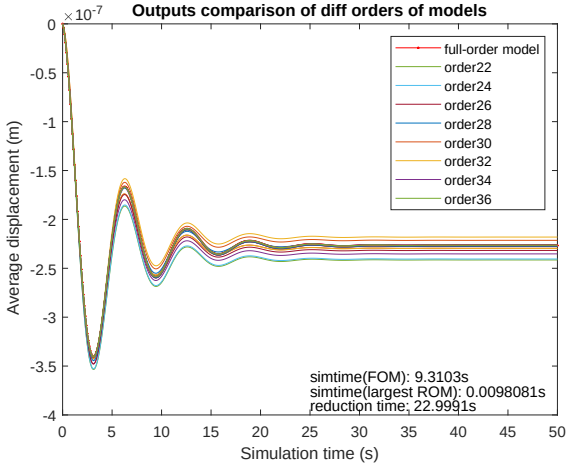
[1]Note that the model comparison does not depend on the simulation results. The simulation is for visualization purposes only



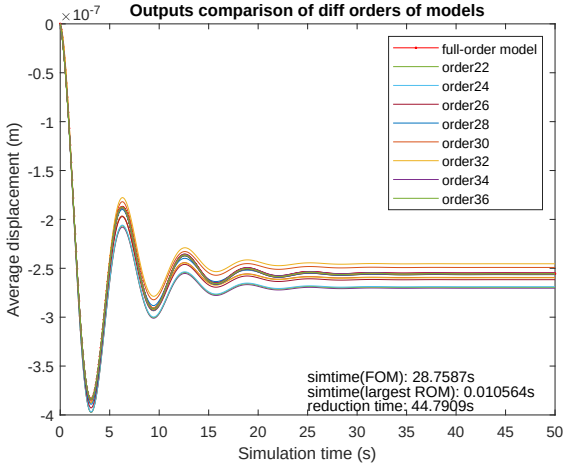
(a) Case 1



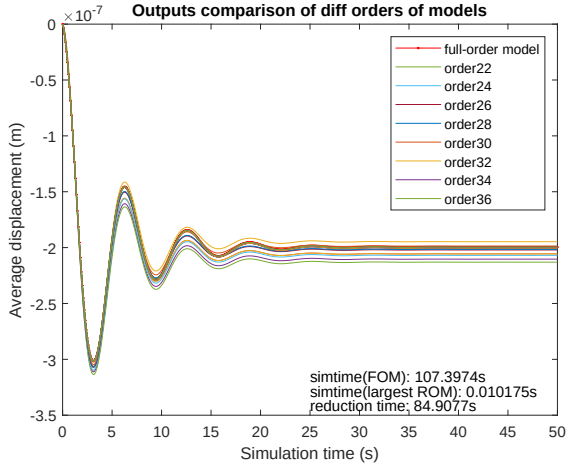
(b) Case 2



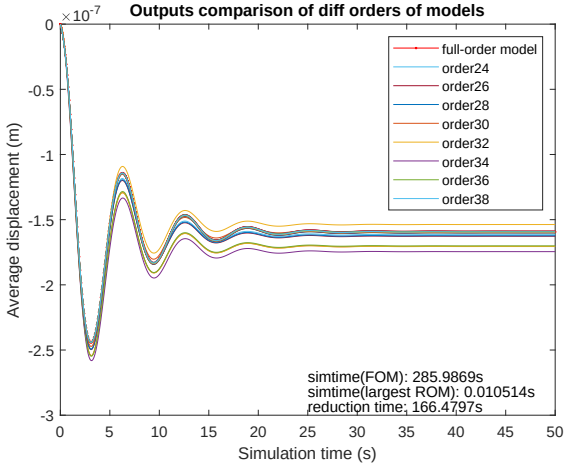
(c) Case 3



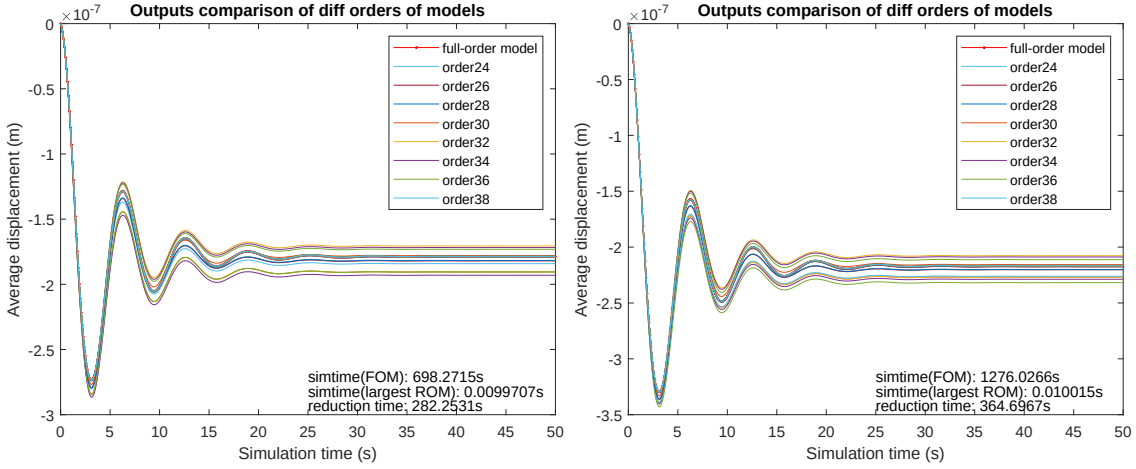
(d) Case 4



(e) Case 5



(f) Case 6



(g) Case 7 (h) Case 8

Figure 5.5: Comparison of the simulation results between FOM and ROM of the hex-rod example

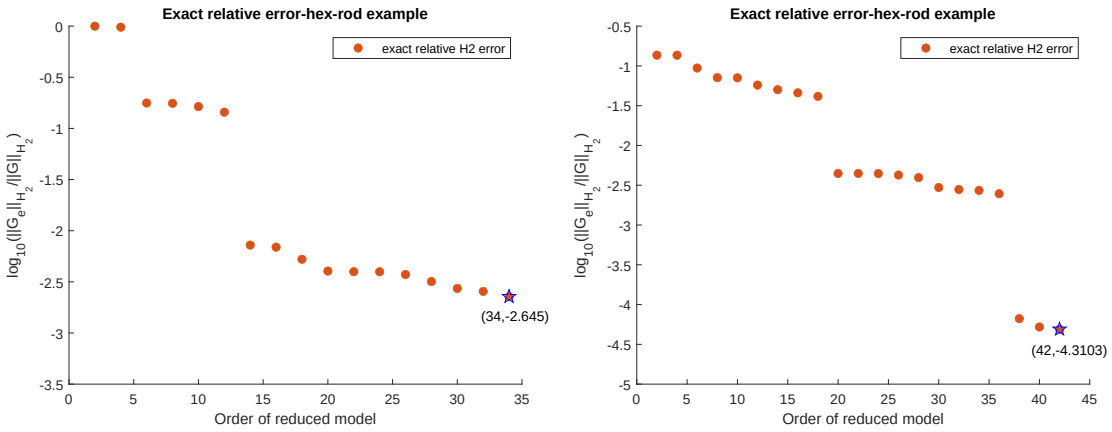
It can be observed that as the ROM order increases, the exact relative $\mathcal{H}_2-$error monotonically decreases. Particularly, as the ROM order reaches $k = 34$ and $k = 42$, the exact relative $\mathcal{H}_2-$errors of the shown first two cases are below $10^{-2.645} \approx 0.226\%$ and $10^{-4.3103} \approx 0.00489\%$, respectively.

Steady-State Error (SSE)

Figure 5.7 shows the steady-state error (ERR), defined by:

$$\text{Steady-State Error } = \lim_{t\to\infty} \frac{\|\mathbf{y}(t) - \mathbf{y}_r(t)\|_2}{\|\mathbf{y}(t)\|_2}, \tag{5.2}$$

where $\|\cdot\|_2$ is the regular (Euclidean) L_2-norm of the output variables, whose asymptotic values as $t \to \infty$ are approximated by their values at a large enough $t_s \gg 0$. It can be seen that all the errors are below 1% when the ROM order is larger than 30. Note that the relationship between the reduced model order and SSE value is not strictly monotonic, unlike the case with the exact relative $\mathcal{H}_2-$error. Although the SPARK+CURE does not come with theoretical guarantees concerning SSE, as



(a) Case 1 (b) Case 2

Figure 5.6: $\mathcal{H}_2$ exact error between FOM and ROM for cases 1 and 2 of the hex-rod example

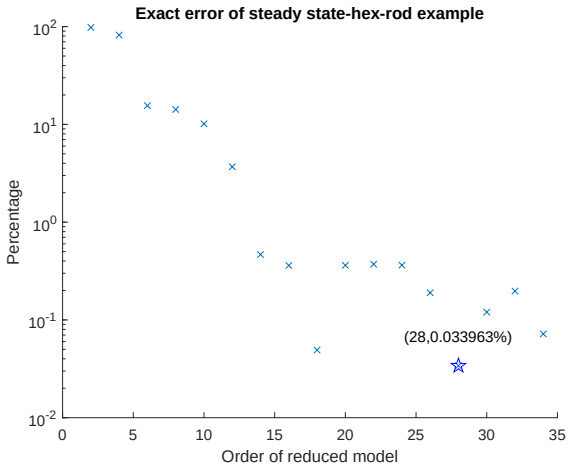
long as the order of ROM is large enough, we can clearly see a downward trend in SSE. As shown in Fig. 5.8a, as we increase the final ROM order from $k = 34$ to 120, a clear overall reduction pattern is observed for the case 1.

Root-Mean-Squared Error (RMSE)

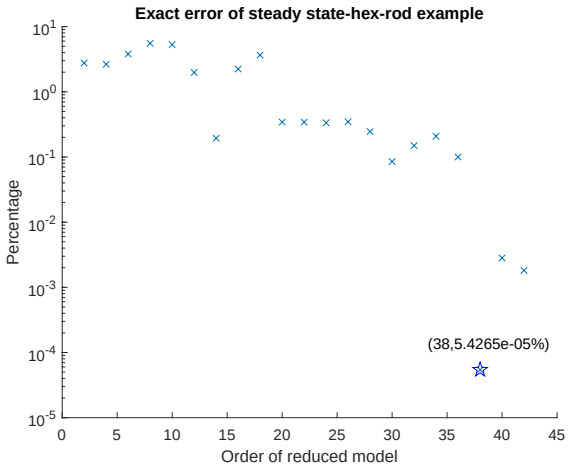
Figure 5.9 shows the root-mean-squared error (RMSE) of the overall output time signal, defined as:

$$\text{Root-Mean-Squarec Error } = \left(\frac{1}{|t_s|} \int_0^{t_s} \|\mathbf{y}(t) - \mathbf{y}_r(t)\|_2 \, dt \right)^{\frac{1}{2}}, \tag{5.3}$$

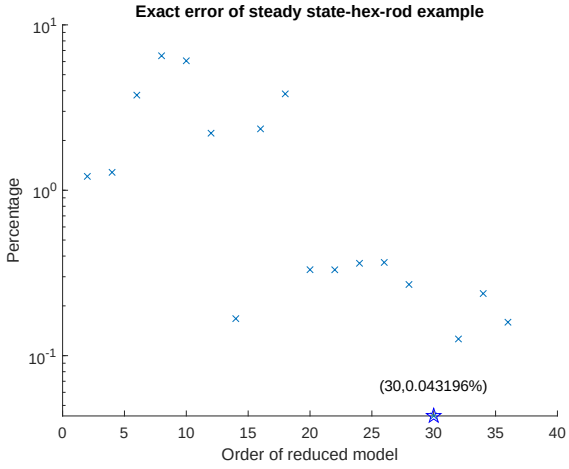
which is approximated by a Riemann sum over a large number of output values sampled along the time axis. Once again, the CURE scheme does not come with RMSE guarantees, although we can see from the results that as long as the ROM order is large enough, the RMSE would be $O(10^{-10}) \sim O(10^{-9})$, which is a few orders of magnitude smaller than the FOM output, which is $O(10^{-7})$. In addition, similar to the SSE, the relation between ROM order and RMSE is not strictly monotonic, but as long as the ROM order is large enough, a clear downward trend can be observed



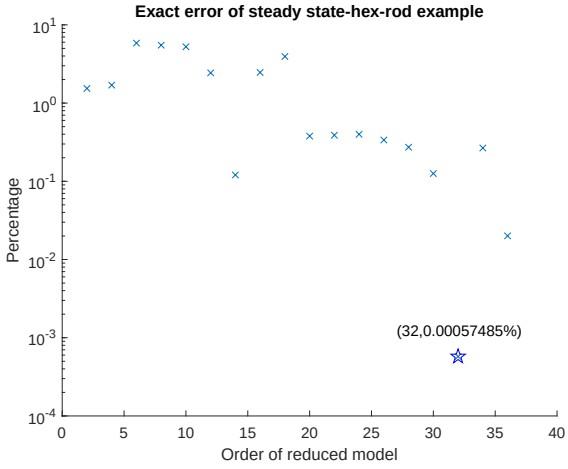
(a) Case 1



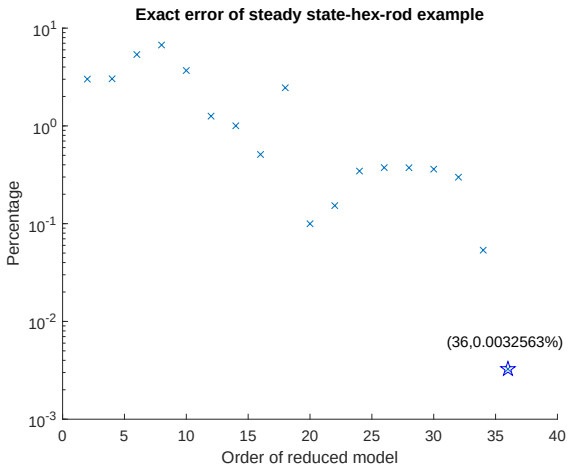
(b) Case 2



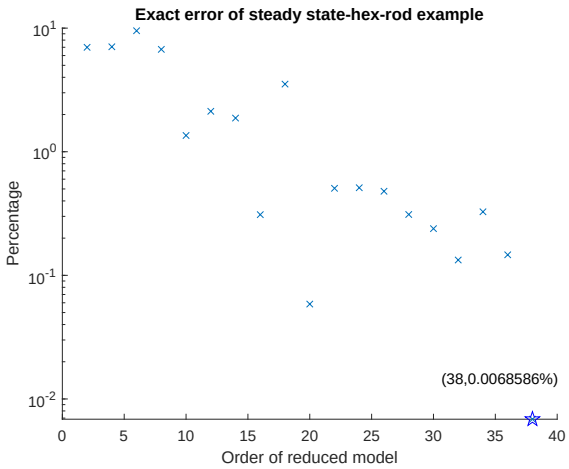
(c) Case 3



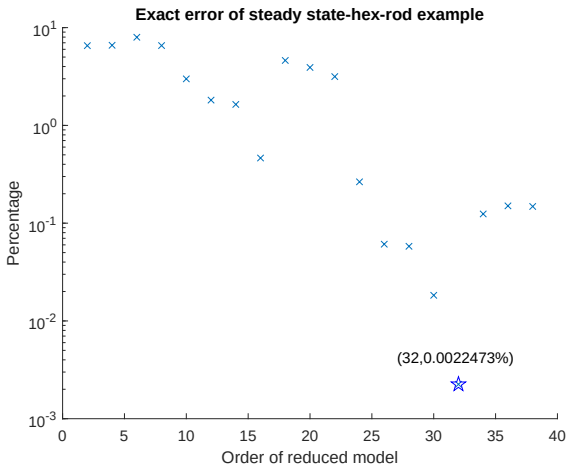
(d) Case 4



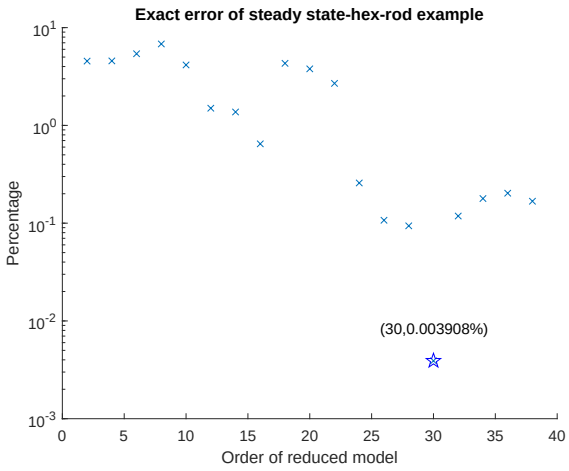
(e) Case 5



(f) Case 6

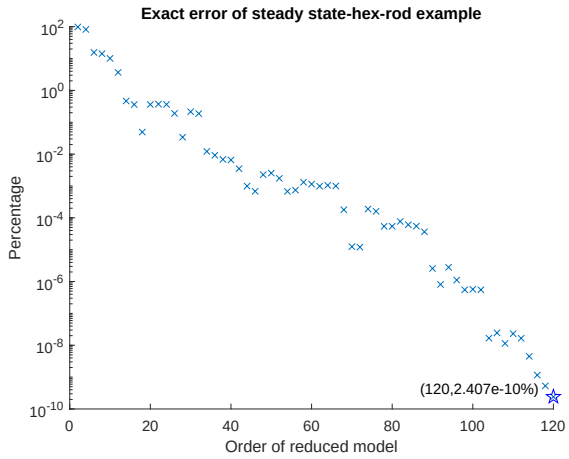


(g) Case 7

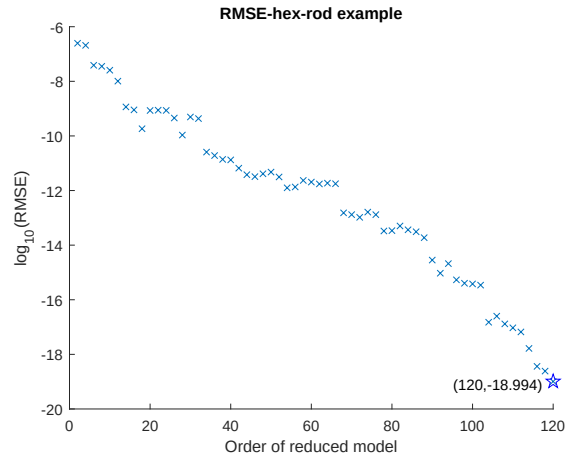


(h) Case 8

Figure 5.7: SSE of the hex-rod example



(a) Downtrend channel of the SSE



(b) Downtrend channel of the RMSE

Figure 5.8: Downtrend channel of the SSE and the RMSE

as shown in Figure 5.8b.

A Priori $\mathcal{H}_2-$Error Bound

A critical property of the CURE scheme is that it provides a priori absolute and relative $\mathcal{H}_2-$error bounds [88]:

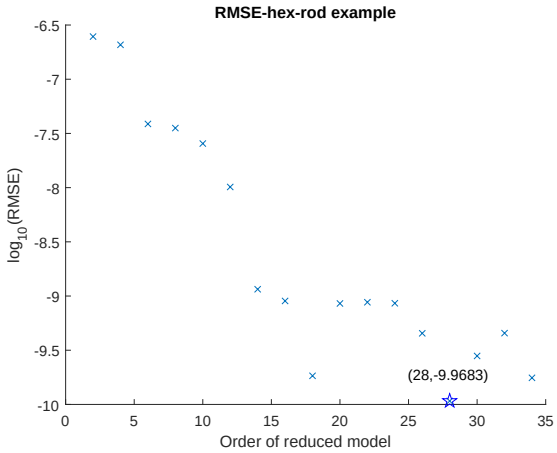
$$\|\mathbf{G}_e(s)\|_{\mathcal{H}_2} \leqslant \|\mathbf{G}_\perp\|_{\mathcal{H}_2} \cdot \left\|\tilde{\mathbf{G}}_r^L(s)\right\|_{\mathcal{H}_\infty} \cdot \left\|\tilde{\mathbf{G}}_r^R(s)\right\|_{\mathcal{H}_\infty} \tag{5.4}$$

$$\frac{\|\mathbf{G}_e(s)\|_{\mathcal{H}_2}}{\|\mathbf{G}(s)\|_{\mathcal{H}_2}} \leqslant \frac{\|\mathbf{G}_\perp\|_{\mathcal{H}_2} \cdot \left\|\tilde{\mathbf{G}}_r^L(s)\right\|_{\mathcal{H}_\infty} \cdot \left\|\tilde{\mathbf{G}}_r^R(s)\right\|_{\mathcal{H}_\infty}}{\|\mathbf{G}_r(s)\|_{\mathcal{H}_2}} \tag{5.5}$$

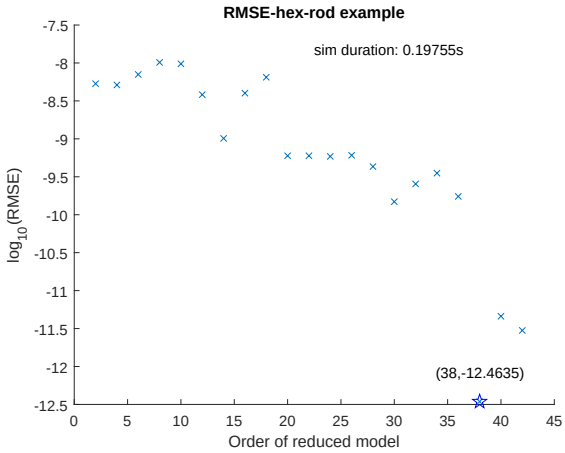
meaning that the exact absolute (Eq.5.4) and relative (Eq.5.5) error between FOM and ROM outputs are guaranteed to stay below a conservative (but tight) threshold, hence there is no need to simulate FOM or ROM to evaluate the loss of accuracy due to MOR. Figure 5.10 shows the a priori relative $\mathcal{H}_2-$error bounds for the aforementioned eight cases. In these plots, the blue dashed line represents the a priori relative $\mathcal{H}_2-$error bound. In the first two plots, the orange dashed line represents the exact relative $\mathcal{H}_2-$error computed from simulations. It can be observed that as the ROM order increases, the exact error monotonically decreases, and so does the error bound and the error bound is always just above the exact error, i.e., it is relatively tight (not overly conservative, hence useful). Importantly, computing these error bounds comes at no extra computational cost, as they are a byproduct of the MOR process. For cases the last six cases out of eight, we only plot the error bounds without comparing to the exact error because the computation of the exact error for these cases is too expensive due to the large order of the FOM. Particularly, similar to the SSE and RMSE, as we increase the final ROM order from k = 34 to 120 for case 1, it can be observed that the error bound is always tight for all the ROMs.

Error analysis between LPM and surrogate LPM

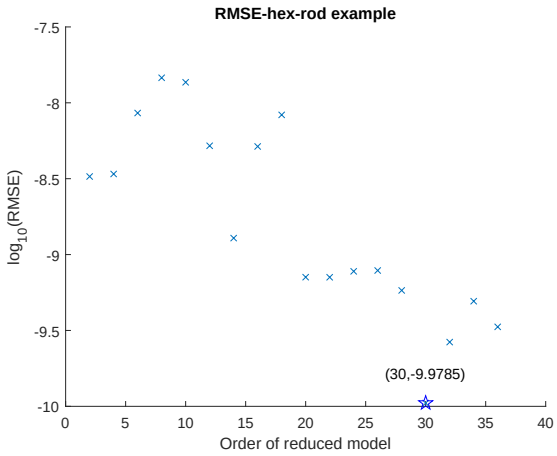
We will select the largest-order ROM obtained from the case 8 as the surrogate LPM of the given DPM because the mesh resolution in this case is the highest of all the eight cases, with around 15000 variables in its governing equation. The order



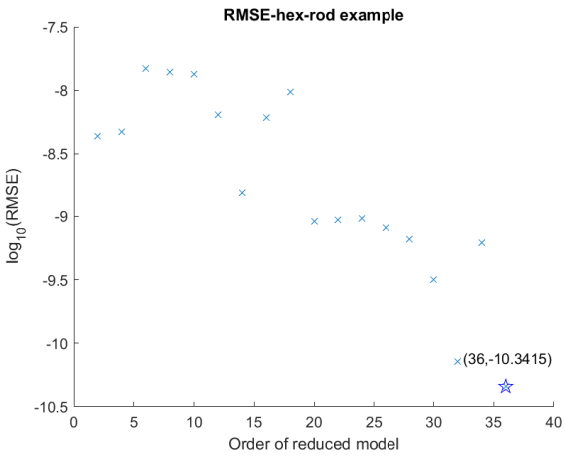
(a) Case 1



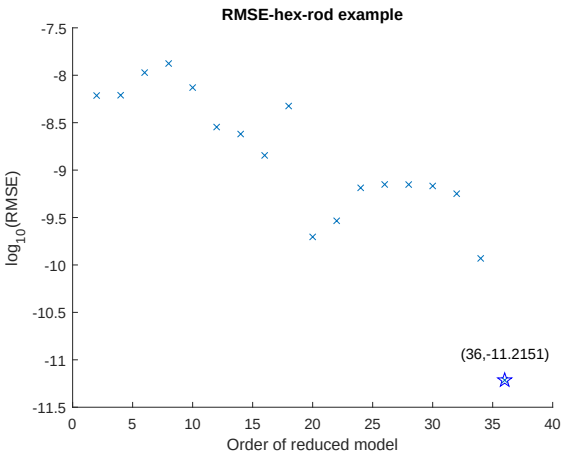
(b) Case 2



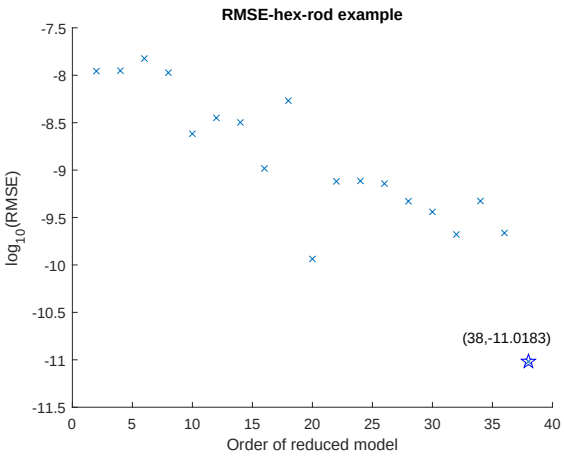
(c) Case 3



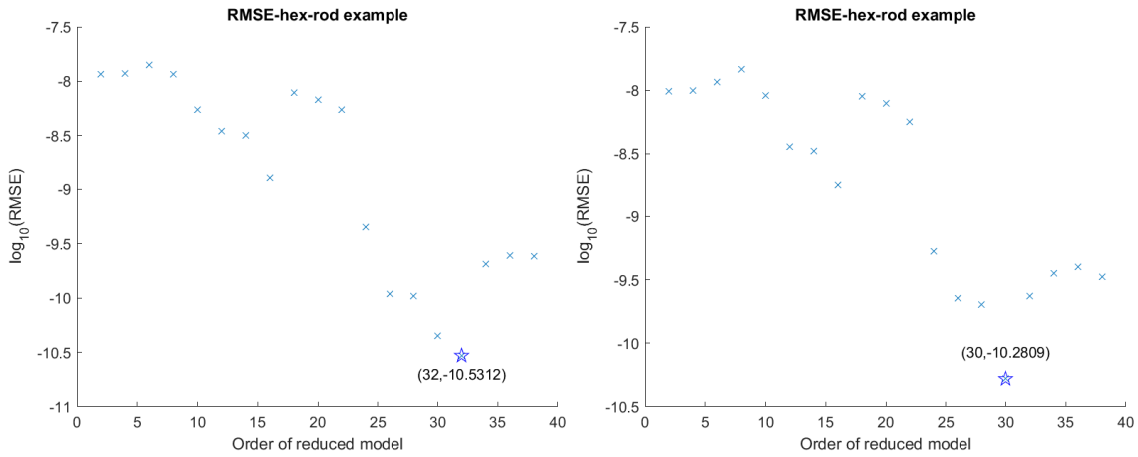
(d) Case 4



(e) Case 5



(f) Case 6



(g) Case 7 (h) Case 8

Figure 5.9: RMSE of the hex-rod example

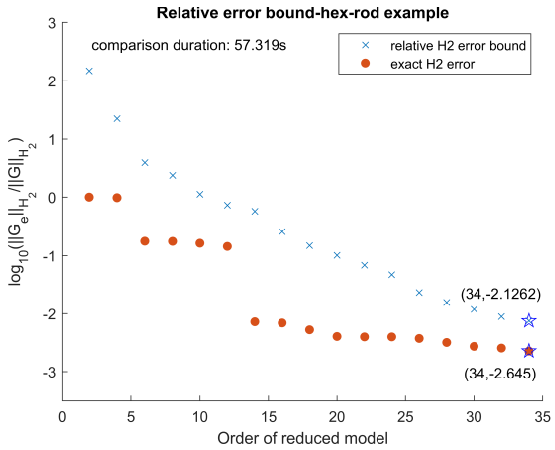
of this ROM is 38, with the relative a priori $\mathcal{H}_2$ error guarantee $10^{-2.152} \approx 0.705\%$ (Figure 5.10h) and a priori $\mathcal{H}_2$ error guarantee 1.9673×10^{-9}, meaning that the difference of transfer functions of the discretized DPM and the surrogate LPM is no more than 1.9673×10^{-9} with respect to the $\mathcal{H}_2$ norm (Eq.5.6). Since the surrogate LPM and the given LPM are both small-scale models, the $\mathcal{H}_2$ error between their transfer functions can be rapidly computed as shown in Eq.5.7.

$$\bar{\varepsilon}_1 = \|\mathbf{G}_\mathrm{d}(s) - \mathbf{G}_\mathrm{r}(s)\|_{\mathcal{H}_2} = 1.9673 \times 10^{-9} \tag{5.6}$$

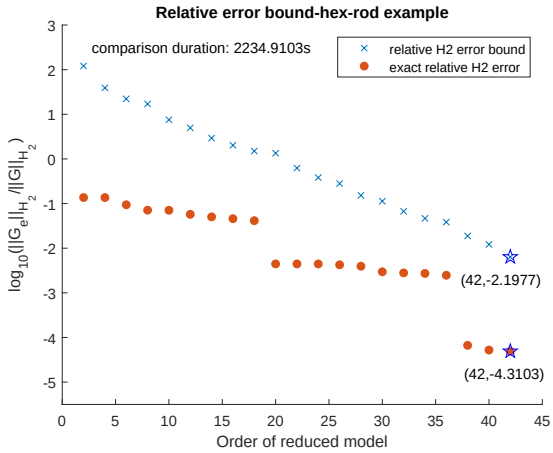
$$\bar{\varepsilon}_2 = \|\mathbf{G}_\mathrm{r}(s) - \mathbf{G}_\mathrm{l}(s)\|_{\mathcal{H}_2} = 1.6058 \times 10^{-9} \tag{5.7}$$

Error analysis between LPM and DPM

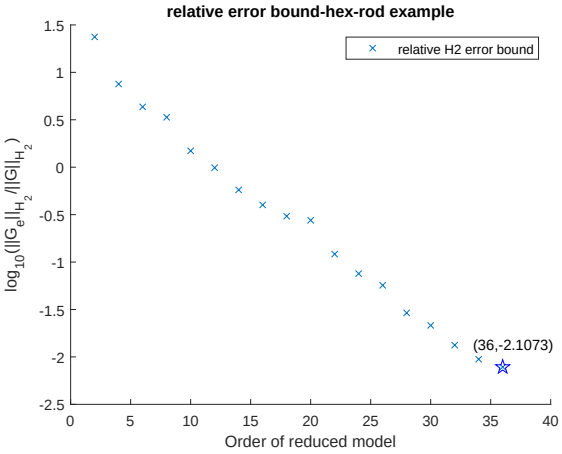
With the obtained a prioir $\mathcal{H}_2$ error guarantee $\bar{\varepsilon}_1$ between the discretized DPM and the surrogate LPM and $\bar{\varepsilon}_2$ between the surrogate LPM and the given LPM, using the Eq.3.19 generates



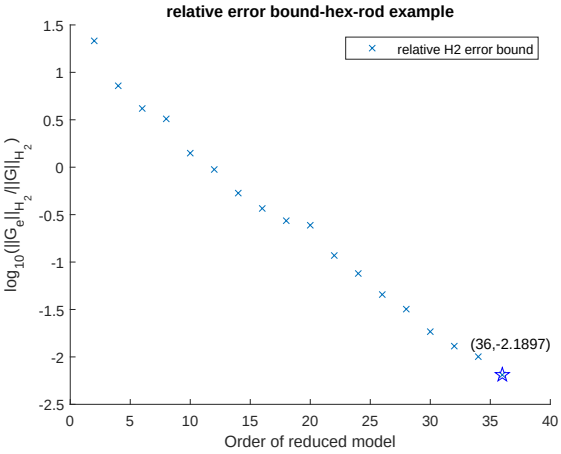
(a) Case 1



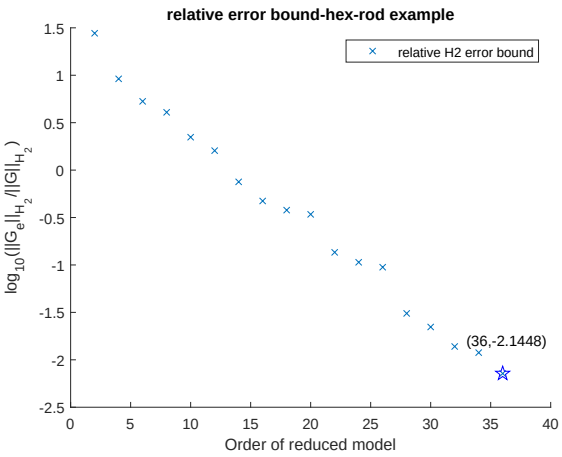
(b) Case 2



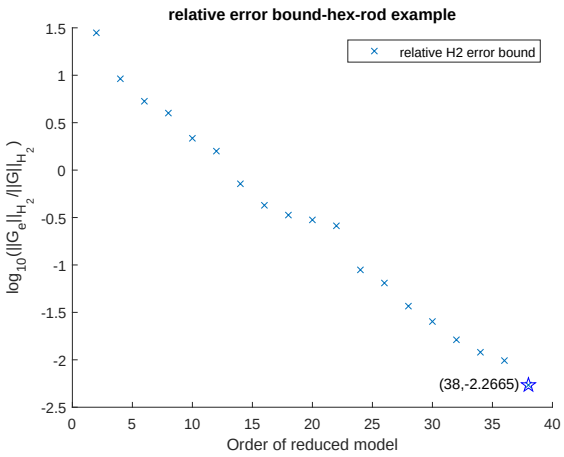
(c) Case 3



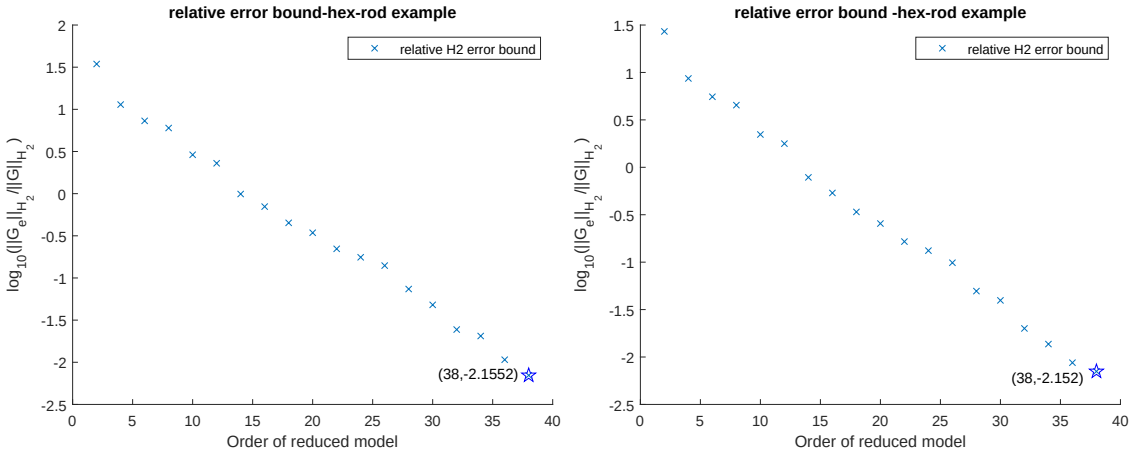
(d) Case 4



(e) Case 5



(f) Case 6



(g) Case 7 (h) Case 8

Figure 5.10: A priori relatively $\mathcal{H}_2$—error bound of the hex-rod example

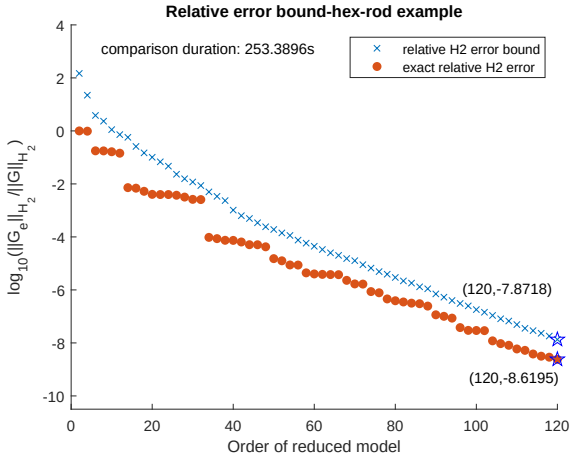
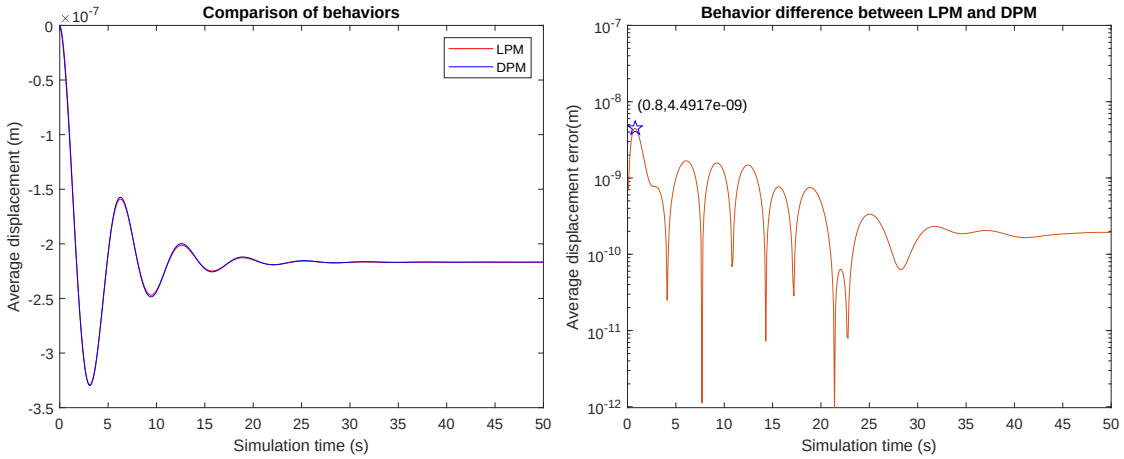


Figure 5.11: A priori relatively $\mathcal{H}_2$—error bound of the hex-rod example - further MOR of case 1

$$\frac{\|\mathbf{G}_d(s) - \mathbf{G}_l(s)\|_{\mathcal{H}_2}}{\|\mathbf{G}_l(s)\|_{\mathcal{H}_2}} \leqslant 0.0129 = \bar{\varepsilon}_{rel} \tag{5.8}$$

If the upper bound value $\bar{\varepsilon}_{rel} = 0.0129$ can be tolerated, then the given LPM



(a) Behavior comparison (b) Behavior difference

Figure 5.12: Behavior comparison between LPM and spatially-discretized DPM - hex-rod example

and DPM are determined to be consistent. For the purpose of visualization, the comparison of the simulation results between the discretized DPM and the given LPM is shown in Figure 5.12a. Figure 5.12b shows the solution difference. It can be seen that the maximum difference between the outputs of interest of these two models over the whole time domain is less than 4.4917×10^{-9}, which is two orders of magnitude smaller than the model output $O(10^{-7})$.

5.2.1.4 Computation time analysis

In addition, we find that if we directly simulate the LPM and the discretized DPM and then compare their solutions, the relationship between the computation time and the order of discretized DPM (i.e., the number of state variables) is a cubic relation, shown by the black curve in Figure 5.13, which is obtained by polynomial regression. However, if we use our proposed simulation-free scheme, such relation would be quadratic, shown by the orange curve in Figure 5.13. Particularly, it is a second-order polynomial with a small coefficient of the second-order term, meaning that the linear property is dominant in the fitted polynomial.

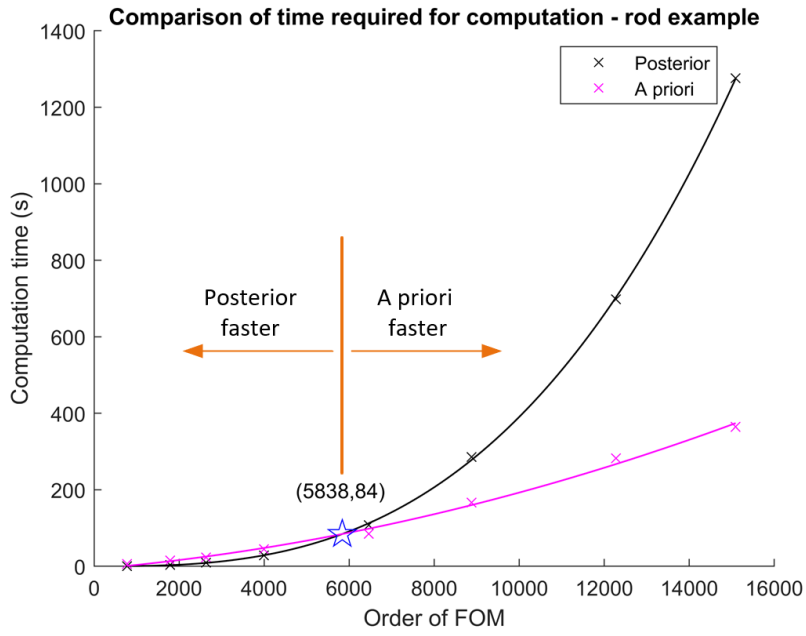


Figure 5.13: Comparison of time required between a posterior and a priori methods

The position of this cross-over point implies that if the order of the discretized DPM is less than 5,838, then solution comparison by simulations has a higher time efficiency; otherwise, our simulation-free scheme will have a higher time efficiency, with at most 1% accuracy loss. Note that if the selected relative $\mathcal{H}_2$—error upper bound is a smaller one (i.e., $< 1\%$), then a new cross-over point (slightly shifted towards larger FOM orders) would be found.

5.2.2 Results of two more complex mechanical examples

5.2.2.1 Model specifications

In addition to the hex-rod example, here we illustrate applying our simulation-free scheme to compute a priori error bound between the given DPM and LPM of two other mechanical systems as shown in Figures 5.14 and 5.15. The two given LPMs are the same two-dof mass-spring-damper model as the one used in the hex rod example (copied below in Figure 5.14), but with different parameters. Specifically,

in the bracket example, $m_1 = 3.8465 \times 10^5 kg$, $m_2 = 3.512 \times 10^3 kg$, $k_1 = 3.316 \times 10^4 N/m$, $k_2 = 4.688 \times 10^3 N/m$, $r_1 = 1.4697 \times 10^5 Ns/m$, $r_2 = 2.9052 \times 10^3 Ns/m$, and $f_1 = 0.005N$; in the bike frame example, $m_1 = 7.997 \times 10^5 kg$, $m_2 = 6.9139 \times 10^4 kg$, $k_1 = 4.8561 \times 10^8 N/m$, $k_2 = 2.2308 \times 10^8 N/m$, $r_1 = 2.8102 \times 10^7 Ns/m$, $r_2 = 1.5075 \times 10^5 Ns/m$, and $f_1 = 2186.56N$.

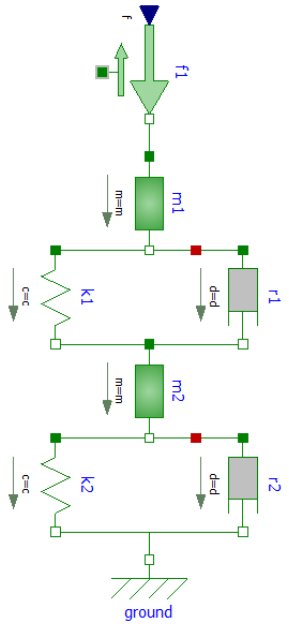
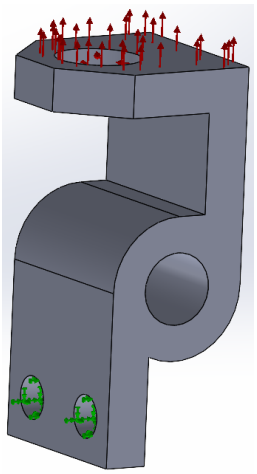


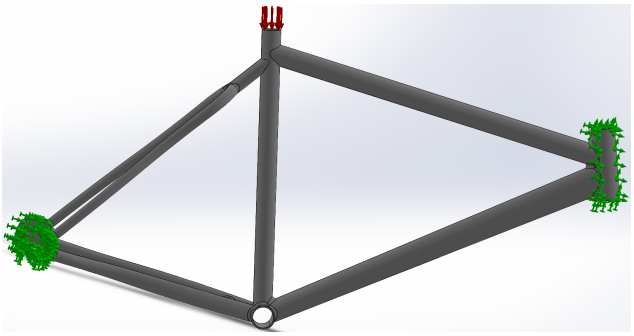
Figure 5.14: Given LPM

The two given DPMs are two other geometric objects under dynamic mechanical loads. The first one is a bracket model (Figure 5.15a) with two holes fixed while the top surface bears a square wave pressure (Figure 5.17a). The second one is a bike frame model (Figure 5.15b) with its three tube ends fixed while the top of the seat tube bears a pressure that combines square and sine wave signal (Figure 5.17b).

The correspondences of initial and boundary conditions T_I, T_B, and field of interest T_f between the given LPM and DPM of these two examples are respectively shown in Figure 5.18 and 5.19, where $T_n : f_1(t) = p_0(t)S_1$; the initial velocity of S_1 and S_{mid} is zero, the same as the initial velocity of m_1 and m_2, respectively; the displacements $u(S_2, t) = 0$ and $w_g(t) = 0$ are both zero. Our goal is to compute the a

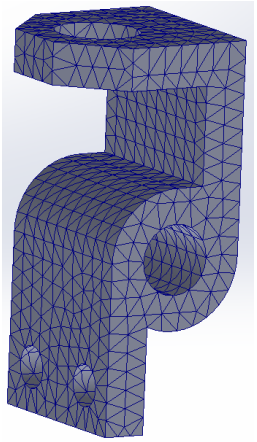


(a) DPM of bracket

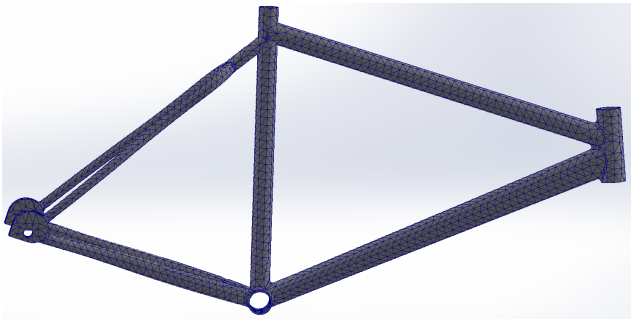


(b) DPM of bike frame

Figure 5.15: Two given DPMs



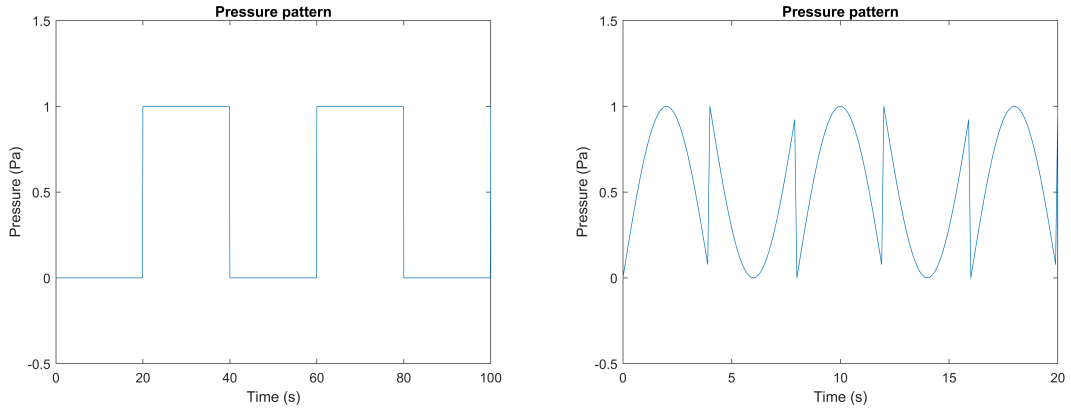
(a) Mesh of bracket



(b) Mesh of bike frame

Figure 5.16: Meshes of bracket and bike frame

priori error bound between the average displacement $\bar{u}(S_1, t)$ and the displacement $w_1(t)$ associated by the field correspondence T_f.



(a) Pressure of bracket (b) Pressure of bike frame

Figure 5.17: Pressures of bracket and bike frame

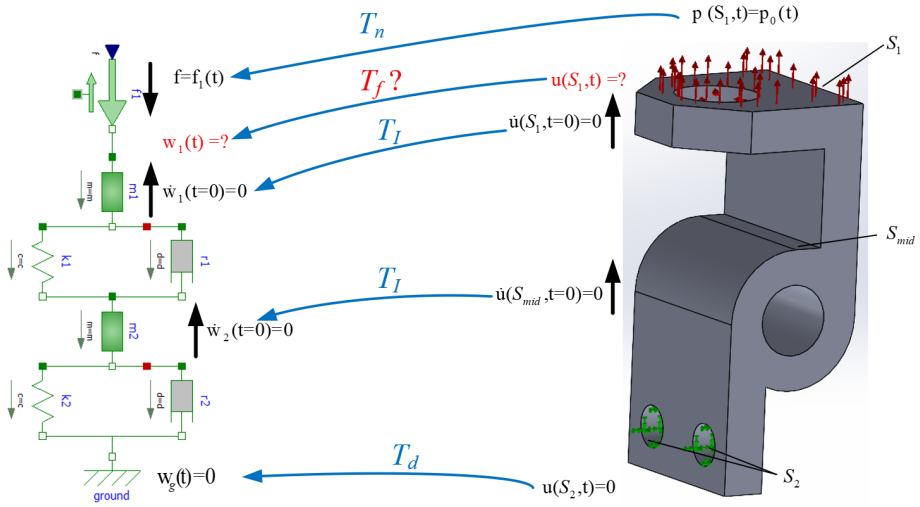


Figure 5.18: Initial, boundary, and field correspondences between LPM and DPM - bracket example

5.2.2.2 Surrogate LPM generation and error analysis

Similar to the hex-rod example, we discretize the geometry of the bracket using tetrahedral finite elements with three different resolutions (only one sketch for each example is shown in Figure 5.17), the geometry of the bike frame with one resolution, and then use SPARK+CURE to generate a family of ROMs for each

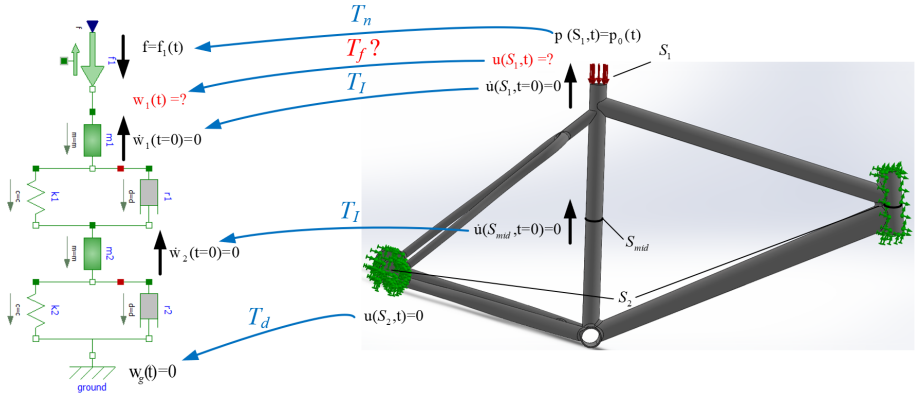
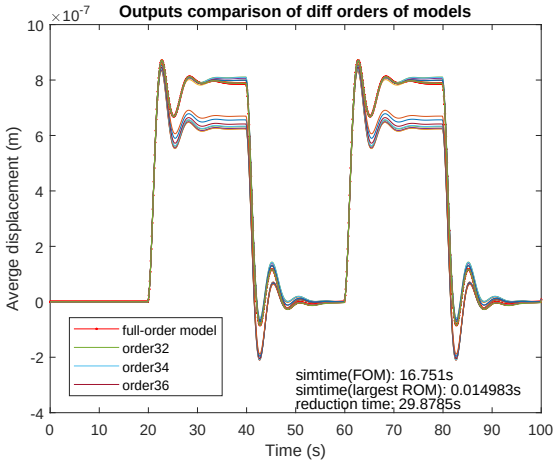


Figure 5.19: Initial, boundary, and field correspondences between LPM and DPM - bike frame example

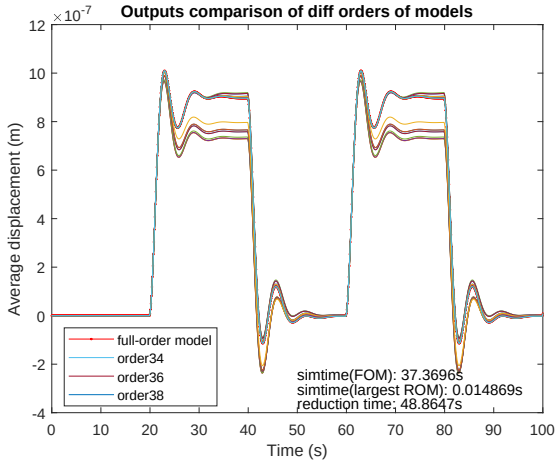
example and stop the model reduction when the a priori relative $\mathcal{H}_2$ error reaches just below 1%. For each example, we simulate the FOM and all the generated ROM and compare their results, as shown in Figure 5.20 and Figure 5.22a. As expected, we find the cross-over point for the bracket example (Figure 5.20d), meaning that if the FOM order is larger than 4637, using our proposed simulation-free scheme for the solution comparison would be more time-efficient.

Since there is no obvious steady state after the non-zero pressure acts, we only computed the RMSE and the relative $\mathcal{H}_2$—error bounds for the bracket example, as shown in Figure 5.21. The RMSE is around $O(10^{-9})$, which is two orders of magnitude smaller than the FOM output of $O(10^{-7})$. The a priori $\mathcal{H}_2$— error bound results show that if the ROM order is larger than 36, 38, and 34 in the three cases (of different resolutions), their exact relative $\mathcal{H}_2$—error would be less than $10^{-2.1506} \approx 0.707\%$, $10^{-2.1859} \approx 0.652\%$ and $10^{-2.1797} \approx 0.661\%$, respectively. In addition, we computed the relative $\mathcal{H}_2$—error bounds for the bike frame example. The result is given in Fig.5.22b. The a priori $\mathcal{H}_2$—error bound results show if the ROM order is larger than 236, then the exact relative $\mathcal{H}_2$—error would be less than $10^{-2.0168} \approx 0.962\%$.

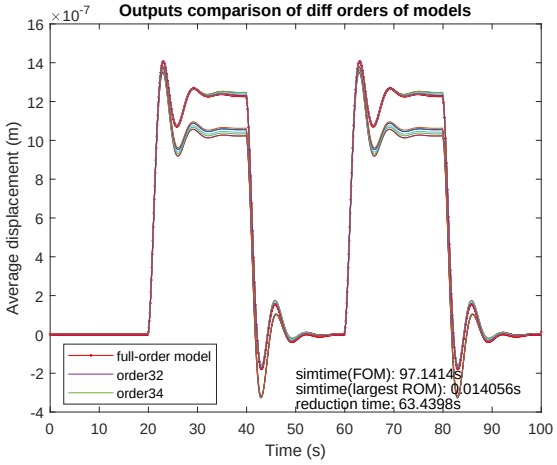
In the bracket example, we select the ROM of order 34 generated from case 3 as the surrogate LPM of the discretized DPM. Its corresponding a priori relative



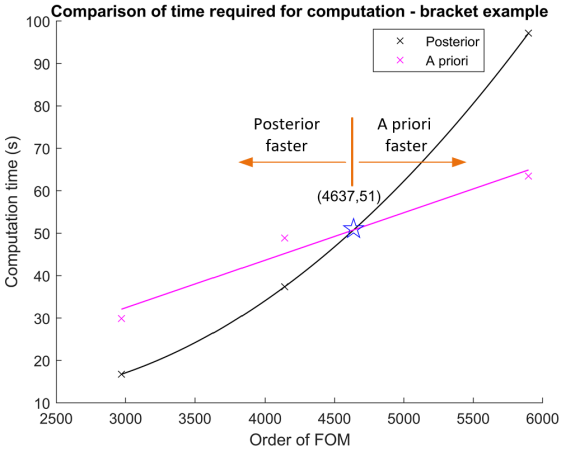
(a) Case 1



(b) Case 2

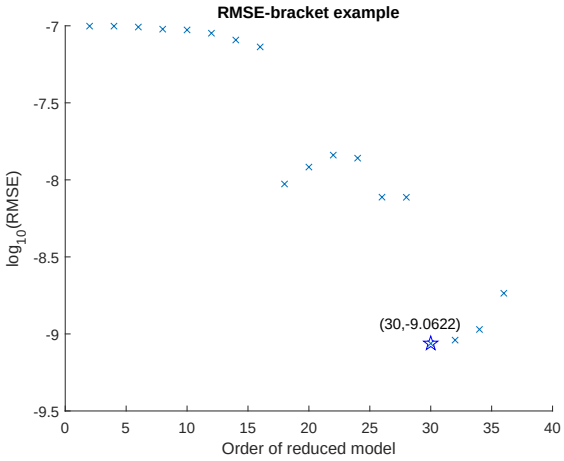


(c) Case 3

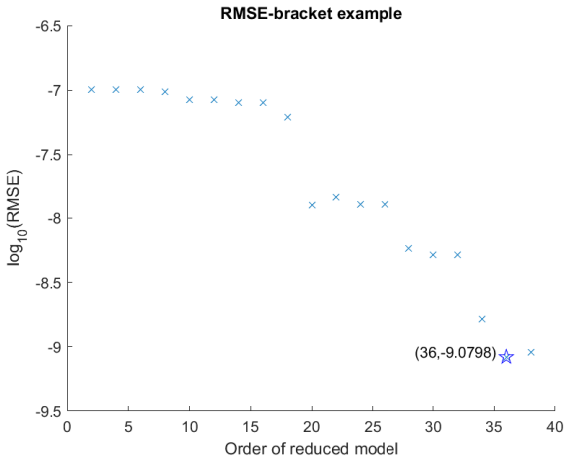


(d) Computation time comparison

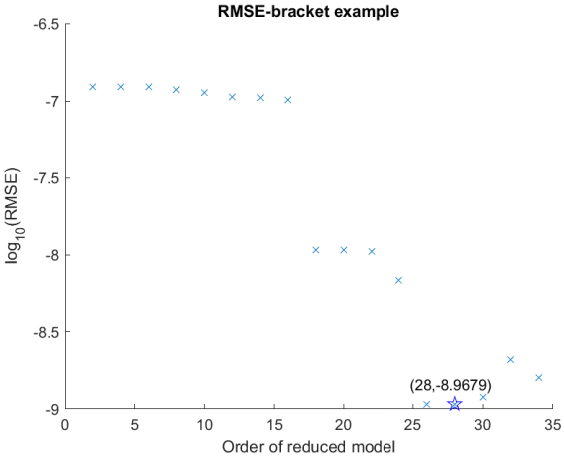
Figure 5.20: Comparison of the simulation results and computation time between FOM and ROM of the bracket example



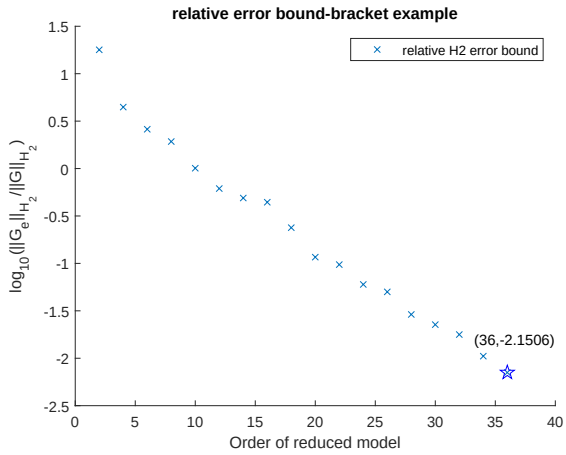
(a) RMSE of Case 1



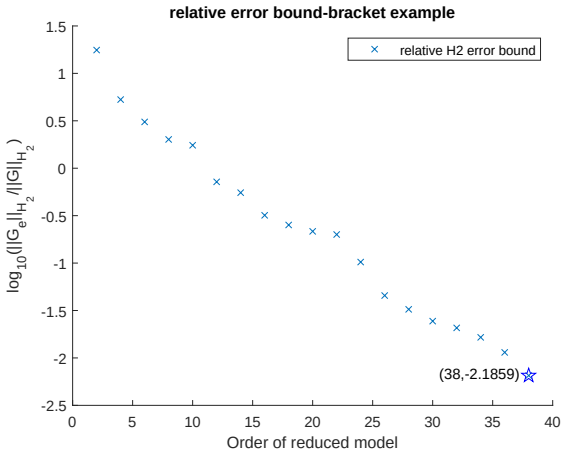
(b) RMSE of Case 2



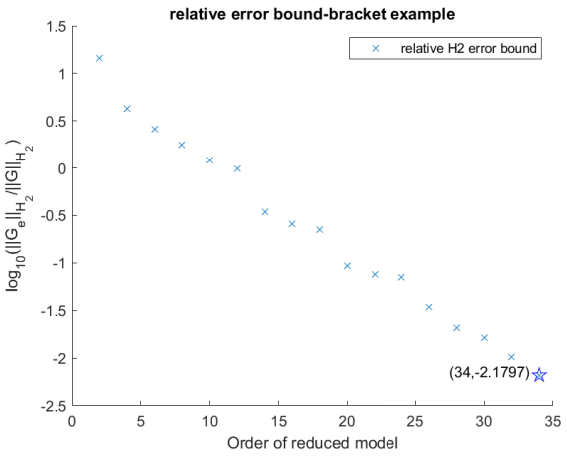
(c) RMSE of Case 3



(d) Error bound of Case 1

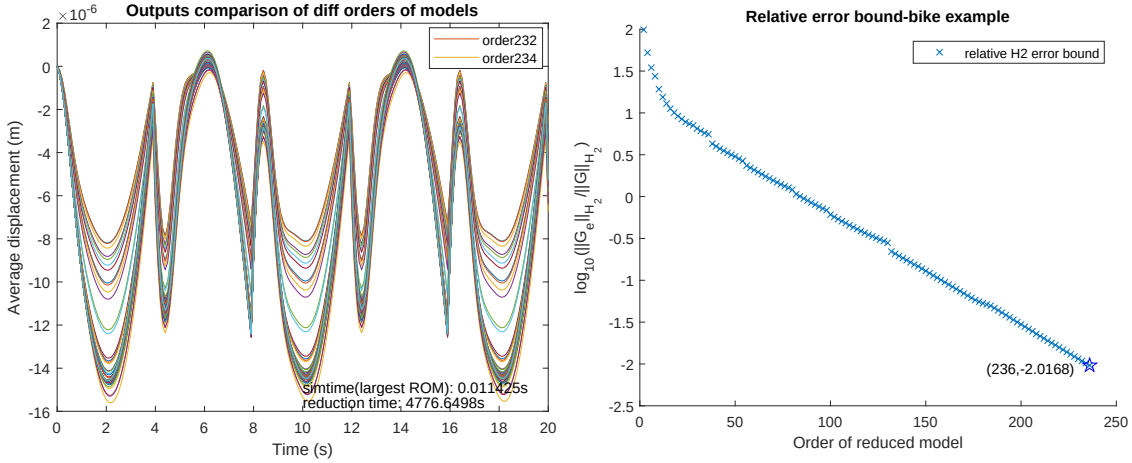


(e) Error bound of Case 2



(f) Error bound of Case 3

Figure 5.21: RMSE and a priori relative $\mathcal{H}_2$ error bound of the bracket example



(a) Comparison of the simulation results

(b) A priori relative $\mathcal{H}_2$ error bound

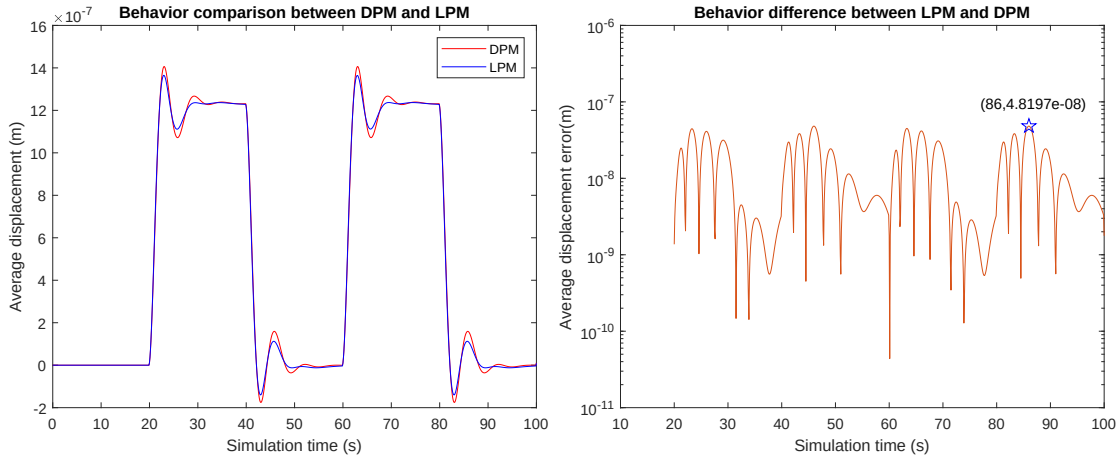
Figure 5.22: Comparison of the simulation results between FOM and ROM of the bike frame and the a priori relative $\mathcal{H}_2$ error bound

$\mathcal{H}_2$-error bound is $10^{-2.1797} \approx 0.661\%$, with which we compute the a priori relative $\mathcal{H}_2$-error bound between the discretized DPM and the given LPM and the result is as follows:

$$\begin{aligned} \frac{\|\mathbf{G}_d(s) - \mathbf{G}_l(s)\|_{\mathcal{H}_2}}{\|\mathbf{G}_l(s)\|_{\mathcal{H}_2}} &\leqslant \frac{\bar{\varepsilon}_1 + \bar{\varepsilon}_2}{\|\mathbf{G}_l(s)\|_{\mathcal{H}_2}} \\ &= 0.0069 + 0.0394 = 0.0463 = \bar{\varepsilon}_{rel} \end{aligned} \tag{5.9}$$

If the user-defined threshold for the relative error is larger than 0.0463, then the LPM and the DPM are determined to be consistent. The comparison of the numerical solutions between discretized DPM and given LPM is shown in Figure 5.23a. Figure 5.23b shows the difference between their numerical solutions. It can be seen that the maximum difference between the outputs of two models over the whole time domain is less than 4.8197×10^{-8}, which is one order of magnitude smaller than the model maximum output $O(10^{-6})$.

In the bike frame example, the largest order of the obtained ROMs is 236, whose a priori relative $\mathcal{H}_2$-error bound is $10^{-2.0168} \approx 0.962\%$ as shown in Figure 5.21f. With



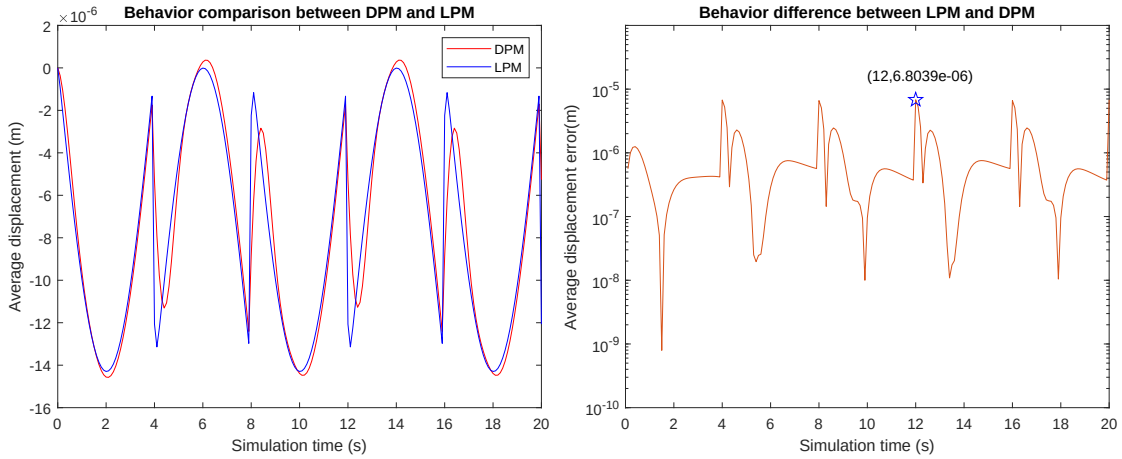
(a) Behavior comparison (b) Behavior difference

Figure 5.23: Behavior comparison between LPM and spatially-discretized DPM - bracket example

this number, the a prioir $\mathcal{H}_2$ relative error error between the discretized DPM and the LPM can be computed as follows:

$$\frac{\|\mathbf{G}_d(s) - \mathbf{G}_l(s)\|_{\mathcal{H}_2}}{\|\mathbf{G}_l(s)\|_{\mathcal{H}_2}} \leqslant \frac{\bar{\varepsilon}_1 + \bar{\varepsilon}_2}{\|\mathbf{G}_l(s)\|_{\mathcal{H}_2}} \tag{5.10}$$
$$= 0.0017 + 0.8212 = 0.8229 = \bar{\varepsilon}_{rel}$$

If the user-defined threshold for the relative error is larger than 0.8229, then the given LPM and DPM are consistent. The simulation results of the discretized DPM and the given LPM are shown in Figure 5.24a, with the difference of simulation results shown in Figure 5.24b. The maximum difference over the whole time domain is less than 6.8039×10^{-6}, which is almost half of the model maximum output value.



(a) Behavior comparison (b) Behavior difference

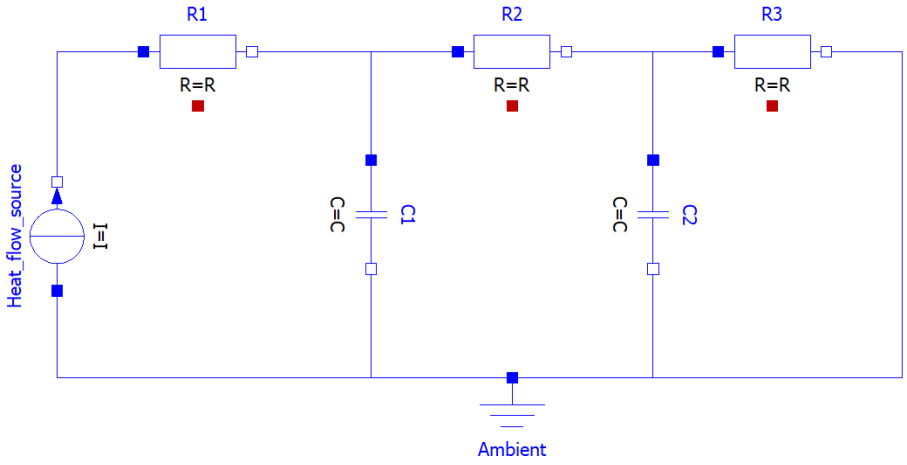
Figure 5.24: Behavior comparison between LPM and semi-discretized DPM - bike frame example

5.2.3 Results of thermal examples

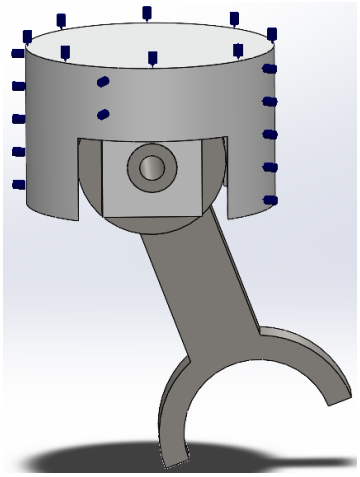
5.2.3.1 Model specifications

In addition to the above three mechanical examples, here we use our proposed scheme to compute a priori error bound between a given DPM and a given LPM of a thermal system as shown in Figure 5.25. The given LPM for the thermal problem shown in Figure 5.25a is a lumped thermal model consisted of three thermal resistors $r_1 = 3.3836 \times 10^3$ (K/W), $r_2 = 1.6148 \times 10^3$ (K/W), and $r_3 = 5.5828 \times 10^6$ (K/W), two thermal capacitors $c_1 = 2.0957 \times 10^7$ (Ws/K and $c_2 = 4.0062 \times 10^5$ (Ws/K)), and a heat source $I(t) = 1.782 \times 10^8$ (W). The DPM of the thermal problem is a piston under a thermal load (Figure 5.25b), where a constant heat flux is added to the top surface of the piston crown, which is represented by small black pins attached to the crown in the figure.

The correspondences of initial conditions T_I, boundary conditions T_B, and field of interest T_f between the given LPM and DPM is shown in Figure 5.26, where $T_n : I(t) = h_0(t)S_1$, which is the relation between the heat flux $h_0(t)$ added to S_1



(a) LPM of piston-thermal



(b) DPM of piston-thermal

Figure 5.25: LPM and DPM of thermal problems

and the lumped heat source $I(t)$; the initial temperatures of surfaces S_1, S_2, and S_3 are zero, so are the initial temperature of each thermal lumped component, as described by T_I; The ambient temperature of two models are both zero, that is $T_d : T_{ambient}(t) = \phi(t) = 0$. The field correspondence T_f is between the average temperature $\bar{T}$ of the surface S_1 and the temperature T_1 associated to the left port of the thermal resistor r_1. With these correspondences, our goal is to compute the a priori error bound between $\bar{T}$ and T_1.

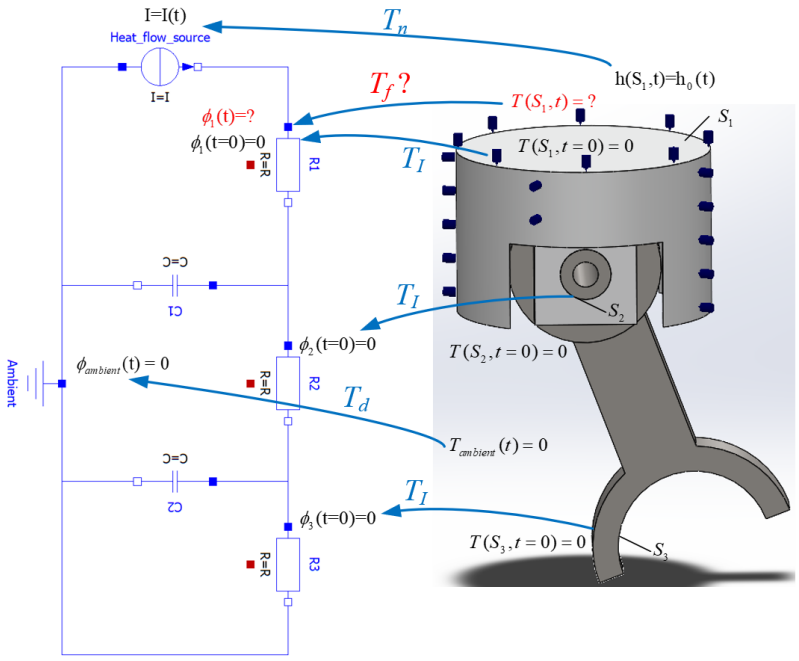


Figure 5.26: Initial, boundary, and field correspondences between LPM and DPM - thermal piston example

5.2.3.2 Surrogate LPM generation and error analysis

We discretize the piston using tetrahedral finite elements with a user-defined resolution (as shown in Figure 5.27), use SPARK+CURE to generate a family of ROMs, and stop the MOR when the a priori relative $\mathcal{H}_2$-error reaches just below 1%. We simulate both FOM and ROMs and compare their results, as shown in Figure 5.28a.

We compute the relative $\mathcal{H}_2$—error bounds for each order (Figure 5.28b). The a priori $\mathcal{H}_2$—error bound results show if the ROM order is larger than 24, the exact relative $\mathcal{H}_2$—error will be less than $10^{-2.0946} \approx 0.804\%$. By using the values of this error bound, $\|\mathbf{G}_l(s)\|_{\mathcal{H}_2}$, and $\|\mathbf{G}_d(s) - \mathbf{G}_l(s)\|_{\mathcal{H}_2}$, we compute the a priori relative error bound between the discretized DPM and the LPM. The result is as below:

$$\begin{aligned} \frac{\|\mathbf{G}_d(s) - \mathbf{G}_l(s)\|_{\mathcal{H}_2}}{\|\mathbf{G}_l(s)\|_{\mathcal{H}_2}} &\leqslant \frac{\bar{\varepsilon}_1 + \bar{\varepsilon}_2}{\|\mathbf{G}_l(s)\|_{\mathcal{H}_2}} \\ &= 0.0072 + 0.0987 = 0.1059 = \bar{\varepsilon}_{rel} \end{aligned} \tag{5.11}$$

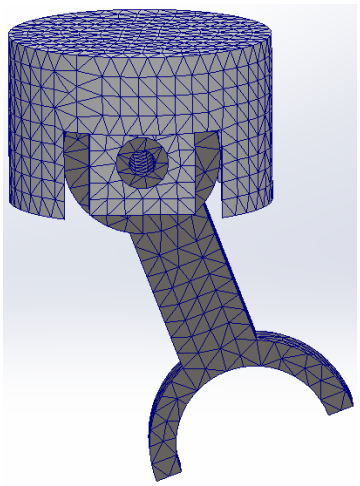


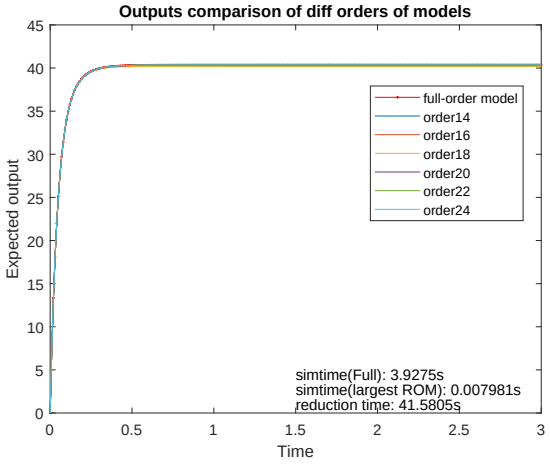
Figure 5.27: Mesh of piston

If $\bar{\varepsilon}_{rel}$ is less than the user-defined threshold, then the given LPM and the given DPM are consistent. Figure 5.29a shows the comparison of the numerical solutions between the discretized DPM and the given LPM is shown and Figure 5.29b shows the numerical solution difference. The maximum difference between simulation results is less than 2.6485, which is around 6.6% of the maximum output.

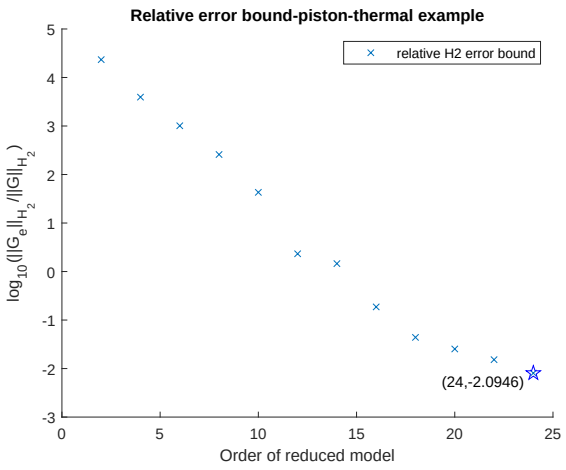
5.2.4 Results of thermomechanical examples

5.2.4.1 Model specifications

In this section, the proposed scheme is used to compute a priori error bound between a given DPM and a given LPM of a multiphysical (theraml + mechanical) system as shown in Figure 5.30. The given LPM (Figure 5.25a) is a two-domain LPM where a lumped mass-spring-damper model is connected to a lumped thermal resistor-capacitor model by a transformer. The parameters of the thermal domain is the same as those of the LPM in the thermal example above (Figure 5.25a) and the parameters of the mechanical domain is $m_1 = 6.0966 \times 10^8 kg$, $m_2 = 8.6614 \times 10^4 kg$, $k_1 = 1.2690 \times 10^8 N/m$, $k_2 = 9.0967 \times 10^4 N/m$, $r_1 = 6.7947 \times 10^7 Ns/m$, $r_2 = 2.7739 \times 10^4 Ns/m$, and $f = 12.659N$.

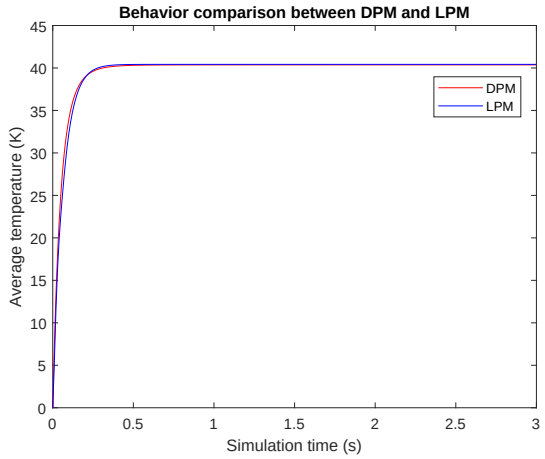


(a) Temperature change of piston under thermal load

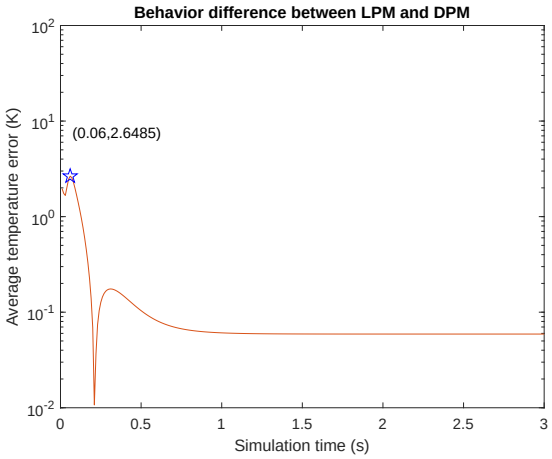


(b) A priori relative $\mathcal{H}_2$ error bound

Figure 5.28: Comparison of the simulation results between FOM and ROM of the thermal example of piston and the a priori relative $\mathcal{H}_2$ error bound



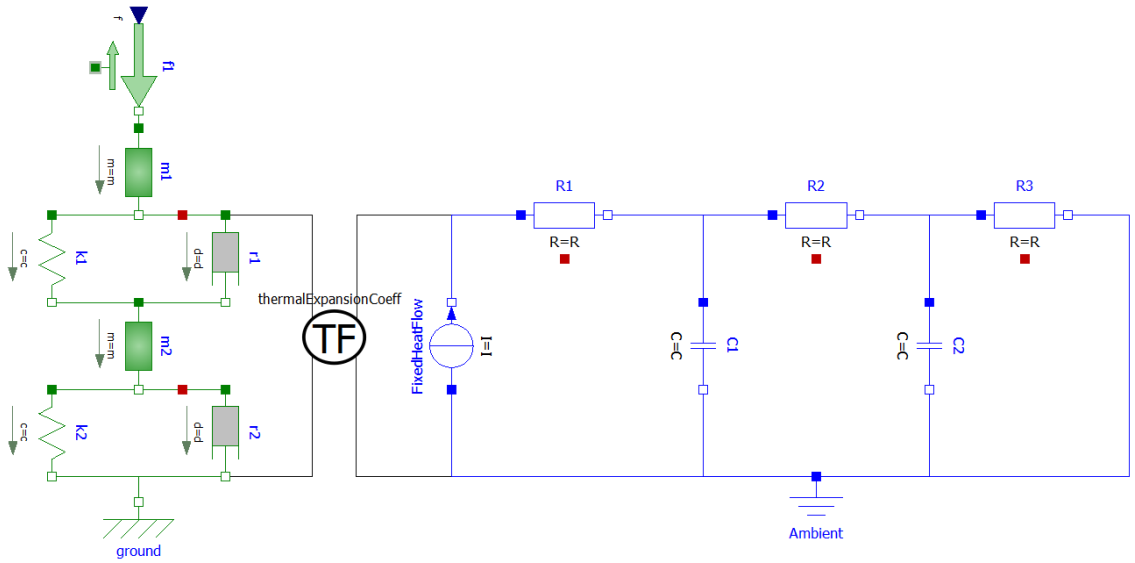
(a) Behavior comparison



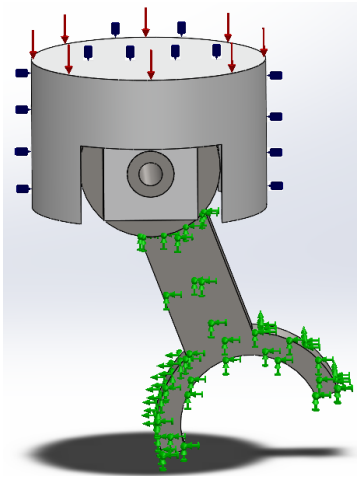
(b) Behavior difference

Figure 5.29: Behavior comparison between LPM and spatially-discretized DPM - thermal piston example

The geometry and the thermal initial/boundary conditions of the given DPM (Figure 5.30b) are the same as those used in the thermal example above. In addition, the piston bears a mechanical load, a constant unit pressure, added to the top surface and the rod of the piston is fixed.



(a) LPM of piston-thermomechanical



(b) DPM of piston-thermomechanical

Figure 5.30: LPM and DPM of the thermal and thermomechanical problem

All the specification correspondences in the thermal domain are the same as those used in the thermal example shown in Figure 5.31. For the specification correspondences in the mechanical domain, we have $T_n : f_1(t) = p_0(t)S_1$; Initial velocities of S_1 and S_{mid} are zero, which are respectively the same as the initial velocity of lumped masses m_1 and m_2; the displacements $u(S_2, t) = 0$ and $w_g(t) = 0$ are both zero, as described by T_d. Our goal is to compute the a priori error bound between the average displacement $\bar{u}(S_1, t)$ and the displacement $w_1(t)$ associated by the field correspondence T_f. Note that both of these two displacements are a superposition of two displacements: (1) the displacement caused by the mechanical load (2) the displacement caused by the thermal expansion because of the temperature change.

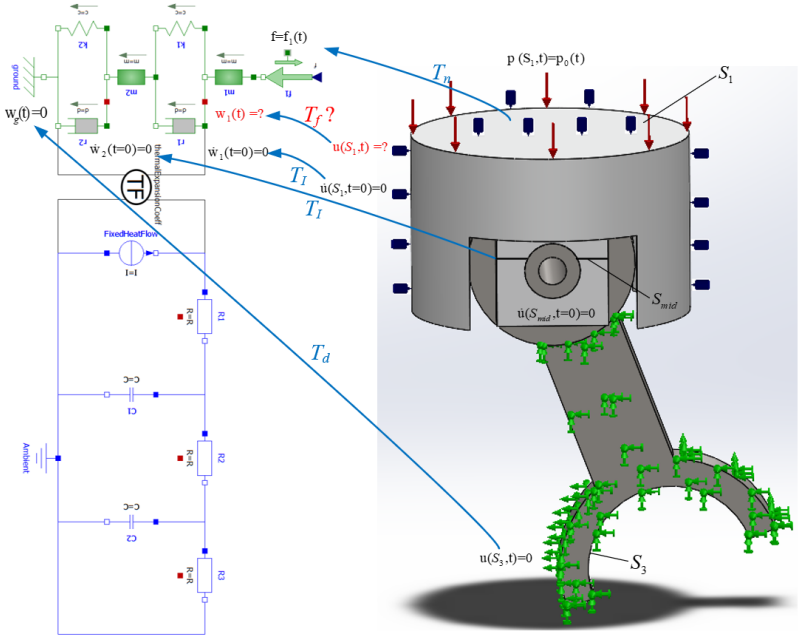


Figure 5.31: Initial, boundary, and field correspondences between LPM and DPM - bracket example (only correspondence in the mechanical domain is shown for clarity)

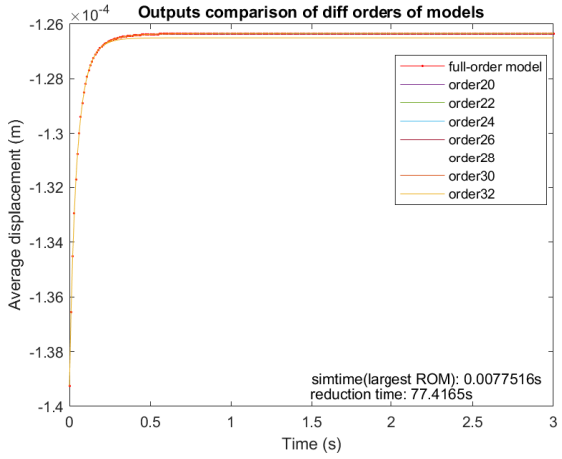
5.2.4.2 Surrogate LPM generation and error analysis

We use the same discretization as the one shown in the thermal example (Figure 5.27) and generate a family of ROMs with the SPARK+CURE with 1% being the a priori relative $\mathcal{H}_2$ error threshold. We simulate both the FOM and the ROMs and compare their results, as shown in Figure 5.32a.

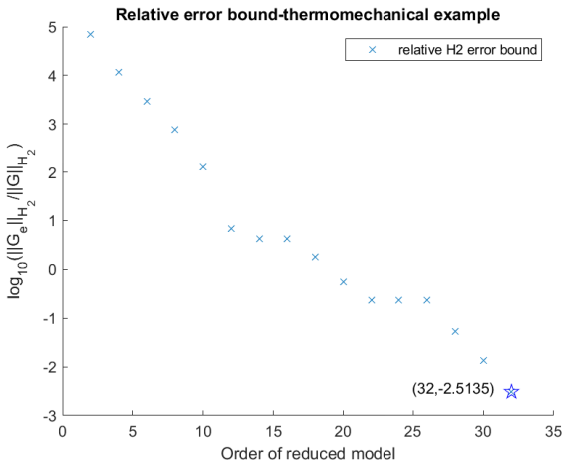
The computed the a priori relative $\mathcal{H}_2-$error bounds for each ROM is shown in Figure 5.32b, which shows if the ROM order is larger than 32, the exact relative $\mathcal{H}_2-$error will be less than $10^{-2.5135} \approx 0.307\%$. By using the value of this error bound, $\|\mathbf{G}_l(s)\|_{\mathcal{H}_2}$, and $\|\mathbf{G}_d(s) - \mathbf{G}_l(s)\|_{\mathcal{H}_2}$, we compute the a priori relative error bound between the discretized DPM and the given LPM as follows:

$$\begin{aligned} \frac{\|\mathbf{G}_d(s) - \mathbf{G}_l(s)\|_{\mathcal{H}_2}}{\|\mathbf{G}_l(s)\|_{\mathcal{H}_2}} &\leqslant \frac{\bar{\varepsilon}_1 + \bar{\varepsilon}_2}{\|\mathbf{G}_l(s)\|_{\mathcal{H}_2}} \\ &= 0.0030 + 0.0093 = 0.0124 = \bar{\varepsilon}_{rel} \end{aligned} \tag{5.12}$$

If the threshold for the relative error is larger than 0.0124, then LPM and DPM will be determined to be consistent. The comparison of the numerical simulation results between the discretized DPM and the given LPM is shown in Figure 5.29a, with their difference shown in Figure 5.29b. It can be observed that the maximum difference is less than 1.3144×10^{-6}, which is two orders smaller than the maximum simulation output $O(10^{-4})$.

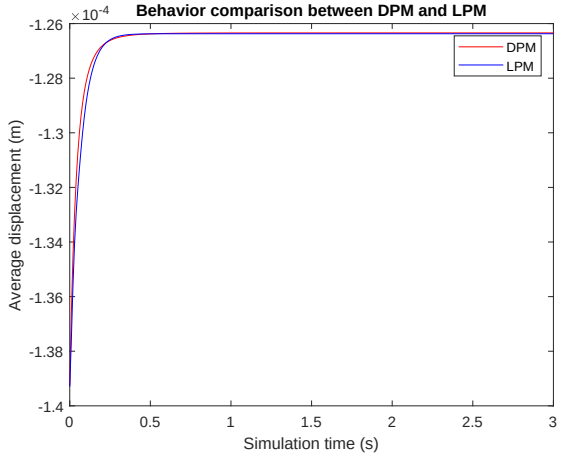


(a) Average displacement of piston under mechanical and thermal loads

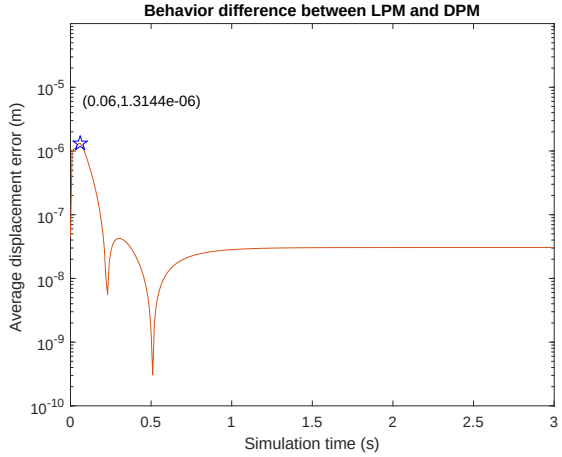


(b) A priori relative $\mathcal{H}_2$ error bound

Figure 5.32: Comparison of the simulation results between FOM and ROM of the thermomechanical example of piston and the a priori relative $\mathcal{H}_2$ error bound



(a) Behavior comparison



(b) Behavior difference

Figure 5.33: Behavior comparison between LPM and spatially-discretized DPM - thermomechanical piston example

6 CONCLUSION AND OPEN ISSUES

Contributions We proposed a framework for consistency analysis between lumped and distributed parameter models for facilitating the integration between the system and geometric designs. In this framework, the common semantic representation of lumped parameter systems is proposed to open the way to avoid expensive customized consistency analysis software between models created in different modeling languages and tools. Approaches to establishing the correspondence of model specifications and solutions between LPMs and DPMs are proposed, which are independent of any modeling languages and tools, numerical methods, and supports a variety of different types of physical models.

Specifically, the proposed common semantics for lumped parameter systems relies only on standard tools from algebraic topology and Tonti diagrams, a complete classification of a variety of physical quantities, theories, and systems. The semantics is independent of any specific implementation assumptions, coordinates, linguistic constructs, or numerical simulation schemes. Particularly, paths over Tonti diagrams correspond to all possible ways to generate governing equations. The proposed semantics provides a single neutral format that can be translated to/from models in different lumped parameter modeling languages such as Modelica, Simulink, bond graphs, and linear graphs. We showed that extended Tonti diagrams provide a canonical method for representing LPMs computationally. Based on known classification of physical theories, behavior of any lumped parameter system may be described either as a collection of interacting single-domain Tonti diagrams or as a single generalized Tonti diagram with energy transduction represented by additional constraints. We demonstrated this representation supports algorithmic generation of all possible forms of the governing state equations as paths in the (collection of) Tonti diagrams. The topological and constitutive operators appear as labels on the edges of the diagram and may be interpreted either symbolically, giving differential equations, or numerically (e.g. as finite difference operators), corresponding to executable approximations of such models.

Importantly, the proposed formal semantic model could be extended to DPMs whose behavior is governed by PDEs, where different paths on the Tonti diagram of field theories correspond to different methods to generate governing equations and replacing symbolic operators by different numerical approximations would lead to different standard numerical simulation methods. Our proposed semantics for lumped parameter systems could be used to eliminate the need for customized point-to-point consistency analysis tools for currently existing and prospective physical modeling languages and tools.

Establishing the model consistency is defined as a process of checking the correspondences of model specifications and solutions at three levels: (1) the model structure level, which ensures the correspondences of initial/boundary conditions and unknown field of interest between the LPM and the DPM obey the design principle of the system-to-geometry design, for instance, the component-to-component realization principle. (2) and (3) are respectively the physical quantity type and the magnitude levels, which guarantees the physical quantities of model specifications and solutions are identical after certain functional mappings (e.g. the integral) in terms of the physical type and numeric values. Particularly, the correspondence at the quantity type level is established using functional mappings between physical quantities associated with the same time and space elements between the common semantic models (Tonti diagrams) of the LPM and the DPM; the correspondence at the physical quantity magnitude level is established by comparing the magnitude of the physical quantities that are related by the correspondences of initial/boundary conditions and unknown fields of interest.

Establishing the correspondence of the model solutions at the physical quantity magnitude level requires checking if the solutions of interest of the governing equations of LPM and DPM are similar. To avoid drawbacks of the numerical simulation such as the low computational efficiency for large-scale models, unstable and nonconvergent simulation solutions due to large time step sizes, etc., we proposed a simulation-free scheme to compare LPM and spatially-discretized DPMs before simulations, where only model specifications are used to provide upfront guarantees of the simulations. The scheme in principle supports all different spatial

discretization methods. Importantly, the CURE scheme, a MOR method that a priori guarantees the accuracy, stability, and convergence is adapted to address the low time efficiency problem caused by large model scales.

The proposed scheme is applied to analyze the model consistency of single and multi-domain mechanical and thermal models, where we demonstrated the CURE scheme could be used to generate a family of different orders of surrogate LPMs of the discretized DPM in a single MOR procedure and provide relatively tight $\mathcal{H}_2$ a priori error bounds for each of them. Both the exact $\mathcal{H}_2$ error and the $\mathcal{H}_2$ error bound monotonically decrease with respect to the increase of the surrogate LPM order, with the gap between them always being narrow. We also demonstrated a strict downtrend of the exact SSE and RMSE with respect to the increase of the surrogate LPM order. Significantly, if the order of the discretized DPM is larger than a certain value (usually several thousand), the model solution comparison using the proposed scheme would be more rapid than the comparison using the numerical simulation.

Limitations and open issues There are several limitations of our proposed framework. The first one is that the scheme to compare model solutions does not consider the error between the exact and numerical solutions of the DPM caused by the spatial discretization. We shift the responsibility for selecting a sufficiently good spatial discretization for the DPM to users, with the understanding that the analytic solution of the DPM will be never known. Nevertheless, there are several existing formulas developed to compute a priori error bound of the solutions between the numerical and the analytic DPMs of different PDE types, discretization resolutions, and dimensions of physical field, etc. [53, 98, 43, 114], which could be combined with our proposed scheme to give more complete and rigorous error bound between the LPM and the DPM.

The second limitation is that the CURE scheme can only be used to reduce the order of dissipative models and the computation of the rigorous a priori guarantees by SPARK+CURE is only valid for LTI models. One possible way to extend our proposed scheme beyond linear models to nonlinear models is dividing nonlinear MOR models into piecewise linear MOR models and then using SPARK+CURE,

similar to the strategy adapted by the Trajectory PieceWise Linear (TPWL) Method [101]. The other possible solution is using MOR methods developed especially for nonlinear models, for example, the Symplectic MOR methods [94], however, most of these methods do not provide rigorous a priori guarantees.

The third limitation underlies the proposed semantics for lumped parameter systems. Many such systems support numerous additional operations and constructs, such as signal flows and arbitrary mathematical transformations, that are not necessarily based on first principles. Whether and how such constructs should be included in the common semantics model is not entirely clear. In principle, they could be treated as special cases of energetic processes (as, for example, is advocated in bond graph literature), or they may require the introduction of additional types of constraints to the underlying topological model.

A.1 Projection-Based MOR

The aim of model-order reduction (MOR) is to find a "good" approximate dynamic model of the original system such that this approximation is numerically efficient, stable, and preserves certain physical and numerical properties.

A.1.1 Linear Time-Invariant Systems

For LTI systems, the overall MOR process can be described with the diagram shown in Fig. A.1, where a full-order model (FOM) parameterized by matrices $(\mathbf{E}, \mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D})$ represented in the state-space form comprising tens or hundreds of thousands, if not millions, of variables is reduced to a reduced-order model (MOR) parameterized by matrices $(\mathbf{E}_r, \mathbf{A}_r, \mathbf{B}_r, \mathbf{C}_r, \mathbf{D}_r)$ that usually have only tens to hundreds of variables [35]. These two models represent a multi-input, multi-output (MIMO) mapping from an array of input signals $\mathbf{u}(t)$ to an (observable) array of output signals $\mathbf{y}(t)$. The state variables $\mathbf{x}(t)$ and input/outputs are all semi-discretized (i.e., spatially discretized, temporally continuum) functions that represent the temporal evolution of localized physical quantities.

The first system of ODE/DAE system $\mathbf{E}\dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t)$ represent the FOM system dynamics, obtained by discretizing PDEs.[1] A similar expression is written for the ROM, as depicted in Figure A.1. The matrices $(\mathbf{E}, \mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D})$ are independent of time in an LTI system. Often times, the original equations are second- or higher-order (due to the Newtonian dynamics) from which a state-space (i.e., first-order) representation can be obtained by defining additional state variables that are first or higher time derivatives of the original state variables.

The second system of equations $\mathbf{y}(t) = \mathbf{C}\mathbf{x}(t) + \mathbf{D}\mathbf{u}(t)$ defines (observable) quantities of interest (QoI) or outputs, as a function of state variables $\mathbf{x}(t)$ of the

[1] If $\mathbf{E} = \mathbf{I}$ is an identity matrix, the DAEs reduce to ODEs, making their numerical simulation much simpler.

FOM and inputs $\mathbf{u}(t)$. A similar expression is written for the ROM, as depicted in Figure A.1. The outputs are physical quantities whose accurate prediction is what we ultimately care about, and with respect to which our error measures will be defined.

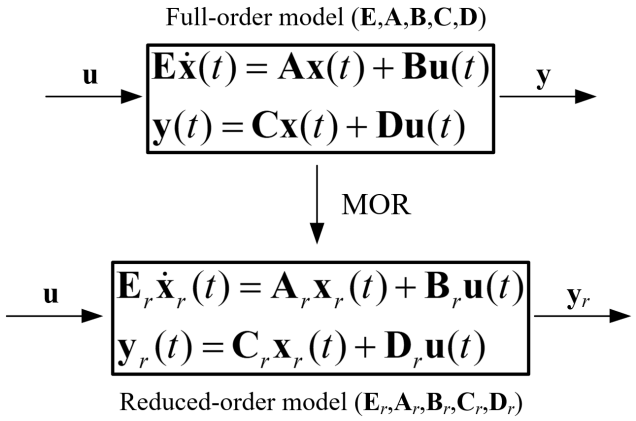


Figure A.1: Model order reduction (MOR) can be viewed as a mapping between the matrices used for state-space representations of FOM and ROM (time domain)

Figure A.1 illustrates the FOM and ROM as a MIMO relationship, constrained by ODE/DAE systems. For LTI systems, such relationships are more easily captured by a (matrix to matrix) transfer function in the frequency domain, as illustrated in Figure A.2. The transfer functions $\mathbf{G}(s)$ of the FOM is an algebraic function that maps the Laplace transform of the inputs $\hat{\mathbf{u}}(t) = \mathcal{L}[\mathbf{u}(t)]$ to the Laplace transform of the output $\hat{\mathbf{y}}(s) = \mathcal{L}[\mathbf{y}(t)]$, noting that differentiation in time domain is converted to an algebraic multiplication in frequency domain [62]. A similar relationship exists for the ROM in terms of the transfer function $\mathbf{G}_r(s)$, as illustrated in Figure A.2. Because of the LTI nature of the systems, the linear Laplace transform commutes with the matrix multiplications, leading to the simple algebraic relationship depicted in Figure A.2.

A commonly-used approach for MOR is based on Petrov-Galerkin projection [96], where the system matrices before and after MOR are related by:

$$\mathbf{E}_r = \mathbf{W}^T\mathbf{E}\mathbf{V},\ \mathbf{A}_r = \mathbf{W}^T\mathbf{A}\mathbf{V},\ \mathbf{B}_r = \mathbf{W}^T\mathbf{B},\ \mathbf{C}_r = \mathbf{C}\mathbf{V},\ \mathbf{D}_r = \mathbf{D} \tag{A.1}$$

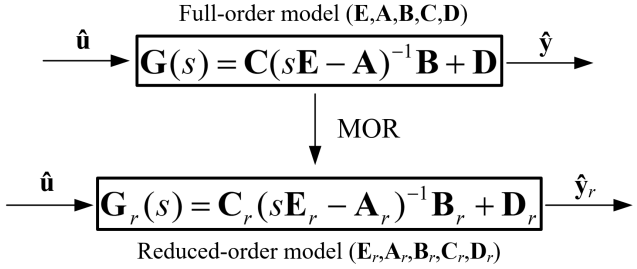


Figure A.2: Model order reduction (MOR) can be viewed as a mapping between the matrices used for state-space representations of FOM and ROM (frequency domain)

Particularly, the $\mathbf{V}$ is called the projection matrix whose column vectors are the basis vectors for the reduced state space and the column vectors in $\mathbf{W}$ are the basis vectors for the space that is orthogonal to the projection direction [35, 17]. How to find a pair of $\mathbf{V}$ and $\mathbf{W}$ depends on specific MOR principles, which differ from one method to another.

A.1.2 Rational Krylov Subspace (RKS) Method

Here, we briefly introduce the basic idea behind a classical MOR method based on projection, called the rational Krylov subspace (KRS) method [10]. To obtain a $k^{\text{th}}-$order ROM that has similar output as the $n^{\text{th}}-$order FOM with the same input,[2] the principle of this method is to match a number $k \ll n$ of most significant terms of the Tayler expansion series of the ROM transfer function, around specific frequencies to those of the FOM. The coefficients of these terms are usually referred to as "moments", hence RKS is also called the "moment-matching" method.

To achieve moment matching, mathematicians found that if we let the column vectors of $\mathbf{V}$ and $\mathbf{W}$ be the bases of the so-called input and output Krylov subspaces $\mathcal{K}_k\left((\mathbf{A}-s_0\mathbf{E})^{-1}\mathbf{E}, (\mathbf{A}-s_0\mathbf{E})^{-1}\mathbf{B}\right)$ and $\mathcal{K}_k\left((\mathbf{A}-s_0\mathbf{E})^{-T}\mathbf{E}^T, (\mathbf{A}-s_0\mathbf{E})^{-T}\mathbf{C}^T\right)$, respectively, then the obtained ROM would have the moment matching property. The Krylov subspace is defined in general as follows:

[2]The "order" of a model is the number of state variables (and the number of ODEs) in the state-space representation.

$$\mathcal{K}_k(\mathbf{M}, \mathbf{v}) = span\left\{\mathbf{v}, \mathbf{M}\mathbf{v}, \mathbf{M}^2\mathbf{v}, \ldots, \mathbf{M}^{k-1}\mathbf{v}\right\}. \tag{A.2}$$

However, RKS has several drawbacks; for example, it does not guarantee preservation of the model stability (i.e., even if FOM is stable, ROM may not be) and the Taylor expansion frequencies have to be selected manually, among others.

A.1.3 Iterative Rational Krylov Algorithm (IRKA)

To overcome the above limitations of RKS, researchers have developed an improved approach based on an iterative algorithm [60], which allows adaptively choosing the expansion frequencies using mirror image of *poles* of the obtained ROM. This algorithm is considered as the "gold standard" among the projection-based MOR methods that minimize the norm of approximation error of ROM compared to FOM for a given target model order $r < n$. In IRKA, the goal is to find a stable ROM for which the $\mathcal{H}_2$—norm [3] of the difference between the transfer functions $\mathbf{G}(s)$ and $\mathbf{G}_r(s)$ of FOM and ROM is minimized [18].

$$\|\mathbf{G} - \mathbf{G}_r\|_{\mathcal{H}_2} = \min_{\mathbf{G}'_r \in \mathbb{G}(k)} \|\mathbf{G} - \mathbf{G}'_r\|_{\mathcal{H}_2}, \tag{A.3}$$

where $\mathbb{G}(k)$ represents the space of all stable rational transfer functions describing dynamics of k^{th}—order LTI systems, i.e., all functions that can be parameterized as $\mathbf{G}'_r(s) = \mathbf{C}'_r(\mathbf{A}'_r - s\mathbf{E}'_r)^{-1}\mathbf{B}'_r + \mathbf{D}'_r$ by matrices $(\mathbf{A}'_r, \mathbf{B}'_r, \mathbf{C}'_r, \mathbf{D}'_r, \mathbf{E}'_r)$ as illustrated in Figure A.2. IRKA gives the "optimal" transfer function $\mathbf{G}_r(s) = \mathbf{C}_r(\mathbf{A}_r - s\mathbf{E}_r)^{-1}\mathbf{B}_r + \mathbf{D}_r$, parameterized by optimal matrices $(\mathbf{A}_r, \mathbf{B}_r, \mathbf{C}_r, \mathbf{D}_r, \mathbf{E}_r)$. The transfer function is stable if all of its poles sit to the left of the imaginary axis in the complex plane.

Particularly, given the FOM $\mathbf{G}(s)$ and a ROM $\mathbf{G}_r(s)$, the $\mathcal{H}_2$—error between them

[3] $\|\mathbf{F}\|_{\mathcal{H}_p} = \left(\int_{-\infty}^{+\infty} |\mathbf{F}(i\omega)|^p \, d\omega\right)^{\frac{1}{p}}$ where $p \geqslant 1$ (most commonly $p = 1, 2, \infty$)

can be computed in terms of their poles and residues as follows:

$$\|\mathbf{G}-\mathbf{G}_r\|_{\mathcal{H}_2}^2 = \sum_{i=1}^{n} \phi_i \mathbf{G}(-\lambda_i) - \phi_i \mathbf{G}_r(-\lambda_i) - \sum_{j=1}^{r} \tilde{\phi}_j \mathbf{G}(-\tilde{\lambda}_j) - \tilde{\phi}_j \mathbf{G}_r(-\tilde{\lambda}_j), \tag{A.4}$$

where λ_j and $\tilde{\lambda}_j$ represent the poles whereas $\phi_i(s)$ and $\tilde{\phi}_i(s)$ represent the residues of $\mathbf{G}(s)$ and $\mathbf{G}_r(s)$, respectively. The poles λ_i of a complex rational function $\mathbf{G}(s)$ are defined as the roots (i.e., zeros) of the denominator polynomial in standard rational representation of the function (i.e., as an irreducible ratio of polynomials). The residue ϕ_i is obtained by removing the $(s-\lambda_i)$ factor(s) from the denominator and evaluating the remaining fraction of nonzero terms at $s := \lambda_i$. For example, if $\mathbf{G}(s)$ is factorized as $(s-z_1)(s-z_2)\cdots(s-z_m)\cdot(s-\lambda_1)^{-1}(s-\lambda_2)^{-1}\cdots(s-\lambda_n)^{-1}$, then $\phi_i = \mathbf{G}(s)(s-\lambda_i)|_{s:=\lambda_i}$. Computationally, the poles of $\mathbf{G}(s) = \mathbf{C}(s\mathbf{E}-\mathbf{A})^{-1}\mathbf{B}+\mathbf{D}$ can be obtained from reachable and observable eigenvalues of $\mathbf{E}^{-1}\mathbf{A}$. See [36] for more details.

It has been showed that to minimize $\|\mathbf{G}-\mathbf{G'}_r\|_{\mathcal{H}_2}^2$ in Eq.A.3, the expansion frequencies must be selected at the mirror image of the poles of the optimal ROM [60]. However, since the optimal ROM is not known a priori, we start from a good initial guess of these expansion frequencies and iteratively update them until we arrive at the optimal set of poles.

In practice, one often uses the first k poles of the FOM with the largest residues for the initial expansion. The reason for this choice is that it would make the first sum in Eq.A.4 as small as possible at the beginning of the iterative process. The poles can be obtained by computing the eigenvalues of $\mathbf{E^{-1}A}$, which is easy when we are dealing with a system of ODEs, i.e., $\mathbf{E} = \mathbf{I}$ is an identity matrix. For DAEs in general, the inversion of $\mathbf{E}$ can be costly. The alternative approach is to compute the roots of the polynomial $\det(\mathbf{E}^{-1}\mathbf{A}-\lambda\mathbf{I})$ or equivalently the generalized eigenvalues of the matrix pair $(\mathbf{E},\mathbf{A})$ [90]. The iterative process in IRKA is as follows:

Step 1: Select initial expansion frequencies $\mathbf{s} = \{s_1, s_2, ..., s_k\}$ and fix a convergence

tolerance $\mathrm{tol} > 0$ for the change of pole locations .

Step 2: Initialize $\mathbf{A} \leftarrow \mathbf{A}_r$ and $\mathbf{E} \leftarrow \mathbf{E}_r$.

Step 3: While the magnitude of the relative change of s_i is larger than tol:

(3b) Let $s_i \leftarrow -\lambda_i[\mathbf{E}_r^{-1}\mathbf{A}_r]$, i.e., select the i^{th} largest eigenvalue of $\mathbf{E}_r^{-1}\mathbf{A}_r$ as the i^{th} expansion frequency, for $i = 1, \ldots, k$.

(3c) Choose $\mathbf{V}$ and $\mathbf{W}$ for Petrov-Galerkin projection as the bases for the following subspaces, respectively:

$$span\left\{(\mathbf{A} - s_1\mathbf{E})^{-1}\mathbf{B}, (\mathbf{A} - s_2\mathbf{E})^{-1}\mathbf{B}, ..., (\mathbf{A} - s_k\mathbf{E})^{-1}\mathbf{B}\right\}, \tag{A.5}$$

$$span\left\{(\mathbf{A} - s_1\mathbf{E})^{-T}\mathbf{C}^T, (\mathbf{A} - s_2\mathbf{E})^{-T}\mathbf{C}^T, ..., (\mathbf{A} - s_k\mathbf{E})^{-T}\mathbf{C}^T\right\}. \tag{A.6}$$

(3a) Update $\mathbf{A}_r \leftarrow \mathbf{W}^T\mathbf{A}\mathbf{V}$ and $\mathbf{E}_r \leftarrow \mathbf{W}^T\mathbf{E}\mathbf{V}$.

Step 4: Compute $\mathbf{B}_r \leftarrow \mathbf{W}^T\mathbf{B}$, $\mathbf{C}_r \leftarrow \mathbf{C}\mathbf{V}$, and $\mathbf{D}_r \leftarrow \mathbf{D}$.

The flowchart in Fig. A.3 illustrates the IRKA.

Compared to RKS, the IRKA can automatically choose expansion frequencies by iteration while maintaining the model stability. However, it can neither guarantee monotonic decrease of the $\mathcal{H}_2$−error with every iteration, nor ensure that the error converges to a local minimum, i.e., the final ROM is optimal as defined in Eq.A.4 [19].

A.1.4 CUmulative REduction (CURE) Scheme

In order to overcome the limitations of IRKA, researchers have developed the CURE scheme [88], which not only allows adaptively choosing the expansion frequencies using the poles of the obtained ROM (e.g., similarly to IRKA), but also enables incrementally increasing the model order by cumulatively updating the ROM error transfer function. Particularly, the $\mathcal{H}_2$−error of the obtained ROM monotonically converges to zero (i.e., ROM converges to FOM) in the accumulation process. In

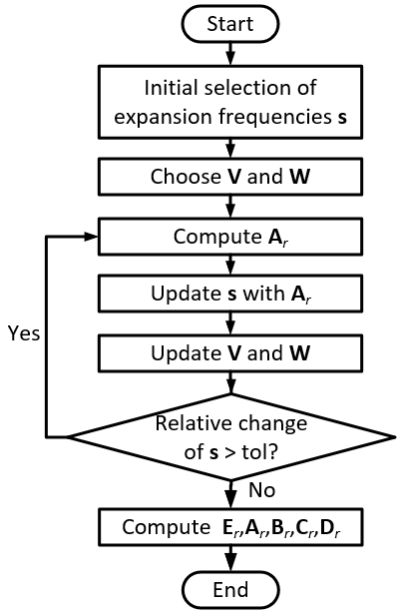


Figure A.3: The flow chart of the iteration process of IRKA

addition, to maintain the model stability, a stability-preserving, adaptive rational Krylov (SPARK) algorithm is developed, which is used to generate a stable ROM of order 2. This algorithm is usually embedded in the CURE framework to generate a family of ROMs with model orders $2, 4, \ldots, 2k$ in k times of accumulation in one MOR process.

The CURE scheme is built based on a decomposition of the transfer function of FOM $\mathbf{G}(s)$ into a summation of the transfer function of ROM $\mathbf{G}_r(s)$ and the transfer function of error model $\mathbf{G}_e(s)$ for the case of $\mathbf{D} := 0$ hence $\mathbf{G}(s) = \mathbf{C}(s\mathbf{E} - \mathbf{A})^{-1}\mathbf{B}$, i.e., the dependence of the output QoI on the input is only through the state variables.[4] Particularly, in the projection-based MOR techniques, $\mathbf{G}_e(s)$ can be rewritten as a product of two transfer functions $\mathbf{G}_\perp(s)$ and $\tilde{\mathbf{G}}_r^R(s)$ if we use one-side reduction where we set $\mathbf{W} = \mathbf{V}$ and $\mathbf{V}$ is the basis for input Krylov subspace $\mathcal{K}_k\left((\mathbf{A} - s_0\mathbf{E})^{-1}\mathbf{E}, (\mathbf{A} - s_0\mathbf{E})^{-1}\mathbf{B}\right)$, the matrix representation $(\mathbf{E}, \mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D})$ of the

[4] In the original CURE scheme in [88], the author assumes $\mathbf{D} = \mathbf{0}$ in the state-space representation of FOM. Nevertheless, the generalization to nonzero $\mathbf{D}$ is straightforward.

three transfer functions are related as follows:

$$\underbrace{\begin{bmatrix} \mathbf{E}_e, \mathbf{A}_e & \mathbf{B}_e \\ \mathbf{C}_e & \mathbf{D}_e \end{bmatrix}}_{\mathbf{G}_e(s)} = \underbrace{\begin{bmatrix} \mathbf{E}, \mathbf{A} & \mathbf{B}_\perp \\ \mathbf{C} & \mathbf{0} \end{bmatrix}}_{\mathbf{G}_\perp(s)} \cdot \underbrace{\begin{bmatrix} \mathbf{E}_r, \mathbf{A}_r & \mathbf{B}_r \\ \tilde{\mathbf{C}}_r & \mathbf{I} \end{bmatrix}}_{\tilde{\mathbf{G}}_r^R(s)}, \tag{A.7}$$

where

$$\mathbf{B}_\perp = \mathbf{B} - \mathbf{E}\mathbf{V}\mathbf{E}_r^{-1}\mathbf{B}_r \tag{A.8}$$

$$\tilde{\mathbf{C}}_r = \left(\mathbf{B}_\perp^T \mathbf{B}_\perp\right)^{-1} \mathbf{B}_\perp^T \left(\mathbf{A}\mathbf{V} - \mathbf{E}\mathbf{V}\mathbf{E}_r^{-1}\mathbf{A}_r\right). \tag{A.9}$$

By contrast, if we set $\mathbf{W} = \mathbf{V}$ and $\mathbf{W}$ is the basis for output Krylov subspace $\mathcal{K}_k\left((\mathbf{A} - s_0\mathbf{E})^{-T}\mathbf{E}^T, (\mathbf{A} - s_0\mathbf{E})^{-T}\mathbf{C}^T\right)$, then $\mathbf{G}_e(s)$ can be rewritten as a product of two transfer functions $\mathbf{G}_\perp(s)$ and $\tilde{\mathbf{G}}_r^L(s)$, we obtain the following decomposition:

$$\underbrace{\begin{bmatrix} \mathbf{E}_e, \mathbf{A}_e & \mathbf{B}_e \\ \mathbf{C}_e & \mathbf{D}_e \end{bmatrix}}_{\mathbf{G}_e(s)} = \underbrace{\begin{bmatrix} \mathbf{E}_r, \mathbf{A}_r & \tilde{\mathbf{B}}_r \\ \mathbf{C}_r & \mathbf{I} \end{bmatrix}}_{\tilde{\mathbf{G}}_r^L(s)} \cdot \underbrace{\begin{bmatrix} \mathbf{E}, \mathbf{A} & \mathbf{B} \\ \mathbf{C}_\perp & \mathbf{0} \end{bmatrix}}_{\mathbf{G}_\perp(s)}, \tag{A.10}$$

where

$$\mathbf{C}_\perp = \mathbf{C} - \mathbf{E}^T\mathbf{W}\mathbf{E}_r^{-T}\mathbf{C}_r, \tag{A.11}$$

$$\tilde{\mathbf{B}}_r = \left(\mathbf{C}_\perp\mathbf{C}_\perp^T\right)^{-1}\mathbf{C}_\perp^T\left(\mathbf{A}^T\mathbf{W} - \mathbf{E}\mathbf{W}\mathbf{E}_r^{-T}\mathbf{A}_\perp^T\right). \tag{A.12}$$

The general idea of the CURE scheme is shown in Figure A.4, in which the first three alternating MOR steps use one-side reduction with $\mathbf{V}$, $\mathbf{W}$, and $\mathbf{V}$, respectively, i.e., first step uses $\mathbf{V}$, second step uses $\mathbf{W}$, and third step uses $\mathbf{V}$. Specifically, in the first MOR step we compute the input-type error decomposition. Then we reduce $\mathbf{G}_{\perp,1}$ in the second step by using $\mathbf{W}$ projection matrix and factorize the obtained $\mathbf{G}_{\perp,2}$ using output-type decomposition. The third reduction to $\mathbf{G}_{\perp,3}$ uses $\mathbf{V}$ projection matrix. In the t^{th} step $(t > 1)$, the transfer function $\mathbf{G}_{r,t}^{\Sigma}(s)$ of the obtained ROM

can be computed by

$$\mathbf{G}^{\Sigma}_{r,t}(s) = \mathbf{G}^{\Sigma}_{r,t-1}(s) + \tilde{\mathbf{G}}^{\Sigma,L}_{r,t-1}(s) \cdot \mathbf{G}_{r,t}(s) \cdot \tilde{\mathbf{G}}^{\Sigma,R}_{r,t-1}(s). \tag{A.13}$$

It can be observed that $\mathbf{G}^{\Sigma}_{r,t}(s)$ is computed in an accumulated way at each step and the order of ROM increases at each step. Specifically, the order of $\mathbf{G}^{\Sigma}_{r,t}(s)$ is the summation of the order of $\mathbf{G}_{r,t}(s)$ and $\mathbf{G}^{\Sigma}_{r,t-1}(s)$ because $\mathbf{G}^{\Sigma}_{r,t-1}(s)$, $\tilde{\mathbf{G}}^{\Sigma,L}_{r,t-1}(s)$, or $\tilde{\mathbf{G}}^{\Sigma,R}_{r,t-1}$ share the same poles. The reasons for improving $\mathbf{G}^{\Sigma}_{r,t-1}(s)$ by performing reduction on $\mathbf{G}_{\perp,t-1}(s)$ are the following:

1. $\mathbf{G}_{\perp,t-1}(s)$ contains the dynamics of FOM that has not been captured by the ROM so far; and

2. $\mathbf{G}_{\perp,t-1}(s)$ is of higher order than the previous ROM and offers the potential to be approximated with other small-scale models $\tilde{\mathbf{G}}^{\Sigma,L}_{r,k-1}(s)$ and $\tilde{\mathbf{G}}^{\Sigma,R}_{r,k-1}(s)$.

The accumulation will stop until the a priori error bound is less than a user-defined threshold or a user-defined ROM order is reached.

Using the $\mathbf{G}_e(s)$ formulas given in Eq.A.7 and Eq.A.10, the CURE scheme can provide an a priori error bound, i.e., the following bound on the $\mathcal{H}_2-$norm of the difference between the transfer functions of FOM and ROM:

$$\|\mathbf{G}_e(s)\|_{\mathcal{H}_2} \leqslant \|\mathbf{G}_\perp(s)\|_{\mathcal{H}_2} \cdot \|\tilde{\mathbf{G}}^{L}_{r}(s)\|_{\mathcal{H}_\infty} \cdot \|\tilde{\mathbf{G}}^{R}_{r}(s)\|_{\mathcal{H}_\infty} \tag{A.14}$$

A.1.5 Stability-Preserving, Adaptive Rational Krylov (SPARK) Algorithm

In the CURE scheme, for each step we can use the IRKA to reduce the order of $\mathbf{G}_{\perp,t}(s)$, however, there is no guarantee that IRKA would converge at all in each step [88]. The SPARK algorithm was developed to overcome this problem by an adaptive rational Krylov subspace algorithm that can preserve the model stability and can converge rapidly to the local minimum of the $\mathcal{H}_2-$error [89]. However, SPARK is bound to generate ROMs of order 2 only. Even though such a low-order

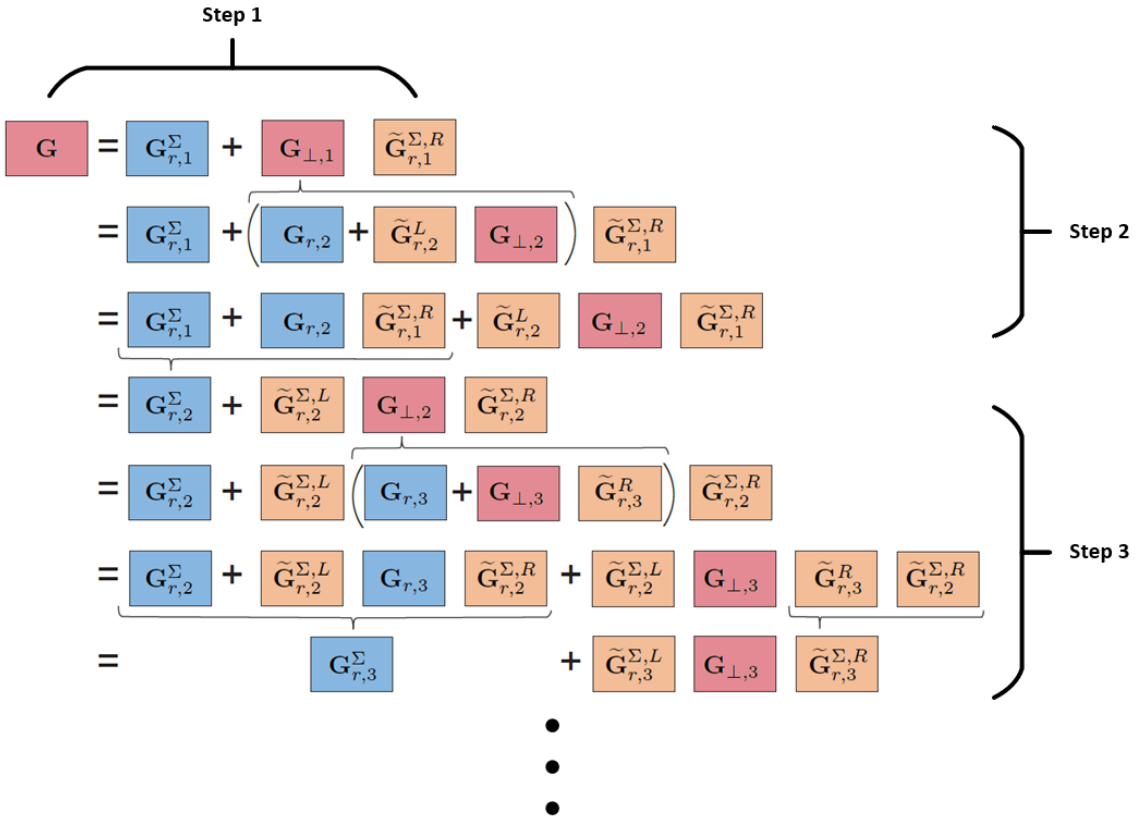


Figure A.4: Model order reduction procedure of the CURE Scheme (Figure taken with slight modification from [88])

approximation may not be useful on its own, it can be useful when SPARK is embedded into the CURE scheme where the overall ROM is incrementally built with many low-order models. Before introducing SPARK, we need to define the concept of an $\mathcal{H}_2$—*pseudo-optimal approximant*.

A ROM $\mathbf{G}_r(s)$ is called a $\mathcal{H}_2$—pseudo-optimal approximant of FOM $\mathbf{G}(s)$ if $\langle \mathbf{G}_r, \mathbf{G}_e \rangle_{\mathcal{H}_2} = \langle \mathbf{G}_r, \mathbf{G} - \mathbf{G}_r \rangle_{\mathcal{H}_2} = 0$, or, equivalently, $\langle \mathbf{G}_r, \mathbf{G} \rangle_{\mathcal{H}_2} = \langle \mathbf{G}_r, \mathbf{G}_r \rangle_{\mathcal{H}_2}$, where $\langle \cdot, \cdot \rangle_{\mathcal{H}_2}$ is the $\mathcal{H}_2$—inner product.[5] With this definition, the $\mathcal{H}_2$—error $\|\mathbf{G}_e\|^2_{\mathcal{H}_2}$ is

[5] $\langle \mathbf{F}, \mathbf{G} \rangle = \int_{-\infty}^{+\infty} \mathbf{F}(i\omega)\overline{\mathbf{G}}(i\omega)\, d\omega$ where $\overline{\mathbf{G}}(i\omega)$ is complex-conjugate of $\mathbf{G}(i\omega)$.

obtained as:

$$\|\mathbf{G}-\mathbf{G}_r\|_{\mathcal{H}_2}^2 = \|\mathbf{G}\|_{\mathcal{H}_2}^2 + \|\mathbf{G}_r\|_{\mathcal{H}_2}^2 - 2\langle \mathbf{G}_r, \mathbf{G}\rangle_{\mathcal{H}_2} \tag{A.15}$$

$$= \|\mathbf{G}\|_{\mathcal{H}_2}^2 - \|\mathbf{G}_r\|_{\mathcal{H}_2}^2 \tag{A.16}$$

noting that $\langle \mathbf{G}_r, \mathbf{G}\rangle_{\mathcal{H}_2} = \langle \mathbf{G}_r, \mathbf{G}_r\rangle_{\mathcal{H}_2} = \|\mathbf{G}_r\|_{\mathcal{H}_2}^2$. The term $\|\mathbf{G}\|_{\mathcal{H}_2}^2$ is independent of the reduction. Hence, minimizing $\mathcal{H}_2-$norm $\|\mathbf{G}_e\|_{\mathcal{H}_2}^2$ amounts to maximizing $\|\mathbf{G}_r\|_{\mathcal{H}_2}^2$.

The maximization process can be solved rapidly to find the optimal $\mathbf{G}_r(s)$ of lowest order ($k := 2$). According to Corollary 4.2 in [88], if let $s_{1,2} = a \pm \sqrt{a^2 - b}$ where $a, b \in (0, \infty)$, then the ROM defined by $\mathbf{E}_r = \mathbf{I}$, $\mathbf{A}_r = \begin{bmatrix} -3a & 1 \\ -3a^2 & b \end{bmatrix}$, $\mathbf{B}_r = \begin{bmatrix} -4a \\ -4a^2 \end{bmatrix}$, and $\mathbf{C}_r = \mathbf{CV}$ is an $\mathcal{H}_2-$pseudo-optimal approximant of the FOM among all the stable models of order 2. With matrices $\mathbf{E}_r$, $\mathbf{A}_r$, $\mathbf{B}_r$ and $\mathbf{C}_r$, the objective function $\|\mathbf{G}_r\|_{\mathcal{H}_2}^2$ can be directly expressed in terms of a and b as follows:

$$\|\mathbf{G}_r\|_{\mathcal{H}_2}^2 = \mathbf{CV} \begin{bmatrix} 4a & 4a^2 \\ 4a^2 & 4a\left(a^2+b\right) \end{bmatrix} (\mathbf{CV})^T, \tag{A.17}$$

where the output matrix $\mathbf{C}$ is given by user and the projection matrix $\mathbf{V}$ is a function of a and b only, because it is a function of the expansion frequencies $s_{1,2}$, $\mathbf{E}_r$, $\mathbf{A}_r$ and $\mathbf{B}_r$, all of which are functions of a and b only. The projection matrix $\mathbf{V}$ will be updated iteratively when we solve the minimization problem in terms of only two variables a and b, which can be solved rapidly [88].

In practice, SPARK is usually embedded into the CURE scheme as follows:

Step 1: Find $\mathbf{G}_r(s)$ of order $k := 2$ by solving the two-variate constrained minimization problem mentioned above.

Step 2: Update overall ROM according to CURE Scheme, described earlier.

Step 3: Repeat the cumulative reduction until the change in the objective function is

smaller than a certain threshold.

However, the SPARK+CURE approach has several limitations: (1) the FOM itself must be dissipative (hence stable) to begin with; (2) the resulting ROM does not necessarily have a physical interpretation; and (3) the approach is only valid for linear time-invariant (LTI) systems, although there are several ways in which it can be generalized to nonlinear systems, for example, dividing nonlinear MOR problems into piecewise linear MOR problems proposed in the Trajectory PieceWise Linear (TPWL) Method [100].

To clarify the second point, remember that the state-space representation as a system of first-order ODE/DAE is often derived from second- or higher-order ODE/DAE obtained from spatial discretization of physics-based (e.g., Newtonian) PDE/PDAE with an underlying topological structure (e.g., 2D or 3D spatial mesh, electrical circuit, system model, and chemical reaction network). This structure is lost once the system is represented in state-space form. After projection-based MOR, it is not always possible to retrieve a topological, geometric, and generally physics-based structure from the reduced first-order system of ODE/DAE. An alternative class of MOR methods that preserve the physics-based (e.g., Hamiltonian) structure of the FOM can remedy this issue, however to the best of our knowledge, a priori error cannot be guaranteed in these MOR methods.

REFERENCES

[1] ABAQUS, CAE. 2012. Analysis user's manual, version 6.12.

[2] ANSYS, ANSYS LS-DYNA. 2017. user's guide, ansys. *Inc., Pennsylvania, US*.

[3] Antoulas, Athanasios C. 2005. *Approximation of large-scale dynamical systems*, vol. 6. Siam.

[4] Antoulas, Athanasios C, Christopher A Beattie, and Serkan Gugercin. 2010. Interpolatory model reduction of large-scale dynamical systems. In *Efficient modeling and control of large-scale systems*, 3–58. Springer.

[5] Antoulas, Athanasios C, Danny C Sorensen, and Serkan Gugercin. 2000. A survey of model reduction methods for large-scale systems.

[6] Arnold, Douglas N, Pavel B Bochev, Richard B Lehoucq, Roy A Nicolaides, and Mikhail Shashkov. 2007. *Compatible spatial discretizations*, vol. 142. Springer Science & Business Media.

[7] Atkinson, Kendall, Weimin Han, and David E Stewart. 2011. *Numerical solution of ordinary differential equations*, vol. 108. John Wiley & Sons.

[8] Babuska, Robert, and Stefano Stramigioli. 1999. Matlab and simulink for modeling and control. *Delft University of Technology*.

[9] Bai, Zhaojun. 2002. Krylov subspace techniques for reduced-order modeling of large-scale dynamical systems. *Applied numerical mathematics* 43(1-2):9–44.

[10] Bai, Zhaojun, and Yangfeng Su. 2005. Dimension reduction of large-scale second-order dynamical systems via a second-order arnoldi method. *SIAM Journal on Scientific Computing* 26(5):1692–1709.

[11] Bamberg, Paul, and Shlomo Sternberg. 1991. *A course in mathematics for students of physics*, vol. 2. Cambridge University Press.

[12] Banks, HT, SS Gates, IG Rosen, and Y Wang. 1988. The identification of a distributed parameter model for a flexible structure. *SIAM journal on control and optimization* 26(4):743–762.

[13] Baraff, David, and Andrew Witkin. 1998. Large steps in cloth simulation. In *Proceedings of the 25th annual conference on computer graphics and interactive techniques*, 43–54.

[14] Bathe, Klaus-Jürgen. 2006. *Finite element procedures*. Klaus-Jurgen Bathe.

[15] Baudet, Vincent, Michael Beuve, Fabrice Jaillet, Behzad Shariat, and Florence Zara. 2007. Integrating tensile parameters in 3d mass-spring system.

[16] Baudet, Vincent, Michaël Beuve, Fabrice Jaillet, Behzad Shariat, and Florence Zara. 2007. New mass-spring system integrating elasticity parameters in 2d.

[17] Baur, Ulrike, Peter Benner, and Lihong Feng. 2014. Model order reduction for linear and nonlinear systems: a system-theoretic perspective. *Archives of Computational Methods in Engineering* 21(4):331–358.

[18] Beattie, Christopher A, and Serkan Gugercin. 2007. Krylov-based minimization for optimal h 2 model reduction. In *2007 46th ieee conference on decision and control*, 4385–4390. IEEE.

[19] ———. 2009. A trust region method for optimal h 2 model reduction. In *Proceedings of the 48h ieee conference on decision and control (cdc) held jointly with 2009 28th chinese control conference*, 5370–5375. IEEE.

[20] Benner, Peter, Volker Mehrmann, and Danny C Sorensen. 2005. *Dimension reduction of large-scale systems*, vol. 35. Springer.

[21] Benner, Peter, and Enrique S Quintana-Ortí. 2005. Model reduction based on spectral projection methods. In *Dimension reduction of large-scale systems*, 5–48. Springer.

[22] Besselink, Bart, Umut Tabak, Agnieszka Lutowska, Nathan Van de Wouw, H Nijmeijer, Daniel J Rixen, ME Hochstenbach, and WHA Schilders. 2013. A comparison of model reduction techniques from structural dynamics, numerical mathematics and systems and control. *Journal of Sound and Vibration* 332(19):4403–4422.

[23] Blazek, Jiri. 2015. *Computational fluid dynamics: principles and applications*. Butterworth-Heinemann.

[24] Bock, Conrad, Raphael Barbau, Ion Matei, and Mehdi Dadfarnia. 2017. An extension of the systems modeling language for physical interaction and signal flow simulation. *Systems Engineering* 20(5):395–431.

[25] Bossavit, Alain. 1998. *Computational electromagnetism: variational formulations, complementarity, edge elements*. Academic Press.

[26] Branin, Franklin H. 1966. The algebraic-topological basis for network analogies and the vector calculus. In *Symposium on generalized networks*, 453–491.

[27] Butcher, John Charles. 1987. *The numerical analysis of ordinary differential equations: Runge-kutta and general linear methods*. Wiley-Interscience.

[28] Butcher, John Charles, and Nicolette Goodwin. 2008. *Numerical methods for ordinary differential equations*, vol. 2. Wiley Online Library.

[29] Callier, Frank M, and Charles A Desoer. 2012. *Linear system theory*. Springer Science & Business Media.

[30] Castagnotto, Alessandro, Maria Cruz Varona, Lisa Jeschek, and Boris Lohmann. 2017. sss & sssmor: Analysis and reduction of large-scale dynamic systems in matlab. *at-Automatisierungstechnik* 65(2):134–150.

[31] Castillo, José E, and Guillermo F Miranda. 2013. *Mimetic discretization methods*. CRC Press.

[32] Chahlaoui, Younes, Kyle A Gallivan, Antoine Vandendorpe, and Paul Van Dooren. 2005. Model reduction of second-order systems. In *Dimension reduction of large-scale systems*, 149–172. Springer.

[33] Chaturvedi, Devendra K. 2009. *Modeling and simulation of systems using matlab and simulink*. CRC Press.

[34] Chen, Yong. 1999. Model order reduction for nonlinear systems. Ph.D. thesis, Massachusetts Institute of Technology.

[35] Cruz Varona, M, Alessandro Castagnotto, Thomas Wolf, Rudy Eid, and Mikhail Pak. 2019. Modeling and reduction of complex systems-part b: Introduction to model order reduction.

[36] Dorf, Richard C, and Robert H Bishop. 2011. *Modern control systems*. Pearson.

[37] Durran, Dale R. 1991. The third-order adams-bashforth method: An attractive alternative to leapfrog time differencing. *Monthly weather review* 119(3): 702–720.

[38] Elmqvist, Hilding, and Sven Erik Mattsson. 1997. An introduction to the physical modeling language modelica. In *Proceedings of the 9th european simulation symposium, ess*, vol. 97, 19–23. Citeseer.

[39] Engelson, Vadim, Peter Bunus, Lucian Popescu, and Peter Fritzson. 2003. Mechanical cad with multibody dynamic analysis based on modelica simulation. In *Proceedings of the 44th scandinavian conference on simulation and modeling*, 18–19.

[40] Eymard, Robert, Thierry Gallouët, and Raphaèle Herbin. 2000. Finite volume methods. *Handbook of numerical analysis* 7:713–1018.

[41] Felippa, Carlos A. 2004. Introduction to finite element methods. *University of Colorado* 885.

[42] Feng, Lihong, and Peter Benner. 2007. Arobust algorithm for parametric model order reduction. In *Pamm: Proceedings in applied mathematics and mechanics*, vol. 7, 1021501–1021502. Wiley Online Library.

[43] Feng, Yuming, Zuliang Lu, Longzhou Cao, Lin Li, and Shuhua Zhang. 2017. A priori error estimates of finite volume methods for general elliptic optimal control problems. *Electr J Differ Equ* 267:1–15.

[44] Ferretti, Elena. 2015. The cell method: An overview on the main features. *Curved and Layered Structures* 2(1).

[45] Feynman, Richard. 2017. *The character of physical law*. MIT press.

[46] Finger, Susan, James Rinderle, et al. 1990. *A transformational approach to mechanical design using a bond graph grammar*. [Carnegie Mellon University], Engineering Design Research Center.

[47] Franklin, Jerrold. 2017. *Classical electromagnetism*. Courier Dover Publications.

[48] Freund, Roland W. 2004. Sprim: structure-preserving reduced-order interconnect macromodeling. In *Ieee/acm international conference on computer aided design, 2004. iccad-2004.*, 80–87. IEEE.

[49] Fritzson, Peter, and Vadim Engelson. 1998. Modelica—a unified object-oriented language for system modeling and simulation. In *Ecoop'98—object-oriented programming*, 67–90. Springer.

[50] Fujimoto, Kenji, and Jacquelien MA Scherpen. 2010. Balanced realization and model order reduction for nonlinear systems based on singular value analysis. *SIAM Journal on Control and Optimization* 48(7):4591–4623.

[51] Gelder, Allen Van. 1998. Approximate simulation of elastic membranes by triangulated spring meshes. *Journal of graphics tools* 3(2):21–41.

[52] Gildin, Eduardo. 2006. Model and controller reduction of large-scale structures based on projection methods. Ph.D. thesis.

[53] Gong, Wei, Michael Hinze, and Zhaojie Zhou. 2016. Finite element method and a priori error estimates for dirichlet boundary control problems governed by parabolic pdes. *Journal of Scientific Computing* 66(3):941–967.

[54] Gong, Yuezheng, Qi Wang, and Zhu Wang. 2017. Structure-preserving galerkin pod reduced-order modeling of hamiltonian systems. *Computer Methods in Applied Mechanics and Engineering* 315:780–798.

[55] Gottlieb, David, and Steven A Orszag. 1977. *Numerical analysis of spectral methods: theory and applications*. SIAM.

[56] Greer, James L, Daniel D Jensen, and Kristin L Wood. 2004. Effort flow analysis: a methodology for directed product evolution. *Design Studies* 25(2): 193–214.

[57] Greer, James LaMonte. 2002. Effort flow analysis: a methodology for directed product evolution using rigid body and compliant mechanisms. Ph.D. thesis, The University of Texas at Austin.

[58] Gressick, William, John T Wen, and Jacob Fish. 2005. Order reduction for large-scale finite element models: A systems perspective. *International Journal for Multiscale Computational Engineering* 3(3).

[59] Grimme, Eric. 1997. Krylov projection methods for model reduction. Ph.D. thesis.

[60] Gugercin, Serkan, Athanasios C Antoulas, and Cchristopher Beattie. 2008. $\mathcal{H}_2$ model reduction for large-scale linear dynamical systems. *SIAM journal on matrix analysis and applications* 30(2):609–638.

[61] Hatcher, Allen. 2002. *Algebraic topology*. Cambridge University Press.

[62] Heij, Christiaan, André CM Ran, and Frederik van Schagen. 2006. *Introduction to mathematical systems theory: linear systems, identification and control*. Springer Science & Business Media.

[63] Higham, Desmond J, and Nicholas J Higham. 2016. *Matlab guide*. SIAM.

[64] Hirani, Anil Nirmal. 2003. Discrete exterior calculus. Ph.D. thesis, California Institute of Technology.

[65] Hughes, Thomas JR, John A Cottrell, and Yuri Bazilevs. 2005. Isogeometric analysis: Cad, finite elements, nurbs, exact geometry and mesh refinement. *Computer methods in applied mechanics and engineering* 194(39-41):4135–4195.

[66] Kähler, Kolja, Jörg Haber, and Hans-Peter Seidel. 2001. Geometry-based muscle modeling for facial animation. In *Graphics interface*, vol. 2001, 37–46.

[67] Kamerlin, Shina CL, Spyridon Vicatos, Anatoly Dryga, and Arieh Warshel. 2011. Coarse-grained (multiscale) simulations in studies of biophysical and chemical systems. *Annual review of physical chemistry* 62:41–64.

[68] Karnopp, Dean C, Donald L Margolis, and Ronald C Rosenberg. 1990. System dynamics: a unified approach.

[69] ———. 2012. *System dynamics: modeling, simulation, and control of mechatronic systems*. John Wiley & Sons.

[70] Kluever, Craig A. 2015. *Dynamic systems: modeling, simulation, and control*. John Wiley & Sons.

[71] Kota, Sridhar, and GK Ananthasuresh. 1995. Designing compliant mechanisms. *Mechanical Engineering-CIME* 117(11):93–97.

[72] Kron, Gabriel. 1963. *Diakoptics: the piecewise solution of large-scale systems*, vol. 2. MacDonald.

[73] Lee, Yuencheng, Demetri Terzopoulos, and Keith Waters. 1995. Realistic modeling for facial animation. In *Proceedings of the 22nd annual conference on computer graphics and interactive techniques*, 55–62.

[74] Leis, Rolf. 2013. *Initial boundary value problems in mathematical physics*. Courier Corporation.

[75] Lloyd, Bryn, Gábor Székely, and Matthias Harders. 2007. Identification of spring parameters for deformable object simulation. *IEEE Transactions on Visualization and Computer Graphics* 13(5).

[76] Lohmann, Boris, and Rudy Eid. 2007. Efficient order reduction of parametric and nonlinear models by superposition of locally reduced models. In *Methoden und anwendungen der regelungstechnik. erlangen-münchener workshops*, 27–36.

[77] Lona, Liliane Maria Ferrareso. 2018. Distributed-parameter models. In *A step by step approach to the modeling of chemical engineering processes*, 49–87. Springer.

[78] Marrink, Siewert J, H Jelger Risselada, Serge Yefimov, D Peter Tieleman, and Alex H De Vries. 2007. The martini force field: coarse grained model for biomolecular simulations. *The journal of physical chemistry B* 111(27): 7812–7824.

[79] Mattiussi, Claudio. 1997. An analysis of finite volume, finite element, and finite difference methods using some concepts from algebraic topology. *Journal of Computational Physics* 133:289–309.

[80] ———. 2000. The finite volume, finite element, and finite difference methods as numerical methods for physical field problems 113:1–146.

[81] Maxwell, James Clerk. 1890. On the mathematical classification of physical quantities. In *The scientific papers of james clerk maxwell*, ed. W.D. Niven, 257 – 266. Cambridge University Press.

[82] Mazumder, Sandip. 2015. *Numerical methods for partial differential equations: finite difference and finite volume methods*. Academic Press.

[83] Mollemans, Wouter, Filip Schutyser, Johan Van Cleynenbreugel, and Paul Suetens. 2004. Fast soft tissue deformation with tetrahedral mass spring

model for maxillofacial surgery planning systems. In *International conference on medical image computing and computer-assisted intervention*, 371–379. Springer.

[84] Multiphysics, COMSOL. 2012. Comsol multiphysics user guide (version 4.3 a). *COMSOL, AB* 39–40.

[85] Natsupakpong, Suriya, and M Cenk Çavuşoğlu. 2010. Determination of elasticity parameters in lumped element (mass-spring) models of deformable objects. *Graphical Models* 72(6):61–73.

[86] Pahl, Gerhard, and Wolfgang Beitz. 2013. *Engineering design: a systematic approach*. Springer Science & Business Media.

[87] Panzer, Heiko, Jan Mohring, Rudy Eid, and Boris Lohmann. 2010. Parametric model order reduction by matrix interpolation. *at-Automatisierungstechnik Methoden und Anwendungen der Steuerungs-, Regelungs-und Informationstechnik* 58(8):475–484.

[88] Panzer, Heiko KF. 2014. Model order reduction by krylov subspace methods with global error bounds and automatic choice of parameters. Ph.D. thesis, Technische Universität München.

[89] Panzer, Heiko KF, Stefan Jaensch, Thomas Wolf, and Boris Lohmann. 2013. A greedy rational krylov method for $\mathcal{H}_2$-pseudooptimal model order reduction with preservation of stability. In *2013 american control conference*, 5512–5517. IEEE.

[90] Panzer, Heiko KF, Thomas Wolf, and Boris Lohmann. 2013. $\mathcal{H}_2$ and $\mathcal{H}_\infty$ error bounds for model order reduction of second order systems by krylov subspace methods. In *2013 european control conference (ecc)*, 4484–4489. IEEE.

[91] Patankar, Suhas V. 2017. *Computation of conduction and duct flow heat transfer*. Routledge.

[92] Paynter, Henry M. 1961. *Analysis and design of engineering systems*. MIT press.

[93] Peeters, Jelle, and Wim Michiels. 2013. Computing the h2 norm of large-scale time-delay systems. *IFAC Proceedings Volumes* 46(3):114–119.

[94] Peng, Liqian, and Kamran Mohseni. 2016. Symplectic model reduction of hamiltonian systems. *SIAM Journal on Scientific Computing* 38(1):A1–A27.

[95] Perelson, Alan S, and George F Oster. 1976. Bond graphs and linear graphs. *Journal of the Franklin Institute* 302(2):159–185.

[96] Pinnau, René. 2008. Model reduction via proper orthogonal decomposition. In *Model order reduction: theory, research aspects and applications*, 95–109. Springer.

[97] Prabhu, DR, and DL Taylor. 1989. Synthesis of systems from specifications containing orientations and positions associated with flow variables. In *Proceedings from 1989 asme design automation conference*, 273–280.

[98] Rannacher, Rolf, and Boris Vexler. 2005. A priori error estimates for the finite element discretization of elliptic parameter identification problems with pointwise measurements. *SIAM Journal on Control and Optimization* 44(5):1844–1863.

[99] Reis, Timo, and Tatjana Stykel. 2008. A survey on model reduction of coupled systems. In *Model order reduction: theory, research aspects and applications*, 133–155. Springer.

[100] Rewienski, Michal, and Jacob White. 2003. A trajectory piecewise-linear approach to model order reduction and fast simulation of nonlinear circuits and micromachined devices. *IEEE Transactions on computer-aided design of integrated circuits and systems* 22(2):155–170.

[101] Rewieński, Michał, and Jacob White. 2006. Model order reduction for nonlinear dynamical systems based on trajectory piecewise-linear approximations. *Linear algebra and its applications* 415(2-3):426–454.

[102] Roth, J Paul. 1955. An application of algebraic topology to numerical analysis: On the existence of a solution to the network problem. *Proceedings of the National Academy of Sciences* 41(7):518–521.

[103] Roth, J. Paul. 1971. Existence and uniqueness of solution to electrical network problem via homology sequences. In *Siam-ams proceedings, vol. iii*, 113–118. American Mathematical Society.

[104] Rowell, Derek, and David N Wormley. 1997. *System dynamics: an introduction*. Prentice Hall.

[105] Rudd, Robert E, and Jeremy Q Broughton. 2005. Coarse-grained molecular dynamics: Nonlinear finite elements and finite temperature. *Physical Review B* 72(14):144104.

[106] Salimbahrami, Seyed Behnam. 2005. Structure preserving order reduction of large scale second order models. Ph.D. thesis, Technische Universität München.

[107] Shapiro, Vadim. 2002. Solid modeling. *Handbook of computer aided geometric design* 20:473–518.

[108] Singleton, Henry E. 1950. Theory of nonlinear transducers. Ph.D. thesis, Massachusetts Institute of Technology.

[109] Stykel, Tatjana. 2002. Analysis and numerical solution of generalized lyapunov equations. *Institut für Mathematik, Technische Universität, Berlin*.

[110] Sudarsan, Rachuri, Steven J Fenves, Ram D Sriram, and Fujun Wang. 2005. A product information modeling framework for product lifecycle management. *Computer-aided design* 37(13):1399–1411.

[111] Tonti, Enzo. 1975. *On the formal structure of physical theories*. Istituto di matematica del Politecnico di Milano.

[112] ———. 2013. *The mathematical structure of classical and relativistic physics*. Springer.

[113] Ulrich, Karl T. 2003. *Product design and development*. Tata McGraw-Hill Education.

[114] Vohralík, Martin. 2010. Unified primal formulation-based a priori and a posteriori error analysis of mixed finite element methods. *Mathematics of computation* 79(272):2001–2032.

[115] Xue, Dingyü, and YangQuan Chen. 2013. *System simulation techniques with matlab and simulink*. John Wiley & Sons.