Multiphysics simulation and model-based system testing of automotive e-powertrains

Borja Rodríguez Frade

Advisors: Francisco Javier González Varela
Miguel Ángel Naya Villaverde

Doctoral thesis

UNIVERSIDADE DA CORUÑA

Programa Oficial de Doutoramento
en Enxeñaría Naval e Industrial

Ferrol, 2021

Dr. Francisco Javier González Varela, Doctor by the University of A Coruña and Dr. Miguel Ángel Naya Villaverde, Doctor by the University of A Coruña, certify that this doctoral dissertation, entitled Multiphysics simulation and model-based system testing of automotive e-powertrains, has been developed by Borja Rodríguez Frade under their supervision in order to obtain the International Doctor mention by the University of A Coruña.

Dr. Francisco Javier González Varela, Doctor por la Universidade da Coruña y Dr. Miguel Ángel Naya Villaverde, Doctor por la Universidade da Coruña, certifican que la presente memoria, titulada Multiphysics simulation and model-based system testing of automotive e-powertrains, ha sido desarrollada por Borja Rodríguez Frade bajo su supervisión para optar al grado de Doctor con mención Internacional por la Universidade da Coruña.

Dr. Francisco Javier González Varela, Doutor pola Universidade da Coruña e Dr. Miguel Ángel Naya Villaverde, Doutor pola Universidade da Coruña, certifican que a presente memoria, titulada Multiphysics simulation and model-based system testing of automotive e-powertrains, foi desenvolvida por Borja Rodríguez Frade baixo a súa supervisión para optar ao grao de Doutor con mención Internacional pola Universidade da Coruña.

Ferrol, 2021.

Borja Rodríguez Frade PhD. student *Doctorando* *Doutorando*	Dr. Francisco J. González Advisor *Director* *Director*	Dr. Miguel Á. Naya Advisor *Director* *Director*

Dedicated to those who have ever believed in me.
Dedicado a todos aquellos que alguna vez creyeron en mí.
Dedicado a todos aqueles que algunha vez creron en min.

Acknowledgments

Since I started this research in 2018, many people have assisted and supported me, and without their contribution this thesis could not be accomplished. It is a pleasure for me to acknowledge those who have taken part in this thesis.

Firstly, I would like to express my sincere gratitude towards my supervisors Francisco Javier González Varela and Miguel Ángel Naya Villaverde for their guidance and help during this period. Additionally, extend this acknowledgement to all my colleagues at Laboratorio de Ingeniería Mecánica (LIM), especially to Francisco Javier Cuadrado Aranda for giving me the opportunity to carry out this research, to Alberto Luaces Fernández for his patience and support offering as many solutions as problems I have found, and also to Antonio Joaquín Rodríguez González, Urbano Lugrís Armesto, Francisco José Mouzo Murujosa and Emilio Sanjurjo Maroño for their suggestions and contributions on the development of the software libraries.

Thanks to the GPPC team from GKN Driveline in Zumaia (Spain), especially to Alberto Peña Rodríguez, Jon García Urbieta, Íñigo García Sierra, Diego Palomar Trullen, and Urtzi Zelaia Larrañaga, for their support in automotive-grade electric motor testing and simulation. I would also like to thank Ideas & Motion HiPERFORM team in Cherasco (Italy), especially to Maurizio Tranchero and Claudio Romano to provide us valuable data about inverter thermal behaviour. Also thanks to Josep Salvador Iborra for his advise on thermal system modelling and his attention during my visit to AVL in Graz (Austria).

In spite of COVID-19 pandemic, I have had the opportunity of carrying out a stay at the Automation and Robotics Laboratory of the Mechanics of Machines Research of the University of Padua (Italy). For this reason, I also want to thank Alberto Trevisani, Dario Richiedei, and Giovanni Boschetti for their guidance during this period, and Giulio Piva, Iacopo Tamellin, Riccardo Minto and Jason Bettega for making my stay in Vicenza more comfortable, as well.

Although the technical support is important, life goes on outside the work. This last part of my acknowledgements is reserved for my family, who have always supported me after all, although some of them have already left us: Begoña, Isabel, Lino, Carlos, Fernando, Maruja, Andrés, Lilí, José, Tamara, Javier, Lurdes, Miriam, and Iraisy Carolina. Thank for all your support, even in the hardest moments.

Agradecimientos

Desde que comencé mi investigación en 2018, mucha gente me ha estado ayudando y apoyándome, siendo su contribución crucial para el desarrollo de esta tesis. Es un placer reconocer a todos aquellos que han participado en ella.

En primer lugar, me gustaría expresar mi agradecimiento hacia mis supervisores Francisco Javier González Varela y Miguel Ángel Naya Villaverde por su ayuda y dirección durante este periodo. Asimismo, extender este agradecimiento a todos mis compañeros del LIM, en especial a Francisco Javier Cuadrado Aranda por darme la oportunidad de realizar esta investigación, a Alberto Luaces Fernández por su paciencia y por ofrecer una solución a los continuos obstáculos que me he ido encontrando, y también a Antonio Joaquín Rodríguez González, Urbano Lugrís Armesto, Francisco José Mouzo Murujosa y Emilio Sanjurjo Maroño por sus propuestas y contribución en el desarrollo de las librerías de software durante esta tesis.

Gracias al grupo de GPPC de GKN Driveline de Zumaia (España), en especial a Alberto Peña Rodríguez, Jon García Urbieta, Íñigo García Sierra, Diego Palomar Trullen, y Urtzi Zelaia Larrañaga, por su ayuda con la simulación y ensayos de los motores eléctricos para automoción. También me gustaría agradecer al equipo de Ideas & Motion de Cherasco (Italia) para el proyecto HiPERFORM, en especial a Maurizio Tranchero y Claudio Romano por proporcionar los datos del comportamiento térmico del inversor. También agredecer a Josep Salvador Iborra por sus consejos en el modelado térmico de sistemas, así como por su tiempo durante mi visita a AVL en Graz (Austria).

A pesar de la pandemia debida al COVID-19, he tenido la oportunidad de realizar una estancia en el Laboratorio de Automatización y Robótica del Grupo de Investigación de Mecanismos de Máquinas de la Universidad de Padua (Italia). Por ello, quiero también agradecer a Alberto Trevisani, Dario Richiedei y Giovanni Boschetti su dirección durante este periodo, así como a Giulio Piva, Iacopo Tamellin, Riccardo Minto y Jason Bettega por hacer mi estancia en Vicenza más agradable.

Aunque el apoyo técnico es importante, la vida va más allá del trabajo. Esta última parte de los agradecimientos está reservada para mi familia, quienes siempre me han apoyado después todo, aunque algunos ya nos hayan dejado: Begoña, Isabel, Lino, Carlos, Fernando, Maruja, Andrés, Lilí, José, Tamara, Javier, Lurdes, Miriam e Iraisy Carolina. Gracias por todo el apoyo recibido, incluso en los momentos más difíciles.

Agradecementos

Dende que comencei a miña investigación en 2018, moita xente estívome a axudar e apoiar, sendo a súa contribución crucial para o desenvolvemento desta tese. É polo tanto un pracer recoñecer a todos aqueles que participaron nela.

En primeiro lugar, gustaríame amosar o meu agradecemento cara aos meus supervisores Francisco Javier González Varela e Miguel Ángel Naya Villaverde pola súa axuda e dirección durante este período. Así mesmo, estender este agradecemento a todos os meus compañeiros do LIM, en especial a Francisco Javier Cuadrado Aranda por darme a oportunidade de realizar esta investigación, a Alberto Luaces Fernández pola súa paciencia e por ofrecer unha solución aos continuos obstáculos cos que me teño atopado, e tamén a Antonio Joaquín Rodríguez González, Urbano Lugrís Armesto, Francisco José Mouzo Murujosa e Emilio Sanjurjo Maroño polas súas propostas e contribucións no desenvolvemento das librerías de software desta tese.

Grazas ao grupo de GPPC de GKN Driveline de Zumaia (España), especialmente a Alberto Peña Rodríguez, Jon García Urbieta, Íñigo García Sierra, Diego Palomar Trullen, e Urtzi Zelaia Larrañaga, pola súa axuda coa simulación e ensaios dos motores eléctricos para automoción. Tamén gustaríame agradecer ao equipo de Ideas & Motion de Cherasco (Italia) para o proyecto HiPERFORM, especialmente a Maurizio Tranchero e Claudio Romano por subministra-los datos do comportamento térmico do inversor. Tamén agradecer a Josep Salvador Iborra polos seus consellos no relacionado co modelado térmico de sistemas, así como polo seu tempo durante a miña visita a AVL en Graz (Austria).

Malia a pandemia da COVID-19, tiven a oportunidade de realizar unha estada no Laboratorio de Automatización e Robótica do Grupo de Investigación de Mecanismos de Máquinas da Universidade de Padua (Italia). Así tamén quero agradecer a Alberto Trevisani, Dario Richiedei e Giovanni Boschetti a súa dirección durante este período, así como a Giulio Piva, Iacopo Tamellin, Riccardo Minto e Jason Bettega por facer a miña estada en Vicenza máis agradábel.

Aínda que o apoio técnico é importante, a vida vai máis aló do traballo. Esta derradeira parte dos agradecementos está reservada para a miña familia, quen sempre apoiáronme ao fin e ao cabo, aínda que algúns deles xa non estean con nós: Begoña, Isabel, Lino, Carlos, Fernando, Maruja, Andrés, Lilí, José, Tamara, Javier, Lurdes, Miriam e Iraisy Carolina. Grazas por todo o voso apoio, mesmo nos momentos máis difíciles.

Abstract

Model-Based System Testing emerges as a new paradigm for the development cycle that is currently gaining momentum, especially in the automotive industry. This novel approach is focused on combining computer simulation and real experimentation to shift the bulk of problem detection and redesign tasks towards the early stages of the developments. Along these lines, Model-Based System Testing is aimed at decreasing the amount of resources invested in these tasks and enabling the early identification of design flaws and operation problems before a full-vehicle prototype is available. The use of Model-Based System Testing, however, requires to implement some critical technologies, three of which will be discussed in this thesis.

The first task addressed in this thesis is the design of a multiplatform framework to assess the description and resolution of the equations of motion of virtual models used in simulation. This framework enables the efficiency evaluation of different modelling and solution methods and implementations. In Model-Based System Testing contexts virtual models interact with physical components, therefore it is mandatory to guarantee their real-time capabilities, regardless of the software or hardware implementations.

Second, estimation techniques based on Kalman Filters are of interest in Model-Based System Testing applications to evaluate parameters, inputs or states of a virtual model of a given system. These procedures can be combined with the use of Digital Twins, virtual counterparts of real systems, with which they exchange information in a two-way communication. The available measurements from the sensors located at a physical system can be fused with the results obtained from the simulation of the virtual model. Thus, this avenue improves the knowledge of the magnitudes that cannot be measured directly by these sensors. In turn, the outcomes obtained from the simulation of the virtual model could serve to make decisions and apply corrective actions onto the physical system.

Third, co-simulation techniques are necessary when a system is split into several subsystems that are coordinated through the exchange of a reduced set of variables at discrete points in time. This is the case with a majority of Model-Based System Testing applications, in which physical and virtual components are coupled through a discrete-time communication gateway. The resulting cyber-physical applications are essentially an example of real-time co-simulation, in which all the subsystems need to achieve real-time performance. Due to the presence of physical components, which cannot iterate over their integration steps, explicit schemes are often mandatory. These, however, introduce errors associated with the inherent delays of a discrete communication interface. These errors can render co-simulation results inaccurate and even unstable unless they are eliminated. This thesis will address this correction by means of an energy-based procedure that considers the power exchange between subsystems.

This research work concludes with an example of a cyber-physical application, in which real components are interfaced to a virtual environment, which requires the application of all the MBST technologies addressed in this thesis.

Resumen

Los ensayos de sistemas basados en modelos emergen como un nuevo paradigma de desarrollo que actualmente está ganando popularidad, especialmente en la industria automotriz. Este nuevo enfoque se centra en combinar la simulación por ordenador con la experimentación para desplazar la mayor parte de la detección de problemas y rediseños hacia las fases tempranas del desarrollo. De esta forma, los ensayos de sistemas basados en modelos se centran en disminuir la cantidad de recursos invertidos en estas tareas y habilitar la identificación temprana de errores de diseño y problemas durante la operación, incluso antes de que los prototipos del vehículo completo estén disponibles. Sin embargo, el uso de esta estrategia requiere implementar algunas tecnologías críticas, tres de las cuales serán tratadas en esta tesis.

La primera tarea abordada en esta tesis es el diseño de un entorno multiplataforma para evaluar la descripción y resolución de las ecuaciones de la dinámica de los modelos virtuales usados en las simulaciones. Este marco permite una evaluación eficiente de las diferentes formas de modelar los sistemas y de los métodos de resolución e implementación. En este contexto de ensayos basados en modelos, los sistemas virtuales interactúan con los componentes de los sistemas físicos, por lo tanto es necesario garantizar sus capacidades de ejecución en tiempo real, independientemente de la plataforma de software y hardware utilizada.

En segundo lugar, las técnicas de estimación basadas en filtros de Kalman son de gran interés en las aplicaciones que usan ensayos basados en modelos para evaluar los parámetros, entradas o estados de los modelos virtuales de un sistema dado. Estos procedimientos se pueden combinar con el uso de gemelos digitales, homólogos virtuales de un sistema físico, con el cual mantienen un flujo bidireccional de intercambio de información. Las medidas disponibles procedentes de los sensores instalados en un sistema físico se pueden combinar con los resultados obtenidos de la simulación del sistema virtual. De este modo, este enfoque mejora el conocimiento de las magnitudes que no pueden ser medidas directamente por los sensores. A su vez, los resultados de la simulación de los sistemas de los modelos virtuales pueden servir para tomar decisiones y aplicar medidas correctivas al sistema real.

En tercer lugar, las técnicas de co-simulación son necesarias cuando un sistema se divide en varios subsistemas, coordinados a través del intercambio de un reducido número de variables en momentos puntuales. Este es el caso de la mayor parte de las aplicaciones que siguen la estrategia de ensayos basados en modelos, en los cuales los componentes físicos y virtuales se acoplan mediante una comunicación en tiempo discreto. Como resultado las aplicaciones ciberfísicas son en esencia un ejemplo de co-simulación en tiempo real, en la que todos los subsistemas necesitan cumplir los requisitos de ejecución en tiempo real. Debido a la presencia de componentes físicos, que no pueden reiterar sus pasos de integración, el uso de esquemas explícitos es frecuentemente necesario. Sin embargo, estos esquemas introducen errores asociados con los retrasos propios de una interfaz de tiempo discreto. Estos errores pueden dar lugar a resultados erróneos e incluso inestabilizar la co-simulación, si no son eliminados. Esta tesis aborda la corrección de la co-simulación a través de métodos energéticos basados en la potencia intercambiada por los subsistemas.

Este trabajo de investigación concluye con un ejemplo de aplicación ciberfísica, en la que se conectan componentes reales a una simulación por ordenador. Esta aplicación requiere la aplicación de las tecnologías de ensayos basados en modelos presentadas a lo largo de esta tesis.

Resumo

Os ensaios de sistemas baseados en modelos xorden como un novo paradigma de desenvolvemento que actualmente está gañando popularidade, especialmente na industria automotriz. Este novo enfoque céntrase en combinar a simulación por ordenador coa experimentación para desprazar a maior parte da detección de problemas e redeseños cara as fases iniciais do ciclo de produto. Deste xeito, os ensaios de sistemas baseados en modelos fundaméntanse en diminuír a cantidade de recursos investidos nestas tarefas e habilitar a identificación temperá de erros de deseño e problemas durante a operación, aínda se os prototipos do vehículo completo non están dispoñíbeis. Porén, o uso desta estratexia require implementar algunhas tecnoloxías críticas, tres das cales serán tratadas nesta tese.

A primeira tarefa tratada nesta tese é o deseño dun entorno multiplataforma para avaliar a descrición e resolución das ecuacións da dinámica dos modelos virtuais empregados nas simulacións. Este entorno permite unha avaliación eficiente dos diferentes xeitos de modelar os sistemas e dos métodos de resolución e implementación. Neste contexto de ensaios baseados en modelos, os sistemas virtuais interactúan cos compoñentes dos sistemas físicos, polo tanto é necesario garantir as súas capacidades de execución en tempo real, independentemente da plataforma de hardware e software escollida.

En segundo lugar, as técnicas de estimación baseadas en filtros de Kalman son de grande interese nas aplicacións que usan ensaios baseados en modelos para avaliar os seus parámetros, entradas ou estados dos modelos virtuais dun certo sistema. Estes procedementos pódense combinar co uso de xemelgos dixitais, homólogos virtuais dun sistema físico, co cal manteñen un fluxo bidireccional de intercambio de información. As medidas dispoñíbeis procedentes dos sensores instalados nun sistema físico pódense combinar cos resultados obtidos da simulación do sistema virtual. Deste xeito, este enfoque mellora o coñecemento das magnitudes que non poden ser medidas directamente polos sensores. Á súa vez, os resultados da simulación dos sistemas dos modelos virtuais poden servir para tomar decisións e aplicar medidas correctivas ao sistema real.

En terceiro lugar, as técnicas de co-simulación son necesarias cando un sistema é dividido en varios subsistemas, coordinados a través do intercambio dun reducido número de variables en momentos puntuais. Este é o caso da maior parte das aplicacións que seguen a estratexia de ensaios baseados en modelos, nos cales os compoñentes físicos e virtuais se acoplan mediante unha comunicación en tempo discreto. Como resultado as aplicacións ciberfísicas son esencialmente un exemplo de co-simulación en tempo real, na que tódolos subsistemas necesitan cumprir os requisitos de execución en tempo real. Debido á presenza de compoñentes físicos, que non poden reiterar os seus pasos de integración, o uso de esquemas explícitos é polo xeral necesario. Con todo, estes esquemas introducen erros asociados cos atrasos derivados dunha interface de tempo discreto. Estes erros poden provocar resultados incorrectos e incluso inestabilizar a co-simulación, de non seren eliminados. Esta tese aborda a corrección da co-simulación a través de métodos enerxéticos baseados na potencia intercambiada polos subsistemas.

Este traballo conclúe cun exemplo de aplicación ciberfísica, na que os compoñentes

reais son conectados a un entorno virtual. Isto require o emprego de tódalas tecnoloxías de ensaios baseadas en modelos presentadas ao longo desta tese.

Contents

List of Figures

List of Tables

Acronyms

AC Alternating Current.

ADAS Advanced Driver Assistance Systems.

API Application Programming Interface.

ARM Advanced RISC Machine.

ASIC Application-Specific Integrated Circuit.

BBB BeagleBone Black.

BDF Backward Differentiation Formula.

BEMF Back-ElectroMotive Force.

CAD Computer-Aided Design.

CCS Column-Compressed Storage.

CFD Computational Fluid Dynamics.

CPS Cyber-Physical System.

CPU Central Processing Unit.

CT Computational Time.

DAE Differential-Algebraical Equation system.

DBC Direct Bonded Copper.

DC Direct Current.

DT Digital Twin.

ECU Electronic Control Unit.

EFE Explicit Forward-Euler.

EKF Extended Kalman Filter.

EV Electric Vehicle.

FEA Finite Element Analysis.

FEM Finite Element Model.

FFT Fast Fourier Transform.

FMI Functional Mock-up Interface.

FMU Functional Mock-up Unit.

FOH First-Order Hold.

FPGA Field-Programmable Gate Array.

GCC GNU Compiler Collection.

GPU Graphics Processing Unit.

GUI Graphical User Interface.

HC Hydraulic Crane.

HiL Hardware-in-the-Loop.

IGBT Insulated Gate Bipolar Transistor.

KCL Kirchhoff's Current Laws.

KF Kalman Filter.

KLU KLU Solver.

KVL Kirchhoff's Voltage Laws.

LIM Laboratorio de Ingeniería Mecánica.

LO Linear Oscillator.

LPTM Lumped-Parameter Thermal Model.

MBD Model-Based Development.

MBS MultiBody System.

MBST Model-Based System Testing.

MNA Modified Nodal Analysis.

MOSFET Metal Oxide Semiconductor Field Effect Transistor.

NA Nodal Analysis.

NLO Nonlinear Oscillator.

NR Newton-Raphson.

ODE Ordinary Differential Equation system.

OOP Object-Oriented Programming.

OS Operating System.

PC Personal Computer.

PISE Parameter-Input-State Estimation.

PMSM Permanent-Magnet Synchronous Motor.

PSD Power Spectral Density.

PWM Pulse-Width Modulation.

R&D Research & Development.

RC Resistor-Capacitor.

RISC Reduced Instruction Set Computer.

RLC Resistor-Inductor-Capacitor.

RPi4 Raspberry Pi 4.

RT real-time.

RWWCT Real World Wall Clock Time.

SBC Single Board Computers.

SCT Simulation Clock Time.

SIFE Semi-Implicit Forward-Euler.

SitL System-in-the-Loop.

SoC System-on-Chip.

SOH Second-Order Hold.

SPKF Sigma-Point Kalman Filter.

STA Sparse Tableau Approach.

TR Trapezoidal Rule.

TRnP Trapezoidal Rule - no Projections.

UDC University of A Coruña.

UKF Unscented Kalman Filter.

VS Visual Studio 2017 Community.

WGN White Gaussian Noise.

WLTP World harmonized Light-duty vehicles Test Procedure.

ZOH Zero-Order Hold.

Chapter 1

Introduction

Vehicles are complex multiphysics engineering systems made up of components with different properties and dynamic behaviour. Mechanical, hydraulic, electric, and electronic effects, among others, play a role in their operation and must be considered during product development cycle. The design of such complex systems is a challenging task due not only to the need to understand accurately these phenomena when considered individually, but also because the interactions between components have a major impact on the overall vehicle performance and behaviour. Traditionally, design concepts and novel component developments have been validated by means of experimental tests with full-vehicle prototypes. Nowadays, however, the automotive industry is undergoing a time of rapid change and profound transformations, in which new mobility solutions, based on electric and hybrid propulsion systems, keep on gaining momentum. Connected and automated vehicles are also quickly becoming a reality in road transportation. The development of these new technologies, as well as the components, materials, and software necessary to enable them, requires intensive and extensive testing procedures to meet the quality and safety standards required by the automotive industry. The consideration of multiple operation scenarios and what-if analyses are of great relevance to ensure the reliable performance of novel technologies that go beyond mere incremental improvements on previously existing, well tested ones. Shifting these evaluations towards the initial stages of product development enables the early detection of design and operation flaws and reduces testing costs, but requires the capacity to conduct experimental validations before a full-vehicle prototype is ready.

Model-Based System Testing (MBST) is emerging as an enabling technology to allow the experimental testing of components and algorithms from the early stages of product development cycle. MBST introduces computational models and simulation as fundamental elements in product validation, as a complement to physical experimentation. This can be done in a variety of ways, as the information flow between simulation and experiments is bidirectional and both can benefit from their mutual interaction [1]. Computer simulation can be employed as a preliminary source of information aimed at streamlining experimental plans, narrowing the range of scenarios to be actually tested on prototypes.

Another possible application would be using a computational model of a system as virtual sensor during an experiment, to gather information that cannot be

directly obtained with sensors on the physical component. Conversely, data from experimental tests can be used to improve computational models and obtain more accurate simulation results.

One of the most challenging applications of MBST is the merging of computer simulation and experimental apparatus into a Cyber-Physical System (CPS) [2] to obtain a test bench in which some components are physical and others are replaced with their computational counterparts. Such cyber-physical test benches offer the possibility to assess critical vehicle parts and their onboard control software, as well as their interaction with the rest of the car and its environment, before a full-vehicle prototype exists. Besides, cyber-physical benches also simplify conducting consistent and repetitive testing under a certain set of conditions, essential to validate hardware and software components, which can be difficult or costly to reproduce in full-vehicle testing, e.g., the replication of dangerous manoeuvres and failure modes.

While it does not remove the need for full-vehicle prototype-based validation, MBST makes it possible to shift the bulk of experimental testing towards the onset of the product development cycle. This accelerates the detection of design flaws and operation problems and results in a reduction of the amount of resources invested in testing procedures [3]. Fig. 1.1 illustrates the comparison between prototype-driven and model-based design approaches, and how the conventional approach identifies defects mostly after assembling a prototype of the complete car, whereas the novel methodology is able to identify them already in the first steps of the project. The early identification of issues has the potential to decrease the need for late product redesign, at stages when project modifications are expensive and time-consuming, and thus to have a positive impact in reducing development expenses.

Testing under the MBST paradigm requires the combination of effective and accurate modelling techniques and information exchange protocols. Computational models are critical in this regard. On the one hand, they must be detailed and accurate enough to guarantee that the results that they deliver are meaningful and reliable. On the other hand, these models have to comply with stringent requirements in terms of predictability, compactness, and efficiency, especially when test environments impose real-time (RT) performance requirements [4], such as in Hardware-in-the-Loop (HiL) and System-in-the-Loop (SitL) setups. RT performance implies not only that a simulation is very fast, but also that its internal time is synchronized with that of the real world; if a misalignment between both clocks occurred, the simulation could no longer be considered RT-compliant.

Another important aspect in MBST applications is the way in which the flow of information between subsystems is managed. The Digital Twin (DT) approach can be used to take advantage of the data generated in cyber-physical experimental setups to enhance the behaviour of both physical components and their computational models. The DT concept includes not only a physical system and its virtual representation, but also the two-way communication and information exchange between them [5], thus extending the meaning of a computational model and at the same time providing a valuable feedback for controlling the physical system in a test bench or during its ordinary operation. Adequate processing of the exchanged data can provide more information about the cyber-physical test bench than the one gathered by the system sensors or obtained from the simulation running on

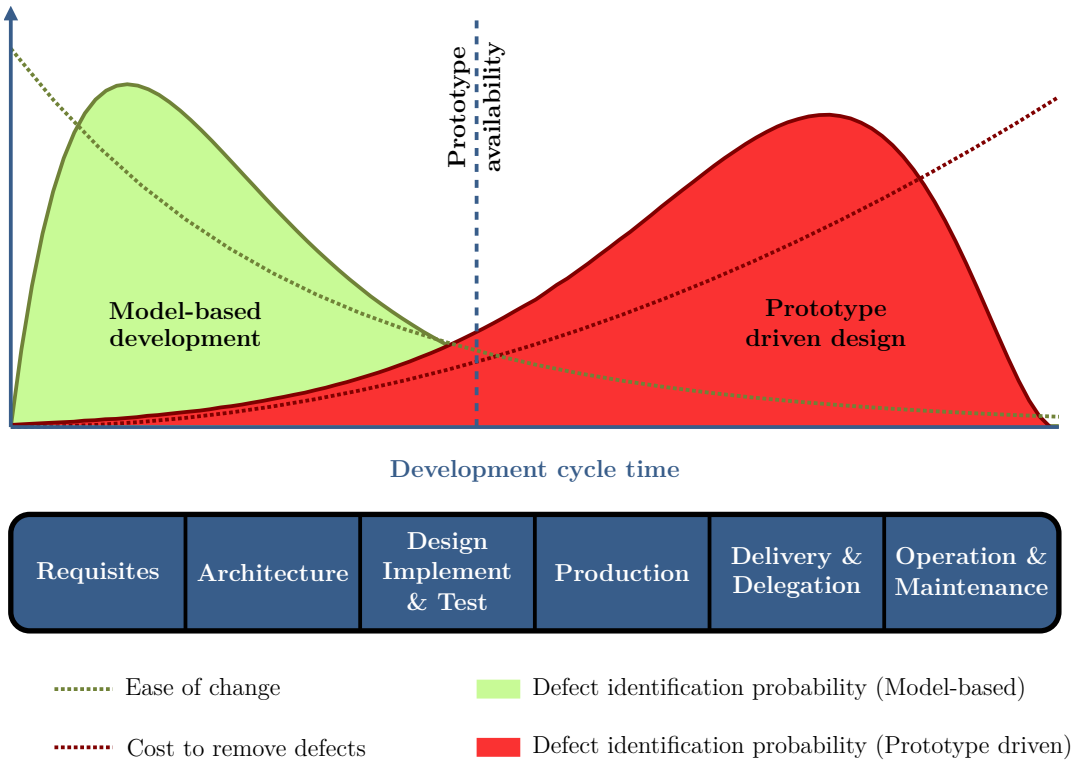


Figure 1.1: Effect on product development cycle of the change of paradigm from prototype-driven development cycle to model-based development cycle.

its own. For instance, the computational models in a DT may be updated and corrected during initialization and at runtime with data from the physical system by means of Parameter-Input-State Estimation (PISE) algorithms [6]. PISE leverages the original concept of state observer, extending it to include also input and parameter estimation, combining to this end sensor readings and computational simulation. Besides delivering simulations that describe more accurately the behaviour of real-world components, PISE-improved DTs can be used to gain insight into the internal state of the physical components in the test bench beyond the readings that conventional sensors can offer.

The data exchange between physical and virtual subsystems also requires the use of appropriate co-simulation schemes to guarantee the stable and accurate execution of the experiments [7]. Co-simulation, or solver coupling, consists in the division of a complex dynamical system into smaller subsystems, each of which features its own dynamics and solver algorithm. Every subsystem proceeds ahead in time independently from its environment, with which it only exchanges information at discrete-time communication points. A master unit, or co-simulation manager, orchestrates subsystem execution and coordinates the information flow between them. Co-simulation lends itself well to multiphysics applications, as it makes it possible to select the most appropriate modelling techniques and solution methods for each component. In the case of MBST test benches, physical and virtual subsystems can be coupled by means of hybrid co-simulation solutions, in which some components are discrete and other continuous. The co-simulation interface for CPS setups also

needs to cope with imperfect transmission of the information [8], different timescales across subsystems, and RT performance requirements. In practical industrial applications, moreover, compliance with a standard interface definition must be used to allow the inclusion of subsystems from different sources in the same co-simulation environment. The Functional Mock-up Interface (FMI) definition has been adopted as de facto standard by a large number of European automotive manufacturers and providers [9]. For this reason, FMI compatibility is advisable in co-simulation solutions for automotive test benches.

As the previous paragraphs illustrate, the MBST paradigm requires the convergence of techniques from different areas in Engineering, ranging from control to high-performance computing. The success of MBST experimental facilities depends on the appropriate design of each component, as well as on the adequate management of the interactions between them.

1.1 Motivation

The Mechanical Engineering Laboratory - Laboratorio de Ingeniería Mecánica (LIM) at the University of A Coruña (UDC) accumulates more than twenty years of expertise in theoretical and applied research in MultiBody System (MBS) dynamics, which have yielded practical results in diverse areas such as biomechanics, machine simulation, and modelling of fishing nets. Techniques and methods for MBS dynamics have also been applied by LIM researchers to automotive systems to address problems such as determining vehicle handling behaviour and designing driving simulators. In the context of this research, the team has worked in experimental validation [10], RT implementations and state observers [11, 12], co-simulation of multiphysics systems [13–15], and embedded systems for data acquisition [16, 17]. This background represents an adequate starting point for applied research on MBST validation methods for the automotive industry.

In particular, the present work stems from a collaboration between LIM and GKN Driveline Zumaia in the context of temperature estimation for automotive electric motors. Along these lines, this research also intends to integrate the knowledge on all the above-mentioned areas of automotive applications and orient it towards its practical use in MBST facilities for the testing and assessment of novel vehicle components. At the same time, it represents an opportunity to adapt the team expertise on automotive dynamics to the current rise of hybrid and electric road vehicles. This demands the development of modelling and simulation methodologies for vehicle components of a non-mechanical nature, e.g., to describe the electronics and thermal effects in e-powertrain components. Work in other areas, such as co-simulation and state observers, builds on previous results obtained at LIM; significant further developments were needed nonetheless to properly address the specific requirements inherent in MBST applications.

1.2 Objectives

The purpose of this thesis is to contribute to the theoretical foundations of MBST via developments in three particular areas, currently open for research: assessment of efficient and RT-capable formulations for multiphysics problems, use of the DT paradigm for RT model update and estimation, and simultaneous operation of physical and virtual components in industrial applications in CPS environments. The main objectives of the thesis can also be classified into three related topics: development of a framework for the assessment of RT-capable formulations, RT algorithms for PISE and their use in DTs, and methods for RT co-simulation.

The above-mentioned general objectives are specified into the following particular objectives:

1. Creation of a benchmark framework to assess the RT performance of a given formulation in an unbiased and straightforward way. This benchmark

   - will allow the evaluation of the RT capabilities of a formulation under test and the selection of the most suitable ones among several options for a specific application, and
   - should be portable, making it possible to use it in different software and hardware environments, e.g., on Windows and Linux, and not only in a Personal Computer (PC), but also in Advanced RISC Machine (ARM) hardware, e.g., in Single Board Computers (SBC). This portability is necessary so that the assessment methodology can be used over the wide variety of hardware platforms that can be used in practical test benches.

2. Development of RT-capable PISE algorithms based on the Kalman Filter (KF), especially for automotive applications, in order to update and correct inputs, parameters, and states of DTs, both during the initialization stage and runtime.

3. Execution of a systematic study of FMI-compliant co-simulation methods, with special focus on the stability, efficiency, and accuracy needs of RT automotive applications. This will be addressed via

   - the implementation of a RT co-simulation framework and master algorithm that can be used in industrial applications,
   - the development of correction methods based on the information carried by the coupling variables exchanged between subsystems to keep co-simulation stable and accurate.

1.3 Thesis structure and contributions

The contents of this thesis have been organized in seven Chapters, as described in the following:

Chapter 1 introduces the outline of this thesis and its motivation. The thesis objectives and its main contributions are briefly described in this Chapter as well.

Chapter 2 summarizes the state of the art of the key technologies required by MBST applications, and details how this concept matches the needs of the industry.

Chapter 3 deals with the evaluation of the RT capabilities of simulation software for multiphysics applications. The main contribution in this Chapter is the definition and implementation of a multiplatform framework to benchmark RT performance. Novel dynamics formulations for electric, electronics, and thermal problems were developed and are put forward in the Chapter and subsequently assessed with this benchmark methodology.

Chapter 4 introduces the concept of DT in CPS testing and describes how its inputs, parameters, and states can be estimated in a RT application from the information delivered by sensors placed on the physical system. This Chapter demonstrates this concept by presenting an PISE algorithm based on KF for the estimation of the thermal behaviour of e-powertrain components.

Chapter 5 discusses co-simulation schemes and guidelines for its use in practical applications, with special consideration of RT setups, together with the need for stabilization and correction methods for explicit co-simulation. The main contributions from this Chapter are the implementation of a RT co-simulation framework and master algorithm compatible with the FMI standard, and the use of energy error indicators to monitor co-simulation quality. Methods to eliminate or reduce energy deviations caused by the discrete-time communication at the co-simulation interface based on such indicators are also presented.

Chapter 6 illustrates how the concepts from Chapters 3, 4, and 5 can be combined to develop MBST equipment to test e-powertrain components. Two SitL test benches for electric motors were designed following the approaches described in the previous Chapters of this thesis. Upon their completion, these are intended to be used for the experimental evaluation of the methods and algorithms presented in this thesis.

Chapter 7 summarizes the main results and conclusions of the thesis and points out possible avenues for future research.

Chapter 2

State of the art

This thesis deals with three critical areas for the introduction of MBST in practical industrial applications. The first is the evaluation and enhancement of the RT capabilities of multiphysics simulation software. Efficient, RT-capable software tools, however, are not enough in most cases to deliver a faithful representation of the dynamics of the system that they model. The exchange of information between physical components and their virtual counterparts is required to obtain high-fidelity descriptions of the system behaviour, and this can be attained via the introduction of the DT concept in MBST setups, which represents the second area discussed in this work. Finally, the execution of virtual models must be synchronized with the performance of physical components; RT co-simulation algorithms are necessary to coordinate and keep stable this process.

The following Sections present an overview of current MBST methodologies, as well as common definitions and strategies used in the context of this methodology.

2.1 Model-based system testing

MBST is currently becoming a strategic technology in the manufacturing industry, in particular in automotive applications [3]. Simulation techniques have been utilized in this context for a relatively long time, serving as a tool to validate and identify component and subsystem parameters, or to explore the effect of design modifications in product performance, among many other uses. Nowadays, however, the engineering systems that are the object of simulation are much more complex and feature a high degree of interdependency between their components. In the case of road vehicles, for instance, their mechanical behaviour cannot be considered separately from the action of the electronic systems that monitor and control them. Safety systems, for example, require the use of programmable electronic controllers to improve their capabilities, which complicates their design, simulation, and test. As this complexity increases, creating and manufacturing prototypes to validate each new design becomes an ineffective development approach [18]. Here lies one of the key points of the MBST approach, the ability to represent complex, multiphysics systems using virtual models that can be interfaced to physical component prototypes. This enables the exploration of a wide variety of new innovative designs, which can be tested through software tools, correcting design errors and malfunc-

tions and improving quality issues while avoiding the need for an actual prototype of the entire automotive system under development.

Nowadays, MBST is spreading in the automotive industry; this approach is not a single tool, but a new testing framework that reformulates how the stages of development have to be carried out. Figure 2.1 summarizes the main technologies on which MBST is built, including control theory, RT systems, virtual sensing, system identification and simulation. This work focuses on RT systems (Chapter 3), virtual sensing (Chapter 4), and simulation (Chapter 5).

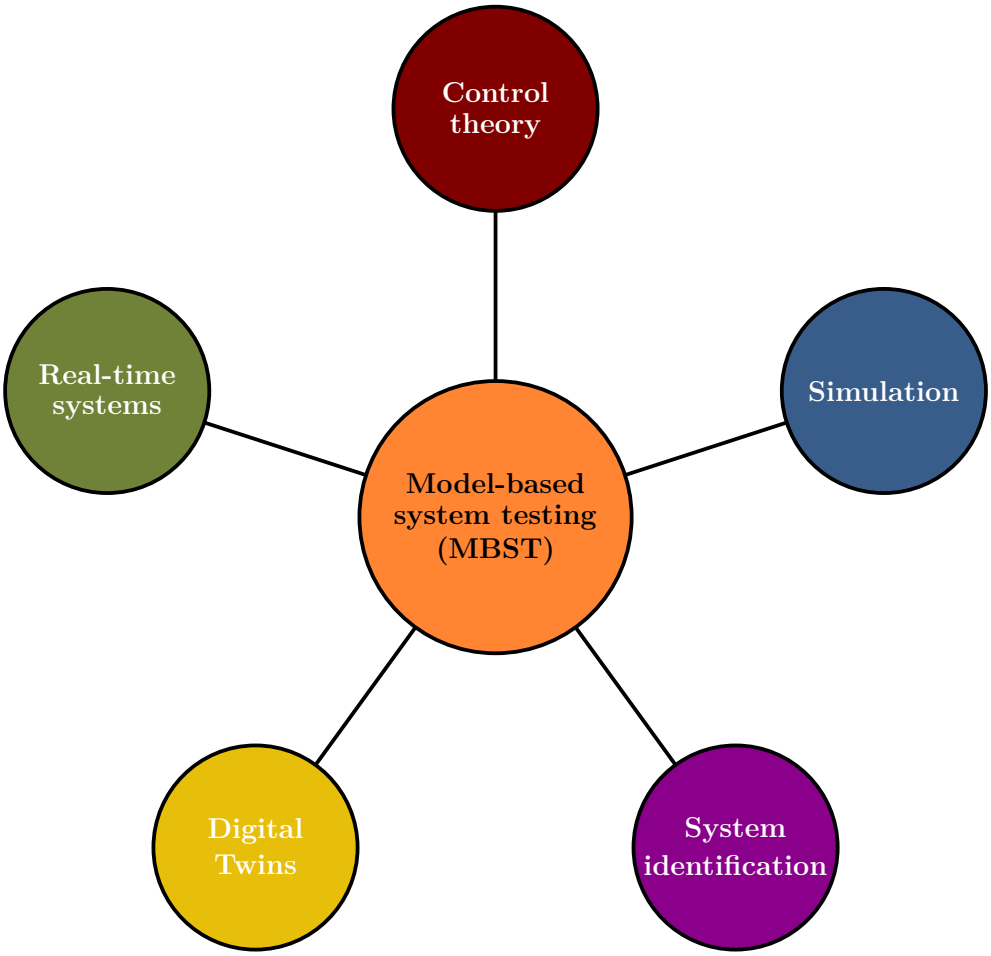


Figure 2.1: Technologies associated with the MBST paradigm.

The MBST paradigm can be applied in different ways to address the particular needs during product development to validate the simulated dynamics through a combination of simulation and test (virtual-only systems, virtual-physical systems and prototypes), which can be classified into three main categories as [1]:

- *Test for simulation:* this first approach uses experimentally obtained data to build, validate, improve, and drive simulation models, its main goal being to improve the accuracy and ensure the consistence of the models throughout a wide range of developments. To accomplish this objective, it is often necessary to collect and process a large quantity of data from testing, in order to, e.g., identify and evaluate system parameters and to finally obtain realistic and

accurate results. An example of this approach can be found in [19], in which an electric drivetrain is modelled using data from experiments with the physical components.

- *Simulation for test:* in this second approach the flow of information goes in the opposite direction: simulation is employed to gather information in order to improve the testing process. Simulation for test procedures provide a better insight into the system and extend data exploitation and the use of models during the lifecycle testing. On the one hand, virtual testing has been used for reducing the high costs of experimentation, e.g., regarding safety-critical parts or hard-to-access components. On the other hand, virtual testing can provide a better insight into the system behaviour. For instance, virtual sensing performed on a simulation of the system behaviour during tests can be used to post-process experimental data and infer quantities of interest, which are not covered by the readings from the sensors used in the experiments and whose direct measurement in a practical setup would be too expensive or even unfeasible. [20] illustrates a simulation-for-test example, in which the Mercedes-Benz C-class electric power steering system is modelled and tested. It also includes a comparison between the model prediction and collected experimental data.

- *Test with simulation:* in this last approach, real and virtual systems interact to complement each other in cyber-physical environments. This allows one to reduce integration problems during product development, such as the risk of late discovery of component and design flaws. Test-with-simulation solutions enable the test of components in laboratories in near real-life conditions by means of RT system simulation [3], in which the component under test is interfaced to a RT simulation of other relevant system components with which it interacts directly, or even the whole system, in a HiL or SitL setup. Thus, experimental component testing can begin before a prototype of the complete system is ready.

  Test with simulation imposes stringent constraints on the computational models that are interfaced to the physical components under test. These must be able to deliver RT performance while keeping a high level of accuracy and remaining representative of true system behaviour. The development of computer models for test-with-simulation applications is a challenging task, which requires efficient modelling approaches and implementation methodologies, e.g., model-order reduction techniques [21].

  For instance, [22] proposes a test-with-simulation application, where a haptic control of a steer-by-wire is used for tracking of target steering feedback torque. This setup provides accurate and realistic force feedbacks to the driver in driving simulators.

The present thesis is oriented towards methodologies for test-with-simulation applications within the MBST paradigm, such as cyber-physical test benches for automotive components. The following Subsections identify three critical areas for the practical introduction of test-with-simulation solutions in industry; these have been selected as the main research avenues of this work. Their importance for the

effective use of MBST solutions is highlighted, and the state of the art is summarized for each of them.

2.2 Real-time simulation

Computer simulation of dynamical systems is currently a necessary and often critical step in a large number of engineering applications [13]. Cyber-physical test benches, as is the case with the majority of test-with-simulation solutions in the MBST paradigm, are paramount examples of this, as RT execution is demanded from the computational models involved, together with an accurate description of real-system behaviour.

The development of computing technologies in the latest decades, both in terms of hardware and software capabilities, has extended significantly the range of problems that can be dealt with by means of simulation, including multiphysics problems of great complexity. Computational efficiency is always a desirable feature in simulation software; RT execution, strictly necessary when physical components are interfaced to virtual environments in HiL and SitL setups, however, goes beyond fast execution. Conducting RT simulation is not equivalent to performing code execution *as fast as possible*. Instead, RT simulation conveys the idea of simulations that are stringently synchronized with an imposed time reference, that of the real world [4]. This reference defines the time-window in which all the computations have to take place [23]. Before continuing, it is important to clarify the different time references involved in a RT simulation:

- Real World Wall Clock Time (RWWCT): is the time reference for the real world clock.
- Simulation Clock Time (SCT): is the time reference for the internal clock of the simulation.
- Computational Time (CT): is the time spent performing the integration and associated numerical computations.

Synchronization is a key issue in RT systems: the exchange of information between components has to occur at the correct time. Synchronism faults may lead to degraded working conditions and results, and even cause damage to the physical components and the operators. For this reason, computer simulations in RT environments have to be not only efficient, but also predictable, in the sense that the elapsed time in computations must match the available time-window imposed by the real-world reference throughout the entire simulation process [8, 24]. Fig. 2.2 is an example of RT simulation, illustrating the synchronization between RWWCT and SCT. Furthermore, the time elapsed in the computations of every step, or CT, has to be strictly smaller than the given time-slot defined by the windows of two RWWCT consecutive steps. Fig. 2.3, on the contrary, describes a non-RT simulation, in which the SCT does not need to be aligned with RWWCT reference. Such an approach should not be used in a HiL or SitL application: inputs from physical components would be received by virtual components at the t_k timestamps defined

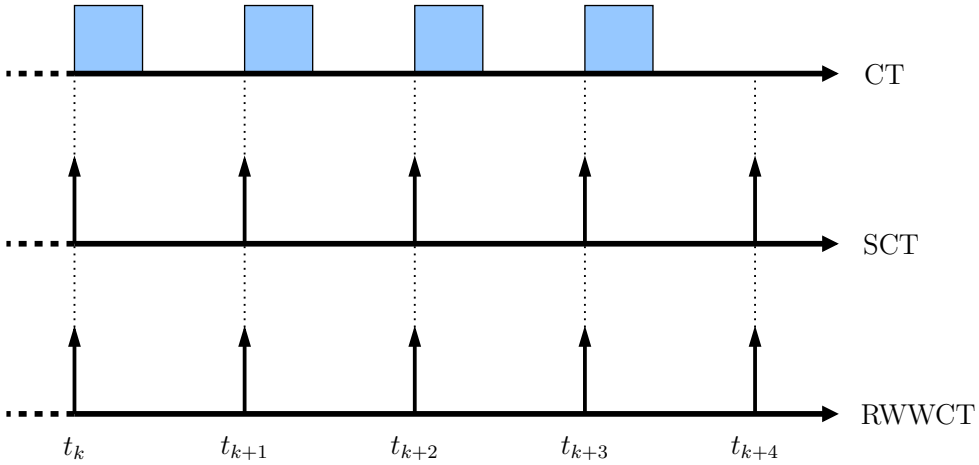


Figure 2.2: Conceptual representation of synchronization in RT simulation: SCT is synchronized with RWWCT and each integration is performed within its time-slot.

by the RWWCT reference, which would not match the internal SCT reference of the simulated environment. In this case, inaccurate results are likely to be obtained, even though computations can be performed within the time-window defined by the SCT timestamps.

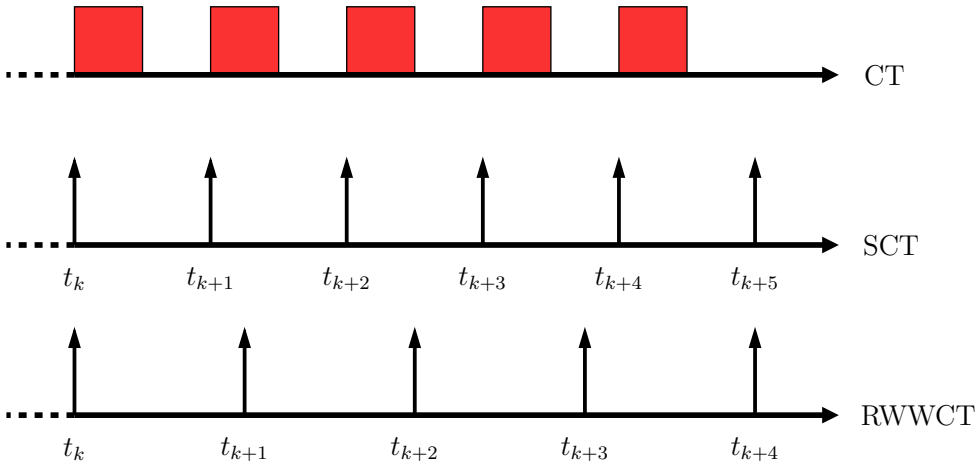


Figure 2.3: Non RT simulation: Synchronization with RWWCT is not necessary. In this case, the SCT goes faster than the RWWCT.

Accordingly, HiL and SitL testing facilities inherently require that every subsystem in the assembly is able to complete the integration of its dynamics equations in RT. The correct and safe performance of the physical components depends on the inputs generated by the virtual ones being delivered at the right time, and vice versa. Cyber-physical test benches for the automotive industry exemplify this statement: the physical vehicle components undergoing assessment are interfaced to a RT simulation of the overall system, which may include mechanical, thermal, electronic, or hydraulic effects, as well as interactions with the driving environment. For the successful operation of such testing platforms, both hardware and software need to be streamlined to contribute to the performant execution of the simulation

codes [25–27].

The memory and computing power of hardware platforms impose limitations on the use of large and detailed computational models during RT tests; however, new hardware and software improvements are continuously pushing forward the simulation limits and redefining what is feasible when combining system simulation and test approach. Nowadays, hardware platforms have been remarkably enhanced and are able to simulate detailed multiphysics systems in reasonable execution times using, to this end, different approaches. Besides the classical solution of using only Central Processing Unit (CPU) resources to deal with computational workloads, today it is possible to use heterogeneous computing platforms, e.g., combinations of CPUs and Graphics Processing Units (GPUs), low-consumption platforms based on ARM chips or similar units, and the use of programmable hardware, such as Field-Programmable Gate Arrays (FPGAs) or Application-Specific Integrated Circuits (ASICs). The GPU approach, for instance, can be used in non-RT simulations, e.g., [28] describes an example, in which a GPU is used for a non-RT Finite Element Analysis (FEA) model; in [29], clusters of GPUs are used for simulating the complex problem of contacts between objects in multibody system dynamics scenarios. GPUs, however, can also be used in RT simulations, in which the execution of computationally intensive tasks can benefit from hardware-based parallelization, e.g., RT shadow generation [30] or RT fluid simulation problems [31]. On the other hand, low-consumption platforms based on ARM chips and FPGAs have been successfully used as accelerators in RT simulation setups. Examples of these implementations can be found in [32], which describes an application in which an ARM-based microcontroller is used for RT simulation of a simple Finite Element Model (FEM), and [33], in which an ARM-based microprocessor deals with the simulation of a RT multibody simulation. New System-on-Chip (SoC) architectures that combine FPGA and ARM platforms have also been employed in a wide range of RT applications, such as multiphysics simulations or detection systems [16, 34–36].

In parallel to the development of reliable and efficient hardware, programming techniques and simulation methodologies have also been developed with the aim to meet the demands of industry and researchers. Examples can be found in a wide variety of areas: efficient algorithms and numerical implementations for multibody system dynamics [37,38], linear algebra methods for the solution of systems of equations [39, 40], and applications that employ FEA [41], to mention only a few.

Cyber-physical test benches for e-powertrains showcase the necessity of combining both efficient hardware and software solutions in a RT environment. Besides the need to employ appropriate computation platforms, they also require the development of effective simulation methods, e.g., for time-domain circuit simulation, that represent engineering systems whose purpose is the management and control of energy flows in industrial applications [25, 42].

When writing software for RT applications, benchmarking is a critical step. Computational efficiency and predictability must be achieved while fulfilling certain precision requirements, which can be verified during this stage using clear validation and assessment criteria [43]. Additionally, the solution of standardized cases with known solutions also serves as a validation check for newly implemented codes. Benchmarking makes it possible to compare several options for the simulation of a

particular problem and provides guidelines for the selection of the most appropriate ones. Benchmark problem sets, clearly defined and easy to replicate, have been put forward in areas like multibody system dynamics, e.g., [44, 45]. They are less common in other areas, like electric and thermal circuit simulation [46], especially when RT behaviour needs to be considered. The proper operation of cyber-physical test facilities, however, requires the knowledge of the simulation capabilities of the software of its virtual environments when it is run on the particular computing platforms employed in the test benches.

2.3 Input-parameter-state estimation

PISE is the second MBST-related technology addressed in this thesis. In many applications, such as automotive safety technologies, it makes it necessary to monitor a large set of variables for improving the comfort and safety of the passengers [47]. All these elements have to satisfy RT constraints, receiving precise and timely information from the real system.

The use of a large set of specific sensors is commonly cost-inefficient and often unfeasible. Hardware and software would also have to be designed for reading and processing this high-rate flow of information coming from the system sensors. Merging data from different real sensors can often introduce low precision results in some demanding working conditions [48]. As an alternative, PISE could be in charge of estimating the magnitudes that cannot directly be measured in the physical system by means of *virtual sensing* [49, 50]. Virtual sensing does not require the use of real sensors, but virtual sensors, which are software programs that emulate the behaviour of a real sensor. These virtual sensors are able to provide indirect measurements of abstract conditions by means of fusing data from a set of physical sensors located at different parts of the real system. For instance, in [51] a virtual sensor is used for determining if a crane exceeds its safe working load by means of physical sensors that can directly measure different magnitudes of the crane. Other examples of virtual sensing can also be found in [52, 53].

Many systems require precise and complete information about the system dynamics. This issue can be addressed by means of the state observers introduced for the very first time with the Wiener filter in the 1940s and the KF in the 1960s. The Wiener filter introduced by Norbert Wiener [54] is focused on a forecasting and smoothing for stationary process problems, whereas the KF introduced by Rudolf E. Kalman [55] extends the use of the Wiener filter to non-stationary processes as well. These state observers are in charge of fusing the data from dissimilar sources, such as real and virtual sensors, in order to provide an optimal estimation of the value not only of the states, but system inputs and parameters. The original KF was widely used in a wide range of applications, because the results obtained are statistically optimal. It assumes that systems are linear and the measurements are also a linear combination of the states with an added Gaussian noise.

Since it is optimal, it remained almost unchanged since it was introduced, however, it has also received contributions to apply it to nonlinear systems. One of the most known variation is the Extended Kalman Filter (EKF), in which the nonlinear transition and observation functions are approximated by a Taylor Series expansion.

Due to the linearization around a working point, this filter may lead to suboptimal solutions. Nevertheless, it is considered the *de facto* standard in nonlinear state estimation. One of the first applications of the EKF was in spacecraft navigation problems [56]. EKF features two important shortcomings that have to be addressed: disregard for the probabilistic spread of the underlying system state and noise, and limited first-order accuracy of propagated means and covariances [57]. Diverse families of KFs have been proposed using deterministic sampling approaches that avoid to calculate the analytical derivatives or Jacobians [58]. This, in turn, allows to classify the KF that use sigma-point approach into a family called Sigma-Point Kalman Filters (SPKFs). A special kind of SPKF is the Unscented Kalman Filter (UKF), in which the location of the sigma-points are selected to capture the most important statistically properties of the system variables [59, 60]. The core of the UKF is an unscented transformation that satisfies: it is easy to perform a nonlinear transformation on a single point, and it is not too hard to find a set of individual points in state space whose sample probability density function approximates the true one of the system state vector [61].

These aforementioned estimators require the use of models to perform the estimation of their states, inputs and parameters. These systems are commonly multiphysics systems, with their behaviour defined by the interaction of mechanical, electronic, thermal, and other phenomena [62]. For instance, this is the case of e-powertrain elements, such as batteries, inverters and electric motors, responsible for the storage and transmission of the vehicle energy to its wheels. The design and operation of e-powertrain components requires the consideration of their multiphysics effects and the coupling between them, as well as the interactions of the components with the rest of the Electric Vehicle (EV) and its environment.

The use of large or high-detailed models, however, hinders the use of in RT applications, such as SitL or HiL. On the other, incomplete information about system dynamics limits the use of some safety systems, such as Advanced Driver Assistance Systems (ADAS), which need to know the current dynamical status [63]. As a consequence, the limited computational power of the computing platforms makes it necessary to reach a trade-off between complexity and accuracy of the models used in RT-performant estimations. One of the simplest models is the so-called random-walk that simply consists of propagating the last value of the states to the next time-step [64]. In spite of the simplicity of this approach and RT capabilities, many application require more accurate models, e.g., multibody models, in order to estimate magnitudes in mechanical systems. For instance, [12] uses a multibody model to estimate states and forces of a four-bar and five-bar linkage. [65] also uses a multibody model of a complete vehicle to estimate the wheel forces and vehicle parameters, such as masses and friction coefficients. [66, 67] use navigation techniques for terrestrial and extraterrestrial applications, parameter estimation solutions for nonlinear dynamics [68], and road vehicle monitoring [69], among many other applications.

Electrical and thermal systems can also take advantage of the use of models to estimate their states or magnitudes, e.g., [70,71] describe electrical models of a three-phase inverter, and Li-ion battery for their use in estimation applications by means of different varieties of KFs. Thermal systems can be modelled in an analogous

way to electrical systems through Lumped-Parameter Thermal Models (LPTMs). A LPTM is a widely used approach to obtain such RT-capable descriptions of the thermal behaviour of the components [72, 73], in the form of a thermal equivalent circuit composed of elementary components such as thermal resistors, capacitors, and sources. These components stand for thermal path resistances, inertias and losses, and they are connected to each other in the form of a Resistor-Capacitor (RC) network. LPTM are modular and can be easily generated using a software library of components; they have also been shown to deliver RT performance when running on platforms with limited computational resources, such as ARM-based single-board computers [74]. Software tools usually formulate the LPTM dynamics using dependent variables as a system of Differential-Algebraical Equation system (DAE), though it is often transformed into a system of Ordinary Differential Equation system (ODE) that is commonly more suitable for effective state and parameter estimation.

LPTMs have special interest, due to thermal effects, in particular, have a critical impact on the behaviour and durability of power electronics hardware and electric machinery [75]. The operation of many components, e.g., in e-powertrain drivetrains must be kept within a range of admissible temperatures to avoid degraded performance and potential failure. Exceeding a certain threshold temperature can, for instance, demagnetize the permanent magnets of an electric motor or cause damage to the semiconductors of an inverter (discussed later on in Chapter 4). Direct measurements of the temperature of these components, however, are often not feasible, as sensors cannot be placed on the most critical locations in many cases [76]. A way to keep track of these critical temperatures during operation is the use of virtual sensors that retrieve results delivered by the simulation of a thermal model of the component. To enable the RT model-based health assessment of these devices, simplified compact representations of the complex thermal behaviour of the elements under study must be used, e.g., by means of model order reduction techniques [77]. On the other hand, the fidelity with which they capture the real system behaviour is critically dependent on the correctness of LPTM topology and the accuracy with which their parameters are determined [78]. LPTMs allow a lower degree of detail compared to FEM; moreover, the characterization of thermal resistances, capacities, and losses is subjected to uncertainties, e.g., those stemming from parameter variation as a consequence of component degradation [76, 79].

LPTM can be used to introduce the consideration of thermal effects experienced by e-powertrain components into their DT, high-fidelity virtual models that are employed to simulate the behaviour of their real-world counterparts and perform RT optimization [2] during product development, testing, and operation. Ideally, DT can be executed in parallel with the physical systems that they represent and used to provide information about their state and performance that cannot be directly obtained in reality, e.g., the junction temperature in power electronics devices or the magnet temperature in electric motors. PISE algorithms enable the use of KFs to estimate parameters with uncertainties using temperature measurements from the device during a preliminary tuning stage. Once the LPTM has been initially adjusted, a two-way communication between DT and their physical environment also allows correcting the operating point and model parameters of the numerical

simulation, preventing it from drifting away from the physical system behaviour due to modelling uncertainties, component degradation, or the accumulation of numerical errors in the integration process. In this case, the data obtained from sensors on the physical system must be fused with the results delivered by the numerical simulation of the DT. The different varieties of the KF [80] are frequently used to this end. Using a KF to fuse results of thermal models and online measurements in power converters to monitor junction temperature has been discussed in a number of publications. In [76], the approach focused on using an estimation that relied on online measurements of the on-state voltage and related it to the junction temperature via look-up tables. In [81], a KF is compared to a trained neural network to estimate the junction temperature of an Insulated Gate Bipolar Transistor (IGBT). Other possibilities include using indirect temperature readings to determine the junction temperature [82]; recent work along these lines [83] makes it possible to account also for input disturbances. Similar strategies can be followed to conduct the modeling and condition monitoring of other e-powertrain components, such as permanent-magnet synchronous motors, e.g., [84, 85].

2.4 Co-simulation

Co-simulation is the third MBST-related technology addressed in this thesis. Co-simulation consists in the simultaneous execution of different solver tools, synchronized via the exchange of *coupling variables* at discrete-time communication points.

When a simulation is performed by one tool with a single and all-encompassing set of equations, the resulting arrangement can be labelled as a *monolithic simulation*. According to this approach, a single solver is in charge of the integration of the entire system. This solver frequently has unlimited access to the details of every component, as these are often required to build and solve the dynamics equations. Monolithic simulation can be used to deal with complex multiphysics problems such as mechatronics and hydraulically actuated devices [86–88]; in this case, the numerical formulation used must be able to handle equations that belong to several physical domains.

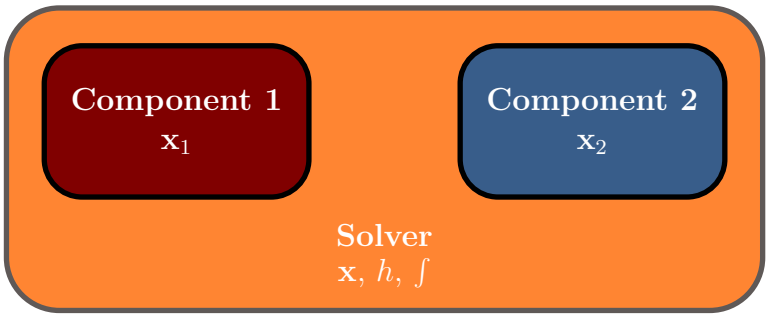


Figure 2.4: Scheme of a monolithic simulation.

Figure 2.4 shows an example of a monolithic simulation environment with two components. The solver integrates the system state $\mathbf{x}$ with a single integration

algorithm; the internal states of each component, $\mathbf{x}_1$ and $\mathbf{x}_2$, are actually subsets of the global state $\mathbf{x}$.

Monolithic solutions can be efficient and reliable, but they also feature drawbacks. Finding a single formulation that is able to deal with dissimilar phenomena, such as electricity, mechanics, or hydraulics, with different time scales and behaviour may not be straightforward. In some cases, a trade-off needs to be made between accuracy and efficiency in one area and others. Besides, the solver needs to have full access to the implementation details of each component. This poses a limitation in industrial applications, when models from different vendors must be simulated together while avoiding the disclosure of internal implementation details, which are often protected by intellectual property rights.

Another possibility is to split the application under study into subsystems, each of which contains its own state and is integrated by its own solver, as in Fig. 2.5, which shows a generic co-simulation example with three subsystems. The exchange of information between the subsystems needs to be coordinated by a co-simulation *manager* or *orchestrator*, responsible for the synchronization of the integration processes of the solvers. The internal states $\mathbf{x}_i$ of subsystem i are not exposed to the other subsystems or the manager; the information provided by each subsystem to its co-simulation environment is limited to a set of output variables $\mathbf{y}_i$. Together with subsystem input $\mathbf{u}_i$, these constitute the *coupling variables* exchanged between the subsystems and the manager at communication points. Between two communication points, each solver proceeds with the integration of its own state without further interaction with its environment, what means that no information about the other subsystems is available until the next communication point. The time interval between two communication points is known as a *macro time-step*.

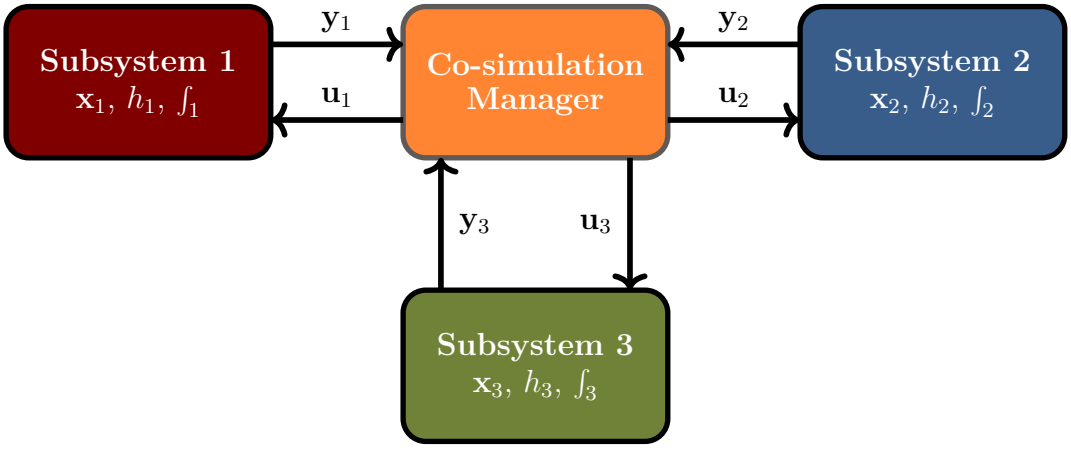


Figure 2.5: Scheme of a three-subsystem co-simulation environment.

Unlike in monolithic simulations, in co-simulation environments each solver can be tailored to the integration of its particular subsystem dynamics. Different time scales and integration step-sizes can be used in the subsystems as well. Additionally, co-simulation setups enable the distribution of the computational workload between several processing units [89, 90], even over network connections. This way, they can be used to improve the computational efficiency by means of parallel execution.

Moreover, there is no need to disclose the internal implementation details or subsystem state to the rest of the application components. Each subsystem effectively behaves as a *black box*, safeguarding intellectual property.

In spite of the above-mentioned advantages, the need to exchange information between subsystems in co-simulation environments gives rise to some issues that do not exist when a monolithic approach is used to simulate an engineering system. From the implementation point of view, it is necessary to define communication standards according to which all the components can exchange information in a unified format [13,91]. This need has been addressed in practice with the definition of the FMI standard [9,92], which specifies data format and exchange interfaces, developed through the collaboration between academic research and industrial groups, with special focus on automotive contexts. However, even if a widely adopted standard for communication is in use, the policy of how to implement and combine it with appropriate integrators and modelling approaches is an ongoing research task [93].

Besides, the discrete-time communication between the co-simulation manager and the subsystems may cause numerical inaccuracies, because the coupling variables are updated only at the communication points. When a subsystem needs to use its inputs to perform a computation at other times, it is necessary to approximate the unknown input values by means of extrapolation techniques [94, 95] or another alternative method [96]. This originates discontinuities in the subsystem inputs at the communication points, which result in inaccurate results and can even lead to the instability of the numerical integration. One of the negative effects of this is the modification of the system dynamics, due to the introduction or removal of energy through the co-simulation interface, which can finally render the simulation unstable if these errors accumulate over time. Even in less severe scenarios, the simulation results may become unreliable, as they are not completely representative of the true system behaviour. As a consequence, this fact has to be taken into account when designing co-simulation interfaces.

2.4.1 Co-simulation in MBST applications

Co-simulation architectures are a necessary component of MBST applications, in which physical components interact with a virtual environment. The latter uses a numerical integrator to evaluate its evolution in time, usually as a discrete dynamical system; physical components, on the other hand, are continuous systems and can be seen as integrated by *the reality*. Both exchange information via a time-discrete interface. Accordingly, the same advantages and issues inherent in co-simulation and mentioned in the previous paragraphs apply to MBST setups as well [97].

Besides, the virtual environments used in MBST setups can be rather complex multiphysics systems in certain cases. For instance, full-vehicle models may be required at different stages of component development, e.g., for electric motor tests, or for design and validation of control or safety systems by means of HiL or SitL setups. These car models can include mechanics, thermal, electric, and hydraulics effects, to name just a few. In such cases, modularity is a convenient feature in virtual environments, in order to be able to integrate models with different level of

detail in the same setup [98,99] or to integrate software tools from different vendors. Co-simulation solutions, as opposed to monolithic ones, lend themselves well to such modular schemes, enabling the quick replacement of components.

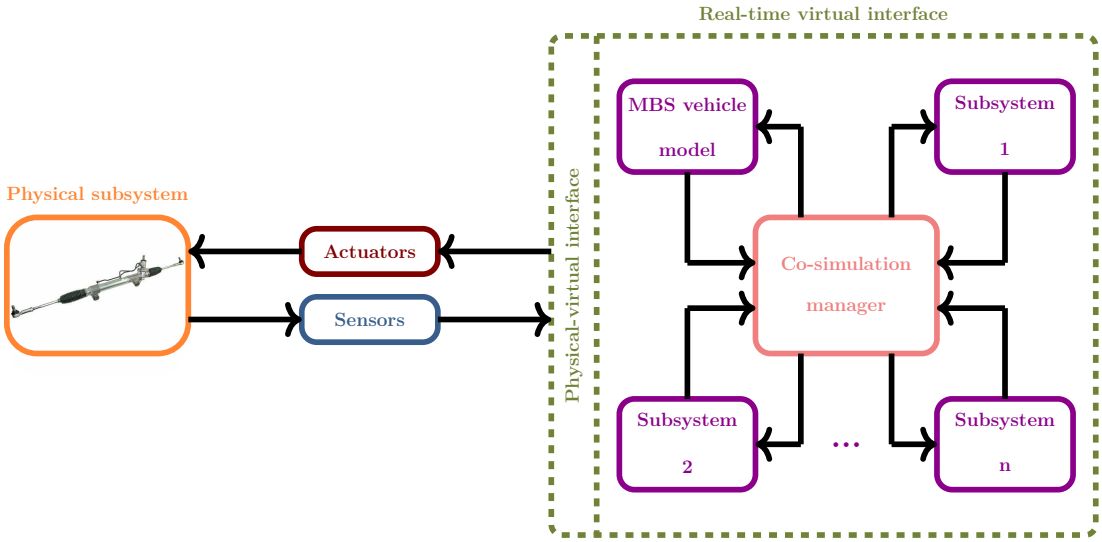


Figure 2.6: Scheme of a generic, RT co-simulation-based SitL testing environment.

Figure 2.6 illustrates these ideas with an example: a MBST test bench for automotive components. An interface is required between the physical subsystem and the virtual environment; moreover, co-simulation can also be used within the latter to coordinate the simulation of different subsystems that represent several physical domains of the model.

MBST applications have a particular set of features that impose certain limitations on the co-simulation techniques that can be used in them. First, discrete and continuous dynamical systems exist in test-with-simulation assemblies. This requires the use of hybrid co-simulation schemes [7,100], able to deal with the different characteristics of each of them. Hybrid co-simulation setups with physical components present additional difficulties with respect to their simulation-only counterparts, such as the presence of noise, data losses, time delays, and other communication problems between the co-simulation manager and the real-world subsystems [8]. Second, the presence of real-world components in the testing setup makes it subjected to RT execution constraints [101], because these systems cannot pause or slow down if the virtual components are not able to complete their integration and information exchange in the time-slot allocated for them [33]. RT execution narrows down the range of co-simulation schemes that can be used in MBST. In the first place, explicit co-simulation schemes are often the option of choice. In an explicit co-simulation arrangement, the subsystems do not retake their macro-steps once they are finalized [102]. Implicit schemes, conversely, are based on the repetition of macro-steps, so that the subsystems can repeat the integration of their dynamics with updated information about their inputs at the next communication point [103]. Physical components are unable to repeat their integration steps, as they can only move forward in time, and so the use of implicit schemes, although not completely impossible, becomes more complicated in MBST environments. There is a second

reason to use explicit co-simulation schemes in RT applications: in some cases, the time constraints imposed by RT execution do not allow for the repetition of macro-steps, because the time invested in computations would exceed the available time-window. Implicit schemes, however, feature a series of advantages over their explicit counterparts. The iterative exchange of coupling variables helps to achieve a more stable numerical integration [104]; simulation results become more accurate and coupling errors are decreased as a result of this iteration. Explicit schemes are prone to the accumulation of coupling errors and may deliver inaccurate results or even go unstable without a proper handling of the exchange of variables at the co-simulation interface.

Among explicit co-simulation methods, Jacobi schemes, in which subsystems can be executed in parallel, are often preferred –although they are not exclusive– in RT applications. They make it possible to use distributed computing environments, in order to divide the whole computational load into different subtasks, which will be computed in separate threads in a single-unit platform or in different hardware units using either homogeneous or heterogeneous computing setups. Configurations such as single- or multi-core CPUs, CPUs and GPUs, or coordinated clusters of CPUs and FPGAs are examples of this [105, 106].

Besides, constant time-steps are often necessary, in order to guarantee one of the (hard) RT requirements: determinism and predictability. Sampling intervals for physical components in testing setups, moreover, often have a fixed length. It is not guaranteed, though, that the dynamics of every subsystem share the same timescale, so multi-rate time grids become necessary when this is not the case. In these cases multi-rate co-simulation schemes are needed to handle this situation [94, 107]. For instance, [87] explores the integration of a mechanic and hydraulic system, in which the mechanics behaviour can be addressed by the use of step-sizes in the order of one millisecond, whereas the hydraulic subsystem is a stiff problem that requires smaller step-sizes. Multi-rate coupling schemes introduce additional complexity in co-simulation setups, and have an impact which is difficult to predict on their stability and accuracy [108–110].

The selection of an appropriate co-simulation methodology is a key element to achieve RT performance while obtaining meaningful and realistic results in MBST cyber-physical applications. On the one hand, in soft-RT applications, in which the simulation may occasionally lag behind the physical time, speeding up the simulation while maintaining the required accuracy levels is advisable. On the other hand, in hard-RT applications, such as HiL and SitL setups, missing synchronization deadlines may cause severe consequences and even system failure [111]. It stands to reason that hard RT applications impose stringent requirements on co-simulation protocols, algorithms, and implementations used in them. At the same time, efficient and predictable code execution must not come at the expense of the accuracy of the results. For this reason, the implementation of co-simulation manager tools for MBST applications must take into consideration the need for monitoring the stability and accurateness of the numerical integration process. At the very least, the manager should be able to detect the unstable behaviour of the co-simulation environment and issue early warnings, so that operations can be stopped before safe functioning is compromised. A further improvement would be having the abil-

ity to detect deviations from true system behaviour as the numerical integration progresses, and to perform corrective actions simultaneously.

A considerable number of publications on how to keep co-simulation stable and accurate have been released in recent years, in particular regarding explicit schemes, and compensate the effect of discontinuities at the discrete-time coupling interface. A possibility is to design special input extrapolation approaches [112, 113]; however, it is difficult to establish general guidelines about the effect of extrapolation techniques on co-simulation behaviour [95]. For instance, using higher-order extrapolation methods reduces co-simulation error in some cases, but increases it in others. Other methods adjust the communication step-size from the information carried by the coupling variables [114] to improve the stability, but such a solution presents problems in RT applications, in which fixed step-sizes are often used. Another option is retrieving information from the partial derivatives of the subsystem states with respect to the coupling variables (*directional derivatives* or simply *Jacobians*) to improve the fidelity of the representation of the interaction between subsystems [115]. Following a similar approach, it is also possible to build *interface models* to provide more accurate estimations of the evolution of subsystem inputs between communication points [15, 96]. This requires certain information from the subsystems internals, which may not always be available, although in some cases it can be estimated from subsystem identification [116].

The above-mentioned methods focus on signal reconstruction. The consideration of system energy opens an alternative path to study co-simulation stability [108]. Discontinuities at the coupling interface cause deviations from the overall system energy balance. If these are properly identified, indicators of the co-simulation quality can be defined. In [117, 118], for instance, the evaluation of the power exchange between subsystems leads to the definition of the power residual as an indicator of the energy that enters or leaves the subsystems through the co-simulation interface. This indicator uses only the information contained in the coupling variables; [117] shows that it can be used to adjust the communication step-size to keep co-simulation stable. Using similar indicators, methods that correct energy deviations in co-simulation assemblies can be defined and implemented, e.g., the correction element in [119], based on the concept of generalized energy, or the correction scheme presented in [109]. The definition and selection of general-purpose monitoring and correction methods for explicit, RT co-simulation remains an open task, especially if the particulars of hybrid co-simulation in cyber-physical environments are to be taken into consideration; advances along these lines represent a valuable contribution to cyber-physical testing applications [120].

Chapter 3

Development and assessment of real-time simulation software

This chapter deals with the first area of interest in MBST environments identified in Chapter 2: the development and evaluation of software tools for RT applications. This is carried out by means of the definition and implementation of a framework for the evaluation of the RT capabilities, efficiency, and accuracy of simulation methods for electric, electronic, and thermal circuits. The conclusions obtained with this framework illustrate the design requirements of RT simulation software; at the same time, the development of circuit simulation tools was of direct interest for the cyber-physical test benches for electric motors that are discussed in Chapter 6 of this thesis.

The proposed framework consists in a series of benchmark examples and a set of criteria and procedures to assess the simulation results. The problems were defined in a simple way to enable their easy replication, and designed to capture at least one challenging aspect of circuit simulation.

Novel dynamics formulations for the general-purpose simulation of circuit dynamics are also presented in this chapter. Their suitability for their execution in RT environments was assessed by means of the above-mentioned benchmarking framework. Several Operating Systems (OSs) and hardware platforms were used to provide a more general scope to the obtained results.

The proposed benchmarking framework makes it possible to compare simulation configurations (required level of accuracy, hardware and OS environment, integrator formula, and formulation) in terms of their impact on the accuracy and computational efficiency of the simulation, evaluating their performance with standard procedures.

3.1 Introduction

The time-domain computer-aided simulation of electric and electronic systems has been the subject of research since the early 1970s [121, 122]. Nonetheless, the need for RT performance has motivated in the last decades the development of simulation tools and platforms able to meet this requirement [123–125]. The system dynamics is often formulated in terms of a set of DAEs, resulting from the application of Kirchhoff's laws and the constitutive equations of the circuit compo-

nents [126]. With appropriate manipulations, numerical methods like the Newmark family of integrators [127] or Backward Differentiation Formula (BDF) [128] can be used to integrate the dynamics. The literature reports different ways to carry out this transformation, e.g., [129–133]. The accuracy and stability of these methods has been assessed in a number of research papers and compared to alternative integration formulas, such as implicit Runge-Kutta methods [134, 135]. Some of these pay special attention to the solution of discontinuous systems that result from switching components, e.g., in electronics circuits [136, 137]. On this matter, numerical methods used for time-domain simulation have been applied to reduced models of dynamical systems, in order to achieve efficient executions. Strategies like index reduction [138] and model order reduction [139–141] have been proposed to streamline the solution of the system equations, also in multiphysics systems.

The capability of the simulation methods to deliver RT execution, however, needs further evaluation. Proper benchmarking of existing techniques requires the definition of clear comparison criteria and standardized problems, representative and easy to replicate [43,44]. Moreover, it is also desirable that the performance of simulation methods can be evaluated using conventional, off-the-shelf PCs and also other platforms, such as ARM processors, with limited computing power and memory, which are currently gaining importance in distributed and heterogeneous applications [4]. The ability to deliver RT performance also depends on the features of the operating system, and so it is interesting to be able to compare different families of OSs, e.g., Windows or based on Linux kernels.

In the particular case of the examples used in this research, besides computational performance aspects, the proposed benchmark problems also enabled the identification of numerical issues in circuit simulation. The use of these problems made it possible to rule out some of the simulation methods under evaluation for general-purpose applications, based on aspects like their inability to deal with redundant algebraic constraints or their tendency to introduce numerical drift in the solution. Problems with the evaluation of the derivatives of the variables were observed with some integrators and a correction strategy consisting on the projection of the derivatives on the constraint manifold was developed to address this issue.

The proposed examples used in this work included linear electric circuits, nonlinear electronics models, and a thermal equivalent model of an electric motor; this latter example extends the scope of the proposed testing framework to systems characterized by slow dynamics and time-varying physical properties [142]. Thermal circuits are modelled in this research by means of the LPTM approach, in which an analogy with electric circuits is used. Node voltages from electric circuits are replaced with temperatures and current flows are also replaced with heat flows through the components.

3.2 Variable selection

Electric, electronic, and thermal subsystems are present in a wide range of multidisciplinary engineering applications, such as EVs. Many of these applications, in which circuit simulation is involved, require to model the systems carefully, especially if RT performance is to be delivered.

A circuit can be described as a series of connected components that obey Kirchhoff's laws and the constitutive component equations, frequently described as a ODE or DAE, see Appendices C and G. The selection of the variables that will be used for formulating these aforementioned equation systems and solving the system dynamics usually has an important impact on its RT performance. This point has already been widely discussed in the literature [122,143], and several different methods have been proposed that use sets of node voltages, component currents, or both, to describe the system dynamics. In the upcoming Sections, the most commonly used methods will be introduced, highlighting their advantages and drawbacks.

3.2.1 Nodal analysis

Nodal Analysis (NA) is based on Kirchhoff's Current Laws (KCL) and uses only node voltages as variables. If the circuit is assumed to have n_V nodes (of which n_{GND} are set to a constant reference value or ground, and n_E is the number of pair of nodes connected to each voltage source), the vector of variables $\mathbf{x}$ is composed of these n_V voltages, where every component of $\mathbf{x}$ represents a voltage node.

For the sake of simplicity, we assume that the n_{GND} nodes connected to a voltage reference and n_E connected to a voltage source are located at the end of the vector $\mathbf{x}$. In these nodes, it is necessary to impose the following constraints

$$V_{\tilde{\alpha}} - V_{\tilde{\alpha}}^{\text{ref}} = 0 \tag{3.1}$$

$$V_{\tilde{\beta}}^{\text{first}} - V_{\tilde{\beta}}^{\text{second}} - V_{\tilde{\beta}}^{\text{source}} = 0 \tag{3.2}$$

where $\tilde{\alpha}$ is an index in the range $\in [n_V - n_E - n_{GND} + 1,\ n_V - n_E]$ and $\tilde{\beta}$ is another index in the range $\in [n_V - n_E + 1,\ n_V]$. The process to obtain the equations of the rest of nodes follows the application of KCL, see Appendix C.1. Once this process finishes, this second set of equations can be expressed in matrix form as

$$\mathbf{Y} \begin{bmatrix} V_1 \\ \vdots \\ V_{n_V} \end{bmatrix} = \begin{bmatrix} I_1 \\ \vdots \\ I_{n_V - n_{GND} - n_E} \end{bmatrix} \tag{3.3}$$

where $\mathbf{Y}$ is the admittance matrix, $V_{\tilde{\alpha}}$ represents the voltage at node $\tilde{\alpha}$ and $I_{\tilde{\alpha}}$ represents the currents from current sources flowing into the node $\tilde{\alpha}$.

NA features a reduced set of variables, which ensures a good use of memory and performance in many topologies. However, when inductors exist in the circuit, the system of equations turns into a system of algebraic and integro-differential equations, which require a somewhat more complicated numerical treatment.

3.2.2 Modified nodal analysis

The Modified Nodal Analysis (MNA) method derives from NA, adding to the vector of variables not only node voltages, but also currents through branches and circuit elements, which have no simple characteristic equations in admittance form [134, 144]. The MNA imposes the KCL on the system, in a similar way to NA, and the component constitutive equations to link node voltages and branch currents.

3.2.3 Mesh analysis

Mesh analysis is useful for small networks and in-hand calculations, but its application is limited to planar meshes. A mesh in an electric circuit can be defined as a loop, a stretch of elements that starts and ends at the same point, which does not enclose other loops. Using this definition in a planar electric circuit, it is possible to identify the n_r loops or meshes that the circuit is composed of. However, this process could be computationally expensive if the circuit comprises a large amount of components.

The basis of the mesh method is to apply the equations of Kirchhoff's Voltage Laws (KVL) to every mesh, using only mesh currents as system variables. Once the KVL are applied to the n_r meshes, the resulting equations are expressed in matrix form as

$$\mathbf{Z} \begin{bmatrix} I_1 \\ \vdots \\ I_{n_r} \end{bmatrix} = \begin{bmatrix} V_1 \\ \vdots \\ V_r \end{bmatrix} \tag{3.4}$$

where $\mathbf{Z}$ is the impedance matrix, $V_{\tilde{\alpha}}$ represents the algebraic sum of the voltage sources in the mesh $\tilde{\alpha}$ and $I_{\tilde{\alpha}}$ represents the current in mesh $\tilde{\alpha}$.

A considerable advantage of this method is that it generates a system of equations with a reduced number of variables; it also suffers from important drawbacks, though. As mentioned, it can only be applied to planar circuits, and the existence of capacitors results into the introduction of integro-differential equations in the formulation of the dynamics.

3.2.4 Sparse tableau approach

The sparse tableau approach uses all the equations that describe the system, including component constitutive equations, fixed value nodes constraints, KCL, and KVL. They are collected into a $n_V + 2n_b \times n_V + 2n_b$ matrix, where n_b is the number of branches of the system [129].

To formulate the aforementioned equations, it is necessary to use as variables the branch voltages and currents and the node voltages, which often results in poor RT performance due to the large size of the involved matrices.

3.2.5 Proposed coordinate selection

In this thesis, the variables used to describe the system are the node voltages V and the currents I that flow through the components. The node voltages of a circuit are grouped in the $n_V \times 1$ term $\mathbf{x}_V$, while the currents that flow through each component are contained in the $n_I \times 1$ term $\mathbf{x}_I$. Together these constitute the system variables

$$\mathbf{x} = \begin{bmatrix} \mathbf{x}_V^{\mathrm{T}} & \mathbf{x}_I^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}} \tag{3.5}$$

which, in general, are not independent, but are subjected to constraints. The total number of variables in the system is $n = n_V + n_I$.

On the one hand, this selection of variables has several advantages, like the following ones:

- it is always possible to formulate the problem, regardless of whether the circuit is planar or not;
- it is not important if a component equation does not exist in impedance or admittance form;
- it is not necessary to retrieve a graph with the connection structure of the circuit; and
- it is guaranteed that the system dynamics is not formulated using integro-differential equations.

On the other hand, the main drawback is the use of as many current variables as basic components involved in the circuit, which could give rise to a poor RT performance. This can be handled, however, removing repeated or redundant variables during a preprocess step. For instance, when there are several circuit components connected in series a single variable could be used to represent the current that flows through all of them.

3.3 Numerical methods

The variables employed with the proposed selection method, Eq. (3.5), are generally not independent. On the contrary, they are subjected to a set of algebraic constraints imposed by the satisfaction of KCL at the circuit nodes, by the specification of the voltage of some nodes, e.g., those connected to ground, and by the constitutive equations of some circuit components, such as resistors. These algebraic constraints can be grouped in an $m \times 1$ term $\mathbf{\Phi}(\mathbf{x}, t)$ where m is the total number of algebraic constraint equations imposed on the system. The system variables $\mathbf{x}$ must, therefore, satisfy

$$\mathbf{\Phi}(\mathbf{x}, t) = \mathbf{0} \tag{3.6}$$

Other components, like capacitors and inductors, introduce differential constraint equations that involve the variables $\mathbf{x}$ and their derivatives $\dot{\mathbf{x}}$ with respect to time. In most cases, these equations can be expressed with the following set of p linear ordinary differential equations

$$\mathbf{\Gamma} = \mathbf{A}\dot{\mathbf{x}} + \mathbf{b} = \mathbf{0} \tag{3.7}$$

where $\mathbf{\Gamma}$ is the $p \times 1$ vector of ODEs, $\mathbf{A} = \mathbf{A}(\mathbf{x})$ is a $p \times n$ matrix and $\mathbf{b} = \mathbf{b}(\mathbf{x}, t)$ is a $p \times 1$ array.

The time-domain simulation of the circuit dynamics requires the solution of the system of DAEs formed by Eqs. (3.6) and (3.7). Several methods exist in the literature to address this problem [128].

3.3.1 First approach: Solvers for systems of ODE

A possibility is to transform the DAE system defined by Eqs. (3.6) and (3.7) into a system of ODEs by differentiating Eq. (3.6) with respect to time to obtain

$$\dot{\mathbf{\Phi}} = \mathbf{\Phi}_{\mathbf{x}}\dot{\mathbf{x}} + \mathbf{\Phi}_t = \mathbf{0} \tag{3.8}$$

where $\mathbf{\Phi_x} = \partial\mathbf{\Phi}/\partial\mathbf{x}$ is the $m \times n$ Jacobian matrix of the algebraic constraints, and $\mathbf{\Phi}_t = \partial\mathbf{\Phi}/\partial t$ is an $m \times 1$ term. Grouping Eqs. (3.7) and (3.8), the following system of ODEs is obtained

$$\left[\begin{array}{c} \mathbf{\Phi_x} \\ \mathbf{A} \end{array}\right] \dot{\mathbf{x}} = \left[\begin{array}{c} -\mathbf{\Phi}_t \\ -\mathbf{b} \end{array}\right], \qquad \text{i. e.,} \qquad \widehat{\mathbf{A}}\dot{\mathbf{x}} = -\widehat{\mathbf{b}} \tag{3.9}$$

In most circuits, the leading matrix $\widehat{\mathbf{A}}$ in Eq. (3.9) is a square non-symmetric matrix of size $(m+p)\times n$. In principle, it is possible to evaluate $\dot{\mathbf{x}}$ from Eq. (3.9) at time step k and integrate its value by means of a numerical integration formula to arrive at the variables $\mathbf{x}$ at time step $k+1$. However, this approach to obtain $\dot{\mathbf{x}}$ suffers from two shortcomings. First, only the derivative-level expression of the algebraic constraints is explicitly enforced by (3.9). Accordingly, the accumulation of integration errors causes the solution to drift away from the exact satisfaction of $\mathbf{\Phi} = \mathbf{0}$. Second, the leading matrix in Eq. (3.9) will not have a full rank if some constraint equations are linearly dependent. This causes the failure of some commonly used linear equation solvers, making it necessary to use special algorithms to arrive at the solution of this system of equations.

3.3.2 Second approach: Solvers for systems of DAE

An alternative approach to solve the system of DAEs under study consists in introducing a numerical integration formula in Eqs. (3.6) and (3.7) to obtain the expression of their satisfaction at the next integration step, $k+1$, as a function of the system variables $\mathbf{x}_{k+1}$, but not their derivatives

$$\left[\begin{array}{c} \mathbf{\Phi} \\ \mathbf{\Gamma} \end{array}\right]_{k+1} = \mathbf{r}\left(\mathbf{x}_{k+1}\right) = \mathbf{0} \tag{3.10}$$

The resulting system of nonlinear equations can be solved by means of Newton-Raphson (NR) iteration

$$\left[\frac{\mathrm{d}\mathbf{r}\left(\mathbf{x}\right)}{\mathrm{d}\mathbf{x}}\right]^{i} \Delta\mathbf{x}^{i+1} = -\left[\mathbf{r}\left(\mathbf{x}\right)\right]^{i} \tag{3.11}$$

$$\mathbf{x}^{i+1} = \mathbf{x}^{i} + \Delta\mathbf{x}^{i+1} \tag{3.12}$$

where i stands for the iteration number. The procedure in Eqs. (3.11) and (3.12) is repeated until convergence is achieved; several options can be used as stopping criterion to determine when this takes place. In this work, the iteration was terminated when the norm of the residual $\mathbf{r}$ in Eq. (3.10) descended below a predetermined admissible value, ε.

Different numerical integration formulas can be used to arrive at Eq. (3.10). BDF are a popular choice in circuit simulation. A BDF is a multistep method that uses values of the system variables computed at previous steps to evaluate the variables and their derivatives at time t_{k+1}. The order ξ of the BDF is the number of already known steps used by the formula. In this research, BDF1, BDF2, and BDF3 methods were tested. It must be noted that the derivatives $\dot{\mathbf{x}}$ are calculated

Table 3.1: Coefficients of BDF integration formulas. Details can be found in Appendix D.

ξ	β_0	β_1	β_2	β_3
1	-1	1	-	-
2	$-3/2$	2	$-1/2$	-
3	$-11/6$	3	$-3/2$	$1/3$

by the method, but they are not used to evaluate $\mathbf{x}$ or $\dot{\mathbf{x}}$ in future time steps. The equilibrium in Eq. (3.10) is expressed as a function of the variables $\mathbf{x}$ alone, replacing their derivatives with

$$\dot{\mathbf{x}}_{k+1} = -\frac{1}{h}\sum_{\tilde{\alpha}=0}^{\xi} \beta_{\tilde{\alpha}}\, \mathbf{x}_{k-\tilde{\alpha}+1} \tag{3.13}$$

where h is the integration step-size, and the $\beta_{\tilde{\alpha}}$ coefficients depend on the order of the BDF [128]; their values are shown in Table 3.1 for BDF orders 1 to 3. With these integration formulas, the system of nonlinear equations to be solved (3.10) becomes

$$\mathbf{r}\left(\mathbf{x}_{k+1}\right) = \left[\begin{array}{c} \boldsymbol{\Phi} \\ \mathbf{A}\left(-\frac{1}{h}\sum_{\tilde{\alpha}=0}^{\xi} \beta_{\tilde{\alpha}}\, \mathbf{x}_{k-\tilde{\alpha}+1}\right) + \mathbf{b} \end{array}\right]_{k+1} = \mathbf{0} \tag{3.14}$$

and the tangent matrix in Eq. (3.11) is

$$\frac{\mathrm{d}\mathbf{r}}{\mathrm{d}\mathbf{x}} = \left[\begin{array}{c} \boldsymbol{\Phi}_{\mathbf{x}} \\ -\frac{\beta_0}{h}\mathbf{A} - \frac{\partial \mathbf{A}}{\partial \mathbf{x}}\left(\frac{1}{h}\sum_{\tilde{\alpha}=0}^{\xi} \beta_{\tilde{\alpha}}\, \mathbf{x}_{k-\tilde{\alpha}+1}\right) + \frac{\partial \mathbf{b}}{\partial \mathbf{x}} \end{array}\right] \tag{3.15}$$

which is a $n \times n$ matrix. This tangent matrix is less likely to be singular than the leading term in Eq. (3.9), due to the presence of term $\partial\mathbf{b}/\partial\mathbf{x}$.

The Trapezoidal Rule (TR), a particular case of the Newmark family of integrators [127], can also be used to obtain Eq. (3.10). The derivatives $\dot{\mathbf{x}}$ at time step $k+1$ are evaluated as

$$\dot{\mathbf{x}}_{k+1} = \frac{2}{h}\mathbf{x}_{k+1} + \hat{\dot{\mathbf{x}}}_k\,; \qquad \hat{\dot{\mathbf{x}}}_k = -\left(\frac{2}{h}\mathbf{x}_k + \dot{\mathbf{x}}_k\right) \tag{3.16}$$

Eq. (3.10) then becomes

$$\mathbf{r}\left(\mathbf{x}_{k+1}\right) = \left[\begin{array}{c} \boldsymbol{\Phi} \\ \mathbf{A}\left(\frac{2}{h}\mathbf{x}_{k+1} + \hat{\dot{\mathbf{x}}}_k\right) + \mathbf{b} \end{array}\right]_{k+1} = \mathbf{0} \tag{3.17}$$

and its corresponding tangent matrix is

$$\frac{\mathrm{d}\mathbf{r}}{\mathrm{d}\mathbf{x}} = \begin{bmatrix} \boldsymbol{\Phi}_{\mathbf{x}} \\ \frac{2}{h}\mathbf{A} + \frac{\partial \mathbf{A}}{\partial \mathbf{x}}\left(\frac{2}{h}\mathbf{x} + \hat{\dot{\mathbf{x}}}_k\right) + \frac{\partial \mathbf{b}}{\partial \mathbf{x}} \end{bmatrix} \tag{3.18}$$

which has a similar structure to that of the tangent matrix in Eq. (3.15).

The TR may suffer from numerical problems in the evaluation of the derivatives $\dot{\mathbf{x}}$. This stems from the fact that Eq. (3.10) imposes the satisfaction of the algebraic constraints $\boldsymbol{\Phi} = \mathbf{0}$, but not their derivative-level expression, $\dot{\boldsymbol{\Phi}} = \mathbf{0}$ in Eq. (3.8); the only conditions that the derivatives $\dot{\mathbf{x}}$ comply with are the p differential equations in (3.7). As a result, term $\dot{\mathbf{x}}_{k+1}$ obtained during the solution of Eq. (3.10) is not unique. The BDF integration evaluates the derivatives $\dot{\mathbf{x}}_{k+1}$ only from the values of the variables $\mathbf{x}$, which are uniquely determined by the satisfaction of $\boldsymbol{\Phi} = \mathbf{0}$. The TR, on the other hand, uses the derivatives in the previous step, $\dot{\mathbf{x}}_k$, to evaluate $\dot{\mathbf{x}}_{k+1}$, as shown by Eq. (3.16); they also affect the evaluation of $\mathbf{x}_{k+1}$ through term $\hat{\dot{\mathbf{x}}}_k$. This may give rise to the accumulation of errors and the steady growth of some elements in term $\dot{\mathbf{x}}$, which may eventually cause numerical inaccuracies or even the failure of the simulation. This problem can be dealt with in a number of ways. In some problems, the errors in the derivative-level constraint equations do not affect noticeably the accuracy of the solution, and can be left untreated. When this is not the case, a possible solution to avoid incurring into these constraint violations is to explicitly include the derivative-level algebraic constraints in Eq. (3.8) in the system of equations to be solved, Eq. (3.10). This presents the disadvantage of making the tangent matrix in Eqs. (3.15) and (3.18) no longer square. An alternative solution consists in the projection of the system derivatives $\dot{\mathbf{x}}$ onto the plane tangent to the manifold defined by the algebraic constraints and the ordinary differential equations. The method is similar to the one described in [145] and [146]. The system derivatives $\hat{\dot{\mathbf{x}}}$ obtained upon convergence of the DAE solver in Eqs. (3.11)–(3.12) do not necessarily satisfy Eq. (3.8). The projection step aims to find the closest set of derivatives $\dot{\mathbf{x}}$ that is compatible with these constraints via the following minimization problem

$$\min \frac{1}{2}\left(\dot{\mathbf{x}} - \dot{\mathbf{x}}^*\right)^{\mathrm{T}} \mathbf{I}\left(\dot{\mathbf{x}} - \dot{\mathbf{x}}^*\right), \qquad \text{subjected to } \left(\widehat{\mathbf{A}}\dot{\mathbf{x}} + \widehat{\mathbf{b}}\right) = \mathbf{0} \tag{3.19}$$

The solution of this problem using a Lagrangian approach leads to the following projection formula

$$\left(\mathbf{I} + \widehat{\mathbf{A}}^{\mathrm{T}}\alpha\widehat{\mathbf{A}}\right)\dot{\mathbf{x}} = \dot{\mathbf{x}}^* - \widehat{\mathbf{A}}^{\mathrm{T}}\alpha\widehat{\mathbf{b}} - \widehat{\mathbf{A}}^{\mathrm{T}}\boldsymbol{\sigma} \tag{3.20}$$

where α is a scalar penalty factor and $\boldsymbol{\sigma}$ is a set of Lagrange multipliers, whose value can initially be set to zero. The leading matrix in Eq. (3.20) is symmetric and semi-positive definite. The projection step can be performed iteratively via the update of the Lagrange multipliers

$$\boldsymbol{\sigma}^{i+1} = \boldsymbol{\sigma}^i + \alpha\left(\widehat{\mathbf{A}}\dot{\mathbf{x}} + \widehat{\mathbf{b}}\right)^i \tag{3.21}$$

and the subsequent re-evaluation of Eq. (3.20).

3.3.3 Initial configuration problem

Every forward-dynamics simulation requires a valid initial system configuration as a starting point. A set of system variables that satisfies the initial problem constraints is necessary at $t = 0$. It is possible, however, that the initial set of variables $\mathbf{x}_0^*$ selected by the analyst is not compatible with the algebraic constraints, so that $\mathbf{\Phi}_0(\mathbf{x}^*) \neq \mathbf{0}$.

Finding the initial set of system variables $\mathbf{x}_0$ requires the solution of the system of nonlinear equations $\mathbf{\Phi}(\mathbf{x}) \neq \mathbf{0}$, which can be attained by means of NR iteration. The increment in the system variables could, in principle, be obtained as

$$\mathbf{\Phi}_\mathbf{x} \Delta\mathbf{x} = -\mathbf{\Phi}_0(\mathbf{x}^*) = -\boldsymbol{\zeta} \tag{3.22}$$

where $\mathbf{x}^*$ is the set of variables obtained in the previous iteration. Matrix $\mathbf{\Phi}_\mathbf{x}$ is not guaranteed to be regular. In practice, a least squares problem is solved as

$$\mathbf{\Phi}_\mathbf{x}^\mathrm{T} \mathbf{\Phi}_\mathbf{x} \Delta\mathbf{x} = -\mathbf{\Phi}_\mathbf{x}^\mathrm{T} \boldsymbol{\zeta} \tag{3.23}$$

Once the increment $\Delta\mathbf{x}$ has been computed, the variables are updated as

$$\mathbf{x}_{i+1} = \mathbf{x}_i + \Delta\mathbf{x}_i \tag{3.24}$$

where subscript i stands for the iteration index.

Upon convergence of the system variables, their derivatives can be directly evaluated using a noniterative procedure. In this step, imposing $\mathbf{\Gamma} = \mathbf{0}$ is mandatory to complete the rank of the linear system. The derivatives of the system variables with respect to time, $\dot{\mathbf{x}}$, can be obtained from the solution of

$$\begin{bmatrix} \mathbf{\Phi}_\mathbf{x} \\ \mathbf{A} \end{bmatrix} \dot{\mathbf{x}} = \begin{bmatrix} -\mathbf{\Phi}_t \\ -\mathbf{b} \end{bmatrix} \tag{3.25}$$

However, in order to make the solution method more robust to the left-hand side matrix losing rank, a least squares approach, similar to the one in Eq. (3.23) could be used here as well.

3.3.4 Implementation

The formulations in this Section were implemented using C++ to build an in-house simulation software tool. The Eigen template library (`http://eigen.tuxfamily.org`) was used to provide the fundamental data structures and algebraic routines; sparse matrices with Column-Compressed Storage (CCS) were used as containers. The KLU Solver (KLU) [147], a direct solver for square non-symmetric matrices, especially conceived for circuit simulation problems, was used for the sparse linear systems in Eqs. (3.9), (3.11), and (3.20). This solver has been shown to be very efficient in the solution of small- and moderate-size problems, whose leading matrix is very sparse, i.e., less than 20% of its elements are structural non-zeros [38]. All the C++ classes and methods to manage information about circuit topology and properties, the numerical solution methods described in this Section, and the control

of code execution were developed and implemented as part of the research work of this thesis.

Most of the matrices required by the methods in this Section have a constant sparsity pattern. This makes it possible to preprocess and reuse the factorizations of the leading matrices required to solve Eqs. (3.9), (3.11), and (3.20). This is also the case of the symmetric rank-k update used to evaluate the leading matrix $\mathbf{I} + \widehat{\mathbf{A}}^{\mathrm{T}} \alpha \widehat{\mathbf{A}}$ in Eq. (3.20) [38]. Moreover, some circuit components such as constant-value capacitors and inductors lead to $\mathbf{\Phi_x}$ and $\mathbf{A}$ matrices that do not need to be re-evaluated at runtime. Unless otherwise specified, this fact was taken into consideration to speed-up code execution.

3.4 Benchmark problems

The numerical methods described in Section 3.3 were evaluated by means of four test problems, which are put forward as benchmarks to evaluate the efficiency and accuracy of time-domain circuit simulations.

3.4.1 Resistor-inductor-capacitor circuit

The first example is a Resistor-Inductor-Capacitor (RLC) circuit with constant coefficients and relatively slow dynamics, shown in Fig. 3.1. As is the case with the rest of benchmark examples in this Section, this problem is intended to be easy to replicate and to implement, while providing relevant information about the performance of the methods under study. Source E_1 is defined by the sinusoidal

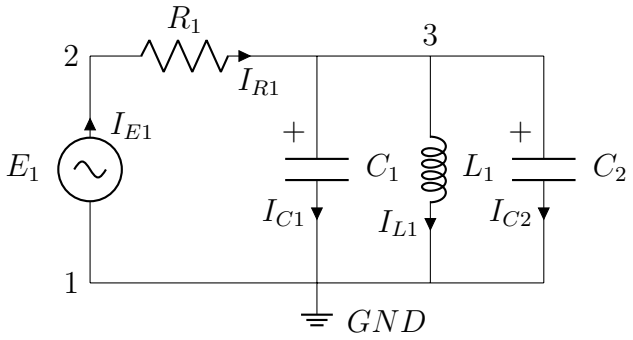


Figure 3.1: Diagram of a RLC circuit with constant coefficients.

function of time $E_1(t) = 100 \sin(10\pi t)$ V. The passive elements in the circuit have constant values $R_1 = 10\ \Omega$, $L_1 = 100$ mH, $C_1 = 400$ mF, and $C_2 = 200$ mF. The benchmark simulation consists in a 10-s integration of the system dynamics, for which the initial values of the current through the inductor, $I_{L1}(0)$, and the voltage of the capacitors, $V_{C1}(0)$ and $V_{C2}(0)$, are set to zero.

In this circuit, capacitors C_1 and C_2 connect nodes 1 and 3 in parallel; their voltages will be related by the two following differential equations

$$C_1 \dot{V}_3 - C_1 \dot{V}_1 - I_{C1} = 0 \tag{3.26}$$

$$C_2 \dot{V}_3 - C_2 \dot{V}_1 - I_{C2} = 0 \tag{3.27}$$

which are linearly dependent in terms of the system derivatives $\dot{\mathbf{x}}$. This means that matrix $\mathbf{A}$ in Eq. (3.7) is row-rank deficient, which may cause some solvers to fail during the solution of the system dynamics. For instance, the leading matrix in Eq. (3.9) becomes singular, thus preventing most conventional linear solvers from finding a solution for the system derivatives.

3.4.2 Scalable resistor-inductor-capacitor circuit

The second example is a scalable RLC circuit with constant coefficients. The number of system variables can be increased by adding new loops to the circuit, as shown in Fig. 3.2. Source E_1 is the sinusoidal function of time $E_1(t) = 100\sin(10\pi t)$ V.

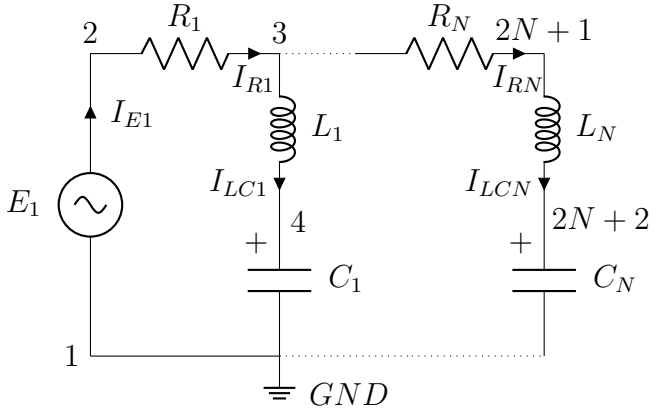


Figure 3.2: Diagram of a scalable RLC circuit with constant coefficients.

The passive elements in the circuit have the same values in every loop, $R_\iota = 10\ \Omega$, $L_\iota = 100$ mH and $C_\iota = 100$ mF, where $\iota = 1 \ldots N$ is the loop number. The benchmark simulation consists in a 1-s integration of the system dynamics, starting from an initial state in which the currents through the inductors, $I_{L\iota}(0)$, and the voltage of the capacitors, $V_{C\iota}(0)$, are set to zero.

This circuit provides a way to test the computational performance of a method as a function of the problem size, which can be adjusted by selecting a number of loops N. In this work, problems with up to $N = 5000$ loops were selected for simulation.

3.4.3 Full wave rectifier circuit

The third example is a model of a full wave rectifier, shown in Fig. 3.3. Source E_1 is defined by the sinusoidal function $E_1(t) = 100\sin(10\pi t)$ V. The passive elements in the circuit have constant values $R_1 = 10\ \Omega$, and $L_1 = 100$ mH. The diodes in this circuit are described using Shockley's model, shown in Fig. 3.4, and defined by the constitutive equation [148]

$$I_D - I_\mathrm{o}\left(e^{\Delta V/(\overline{n}V_\mathrm{t})} - 1\right) = 0 \tag{3.28}$$

where $\Delta V = V_\mathrm{a} - V_\mathrm{c}$ is the voltage difference between the anode and the cathode of the diode, $I_\mathrm{o} = 1$ fA is the reverse bias saturation current, $\overline{n} = 1.5$ is the ideality

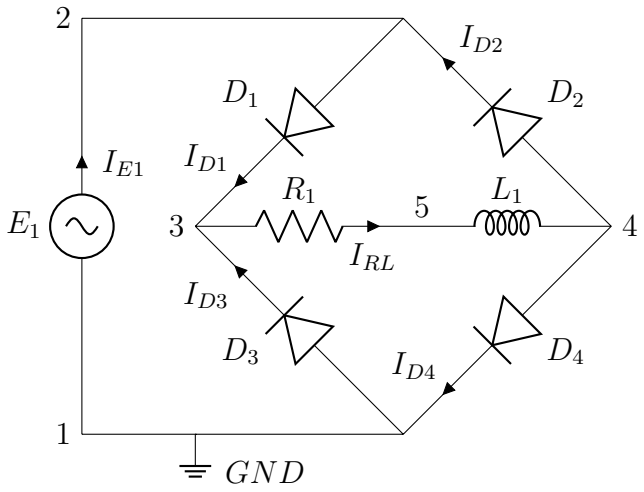


Figure 3.3: Model of a full wave rectifier.

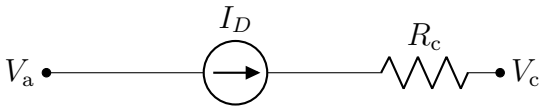


Figure 3.4: Diode model used in the full wave rectifier.

factor, and V_t is the thermal voltage defined by

$$V_t = \frac{\overline{k}\,T}{\overline{q}} \tag{3.29}$$

where $\overline{q}$ is the electron charge, T is the temperature and $\overline{k}$ is the Boltzmann constant. It has an approximate value of 26 mV at 300 K. The series resistor in Fig. 3.4 has a value of $R_c = 1$ mΩ. The benchmark test for this problem consists in a 1-s simulation of the circuit dynamics in which the initial value of the current that flows through the inductor is $I_{RL}(0) = 0$ A.

This problem is an example of nonlinear electronic circuit with fast changes in the system state during the transitions between the forward and reverse regions of the diodes.

3.4.4 Permanent-magnet synchronous motor thermal circuit

The fourth benchmark example is a thermal equivalent circuit to evaluate temperatures and heat flows in a Permanent-Magnet Synchronous Motor (PMSM), shown in Fig. 3.5. Fig. 3.6 represents an example of a PMSM used in automotive applications.

Thermal equivalent circuits describe the heat transfer, thermal losses, and thermal inertia properties of a physical system by means of lumped components comparable to those of an electric circuit. Similar models can be found in the literature, e.g., [73, 149]. The one presented here aims at balancing the complexity of

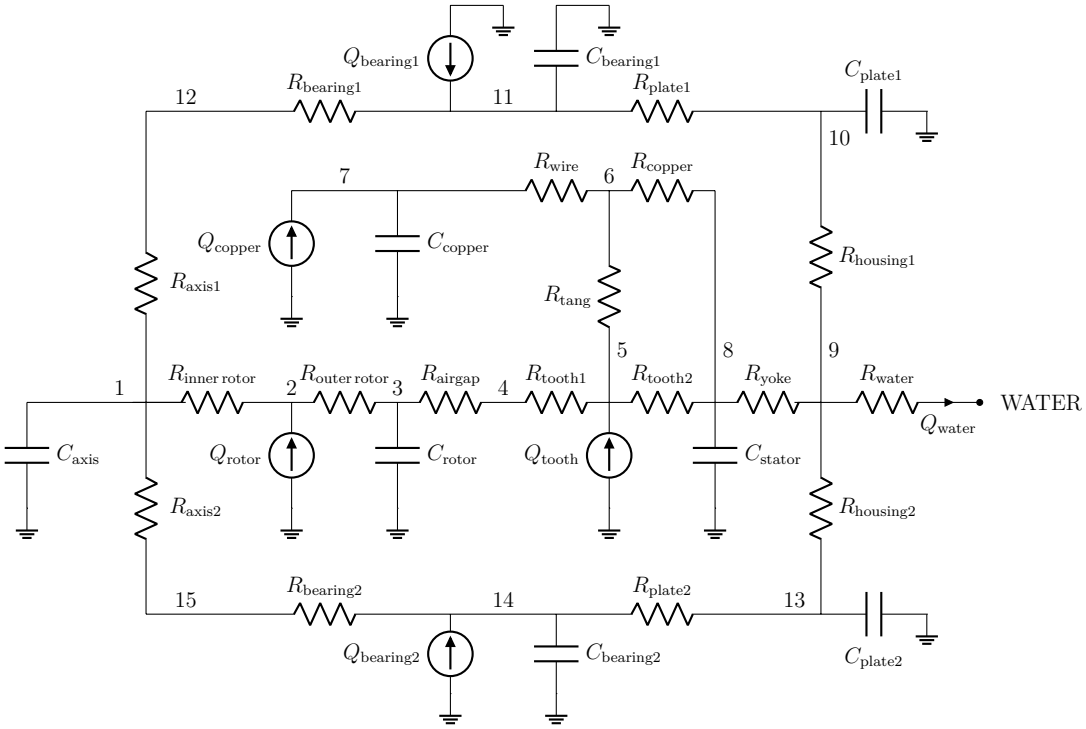


Figure 3.5: Thermal model of a PMSM.

Figure 3.6: A PMSM for automotive applications.

accurately describing the thermal phenomena that occur in a PMSM and the compactness desirable in a benchmark problem. Each node in the circuit corresponds to a representative point in the physical system and has a certain temperature T associated with it. Heat flows Q between nodes play a role similar to currents in an electric circuit. Thermal conductivity and convection are represented with thermal

resistors. Thermal inertia is modelled with capacitors, and heat generation such as Joule effect losses is represented with current sources. The values of thermal resistances, capacitances, and heat sources may be variable if these properties vary as a result of changes in the system temperatures or operation conditions. This is often the case with convective resistors and Joule effect sources. In the thermal

Table 3.2: Parameters of the thermal model of the PMSM.

Resistor	**Value** [K/W]	**Capacitor**	**Value** [J/K]	**Loss**	**Value** [W]
R_{axis1}	1	C_{axis}	$2,000$	Q_{rotor}	150
R_{axis2}	1	C_{rotor}	$5,000$	Q_{copper}	$2,000$
$R_{\text{inner rotor}}$	0.025	C_{copper}	$2,000$	Q_{tooth}	200
$R_{\text{outer rotor}}$	0.015	C_{stator}	$1,500$	Q_{bearing1}	50
R_{tooth1}	0.025	C_{housing}	$3,000$	Q_{bearing2}	50
R_{tang}	0.01	C_{plate1}	300		
R_{tooth2}	0.01	C_{plate2}	300		
R_{copper}	0.001	C_{bearing1}	500		
R_{wire}	0.05	C_{bearing2}	500		
R_{yoke}	0.0025				
R_{housing1}	0.25				
R_{housing2}	0.25				
R_{water}	0.001				
R_{plate1}	0.5				
R_{plate2}	0.5				
R_{bearing1}	3				
R_{housing2}	3				

model of the PMSM in Fig. 3.5, all resistances, capacitances, and heat sources are constant, with the exception of the resistance of the air gap between the rotor and the stator (R_{airgap}), see Appendix B. These parameters are shown in Table 3.2. The air gap resistance is calculated as a function of the Nusselt number, as detailed in the Appendix. The benchmark simulation consists in a 5000-s long integration of the system dynamics. The motor is assumed to operate at a constant angular speed $\omega = 1570$ rad/s. The temperatures of the cooling fluid and the environment are constant, $T_{\text{water}} = 320$ K and $T_{\text{env}} = 300$ K. Initially, the temperature difference between the two nodes connected by a capacitor is zero, which means that the system is at ambient temperature.

This system showcases the characteristics of most thermal equivalent circuits, namely very slow dynamics and physical properties of some components that may vary as the simulation progresses.

3.4.5 Reference solutions of the benchmark problems

Reference solutions are a key component in benchmark problems [44], as they are required to verify the results provided by a certain solver tool, and to compare two or more solutions in terms of accuracy or performance. Analytical closed-form expressions can be obtained for some problems and used as reference solutions. Often, however, these are not available; in such cases, solutions *at convergence* can be used for the same purpose. These can be found by decreasing the integration tolerances and step-size with which the benchmark problem is solved, until the difference between the obtained results remains below a defined threshold. Solutions at convergence should be obtained with a minimum of two different methods.

In this work, reference solutions were obtained by convergence of two of the DAE solvers described in Section 3.3.2, namely the ones that used as integrators the trapezoidal rule and the BDF3 formula, decreasing the integration step-size down to 1 μs. Both methods delivered almost identical solutions in all the problems, with a maximum difference in the obtained values of the system variables $\mathbf{x}$ below $2 \cdot 10^{-8}$ A or V in the electric and electronic circuits. For the thermal circuit in Section 3.4.4, the differences were kept below 10^{-8} K for the temperatures T, and below $4 \cdot 10^{-3}$ W for the heat flows Q. For each benchmark problem, a few

Table 3.3: Number of variables and constraints of proposed benchmark problems.

Problem	**Variables**	**Algebraic ctr.**	**Differential ctr.**
	n	m	p
RLC	8	5	3
Sc. RLC	$3+5N$	$3+3N$	$2N$
Rectifier	20	19	1
Thermal	49	40	9

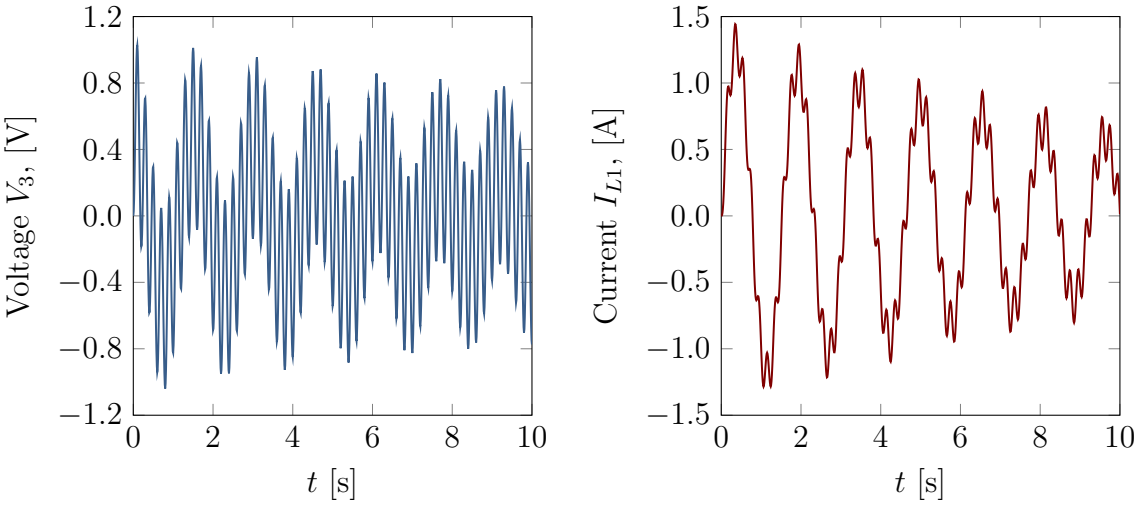
representative variables (*monitored variables*) among the n contained in term $\mathbf{x}$ were selected to enable the fast and simple comparison of the simulation results. This set of chosen variables should describe an important behaviour aspect of the circuit under study, or critical operational values that need to be kept under control. The total number of variables, algebraic, and differential constraint equations of each benchmark problem is summarized in Table 3.3.

3.4.5.1 Resistor-inductor-capacitor circuit

The monitored variables in the RLC circuit in Section 3.4.1 are the voltage at node 3, V_3, and the current through the inductor, I_{L1}. The reference solutions for these values are shown in Figs. 3.7a and 3.7b.

3.4.5.2 Scalable resistor-inductor-capacitor circuit

For the scalable RLC circuit in Section 3.4.2, the monitored variables are the voltage at node $2N+2$ and the current through the last branch in the system, I_{LCN}.

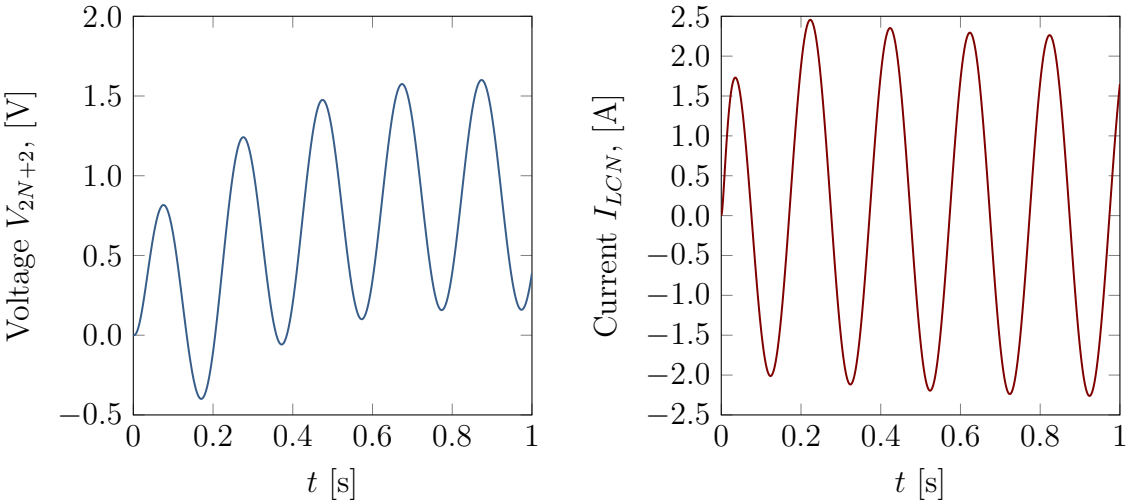


(a) Voltage in node 3, V_3.

(b) Current through coil, I_{L1}.

Figure 3.7: Reference solutions for RLC circuit.

The time-history of these two variables is shown in Figs. 3.8a and 3.8b for $N = 2$ loops.



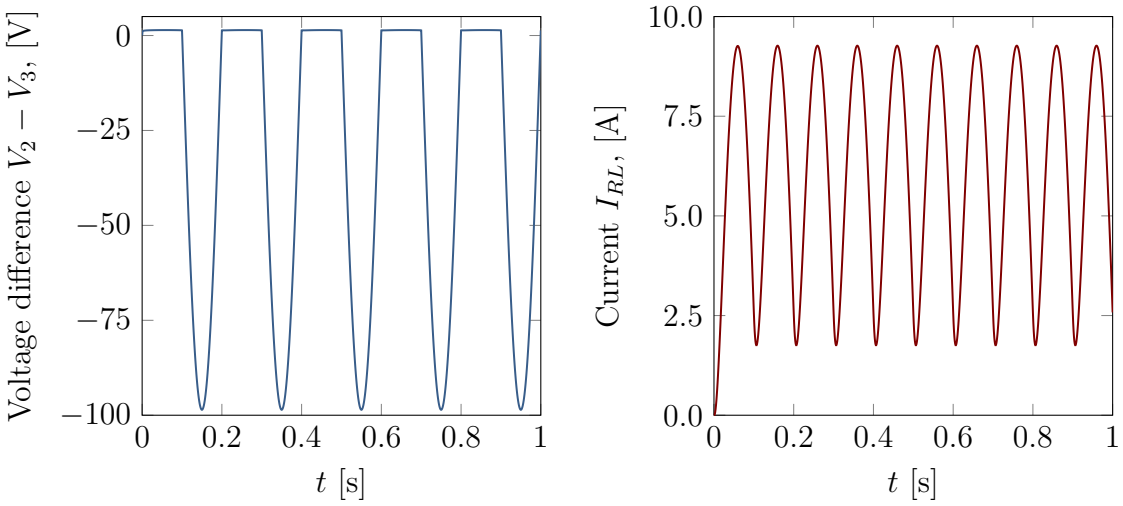
(a) Voltage in node $2N + 2$, V_{2N+2}.

(b) Current through the coil in the last loop, I_{LCN}.

Figure 3.8: Reference solutions for the scalable RLC circuit, with $N = 2$.

3.4.5.3 Full wave rectifier circuit

For the full wave rectifier in Section 3.4.3, the monitored variables are the voltage difference between the anode and the cathode of diode 1, $V_2 - V_3$, and the current through the resistor and the inductance, I_{RL}, shown in Figs. 3.9a and 3.9b respectively.



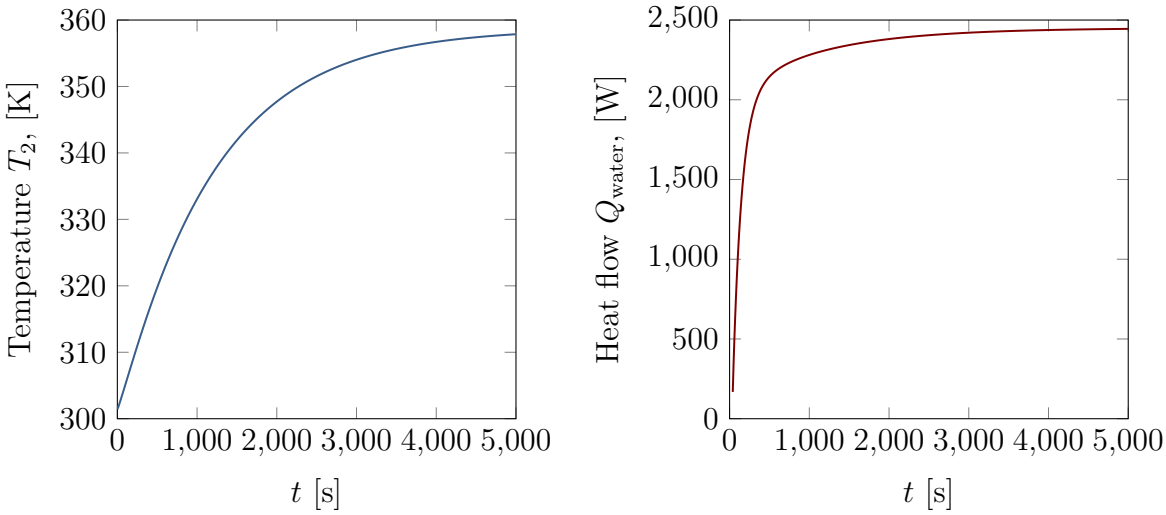
(a) Voltage difference $V_2 - V_3$.

(b) Current through the resistor, I_{RL}.

Figure 3.9: Reference solutions for the full wave rectifier.

3.4.5.4 Permament-magnet synchronous motor thermal circuit

For the thermal circuit of the PMSM, the monitored variables are the temperature of the permanent magnets, T_2, and the heat flow from the housing to the refrigerant, Q_{water}, which are shown in Figs. 3.10a and 3.10b.



(a) Temperature in node 2, T_2.

(b) Heat flow to the cooling water, Q_{water}.

Figure 3.10: Reference solutions for the PMSM.

3.5 Error measurement and performance evaluation

A main goal of benchmark testing is to determine the performance of a given computational method in the solution of representative problems. For this performance evaluation to be meaningful, comparisons between different methods have to be carried out for the same level of accuracy in the obtained results. This accuracy is measured with respect to the reference solutions discussed in Section 3.4.5; doing this requires a standard way to evaluate the error of a solution, which is here adapted from [43].

If, for a given point in time t_p, $y_\chi\left(t_\mathrm{p}\right)$ is the solution delivered by a method under study for a variable χ, and $y_\chi\left(t_\mathrm{p}\right)^{\mathrm{ref}}$ is the reference solution for the same variable, then the error in χ at time t_p can be defined as

$$\Xi_\chi\left(t_\mathrm{p}\right) = y_\chi\left(t_\mathrm{p}\right) - y_\chi\left(t_\mathrm{p}\right)^{\mathrm{ref}}. \tag{3.30}$$

A number n_s of sampling points equally spaced in time is defined for each benchmark problem. The total error in the simulation for variable χ is the aggregate

$$\hat{\Xi}_\chi = \sqrt{\frac{1}{n_s}\sum_{\tilde{\alpha}=1}^{n_s}\left(\Xi_\chi\left(t_i\right)\right)^2}. \tag{3.31}$$

The number of sampling points n_s for each problem is shown in Table 3.4. Two accuracy levels, "low" and "high", were defined for the simulation of the benchmark examples in this Section. These are at least two orders of magnitude larger than the error margins of the reference solutions, pointed out in Section 3.4.5. Using two accuracy levels helps to determine whether the efficiency properties of a given solution method hold when the precision requirements are made more stringent. A simulation is considered acceptable if the error obtained after the numerical integration process, evaluated with Eq. (3.31), is below the thresholds shown in Table 3.5 for every monitored variable.

The comparison of the computational efficiency of two or more simulation methods must be carried out for the same level of accuracy. This criterion was observed in the numerical experiments reported in Section 3.6.1.

Table 3.4: Simulation length and required number of equally spaced sample points for each benchmark problem.

Circuit	**Simulation time** [s]	**Sample points** n_s
RLC	10	1000
Sc. RLC	1	100
Rectifier	1	1000
Thermal	5000	5000

Table 3.5: Acceptable errors in the monitored variables.

Circuit	**Precision**	$\hat{\Xi}_V$ **or** $\hat{\Xi}_T$	$\hat{\Xi}_I$ **or** $\hat{\Xi}_Q$
RLC / Sc. RLC	Low	1 mV	1 mA
	High	10 μV	10 μA
Rectifier	Low	1 mV	1 mA
	High	10 μV	10 μA
Thermal	Low	0.1 K	1 W
	High	0.01 K	0.1 W

3.5.1 Simulation environments

In this research, the use of the benchmark problems and solver implementations was tested under three different computing platforms, namely

- a conventional desktop PC (Desktop),
- a BeagleBone Black (BBB), and
- a Raspberry Pi 4 (RPi4).

The last two environments are ARM-based single-board computers. Testing the benchmark framework and the solver implementations under different software and hardware platforms shows that their application is not restricted to a particular development environment. Moreover, it also serves as a test of the ability of ARM platforms to take on the simulation of particular components in RT computation environments. The features of each simulation environment are summarized in Table 3.6.

Table 3.6: Simulation platform features.

Platform	**CPU**	**Clock freq.** [GHz]	**Cores/ Threads**	**RAM** [GB]	**L1 cache** [kB]	**L2 cache** [MB]	**L3 cache** [MB]	**OS**
Desktop	Intel i7-8700K	3.7	6/12	8	192/192	1.5	12	Windows 10 2019 H2
RPi4	ARM Cortex-A72	1.5	4/4	4	48/32	1	0	Raspbian 4.19.57
BBB	ARM Cortex-A8	1.0	1/1	0.5	32/32	0.25	0	Debian 9.9 IoT

Visual Studio 2017 Community (VS) was used as compiler under Windows, while GNU Compiler Collection (GCC) version 6.3.1 was selected to build the Linux executables for the ARM architectures.

3.6 Numerical experiments

The benchmark problems presented in Section 3.4 were used to compare the performance and features of the formulations in Section 3.3. This comparison was

carried out taking into consideration the particular requirements of RT simulation. The time invested by the numerical methods in the integration of a time-step must be not only as short as possible, but also predictable. This means that the number of iterations required for the convergence of the NR iteration to solve Eq. (3.10) must be limited. In principle, the number of iterations could be decreased for those steps that are computationally less challenging, e.g., the intervals between switching points in the rectifier circuit in Section 3.4.5.3. This would result in overall shorter computation times. However, most RT applications require that every integration step is solved while meeting RT deadlines. Accordingly, in this work, the number of solver iterations was set to a fixed value, equal for all the integration steps in every method and problem.

The methods in Section 3.3 were compared in terms of the computation time that was needed to complete the simulation of the benchmark examples within the precision requirements specified in Table 3.5. Three parameters were adjusted for each method in order to determine its optimal performance in each case: the integration step-size h, the number of solver iterations per time step, γ, and the integrator tolerance, i. e., its maximum admissible error during iteration, ϵ. Two cases were distinguished when the TR was used as integrator: one in which one projection step was performed upon convergence of the NR iteration, reported as TR in the following, and another one in which this projection was omitted, labelled Trapezoidal Rule - no Projections (TRnP). This makes it possible to determine the impact of the projection step on computation times.

3.6.1 Numerical results

The set of solver parameters, ϵ, h, and γ, was adjusted for each solver and test problem to find out the configuration that delivered the best performance. It was found that decreasing the maximum admissible norm of the residual ϵ below 10^{-3} did not result in significant accuracy improvements in any of the examples, and so it was fixed to this value in all the tests. It must be noted that ξ-order BDF methods need a special initialization during their first ξ steps. These methods use the previous ξ values of the variables $\mathbf{x}$ to take an integration step; during initialization these are not available, and so a modified version of the algorithm must be used. In this work, the TR was use as replacement method for these initial steps. On the other hand, the accuracy requirements for low- and high-precision solutions displayed in Table 3.5 were enforced in all cases.

3.6.1.1 Resistor-inductor-capacitor circuit

The direct ODE integration approach in Section 3.3.1 is unable to perform the simulation of the RLC circuit in Fig. 3.1. Capacitors C_1 and C_2 connect nodes 1 and 3 in parallel, making the leading matrix $\widehat{\mathbf{A}}$ in Eq. (3.7) rank deficient. Conversely, all the DAE solvers in Section 3.3.2 were able to complete the integration while attaining the required precision.
Tables 3.7 and 3.8 show the selected DAE integrator configurations (step-size h and number of iterations γ) that resulted in the most efficient solution of the RLC circuit simulation while meeting the low and high precision requirements, respectively.

Table 3.7: RLC circuit: Low-precision configuration of the DAE integration methods.

Method	h [ms]	γ	$\hat{\Xi}_V$ [mV]	$\hat{\Xi}_I$ [mA]
TR	2.5	2	0.25	0.38
BDF1	$2.5 \cdot 10^{-2}$	1	0.30	0.63
BDF2	2.5	2	0.84	0.76
BDF3	2.5	2	0.14	0.33

Table 3.8: RLC circuit: High-precision configuration of the DAE integration methods.

Method	h [ms]	γ	$\hat{\Xi}_V$ [μV]	$\hat{\Xi}_I$ [μA]
TR	0.25	2	2.5	3.8
BDF1	$2.5 \cdot 10^{-4}$	1	3.0	6.3
BDF2	0.25	1	8.4	7.6
BDF3	0.25	1	1.3	3.3

As expected, the errors in the monitored variables scale linearly with the integration step-size h for each method. As shown in Table 3.8, when BDF integrators are used, decreasing the step-size to achieve high precision makes it possible to reach the desired error levels with a single iteration of the solver. The TR, on the other hand, still required two iterations with the same step-size $h = 0.25$ ms. Tables 3.7 and 3.8

Table 3.9: Elapsed times in the simulation of the RLC circuit.

Method	**Prec.**	Elapsed time Desktop [ms]	Elapsed time BBB [ms]	Elapsed time RPi4 [ms]
TR		6.5	176.8	50.6
TRnP		3.0	92.6	34.85
BDF1	Low	190.6	4,988.0	815.0
BDF2		3.2	108.0	38.5
BDF3		3.4	112.8	39.6
TR		69.4	1,708.1	307.0
TRnP		32.3	856.7	162.3
BDF1	High	16,764.9	496,852.0	77,287.4
BDF2		23.3	587.8	128.6
BDF3		24.8	636.3	133.6

must be interpreted in the light of the results contained in Table 3.9, which shows

the times elapsed in the simulation of the RLC system with each DAE integrator, with the configurations defined in Tables 3.7 and 3.8. Table 3.9 shows the efficiency delivered by each method for the two accuracy levels, namely high and low, in the solution of the test problem.

With the exception of BDF1, all methods would be able to deliver RT performance in the three computing platforms described in Table 3.6, both for low and high precision. Computations on the BBB and the RPi4 were about 30 and 5 times slower than on the PC, respectively. Moreover, results show that the projection stage after converge of the TR required as much computation time as the convergence of the NR iteration of the main solver. BDF2 and BDF3 delivered the best efficiency in this test problem. It must be mentioned that the tangent in Eq. (3.11) is a constant matrix for this example. This makes it possible to evaluate and factorize it during preprocess; accordingly, only the back-substitution required for the solution of Eq. (3.11) needs to be performed during runtime.

3.6.1.2 Scalable resistor-inductor-capacitor circuit

The scalable RLC circuit in Fig. 3.2 was solved using the DAE integration methods in Section 3.3.2. The integration step-sizes h and number of iterations required by each solver are shown in Tables 3.10 and 3.11 for low and high precision, respectively.

Table 3.10: Scalable RLC circuit ($N = 2$ loops): Low-precision configuration of the DAE integration methods.

Method	h [ms]	γ	$\hat{\Xi}_V$ [mV]	$\hat{\Xi}_I$ [mA]
TR	2.5	2	0.41	0.94
BDF1	$2.5 \cdot 10^{-2}$	1	0.15	0.65
BDF2	1	2	0.14	0.60
BDF3	2.5	2	0.38	0.54

Table 3.11: Scalable RLC circuit ($N = 2$ loops): High-precision configuration of the DAE integration methods.

Method	h [ms]	γ	$\hat{\Xi}_V$ [μV]	$\hat{\Xi}_I$ [μA]
TR	0.25	2	4.0	9.6
BDF1	$2.5 \cdot 10^{-4}$	1	1.5	6.5
BDF2	0.1	2	1.4	5.9
BDF3	0.25	2	3.7	2.0

Tables 3.10 and 3.11 were obtained for a circuit with $N = 2$ loops. However, it

was verified that these configurations were also able to keep simulation errors with respect to the reference solution below the admissible threshold for circuits with up to $N = 5000$ loops. Accordingly, these configurations were used for all the numerical experiments reported in this Subsection.

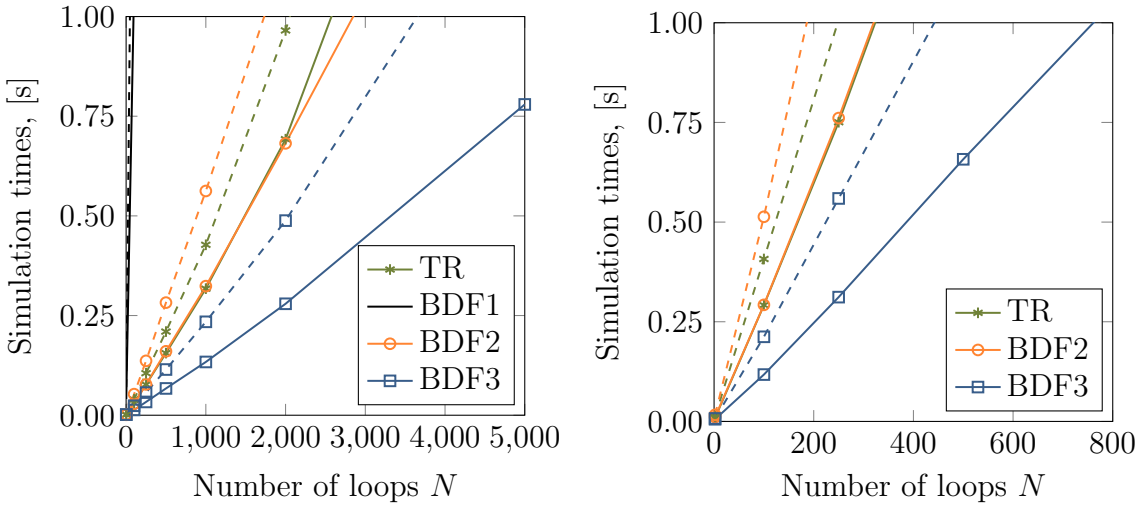
Table 3.12 shows the times elapsed by each method in the solution of the RLC circuit with $N = 2$ loops. As expected, they feature similar trends to those obtained in Section 3.6.1.1 with the simple RLC circuit. The tangent matrix in Eq. (3.11) is constant in this problem as well, and so the times in Table 3.12 do not include its evaluation and factorization. The simulation of this RLC circuit was repeated

Table 3.12: Elapsed times in the simulation of the scalable RLC circuit ($N = 2$ loops).

Method	**Prec.**	Elapsed time Desktop [ms]	Elapsed time BBB [ms]	Elapsed time RPi4 [ms]
TR	Low	0.9	32.9	10.4
TRnP		0.4	21.5	4.6
BDF1		24.0	713.3	153.7
BDF2		1.1	40.0	12.4
BDF3		0.5	21.7	5.3
TR	High	9.8	236.6	82.8
TRnP		3.9	116.1	37.6
BDF1		2,191.3	65,542.3	9,977.6
BDF2		10.2	337.3	74.2
BDF3		4.3	143.6	39.6

varying the number of loops ($N = 2$, 100, 250, 500, 1000, 2000, 5000) to gain insight into the effect of system size on computational efficiency. These numerical experiments were conducted for both low and high precision. Moreover, an additional set of tests was performed forcing the simulation software to ignore the fact that the tangent matrix is constant and to perform its evaluation and factorization in every solver iteration. Figures 3.11a and 3.11b display the elapsed times delivered by each solution approach in the PC simulation platform, for low and high precision respectively. Elapsed times for the TR include carrying out the velocity projections. Results for BDF1 were omitted in Fig. 3.11b because the computation times that it required exceeded considerably the ones delivered by the other methods. Figs. 3.11a and 3.11b show the computational overhead due to the evaluation and factorization of the tangent matrix. They also show that the increase of elapsed time with the number of loops N (and, as a consequence, with the number of variables n) is approximately linear. This is the result of using sparse matrix implementations and solvers, and agrees with previous results in different simulation domains [38].

The maximum theoretically achievable system sizes that could be simulated while fulfilling RT execution requirements are shown in Table 3.13, for the PC simulation environment. BDF1 experienced convergence issues for large systems in the high-precision simulation, and so results for this method are omitted in this case. Results



(a) Low precision scenario.

(b) High precision scenario.

Figure 3.11: Scalable RLC circuit: Simulation time as a function of the number of loops (N). Simulations with a constant tangent matrix are represented with solid lines; dashed lines mean refactorization of the tangent matrix in each solver iteration.

Table 3.13: Scalable RLC circuit: Maximum system size compatible with RT execution on the PC simulation environment.

Method	**Precision**	**Loops** $[N]$	**System size** $[n]$
TR	Low	2,053	10,268
BDF1		48	243
BDF2		1,790	8,953
BDF3		3,661	18,308
TR	High	248	1,243
BDF1		-	-
BDF2		186	933
BDF3		447	2,238

in Table 3.13 include the evaluation and factorization of the tangent matrix. They provide an orientation regarding the maximum system size that the tested methods would be able to handle in a RT simulation environment with the computing power of the PC in Table 3.6.

3.6.1.3 Full wave rectifier circuit

The full wave rectifier was solved using the DAE integration methods. The integration step-sizes h and number of iterations required by each integrator are shown in Tables 3.14 and 3.15 for low and high precision, respectively. Table 3.16 contains

Table 3.14: Full wave rectifier: Low-precision configuration of DAE integration methods.

Method	h [ms]	γ	$\hat{\Xi}_V$ [mV]	$\hat{\Xi}_I$ [mA]
TR	0.5	16	0.17	0.29
BDF1	0.01	3	0.47	0.03
BDF2	0.25	7	0.18	0.19
BDF3	0.1	9	$4.0 \cdot 10^{-3}$	$7.9 \cdot 10^{-3}$

Table 3.15: Full wave rectifier: High-precision configuration of DAE integration methods.

Method	h [ms]	γ	$\hat{\Xi}_V$ [μV]	$\hat{\Xi}_I$ [μA]
TR	0.05	6	2.0	1.8
BDF1	10^{-4}	1	4.7	0.3
BDF2	0.05	9	7.9	6.5
BDF3	0.1	9	4.0	7.9

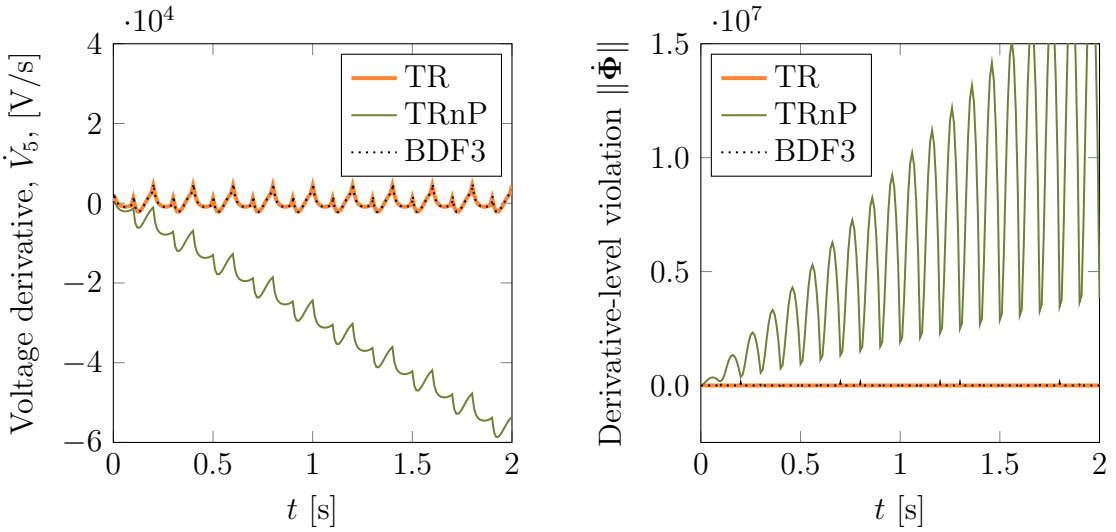
the elapsed times for each method and error criterion. Unlike the RLC examples,

Table 3.16: Elapsed times in the simulation of the full wave rectifier circuit.

Method	**Prec.**	Elapsed time Desktop [ms]	Elapsed time BBB [ms]	Elapsed time RPi4 [ms]
TR	Low	47.0	1,103.8	223.5
TRnP		42.7	984.0	220.3
BDF1		391.4	9,821.6	1,719.3
BDF2		37.8	924.1	223.5
BDF3		125.1	3,025.8	574.2
TR	High	206.3	4,817.1	913.8
TRnP		161.2	3,805.3	724.9
BDF1		14,009.5	141,564.0	61,429.0
BDF2		241.6	5,897.7	1,060.0
BDF3		125.1	3,025.8	574.2

the rectifier is a nonlinear problem with relatively fast changes in its dynamics. Each diode in this system can be in a reverse or forward state; iterative solvers may run into numerical issues during the transitions between them. Moreover, the tangent matrix of the system is not constant and has to be evaluated and factorized at every solver iteration. This explains the comparatively longer computation times and

higher number of iterations required for convergence in this example. The abruptly changing dynamics of the rectifier results in the need to use integration steps smaller than $h = 0.1$ ms with the BDF3 solver, even for low precision requirements. Increasing the step-size beyond this limit causes the errors with respect to the reference solution to rise quickly and, eventually, leads to the failure of the simulation. This fact points towards the difficulty that high-order multi-step integrators experience to keep the simulation of discontinuous systems stable. When low precision results are acceptable, TR and BDF2 are preferable in terms of efficiency, as confirmed by the results in Table 3.16. The full wave rectifier example can also be used to high-



(a) Full wave rectifier: Derivative with respect to time of the voltage at node 5, $\dot{V}_5$.

(b) Full wave rectifier: Norm of derivative-level constraint violations term, $\|\dot{\mathbf{\Phi}}\|$.

Figure 3.12: Full wave rectifier: Derivatives.

light the importance of keeping the derivative-level violation of algebraic constraints under control when using certain integrators. Fig. 3.12a shows the derivative with respect to time of the voltage at node 5, $\dot{V}_5$, obtained with three different integration approaches, namely BDF3 and the TR with and without the projection step in Eq. (3.20). As expected, BDF3 and the projected TR deliver the same results. The uncorrected TR, however, delivers a magnitude of $\dot{V}_5$ that consistently increases with time. It must be stressed that the three simulations used to obtain Fig. 3.12a satisfied the algebraic constraints $\mathbf{\Phi} = \mathbf{0}$ and the differential equations $\mathbf{A}\dot{\mathbf{x}} + \mathbf{b} = \mathbf{0}$, explicitly imposed by Eq. (3.12). Accordingly, they obtain very similar values of the system variables $\mathbf{x}$, within the acceptable values in Table 3.5. This is not the case with the derivatives with respect to time $\dot{\mathbf{\Phi}} = \mathbf{\Phi}_\mathbf{x}\dot{\mathbf{x}} + \mathbf{\Phi}_t = \mathbf{0}$. The TR integration formula is compatible with the accumulation of derivative-level errors, as long as the obtained values of $\dot{\mathbf{x}}$ satisfy the differential equations of the circuit. The BDF3 method suffers from the same limitation, but it does not reuse the derivatives to evaluate future values of $\dot{\mathbf{x}}$. As shown in Figs. 3.12a and 3.12b, the projection step removes the accumulation of derivative-level constraint violations and prevents $\dot{\mathbf{x}}$ from reaching excessive values that could eventually lead to numerical errors in the computations or even to the simulation failure by overflow.

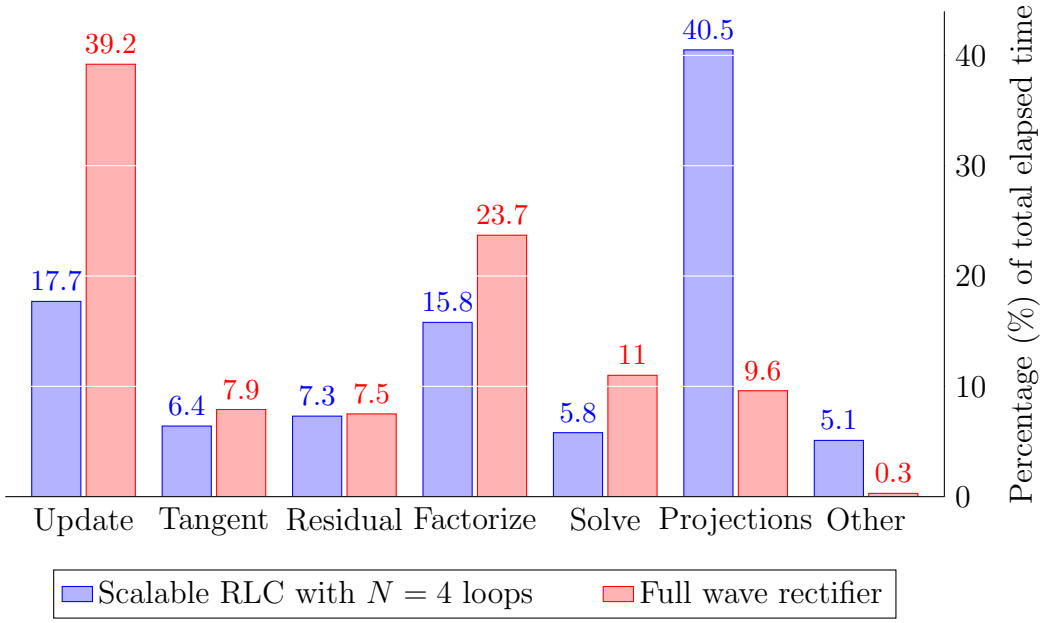


Figure 3.13: Percentage of total runtime elapsed by the TR method in each stage of the time-step.

Fig. 3.13 shows the distribution of computational workload between the different tasks during a time-step for the full wave rectifier, solved with low precision requirements. The trapezoidal rule was used as integrator. Results are compared to those delivered by the scalable RLC circuit with $N = 4$ loops, which is similar to the rectifier in terms of system size. The solver was forced to repeat the evaluation and factorization of the tangent matrix in each solver iteration in both examples. The integration step can be divided into the following tasks:

- the update of the dynamic terms $\mathbf{\Phi}$, $\mathbf{A}$, $\mathbf{b}$, and their derivatives;
- the evaluation of the residual $\mathbf{r}$ and the tangent matrix $\mathrm{d}\mathbf{r}/\mathrm{d}\mathbf{x}$ in Eq. (3.11);
- the factorization and backsubstitution of the linear solver in Eq. (3.11);
- the projections in Eq. (3.20); and
- other operations, such as the update of variables and regulation of code execution.

The profiling results confirmed that the computational effort of the projection step amounts to two or three iterations of the main system solver for the studied systems. For circuits that require a considerable number of iteration solvers, such as the rectifier, the overhead of the projection step is less relevant in relative terms than it would be in RLC systems, where convergence is achieved with one or two iterations.

3.6.1.4 Permanent-magnet synchronous motor thermal circuit

The thermal equivalent circuit that corresponds to the synchronous motor was also solved using the DAE integration methods. The step-sizes h and number of

iterations per step are shown in Tables 3.17 and 3.18 for low and high precision, respectively. Tables 3.17 and 3.18 show that the order of magnitude of the error in

Table 3.17: Synchronous motor: Low-precision configuration of the DAE integration methods.

Method	h [ms]	γ	$\hat{\Xi}_T$ [K]	$\hat{\Xi}_Q$ [W]
TR	500	2	$9.36 \cdot 10^{-7}$	0.612
BDF1	25	2	$1.49 \cdot 10^{-4}$	0.839
BDF2	250	2	$9.01 \cdot 10^{-7}$	0.523
BDF3	500	2	$3.12 \cdot 10^{-7}$	0.388

Table 3.18: Synchronous motor: High-precision configuration of the DAE integration methods.

Method	h [ms]	γ	$\hat{\Xi}_T$ [K]	$\hat{\Xi}_Q$ [W]
TR	100	2	$6.03 \cdot 10^{-8}$	0.024
BDF1	2.5	2	$1.49 \cdot 10^{-5}$	0.085
BDF2	100	2	$1.56 \cdot 10^{-7}$	0.092
BDF3	250	2	$6.54 \cdot 10^{-8}$	0.060

heat flows is much larger than that of the error in temperatures; the accuracy in the calculation of the heat flows becomes thus more critical to determine if a simulation is correct or not. Table 3.19 contains the elapsed times for every configuration and integration method. All methods are well below the RT limit in all cases, regardless of the selected computation environment.

Besides having slower dynamics and time-varying properties, thermal equivalent circuits differ from their conventional RC counterparts in that they are often subjected to excitations that are not periodic. The system behaviour during transients is a relevant part of the dynamics and needs to be determined accurately in most applications. The initialization method plays an important role in the evaluation of the first transient in the simulation; unsuitable initialization approaches result in incorrect predictions during the first integration steps that affect negatively the simulation accuracy long after the initialization phase is complete. Fig. 3.14 shows that the temperature T_9 delivered by the BDF3 integrator using the TR as initialization routine matches the reference solution, even with an integration step-size $h = 1$ s. On the other hand, if the initialization is not properly performed, for instance assuming a constant value of the temperature before $t = 0$ (BDF3*), the effect of initialization errors can be appreciated much later in the simulation, rendering it visibly inaccurate. This behaviour was not observed in the electric and electronic

Table 3.19: Elapsed times in the simulation of the thermal circuit of the synchronous motor.

Method	**Prec.**	Elapsed time Desktop [s]	Elapsed time BBB [s]	Elapsed time RPi4 [s]
TR	Low	0.12	2.81	0.59
TRnP		0.06	1.28	0.27
BDF1		1.12	26.29	5.79
BDF2		0.12	2.86	0.58
BDF3		0.06	1.60	0.32
TR	High	0.55	14.03	2.82
TRnP		0.27	6.34	1.40
BDF1		10.88	264.15	56.08
BDF2		0.29	7.17	1.49
BDF3		0.12	3.20	0.64

circuits in Sections 3.6.1.1–3.6.1.3, in which errors derived from poor initialization approaches were quickly corrected upon completion of the initial transient.

3.7 Conclusions

This Chapter puts forward a methodology to evaluate the performance of time-domain simulation methods for electric, electronic, and thermal equivalent circuits, oriented towards the determination of their suitability to be used in RT environments. Four benchmark problems with their corresponding reference solutions were defined, together with error metrics and accuracy criteria to enable the comparison of different methods. These examples are easy to reproduce and representative of significant problem types in circuit simulation.

The proposed methodology was used to assess the capability of ODE and DAE solver methods to deliver RT performance in three different hardware and software environments, including ARM boards with limited computing capabilities. The tested methods included transforming the circuit equations into a system of ODEs prior to their numerical integration, and NR iterative solvers for nonlinear systems of DAEs. BDF methods and the TR were used as integration formulas.

The numerical experiments performed in this study demonstrated that the benchmarking methodology and the examples can be used to obtain relevant information regarding the performance and features of a given solution method. Results made it possible to discard the direct ODE integration as a generally valid approach, at least in the form presented in the methods Section in this Chapter. Among the DAE solver methods, the first-order BDF integrator consistently showed low performance, requiring very small integration step-sizes, down to three orders of magnitude smaller than those required by the other formulas, to attain the accuracy requirements defined in the benchmark. This rendered it unsuitable except for the simulation of the simplest test cases.

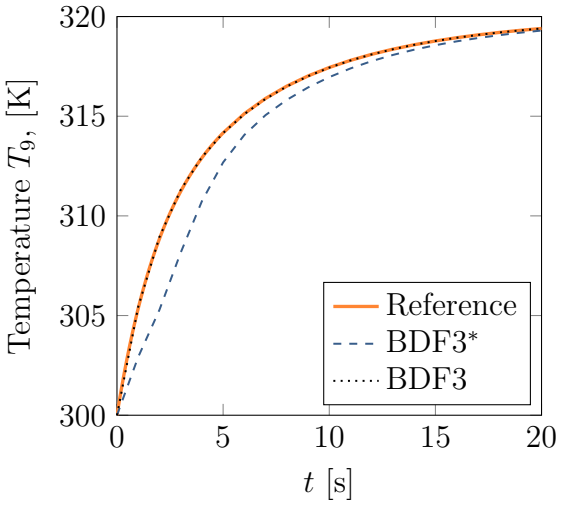


Figure 3.14: Synchronous motor: First 20 seconds of the time history of the temperature at node 9, T_9, obtained with $h = 1$ s.

Implementations based on second and third order BDF formulas and the TR delivered comparable performances. For systems with relatively slow dynamics, BDF3 always delivered the best efficiency. TR and BDF2, on the other hand, could be used to obtain faster computations in circuits with nonsmooth behaviour when the level of accuracy can be relaxed. A proper initialization of BDF methods is required to obtain accurate simulation results, especially in circuits in which the excitation is not periodic.

Regarding the RT ability of the tested methods, the benchmark framework can be used to provide orientations regarding the scope of applicability of a method in a certain computing platform. With the PC used in this study, BDF2, BDF3, and TR were able to solve all the problems in the benchmark below the RT mark. For RLC circuits, systems of up to a few thousand variables, depending on the requested precision, could be integrated below this limit. Results also show that ARM single-board computers can also be used to simulate most problems in the benchmark set. However, more factors than computational efficiency determine the usability of a method in RT computing, such as system latency and communication delays. The results from benchmarking can be used to rule out inefficient or inadequate approaches, and to choose the most effective options when dealing with the solution of a given problem.

Besides conclusions about efficiency, the simulation of the benchmark problems led to the identification of issues with the evaluation of the derivatives with respect to time of the system variables. Integration methods that determine the derivatives in future steps using previously computed values of the same derivatives need to enforce the satisfaction of the derivatives with respect to time of the algebraic equations of the circuit. Failure to do so may result in the accumulation of errors in the system derivatives, which can eventually lead to numerical issues in problem solutions. The TR is one of these methods; a projection step was introduced in this Chapter to address this need of the integrator. It was observed that the computational requirements of the projection step are comparable to those of two or three iterations

of the DAE solver.

Chapter 4

State, parameter, and input estimation for digital twins applied to linear thermal systems

This Chapter deals with the second area of interest in MBST environments identified in Chapter 2: the development of methodologies and software tools for state, parameter, and input estimation by means of sensor measurements, namely PISE. This technique can be of help in a wide variety of applications; for instance, it can be used as a component to build DTs of physical systems under demanding working conditions. In this Chapter, it is employed to generate DTs of e-powertrain components. These are key elements of electric vehicles and their performance is heavily affected by their thermal state. The correct operation of e-powertrains requires that some critical temperatures are monitored and kept below certain thresholds to avoid performance drops and even the failure of the complete system. However, not all magnitudes can directly be measured by means of sensors placed exactly at the most relevant locations. This difficulty can be addressed by the use of virtual sensing, i.e., employing computational system models that aim to represent the behaviour of their real-world counterpart, and making use of an estimator, e.g., a KF, to fuse the simulation results and the measurements from the sensors mounted on the systems. The estimation thus serves to prevent the simulation from drifting away from the actual component responses.

This Chapter details a method to generate DTs of linear thermal systems, starting from the DAE modelling introduced in Chapter 3, and transforming it into an ODE formulation more suitable for its use in estimation algorithms. The resulting estimation method can be used to perform the initial parameter adjustment of the model, as well as parameter, input, and state corrections during runtime. The proposed method is benchmarked using a simple RC circuit and a compact LPTM of a three-phase inverter used in EVs. Several OSs and hardware platforms were used to demonstrate the RT capabilities of the method, necessary when DTs are used in MBST applications, in which RT performance has to be guaranteed.

4.1 Introduction

Nowadays, a considerable number of industrial applications of interest are composed of large sets of diverse components. As a consequence, keeping track of their internal state for monitoring purposes becomes a challenging task. As the number of relevant magnitudes to control grows, the sensing requirements increase; in many cases, using dedicated sensors for each magnitude is an expensive, inefficient, or even unfeasible solution. The use of multiple sensors could also hinder the performance of RT applications, due to the high-rate flow of data that needs to be processed. An alternative approach is the use of PISE techniques, in which some magnitudes are not directly measured in the physical system, but estimated by means of a simulation. This way, PISE becomes a procedure to fuse the readings of a limited set of sensors located at a given physical system with data from the simulation of that system model, in order to extend the scope of the monitored magnitudes.

It is also possible to consider PISE the other way around, as a means to use sensor readings to make computer simulations represent more faithfully the behaviour of the physical system that they describe. In most cases, the virtual models used in simulation suffer from some modelling issues, derived from the need to simplify some aspects from reality, or from errors in the characterization of the model parameters or input values. PISE can play an important role to alleviate these uncertainties and limitations of the computational models and enhance the accuracy of the results obtained in simulation. Both aspects are significant contributions when building DTs of physical components or systems.

The DT concept was initially introduced in 2003 at the University of Michigan. This term is used to denote a computational representation of a real-world component by using a detailed model, which should be –in theory– indistinguishable from its real counterpart [5]. A bi-directional flow of information between virtual models and real world components, from which both parts can benefit, is also a fundamental component of the DT concept. Ideally, DTs can be employed to simulate the behaviour of their real-world counterparts and perform RT optimization [2] during product development, testing, and operation within the MBST paradigm. Virtual models used in DTs can be generated following diverse strategies, e.g., prioritising accuracy in high-fidelity models or compactness in reduced ones. For example, reduced virtual models could be used in DTs, instead of a high-detailed ones, when high efficiency is a design requirement. These reduced models could guarantee RT performance and their modelling errors may be corrected by means of the data from the physical sensors.

Cyber-physical test benches, in which real-world components under test are interfaced to a computer simulation of the overall system and its environment, represent one of the most interesting applications of DTs in the MBST realm. Consider, for instance, an automotive test bench for e-powertrains, in which electric motors or inverters are connected to a MBS dynamics simulation that follows the commands of a human driver. In principle, it should be possible to determine the temperatures of the e-powertrain components in this setup from the simulation of their thermal behaviour. In practice, the simulation results may differ significantly from the actual values, because of modelling and solution errors. This information, on

the other hand, could be obtained from temperature sensors on the physical system; however, these often cannot be placed directly on the locations of interest. The use of the DT approach, via PISE techniques, makes it possible to address both concerns. First, it can be employed to gain insight into the information provided by the sensors mounted on the system and monitor its behaviour beyond directly available measurements. Moreover, the results obtained this way can be used to enhance the system modelling and simulation, which in turn results into improved user safety, preventing accidents, malfunctions or early breakdowns.

The above-mentioned example illustrates the necessity of PISE methods in MBST applications. The whole computational representation of the employed DTs needs to be correctly calculated. Modelling and numerical errors lead to non-realistic simulation results, due to the uncertainties inherent in virtual models. Therefore, the use of virtual sensors is almost mandatory to guarantee the precision of the results. The information from real sensors can also be used to correct inputs and parameters at the validation stage and during operation. Different algorithms exist to fuse the information coming from real sensor measurements and virtual model simulations. Some varieties of the KF [80] are frequently used as state estimator in the context of PISE. For instance, methods based on KF have been used to develop state and force observers for multibody systems [12], navigation techniques for terrestrial and extraterrestrial applications [66, 67], and parameter estimation solutions for nonlinear dynamics [68], among many other applications.

This Chapter puts forward a PISE-based DT solution for the study of the thermal behaviour of e-powertrain components. As pointed out earlier, thermal conditions have a relevant impact on the overall performance of e-powertrains. For instance, motor magnets can suffer a permanent demagnetization if their temperature exceeds a certain threshold; silicon components may also break down in high-temperature working conditions. Several publications have proposed the use of PISEs methods to monitor thermal effects in inverters and electric motors, e.g., [75, 76]. Indirect temperature readings have been used to determine the junction temperature of Metal Oxide Semiconductor Field Effect Transistors (MOSFETs) or IGBTs [82]; recent work along these lines [83] makes it possible to account for input disturbances as well. Similar strategies can be followed to conduct the modelling and condition monitoring of other e-powertrain components, such as PMSMs, e.g., [84, 85].

The method described in this Chapter starts from a detailed representation of the thermal phenomena in the component by means of a LPTM, and generates a compact representation of the system equations, adequate for its use in RT estimation algorithms. This formulation enables the use of KFs to estimate parameters with uncertainties using temperature measurements from the device during a preliminary tuning stage. Once the LPTM has been initially adjusted, the KF formulation can be used during operation runtime to correct errors in the simulation results that stem from alterations in the original system inputs and parameters, e.g., due to component degradation, and numerical errors in the integration process. The method is not limited to the study of a particular component and was tested in the simulation of a simple LPTM benchmark example and the thermal model of an inverter for automotive applications.

4.2 Modelling and estimation methods

The modelling and estimation methods required for the consideration of thermal effects in DTs of e-powertrain components need to deliver accurate results in an efficient way, guaranteeing the RT performance of code execution in most practical applications. A possible use of the concept is illustrated in Fig. 4.1, which shows the diagram of a SitL test bench for automotive inverters, an example of CPS in which a physical e-powertrain inverter (a) is tested in an environment that includes virtual elements, like the computer simulation of the vehicle dynamics (b). Some components in this setup, such as the inverter controllers (c) or the vehicle driver (d), may be physical or virtual, depending on the testing needs. The subsystems in this assembly exchange information via a RT co-simulation interface (e), responsible for orchestrating the simulation of the virtual components and coordinating the input and output exchanges between them and the physical systems.

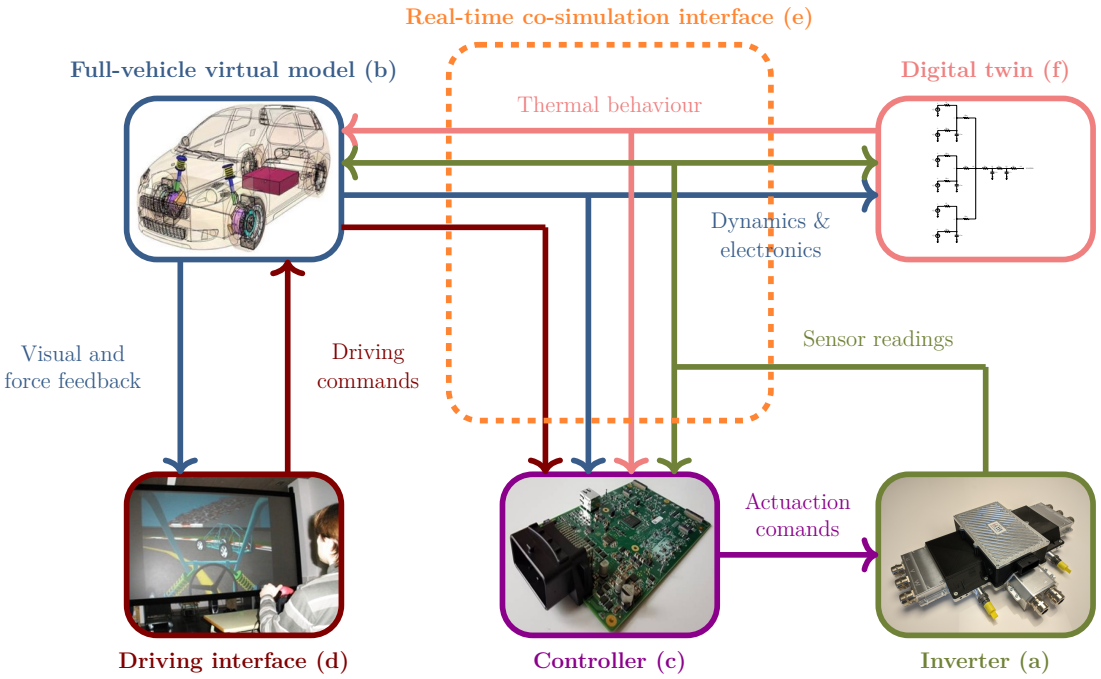


Figure 4.1: Example application: components of a SitL testbench for e-powertrain inverters.

Relevant magnitudes, such as the junction temperature of the inverter electronic components, often cannot be directly measured on the physical component [76], but can be estimated employing a DT of the inverter (f). The DT receives information from the temperature sensors mounted at some other locations on the inverter and from the results delivered by the numerical simulation of the vehicle. It also features a simplified thermal model of the inverter, generated from a LPTM obtained from a general-purpose circuit simulation software, e.g., the framework developed in Chapter 3 to simulate electric, electronic, and thermal circuits. Data from these two sources of information are fused by a KF to improve the representation of the physical inverter behaviour delivered by the DT.

The starting point of data fusion in the DT is a computational model of the real system. This model has to be efficient and it has to be possible to adapt it to the structure of the KF equations. When the original system dynamics is formulated using dependent variables as a DAE, frequently it has to be transformed into a more suitable form for effective state and parameter estimation.

4.2.1 Kalman filter equations

The discrete KF requires that the dynamics of the system to be observed is provided in the following discrete-time form

$$\mathbf{q}_{k+1} = \mathbf{F}_k \mathbf{q}_k + \mathbf{G}_k \mathbf{u}_k + \boldsymbol{\omega}_k \tag{4.1}$$

Eq. (4.1) expresses the state $\mathbf{q}$ at time-step $k+1$ as a function of the state and the input $\mathbf{u}$ at the previous instant, k. Terms $\mathbf{F}$ and $\mathbf{G}$ stand for the discrete-time system and input matrices and can be considered invariant between instants k and $k+1$; $\boldsymbol{\omega}$ represents the system noise, which is usually assumed to be White Gaussian Noise (WGN). The s sensors mounted on the plant deliver an $s \times 1$ array of measurements $\mathbf{o}$, which, in the absence of errors, could be expressed as a linear combination $\mathbf{h}_{s\times 1}$ of the system state and input

$$\mathbf{h}_k = \mathbf{H}_k \mathbf{q}_k + \mathbf{N}_k \mathbf{u}_k + \boldsymbol{\nu}_k \tag{4.2}$$

where $\mathbf{H}$ and $\mathbf{N}$ are linear combination matrices and $\boldsymbol{\nu}$ represents the sensor noise. The noise covariance matrices of the system and the measurements are $\mathbf{Q}$ and $\mathbf{R}$, respectively.

The discrete-time KF follows a predictor-corrector scheme to estimate the state at the next time instant, $k+1$, from the estimated state $\hat{\mathbf{q}}_k^+$ and the inputs $\mathbf{u}_k$ at current step k [80]. The predictor yields a first approximation of the state

$$\hat{\mathbf{q}}_{k+1}^- = \mathbf{F}_k \hat{\mathbf{q}}_k^+ + \mathbf{G}_k \mathbf{u}_k \tag{4.3}$$

The estimated error covariance matrix $\mathbf{P}$ is propagated as

$$\mathbf{P}_{k+1}^- = \mathbf{F}_k \mathbf{P}_k^+ \mathbf{F}_k^{\mathrm{T}} + \mathbf{Q} \tag{4.4}$$

where superscript $()^-$ stands for the *a priori* estimated values. Once these first values have been calculated, the correction step improves the predictions at time $k+1$

$$\hat{\mathbf{q}}_{k+1}^+ = \hat{\mathbf{q}}_{k+1}^- + \mathbf{K}_{k+1} \left(\mathbf{o}_{k+1} - \mathbf{h}_{k+1} \right) \tag{4.5}$$

$$\mathbf{P}_{k+1}^+ = \left(\mathbf{I} - \mathbf{K}_{k+1} \mathbf{H}_{k+1} \right) \mathbf{P}_{k+1}^- \tag{4.6}$$

where subscript $()^+$ now denotes *a posteriori* estimated values, $\mathbf{I}$ is the identity matrix and term $\mathbf{K}$ stands for the Kalman gain, evaluated as

$$\mathbf{K}_{k+1} = \mathbf{P}_{k+1}^- \mathbf{H}_{k+1}^{\mathrm{T}} \left(\mathbf{H}_{k+1} \mathbf{P}_{k+1}^- \mathbf{H}_{k+1}^{\mathrm{T}} + \mathbf{R} \right)^{-1} \tag{4.7}$$

The KF is at the core of the DT ability to accurately represent thermal effects in its physical counterpart, and it plays a twofold role. In the first place, it makes

it possible to adjust uncertain LPTM parameters to match hardware behaviour. Second, it enables the estimation of temperature values that cannot be directly measured during operation. Fig. 4.2 illustrates the operation of the KF inside the DT for the latter role: data from the temperature sensors (T_{sensors}) are fused with the results of the numerical simulation of the LPTM (T_{model}) to deliver the estimated system temperatures ($T_{\text{estimated}}$). The dynamics equations of the LPTM used as

Figure 4.2: Conceptual scheme of the KF usage.

system model by the KF, however, are generally not in the form of Eq. (4.1) and must be transformed for their use in the estimation algorithm. Additionally, RT performance is a requirement in DT applications, so the resulting thermal dynamics formulation has to be both compact and efficient. The following Sections discuss how to arrive at such a formulation starting from general purpose circuit simulation equations.

4.2.2 Lumped-parameter thermal model general-purpose equations

In this work, the thermal model required by the KF in Section 4.2.1 is generated from a LPTM of the component under study. Lumped-parameter equivalent circuits are a compact way to describe the thermal behaviour of e-powertrain components. They represent the heat transfer, thermal losses, and thermal inertia properties of a physical system by means of lumped components comparable to those of electric circuits. Each node in a LPTM corresponds to a representative point in the physical system and has a temperature T associated with it, which is analogous to voltage in an electric circuit; heat flows Q between nodes play a role similar to currents. Thermal conductivity and convection are represented with thermal resistors, thermal inertia is modelled with capacitors, and heat generation, e.g., Joule effect losses, is introduced in the model by means of current sources [150, 151].

The starting point for the computational methods in this paper is the modelling framework for electronic and thermal circuits introduced in Chapter 3, which represents a systematic way to assemble the dynamics circuit equations starting from its topology and component properties, although alternative descriptions could be used as well. Following this approach in the case of a thermal equivalent circuit, the variables used to describe the system are the temperature T of each node and the heat flow Q through every thermal resistor, source, and capacitor. They can be grouped in an $n \times 1$ array of system variables

$$\mathbf{x} = \begin{bmatrix} \mathbf{x}_T^{\mathrm{T}} & \mathbf{x}_Q^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}} \tag{4.8}$$

where terms $\mathbf{x}_T$ and $\mathbf{x}_Q$ contain n_T node temperatures and n_Q heat flows, respectively. The n variables in $\mathbf{x}$ are not independent, but are subjected to m algebraic

constraints

$$\mathbf{\Phi}\left(\mathbf{x}, \mathbf{v}, t\right) = \mathbf{0} \tag{4.9}$$

imposed by the satisfaction of KCL at the nodes, temperature specifications at certain nodes, and the constitutive equations of thermal resistors, namely $\Delta T = QR$ where ΔT is the temperature difference between the nodes connected by the resistor, Q is the heat flow through it, and R is the thermal resistance. Note that, in general, Eq. (4.9) is also a function of the system input, $\mathbf{v}$. Thermal capacitors, in turn, introduce a set of p linear ODEs in the form

$$\mathbf{\Gamma} = \mathbf{A}\dot{\mathbf{x}} + \mathbf{b} = \mathbf{0} \tag{4.10}$$

where $\mathbf{A}$ and $\mathbf{b}$ are $p \times n$ and $p \times 1$ terms. In general, circuit solvability requires that $n = m + p$.

Together, Eqs. (4.9) and (4.10) form a system of DAEs that is suitable to describe the dynamics of a wide array of electric, electronic, and thermal circuits in a straightforward way. They were used in [74] as the foundation of a general-purpose forward-dynamics circuit simulator. Section 4.3 below introduces a benchmark problem to illustrate the application of this modelling approach to a simple thermal network.

Expressing the system dynamics as a DAE system, however, is not convenient for its use in DTs. Redundant variables have a negative impact on computational efficiency and would add complexity to the estimation algorithm used to fuse system dynamics and sensor information, described in Section 4.2.1. A more advantageous formulation is obtained expressing the dynamics in terms of a reduced set of variables $\mathbf{z}$, via the elimination of the algebraic constraints in Eq. (4.9), as detailed in the following Section 4.2.3.

Thermal equivalent circuits have a particular structure that can be exploited to perform this coordinate reduction and develop efficient estimation algorithms. In the first place, unlike in most electronic circuits, the algebraic constraints $\mathbf{\Phi} = \mathbf{0}$ in Eq. (4.9) can be expressed as a linear combination of the system variables $\mathbf{x}$ and a set of r input values $\mathbf{v}$ that evolve as the simulation progresses

$$\mathbf{\Phi} = \mathbf{\Phi}_\mathbf{x}\mathbf{x} + \mathbf{\Phi}_\mathbf{v}\mathbf{v} = \mathbf{0} \tag{4.11}$$

where $\mathbf{\Phi}_\mathbf{x}$ and $\mathbf{\Phi}_\mathbf{v}$ are $m \times n$ and $m \times r$ matrices, respectively. It will also be assumed that term $\mathbf{b}$ in Eq. (4.10) can be expressed as

$$\mathbf{b} = \mathbf{A}_1\mathbf{x} \tag{4.12}$$

where $\mathbf{A}_1$ is a $p \times n$ matrix. The differential equations of a regular thermal capacitor are compatible with the use of Eq. (4.12). Terms $\mathbf{\Phi}_\mathbf{x}$, $\mathbf{\Phi}_\mathbf{v}$, $\mathbf{A}$, and $\mathbf{A}_1$ can be considered to remain constant during the numerical integration of the system dynamics from time instant k to the next one, $k+1$, although, in fact, some elements in these matrices, such as thermal resistance values, may vary as node temperatures evolve.

4.2.3 Reduction to minimal variables

The coordinate reduction put forward in this Section is based on the elimination of the algebraic constraints in Eq. (4.9). It is conceptually similar to the velocity transformation methods commonly used in multibody system dynamics [146],

although in this case the transformation can be directly performed on the system variables given the linearity of Eq. (4.11). The circuit dynamics thus becomes defined by the p differential equations in (4.10), a reduced set of p variables, $\mathbf{z}$, and the r inputs in $\mathbf{v}$. The reduced variable set $\mathbf{z}$ could be selected in several ways; in thermal circuits, the temperature of the nodes to which the capacitors are connected, is a convenient choice. Here, we will express $\mathbf{z}$ as a linear combination of the dependent variables $\mathbf{x}$

$$\mathbf{z} = \mathbf{B}_0\mathbf{x} \tag{4.13}$$

where $\mathbf{B}_0$ is a $p \times n$ matrix that will remain constant as long as the selection of independent variables and the circuit topology are not modified. Eqs. (4.13) and (4.11) can be grouped to form the system of equations

$$\begin{bmatrix} \mathbf{\Phi_x} \\ \mathbf{B}_0 \end{bmatrix} \mathbf{x} + \begin{bmatrix} \mathbf{\Phi_v} & \mathbf{0} \\ \mathbf{0} & -\mathbf{I}_p \end{bmatrix} \begin{bmatrix} \mathbf{v} \\ \mathbf{z} \end{bmatrix} = \mathbf{0} \tag{4.14}$$

where $\mathbf{I}_p$ is the $p \times p$ identity matrix. Eq. (4.14) can be solved for $\mathbf{x}$ if its leading matrix is regular. This reveals the requirement that $\mathbf{B}_0$ must complete the row rank of term $\mathbf{\Phi_x}$. The inverse of the leading matrix is partitioned as

$$\begin{bmatrix} \mathbf{\Phi_x} \\ \mathbf{B}_0 \end{bmatrix}^{-1} = \begin{bmatrix} \mathbf{S} & \mathbf{T} \end{bmatrix} \tag{4.15}$$

where $\mathbf{S}$ spans the first m columns of the $n \times n$ inverse matrix in Eq. (4.15), and $\mathbf{T}$ the last p ones. This provides the expression of the coordinate reduction

$$\mathbf{x} = \mathbf{T}\mathbf{z} - \mathbf{S}\mathbf{\Phi_v}\mathbf{v} = \mathbf{T}\mathbf{z} + \mathbf{B}\mathbf{v} \tag{4.16}$$

Matrices $\mathbf{T}$ and $\mathbf{B}$ can be interpreted as

$$\mathbf{T} = \frac{\partial \mathbf{x}}{\partial \mathbf{z}}\,; \qquad \mathbf{B} = \frac{\partial \mathbf{x}}{\partial \mathbf{v}} \tag{4.17}$$

Differentiation of Eq. (4.16) with respect to time provides the derivatives-level expression of the transformation

$$\dot{\mathbf{x}} = \mathbf{T}\dot{\mathbf{z}} + \dot{\mathbf{T}}\mathbf{z} + \mathbf{B}\dot{\mathbf{v}} + \dot{\mathbf{B}}\mathbf{v} \tag{4.18}$$

which can be substituted in Eq. (4.10) to obtain

$$\mathbf{\Gamma} = \mathbf{A}\left(\mathbf{T}\dot{\mathbf{z}} + \dot{\mathbf{T}}\mathbf{z} + \mathbf{B}\dot{\mathbf{v}} + \dot{\mathbf{B}}\mathbf{v}\right) + \mathbf{b} = \mathbf{0} \tag{4.19}$$

The system of ODEs in Eq. (4.19) expresses the dynamics of the LPTM in terms of the minimal set of variables $\mathbf{z}$ and the system input $\mathbf{v}$ and its derivatives.

4.2.4 Discrete state-space representation

The discrete-time state-space matrices $\mathbf{F}$ and $\mathbf{G}$ required by Eq. (4.1) can be obtained from Eq. (4.19). Substituting term $\mathbf{b}$ from Eq. (4.12) and the expression of $\mathbf{x}$ from Eq. (4.16) yields

$$\mathbf{\Gamma} = \mathbf{A}\mathbf{T}\dot{\mathbf{z}} + \mathbf{A}\dot{\mathbf{T}}\mathbf{z} + \mathbf{A}\mathbf{B}\dot{\mathbf{v}} + \mathbf{A}\dot{\mathbf{B}}\mathbf{v} + \mathbf{A}_1\left(\mathbf{T}\mathbf{z} + \mathbf{B}\mathbf{v}\right) = \mathbf{0} \tag{4.20}$$

Eq. (4.20) can be rearranged as

$$\dot{\mathbf{z}} = -\left(\mathbf{AT}\right)^{-1}\left(\mathbf{A}_1\mathbf{T} + \mathbf{A}\dot{\mathbf{T}}\right)\mathbf{z} - \left(\mathbf{AT}\right)^{-1}\left(\mathbf{A}_1\mathbf{B}\right)\mathbf{v} - \left(\mathbf{AT}\right)^{-1}\left(\mathbf{AB}\right)\dot{\mathbf{v}} \tag{4.21}$$

Grouping input $\mathbf{v}$ and its derivatives in a single term

$$\mathbf{u} = \begin{bmatrix} \mathbf{v}^{\mathrm{T}} & \dot{\mathbf{v}}^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}} \tag{4.22}$$

the $p \times p$ state and $p \times 2r$ input matrices $\mathbf{F}_{\mathrm{c}}$ and $\mathbf{G}_{\mathrm{c}}$ that correspond to the continuous state-space system equations are identified from Eq. (4.21)

$$\mathbf{F}_{\mathrm{c}} = -\left(\mathbf{AT}\right)^{-1}\left(\mathbf{A}_1\mathbf{T} + \mathbf{A}\dot{\mathbf{T}}\right) \tag{4.23}$$

$$\mathbf{G}_{\mathrm{c}} = \begin{bmatrix} \mathbf{G}_{\mathrm{c1}} & \mathbf{G}_{\mathrm{c2}} \end{bmatrix} \tag{4.24}$$

where

$$\mathbf{G}_{\mathrm{c1}} = -\left(\mathbf{AT}\right)^{-1}\left(\mathbf{A}_1\mathbf{B}\right) \tag{4.25}$$

$$\mathbf{G}_{\mathrm{c2}} = -\left(\mathbf{AT}\right)^{-1}\left(\mathbf{AB}\right) \tag{4.26}$$

The discrete-time counterparts $\mathbf{F}$ and $\mathbf{G}$ of the matrices in Eqs. (4.23) and (4.24) are evaluated as

$$\begin{bmatrix} \mathbf{F} & \mathbf{G} \\ \mathbf{0}_{2r\times p} & \mathbf{I}_{2r} \end{bmatrix} = e^{\begin{bmatrix} \mathbf{F}_{\mathrm{c}} & \mathbf{G}_{\mathrm{c}} \\ \mathbf{0}_{2r\times p} & \mathbf{0}_{2r\times 2r} \end{bmatrix}_h} \tag{4.27}$$

where h is the time interval between instants k and $k+1$. See Appendix E for the approximate expressions.

4.2.5 Application of the filter to state and parameter estimation

The state space matrices in Eq. (4.27) can be directly used in the KF expression in Eq. (4.1) if the filter is employed to carry out the estimation of the system state. In this case, $\mathbf{q} = \mathbf{z}$.

In some cases, however, it is also necessary to deal with uncertainties that affect the thermal parameters of the system, as the exact value of some thermal parameters could be initially unknown or vary during the operation of the component under study. In such a case, system parameters could be initially adjusted by means of global optimization approaches, especially when discussing moderate-size LPTMs [152]. The KF can also be applied to the estimation of uncertain parameters, besides the system state. However, the filter formulation in Section 4.2.4 cannot directly handle such situations, as they require dealing with a new system of equations, which is nonlinear. Instead, it is possible to use the EKF to this end. These uncertain parameters can be grouped in term $\boldsymbol{\rho}$, of size $s \times 1$, and added to the following extended state vector

$$\bar{\mathbf{q}} = \begin{bmatrix} \mathbf{z} \\ \dot{\mathbf{z}} \\ \boldsymbol{\rho} \end{bmatrix} \tag{4.28}$$

Although adding state derivatives $\dot{\mathbf{z}}$ in (4.28) is not mandatory, it eases the evaluation of the Jacobian matrices in the upcoming calculations.

The variables in the new state vector in Eq. (4.28) are no longer independent, as they must satisfy the condition $\boldsymbol{\Gamma} = \mathbf{0}$ imposed by Eq. (4.19). Therefore, the fulfillment of Eq. (4.19) must be imposed during the application of the EKF. In this work, the perfect measurements approach [153] is used to this end. The vector of measurements $\mathbf{o}$ and term $\mathbf{h}$ in Eq. (4.2) have to be modified accordingly. Equations $\boldsymbol{\Gamma} = \mathbf{0}$, now referred to as *perfect measurements*, are added to Eq. (4.2) to obtain

$$\bar{\mathbf{h}} = \begin{bmatrix} \mathbf{h}^{\mathrm{T}} & \boldsymbol{\Gamma}^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}} \tag{4.29}$$

while the vector of measurements is enlarged with p zeros

$$\bar{\mathbf{o}} = \begin{bmatrix} \mathbf{o}^{\mathrm{T}} & \mathbf{0}_{1\times p} \end{bmatrix}^{\mathrm{T}} \tag{4.30}$$

Term $\bar{\mathbf{H}}$ is then defined as

$$\bar{\mathbf{H}} = \frac{\partial \bar{\mathbf{h}}}{\partial \bar{\mathbf{q}}} = \begin{bmatrix} \frac{\partial \bar{\mathbf{h}}}{\partial \mathbf{z}} & \frac{\partial \bar{\mathbf{h}}}{\partial \dot{\mathbf{z}}} & \frac{\partial \bar{\mathbf{h}}}{\partial \boldsymbol{\rho}} \end{bmatrix} \tag{4.31}$$

The evaluation of term $\bar{\mathbf{H}}$ in Eq. (4.31) is made simpler if $\bar{\mathbf{h}}$ is expressed in terms of the independent variables $\mathbf{z}$ and their derivatives. Term $\mathbf{h}$ is often a subset of the dependent variables $\mathbf{x}$, because sensors mounted on thermal systems usually monitor the node temperatures or heat fluxes through components. For this reason, $\mathbf{h}$ can be expressed as

$$\mathbf{h}(\mathbf{x}) = \mathbf{C}\mathbf{x} \tag{4.32}$$

where $\mathbf{C}$ is a constant $s \times n$ matrix. The expression of the differential equations $\boldsymbol{\Gamma}$ in terms of the dependent variables is given in turn by Eqs. (4.10) and (4.12)

$$\boldsymbol{\Gamma}(\mathbf{x}, \dot{\mathbf{x}}) = \mathbf{A}\dot{\mathbf{x}} + \mathbf{A}_1\mathbf{x} \tag{4.33}$$

From Eqs. (4.29), (4.32) and (4.33)

$$\bar{\mathbf{h}}(\mathbf{x}, \dot{\mathbf{x}}) = \begin{bmatrix} \mathbf{C}\mathbf{x} \\ \mathbf{A}\dot{\mathbf{x}} + \mathbf{A}_1\mathbf{x} \end{bmatrix} \tag{4.34}$$

Eq. (4.34) is then rewritten in terms of the independent variables $\mathbf{z}$, the input $\mathbf{v}$, and their derivatives $\dot{\mathbf{z}}$ and $\dot{\mathbf{v}}$ using the transformations in Eqs. (4.16) and (4.18)

$$\bar{\mathbf{h}} = \begin{bmatrix} \mathbf{C}\mathbf{T}\mathbf{z} + \mathbf{C}\mathbf{B}\mathbf{v} \\ \mathbf{A}\left(\mathbf{T}\dot{\mathbf{z}} + \dot{\mathbf{T}}\mathbf{z} + \mathbf{B}\dot{\mathbf{v}} + \dot{\mathbf{B}}\mathbf{v}\right) + \mathbf{A}_1\left(\mathbf{T}\mathbf{z} + \mathbf{B}\mathbf{v}\right) \end{bmatrix} \tag{4.35}$$

The partial derivatives of terms in Eq. (4.35) are straightforward in most cases; for matrices $\mathbf{T}$ and $\mathbf{B}$ the inverse matrix derivative is used together with Eqs. (4.15) and (4.16), e.g.,

$$\frac{\partial \begin{bmatrix} \mathbf{S} & \mathbf{T} \end{bmatrix}}{\partial \boldsymbol{\rho}} = -\begin{bmatrix} \mathbf{S} & \mathbf{T} \end{bmatrix} \frac{\partial \begin{bmatrix} \boldsymbol{\Phi}_{\mathbf{x}} \\ \mathbf{B}_0 \end{bmatrix}}{\partial \boldsymbol{\rho}} \begin{bmatrix} \mathbf{S} & \mathbf{T} \end{bmatrix} \tag{4.36}$$

The continuous state-space form of the dynamics with the expanded state $\bar{\mathbf{q}}$ is

$$\dot{\bar{\mathbf{q}}} = \bar{\mathbf{F}}_{\mathrm{c}}\bar{\mathbf{q}} + \bar{\mathbf{G}}_{\mathrm{c}}\mathbf{u} \tag{4.37}$$

where $\bar{\mathbf{F}}_{\mathrm{c}}$ and $\bar{\mathbf{G}}_{\mathrm{c}}$ stand for the new state and input matrices, respectively. Assuming that the system properties can be considered constant during an integration step, these terms can be approximated as

$$\bar{\mathbf{F}}_{\mathrm{c}} \approx \begin{bmatrix} \mathbf{0}_{p\times p} & \mathbf{I}_p & \mathbf{0}_{p\times s} \\ \mathbf{0}_{p\times p} & \mathbf{F}_{\mathrm{c}} & \mathbf{0}_{p\times s} \\ \mathbf{0}_{s\times p} & \mathbf{0}_{s\times p} & \mathbf{0}_{s\times s} \end{bmatrix}; \qquad \bar{\mathbf{G}}_{\mathrm{c}} \approx \begin{bmatrix} \mathbf{0}_{p\times r} & \mathbf{0}_{p\times r} \\ \mathbf{0}_{p\times r} & \mathbf{G}_{\mathrm{c}1} \\ \mathbf{0}_{s\times r} & \mathbf{0}_{s\times r} \end{bmatrix} \tag{4.38}$$

where $\mathbf{I}_n$ is the $n \times n$ identity matrix. The use of matrices $\bar{\mathbf{F}}_{\mathrm{c}}$ and $\bar{\mathbf{G}}_{\mathrm{c}}$ does not result in an exact fulfilment of Eq. (4.37); accordingly, the correction step of the EKF is responsible for the satisfaction of Eq. (4.19). The discrete-form expression of matrices $\bar{\mathbf{F}}_{\mathrm{c}}$ and $\bar{\mathbf{G}}_{\mathrm{c}}$ can be obtained as

$$\begin{bmatrix} \bar{\mathbf{F}} & \bar{\mathbf{G}} \\ \mathbf{0}_{2r\times p} & \mathbf{I}_{2r} \end{bmatrix} = e^{\begin{bmatrix} \bar{\mathbf{F}}_{\mathrm{c}} & \bar{\mathbf{G}}_{\mathrm{c}} \\ \mathbf{0}_{2r\times p} & \mathbf{0}_{2r\times 2r} \end{bmatrix} h} \tag{4.39}$$

and the KF equations (4.1) and (4.2) are rewritten as

$$\bar{\mathbf{q}}_{k+1} = \bar{\mathbf{F}}_k\bar{\mathbf{q}}_k + \bar{\mathbf{G}}_k\mathbf{u}_k \tag{4.40}$$

$$\bar{\mathbf{h}}_k = \bar{\mathbf{H}}_k\bar{\mathbf{q}}_k + \bar{\mathbf{N}}_k\mathbf{u}_k \tag{4.41}$$

where $\bar{\mathbf{N}}$ is the feedthrough matrix $\mathbf{N}$ enlarged with p rows of zeros. The resulting KF equations then parallel the expressions in Section 4.2.1.

4.2.5.1 Parameter estimation during initial tuning of LPTM

The approach described in Section 4.2.5 makes it possible to estimate LPTM parameters, besides the system state. This can be done at two stages in the simulation cycle. First, the method can be used for a preliminary tuning process of the model parameters, in which modelling deviations between the LPTM and the actual system properties are corrected. After this initial tuning, the filter can be used during operation runtime to keep track of variations of the system properties, performing an online adjustment of the corresponding parameters. Both approaches can be used independently from each other; in fact, the EKF online estimation can be used even if the preliminary parameter adjustment was conducted using a different optimization method.

The preliminary tuning process can be performed in an iterative way. The initial set of parameters of the LPTM is $\boldsymbol{\rho}_0$. Then, a test run of the system under study is performed and sensor readings are fed to the KF defined by Eqs. (4.40) and (4.41). A new set of parameters, $\boldsymbol{\rho}_1$, is obtained as a result and used to update the LPTM properties. This procedure may have to be repeated in cases in which the convergence of the parameter values is slow. In this case, the operation continues

until the difference δ between two consecutive iterations $\tilde{\alpha}$ and $\tilde{\alpha}+1$ converges below an user-defined threshold error ε, i.e.,

$$\delta = \|\boldsymbol{\rho}_{\tilde{\alpha}} - \boldsymbol{\rho}_{\tilde{\alpha}+1}\| < \varepsilon \tag{4.42}$$

In offline parameter estimation, it would be advisable to repeat the procedure starting from different initial parameter sets, $\boldsymbol{\rho}_0$, to ensure that the algorithm converges to the same solution. It should be pointed out that the estimation methodology in this Section requires that the number, type, and position of the sensors are compatible with the estimation goals; the observability test [61] must be satisfied. The system excitation used during parameter identification must be sufficient to identify relevant values, especially thermal capacitances, which require a transient phase as described in Sections 4.3.2.2 and 4.4.4.2.

4.3 Benchmark problem

The methods described in Section 4.2 were tested in the simulation of the thermal dynamics of two examples. The first one is a simple thermal RC circuit that can be used as benchmark problem, shown in Fig. 4.3. It should be noted that this benchmark does not represent any physical system, and is intended to serve as a test case to illustrate the application of the method in Section 4.2. The results obtained from the simulation of the benchmark are used as reference solution; these will be used to determine the ability of the proposed method to correct modelling errors in the system parameters and disturbances in its input. The benchmark problem also illustrates the use of the method and provides a means to replicate its results in a straightforward fashion. The circuit consists of one heat source Q_0 connected to three resistors and two thermal capacitors. The system contains four nodes, of which number 4 is assumed to represent air at a constant temperature $T_{\text{AIR}} = 300$ K. The heat generated at the source has a constant value $Q_0 = 10$ W,

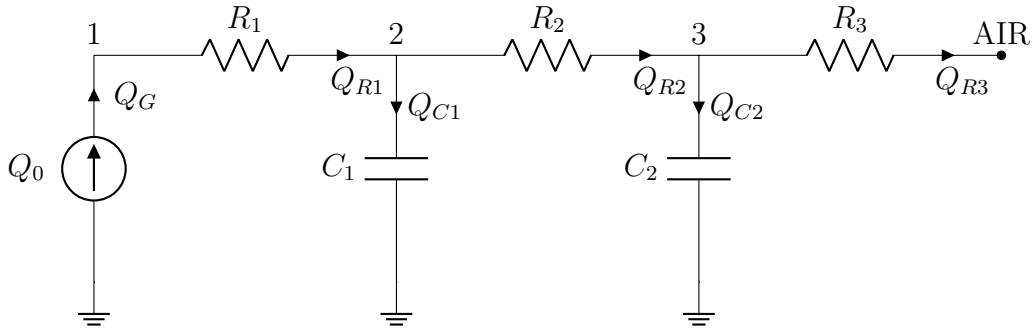


Figure 4.3: Benchmark RC thermal circuit.

resistors R_1, R_2, and R_3 have values of 1, 2, and 3 K/W respectively. The parameters of the capacitors are $C_1 = 0.1$ J/K and $C_2 = 0.2$ J/K and their initial temperatures are set to $T_2^0 = 299$ K and $T_3^0 = 301$ K. Temperature sensors can be placed at nodes 1, 2, and 3. Four estimation scenarios are considered:

- *Resistor estimation*: Parameters R_1, R_2, and R_3 are estimated using sensors in nodes 1, 2, and 3. Initial values $R_1 = R_2 = R_3 = 10$ K/W are assumed for the resistances.
- *Capacitor estimation*: Parameters C_1 and C_2 are estimated using sensors in nodes 2 and 3. Initial values for the capacitance of these elements are $C_1 = 1$ J/K and $C_2 = 10$ J/K.
- *Source parameter estimation*: Q_0 is selected as uncertain parameter to be determined. An initial value $Q_0 = 1$ W is assumed and corrected using a single temperature sensor placed on node 3.
- *State estimation with input disturbance*: The full system state is estimated using temperature measurements from nodes 2 and 3. A modelling error is introduced in the heat source: the thermal model used in the estimation features a constant heat source with $Q_0 = 10$ W, while the actual heat generation follows a sinusoidal function, $Q_0 = 10(1 + \sin(10\pi t))$ W.

For the purposes of this benchmark problem, it is assumed in each scenario that only the target parameters or input are uncertain, while every other parameter in the model is accurately known. Often, however, uncertainties can be found in several parameters simultaneously. For instance, thermal resistance and capacitance parameters may need to be characterized for the same circuit. In such a case, resistors can be adjusted from the steady-state results of the component at hand, even if the capacitance values of the system are not known yet. Once the resistors have been corrected, the system capacitances can be estimated from the transient response of the physical system. This procedure is illustrated with the problem in Section 4.4.

In all cases, a simulation of the thermal dynamics of the circuit with exact parameters was used as reference solution and as source of sensor measurements. Reference temperatures at nodes 1, 2, and 3 are shown in Figs. 4.4 and 4.5 for the two types of heat sources, namely constant and sinusoidal. The estimation scenarios are summarized in Table 4.1.

Table 4.1: Scenarios for testing the proposed KF.

Scenario	**Parameters** $\boldsymbol{\rho}$	**Sensors in nodes**	$Q_0(t)$ [W]
Resistors	R_1, R_2, R_3	$1, 2, 3$	10
Capacitors	C_1, C_2	$2, 3$	10
Source	Q_0	3	10
State	Q_0	$2, 3$	$10(1 + \sin(10\pi t))$

4.3.1 System modelling

The benchmark RC thermal circuit can be modelled with a set of $n = 10$ dependent generalized variables $\mathbf{x}$. Of these, four correspond to the temperatures of

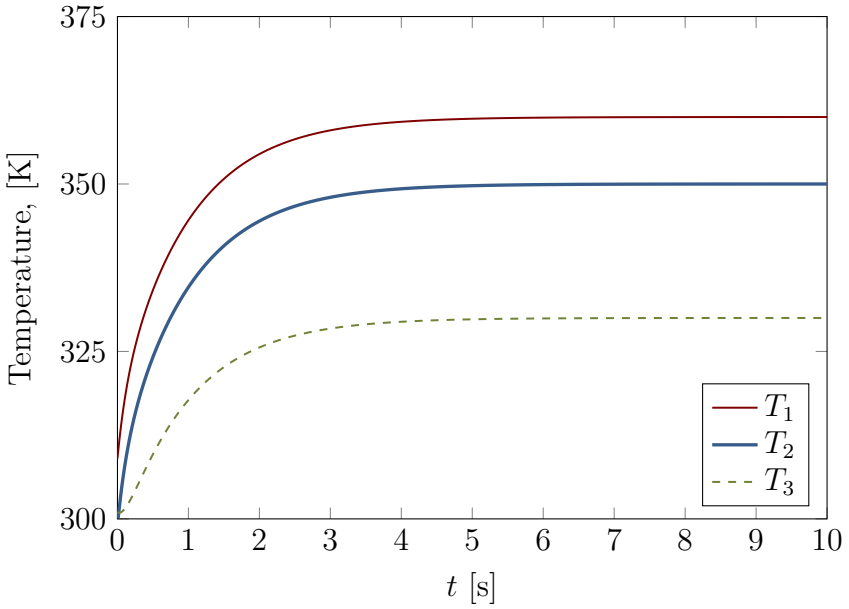


Figure 4.4: Time-history of the node temperatures in the benchmark problem for $Q_0 = 10$ W.

the nodes and the other six to the heat flows through the components, shown in Fig. 4.3. The corresponding variables in Eq. (4.8) for this example are

$$\mathbf{x}_T = \begin{bmatrix} T_1 & T_2 & T_3 & T_4 \end{bmatrix}^{\mathrm{T}} \tag{4.43}$$

$$\mathbf{x}_Q = \begin{bmatrix} Q_G & Q_{R1} & Q_{R2} & Q_{R3} & Q_{C1} & Q_{C2} \end{bmatrix}^{\mathrm{T}} \tag{4.44}$$

The input for this system consists of the heat introduced by the source and the air temperature at node 4

$$\mathbf{v} = \begin{bmatrix} Q_0 & T_{\mathrm{AIR}} \end{bmatrix}^{\mathrm{T}} \tag{4.45}$$

so $r = 2$. The variables in $\mathbf{x}$ and the input $\mathbf{v}$ must fulfil $m = 8$ algebraic equations $\mathbf{\Phi} = \mathbf{0}$ imposed by Kirchhoff's equations at each node, the fixed temperature of node 4, and the constitutive equations of heat sources and thermal resistors, which relate the heat flow through each component to its physical properties. The expression of Eq. (4.9) for this RC circuit is

$$\mathbf{\Phi} = \begin{bmatrix} Q_G - Q_{R1} \\ Q_{R1} - Q_{R2} - Q_{C1} \\ Q_{R2} - Q_{R3} - Q_{C2} \\ T_4 - T_{AIR} \\ Q_0 - Q_G \\ T_1 - T_2 - Q_{R1} R_1 \\ T_2 - T_3 - Q_{R2} R_2 \\ T_3 - T_4 - Q_{R3} R_3 \end{bmatrix} = \mathbf{0} \tag{4.46}$$

Finally, each thermal capacitor introduces a differential equation in the form

$$C \dot{T}_{\mathrm{a}} = Q_C \tag{4.47}$$

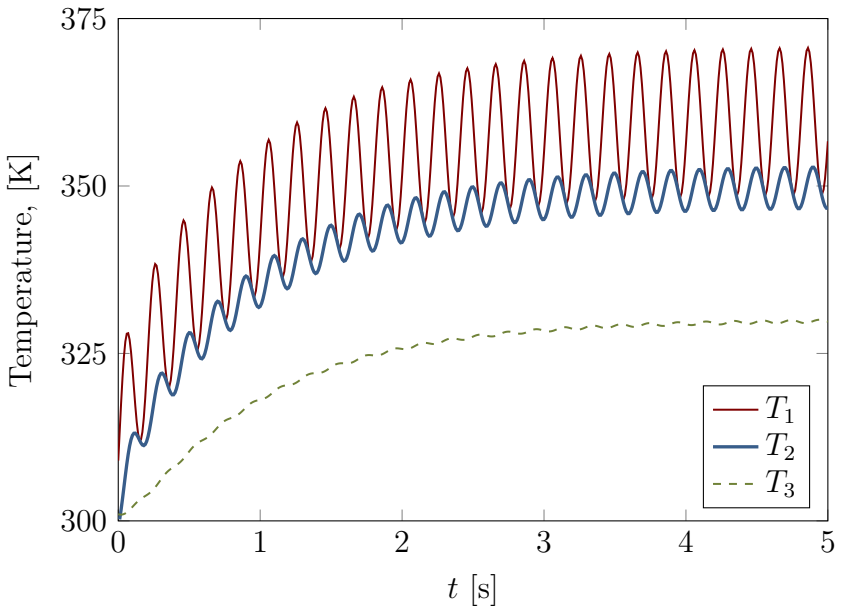


Figure 4.5: Time-history of node temperatures in the benchmark problem for $Q_0 = 10(1 + \sin(10\pi t))$ W.

where C is the capacitance, T_a is the temperature of the node to which the capacitor is connected, and Q_C is the heat flowing into the component. The two capacitors in this example introduce $p = 2$ ODEs in the form of Eq. (4.10), where terms $\mathbf{A}$ and $\mathbf{b}$ take the form

$$\mathbf{A} = \begin{bmatrix} 0 & -C_1 & 0 & \\ 0 & 0 & -C_2 & \mathbf{0}_{2\times 7} \end{bmatrix} \tag{4.48}$$

$$\mathbf{b} = \begin{bmatrix} Q_{C1} \\ Q_{C2} \end{bmatrix} \tag{4.49}$$

The resulting system of DAE given by Eqs. (4.46), (4.48), and (4.49) can be transformed into a system of DAE through the selection of an appropriate set of minimal variables via Eq. (4.13). The number of independent variables matches the name of ODE in the problem, so the temperature drop at the thermal capacitors is a reasonable choice

$$\mathbf{z} = \begin{bmatrix} T_2 \\ T_3 \end{bmatrix} \tag{4.50}$$

Minimal variables $\mathbf{z}$ can be related to the generalized set $\mathbf{x}$ through the constant transformation matrix

$$\mathbf{B}_0 = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{4.51}$$

4.3.1.1 Estimation

Parameter and state estimation for this example are conducted using the methods in Section 4.2. The time-step h is set to 1 ms and sensor measurements are introduced into the correction step of the filter every time a step is taken.

Sensor measurements are assumed to present WGN with mean 0 K and standard deviation 0.5 K. Term $\mathbf{R}$ is the covariance matrix of the measurement noise, a diagonal matrix whose elements are set to 0.25 K^2 for the physical sensors and to 0 K^2 for perfect measurements. Term $\mathbf{P}$ is a diagonal matrix as well, with elements initially set to 10 in SI units for the parameters and 0.01 in SI units for the states. Matrix $\mathbf{Q}$ was adjusted to achieve a constant Power Spectral Density (PSD) of the innovation.

4.3.2 Results

The results obtained in the solution of the four problems described in Section 4.3, namely resistance, capacitance, heat source, and temperature estimation in the benchmark circuit are shown next.

4.3.2.1 Resistor estimation

System parameters, including resistance and capacitance values, must be adjusted before using the model during operation. The first estimation scenario considered involves correcting resistances R_1, R_2, and R_3 in the benchmark circuit. Initial values $R_1 = R_2 = R_3 = 10$ K/W were used as first approximation to the actual system parameters

$$\boldsymbol{\rho} = \begin{bmatrix} R_1 & R_2 & R_3 \end{bmatrix}^{\mathrm{T}} \tag{4.52}$$

The sensor readings used in this case were the temperatures of nodes 1, 2, and 3.

$$\mathbf{o} = \begin{bmatrix} T_1^{\text{sensor}} & T_2^{\text{sensor}} & T_3^{\text{sensor}} \end{bmatrix}^{\mathrm{T}} \tag{4.53}$$

These can be expressed in terms of the system independent variables and inputs, and the corresponding virtual sensors take the expression

$$\mathbf{h} = \begin{bmatrix} T_2 + Q_0 R_1 \\ T_2 \\ T_3 \end{bmatrix} \tag{4.54}$$

Fig. 4.6 shows the time-history of the resistance parameters as the estimation progresses. A single simulation run was sufficient to achieve convergence to the reference values, namely 1, 2, and 3 K/W respectively.

4.3.2.2 Capacitor estimation

The second scenario consists in the adjustment of the capacitance parameters C_1 and C_2,

$$\boldsymbol{\rho} = \begin{bmatrix} C_1 & C_2 \end{bmatrix}^{\mathrm{T}} \tag{4.55}$$

Initially, $C_1 = 1$ J/K and $C_2 = 10$ J/K. The vectors of sensor measurements $\mathbf{o}$ and virtual sensors $\mathbf{h}$ in this case are

$$\mathbf{o} = \begin{bmatrix} T_2^{\text{sensor}} & T_3^{\text{sensor}} \end{bmatrix}^{\mathrm{T}} \tag{4.56}$$

$$\mathbf{h} = \begin{bmatrix} T_2 \\ T_3 \end{bmatrix} \tag{4.57}$$

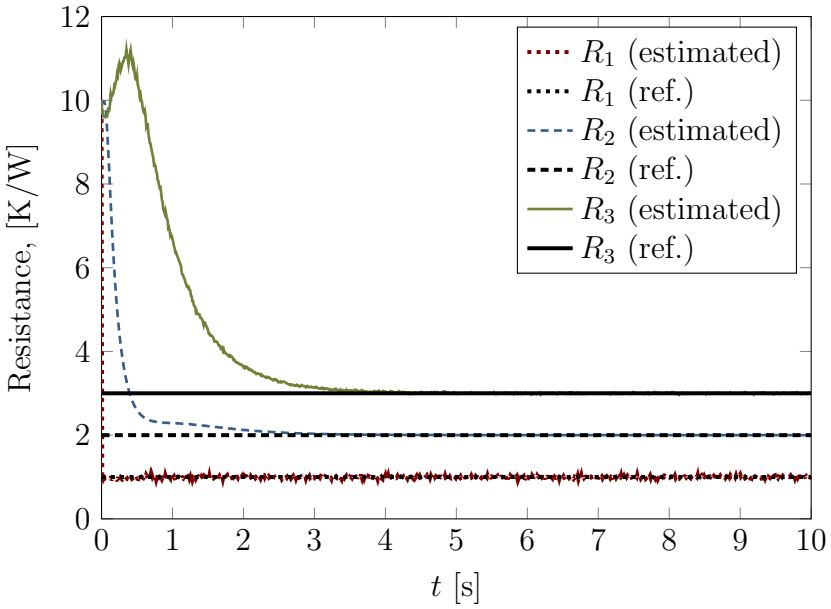


Figure 4.6: RC circuit: Evolution of resistance values during estimation.

Noticeable differences exist between resistance and capacitance estimation. The convergence of the capacitor parameters is slower; moreover, relevant information for this process can only be obtained during transients, as the system becomes non-observable once steady state is reached, when the heat flow through the capacitors falls to zero. Accordingly, large variations of the capacitor temperatures are required to make the parameter estimation possible, and an iterative process is likely to be necessary. In this case, an admissible error $\varepsilon = 10^{-10}$ J/K was set and convergence was achieved after 11 iterations. Fig. 4.7 shows the evolution C_1 and C_2 during the procedure.

4.3.2.3 Source heat estimation

The purpose of this test scenario is the correction of input disturbances. The constant heat source in the LPTM is initially assumed to deliver $Q_0 = 1$ W, when its actual value is $Q_0 = 10$ W. The problem is addressed performing a parameter estimation, in which

$$\boldsymbol{\rho} = \left[\; Q_0 \; \right] \tag{4.58}$$

Sensor readings include only the temperature at node 3.

$$\mathbf{o} = \left[\; T_3^{\text{sensor}} \; \right] \tag{4.59}$$

$$\mathbf{h} = \left[\; T_3 \; \right] \tag{4.60}$$

Fig. 4.8a shows the estimation results for input Q_0, and Fig. 4.8b its residual from $t = 4$ s to $t = 10$ s.

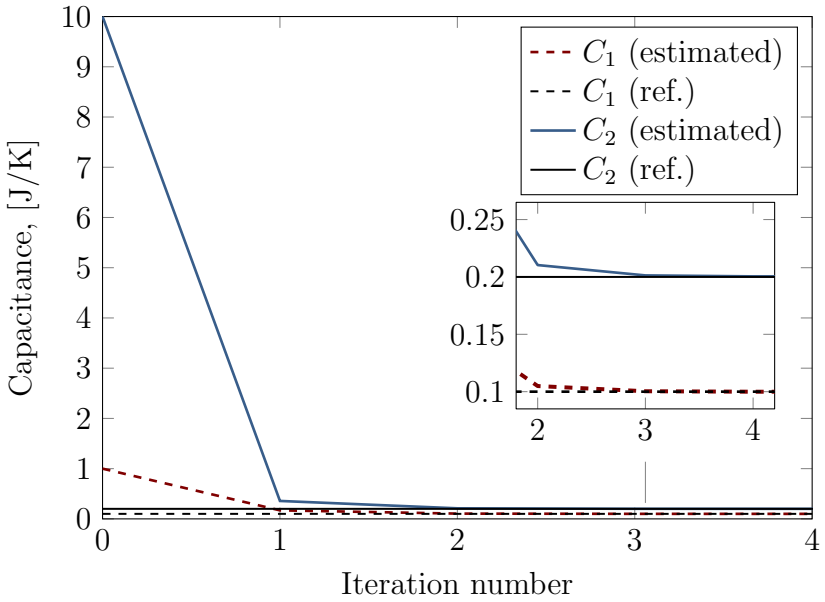


Figure 4.7: RC circuit: Convergence of capacitance values during iterative adjustment process.

4.3.2.4 State estimation with input disturbance

The final numerical experiment with the RC thermal circuit consists in the estimation of the system state when the input is subjected to unknown disturbances. The model used in the estimation assumes a constant heat input at the source $Q_0 = 10$ W, whereas the actual heat generation follows the expression $Q_0 = 10(1 + \sin(10\pi t))$ W. The unknown input is treated as a parameter,

$$\boldsymbol{\rho} = \left[\; Q_0 \; \right] \tag{4.61}$$

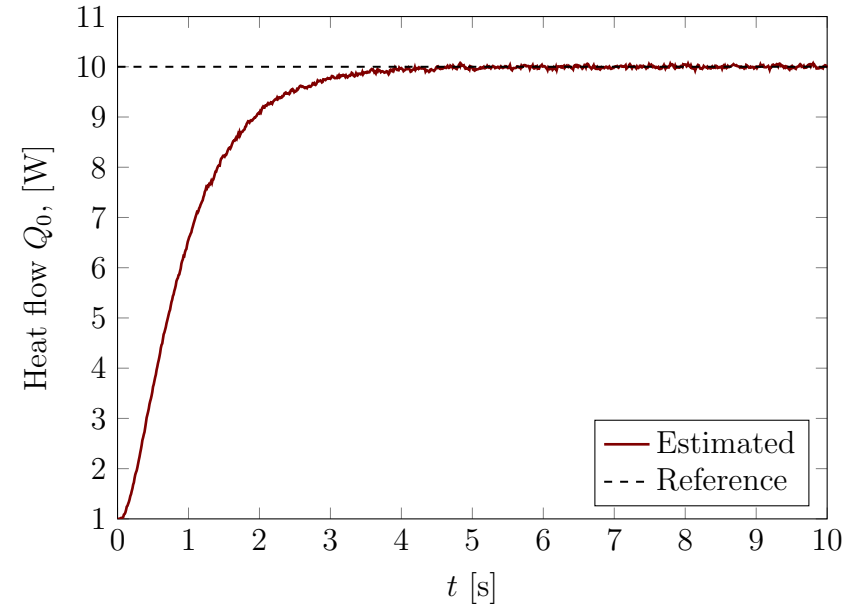
Two temperature sensors were placed at nodes 2 and 3.

Fig. 4.9a compares the temperature of the capacitor 1, T_2, obtained with the state estimation method to the reference solution delivered by the simulation of the system dynamics with exact parameters and input. The plot also shows the value of T_2 obtained in the simulation of the circuit dynamics if the input disturbance is not corrected. Fig. 4.9b contains the residual of the temperature of the node 2.

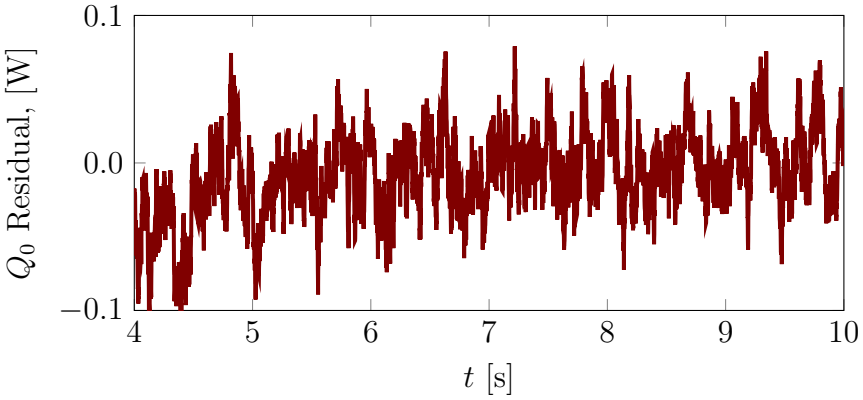
The computations required for the solution of this problem were performed on two different platforms, namely

- a conventional Dell XPS 15 7590 laptop running Linux (Laptop),
- a Raspberry Pi 4 (RPi4).

The features of each simulation environment are summarized in Table 4.2. These selected platforms can be integrated in an application like the one depicted in Fig. 4.1. The GCC version 10.2.0 was selected to build the Linux executables on Kubuntu, whereas version 8.3.0 was used on Raspbian OS.



(a) Estimation.



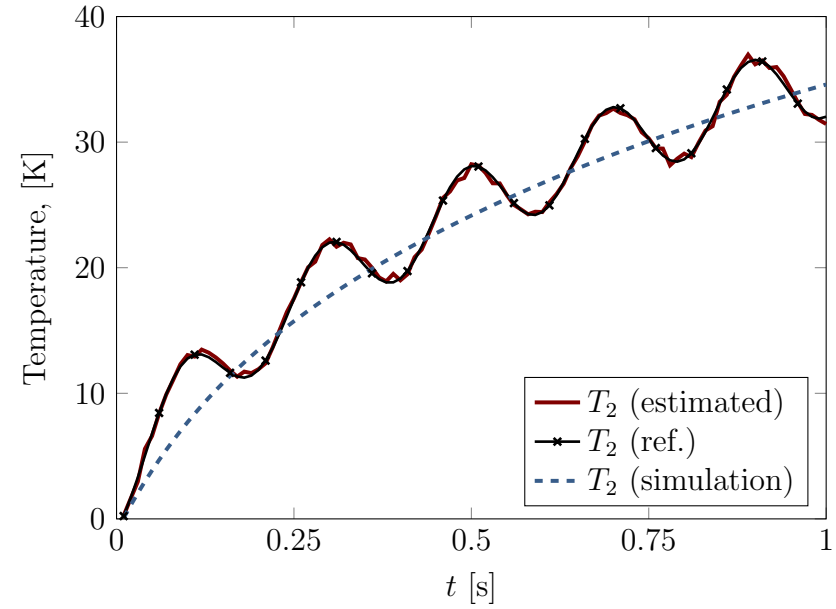
(b) Residual.

Figure 4.8: RC circuit: Evolution of the input Q_0 and its residual during source estimation.

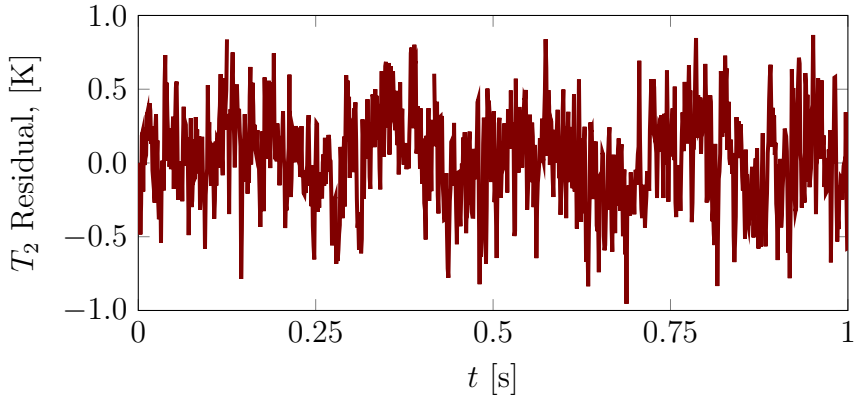
Table 4.2: Simulation platform features.

Platform	**CPU**	**Clock freq.** [GHz]	**Cores/ Threads**	**RAM** [GB]	**L1 cache** [kB]	**L2 cache** [MB]	**L3 cache** [MB]	**OS**
Laptop	Intel i7-9750H	2.6	6/12	16	192/192	1.5	12	Kubuntu 20.10
RPi4	ARM Cortex-A72	1.5	4/4	4	48/32	1	0	Raspbian 5.4.83

Table 4.3 shows the elapsed times in the solution of a 5-s state estimation of Case 4 (State estimation with input disturbance) of the benchmark problem. The RPi4 was able to perform almost 15 times faster than RT, whereas the Laptop environment completed the estimation around 86 times faster than RT.



(a) Estimation.



(b) Residual.

Figure 4.9: RC circuit: Temperature at node 2, T_2, with heat generation $Q_0 = 10(1 + \sin(10\pi t))$ W and its residual.

Table 4.3: Elapsed times in a 5-s state estimation of scenario 4 of the benchmark circuit.

Platform	**Elapsed time** [ms]
Laptop	58.4
RPi4	317.4

4.4 Industrial example: Three-phase inverter

The estimation methods in Section 4.2 were also applied to a three-phase inverter for automotive applications, shown in Fig. 4.10. An inverter is an Electronic Control Unit (ECU) that converts Direct Current (DC) into Alternating Current

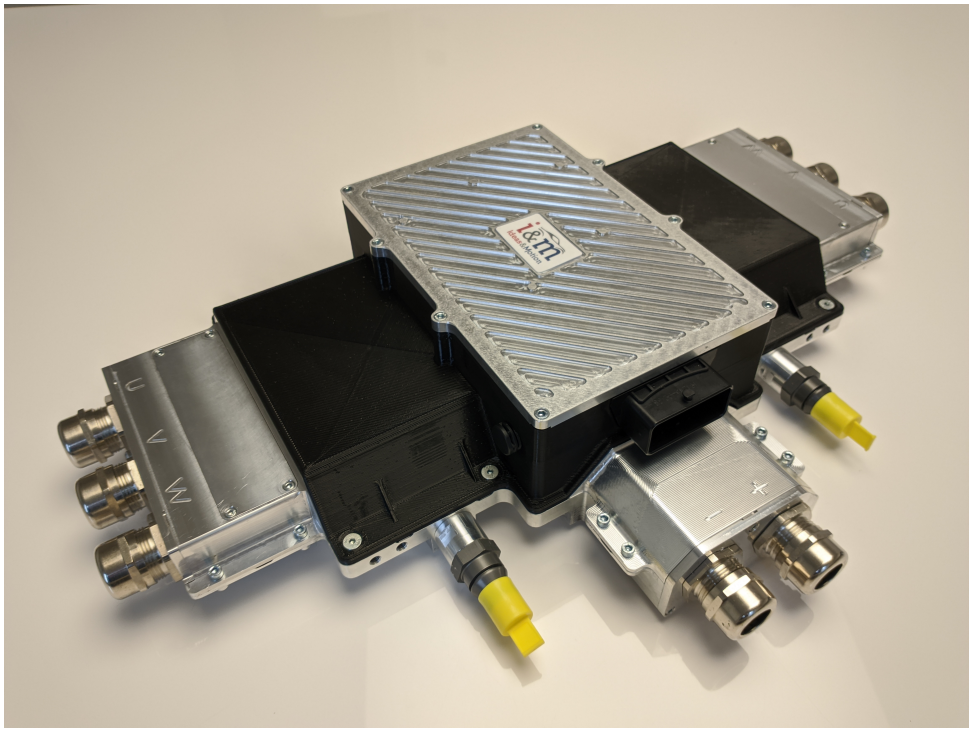


Figure 4.10: A dual, three-phase inverter for automotive applications.

(AC). Energy conversion is performed by commuting output potentials between DC voltage terminals and producing a moving average value that is applied to the load. Crucial components in this ECU are the power devices used to perform high voltage commuting under variable current loads at high frequency; this activity causes considerable heat losses that need to be dissipated.

The power modules in this research evacuate heat through a cooled aluminum heatsink on which three Direct Bonded Copper (DBC) substrate layers are mounted [154]. Each DBC layer supports an inverter phase that contains two MOSFET blocks. An overview of the power module structure and material stack-up are shown in Figs. 4.11 and 4.12.

Figs. 4.13 and 4.14 show the thermal equivalent circuits of a single branch and the heatsink and coolant part of the inverter, respectively. Each MOSFET block is modelled with a heat source $Q_{\tilde{\alpha}}$, where $\tilde{\alpha} = 1, ..., 6$, which represents its thermal losses, and a resistor R_{die} that stands for its thermal resistance. Due to its relatively small mass, the capacitance of the MOSFET block was neglected. The DBC is represented using a resistor R_{dbc} and a thermal inertia C_{dbc}. The welding between each DBC and the heatsink is a thin layer of silver paste represented by R_{sp}.

The last part of the inverter, the heatsink, is made up of two aluminum blocks. The first one is solid and represented by R_{hs} and C_{hs}, whereas the one in contact with the coolant is a pin-fin block defined by R_{w} and C_{w}. The LPTM of this inverter includes 14 nodes and 23 components: 6 heat sources, 12 resistors and 5 capacitors. When modelled with dependent variables, the thermal dynamics is described by a system of DAE with 5 differential and 32 algebraic equations, in which $n = 37$, $p = 5$, $m = 32$. The number of input values is $r = 8$, including the heat generation

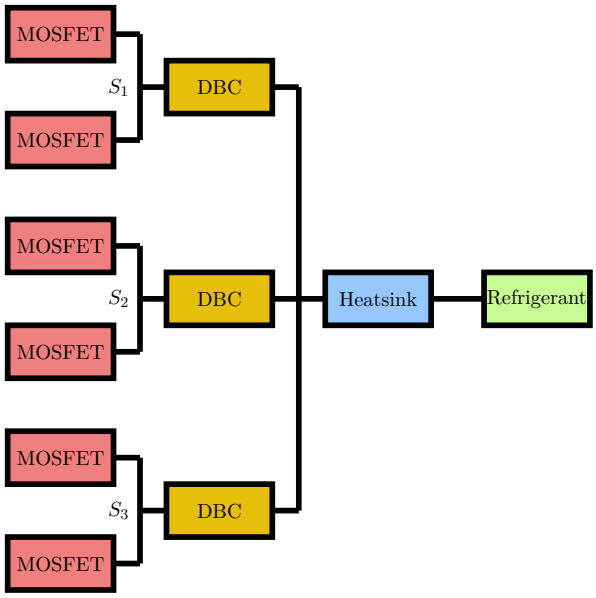


Figure 4.11: Power module structure. S_1, S_2, and S_3 denote the position of the temperature sensors used in this example.

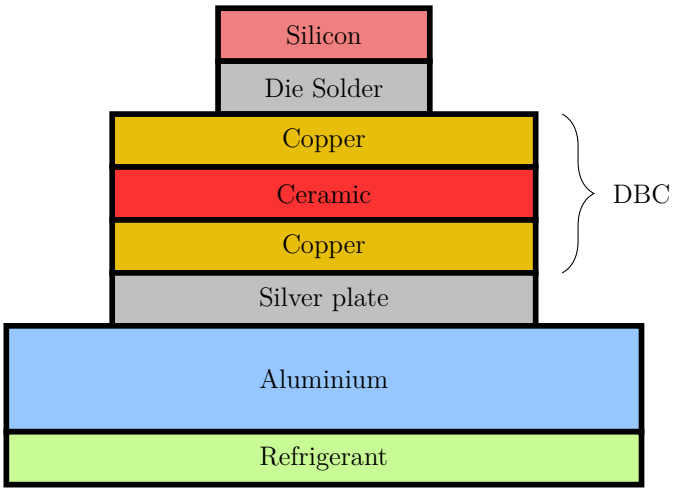


Figure 4.12: Inverter material stack-up (thicknesses are not to scale).

at each MOSFET block and the temperatures of the air and the refrigerant.

The estimation of junction temperatures is an important application of LPTM [76,82,83]. If this temperature rises above admissible levels the inverter performance degrades and may eventually lead to permanent damages in the component. Temperature sensors, however, cannot be placed at the MOSFET junction. In the majority of commercial power modules, temperature sensors are placed on the copper layer, at the points labelled S_1, S_2, and S_3 in Fig. 4.11. The junction temperature can be estimated from these readings using the methods in Section 4.2 and used later as input for temperature control methods, e.g., [155, 156].

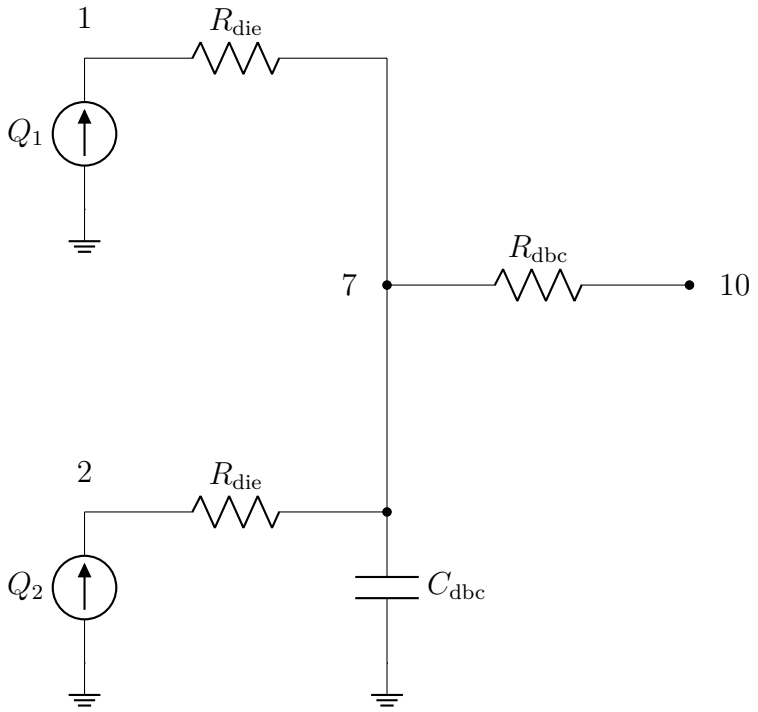


Figure 4.13: One branch in the thermal model of inverter, including two MOSFET and one DBC blocks.

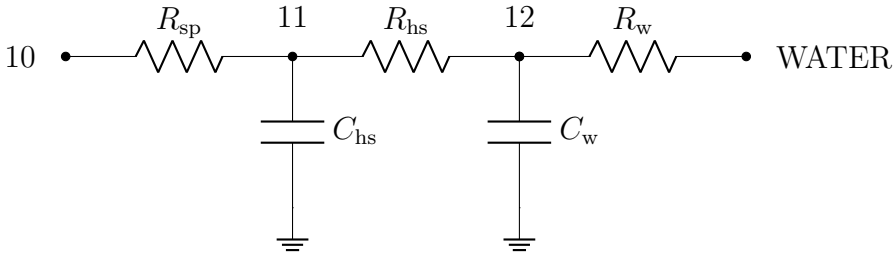


Figure 4.14: Heatsink and coolant blocks in the thermal model of the inverter.

4.4.1 Reference solution

A reference solution was generated for this example using a Computational Fluid Dynamics (CFD) simulation of the inverter. For the purposes of this study, this solution is considered to be exact. The heat losses at each MOSFET block had a constant value $Q_0 = 208$ W; the refrigerant temperature was $T_{\text{water}} = 343.15$ K.

4.4.2 LPTM adjustment

The LPTM that represents the thermal behaviour of the inverter needs to be adjusted so that it matches the reference solution. Once a topology is selected for the equivalent thermal circuit, initial values of its R and C parameters are assigned based on the physical properties of the components. Due to modelling uncertainties, these initial values may not exactly represent the actual ones necessary to describe the system dynamics. A first use of the estimation methods in Section 4.2 is the

determination of the values of the resistors and capacitors that need to be used in the LPTM. During this initial tuning phase, additional temperature sensor readings, other than the ones at points S_1, S_2, and S_3, are used. It is possible to use extra sensors during the characterization of the component in a test bench; however, these will not be available during regular operation of the component.

In all estimations of this example, the covariance matrix of the measurement noise $\mathbf{R}$ is a diagonal matrix whose elements are set to $0.25\ \mathrm{K}^2$ for the physical sensors and to $0\ \mathrm{K}^2$ for perfect measurements. Term $\mathbf{P}$ is a diagonal matrix as well, with elements initially set to 10 in SI units for the parameters and 1 in SI units for the states. Matrix $\mathbf{Q}$ was adjusted to make the PSD of the innovation constant.

4.4.3 Junction temperature estimation

Once the LPTM parameters have been adjusted, the equivalent circuit can be used to estimate the junction temperature of the inverter during operation. Four scenarios are considered:

- *Case 1*: The heat losses at the MOSFET are constant and equal to those in the reference solution, $Q_{01} = Q_0 = 208$ W.
- *Case 2*: MOSFET heat losses follow the piecewise function

$$Q_{02}(t) = \begin{cases} 0\ \mathrm{W}, & t < 0.1\ \mathrm{s} \\ 208\ \mathrm{W}, & 0.1\ \mathrm{s} \leq t < 1\ \mathrm{s} \\ 325\ \mathrm{W}, & 1\ \mathrm{s} \leq t < 2\ \mathrm{s} \\ 125\ \mathrm{W}, & t \geq 2\ \mathrm{s} \end{cases} \tag{4.62}$$

  shown in Fig. 4.15a.
- *Case 3*: MOSFET heat losses depend on the junction temperature T_{junc}

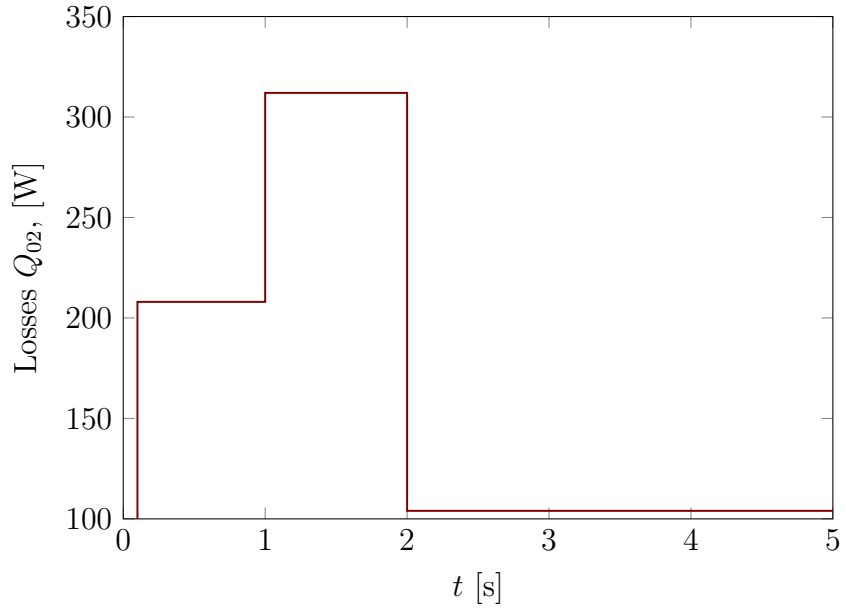
$$Q_{03}\left(T_{\mathrm{junc}}\right) = 1.355 T_{\mathrm{junc}} - 206.58\ \mathrm{W} \tag{4.63}$$

  shown in Fig. 4.15b.
- *Case 4*: The heat losses at the MOSFET are constant and equal to $Q_{04} = 600$ W, and the temperature of the refrigerant evolves according to the expression $306.15 - 13e^{-0.75t}$ K.

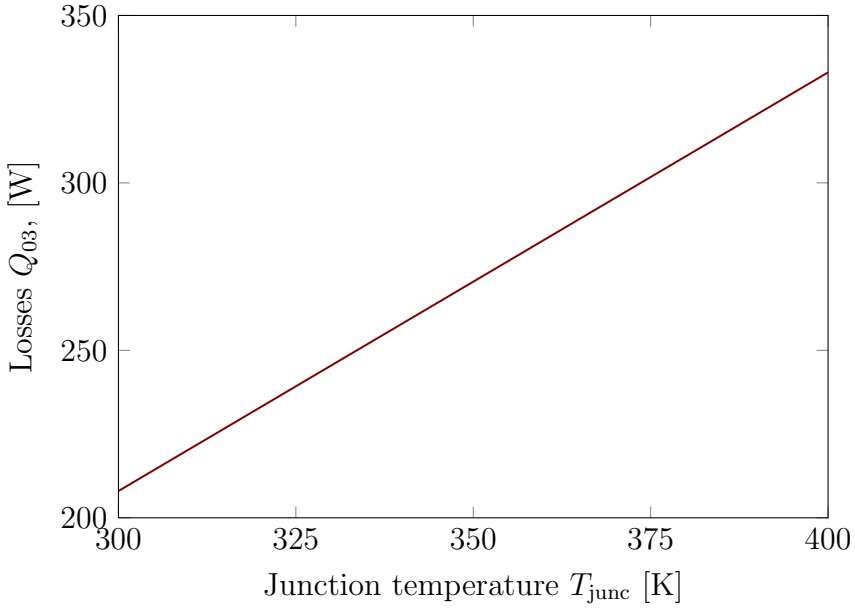
Temperature estimation during operation relies only on sensor measurements at points S_1, S_2, and S_3, which are the only readings available from the point of view of the DT of the inverter.

4.4.4 Results

The resistors and capacitors of the inverter must be adjusted prior to the use of the LPTM during operation inside a DT. In this Section, the EKF is first used to adjust the thermal parameters of the inverter, starting from a set of values determined using heat transfer formulas. Then, the adjusted thermal circuit is used during operation to estimate the junction temperature of the MOSFET blocks, using temperature sensors placed on the accessible nodes of the components.



(a) Case 2.



(b) Case 3.

Figure 4.15: MOSFET heat losses in inverter.

4.4.4.1 Resistor estimation

As mentioned in Section 4.4, results from the CFD simulation of the inverter are used as reference values in this example and considered as exact solution. During the parameter-adjustment stage, additional sensors, other than the ones placed at S_1, S_2, and S_3, are required to estimate the sought parameters. This is analogous to the instrumentation of a physical inverter on a test bench during its characterization phase. To keep the estimation process realistic, we avoided placing these sensors at nodes where it would have been difficult to install them on the physical component,

such as the MOSFET junction (nodes 1-6) or the silver plate between the DBC and the heatsink (node 10). For the characterization of LPTM in Figs. 4.13 and 4.14 three sensors were placed at nodes 7, 11, and 12, namely the copper layer on the DBC and the upper and lower parts of the heatsink. The measurements were assumed to have a WGN with mean 0 K and standard deviation 0.5 K as in Section 4.3. It must be mentioned that, with these sensors, the values of R_{die} and R_{sp} are not observable and cannot be corrected via estimation with the KF due to the absence of sensor readings in the welding layer and in the MOSFET junctions during the characterization stage.

Initial values for the resistors were determined using only geometry and material properties [157] as

$$R = \frac{e}{\kappa A} \tag{4.64}$$

where e is the thickness of the material, κ is the thermal conductivity and A is the cross section of the considered component.

The resistance values yielded by Eq. (4.64) will not generally match the actual properties of the inverter components. In the first place, Eq. (4.64) assumes that all the heat flow is perpendicular to the component section, which is often not the case. Moreover, the thermal and geometrical properties of the inverter elements are subjected to uncertainty and cannot be determined with absolute accuracy. Accordingly, the starting values delivered by Eq. (4.64) need to be corrected to describe the system dynamics appropriately.

The initial set of resistance values was improved through estimation with the KF. A 5-s simulation of the system dynamics during estimation case 1 was carried out to this end. In order to tune the resistors, only steady-state readings of the system magnitudes were used. The circuit capacitors are not yet adjusted in this stage, and they introduce deviations from the actual system behaviour during transients. For this reason, only sensor measurements between $t = 4$ s and $t = 5$ s were used. An iterative procedure was necessary to ensure the convergence of the resistance parameters below an admissible threshold $\varepsilon < 10^{-10}$ K/W; the convergence is shown in Fig. 4.16.

Only for comparison purposes, the CFD data are used here as well to calculate the equivalent resistors that would fulfill Fourier's Law between each pair of nodes of the LPTM as

$$R_{\text{ab}} = \frac{T_{\text{a}} - T_{\text{b}}}{Q_{\text{ab}}} \tag{4.65}$$

where T_{a} and T_{b} are the temperatures at the pair of nodes a and b, R_{ab} is the equivalent resistor between these nodes, and Q_{ab} is the heat flow through the resistor in steady-state. It should be stressed that these values are not used during the estimation process. The resistance parameters before and after estimation are shown in Table 4.4. Resistors R_{die} and R_{sp} could not be adjusted during estimation and their values were left unchanged.

The EKF estimation procedure may not converge depending on the starting values determined using Eq. (4.65). Table 4.5 summarizes the convergence limit of the initial parameter set. It is possible to achieve convergence starting from thermal resistance values about 50 times greater than the real values or one or two orders of magnitude smaller.

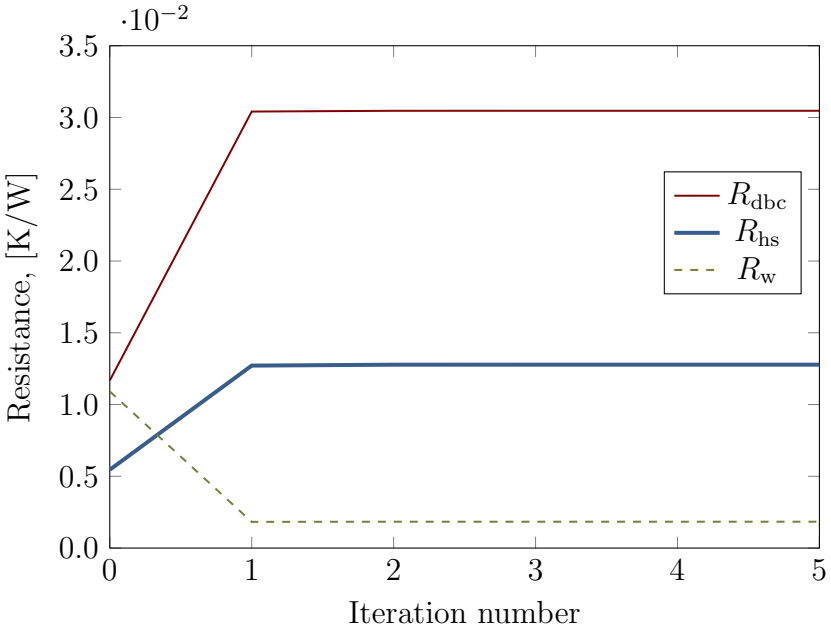


Figure 4.16: Convergence of the inverter resistor values.

Table 4.4: Initial and adjusted values of the thermal resistors in the LPTM of the inverter.

Resistor	**Initial value** [mK/W], Eq. (4.64)	**After estimation** [mK/W]	**CFD value** [mK/W] Eq. (4.65)
R^*_{die}	7.516	7.516	5.163
R_{dbc}	11.68	30.47	33.74
R^*_{sp}	3.771	3.771	2.579
R_{hs}	5.457	12.78	12.75
R_{w}	10.91	1.837	1.713

Table 4.5: Stability boundaries for the uncorrected starting values of the thermal resistor estimation for the inverter.

Limits	**Resistor**		
	R_{dbc} [mK/W]	R_{hs} [mK/W]	R_{w} [mK/W]
Upper bound	584	273	545
Lower bound	0.01	0.1	0.3

4.4.4.2 Capacitor estimation

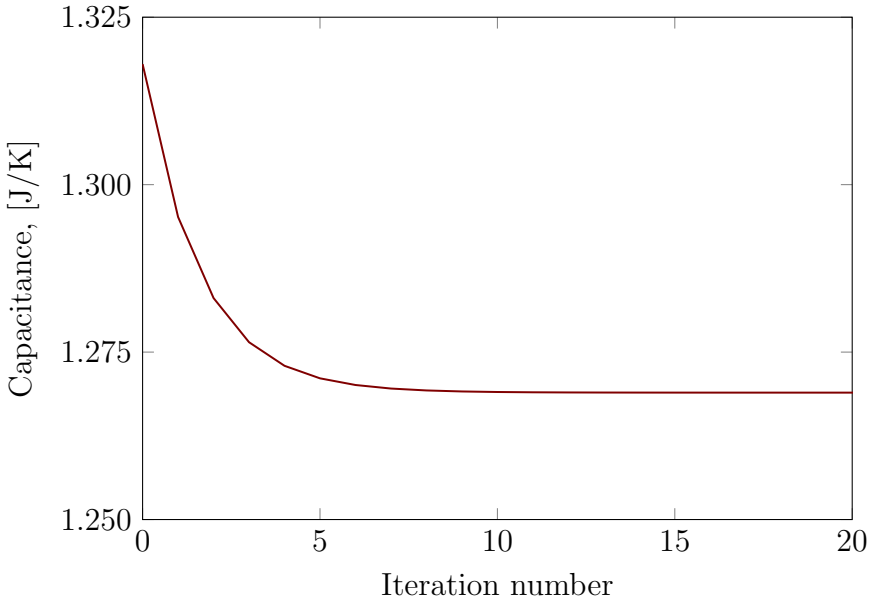
After the correction of the circuit resistors, capacitors C_{dbc}, C_{hs}, and C_{w} in Figs. 4.13 and 4.14 were adjusted with the KF. This adjustment is not possible using steady-state sensor readings, as they are not affected by capacitance values, so measurements during transients must be used instead. Each capacitor was tuned using the readings from the closest sensor; the length of the transient necessary to

perform the estimation was determined by the evolution of the node temperature. Again, an iterative process was required to achieve convergence; the threshold error was $\varepsilon = 10^{-10}$ J/K.

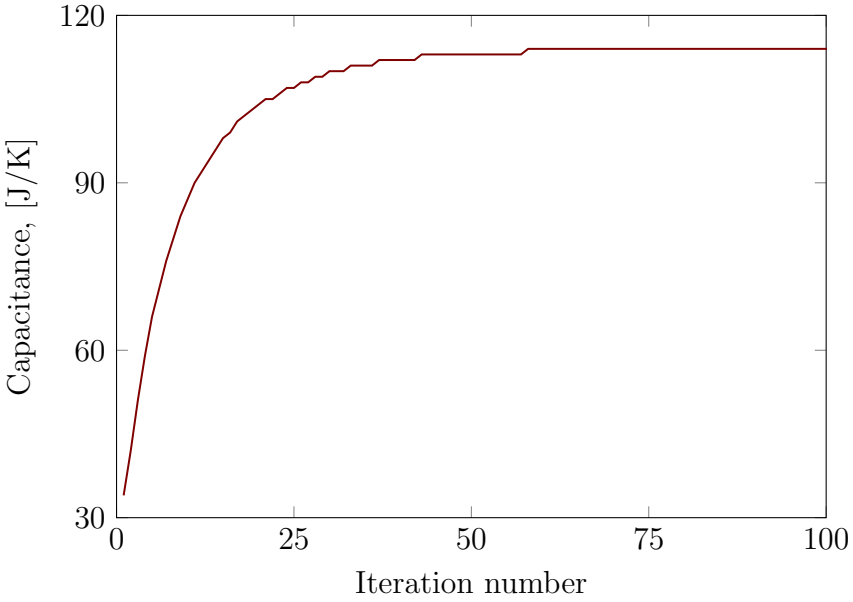
The sensors used to adjust resistors and capacitors were the same. Initial capacitance values were calculated as

$$C_{\mathrm{a}} = m_{\mathrm{a}}\, c_{\mathrm{p}}^{\mathrm{a}} \tag{4.66}$$

where m_{a} is the mass associated to node a, and $c_{\mathrm{p}}^{\mathrm{a}}$ is the specific heat of the material.



(a) Parameter C_{dbc}.



(b) Parameter C_{w}.

Figure 4.17: Convergence of the capacitors of the inverter.

Figs. 4.17a and 4.17b show the convergence of the adjustment process of capacitors C_{dbc} and C_{w}. The initial and corrected capacitance values are displayed on Table 4.6. Table 4.7 shows the effect of initial capacitance values on the convergence of the capacitor estimation. Admissible values range between 20-50 times greater and 100 times smaller than the true ones. It has been observed, however, that the number of iterations required to achieve convergence increases for large differences between the actual value and the starting point. Fig. 4.18 compares the tempera-

Table 4.6: Initial and adjusted values of the thermal capacitors in the LPTM of the inverter.

Capacitor	**Initial value** [J/K], Eq. (4.66)	**After estimation** [J/K]
C_{dbc}	1.32	1.27
C_{hs}	14.97	30.79
C_{w}	29.94	114.42

Table 4.7: Stability boundaries for the uncorrected starting values of the capacitor estimation.

Limits	**Resistor**		
	C_{dbc} [J/K]	C_{hs} [J/K]	C_{w} [J/K]
Upper bound	25	$1,500$	$3,000$
Lower bound	0.01	0.15	0.3

ture at node 7 of the inverter during a 5-s simulation before and after correcting the LPTM capacitor and resistor parameters adjusted with the EKF. The reference solution corresponds to the values delivered by the CFD simulation of the component.

4.4.4.3 Junction temperature estimation during operation

Once the system parameters have been adjusted, the LPTM can be used as thermal DT of the inverter during operation. The four cases in Section 4.4.3 were used as test scenarios. Only sensors placed on the DBC, at nodes S_1, S_2 and S_3, which are actually mounted on the inverter during operation, were used to estimate the junction temperature of the MOSFET. The sensors are considered to have a WGN with mean 0 K and standard deviation 0.5 K.

Fig. 4.19a shows the junction temperature obtained during a 5-s simulation of the circuit dynamics with the constant value of $Q_0 = 208$ W used in Case 1. In this case there are not input disturbances and heat losses at the MOSFET blocks are assumed to be known accurately. Both the direct forward-dynamics of the LPTM after correction of its parameters and the estimation with EKF were able to follow the reference value of T_1 obtained with CFD simulation.

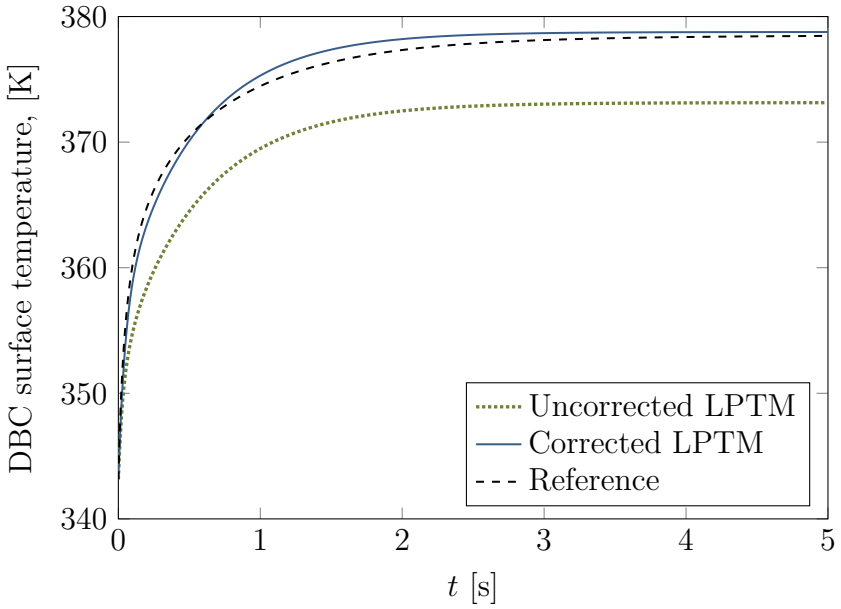


Figure 4.18: Temperature of node 7 of the inverter during simulation with uncorrected and corrected LPTM resistor and capacitor parameters.

Figs. 4.20a and 4.21a represent the simulation results obtained in test cases 2 and 3, in which the input heat losses are no longer constant, but are functions of time and temperature as defined in Eqs. (4.62) and (4.63). Fig. 4.22a presents the simulation in test case 4, with constant heat losses and a variable refrigerant temperature.

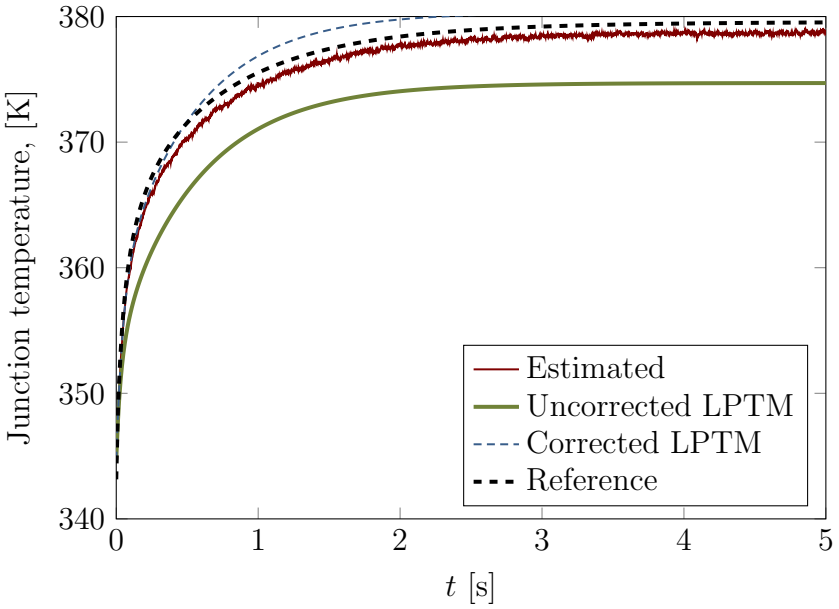
The estimation method was able to appropriately handle the uncertainty in $flow_0$ from the readings of the sensors mounted on the system. The direct simulation of the circuit dynamics is unable to follow these changes, even after adjusting the circuit parameters.

Is it worth mentioning that resistances R_{die} and R_{sp} could not be adjusted prior to operation due to the lack of appropriate sensor readings. In spite of this, the estimation of the junction temperature in all cases shows a reasonably low error.

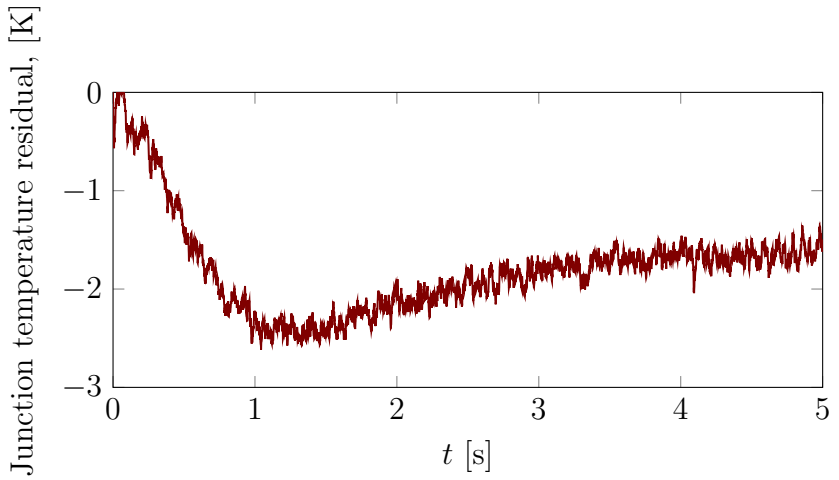
The obtained results confirm that the estimation methods in Section 4.2 can be used to overcome the effect of parameter and input uncertainties in the simulation or thermal equivalent circuits. In particular, when appropriate sensor readings are available, input disturbances can be accounted for in real-time during component operation.

Table 4.8: Elapsed times in a 5-s junction temperature estimation of each inverter case.

Platform	**Elapsed time**			
	Case 1 [ms]	Case 2 [ms]	Case 3 [ms]	Case 4 [ms]
Laptop	167.9	168.6	176.4	168.2
RPi4	847.9	855.8	850.9	864.1



(a) Estimation.



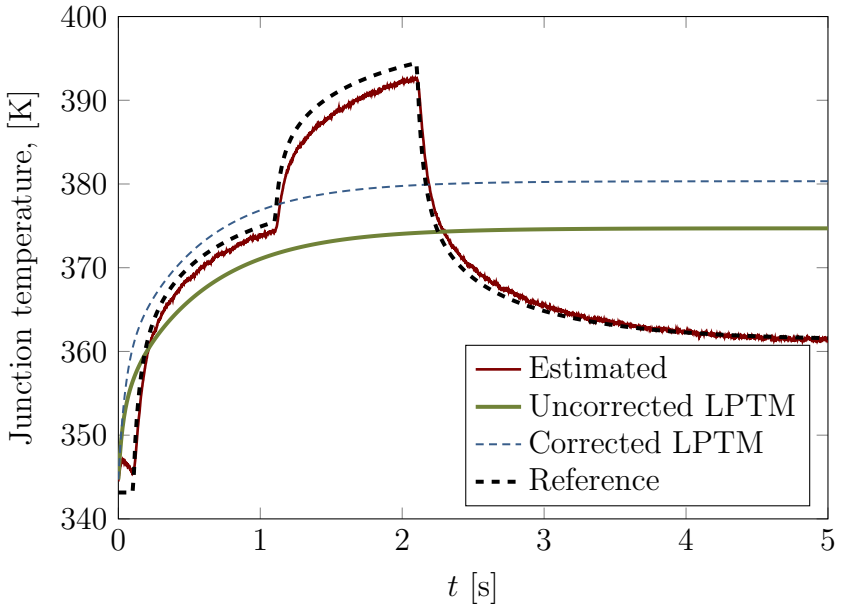
(b) Residual.

Figure 4.19: Estimation of the junction temperature of the inverter and its residual using a sensor on the copper layer for case 1.

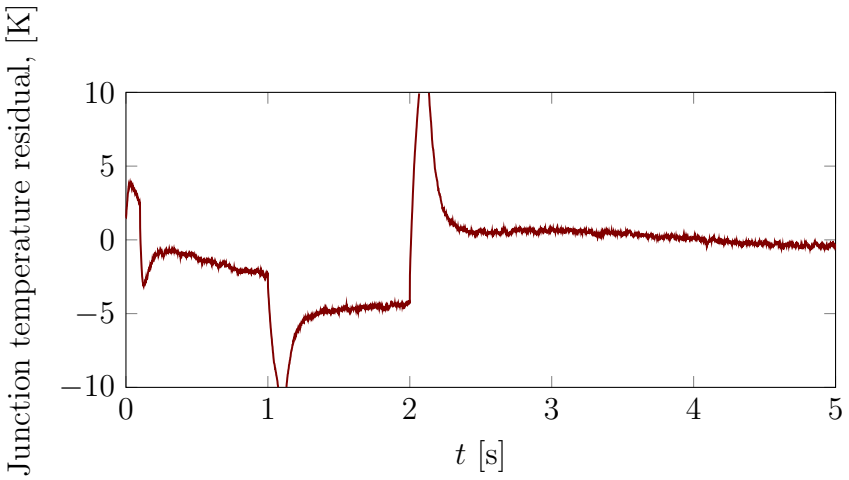
Table 4.8 shows the times elapsed in the solution of the four junction temperature estimation scenarios. These results confirm that the estimation method used is compatible with its use in RT applications with the computing platforms described in Table 4.2.

4.5 Conclusions

Monitoring the thermal behaviour of electronics components in powertrains makes it possible to improve their performance while avoiding excessive temperatures that could lead to their malfunction and damage. In most cases, however, sensors cannot



(a) Estimation.

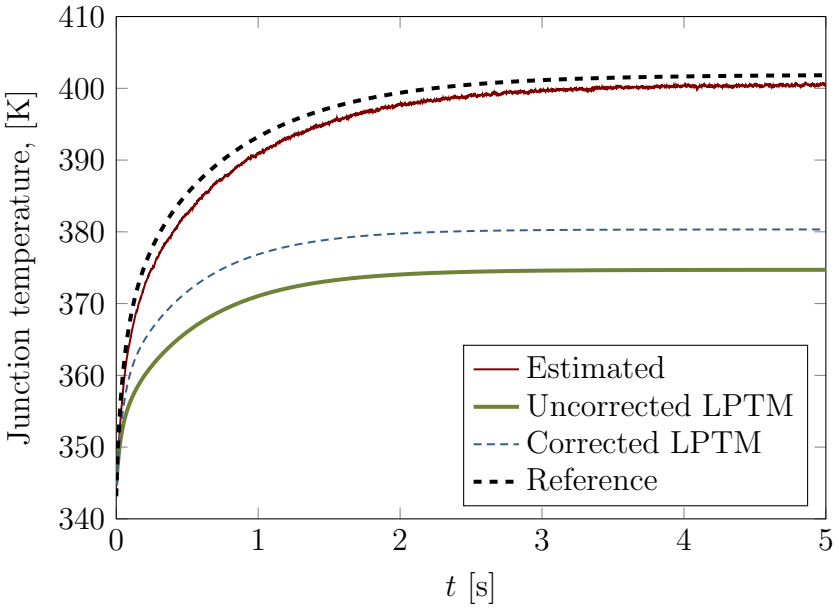


(b) Residual.

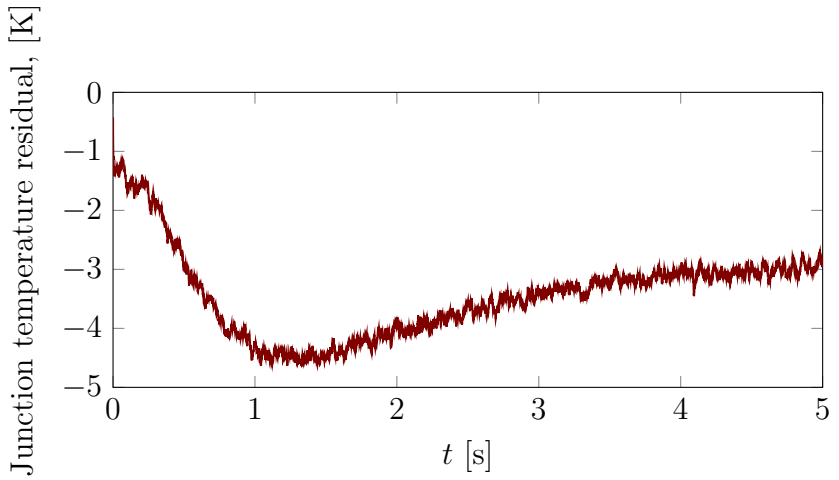
Figure 4.20: Estimation of the junction temperature of the inverter and its residual using a sensor on the copper layer for case 2.

be directly placed on critical locations. LPTMs, together with appropriate estimation techniques, can be used to develop DTs of e-powertrain components and keep track of these relevant temperatures during system operation.

Efficient LPTMs of e-powertrain components can be obtained with the methodology described in this Chapter, starting from a general-purpose formulation of the circuit dynamics based on dependent variables. This formulation enables the simple definition and assembly of individual circuit components, such as thermal resistors and capacitors, establishing their thermal parameters from the nature of the physical properties of the elements that they represent. The dynamics equations thus formulated are later transformed into a minimal set of differential equations that



(a) Estimation.

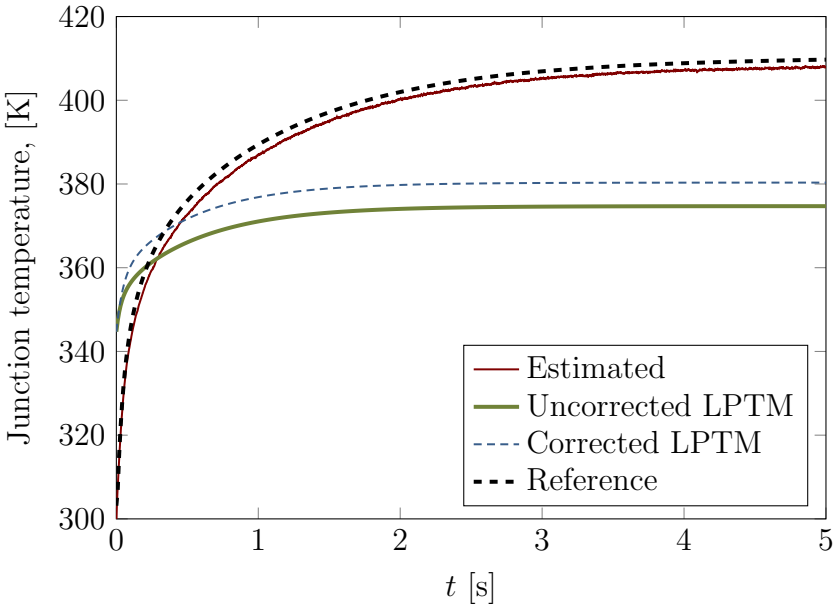


(b) Residual.

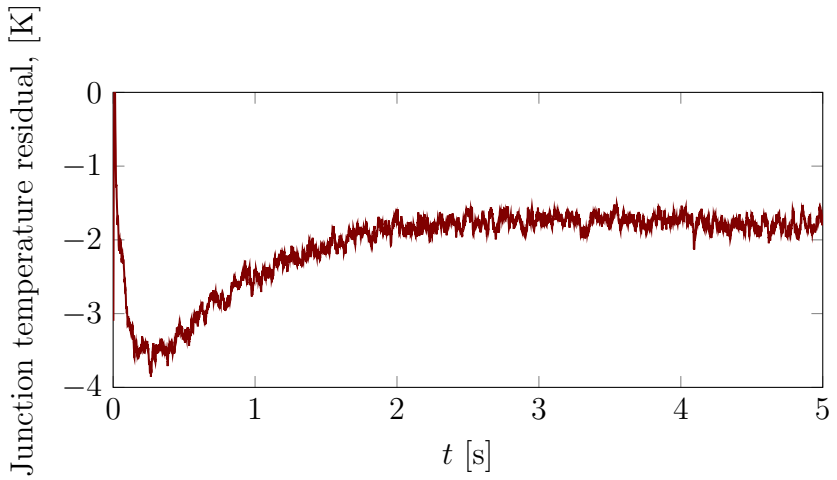
Figure 4.21: Estimation of the junction temperature of the inverter and its residual using a sensor on the copper layer for case 3.

can be used to develop computationally efficient estimation methods. The proposed method automatically generates this minimal set of equations of the LPTM for its use in estimation algorithms.

An input, parameter, and state estimator was put forward in this Chapter using the above-mentioned formulations and the EKF. The proposed method can be used in two stages. Prior to the operation of the component, the parameters of the LPTM can be estimated and adjusted to match the behaviour of the system that they represent. Once this stage is complete, the estimation method can be used to handle input disturbances and accurately monitor relevant temperatures in the component under surveillance, fusing information coming from both the LPTM that



(a) Estimation.



(b) Residual.

Figure 4.22: Estimation of the junction temperature of the inverter and its residual using a sensor on the copper layer for case 4.

represents the system and sensors mounted on the actual device. The methods were tested in the simulation of a benchmark RC thermal circuit and the thermal model of an automotive inverter. Results confirmed the ability of the proposed estimation approach to provide meaningful information about component temperatures, even in the presence of significant input disturbances.

Chapter 5

Co-simulation methods for real-time model-based system testing

This Chapter deals with the third area of interest in MBST environments identified in Chapter 2: the development of RT co-simulation algorithms to synchronize and keep stable the execution of virtual and cyber-physical systems. Following a co-simulation approach involves dividing the overall system under study into several subsystems and using a manager algorithm to orchestrate their execution and coordinate the exchange of data between them. From the point of view of the manager, each subsystem behaves as a *black box*, whose internals are unknown; the interaction with the rest of its environment is reduced to the exchange of a limited set of data at particular points in time. On the other hand, this strategy makes it possible to tailor the solver of each subsystem to its particular dynamics, time scale, and level of detail with which the modelling needs to be performed. The resulting solution is much more modular and flexible than using a single solver tool for the whole assembly.

In the context of this thesis, a framework based on co-simulation techniques has been developed to test and validate diverse algorithms, and also adapted to handle physical and virtual subsystems. This framework has been designed following the FMI standard specifications. Three important constraints have been identified when real components are involved in co-simulation environments: RT performance is mandatory, the use of constant integration step-sizes is a practical necessity in most applications, and explicit (noniterative) co-simulation schemes are required, as repeating integration steps is not possible for physical components. The use of explicit co-simulation schemes, however, often gives rise to the introduction of artificial energy in the system dynamics due to the discrete-time communication between the subsystems and the co-simulation manager. This excess energy degrades co-simulation accuracy and may eventually compromise the stability of the overall integration process, unless it is properly handled by the co-simulation architecture.

In order to address this problem, an indicator to keep track of the excess energy was put forward to monitor the accuracy and stability of co-simulation results. A novel method to eliminate the excess energy, based on extending the original

purpose of the above-mentioned monitoring indicator, was also implemented within this framework. The ability of this method to deliver stable and accurate explicit co-simulation results was tested with several mechanical and multiphysics examples.

5.1 Introduction

Since its introduction in industrial environments in the 1960s, computer simulation of engineering systems has proved itself a valuable tool to reduce Research & Development costs and shorten product development cycles. Physical dynamic simulations have been used widely with success. In particular, the use of simulations in MBS dynamics has eased the study of complex phenomena or large systems, which would have been very complicated and time-demanding to do by hand, such as contacts and flexibility [158, 159]. Similar examples can be pointed out in other fields, such as the simulation of electronic circuits [160, 161].

Nowadays, most systems for industrial applications are usually described by complex dynamics equations, in which the involved components have disparate natures and time-scales. In order to build correct computational models of such setups, it is necessary to generate realistic descriptions of the components and the interactions between them. An electric vehicle can be a representative example of this, in which mechanic, electric, electronic and even thermal effects are related and influence each other. The selection of the right solvers is a key element here. A solver is an algorithm that computes the evolution of the states and outputs of a dynamical system [7]. The set of trajectories followed by the states and outputs is known as the behaviour trace of the system. System-level simulations can be conducted following a *monolithic* or a *co-simulation* approach.

Monolithic simulation describes the response of a system with a single and all-encompassing set of equations. In this configuration, a single solver is in charge of the integration of the whole system. This solver typically has unlimited access to the details of every component of the system, as these details are often required to assemble and solve their dynamics equations [86]. Monolithic integration, however, suffers from two large weaknesses. On the one hand, a generic solver is required to integrate the complete set of equations regardless of their nature, which may work against the overall performance of the simulation. Besides, this solver needs to be adapted to the step-size that corresponds to the fastest system dynamics. On the other hand, the solver needs to have full access to the implementation details of each component in most cases. In a considerable number of industrial applications, this is not possible, because of intellectual property rights. In these cases, it is frequently not possible to know the details of component models from the point of view of the system simulation manager. This is often the case when combining component models from different vendors. Nevertheless, monolithic solutions tend to be robust and efficient, and have been successfully used in a wide range of applications, e.g., mechatronics [86] and hydraulically actuated mechanical systems [87, 162].

Co-simulation, conversely, consists in describing a certain system as a set of subsystems that are integrated separately and exchange information between them. The coordination and synchronization of this data exchange is borne by an *orchestrator* or *co-simulation manager*. Each subsystem has its own set of variables, step-size,

and *solver*, which can be tailored to its particular modelling and solution needs. Unlike monolithic simulations, co-simulation features two important advantages. On the one hand, each subsystem is integrated by a specialized solver with a step-size and features adapted to its own dynamics. On the other hand, each subsystem only exposes to the co-simulation environment a reduced set of *coupling variables*, which effectively prevents the revelation of confidential implementation details. This makes it easier to assemble systems made up of models developed by several vendors and implemented with different software tools; moreover, this task is further simplified if co-simulation standards are adopted by all participants.

Co-simulation also presents practical drawbacks. The communication between the co-simulation manager and subsystems takes place only at discrete-time points. Once each subsystem reaches a communication point, it sends its new outputs to the co-simulation manager and this, in turn, passes the new inputs to each subsystem. The time interval between two communication points is commonly denoted as *macro time-step*. Between two consecutive communication points, each solver proceeds with the integration of its own states without interacting with its environment, i.e., no new information from the other subsystems is available until the next communication point is reached. When a subsystem requires the use of updated inputs at other times, diverse techniques, such as extrapolation, are used to approximate the real value of the input at that point in time. This time-discrete exchange of information causes discontinuities in the subsystem inputs, which deteriorates the quality of the results and may introduce high-frequency dynamic components in the system motion [119].

The co-simulation process consists of two stages: initialization and execution. The initialization stage may require an iterative procedure, in which the subsystems exchange information until certain initial constraints are satisfied. For instance, [93] describes a general algorithm to initialize co-simulation schemes. Once the initialization is complete, the execution phase –the actual numerical integration of the system dynamics– can begin.

In a MBST context, co-simulation appears as a necessary component in cyber-physical applications. In CPS setups, real-world components can be seen as time-continuous systems, while the virtual ones are time-discrete ones. Both exchange information at synchronization or sampling points; between these, their dynamics evolve separately. These characteristics make CPSs a particular case of co-simulation environments, which will feature the advantages and disadvantages pointed out in the previous paragraphs. Co-simulation techniques address these issues and allow to simulate and synchronize the data exchange between real and virtual components. Moreover, co-simulation makes it possible to connect several components in the virtual environment that are simulated using different solver tools, each with its own tailored integrator, step-size, and level of detail. Co-simulation techniques can then be used to build virtual environments in an effective, modular way, including subsystems that in practice behave like black boxes, without providing information about its internal behaviour. A CPS test bench for e-powertrain components clearly illustrates the relevance of co-simulation in this kind of application. In it, the physical components under test are interfaced to a computer simulation of the full vehicle and its environment. This interface requires the use of *hybrid*, i.e., continuous-discrete,

co-simulation methods. Additionally, the virtual vehicle simulation may include several phenomena, e.g., mechanical, electric, and hydraulic effects, thus becoming a multiphysics problem. In these cases, the use of co-simulation is a reasonable option to develop an effective solution platform, especially considering that RT performance is required, due to the interaction with real-world components, and that co-simulation can be used to enable or ease the parallelization of code execution.

RT co-simulation, however, features a series of particularities that need to be properly addressed in practical applications. In general, constant step-sizes are considered a necessity due to the difficulty to modify them in physical components. Besides, explicit, noninterative coupling schemes are often mandatory in cyber-physical applications, because the real-world components cannot retake their integration steps, which would require them to go back in time. Virtual subsystems could indeed retake their steps using updated inputs delivered by the physical subsystems, but in practice this scheme is rarely used. Explicit, fixed-step schemes thus become the standard in RT and cyber-physical co-simulation applications. This introduces an additional complexity into the co-simulation environment, as explicit schemes are more prone to instability and numerical issues than their implicit counterparts, in which the iterative process can be used to correct errors from their initial predictor step. Noniterative schemes often modify the overall system energy, due to the need to perform input extrapolation at the discrete-time interface. Accordingly, when explicit co-simulation is used, it is necessary to monitor the stability and accuracy of the results, e.g., by means of specific indicators [117]. This information can then be used to design correction algorithms to keep the co-simulation process stable and representative of the true system behaviour [109, 119]. This Chapter presents a method to evaluate such indicators from the coupling variables exchanged at the co-simulation interface, and to correct energy deviations introduced by input extrapolation, which can be applied to a wide range of mechanical and multiphysics problems.

5.2 Co-simulation configuration options

Solver coupling or co-simulation can be performed in a wide variety of configurations. In general, a large amount of options can be adjusted within a single simulation problem, in order to arrive at a particular *co-simulation configuration*. The configuration selection has, in most cases, a critical impact on the results delivered by the numerical integration. This Section includes a summary of the most relevant co-simulation options, which can be categorized into the following groups:

- *Co-simulation schemes*
- *Time grids*
- *Coupling variables selection*
- *Input extrapolation*

Besides, a brief discussion of the FMI standard is included, because of its importance in industrial applications of co-simulation and the conditions that it imposes on the selection of configuration options.

5.2.1 Co-simulation schemes

A common classification used in co-simulation is to divide the co-simulation schemes into *explicit* and *implicit*. Explicit schemes are not allowed to repeat an integration step to obtain a more accurate solution using updated outputs from the subsystems. Implicit schemes, on the other hand, iterate over the macro-step to arrive at improved state and output values at the following communication point. This requires that the involved subsystems are able to perform *rollback*, i.e., resetting their state to that of the previous communication point to retake the step with the new information available, in a predictor-corrector fashion [102]. In general, explicit schemes are less stable and accurate than implicit ones [104].

It must be noted that the nomenclature and notation in this area have not fully converged yet [163], and so the names given to the methods here described may not match those used in some publications in the literature. Explicit, noniterative methods are sometimes labelled as *weak coupling*. In other publications, this term refers to co-simulation in general. Implicit, iterative co-simulation schemes are sometimes referred to as strong coupling, although this may lead to confusion with monolithic schemes. In some publications, semi-implicit coupling means an iterative coupling scheme, in which the iteration procedure is interrupted after a fixed number of repetitions of the macro step, as opposed to iterate upon a certain threshold error is attained. In this thesis, the terms explicit/noninterative and implicit/iterative will be used preferentially.

According to the order in which subsystems are evaluated, one can distinguish between parallel and sequential coupling schemes. The *Jacobi* and *Gauss-Seidel* schedules are commonly selected options, and are described next.

5.2.1.1 Non-iterative Jacobi scheme

A Jacobi scheme integrates all the subsystems in parallel. At every communication point the co-simulation manager receives the outputs of each subsystem and returns its inputs. After the information exchange, each subsystem moves forward from the current communication point t_k to the next one t_{k+1} with no further interaction with its environment. The Jacobi scheme is widely used to take advantage of parallel co-simulation architectures.

Figure 5.1 represents the diagram of a non-iterative Jacobi scheme for a two-subsystem co-simulation. The scheme process is summarized in four stages:

- *Stage 1:* At communication time t_k each subsystem $\tilde{\alpha}$ sends its outputs $\mathbf{y}_{\tilde{\alpha}}^k$ to the co-simulation manager.

- *Stage 2:* The co-simulation manager processes the outputs and send the inputs $\mathbf{u}_{\tilde{\alpha}}^k$ to each subsystem $\tilde{\alpha}$.

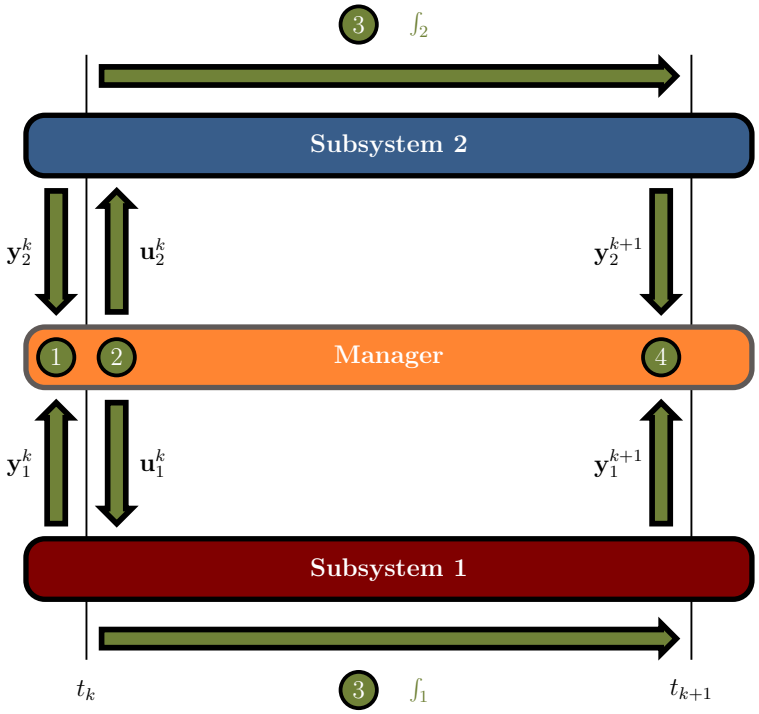


Figure 5.1: Diagram of a non-iterative Jacobi coupling scheme.

- *Stage 3:* Subsystem solvers integrate the states and advance in time until the next communication point, at time t_{k+1}. The integration can be executed concurrently, because each subsystem does not require further information from its environment.
- *Stage 4:* The process starts again at the next communication step t_{k+1}.

5.2.1.2 Iterative Jacobi scheme

The iterative Jacobi coupling scheme is the implicit counterpart of the one described in the previous Section 5.2.1.1. Unlike the explicit Jacobi, this iterative scheme requires the subsystems to retake their integration steps to achieve the fulfilment of a certain tolerance criterion, defined for instance as

$$\Xi_{\tilde{\alpha}}\left(t_{k+1}\right) = \left\| \mathbf{y}_{\tilde{\alpha}}^{i}\left(t_{k+1}\right) - \mathbf{y}_{\tilde{\alpha}}^{i-1}\left(t_{k+1}\right) \right\| < \varepsilon_{\tilde{\alpha}} \qquad \forall \tilde{\alpha} \in [1, n_{ss}] \tag{5.1}$$

where i is the iteration number, $\tilde{\alpha}$ is the subsystem index, n_{ss} is the number of subsystems, $\varepsilon_{\tilde{\alpha}}$ is the tolerance error for the subsystem $\tilde{\alpha}$, and $\Xi_{\tilde{\alpha}}\left(t_{k+1}\right)$ is the norm of the difference in the subsystem $\tilde{\alpha}$ outputs between two consecutive iterations.

Figure 5.2 represents the diagram of an iterative Jacobi scheme for a two-subsystem co-simulation. The process is summarized in 5 stages:

- *Stages 1-4:* These are identical to those in the non-iterative Jacobi scheme described in Section 5.2.1.1.
- *Stage 5:* The subsystems perform a rollback to t_k. The outputs delivered from each subsystem to the co-simulation manager at timestep t_{k+1} are used as inputs to retake the integration step from t_k to t_{k+1}.

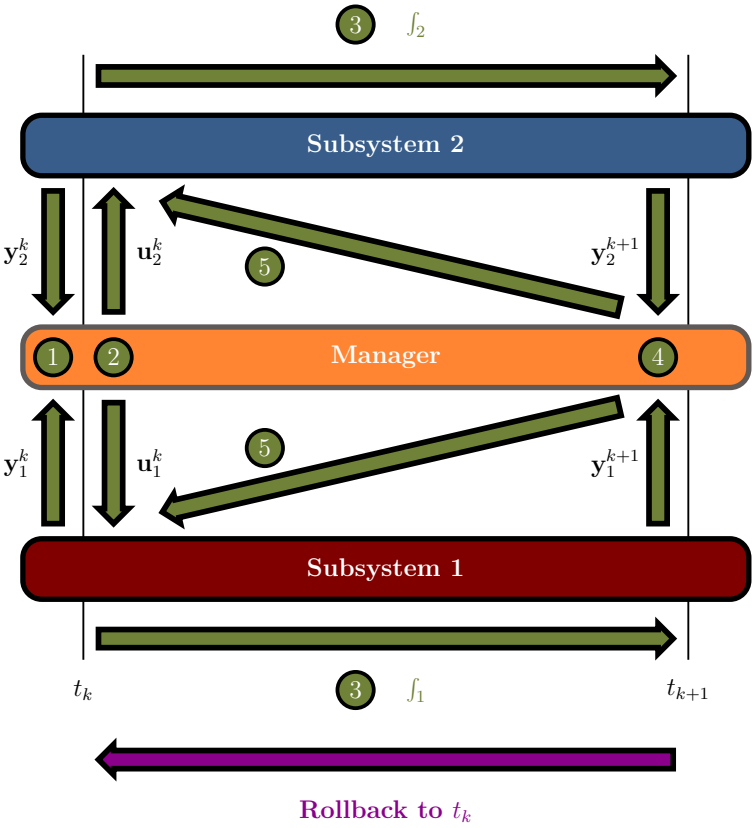


Figure 5.2: Diagram of an iterative Jacobi coupling scheme.

Stages 1-5 are repeated until a convergence criterion like the one in Eq. (5.1) is satisfied.

5.2.1.3 Non-iterative Gauss-Seidel scheme

The Gauss-Seidel scheme, also called zig-zag, integrates the subsystems sequentially. This means that the outputs of a subsystem obtained upon completion of its integration at communication point t_{k+1} can be available as inputs at time t_k for other subsystems that have not started their integration yet. In contrast to the Jacobi scheme, the Gauss-Seidel schedule can be executed in different manners depending on the subsystem execution order. As a matter of fact, the co-simulation results may differ significantly as a result of modifications of this order.

Figure 5.3 represents the diagram of a non-iterative Gauss-Seidel scheme for a two-subsystem co-simulation. The process is summarized in five stages:

- *Stage 1:* At communication point t_k one of the subsystems, e.g., subsystem 2 sends their outputs $\mathbf{y}_{\tilde{\alpha}}^k$ to the co-simulation manager.
- *Stage 2:* This subsystem 2 advances in time until the next communication point t_{k+1}.
- *Stage 3:* The outputs of this subsystem $\mathbf{y}_2^k$ at time t_{k+1} are now available for the rest of the subsystems as inputs at communication point t_k.

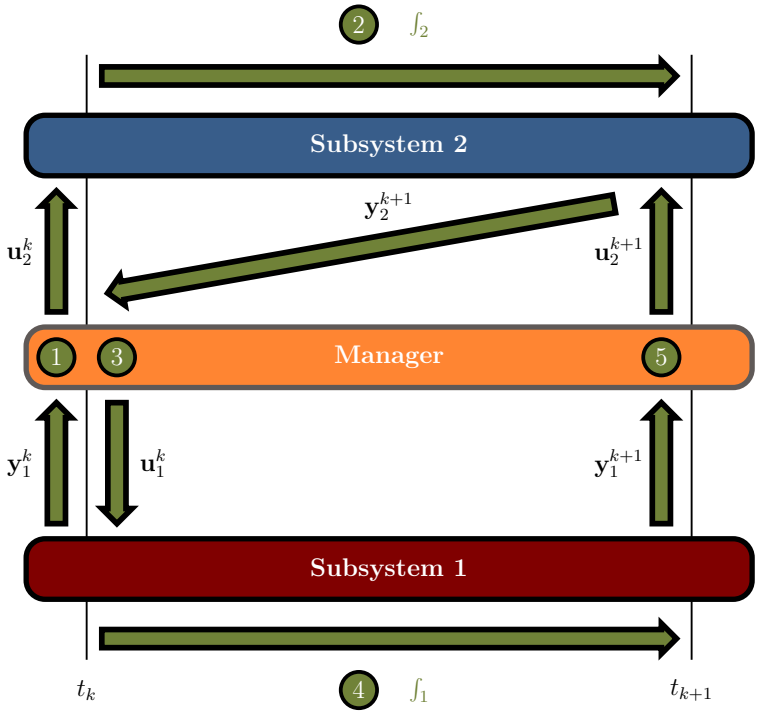


Figure 5.3: Diagram of a non-iterative Gauss-Seidel coupling scheme.

- *Stage 4:* Stages 1 to 3 are repeated for the rest of subsystems sequentially, according to the defined order of subsystem execution.
- *Stage 5:* Once the integration of all the subsystems has been completed, the process starts again at time t_{k+1}.

Gauss-Seidel schemes are in general more stable than Jacobi ones; however, they hinder the parallelization of subsystem execution, because some subsystems cannot start the integration of their dynamics at time t_k until others have completed theirs and are ready to deliver their outputs at a future time t_{k+1}.

5.2.1.4 Iterative Gauss-Seidel scheme

An iterative Gauss-Seidel coupling scheme is the implicit counterpart of the one described in the previous Section 5.2.1.3. Subsystems in an iterative Gauss-Seidel schedule must roll back in time until a convergence criterion, similar to the one described in Eq. (5.1), is reached.

Figure 5.4 represents the diagram of an iterative Gauss-Seidel scheme for a two-subsystem co-simulation. The process is summarized in six stages:

- *Stages 1-5:* These represent the same process of the non-iterative Gauss-Seidel scheme described in Section 5.2.1.3.
- *Stage 6:* The subsystems roll back to t_k. The outputs delivered by each subsystem to the co-simulation manager at time t_{k+1} are used as inputs to retake the integration step at t_k.

These stages are repeated until convergence is reached.

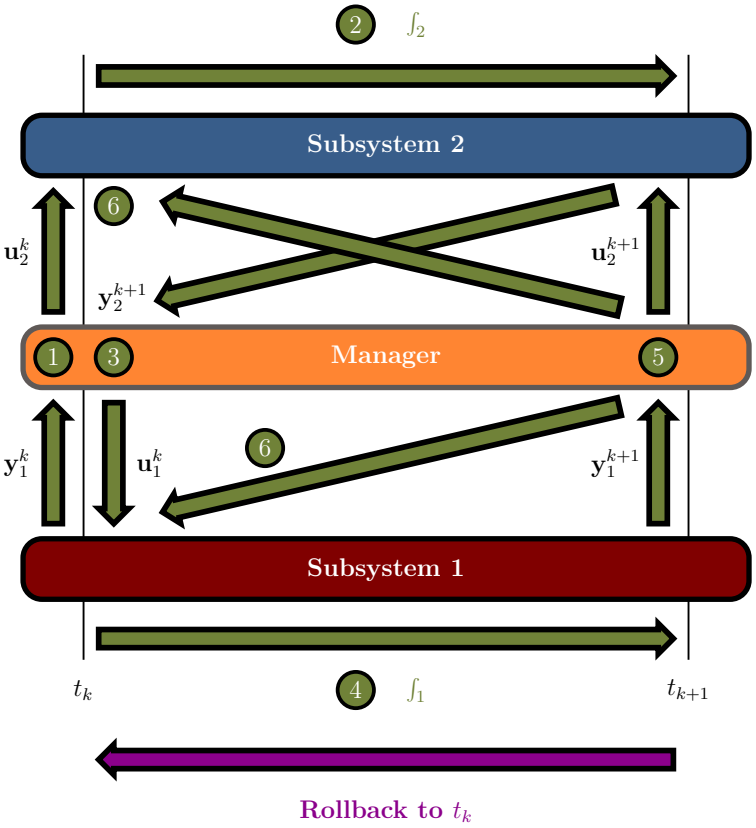


Figure 5.4: Diagram of an iterative Gauss-Seidel coupling scheme.

5.2.2 Time grids

A time grid denotes the specific set of time points obtained after the discretization of time in each subsystem. A co-simulation environment contains at least as many time grids as subsystems, because establishing an integration step-size, or *micro step-size*, in each subsystem determines a time discretization in it. Moreover, the communication interval with the co-simulation manager, or *macro step-size*, does not necessarily have to be aligned with the grid defined by the subsystem integrator, and so it defines its own time grid as well. For the sake of clarity, the following discussions refer only to the grids determined by the subsystem macro step; the issue of aligning macro and micro grids is not considered here.

Depending on the kind of time grids used, one can distinguish

- *Matching and non-matching time grids:* If the step-sizes of all the subsystems are multiples of each other, the resulting time grids are matching. This means that there is a certain macro step-size that defines a set of communication points at which all the subsystems exchange information with the co-simulation manager. It is the most used time grid in co-simulation. If this condition is not fulfilled, the time grids are non-matching. The implementation and synchronization of these co-simulation setups is more complicated [108] and will not be used in this Chapter.

- *Constant and variable step-sizes:* The communication step-size can be kept

constant during the length of the simulation; this configuration is commonly used in RT applications. It is also possible to adapt this step-size during runtime to match the subsystem dynamics and obtain a more stable integration [117, 164].

If matching time grids with constant step-sizes are used, an additional classification can be made:

- *Single-rate co-simulation:* All the subsystems share the same communication step or macro step-size H. Figure 5.5 illustrates a single-rate co-simulation scheme for two subsystems, in which, additionally, the micro steps of the subsystems are equal to the communication step-size: $h_1 = h_2 = H$.

  When the subsystems have different time scales, this scheme often leads to a less computationally efficient execution, due to the use of a single step-size, which is usually determined by the fastest subsystem. For instance, when coupling electronic and mechanical subsystems, using a step-size of microseconds may be necessary to capture electronics phenomena, but integrating the mechanical subsystem with such a step-size will lead to a considerable computational overhead that will not result in improved simulation results.

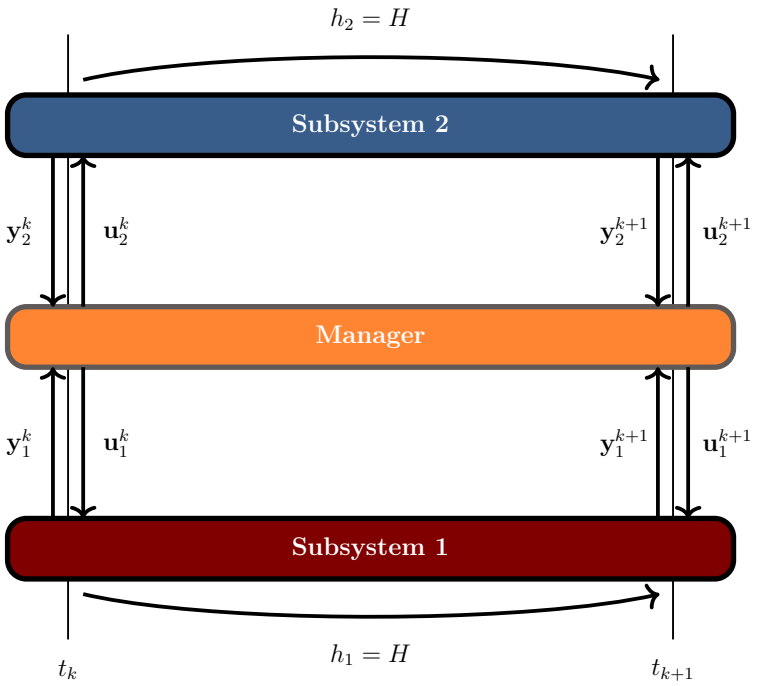


Figure 5.5: An explicit, single-rate co-simulation scheme.

- *Multi-rate co-simulation:* Different time scales in the subsystems can be dealt with by selecting appropriate macro step-sizes for each of them. This scheme is often computationally advantageous in multiphysics setups. In this case, the co-simulation manager has to orchestrate the exchange of outputs and inputs with their correct time-stamps. In particular, fast subsystems may need to receive their inputs at times when outputs from slower subsystems have not

been evaluated, as shown in Fig. 5.6. This issue can be addressed by the use of extrapolation techniques in the manager block, as described in Section 5.2.4.

Figure 5.6 illustrates an example of multi-rate co-simulation with two subsystems, in which subsystem 1 features a macro step three times shorter than subsystem 2. Again, the integration step-sizes of the subsystems, h_2 and h_1 match their macro ones. For subsystem 1, at intermediate communication points $t_k + h_1$ and $t_k + 2h_1$, the outputs from subsystem 2 are not available, and so the co-simulation manager has to provide an extrapolation based on previously known values, or another form of approximation.

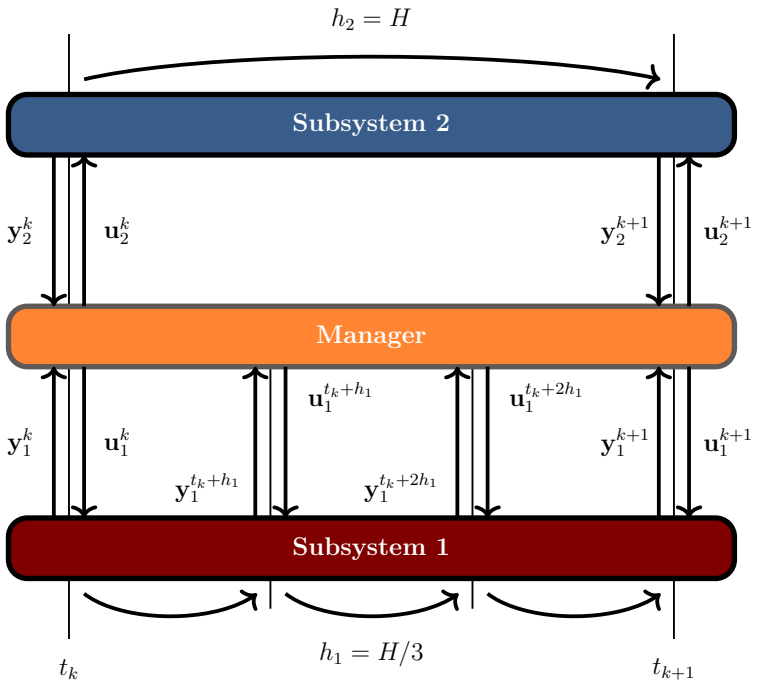


Figure 5.6: An explicit, multi-rate co-simulation scheme.

5.2.3 Selection of coupling variables

The selection of coupling variables is closely related to the way in which the overall application under study is split into subsystems, and this in turn influences considerably co-simulation results. The analyst may or may not have the capacity to decide how this division is carried out: for instance, when some components are supplied by external providers it may not be possible to modify the selection of coupling variables at all.

In general, regarding the nature of the coupling equations, subsystems can be coupled in two different ways. A possibility is using *algebraic constraints* to relate to each other the dynamics of the different subsystems. In the case of mechanical systems, this would be equivalent to imposing a rigid constraint between them, in the form $\varphi(\mathbf{x}_1, \mathbf{x}_2) = 0$, where $\mathbf{x}_1$ and $\mathbf{x}_2$ would be the subsystem states. The co-simulation of systems coupled via algebraic constraints often requires the use of particular solution methods, such as implicit schemes and the use of directional

derivatives [165–167]. Another possibility is replacing the algebraic constraints with controlled spring-damper systems to approximate the system dynamics. However, this approximation modifies the system energy and usually requires to reduce the macro step-size to obtain valid results [89].

In this thesis, the alternative solution, namely coupling the subsystems by means of *constitutive relations*, is discussed. In the case of mechanical systems, this means that a certain compliance is assumed at the connections between subsystems, which behave in a similar way to spring-damper systems. When constitutive relations are used, two scenarios are possible:

- *The product of input-output variables of the subsystems has power units*: e.g., force-displacement for mechanical systems or voltage-current for electric ones. This set of variables provides information about the power exchange at the interface between the subsystems and the co-simulation manager. Some monitoring and correction methods to keep the co-simulation accurate and stable can be based on this selection [109, 117, 118].

- *The product of input-output variables of the subsystems does not have power units*: e.g., displacement-displacement in mechanical systems. The indicators based on power exchanges can no longer be used, although others can be defined based on generalized energy approaches [119]. The use of this variable selection may be advantageous in some cases, in which it is associated with the definition of subsystems without direct feedthrough, which feature better convergence properties [168].

5.2.3.1 Direct feedthrough

The concept of direct feedthrough denotes an explicit dependence of a subsystem output from its inputs. The presence of direct feedthrough in a co-simulation scheme may lead to the existence of algebraic loops, which often motivates the necessity to perform the initialization stage in an iterative way [93]. During runtime, direct feedthrough is associated with a deterioration of the integration convergence and stability [168].

The presence of direct feedthrough can be identified assuming that the dynamics of a given subsystem can be expressed using a time-continuous state-space notation as

$$\begin{bmatrix} \dot{\mathbf{x}} \\ \mathbf{y} \end{bmatrix} = \begin{bmatrix} \mathbf{F} & \mathbf{G} \\ \mathbf{H} & \mathbf{N} \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ \mathbf{u} \end{bmatrix} \tag{5.2}$$

where $\mathbf{x}$ is the state, $\mathbf{F}$ the state matrix, $\mathbf{G}$ the input matrix, $\mathbf{H}$ the output matrix, and $\mathbf{N}$ the feedthrough matrix.

In a co-simulation setup, the communication between subsystems occurs in a discrete fashion. If a single-rate Jacobi scheme is considered, a subsystem will receive its inputs at time t_k. With these, it is able to directly evaluate its state derivatives $\dot{\mathbf{x}}$ and subsequently integrate its dynamics until time t_{k+1}. The subsystem outputs $\mathbf{y}$, on the other hand, must be evaluated and returned at time t_{k+1}. According to Eq. (5.2), the input at time t_{k+1} is necessary to evaluate the outputs; however, in a

Jacobi scheme like the one described in 5.1, outputs must be returned before inputs are received at time t_{k+1}, and so $\mathbf{u}^{k+1}$ cannot be used to evaluate $\mathbf{y}^{k+1}$.

If the feedthrough matrix is null, i.e., $\mathbf{N} = \mathbf{0}$, then it does not matter that $\mathbf{u}^{k+1}$ is not available, because $\mathbf{y}^{k+1}$ can be evaluated without this information as $\mathbf{y} = \mathbf{Hx}$. On the contrary, a subsystem with direct feedthrough will have $\mathbf{N} \neq \mathbf{0}$, and this requires the knowledge of $\mathbf{u}^{k+1}$ to evaluate the outputs at time t_{k+1}. This means that inputs $\mathbf{u}^{k+1}$ will have to be approximated in some way, e.g., via polynomial extrapolation, and this will introduce an additional error in the evaluation of the system dynamics, which will deteriorate the accuracy of the co-simulation results.

As mentioned, Fig. 5.1 illustrates an example of a two-subsystem co-simulation setup that follows an explicit single-rate Jacobi schedule. To illustrate the issue with direct feedthrough, it can be considered that subsystem 1 features it, while subsystem 2 does not. Moreover, both subsystems are assumed to use explicit integration formulas. Accordingly, subsystem 1, introduces an error in the evaluation of its outputs $\mathbf{y}_1^{k+1}$ before receiving updated inputs at stage 4. These errors are especially relevant in explicit schemes, in which it is not possible to retake the last integration step to improve convergence. On the other hand, the second subsystem, which does not feature direct feedthrough, does not require $\mathbf{u}_2^{k+1}$ to calculate its outputs, and so these will be accurately evaluated; the only error that will be propagated with them is the error of the numerical integration of the state from time t_k to t_{k+1}.

Nevertheless, in multi-rate schemes an additional problem appears when a subsystem requires updated inputs at intermediate points to compute an integration step as shown in Figure 5.6. At time $t_k + h_1$ subsystem 1 needs to evaluate its derivatives, although there is no available information from subsystem 2 until the next communication point. In fact, this problem exists whenever the integration step-sizes within the subsystems do not match the macro step-size used to communicate with the co-simulation manager. Input extrapolation techniques will be required to evaluate the state derivatives at intermediate times between communication points, except in the rather infrequent case in which $\mathbf{G} = \mathbf{0}$.

In a nutshell, in a single-rate co-simulation with explicit integration the subsystems do not incur in errors associated with input extrapolation unless they feature direct feedthrough. This is not the case in multi-rate co-simulation. These circumstances have to be taken into account for a correct evaluation of the energy errors introduced at the coupling interface, regardless of other errors, e.g., numerical errors in the integration process. This will be addressed in the Sections 5.5 – 5.6, aimed at discussing how to calculate the energy errors introduced by the co-simulation and how to eliminate this artificial modification of the system energy.

5.2.4 Input extrapolation

Co-simulation schemes frequently need to have recourse to extrapolation methods to evaluate subsystem inputs at some communication steps. The non-iterative Jacobi scheme in Figs. 5.1 and 5.6 is an example of this. At time t_{k+1} (step 4) each subsystem has to evaluate its outputs without having access to the values of its inputs at this time. An approximate value $\tilde{\mathbf{u}}^{k+1}$ is used, instead of the real one $\mathbf{u}^{k+1}$, which is not available yet. In the multi-rate scenario in Fig. 5.6, the co-simulation

manager can handle the input extrapolation at intermediate steps t_k+h_1 and t_k+2h_1. More generally, the subsystems may incorporate their own extrapolation routines.

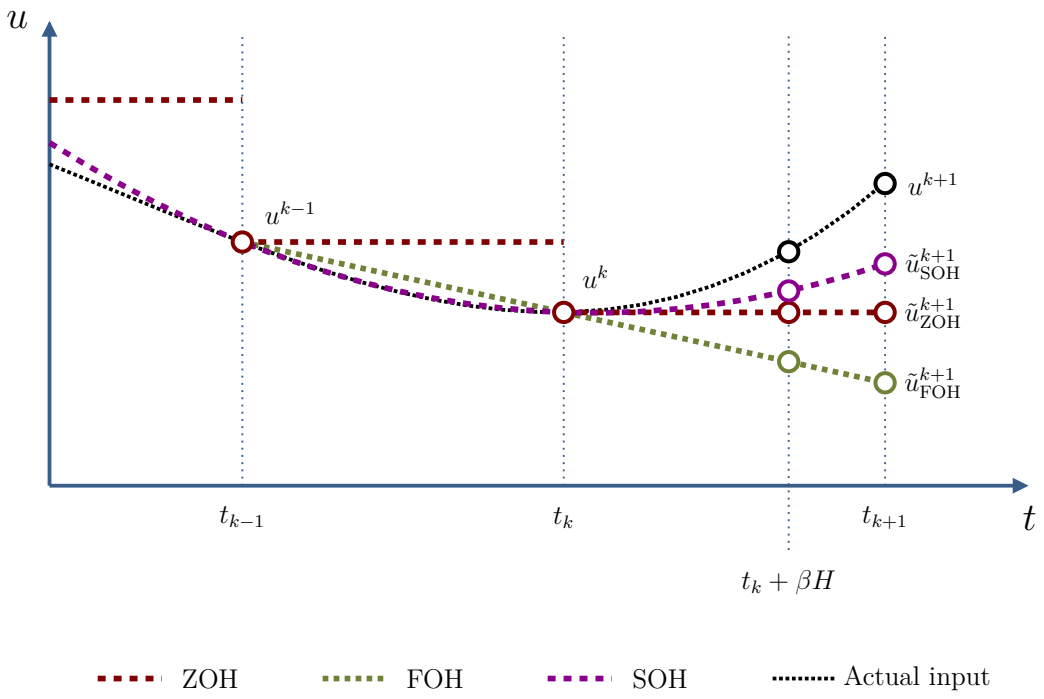


Figure 5.7: Effect of ZOH, FOH, and SOH extrapolation on the prediction of subsystem inputs between t_k and t_{k+1}, where $\beta \in [0, 1]$.

Diverse approximation techniques are commonly used in co-simulation, however, polynomial extrapolation is very popular for its simplicity and good results in a wide range of co-simulation setups [94, 95]. Figure 5.7 illustrates some examples of polynomial extrapolation, which are described below:

- *Zero-Order Hold:* Constant or zero-order polynomial extrapolation, also known as Zero-Order Hold (ZOH) is widely used due to its simplicity. In this approach, the input value is considered constant during the interval as

$$\tilde{u}_{ZOH}(t) = u^k \qquad t \in [t_k, t_{k+1}] \tag{5.3}$$

- *First-Order Hold:* Linear or first-order polynomial extrapolation, also known as First-Order Hold (FOH) has been often used in an attempt to make the integration process of the subsystems more stable, e.g., [169], in multi-rate environments. In a linear extrapolation the value of the input is defined as

$$\tilde{u}_{FOH}(t) = u^k + (t - t_k)\frac{u^k - u^{k-1}}{t_k - t_{k-1}} \qquad t \in [t_k, t_{k+1}] \tag{5.4}$$

- *Second-Order Hold:* Quadratic or second-order polynomial extrapolation, also known as Second-Order Hold (SOH) uses a second-order polynomial to evaluate the inputs as

$$\tilde{u}_{SOH}(t) = at^2 + bt + c \qquad t \in [t_k, t_{k+1}] \tag{5.5}$$

where the coefficients a, b and c are obtained after solving the following linear system of equations using input values at time steps t_k, t_{k-1} and t_{k-2} as

$$\begin{bmatrix} t_k^2 & t_k & 1 \\ t_{k-1}^2 & t_{k-1} & 1 \\ t_{k-2}^2 & t_{k-2} & 1 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \end{bmatrix} = \begin{bmatrix} u^k \\ u^{k-1} \\ u^{k-2} \end{bmatrix} \tag{5.6}$$

Second-order and higher-order extrapolations are expected to increase the coupling bandwidth [114] and accuracy, although they are also more sensitive to discontinuities in subsystem dynamics and they are not so commonly used as ZOH and FOH.

5.2.5 The FMI standard

Interface definitions and implementations are not, strictly speaking, co-simulation configuration options. They are, however, a necessary building block in co-simulation environments, especially for industrial applications. Interface definitions make it possible to integrate together components developed by different partners, minimizing the effort to keep them compatible. This may lead to trade-offs, as interface definitions must be robust and relatively easy to adopt, or they will not be useful as standards; at the same time, some coupling schemes or correction methods may not be supported by standardized interfaces. In this sense, the selection of a particular coupling interface influences the functionality of co-simulation and, accordingly, has an impact on its results.

The FMI is a standard conceived to address the necessity of a common, clearly defined protocol for co-simulation setups. It establishes an interface specification via XML[1] files and a series of standard function prototypes written in C to perform communication functions. Subsystems in a FMI co-simulation architecture are stored as Functional Mock-up Units (FMUs), zipped files that contain the subsystem description either as binaries or source code, and whose interface to the co-simulation environment is compatible with the standard. The development of FMI was started by Daimler AG to ease the exchange of simulation models between suppliers and manufacturers, and is currently fostered and maintained by the Modelica Association [92,170]. FMI version 1.0 was released in 2010, version 2.0 in 2014, version 2.01 in 2019; version 3.0 (beta) was released in 2021.

Figure 5.8 shows a conceptual scheme of a FMU, which contains a model of the subsystem, whose co-simulation interface is described in an XML file, implemented as DLL/SO[2] binaries. The FMU wrapper includes its own solver, which has access to the internal states and local variables of the subsystem. This solver is responsible for integrating the states between communication points, using the mathematical (model, local variables and parameters) and computational (DLL/SO binaries and XML) description of the subsystem, as well as the information provided as inputs by

[1] eXtensible Markup Language is a markup language that defines a set of rules for encoding documents in a format that is both human-readable and machine-readable.

[2] DLL (Dynamic-link library) and SO (dynamically linked Shared Object library) are the extensions for binary libraries on Windows and Linux platforms respectively.

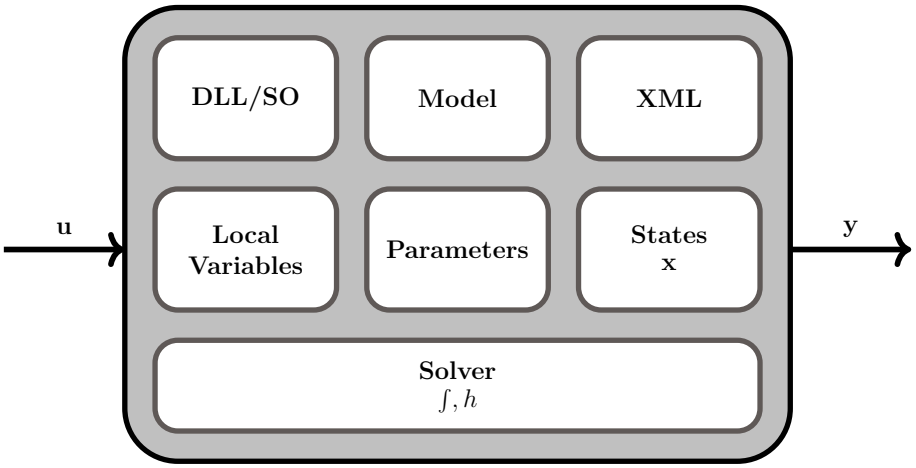


Figure 5.8: Scheme of a FMU wrapper.

the co-simulation manager. When the subsystem reaches the next communication point, it sends its outputs to the co-simulation manager and starts this process again.

5.3 Explicit co-simulation for real-time applications

The term RT co-simulation is used to allude to those applications in which all computations and information exchanges between two consecutive communications points must be conducted in a RWWCT that spans a period shorter than the length of the communication step H. The simulation outputs not only have to be correct: they also have to be delivered at the right time [4]. Fast code execution is essential for RT co-simulation, but so is predictability with regard to the duration of the computation intervals.

RT co-simulation allows one to simulate environments in which real and virtual elements are mixed in physical-virtual applications, such as test benches or haptic simulators, for instance. A haptic simulator can be regarded as a physical-virtual application under a RT co-simulation setup, in which a user specifies the desired motion and the virtual system delivers the simulated force, as the real component would do. Other uses of RT co-simulation include HiL simulators, like ADAS with a virtual model of the vehicle on which they are mounted, and SitL test benches, in which physical components are tested in a virtual environment. The particular case of physical-virtual test benches is gaining popularity in the automotive industry to test e-powertrain parts under RT co-simulation setups. This application will be described in further detail in Chapter 6.

Simulators that are coupled to real components are subjected to RT execution constraints, because these systems cannot be paused or slowed down if the virtual components are not able to complete the integration and information exchange in the interval given by the macro step-size H. In order to meet this requirement, RT co-simulation setups usually incorporate the following features:

- *Constant communication macro step-sizes* are commonly used in the subsystems and the co-simulation manager, due to physical components being often sampled at constant rates, which cannot be modified during runtime.
- *Matching time grids* are almost exclusively used in the majority of practical applications.
- *Explicit co-simulation schemes* are used preferentially in RT co-simulation setups. Physical components cannot perform rollbacks and retake their integration steps, i.e., the time in a real subsystem cannot move backwards. For the sake of simplicity and in order to enable parallel execution, Jacobi schemes are the preferred option in most cases, although in some setups the Gauss-Seidel approach is also used.

5.4 Benchmark problems

In this Section four benchmarks are presented and described, in which the configuration selection described in Section 5.2 can be evaluated. These systems will also be used to validate the monitoring and correction algorithms developed in Sections 5.5 – 5.6.

5.4.1 Linear oscillator

The first problem used as benchmark is a Linear Oscillator (LO), a two-degree-of-freedom mechanical system composed of two masses (m_1 and m_2) connected to each other and to the ground by means of dampers (c_1, c_2 and c_c) and springs (k_1, k_2, and k_c) with constant coefficients, as shown in Fig. 5.9.

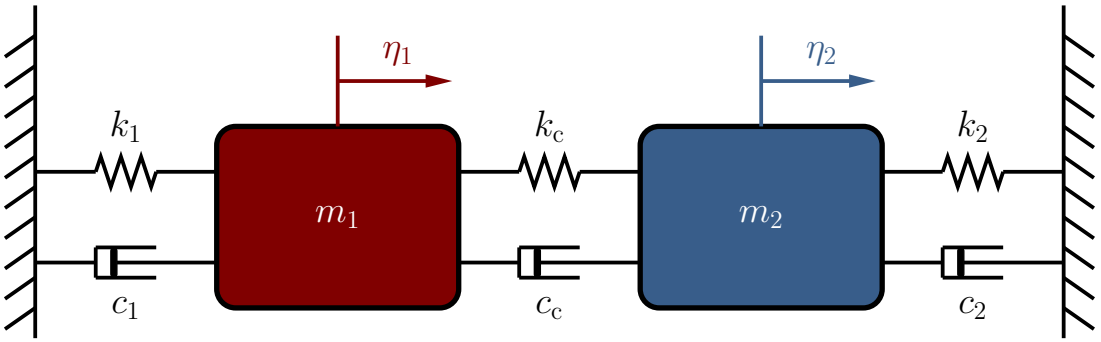


Figure 5.9: A two-degree-of-freedom linear oscillator.

The state $\mathbf{x}$ of this system contains the displacements with respect to the equilibrium configuration η and the velocities $\dot{\eta}$ of both masses,

$$\mathbf{x} = \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \end{bmatrix} = \begin{bmatrix} \eta_1 & \dot{\eta}_1 & \eta_2 & \dot{\eta}_2 \end{bmatrix}^{\mathrm{T}} \tag{5.7}$$

LOs have been widely used in the literature, e.g., [7, 109], as benchmark problems due to the availability of an analytical solution for their motion. In the absence of

externally applied forces, the dynamics equations of the LO can be written as

$$\begin{aligned} m_1\ddot{\eta}_1 + (c_1 + c_c)\dot{\eta}_1 - c_c\dot{\eta}_2 + (k_1 + k_c)\eta_1 - k_c\eta_2 = 0 \\ m_2\ddot{\eta}_2 + (c_2 + c_c)\dot{\eta}_2 - c_c\dot{\eta}_1 + (k_2 + k_c)\eta_2 - k_c\eta_1 = 0 \end{aligned} \tag{5.8}$$

This system of differential equations can be rearranged in matrix form as

$$\dot{\mathbf{x}} = \boldsymbol{\vartheta}\mathbf{x} \tag{5.9}$$

where $\boldsymbol{\vartheta}$ is a square matrix defined as

$$\boldsymbol{\vartheta} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -\dfrac{k_1 + k_c}{m_1} & -\dfrac{c_1 + c_c}{m_1} & \dfrac{k_c}{m_1} & \dfrac{c_c}{m_1} \\ 0 & 0 & 0 & 1 \\ \dfrac{k_c}{m_2} & \dfrac{c_c}{m_2} & -\dfrac{k_2 + k_c}{m_2} & -\dfrac{c_2 + c_c}{m_2} \end{bmatrix} \tag{5.10}$$

If the coefficients in Eqs. (5.8) are constant, the solution of Eq. (5.9) is

$$\mathbf{x}(t) = \mathbf{x}_0 e^{\boldsymbol{\vartheta}(t - t_0)} \tag{5.11}$$

where t_0 is the starting time and $\mathbf{x}_0$ the initial state.

5.4.1.1 Physical properties

Two different values of the system parameters were selected to represent damped and undamped versions of the LO. Table 5.1 summarizes the system parameters for both cases.

Table 5.1: Combinations of system parameters used in LO benchmarks.

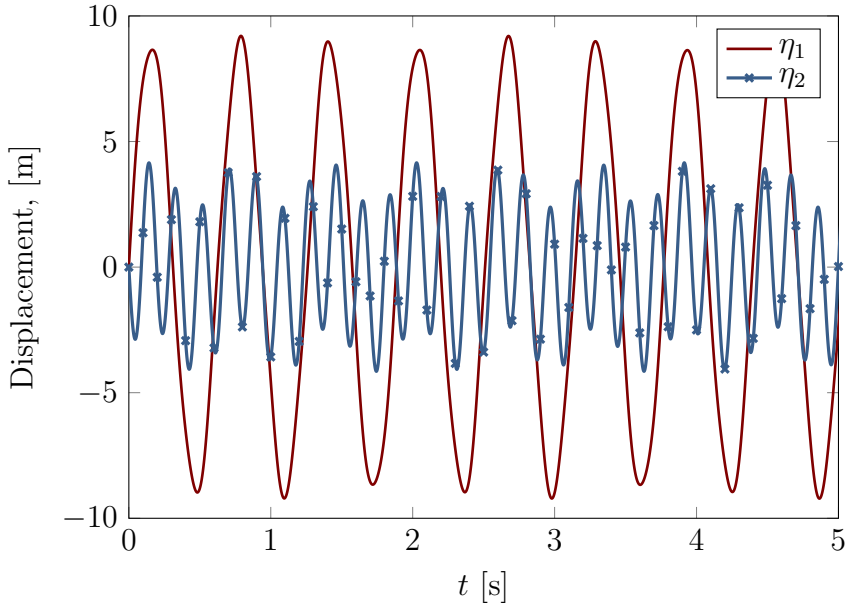
Case	m_1 [kg]	m_2 [kg]	k_1 [N/m]	k_c [N/m]	k_2 [N/m]	c_1 [Ns/m]	c_c [Ns/m]	c_2 [Ns/m]
1	1	1	10	100	1000	0	0	0
2	1	1	10	100	1000	0.1	0.1	0.1

The initial system displacements were set to $\eta_1(0) = \eta_2(0) = 0$ m; the spring forces are zero in this configuration. The initial system velocities were $\dot{\eta}_1(0) = 100$ m/s and $\dot{\eta}_2(0) = -100$ m/s. These values were used in other references in the literature, e.g., [109, 167, 171].

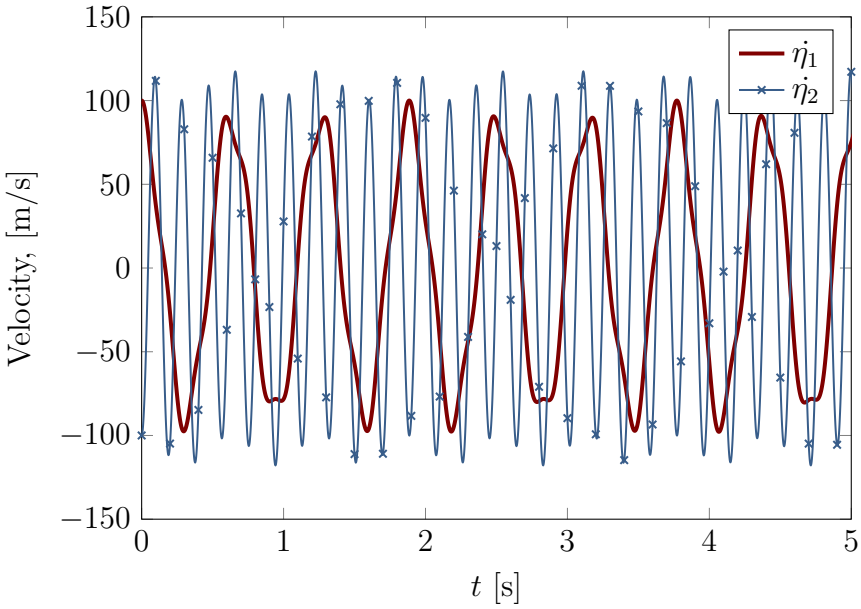
5.4.1.2 Reference solutions

Reference solutions are a key component in benchmark problems [44], as they are required to verify the results provided by a given implementation of a problem solution method. The parameters in Table 5.1 have constant values, and so they

describe LOs whose state (displacements and velocities) can be expressed in the compact analytical form given by Eq. (5.11). Figures 5.10 and 5.11 show the analytical solutions for displacements and velocities in both scenarios defined in Section 5.4.1.1.



(a) Displacements η_1 and η_2.

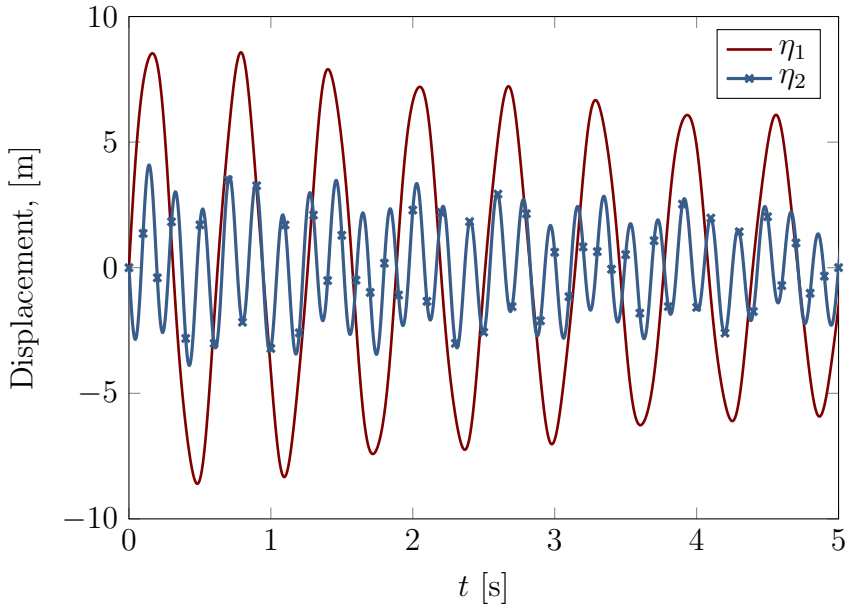


(b) Velocities $\dot{\eta}_1$ and $\dot{\eta}_2$.

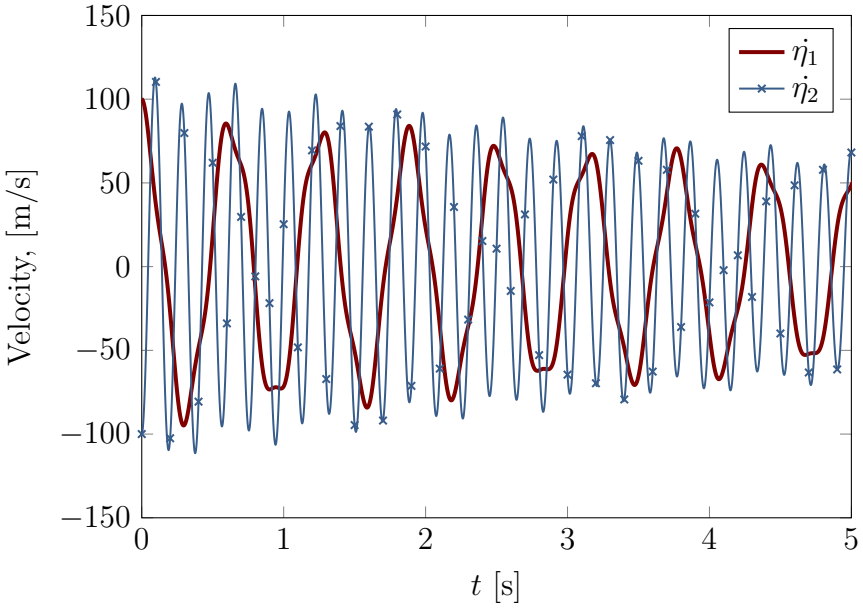
Figure 5.10: LO: Mass displacements and velocities, reference solution for case 1.

5.4.1.3 Force-displacement co-simulation

The division of the LO into subsystems can be performed in different ways. The first one considered here consists in the establishment of a force-displacement



(a) Displacements η_1 and η_2.



(b) Velocities $\dot{\eta}_1$ and $\dot{\eta}_2$.

Figure 5.11: LO: Mass displacements and velocities, reference solution for case 2.

scheme, as shown in Fig. 5.12. The first subsystem $\mathcal{M}_1$ integrates the dynamics of the first mass and passes the coupling force f as output to the second subsystem $\mathcal{M}_2$, which handles the motion of the second mass and, in turn, delivers its displacement and velocity as outputs, denoted here as $\mathbf{x}_2$.

The integration step-sizes are defined as h_1 and h_2 for the first and second subsystems, respectively. The macro step-size used to communicate the subsystems and the co-simulation manager is denoted by H. This nomenclature will be used in all the coupling schemes in which a LO is involved.

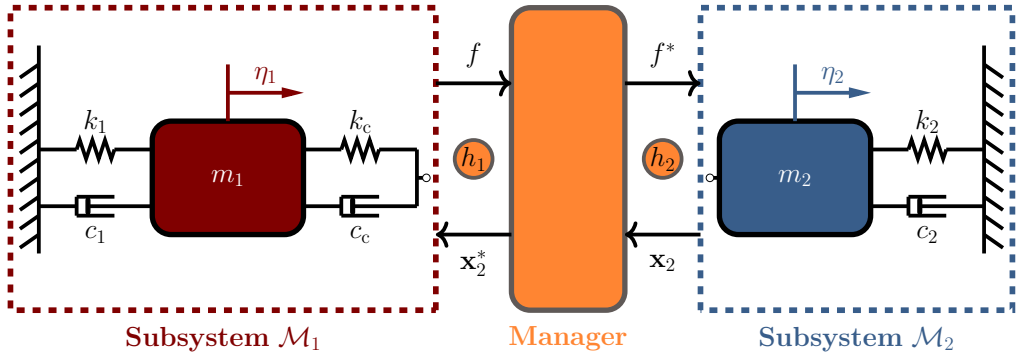


Figure 5.12: The linear oscillator arranged following a force-displacement coupling scheme.

In this coupling scheme, subsystem $\mathcal{M}_1$ is subjected to direct feedthrough $\mathbf{N}_{\mathcal{M}_1} \neq \mathbf{0}$, i.e., the evaluation of its output requires the availability of the inputs at any given instant in time. The inputs $\mathbf{u}$ and outputs $\mathbf{y}$ for each subsystem are given by

$$\begin{aligned} \mathbf{u}_{\mathcal{M}_1} &= \begin{bmatrix} \eta_2^* \\ \dot{\eta}_2^* \end{bmatrix} & \mathbf{u}_{\mathcal{M}_2} &= \begin{bmatrix} f^* \end{bmatrix} \\ \mathbf{y}_{\mathcal{M}_1} &= \begin{bmatrix} f \end{bmatrix} & \mathbf{y}_{\mathcal{M}_2} &= \begin{bmatrix} \eta_2 \\ \dot{\eta}_2 \end{bmatrix} \end{aligned} \tag{5.12}$$

where superscript $()^*$ points out the fact that the input of a subsystem might not necessarily be equal to the output of the other one, because the co-simulation manager may perform modifications in their values, e.g., extrapolation. Thus, the dynamics of subsystem $\mathcal{M}_1$ can be expressed as

$$m_1\ddot{\eta}_1 + (c_1 + c_c)\dot{\eta}_1 - c_c\dot{\eta}_2^* + (k_1 + k_c)\eta_1 - k_c\eta_2^* = 0 \tag{5.13}$$

where the coupling force f is evaluated in subsystem $\mathcal{M}_1$ as

$$f = c_c(\dot{\eta}_1 - \dot{\eta}_2^*) + k_c(\eta_1 - \eta_2^*) \tag{5.14}$$

The dynamics of subsystem $\mathcal{M}_2$, in turn, can be written as

$$m_2\ddot{\eta}_2 + c_2\dot{\eta}_2 + k_2\eta_2 = f^* \tag{5.15}$$

As mentioned, in principle, $f \neq f^*$, $\eta_2 \neq \eta_2^*$ and $\dot{\eta}_2 \neq \dot{\eta}_2^*$, because input extrapolation or some other kind of input processing can be performed by the co-simulation manager.

5.4.1.4 Displacement-displacement co-simulation

A second option to divide a LO into two subsystems is following a displacement-displacement scheme as shown in Fig. 5.13. In this configuration, both subsystems need access to the information about the stiffness and damping of the coupling, because both evaluate internally the coupling force. The first and second subsystems integrate their own dynamics receiving as input the displacement and velocity

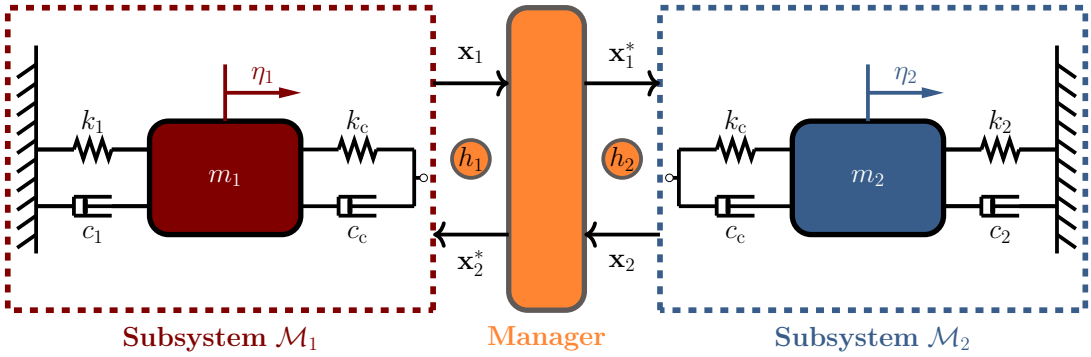


Figure 5.13: The linear oscillator arranged following a displacement-displacement coupling scheme.

from the other subsystem, and delivering their own displacements and velocities as outputs, which are denoted as $\mathbf{x}_1$ and $\mathbf{x}_2$, respectively.

In this coupling scheme, no subsystem is subjected to direct feedthrough $\mathbf{N}_{\mathcal{M}_1} = \mathbf{N}_{\mathcal{M}_2} = \mathbf{0}$. Thus, the vectors of inputs $\mathbf{u}$ and outputs $\mathbf{y}$ for each subsystem are given by

$$\begin{aligned} \mathbf{u}_{\mathcal{M}_1} &= \begin{bmatrix} \eta_2^* \\ \dot{\eta}_2^* \end{bmatrix} & \mathbf{u}_{\mathcal{M}_2} &= \begin{bmatrix} \eta_1^* \\ \dot{\eta}_1^* \end{bmatrix} \\ \mathbf{y}_{\mathcal{M}_1} &= \begin{bmatrix} \eta_1 \\ \dot{\eta}_1 \end{bmatrix} & \mathbf{y}_{\mathcal{M}_2} &= \begin{bmatrix} \eta_2 \\ \dot{\eta}_2 \end{bmatrix} \end{aligned} \tag{5.16}$$

The dynamics equations for this configuration are very similar to the monolithic ones in Eq. (5.8). They can be expressed as

$$m_1\ddot{\eta}_1 + (c_1 + c_c)\dot{\eta}_1 - c_c\dot{\eta}_2^* + (k_1 + k_c)\eta_1 - k_c\eta_2^* = 0 \tag{5.17}$$

$$m_2\ddot{\eta}_2 + (c_2 + c_c)\dot{\eta}_2 - c_c\dot{\eta}_1^* + (k_2 + k_c)\eta_2 - k_c\eta_1^* = 0 \tag{5.18}$$

where, again, in general $\eta_1 \neq \eta_1^*$, $\eta_2 \neq \eta_2^*$, $\dot{\eta}_1 \neq \dot{\eta}_1^*$ and $\dot{\eta}_2 \neq \dot{\eta}_2^*$.

5.4.1.5 Force-force co-simulation

The last option considered here to separate a LO into two subsystems is following a force-force scheme as shown in Fig. 5.14.

In this configuration, the subsystems do not need access to the information about the coupling properties, which is handled directly by the manager. Both subsystems integrate their dynamics receiving as input the coupling force f, and delivering their displacements and velocities as outputs, namely $\mathbf{x}_1$ and $\mathbf{x}_2$, respectively. Thus, the arrays of inputs $\mathbf{u}$ and outputs $\mathbf{y}$ for each subsystem are given by

$$\begin{aligned} \mathbf{u}_{\mathcal{M}_1} &= \begin{bmatrix} f \end{bmatrix} & \mathbf{u}_{\mathcal{M}_2} &= \begin{bmatrix} f \end{bmatrix} \\ \mathbf{y}_{\mathcal{M}_1} &= \begin{bmatrix} \eta_1 \\ \dot{\eta}_1 \end{bmatrix} & \mathbf{y}_{\mathcal{M}_2} &= \begin{bmatrix} \eta_2 \\ \dot{\eta}_2 \end{bmatrix} \end{aligned} \tag{5.19}$$

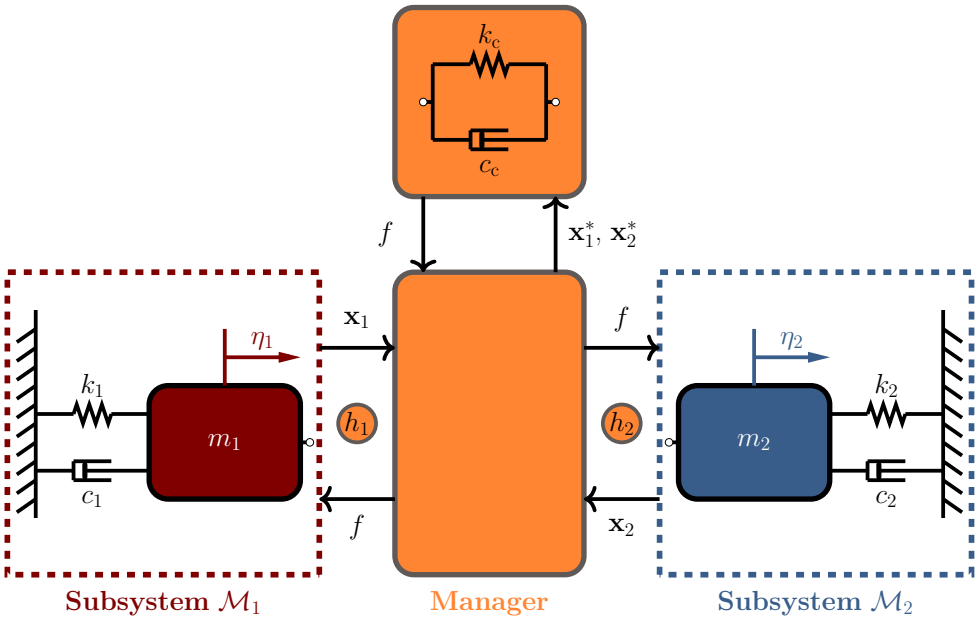


Figure 5.14: The linear oscillator arranged following a force-force coupling scheme.

In this coupling scheme, the dynamics equations of the subsystems are expressed as

$$\begin{aligned} m_1\ddot{\eta}_1 + c_1\dot{\eta}_1 + k_1\eta_1 &= -f \\ m_2\ddot{\eta}_2 + c_2\dot{\eta}_2 + k_2\eta_2 &= f \end{aligned} \tag{5.20}$$

where the coupling force f is evaluated by the co-simulation manager as

$$f = c_c(\dot{\eta}_1^* - \dot{\eta}_2^*) + k_c(\eta_1^* - \eta_2^*) \tag{5.21}$$

where, in general $\eta_1 \neq \eta_1^*$, $\eta_2 \neq \eta_2^*$, $\dot{\eta}_1 \neq \dot{\eta}_1^*$ and $\dot{\eta}_2 \neq \dot{\eta}_2^*$.

5.4.2 Nonlinear oscillator

The second proposed benchmark is a modification of the LO described in Section 5.4.1. In this example, the three linear spring-dampers have been replaced with their nonlinear counterparts and tested in the following scenarios:

- *Case 1:* nonlinear springs are implemented in this scenario. The spring forces follow a square root function of the spring elongation as

$$f = -k\ \mathrm{sgn}(\eta)\ \sqrt{|\eta|} \tag{5.22}$$

  where sgn() is the sign function, η is the elongation of the spring, and k is the linear stiffness coefficient. No damping was considered in this case.

- *Nonlinear damper:* the damping force exerted follows a quadratic function of the variation of damper elongation as

$$f = -c\ \mathrm{sgn}(\dot{\eta})\ |\dot{\eta}|^2 \tag{5.23}$$

  where $\dot{\eta}$ is the variation of the spring elongation, and c is the linear damping coefficient. Spring forces are considered linear in this scenario.

5.4.2.1 Physical properties

Two different values of the system parameters were selected to represent an example of a damped and an undamped Nonlinear Oscillator (NLO). Table 5.2 summarizes the system parameters in both cases.

Table 5.2: Combinations of system parameters used in the NLO benchmark.

Case	m_1 [kg]	m_2 [kg]	k_1 [N/m]	k_c [N/m]	k_2 [N/m]	c_1 [Ns/m]	c_c [Ns/m]	c_2 [Ns/m]
1	1	1	10	100	1000	0	0	0
2	1	1	10	100	1000	0.001	0.001	0.001

The initial system displacements were set to $\eta_1(0) = \eta_2(0) = 0$ m; the spring forces are zero in this configuration. The initial system velocities are $\dot{\eta}_1(0) = 100$ m/s and $\dot{\eta}_2(0) = -100$ m/s.

5.4.2.2 Reference solutions

In general, NLOs have no analytical solution to compare with the results obtained with each simulation setup. A numerical solution, however, can be obtained *at convergence*, i.e. running several monolithic simulations with decreasing step-sizes until their solutions converge within a certain range. The process stops when the difference between two consecutive solutions are below an user-defined tolerance ε. The simulation with the smallest step-size is considered as the reference solution of the system behaviour.

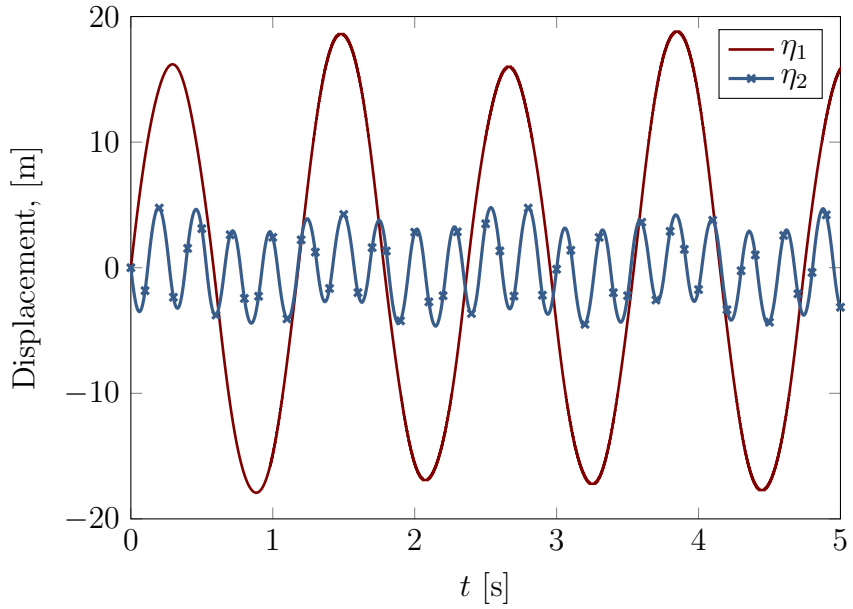
Figures 5.15 and 5.16 represent the solutions obtained *at convergence* for the displacements and velocities in both scenarios defined Section 5.4.2.1.

5.4.2.3 Coupling variable selection

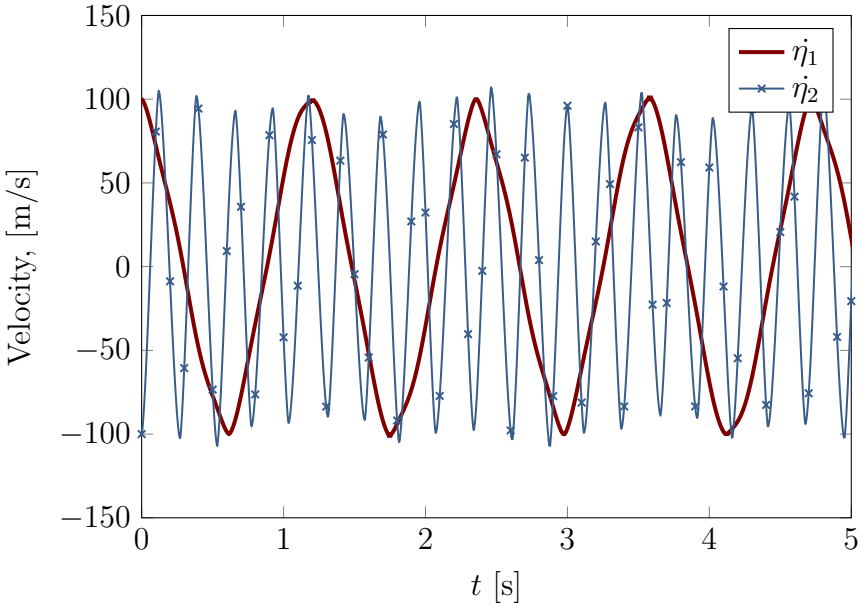
Regarding the division of the NLO into subsystems and the corresponding selection of coupling variables, the same divisions used for the LO, discussed in Sections 5.4.1.3 – 5.4.1.5.

5.4.3 Two pendula connected by a spring-damper at their tips

The third system considered as benchmark is a two-degree-of-freedom mechanism composed of two pendula, pinned to the ground at points A and B and connected to each other by a spring-damper at their opposite tips. Two masses (m_1 and m_2) are located at the free ends of the pendulum rods. The spring-damper, whose tips are also connected to these points, features a stiffness coefficient k and a damping coefficient c. Figure 5.17 shows a scheme of this system.



(a) Displacements η_1 and η_2.



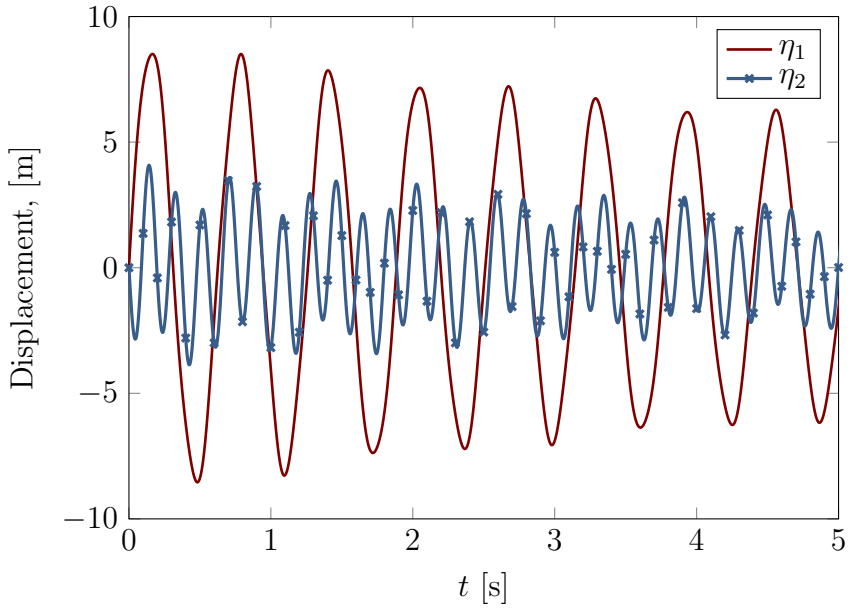
(b) Velocities $\dot{\eta}_1$ and $\dot{\eta}_2$.

Figure 5.15: NLO: Mass displacements and velocities, reference solution for case 1.

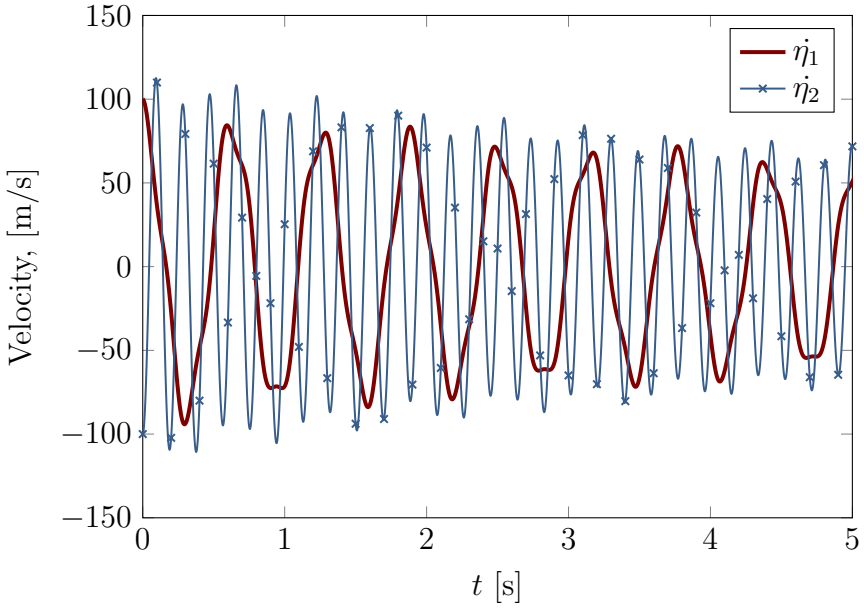
The following set of generalized coordinates $\mathbf{x}$, which contains the global positions (x, y) of both rod tips, is used to describe the system motion

$$\mathbf{x} = \begin{bmatrix} x_1 & y_1 & x_2 & y_2 \end{bmatrix}^{\mathrm{T}} \tag{5.24}$$

With this coordinate selection, a correct consideration of the system dynamics requires the introduction of two kinematic constraints that enforce that the length of



(a) Displacements η_1 and η_2.



(b) Velocities $\dot{\eta}_1$ and $\dot{\eta}_2$.

Figure 5.16: NLO: Mass displacements and velocities, reference solution for case 2.

each pendulum remains constant during motion,

$$\mathbf{\Phi} = \begin{bmatrix} (x_1 - x_{\mathsf{A}})^2 + (y_1 - y_{\mathsf{A}})^2 - l_1^2 \\ (x_2 - x_{\mathsf{B}})^2 + (y_2 - y_{\mathsf{B}})^2 - l_2^2 \end{bmatrix} \tag{5.25}$$

where l_1 and l_2 are the lengths of the rods, and x_{A}, y_{A}, x_{B}, and y_{B} denote the global coordinates of fixed points A and B, which leads to the formulation of the system dynamics in the form of a series of DAEs. Following a Lagrangian approach, [172],

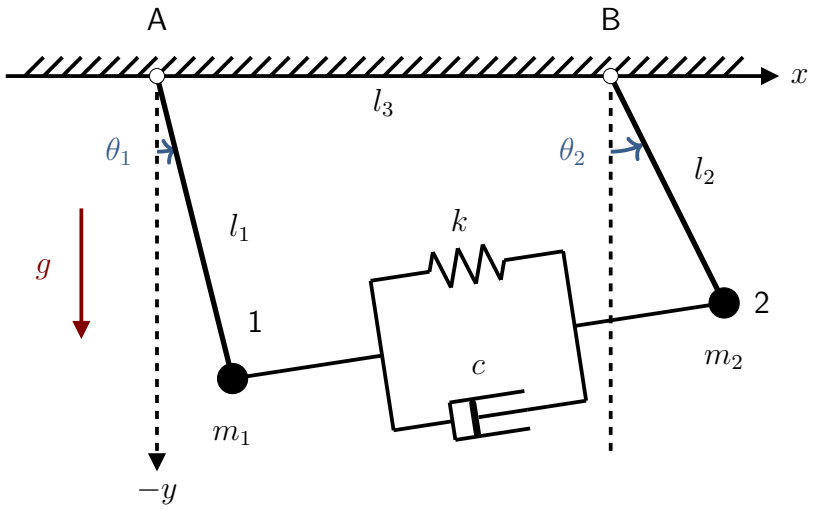


Figure 5.17: A two-degree-of-freedom two pendula connected by a spring-damper at their tips.

the equations of motion of this system can be expressed as

$$\mathbf{M}\ddot{\mathbf{x}} + \mathbf{\Phi}_{\mathbf{x}}^{\mathrm{T}}\boldsymbol{\sigma} = \mathbf{Q}$$
$$\mathbf{\Phi} = \mathbf{0} \tag{5.26}$$

where $\boldsymbol{\sigma}$ is a set of two Lagrange multipliers, related to the forces that impose the constant distance constraints in Eq. (5.25). The mass matrix $\mathbf{M}$ is directly obtained as

$$\mathbf{M} = \begin{bmatrix} m_1 & 0 & 0 & 0 \\ 0 & m_1 & 0 & 0 \\ 0 & 0 & m_2 & 0 \\ 0 & 0 & 0 & m_2 \end{bmatrix} \tag{5.27}$$

and the term of generalized forces $\mathbf{Q}$ is of the form

$$\mathbf{Q} = \begin{bmatrix} k(x_2 - x_1) + c(\dot{x}_2 - \dot{x}_1) \\ k(y_2 - y_1) + c(\dot{y}_2 - \dot{y}_1) - m_1 g \\ k(x_1 - x_2) + c(\dot{x}_1 - \dot{x}_2) \\ k(y_1 - y_2) + c(\dot{y}_1 - \dot{y}_2) - m_2 g \end{bmatrix} \tag{5.28}$$

where g is the acceleration of gravity. Equations (5.26) can be solved by means of an Augmented Lagrangian approach and integrated by the Forward-Euler method.

Table 5.3 summarizes the physical parameters used to describe both pendula.

The initial system angles were set to $\theta_1(0) = \pi/4$ rad and $\theta_2(0) = \pi/18$ rad. The initial system velocities were $\dot{\theta}_1(0) = \dot{\theta}_2(0) = 0$ rad/s.

5.4.3.1 Reference solutions

The reference solutions for this benchmark were also obtained *at convergence* using a monolithic simulation of the system motion. The horizontal and vertical positions and velocities of both tips, namely 1 and 2, are presented in Fig. 5.18.

Table 5.3: Combinations of system parameters used in the two-pendulum benchmark.

m_1 [kg]	m_2 [kg]	l_1 [m]	l_2 [m]	k [N/m]	c [Ns/m]	A [m]	B [m]
1	2	2.5	3.5	6	0	$(0, 0)$	$(7, 0)$

5.4.3.2 Coupling schemes

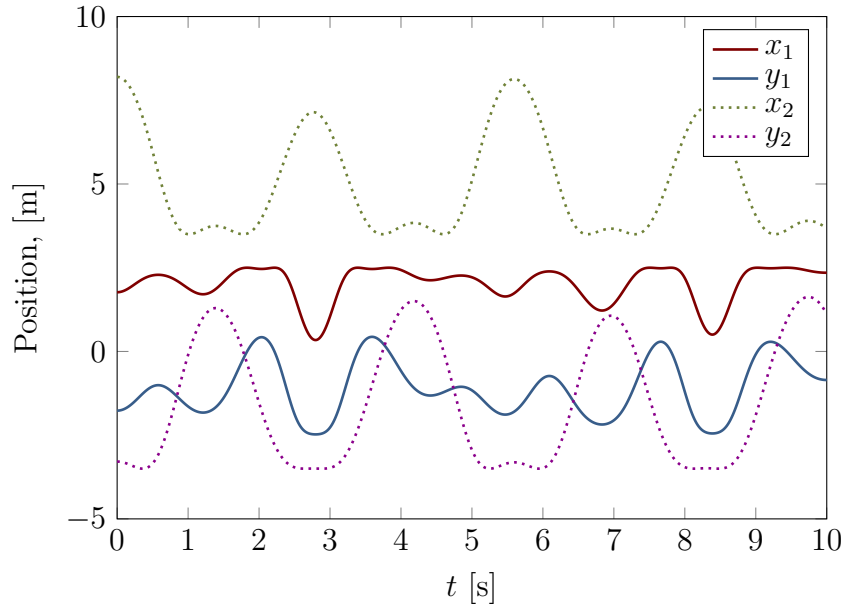
This mechanical system can be arranged to follow the same coupling schemes than the LO and NLO. Subsystems are also named as $\mathcal{M}_1$ and $\mathcal{M}_2$. These coupling strategies have already been described in Sections 5.4.1.3, 5.4.1.4 and 5.4.1.5. In particular:

- *Force-displacement:* Only one subsystem contains information about the coupling. This first subsystem passes the force exerted by the spring-damper to the second subsystem. The second subsystem, in turn, delivers the position and velocity of its rod tip to the first subystem. In this configuration, the second subsystem does not feature direct feedthrough.
- *Displacement-displacement:* Both subsystems need to have access to the coupling information. The two subsystems send their position and velocity to each other. In this scheme, no subsystem features direct feedthrough.
- *Force-force:* The subsystems do not need information about the physical properties of the coupling. The co-simulation manager implements an function to evaluate the coupling force, which will be sent to both subsystems as input. Meanwhile, subsystems send their position and velocity to the manager. In this scheme, no subsystem features direct feedthrough.

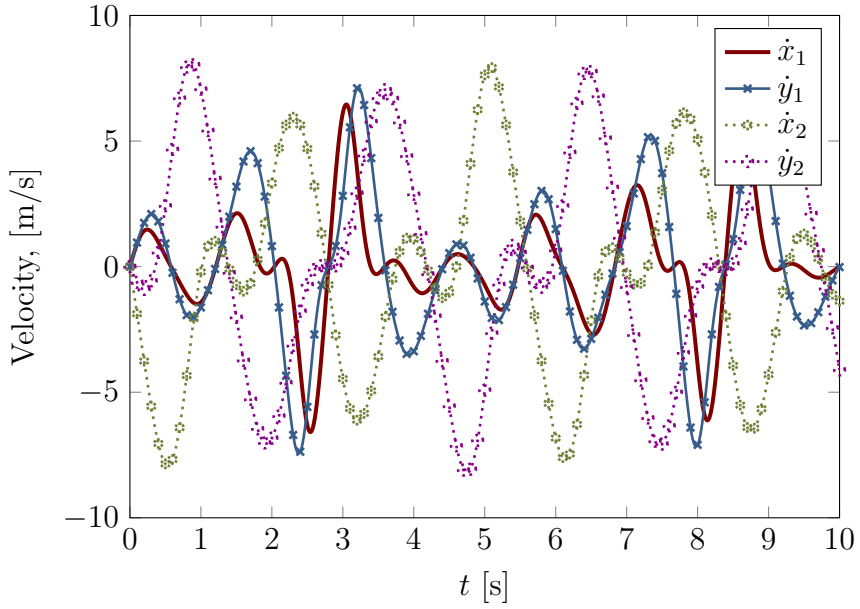
5.4.4 Hydraulic crane

The last system used as benchmark is a Hydraulic Crane (HC) that consists of a two-degree-of-freedom multiphysics system composed of a two-link robotic arm and a hydraulic actuator, as shown in Fig. 5.19. A similar system was initially described in [87], and modified versions were later used as benchmarks in [96] and [110].

The HC example is a nonlinear and multiphysics system that can be considered to be composed of a mechanical subsystem $\mathcal{M}$ and a hydraulic one $\mathcal{H}$. The mechanical subsystem $\mathcal{M}$ consists of two links of lengths l_1 and l_2, respectively. Link 1 has a distributed mass m_1, whereas link 2 is considered to be massless. Two additional masses m_2 and m_3 are located at points Q and R, respectively. The assembly moves under gravity effects and is actuated with a hydraulic actuator, which is also part of the hydraulic subsystem $\mathcal{H}$. This actuator consists in a piston in a cylinder with two chambers that exerts a force onto the mechanical subsystem. The hydraulic subsystem also includes a pump and a tank, and a set of valves that regulate the pressure distribution in the different components. The magnitude of the hydraulic



(a) Positions.



(b) Velocities.

Figure 5.18: Two pendula: Time-history of the position and velocities of the points 1 and 2.

force is evaluated using the pressure difference between both chambers, namely $p_2 - p_1$, as

$$f_{\mathrm{h}} = a_p \left(p_2 - p_1\right) - c\dot{s}_c \tag{5.29}$$

where a_p is the piston surface area, and c is a damping coefficient that represents its internal dissipation. A detailed description of the example can be found in [96].

The system parameters that correspond to the HC described in [96] are summarized in Table 5.4.

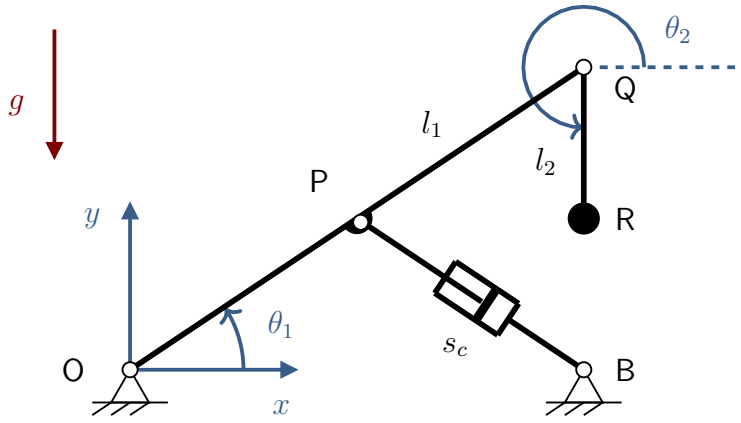


Figure 5.19: Planar model of a manipulator with a single hydraulic actuator.

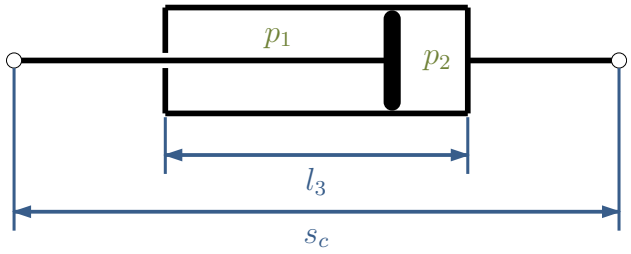


Figure 5.20: Schematic of the hydraulic actuator.

5.4.4.1 Manoeuvre

The overall system response is controlled by a valve that modifies the pressure in both chambers of the cylinder by means of a control parameter κ, the valve spool displacement, in the interval $[0, 1]$, where 0 and 1 stand for a completely closed and a completely open valve, respectively.

In this benchmark, the proposed manoeuvre commands the valve displacement to follow a sinusoidal actuation law

$$\kappa = \kappa_0 \left(1 - A \sin\left(2\pi\omega t\right)\right) \ , \qquad \text{where} \quad A = \begin{cases} 0.1t \, , & 0 \text{ s} \leq t < 1 \text{ s} \\ 0.1 \, , & 1 \text{ s} \leq t < 8 \text{ s} \\ 0.1 \left(9 - t\right) \, , & 8 \text{ s} \leq t < 9 \text{ s} \\ 0 \, , & t \geq 9 \text{ s} \end{cases} \tag{5.30}$$

where $\omega = 2$ rad/s and A is the amplitude of the oscillation, which varies with time. Initially, the spool displacement is set to the value $\kappa_0 \approx 0.45435$ that results in the static equilibrium of the system. Figure 5.21 represents the time history of κ during this manoeuvre, following the sinusoidal expression in Eq. (5.30).

5.4.4.2 Reference solution

The HC as described in this Section does not have an analytical solution that can be used as reference to compare the results obtained during the tests. A numerical solution, however, was obtained *at convergence* running several monolithic simulations with the formulation introduced in [87].

Table 5.4: System parameters of the single-actuated model

Length of link 1	l_1	1.0 m
Length of link 2	l_2	0.5 m
Mass of link 1	m_1	200 kg
Point mass at Q	m_2	250 kg
Point mass at R	m_3	100 kg
Coordinates of fixed point B	$(x_\mathsf{B}, y_\mathsf{B})$	$\left(\sqrt{3}/2, 0\right)$ m
Initial angle, link 1	$(\theta_1)_0$	$\pi/6$ rad
Initial angle, link 2	$(\theta_2)_0$	$3\pi/2$ rad
Gravity	g	-9.81 m/s^2
Friction coefficient	c	10^5 Ns/m
Piston area	a_p	$65 \cdot 10^{-4}$ m^2
Cylinder length	l_3	0.442 m

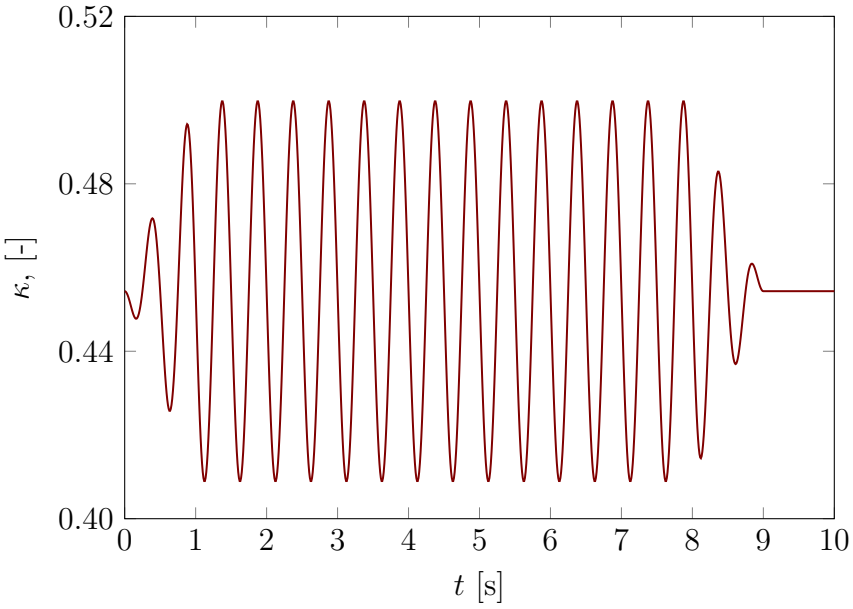
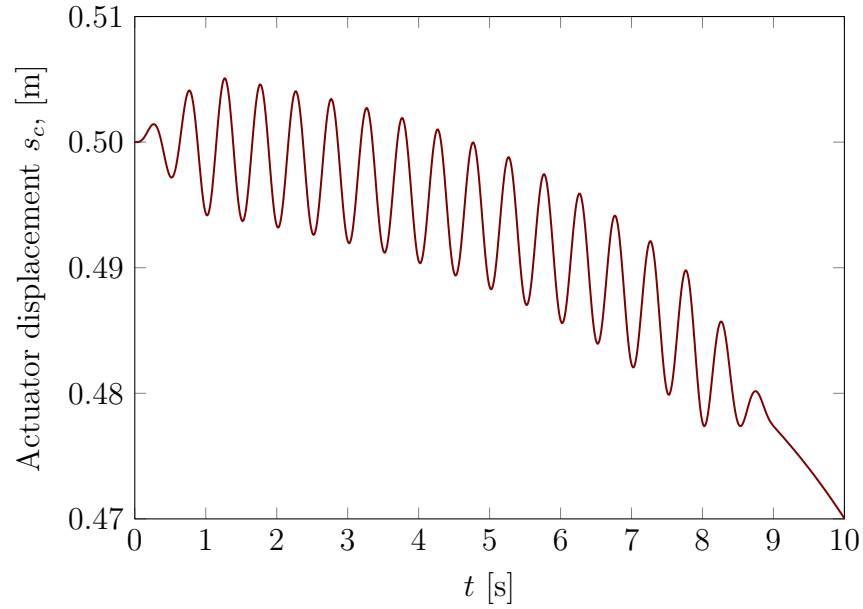


Figure 5.21: HC: Time history of valve spool displacement κ.

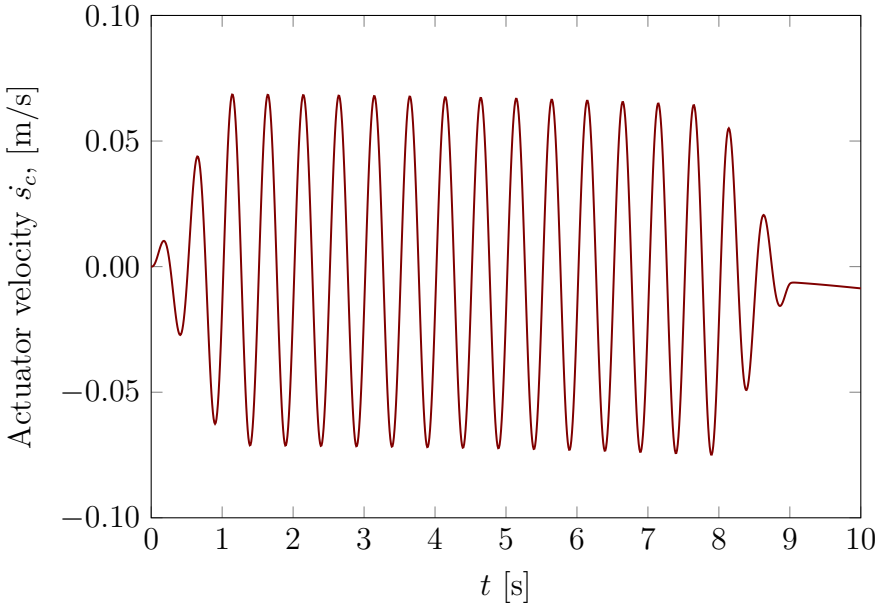
The displacement and velocity of the hydraulic actuator when the regulation valve is actuated with the control law in Eq. (5.30) are illustrated in Fig. 5.22. These results can be considered the reference solution of this benchmark example.

5.4.4.3 Force-displacement co-simulation

Again, several possibilities exist to conduct the division into subsystems of the HC example in this Section. The first option considered here is adopting a force-



(a) Displacement s_c.



(b) Velocity $\dot{s}_c$.

Figure 5.22: HC: Reference solution for the hydraulic actuator displacement and velocity.

displacement scheme as shown in Fig. 5.23. Subsystem $\mathcal{M}$ integrates the multibody dynamics and returns as outputs the displacement and velocity of the hydraulic actuator. Subsystem $\mathcal{H}$, in turn, delivers the hydraulic force f_h exerted by the piston.

The integration step-sizes of each subsystem are denoted as $h_{\mathcal{M}}$ and $h_{\mathcal{H}}$, for the mechanics and hydraulics subsystems, respectively. The macro step-size used to communicate the subsystems and the co-simulation manager is denoted by H. This

Figure 5.23: The hydraulic crane arranged following a force-displacement coupling scheme.

nomenclature will be used in all the coupling schemes, in which the HC example is involved.

In this coupling scheme, only the hydraulic subsystem is subjected to direct feedthrough $\mathbf{N}_{\mathcal{H}} \neq \mathbf{0}$. The arrays of inputs $\mathbf{u}$ and outputs $\mathbf{y}$ for each subsystem are given by

$$\mathbf{u}_{\mathcal{H}} = \begin{bmatrix} s_c^* \\ \dot{s}_c{}^* \end{bmatrix} \qquad \mathbf{u}_{\mathcal{M}} = \begin{bmatrix} f_h^* \end{bmatrix}$$
$$\mathbf{y}_{\mathcal{H}} = \begin{bmatrix} f_h \end{bmatrix} \qquad \mathbf{y}_{\mathcal{M}} = \begin{bmatrix} s_c \\ \dot{s}_c \end{bmatrix} \tag{5.31}$$

In principle, $f_h \neq f_h^*$, $s_c \neq s_c^*$ and $\dot{s}_c \neq \dot{s}_c{}^*$, because input extrapolation or some other kind of input processing can be performed by the co-simulation manager.

5.4.4.4 Pressure-displacement co-simulation

The second proposed option to split the HC into two subsystems is employing a pressure-displacement scheme as shown in Fig. 5.24. Subsystem $\mathcal{M}$ integrates the multibody dynamics and passes the displacement and velocity, which the hydraulic actuator is required to fulfil. Subsystem $\mathcal{H}$, in turn, delivers the hydraulic pressures p_1, p_2 in both chambers of the cylinder.

Figure 5.24: The hydraulic crane arranged following a pressure-displacement coupling scheme.

In this configuration, none of the subsystems is subjected to direct feedthrough. On the other hand, the mechanical subsystem needs to know the properties of the piston (area and dissipation characteristics) to evaluate the force that it exerts on link 1, according to Eq. (5.29). The vectors of inputs $\mathbf{u}$ and outputs $\mathbf{y}$ for each

subsystem are given by

$$\mathbf{u}_{\mathcal{H}} = \begin{bmatrix} s_c^* \\ \dot{s}_c{}^* \end{bmatrix} \qquad \mathbf{u}_{\mathcal{M}} = \begin{bmatrix} p_1^* \\ p_2^* \end{bmatrix}$$
$$\mathbf{y}_{\mathcal{H}} = \begin{bmatrix} p_1 \\ p_2 \end{bmatrix} \qquad \mathbf{y}_{\mathcal{M}} = \begin{bmatrix} s_c \\ \dot{s}_c \end{bmatrix} \tag{5.32}$$

As usual, the $()^*$ notation denotes the modifications in input variables performed by the co-simulation manager.

5.5 Indicators for co-simulation error evaluation

A common concern in RT applications is being able to predict if the numerical integration of the virtual subsystems is going to experience instability issues. Moreover, even if the errors introduced at the co-simulation interface do not cause the numerical integration to become unstable, there exists the risk of obtaining inaccurate results that are unreliable. These issues are present in every co-simulation setup, but they are especially critical in RT applications, such as in cyber-physical assemblies, because of the explicit coupling schemes used. The numerical errors caused by the discrete-time interface and passed to the physical subsystems could give rise to failures or lead the system into hazardous working conditions, damaging the machinery or even posing a threat to the users.

In most co-simulation applications, each subsystem can be considered a *black box* that only exposes a reduced set of variables to its environment. This is advantageous in the sense that it removes the need to disclose the subsystem internals, and so it is easy to put together simulation environments composed of several subsystems from different vendors without having to worry about how to protect intellectual property. This is especially true if their common interface is described by means of a standard, such as FMI. However, this also means that in most cases the co-simulation manager does not have any knowledge of the subsystem state, other than the information carried by the coupling variables. This makes it difficult to assess the overall energy level of the assembly, and thus identify and eventually correct any deviations from appropriate behaviour.

For the reasons pointed out above, determining the stability of a co-simulation application is a complicated task. Quantifying how much it deviates from its theoretically correct solution is even more challenging. For relatively simple systems like the LO in Section 5.4.1, if the numerical integration methods used in the subsystems are known, one could evaluate the stability of the coupling scheme by means of its spectral radius [102, 112, 173]. Determining this indicator becomes difficult for more complex problems and, moreover, a measure of stability does not convey all the relevant information about the accuracy of the results. One must bear in mind that a reference solution is not available during runtime for most practical applications, so the obtained results cannot be directly compared to it at runtime. It is possible to evaluate the convergence of particular coupling methods and schemes [102, 168] but it is not straightforward to use these to quantify the actual error introduced

by a particular simulation case, although error bounds can be defined in particular cases [174].

Some methods to quantify the error introduced in co-simulation results have been put forward nonetheless. In general, these require some information about the subsystem internals, either directly obtained [164] or provided through the directional derivatives of the subsystem state [116]. Corrective actions can be performed based on the information thus obtained, for instance adjusting the macro step-size to avoid incurring in unstable behaviour. An alternative is the use of energy-based indicators. The power residual and the concept of power bonds were introduced in [117] and used to indicate the energy error resulting from the use of discrete-time interfaces in co-simulation, and adjust the communication step-size to keep the numerical integration stable. This method was later combined with the removal of excess energy from the system to improve its accuracy [118], following an approach similar to the one used by the NEPCE method in [119]. In [109], the energy balance of the overall co-simulated system was used to determine and remove the excess energy generated at the co-simulation interface; this solution is related to the passivity-control algorithms from haptics, e.g., [175], and requires that all the subsystems provide the manager with their internal energy state.

This Section discusses the use of energy-based indicators to monitor co-simulation quality. Their meaning is interpreted depending on the type of co-simulation scheme employed, and they are assessed using the benchmark problems put forward in this Chapter.

5.5.1 Energy-based indicators

Consider a monolithic implementation that simulates a system made up of three different components and integrated by a single solver as depicted in Fig. 5.25.

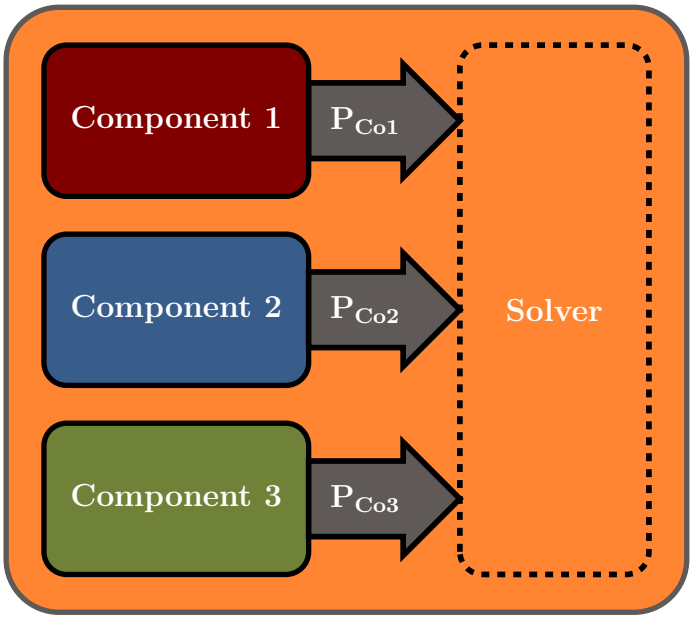


Figure 5.25: Power flows within a three-component monolithic simulation.

If the numerical errors in the integration process are not considered, the overall balance of energy of the system will be satisfied. For a mechanical assembly, for instance

$$E\left(t\right) - E_0 - W_{\mathrm{nc}}\left(t\right) = 0 \tag{5.33}$$

where E is the mechanical energy of the system, E_0 is the initial energy level, and W_{nc} is the work of nonconservative forces. Ideally, Eq. (5.33) must hold at all times. Thus, the power exchange between the three components must fulfil the conservation of energy expressed as

$$\sum_{\tilde{\alpha}=1}^{n_{\mathrm{co}}=3} P_{Co\tilde{\alpha}} = P_{Co1} + P_{Co2} + P_{Co3} = 0 \tag{5.34}$$

where $P_{Co\tilde{\alpha}}$, $\tilde{\alpha} = 1, 2, 3$, is the power exchanged between each component and the rest of the system, and n_{co} is the number of components of the system. Equation (5.34) means that, in the absence of external sources or sinks of energy, all the power flows without losses at the interface between components. This integration scheme contains a single integrator, which takes care of the numerical integration of the complete system and is allowed to access the internal information, e.g., the state, of each component at every time point.

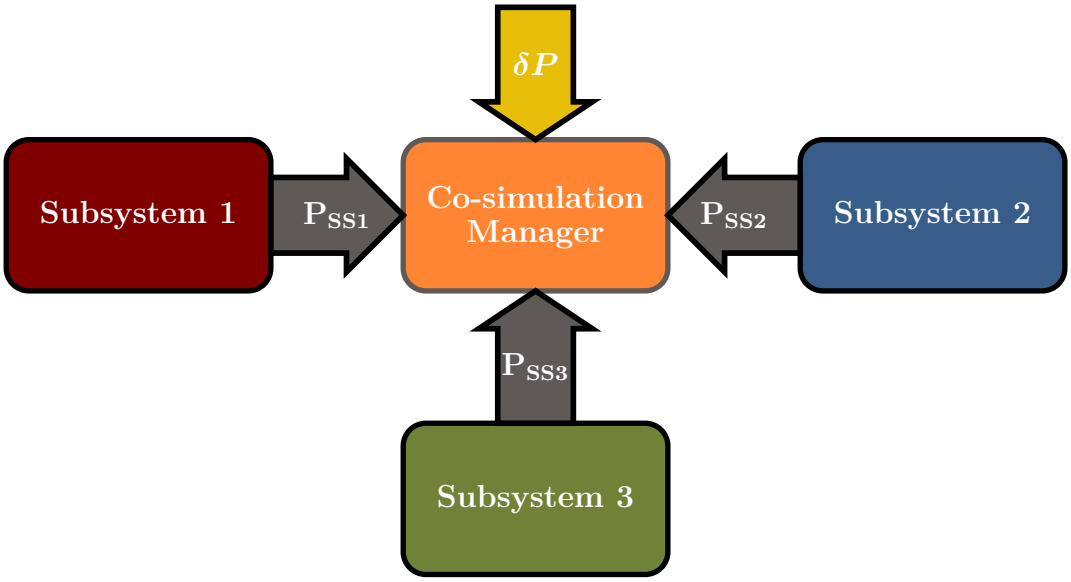


Figure 5.26: Power balance of a three-subsystem co-simulation environment.

If a co-simulation scheme is used instead, as illustrated in Fig. 5.26, the power exchanged at the interface should also verify Eq. (5.34). In practice, despite the fact that the algebraic sum of exchanged power within a co-simulation environment should be zero, the energy balance is not fulfilled. Even if numerical integration errors were considered to be zero, the system energy is modified by the the co-simulation scheme due to the discrete-time communication between the manager and the subsystems [109,117]. In terms of power, the violation of the power balance conservation can be calculated introducing the *residual power* δP as

$$\sum_{\tilde{\alpha}=1}^{n_{ss}=3} P_{SS\tilde{\alpha}} = P_{SS1} + P_{SS2} + P_{SS3} = -\delta P \neq 0 \tag{5.35}$$

where $P_{SS\tilde{\alpha}}$, $\tilde{\alpha} = 1, 2, 3$ is the power exchanged between a subsystem and the co-simulation manager through the interface, and n_{ss} is the number of subsystems.

Indicators for co-simulation quality can be derived based on the residual power δP and used to monitor the stability and accuracy of the results. The evaluation of such indicators, as will be shown in Section 5.5.2, requires access to the historical values of the coupling variables, in order to calculate the power or energy errors resulting from the selection of a certain co-simulation configuration [95, 110], but not to the internal state of the subsystems.

The indicators used in this thesis require that the product of input and output variables of the subsystems yields units of power; this is frequently the case in practical co-simulation setups. Some examples of these couplings are:

- *Force-displacement:* commonly used in systems with mechanical components.
- *Torque-angular speed:* also widely used in mechanical systems, in which rotative motors are involved.
- *Voltage-current:* used in the presence of electric or electronic components.

It is worth pointing out that, in some cases, the product of these variables does not directly result in power units, but in a function related to the power exchange. The pressure-displacement coupling in Section 5.4.4.4 is an example of this, in which the input times output product would yield W/m^2. In these situations it is also possible to evaluate the power exchange and monitor the error due to the co-simulation interface and the indicators can also be used. Other coupling schemes, conversely, do not allow the direct use of these indicators. This is the case of the displacement-displacement case with the linear oscillator in Section 5.4.1. Using the indicators with these schemes would require the evaluation of the coupling force outside the subsystems, e.g., inside the manager.

5.5.2 Evaluation and meaning of energy-based indicators

Assuming that the product of coupling variables in a certain co-simulation configuration has power units, it is possible to use the power residual δP from Eq. (5.35) as indicator. Its evaluation was presented in [117] for Jacobi single-rate time grids with two subsystems as

$$\delta P^{k+1} = -\left(\mathbf{u}^{k+1}\right)^{\mathrm{T}} \mathbf{y}^{k+1} \tag{5.36}$$

where $\mathbf{u}$ and $\mathbf{y}$ are the arrays of inputs and outputs handled by the co-simulation manager. Ideally, Eq. (5.36) should equal zero if no errors existed at the coupling interface.

Figure 5.27 illustrates the process of power exchange in a single-rate explicit Jacobi scheme. Subscripts $()_1$ and $()_2$ denote the first and second subsystems, whereas superscripts $()^k$ and $()^{k+1}$ refer to timesteps t_k and t_{k+1}, respectively. Figure 5.27 also shows the issue with the evaluation of subsystem outputs at time t_{k+1}. As mentioned in Section 5.2.3.1, the input $\mathbf{u}^{k+1}$ is necessary for the evaluation of the subsystem outputs $\mathbf{y}^{k+1}$ in subsystems with direct feedthrough. As a matter of fact, $\mathbf{u}^{k+1}$ is also necessary to complete the integration of the subsystem dynamics from

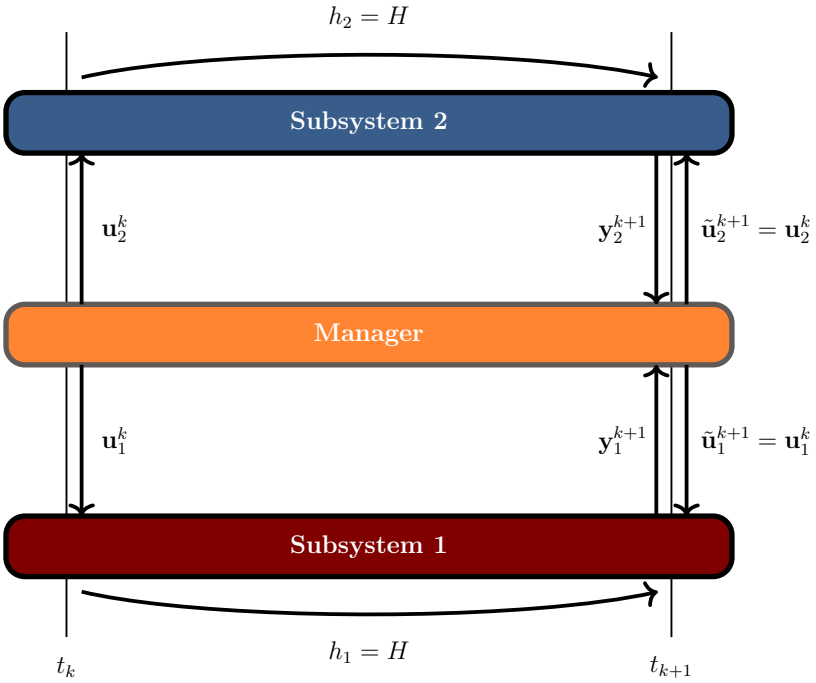


Figure 5.27: Evaluation of the residual power δP in a single-rate, explicit Jacobi scheme.

t_k to t_{k+1} if implicit integrators, like the Newmark family, are employed. For the sake of simplicity, the discussion here is limited to subsystems that integrate their dynamics with explicit integrators, e.g., the semi-implicit Euler formula.

Because input $\mathbf{u}^{k+1}$ is unknown at time t_{k+1}, systems with direct feedthrough need to approximate their value; polynomial extrapolations like ZOH or FOH are a common way to do this. Accordingly, instead of the exact input value $\mathbf{u}^{k+1}$, an approximation $\tilde{\mathbf{u}}^{k+1}$ is used to evaluate the outputs. As a consequence, the residual power becomes

$$\delta P^{k+1} \approx -\left(\tilde{\mathbf{u}}^{k+1}\right)^{\mathrm{T}} \mathbf{y}^{k+1} \tag{5.37}$$

which, in general, will no longer satisfy $\delta P = 0$.

The evaluation of the residual power corresponds to the co-simulation manager, as in general it cannot be assumed that the subsystems will return this indicator as part of their coupling variables. Because the manager does not have, in principle, information about the method that the subsystems use to approximate their inputs, some assumption has to be made regarding this. The proposed method considers that $\tilde{\mathbf{u}}^{k+1} = \mathbf{u}^k$ in the evaluation of δP in Eq. (5.37). This is equivalent to assuming that a ZOH is used inside the subsystems. The definition of the indicator can be generalized to multi-rate coupling schemes, like the one in Fig. 5.28.

Going back to Eq. (5.33), the deviations from the exact system solution caused by input extrapolation will introduce an error in the energy balance of the overall system,

$$E\left(t\right) - E_0 - W_{nc}\left(t\right) = \varsigma\left(t\right) \tag{5.38}$$

where $\varsigma\left(t\right)$ is the deviation from the theoretical energy balance at time t due to the numerical errors caused by the discrete-time interface.

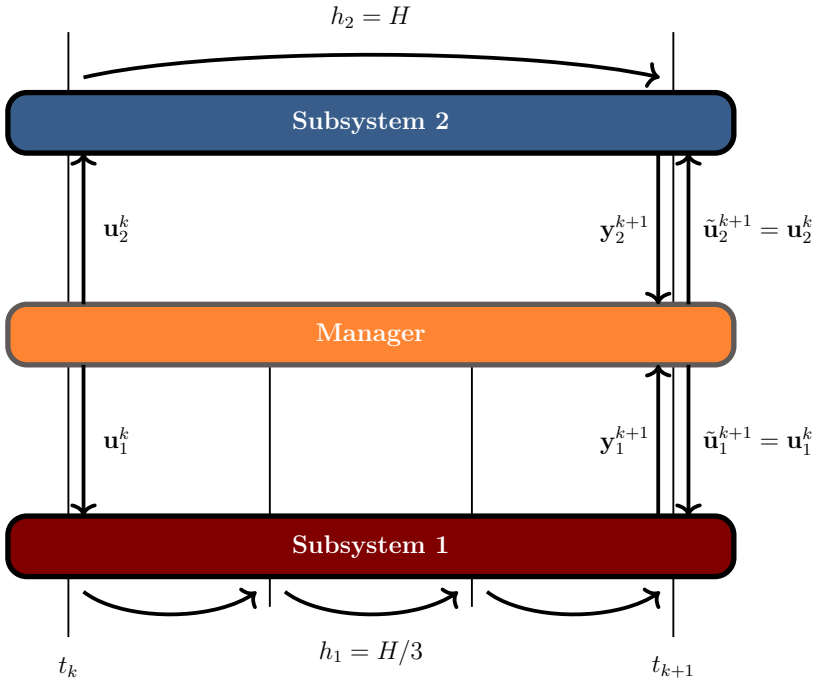


Figure 5.28: Evaluation of the residual power during an explicit co-simulation macro-step in Jacobi multi-rate schemes.

The existence of a relation between the error in the energy balance ς and the residual power δP was confirmed in [95]. However, the energy error ς of a co-simulation cannot be directly obtained integrating the δP over time. In fact, depending on the selected coupling scheme and the properties of the subsystems, it will be possible to approximate it with a fraction μ of this integral. For the first co-simulation step, for instance

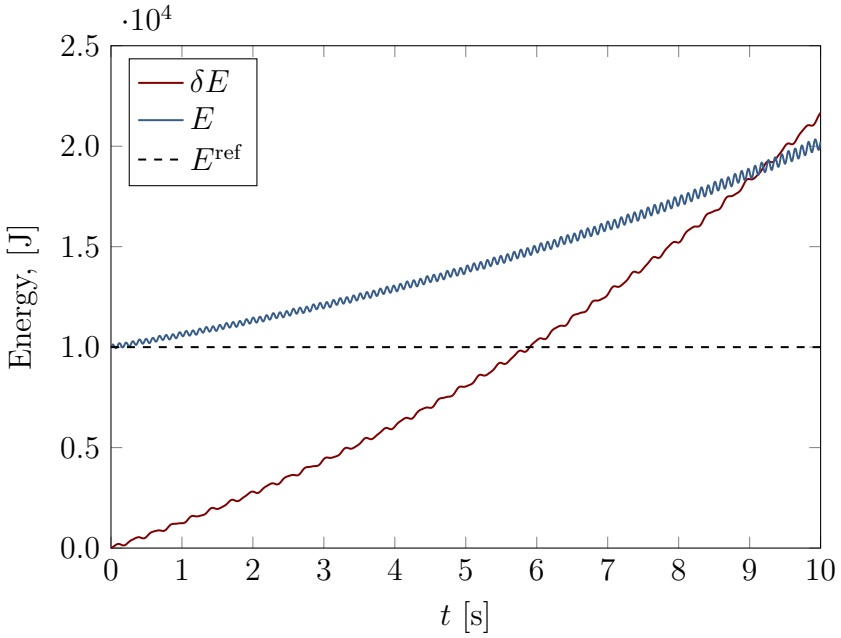
$$\varsigma\left(t_1\right) \approx \mu \int_{t_0}^{t_1} \delta P \, \mathrm{d}t \approx \mu \, H \, \delta P^1 \approx -\mu \, H \, \left(\mathbf{u}^0\right)^{\mathrm{T}} \, \mathbf{y}^1 \tag{5.39}$$

As the integration proceeds in time, however, the relation given by μ may lose accuracy, because the accumulation and propagation of the errors brings the numerical integration further away from the theoretical solution. For this reason, the value of the accumulated energy error at any instant in time cannot be calculated directly as $\varsigma\left(t\right) \neq \mu \int_{t_0}^{t} \delta P\left(t\right) \, \mathrm{d}t$; the use of expression (5.39) must be limited to the calculation of the energy error introduced by the co-simulation environment between two consecutive communication points.

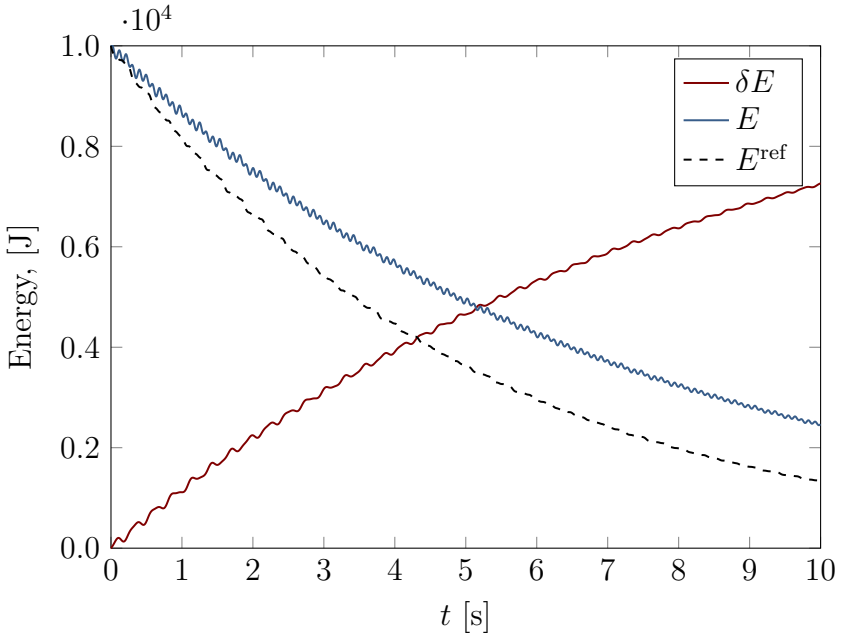
Figure 5.29 compares the mechanical energy E of a linear oscillator co-simulated following an explicit, single rate Jacobi scheme to the mechanical energy of the reference solution of this benchmark problem, E^{ref}. Results for cases 1 and 2 in Table 5.2 are shown. In both cases, the energy residual, defined as

$$\delta E = \int_{t_0}^{t} \delta P\left(t\right) \, \mathrm{d}t \tag{5.40}$$

is shown as well; in general $\varsigma\left(t\right) \neq \delta E\left(t\right)$. In systems without internal damping, the residual power δP and its integral δE indicate the increase of mechanical energy in



(a) Case 1: no damping.



(b) Case 2: with damping.

Figure 5.29: LO example: comparison of the residual energy δE, co-simulated system energy E, and reference solution energy E^{ref} in force-displacement coupling scheme and a single-rate Jacobi scheme.

the system. In dissipative systems, they indicate instead a deviation from the energy balance of the system, in which part of the error affects the mechanical energy of the assembly, and part affects the work exerted by nonconservative forces.

It must be noted that Fig. 5.29 (b) highlights two interesting facts. First, the results delivered by the co-simulation of the example show that the numerical inte-

gration is *stable*, in the sense that the system energy decreases with time; however, they also evidence that the simulation is not *accurate*, because the system energy does not follow that of the reference solution. Second, the presence of dissipation within the subsystems decreases the severity of the energy errors introduced by the time-discrete coupling interface, bringing the co-simulated solution closer to its theoretically correct counterpart.

In Equation (5.39), if the numerical integration errors in the subsystems are neglected, $\varsigma(t_1)$ can be considered the error introduced by the co-simulation scheme during the first macro step-size. Determining the actual value of fraction μ, however, requires further considerations. Its value depends on the presence of direct feedthrough in the subsystems. Moreover, in multi-rate co-simulation schemes, the ratio between macro step-sizes and the extrapolation order at the co-simulation manager also need to be considered.

In single-rate schemes, the coefficient μ equals 0.5 if one of the subsystems is subjected to direct feedthrough; its value is zero otherwise, as the coupling scheme would not introduce input extrapolation errors.

In multi-rate schemes, the evaluation of the coefficient μ becomes more complicated and it is necessary to consider the extrapolation order and the ratio between subsystem step-sizes

$$\mathcal{R} = \frac{h_{\mathrm{df}}}{h_{\mathrm{ndf}}} \tag{5.41}$$

where h_{df} and h_{ndf} are the macro step-sizes of the subsystems with and without direct-feedthrough, respectively. The value of the coefficient μ can then be calculated using the following expressions:

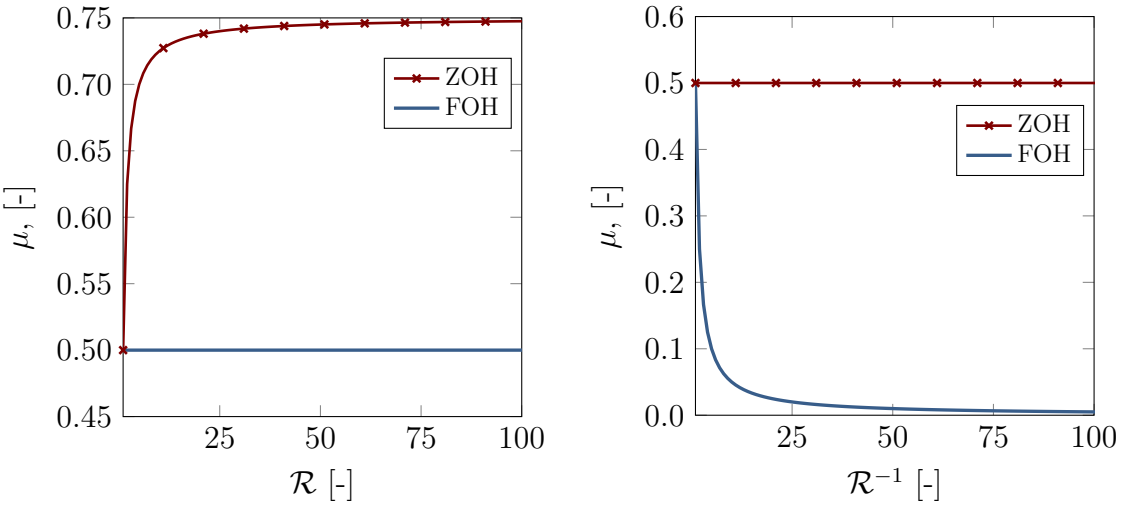
- *The subsystem with direct feedthrough has the largest step-size:* the corrective coefficient is obtained as

$$\begin{aligned} \mu_{ZOH} &= \mu_{SR}\,(1.5 - \frac{0.5}{\mathcal{R}}) \\ \mu_{FOH} &= \mu_{SR} \end{aligned} \tag{5.42}$$

- *The subsystem with direct feedthrough has the smallest step-size:* the corrective coefficient is obtained as

$$\begin{aligned} \mu_{ZOH} &= \mu_{SR} \\ \mu_{FOH} &= \mu_{SR}\,\mathcal{R} \end{aligned} \tag{5.43}$$

Figure 5.30 shows the values of μ calculated with Eqs. (5.42) and (5.43) for different values of the step-size ratio $\mathcal{R}$ and two input approximation methods, namely constant and linear extrapolation. Although they are not represented in the figure, the μ for higher order extrapolation is the same as the one used for FOH. The figure and the equations were obtained tuning the value of the μ coefficient in simulations with the linear oscillator in Section 5.4.1 to enforce the satisfaction of Eq. (5.39). The general validity of these expressions will be verified in the following Sections with different examples, and with the results of the correction method presented in 5.6.



(a) The subsystem with the largest step-size has direct feedthrough.

(b) The subsystem with the smallest step-size has direct feedthrough.

Figure 5.30: Selection of correction factor μ in a two-subsystem co-simulation, as a function of the step-size ratio and extrapolation order.

5.5.3 Monitoring co-simulation energy errors

The foregoing discussion points out that the residual power δP and its integral over time, the residual energy δE can serve as indicators to monitor co-simulation quality and identify stability and accuracy issues as the simulation progresses, in a qualitative way, with a minimal knowledge of the subsystems in the environment. This is an interesting feature for real-time applications, especially cyber-physical ones, in which unstable behaviour can result in damages to the equipment and safety hazards.

Monitoring these indicators as the numerical integration progresses provides information about the stability and the accuracy of the simulation. The indicator values can, for example, oscillate within an admissible range. This range is greatly dependent on the application at hand, and it must often be defined based on some previous knowledge of the system behaviour. In this case, the co-simulated dynamics would be slightly affected, and the results could be considered accurate. Conversely, the presence of sharp changes in the indicator values can be a symptom of unstable behaviour and degraded accuracy in the results. The benchmark examples presented in Section 5.4 are used next to illustrate these statements.

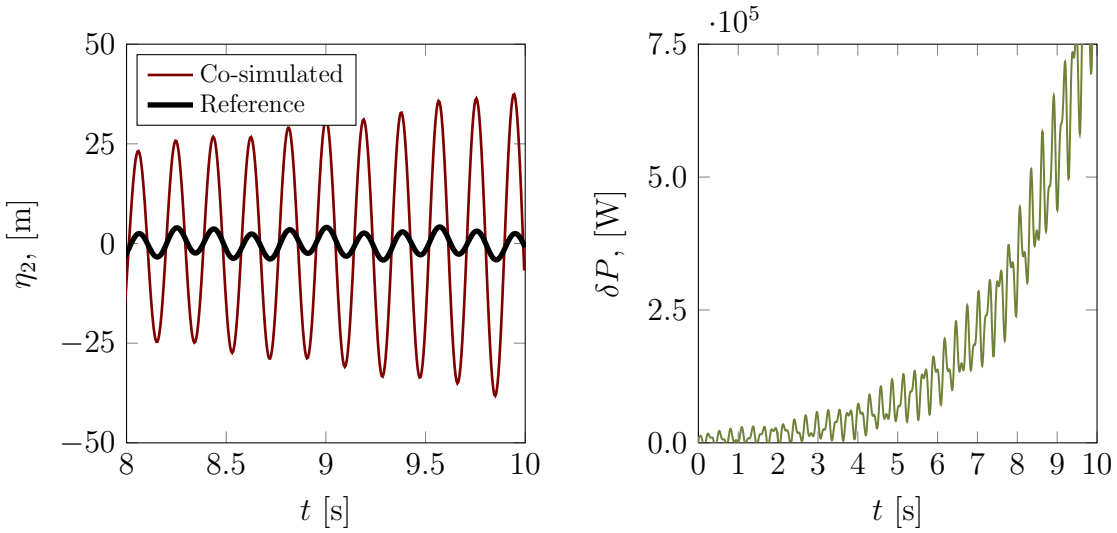
5.5.3.1 Benchmark results

The benchmark examples in Section 5.4 were used to determine the ability of the residual power δP to indicate co-simulation quality. They were simulated using single-rate and multi-rate explicit Jacobi schemes. Because the δP indicates errors associated with input extrapolation, the simulations focused on cases with direct feedthrough.

The first benchmark example tested was the linear oscillator in Section 5.4.1.

Table 5.1 defines two sets of physical parameters that represent a damped and undamped case, respectively. Both were simulated using a force-displacement configuration, which caused the first subsystem to be affected by direct feedthrough. The macro step-sizes used to simulate the system motion were varied in a range between 0.1 ms and 5 ms, both for single- and multi-rate schemes.

The undamped case (case 1 in Table 5.1) is a conservative system, so the LO should oscillate with a constant amplitude, keeping its initial level of energy. Explicit Jacobi co-simulation, both in single-rate and multi-rate configurations, resulted in the uncontrolled increase of the mechanical energy of the system, something which is in agreement with the results shown in [109]. The amplitude of the oscillation of the two masses increased with time, accordingly, in all the simulations, as a result of the energy introduced at the coupling interface.



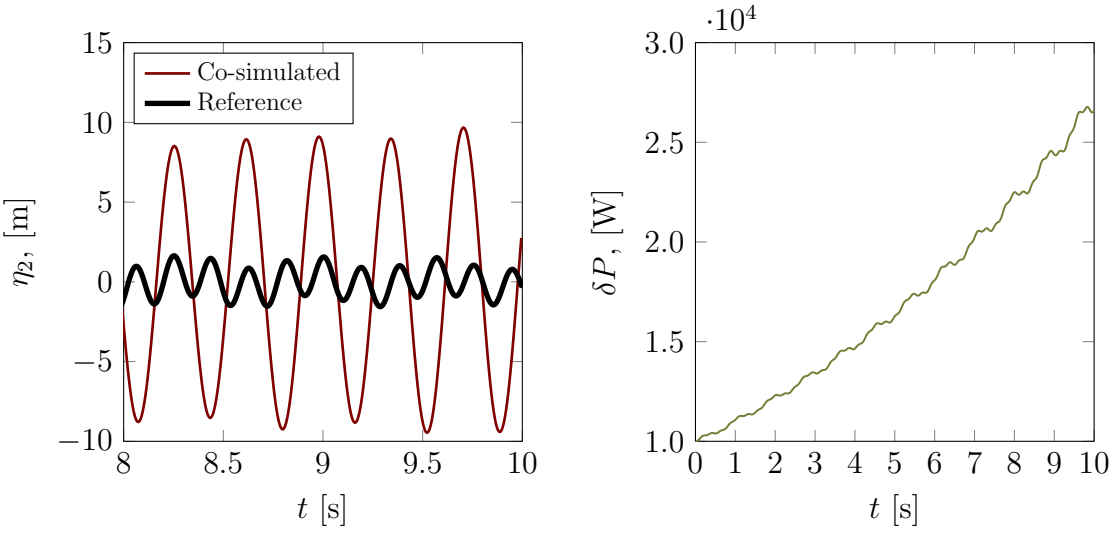
(a) Displacement η_2 of the second mass.

(b) Residual power δP.

Figure 5.31: Linear oscillator: Displacement η_2 of the second mass and residual power δP in the single-rate co-simulation of case 1: $H = h_{\mathcal{M}_1} = h_{\mathcal{M}_2} = 5$ ms.

Figure 5.31 shows the displacement η_2 of the second mass of the oscillator and compares it to the reference solution, for the single-rate co-simulation of the system with $H = h_{\mathcal{M}_1} = h_{\mathcal{M}_2} = 5$ ms. The figure also shows the residual power δP, which increases indefinitely with time; this matches the growth in the amplitude of the oscillation of the system masses. For this undamped case, it was also verified that the energy residual δE was close to half the increment in the mechanical energy of the system.

Figure 5.32 shows results from the second scenario, in which damping is present in the system. Similarly to what happens in case 1, the power residual follows an upwards trend. The system behaviour is still unstable, although the increase of both its internal energy and the amplitude of oscillations is less severe than in the undamped case. The δP captures this behaviour correctly. It must be pointed out that the energy residual δE also increased with time, but in this case its relation to the increase in the mechanical energy of the system was lost, because part of the energy errors caused by input extrapolation affect the work of the nonconservative



(a) Displacement η_2 of the second mass.

(b) Residual power δP.

Figure 5.32: Linear oscillator: Displacement η_2 of the second mass and residual power δP in the single-rate co-simulation of case 2: $H = h_{\mathcal{M}_1} = h_{\mathcal{M}_2} = 5$ ms.

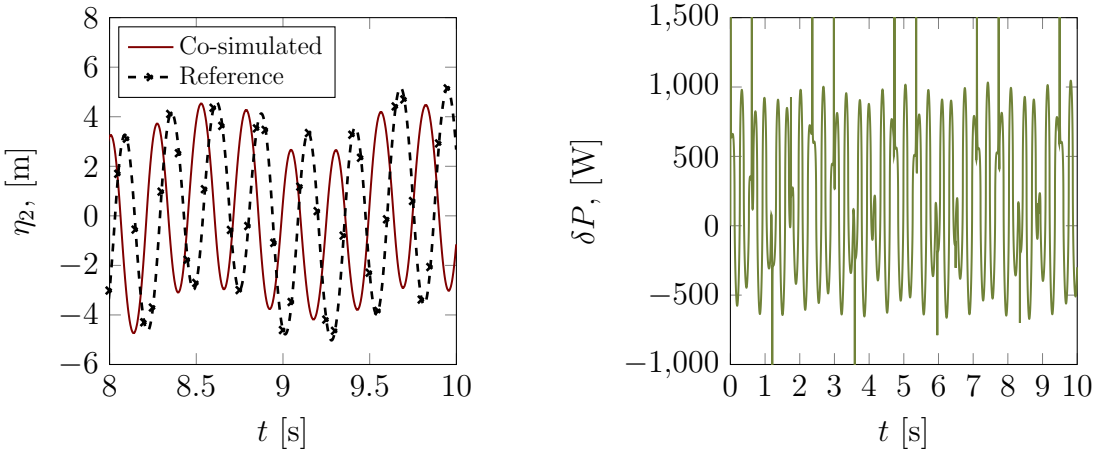
forces.

In order to verify that the indicator represents the energy behaviour of the simulation for different values of the system properties, additional numerical experiments were performed varying the physical parameters of the oscillator. Masses m_1 and m_2 were varied in the range $[0.1, 10]$ kg, and the stiffness values k_1, k_c and k_2, in the range $[10, 1000]$ N/m.

The two parameter selections in Table 5.2 were also used to define two cases (damped and undamped) with the nonlinear oscillator in Section 5.4.2. The first case considers nonlinear spring forces and no damping. In the second one, linear springs and a quadratic expression of damping are used. The same co-simulation configurations tested with the linear oscillator were assessed with this nonlinear example as well.

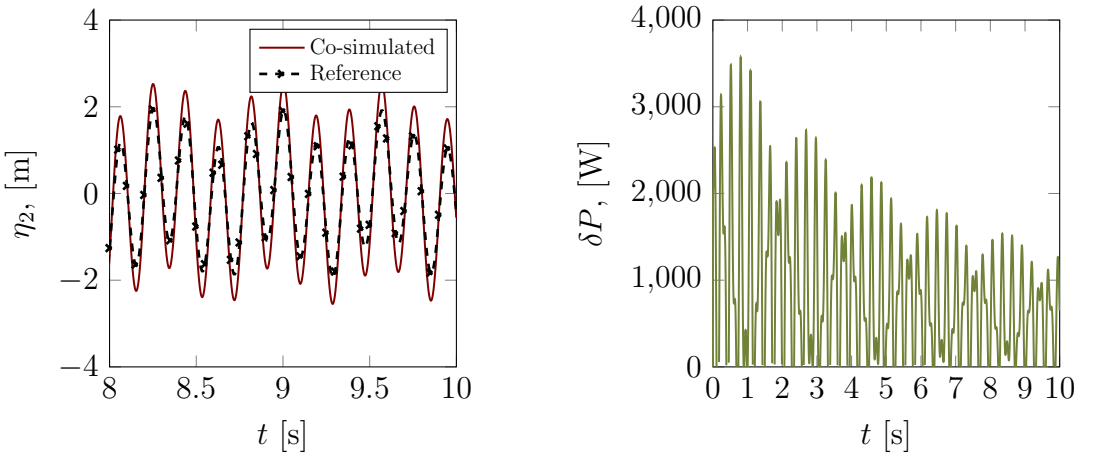
Results with this nonlinear example confirmed the findings obtained with its linear counterpart. Figure 5.33 shows the power residual and the computed motion of the second mass for one of the tested co-simulation configurations ($H = h_{\mathcal{M}_1} = h_{\mathcal{M}_2} = 1$ ms) and case 1. In this simulation, the power residual δP stayed in a consistent range during motion. The mean value of the power residual is slightly above zero, which introduces a certain amount of artificial energy in the system during runtime. This results in a small growth in the motion amplitude; a time-offset with respect to the reference solution can also be observed in Fig. 5.33.

Figure 5.34 shows the results obtained with the same configuration in case 2. The quadratic damping helps stabilize the numerical integration of the system dynamics. In fact, the residual power δP shows relatively high values at the simulation onset, but they decrease as time progresses, as a consequence of dissipation. An excess energy is introduced nonetheless in the dynamics of the oscillator and its motion amplitude is larger than that of its reference solution. Again, even though the simulation is stable, accuracy issues still remain in the solution.



(a) Displacement η_2 of the second mass. (b) Residual power δP.

Figure 5.33: Nonlinear oscillator: Displacement η_2 of the second mass and residual power δP in the single-rate co-simulation of case 1: $H = h_{\mathcal{M}_1} = h_{\mathcal{M}_2} = 1$ ms.

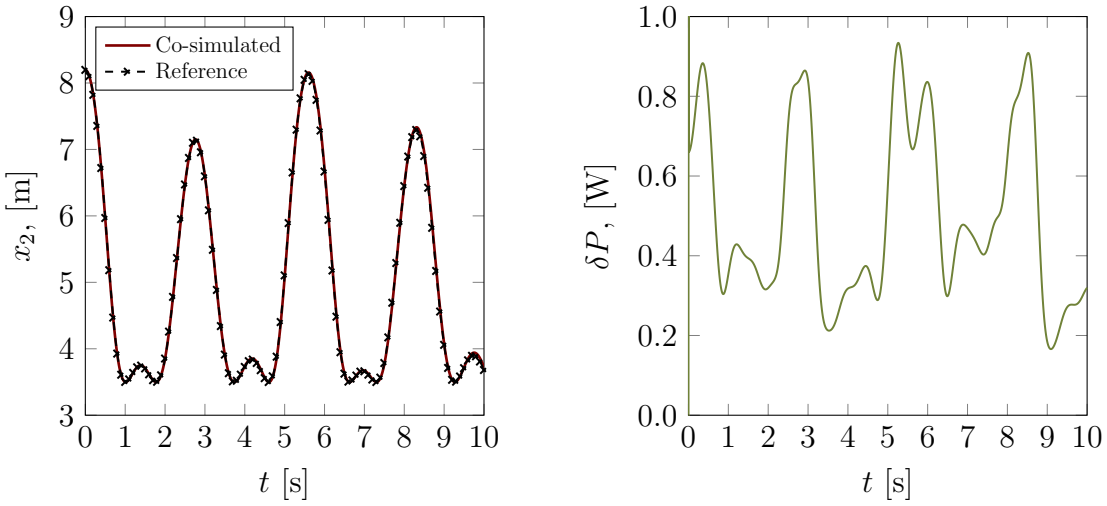


(a) Displacement η_2 of the second mass. (b) Residual power δP.

Figure 5.34: Nonlinear oscillator: Displacement η_2 of the second mass and residual power δP in the single-rate co-simulation of case 2: $H = h_{\mathcal{M}_1} = h_{\mathcal{M}_2} = 1$ ms.

In the third benchmark problem, namely the two pendula coupled by a spring-damper system, the simulation of case 1 in Table 5.3 delivers relatively low values of the residual power δP. Figure 5.35 shows results for the single-rate configuration in which $H = h_{\mathcal{M}_1} = h_{\mathcal{M}_2} = 1$ ms. The obtained power residual remains under 1 W during the first ten seconds of motion, although its value is always greater than zero. As such, energy is continuously being injected in the system; this leads to small deviations with respect to the reference solution.

The last example, the hydraulic crane in Section 5.4.4, is particularly interesting for two reasons. In the first place, unlike the other examples, it represents a system with an external energy supply that is a priori unknown. The cylinder in the assembly is connected to a hydraulic circuit that modifies the oil pressure inside the actuator and, thus, injects energy into the system. The amount of energy intro-

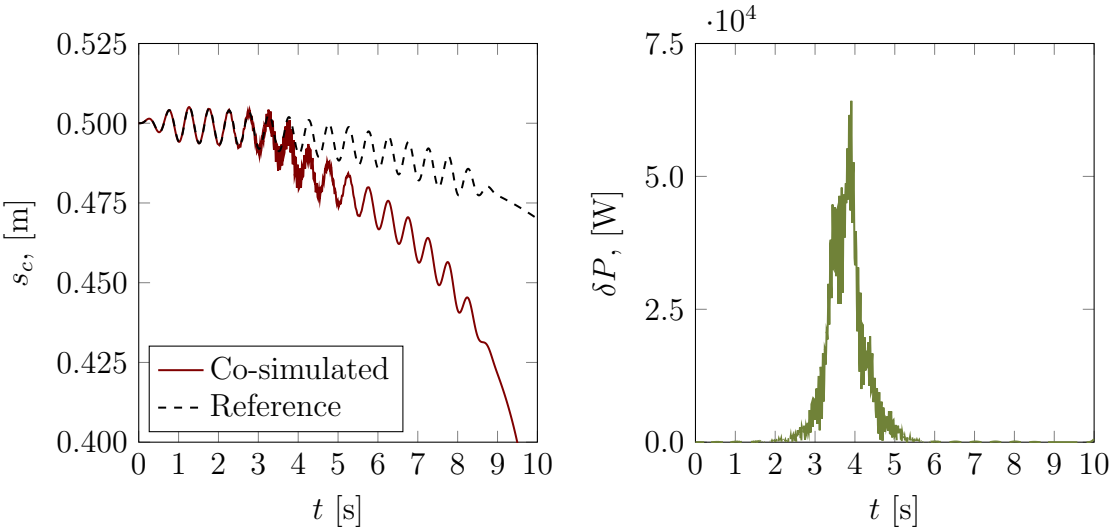


(a) Horizontal position x_2.

(b) Residual power δP.

Figure 5.35: Two-pendulum example: Coordinate x_2 and residual power δP in the single-rate co-simulation of case 1: $H = h_{\mathcal{M}_1} = h_{\mathcal{M}_2} = 1$ ms.

duced is not directly controlled, but depends on the displacement of the spool of an actuated valve, the κ magnitude in Eq. (5.30). Moreover, it is a highly dissipative system, due to the damping that acts on the hydraulic piston; this means that the numerical errors associated with co-simulation interfaces could be masked by this dissipation and lead to stable but inaccurate solutions.



(a) Cylinder displacement s_c.

(b) Residual power δP.

Figure 5.36: Hydraulic crane: Cylinder displacement s_c and residual power δP in the multi-rate co-simulation of the system motion: $H = h_{\mathcal{M}} = 11.5$ ms and $h_{\mathcal{H}} = 0.1$ ms with FOH extrapolation for the inputs of the hydraulic subsystem.

The co-simulation of the HC example in a force-displacement configuration causes the hydraulic subsystem to be subjected to direct feedthrough. Preliminary simula-

tions confirmed that, with this configuration, the integration of the system dynamics requires small step-sizes for the hydraulics subsystem, in the order of 0.1 ms, to keep the stability of the process. The multibody subsystem, conversely, can integrate its dynamics with macro steps longer than 10 ms, a property that had been identified in [96]. The example lends itself well to the use of multi-rate co-simulation schemes. It must be noted that the simulation is at risk of becoming unstable if the macro step in the multibody subsystem exceeds a certain threshold, which depends on the manoeuvre performed with the system, but is typically above 10 ms.

Figure 5.36 corresponds to the multi-rate co-simulation of the crane operation with $H = h_{\mathcal{M}} = 11.5$ ms and $h_{\mathcal{H}} = 0.1$ ms in a force-displacement coupling scheme. Between communication points of the multibody subsystem, the inputs for the hydraulics were extrapolated by the manager using a FOH approach. The figure shows that the cylinder displacement suffers high-frequency oscillations and deviates from the reference solution at $t \approx 3$ s. The system motion remains stable, partly because the dissipation at the cylinder enables it to recover from the excess energy introduced by the coupling. If a reference solution were not available, however, it would be difficult to determine whether the predicted motion is correct, and the oscillations correspond to the actual system behaviour, or are a numerical artefact caused by the co-simulation interface. The residual power, also shown in Fig. 5.36, permits the identification of the oscillations as the result of numerical errors, thus labelling the simulation as unreliable.

5.6 Correction of co-simulation energy

In Section 5.5.3 the residual power δP was used to monitor co-simulation and detect the energy errors introduced by discrete coupling interfaces. The indicator can be used to provide warnings about unreliable results and to prevent numerical integrations from going unstable. As it was shown by the simulation of the benchmark examples, the indicator provides a qualitative insight into the system behaviour and its use requires a certain degree of knowledge of its dynamics.

In this Section, the information conveyed by the residual power is used to develop a correction method aimed at removing the energy deviations caused by input extrapolation. The presented method is similar to the energy correction solution described in [109], in the sense that its objective is enforcing the satisfaction of the energy balance of the overall system. A major difference is that this method does not require the subsystems to provide information about their internal energy state; this requirement may not be possible to achieve in a number of practical applications. In cyber-physical setups, for instance, the only information sent from physical components to the co-simulation environment is the one contained in their coupling variables. Other energy-correction approaches for co-simulation, like the ones in [119] and [118], are also based on controlling the energy exchanged at the interface between subsystems. The method presented in this Section does not modify the macro step-size of the co-simulation; instead, it introduces a correction force at the coupling interface that dissipates excess energy, or introduces back in the system the energy that has been lost, similarly to what an adaptive damping element would do, with the aim of minimizing the impact on the system dynamics caused by the

energy errors associated with certain co-simulation schemes.

As mentioned in Section 5.5.3, the integral of the residual power over time does not correspond exactly with the accumulated energy error of the overall system, and needs to be corrected by means of a coefficient μ. The energy error between two consecutive communication points can then be approximated as

$$\delta E^{*,k+1} = \mu \, \delta E^{k+1} \approx \mu \, \delta P^{k+1} \, H \tag{5.44}$$

where $\delta E^{*,k+1}$ denotes the energy error added to the system between communication points at times t_k and t_{k+1}, and δP^{k+1} is the power residual evaluated at time t_{k+1}. The energy error in Eq. (5.44) can be used to select the value of the corrective action that will introduce or remove energy during the next macro step-size, from t_{k+1} to t_{k+2}, in order to readjust the energy level of the system. The nature of this action depends on the kind of coupling variables exchanged at the interface; in a force-displacement configuration, for instance, it will be a force.

Figure 5.37: Energy correction via modification of the coupling variables.

Figure 5.37 conceptually illustrates the correction scheme used. In the shown co-simulation graph, both subsystems send their outputs $\mathcal{A}$ and $\mathcal{B}$ to the co-simulation manager in a force-displacement coupling or similar configuration, e.g., torque-angular speed. Here, subsystem 1 is assumed to deliver as output a force or torque (an *action*), while the other subsystem will return kinematic quantities, such as position or angular displacement. In general, subsystems that return actions are more likely to present direct feedthrough, as they often need to know the position or velocity of the other subsystem to compute their interaction correctly. For the sake of clarity, it is assumed here that $\mathcal{A}$ represents a velocity and $\mathcal{B}$, a force. The correction method in this Section is based on modifying the action sent as input to the second subsystem with an additional term $\mathcal{B}_{corr}$ that corrects the energy error from the previous macro step,

$$-\delta E^{*,k+1} = \mathcal{B}_{corr}^{k+1} \, \mathcal{A}^{k+1} \, H \qquad \rightarrow \qquad \mathcal{B}_{corr}^{k+1} = \frac{-\delta E^{*,k+1}}{\mathcal{A}^{k+1} \, H} \tag{5.45}$$

In general, it is advisable to apply the correction in Eq. (5.45) to variables that represent actions, and not kinematic variables. This can be seen as introducing a corrective action in the system, as opposed to modifying the position or velocity of a component. In cyber-physical applications, it is usually easier to apply the correction to the virtual subsystems than to the physical ones. Even if the input to a physical component is a force, the actuators that interact with it may not be able to apply the necessary corrections, e.g., because of a limited motion range or because the corrective action needed may not be aligned with the direction of the actuator. Virtual systems, in contrast, usually do not have such limitations.

The applicability of Eq. (5.45) is limited by several facts. Firstly, if the value of the coupling variable $\mathcal{A}$ is zero or close to zero, Eq. (5.45) will result in a large value of the correction term, which is generally not compatible with the system dynamics. The value of the correction force must be limited to avoid introducing an instability in the co-simulation dynamics. Moreover, for very small values of $\mathcal{A}$, it may not be possible to perform any correction of the system energy, and so it will not be possible to apply the method at certain communication points. Secondly, the correction is calculated using the values of energy and velocity obtained at the end of the macro step between t_k and t_{k+1}, but the correction action $\mathcal{B}_{corr}^{k+1}$ has to be applied during the next macro step, in which the value of the coupling velocity $\mathcal{A}^{k+1}$ used to evaluate the correction is unlikely to remain constant. For this reason, the energy correction through the introduction of $\mathcal{B}_{corr}^{k+1}$ will not match exactly the value of $\delta E^{*,k+1}$ intended to remove, but a difference ΔE will exist between them. A possible way to estimate this value is evaluating the difference between the intended correction and the one that would result using the coupling velocity at time $k+2$

$$\Delta E^{k+1} \approx \delta E^{*,k+1} + \mathcal{B}_{corr}^{k+1}\, \mathcal{A}^{k+2}\, H \tag{5.46}$$

Term ΔE^{k+1} in Eq. (5.46) can be seen as the energy deviation that the method has been unable to correct during the last macro step. The accumulation of these terms over time leads gives rise to an uncorrected energy deviation $\mathcal{E}$ in the overall system dynamics

$$\mathcal{E}^{k+1} = \mathcal{E}^{k} + \Delta E^{k+1} \quad ; \qquad \text{where} \quad \mathcal{E}^{0} = 0 \text{ J} \tag{5.47}$$

An extra term can be introduced in Eq. (5.45), aiming at the removal of $\mathcal{E}^{k+1}$

$$\mathcal{B}_{corr}^{k+1} = \frac{-\delta E^{*,k+1}}{\mathcal{A}^{k+1}\, H} - k_{\mathrm{i}} \frac{\mathcal{E}^{k+1}}{\mathcal{A}^{k+1}\, H} \tag{5.48}$$

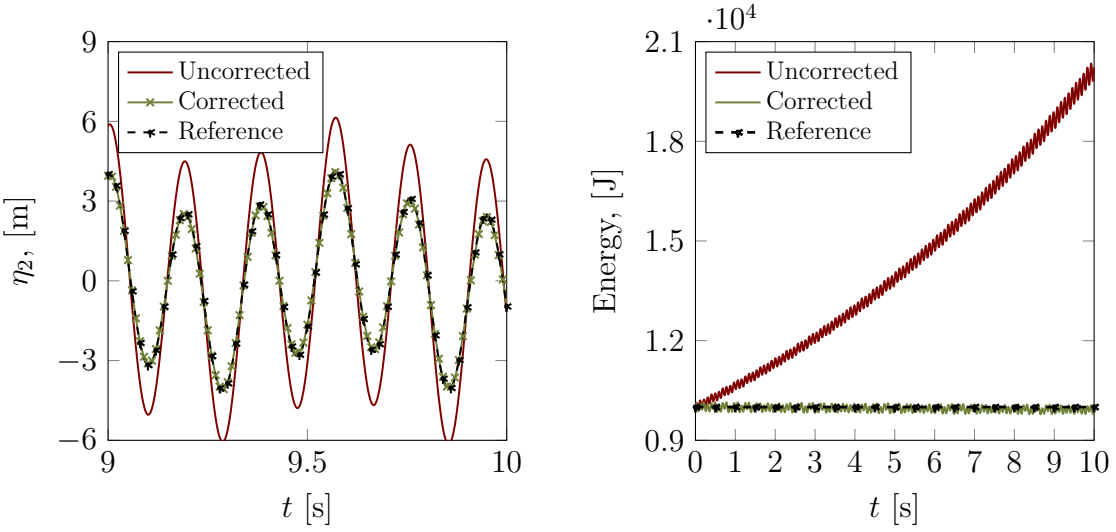
where k_{i} is a weight coefficient similar to an integral gain, in the range $[0, 1]$.

Moreover, additional improvements could be performed on the method, such as using other integration formulas in Eq. (5.44) or conducting some prediction of the velocity during the next communication step when calculating the correction action in Eq. (5.45). However, the results shown next in the simulation of the benchmark problems use the formula in Eq. (5.48).

5.6.1 Application to benchmark problems

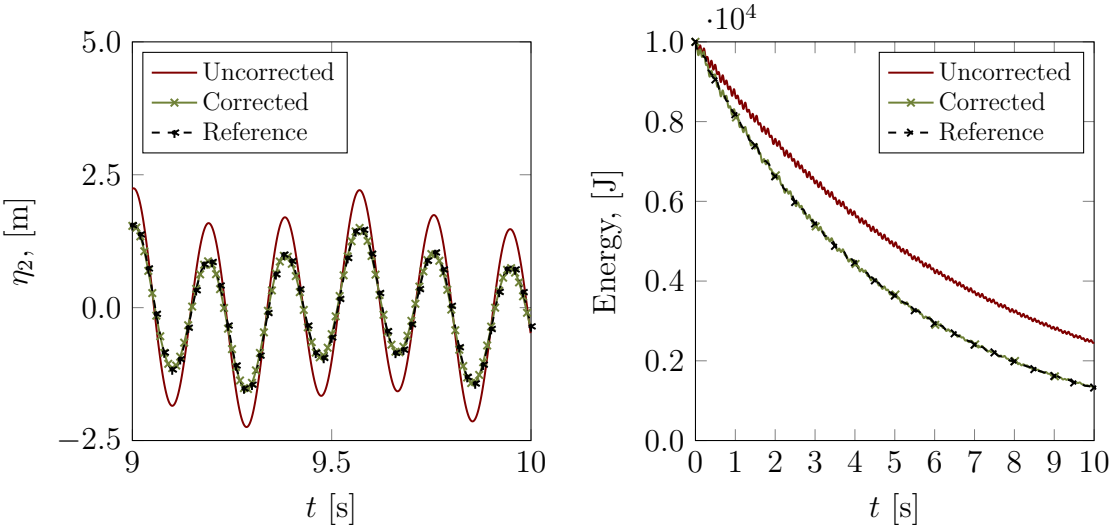
The energy correction method in Eq. (5.48) was applied to the co-simulation of the benchmark examples in Section 5.4. Explicit Jacobi schemes, both single- and multi-rate, were tested. Again, coupling variable selections leading to the existing of direct feedthrough were used to give rise to the need for input extrapolation. The same simulation cases and scenarios in Section 5.5.3 were evaluated with this energy correction approach. In all cases, the coefficient μ necessary to interpret correctly the physical meaning of the residual power δP was adjusted beforehand using the expressions in Eqs. (5.42) and (5.43).

Figures 5.38 and 5.39 show two instances of the single-rate co-simulation of the linear oscillator introduced as benchmark problem in Section 5.4.1. The plots contain



(a) Displacement η_2 of the second mass. (b) Mechanical energy.

Figure 5.38: Linear oscillator, case 1: Displacement η_2 and mechanical energy in single rate co-simulation. $H = h_{\mathcal{M}_1} = h_{\mathcal{M}_2} = 1$ ms, $\mu = 0.5$.

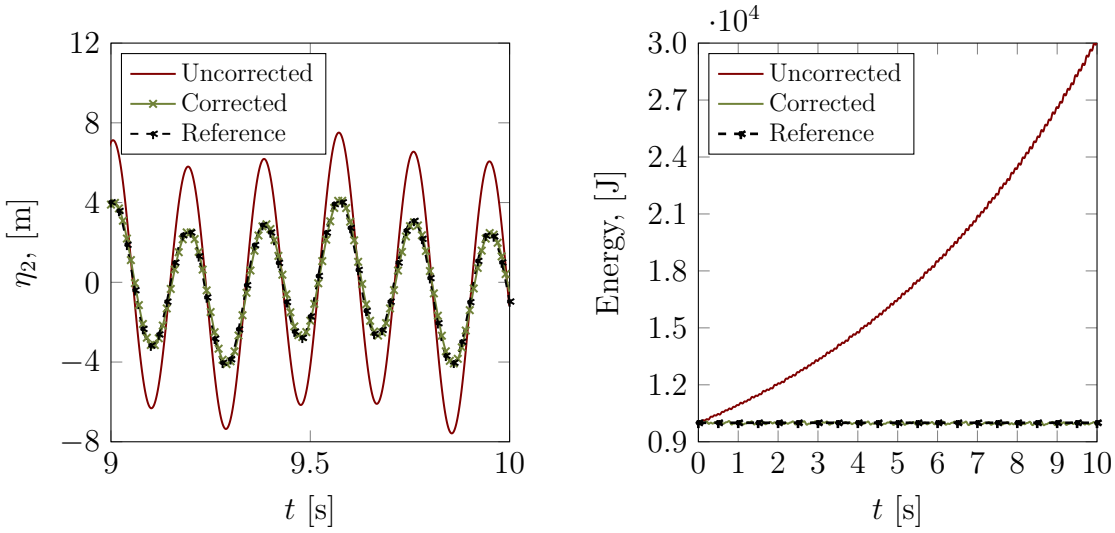


(a) Displacement η_2 of the second mass. (b) Mechanical energy.

Figure 5.39: Linear oscillator, case 2: Displacement η_2 and mechanical energy in single rate co-simulation. $H = h_{\mathcal{M}_1} = h_{\mathcal{M}_2} = 1$ ms, $\mu = 0.5$.

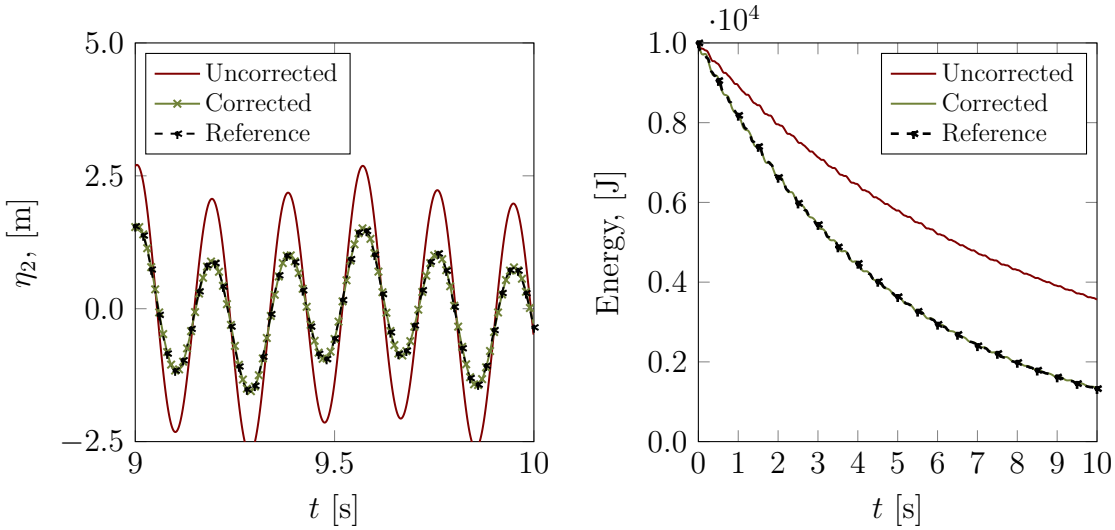
the displacement η_2 of the second subsystem mass and the mechanical energy of the whole system. The uncorrected co-simulation shows significant deviations with respect to the reference solution, both in the mechanical energy of the system and the predicted motion. The co-simulation of the undamped oscillator is unstable, and its energy grows indefinitely over time, as can be seen in Fig. 5.38b. The mechanical energy in the damped case does not grow, so the simulation would be stable; however, the motion does not agree with the theoretical solution of the problem. Results confirmed that the correction method was able to remove energy

errors from the solution and match the reference in all cases. The obtained motion is not only stable, but also accurate. Because of the single-rate coupling scheme, the coefficient μ was set to 0.5. The oscillator is a linear example, which did not require the use of the integral correction in Eq. (5.48).



(a) Displacement η_2 of the second mass. (b) Mechanical energy.

Figure 5.40: Linear oscillator, case 1: Displacement η_2 and mechanical energy in multi-rate co-simulation. $H = h_{\mathcal{M}_1} = 1$ ms, $h_{\mathcal{M}_2} = 0.1$ ms, ZOH extrapolation, $\mu = 0.725$.



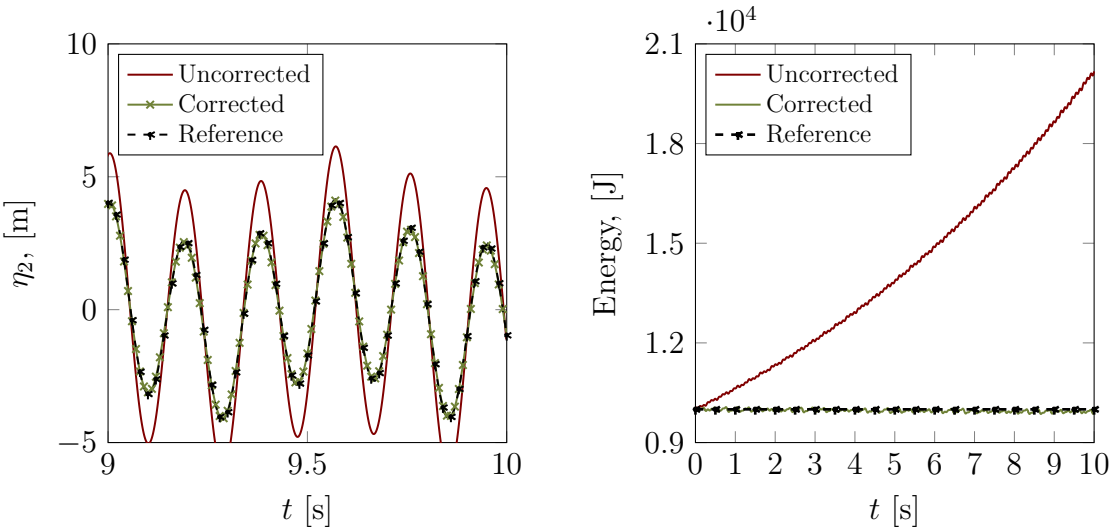
(a) Displacement η_2 of the second mass. (b) Mechanical energy.

Figure 5.41: Linear oscillator, case 2: Displacement η_2 and mechanical energy in multi-rate co-simulation. $H = h_{\mathcal{M}_1} = 1$ ms, $h_{\mathcal{M}_2} = 0.1$ ms, ZOH extrapolation, $\mu = 0.725$.

The multi-rate co-simulation of the linear oscillator requires the adjustment of the correction coefficient μ with Eqs. (5.42) and (5.43). The input extrapolation

order used for the fast subsystem at the co-simulation manager needs to be taken into consideration as well.

Figures 5.40 and 5.41 illustrate the multi-rate scheme with ZOH extrapolation for the evaluation of the inputs of the fast subsystem, in this case the second mass. It is worth mentioning that the reduction of the macro-step in subsystem 2 did not improve the behaviour of the uncorrected co-simulation with respect to the single-rate scheme. On the contrary, the simulation results deviated further from the reference solution. The use of FOH extrapolation improved these a bit, as can be appreciated in Figs. 5.42 and 5.43, but significant deviations with respect to the reference solution remain in the results.



(a) Displacement η_2 of the second mass.

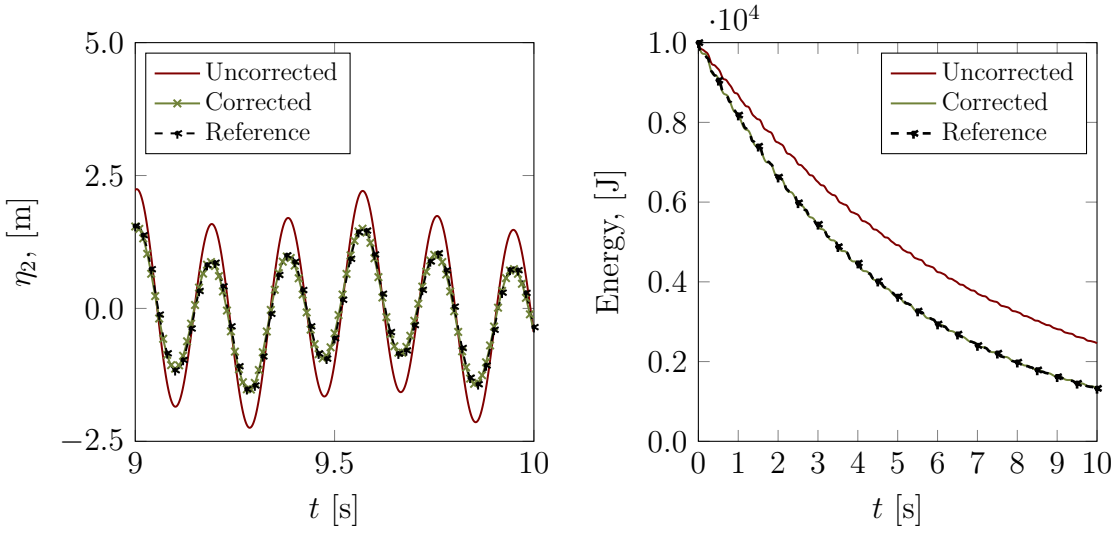
(b) Mechanical energy.

Figure 5.42: Linear oscillator, case 1: Displacement η_2 and mechanical energy in multi-rate co-simulation. $H = h_{\mathcal{M}_1} = 1$ ms, $h_{\mathcal{M}_2} = 0.1$ ms, FOH extrapolation, $\mu = 0.5$.

Regardless of the considered case, the energy correction in Eq. (5.48) removed the errors introduced by the coupling interface. The integral term of the correction was not necessary either in the multi-rate schemes.

The proposed method was also able to handle energy errors in the co-simulation of the nonlinear oscillator in Section 5.4.2. Figure 5.44 shows the results in the single-rate simulation of case 1, with nonlinear spring forces and no dissipation. The uncorrected co-simulation does not introduce an amount of energy into the system as large as in the case of the linear oscillator; however, it deteriorates the dynamics response with a visible offset between the reference and the co-simulated dynamics. Moreover, the mechanical energy in the uncorrected simulation grows consistently, which renders the integration unstable. In case 2, with linear spring forces and nonlinear damping, excess energy caused similar problems to those found in the co-simulation of the linear oscillator.

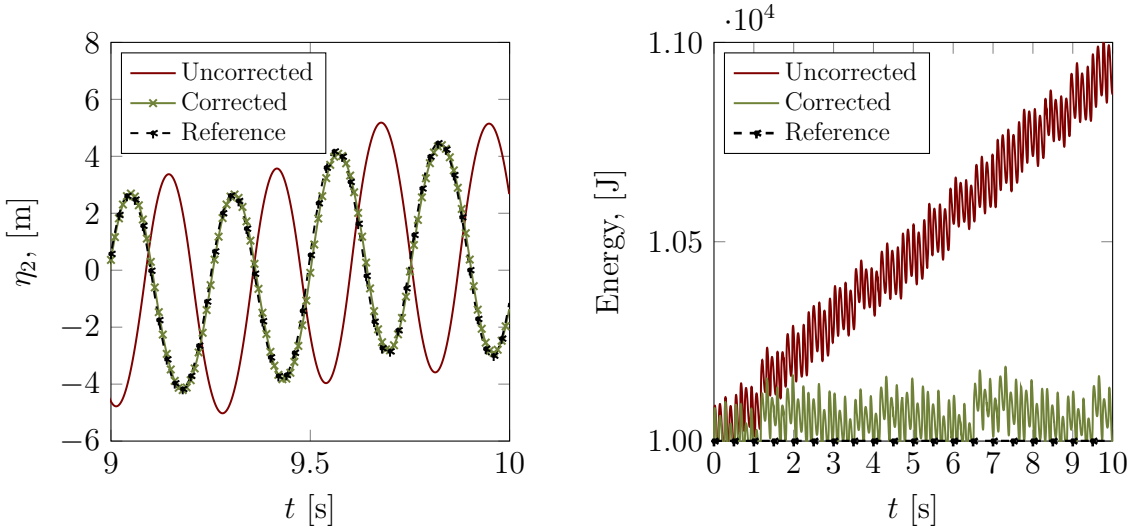
Again, the correction method removed the excess energy from the system and brought its dynamics closer to those of the reference solution. This example, however, required the use of the integral term in the method in case 1.



(a) Displacement η_2 of the second mass.

(b) Mechanical energy.

Figure 5.43: Linear oscillator, case 2: Displacement η_2 and mechanical energy in multi-rate co-simulation. $H = h_{\mathcal{M}_1} = 1$ ms, $h_{\mathcal{M}_2} = 0.1$ ms, FOH extrapolation, $\mu = 0.5$.

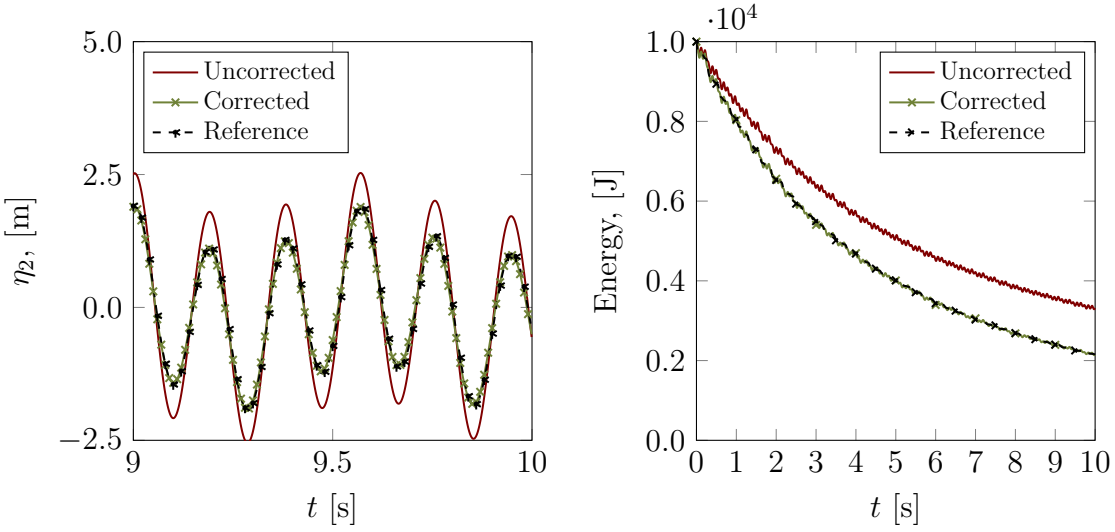


(a) Displacement η_2 of the second mass.

(b) Mechanical energy.

Figure 5.44: Nonlinear oscillator, case 1: Displacement η_2 and mechanical energy in single-rate co-simulation. $H = h_{\mathcal{M}_1} = h_{\mathcal{M}_2} = 1$ ms, $\mu = 0.5$, $k_\mathrm{i} = 0.25$.

The uncorrected co-simulation of the set of pendula used as third benchmark problem does not introduce large energy errors in the system dynamics, as indicated by the relatively low value of the δP in Fig. 5.35b. The mild deviation from the reference solution of the system motion can nonetheless be corrected by acting on the energy errors at the coupling interface. Figure 5.46 shows this and also the fact that the system energy grows steadily if not corrected. The modifications introduced by the correction method did not modify the dynamics significantly, but prevented

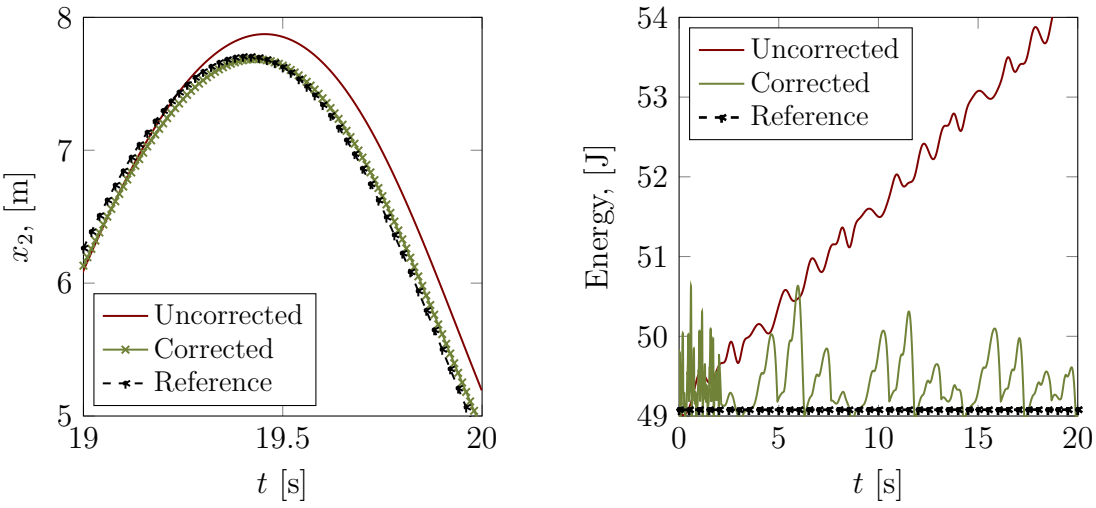


(a) Displacement η_2 of the second mass.

(b) Mechanical energy.

Figure 5.45: Nonlinear oscillator, case 2: Displacement η_2 and mechanical energy in single-rate co-simulation. $H = h_{\mathcal{M}_1} = h_{\mathcal{M}_2} = 1$ ms, $\mu = 0.5$.

the indefinite accumulation of spurious energy inside the system.



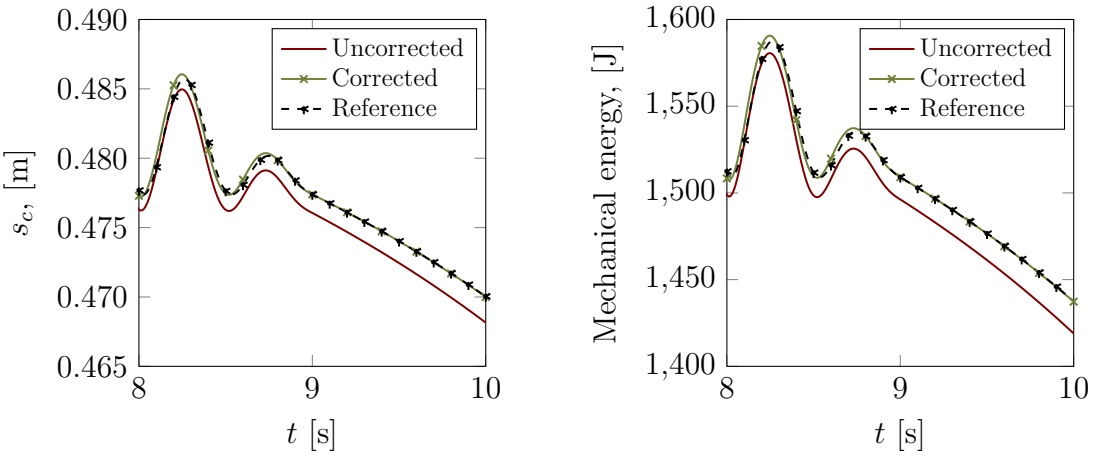
(a) Horizontal position x_2.

(b) Mechanical energy.

Figure 5.46: Two-pendulum example: Coordinate x_2 and mechanical energy in single-rate co-simulation. $H = h_{\mathcal{M}1} = h_{\mathcal{M}2} = 1$ ms, $\mu = 0.5$, $k_\mathrm{i} = 1.0$.

The last benchmark example, the hydraulic crane in Section 5.4.4, was used to test the energy correction method in the multi-rate simulation of a multiphysics system. As mentioned in the previous Section, this example receives a supply of energy from the hydraulics circuit which is not evaluated or known by the co-simulation manager. It must be noted that, due to the significant dissipation introduced at the hydraulic cylinder, the deviations of the co-simulated dynamics with respect to the reference solution remained relatively small in terms of both motion and energy,

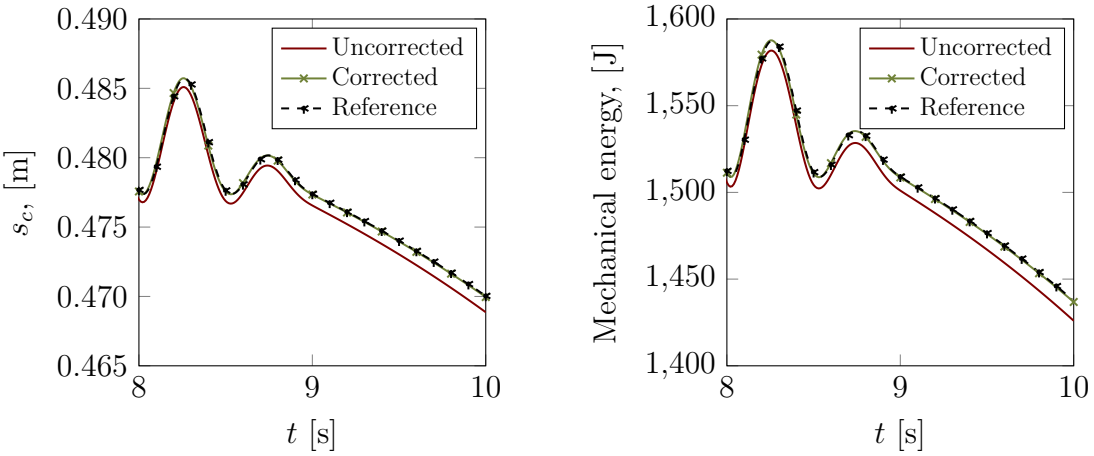
as long as the macro-steps of the subsystems, $h_{\mathcal{M}}$ and $h_{\mathcal{H}}$, remained below their instability thresholds.



(a) Actuator displacement s_c.

(b) Mechanical energy.

Figure 5.47: Hydraulic crane: Cylinder displacement s_c and mechanical energy in multi-rate co-simulation. $H = h_{\mathcal{M}} = 10$ ms, $h_{\mathcal{H}} = 0.25$ ms, $\mu = 0.5$, ZOH extrapolation.



(a) Actuator displacement s_c.

(b) Mechanical energy.

Figure 5.48: Hydraulic crane: Cylinder displacement s_c and mechanical energy in multi-rate co-simulation. $H = h_{\mathcal{M}} = 10$ ms, $h_{\mathcal{H}} = 0.25$ ms, $\mu = 0.0125$, FOH extrapolation.

Figures 5.47 and 5.48 show the results obtained with ZOH and FOH extrapolation, respectively, for $h_{\mathcal{M}} = 10$ ms and $h_{\mathcal{H}} = 0.25$ ms, which corresponds to a step-size ratio $\mathcal{R} = 1/40$. These values are slightly below the step-sizes that would lead to significant deviations from the reference, or even the instability of the simulation, as in Fig. 5.36, in which $h_{\mathcal{M}} = 11.5$ ms. Under these circumstances, the actuator displacement follows closely, within a 2 mm-margin, the reference motion.

Unlike in the previous examples, coupling errors at the discrete-time interface removed energy from the system, instead of increasing the energy level. Both these deviations can be corrected with the use of Eq. (5.48). It was not possible, however, to render the simulation stable for larger values of the subsystem macro step-sizes.

5.7 Conclusions

Co-simulation and monolithic simulation are two complementary approaches to simulate the behaviour of dynamical systems. Co-simulation involves dividing the application under study into a set of subsystems that evolve separately and exchange data between them at discrete points in time; the consistency of the simulation must be enforced by a co-simulation manager, which synchronizes the integration of the dynamics. The use of co-simulation enables the use of dedicated solvers and appropriate time scales for each subsystem, and protects intellectual property by hiding implementation details behind a coupling interface that only reveals to its environment a set of minimal variables. Thus, it emerges as an advantageous tool that allows to simulate in an efficient way virtual systems, with components of disparate natures, using models implemented in different software tools by several vendors, and without violating intellectual property rights. However, selecting all the appropriate configuration options to make a co-simulation process stable and truly representative of the behaviour of the system that describes is not straightforward. It often requires nontrivial knowledge about the dynamics of the involved subsystems and the computational hardware to select the most adequate co-simulation configurations.

In cyber-physical applications, like MBST test benches, real components are interfaced to a computer simulation via the exchange of a reduced set of magnitudes. This is a particular case of RT co-simulation, in which the co-simulation manager has to deal with the synchronization of physical and virtual components. Co-simulation is, therefore, a necessity to communicate the subsystems to each other by means of a compatible framework. It can also be employed to enable the parallel simulation of virtual systems in a RT environment.

The use of explicit schemes is common in RT applications and mandatory for most CPS, in which iterative solutions that require retaking macro steps are not allowed. These explicit schedules tend to modify the energy balance because of the errors caused by the discrete-time communication at the co-simulation interface, in particular those associated with input extrapolation. Diverse strategies have been proposed to develop indicators that can eliminate or minimize the errors in certain co-simulation schemes. Energy-based indicators, like the power and energy residuals used in this Chapter, can be used to keep track of the spurious energy introduced at the coupling interface and to provide warnings about unstable or unreliable simulations during runtime. These can be evaluated directly from the information carried by the coupling variables alone when the product of the subsystem inputs and outputs has units of power. If the co-simulation coupling fulfils this requirement, this family of indicators can also be used to develop correction methods to prevent energy deviations in the simulation.

The implementation and use of power and energy residuals, together with an en-

ergy correction method based on their evaluation, were demonstrated in this Chapter with a set of benchmark examples, easy to define and implement but representative of the issues that exist in co-simulation setups. Results were obtained in the single-rate and multi-rate explicit Jacobi co-simulation of these examples and confirmed the validity of the proposed indicators and correction technique. The automated tuning of the parameters of the correction method requires further evaluation and represents an open avenue for future work.

Chapter 6

Implementation: building a model-based test bench for electric motors

This Chapter illustrates how the MBST technologies discussed in Chapters 3 – 5 come together in an example of cyber-physical application for model-based system testing. These technologies are necessary for the correct assembly and operation of the application selected here as demonstrator: a CPS test bench for electric motors. Its virtual components need to be developed using RT-capable models and formulations; PISE is a critical building block to evaluate magnitudes of interest when direct information of them is not available; co-simulation techniques, finally, are necessary to interface real and virtual components in the setup, integrating them separately and synchronizing their dynamics via the exchange of a limited data set at particular points in time.

The proposed test bench intends to enable the testing of e-powertrain components for automotive applications, in particular electric motors. Their evaluation in the controlled and repeatable environment created in a MBST testing facility is beneficial for the systematic gathering of experimental data and makes it easier and safer to reproduce dangerous situations such as emergency manoeuvres. This approach would not be simple, or even feasible to execute, if a prototype-driven strategy were to be followed. For the particular case of electric motors, the evaluation of their thermal response is of interest for the design and improvement of efficient control algorithms that make the most of the component capabilities without compromising their safe operation, avoiding excessive temperatures that could damage them.

This Chapter focuses on the design, construction and operation of two test benches in which physical components interact with virtual environments. The main test bench is aimed at testing automotive-grade electric motors to evaluate their response under demanding driving conditions. These trials allow one to gain insight into the performance of the motors in real-world working conditions and to enhance their global performance in a safe and repeatable way. A secondary test bench was planned and executed as a prototype of the main application and also as demonstrator of its operation and principles. This prototype bench provided a

first contact with an important share of the problems that may appear during the setup and operation of the full-size test bench. For this scaled-down application, low-power motors have been used, which also minimized the risk and severity of accidents and damaging the equipment.

6.1 Introduction

In recent years, due to environmental issues related to pollution, increment of diseases in the big cities or climate change, the process of electrifying the transport has gained popularity in the automotive industry. The most important carmakers have moved their combustion engine roadmaps towards electric-powered vehicles, backed by the plans of the governments in the majority of the industrialized countries. However, the transport electrification could not be possible without recent developments on Lithium-ion batteries and power electronics, which have enhanced the range and performance of the new generation of electrified vehicles [176].

Transitioning from combustion engine powertrains to e-powertrains requires the building of new test benches to replace the existing ones, which had been originally designed for internal combustion engines. Replacing the current test benches with new ones for e-powertrain components represents, in fact, an opportunity to take advantage of recent technological developments that include enhanced simulation capabilities and advanced data gathering and processing solutions. These make it possible to consider simultaneously several phenomena that take place in e-powertrain components and evaluate how they affect the behaviour of the different elements, their interactions, and also their overall performance. For instance, the electric and electronic effects in motors and inverters in an e-powertrain are closely related to their thermal behaviour, and all of them interact directly with the way in which the vehicle is driven and its dynamics. The simultaneous consideration of these phenomena and their RT simulation is nowadays possible in the form of multiphysics environments. In MBST facilities, these can be interfaced to physical components, and replace the whole vehicle prototype that would be required to test them in a conventional experimental setting. As mentioned, the use of MBST approaches eases and speeds up the test campaigns, shifting the detection of design and performance issues towards early development stages. The possibility of subjecting the components under test to repeatable testing conditions is also an attractive feature of MBST solutions. For instance, different driving profiles can be loaded onto the simulation platform and used to drive the virtual vehicle, thus enabling the comparison of different e-motor control algorithms with identical input signals.

6.1.1 Overview of the proposed test bench

The research plan of this thesis envisions the building of a MBST test bench for automotive-grade electrical motors, which will make it possible to evaluate the theoretical developments presented in Chapters 3 – 5 in an environment comparable to those used in industry. The test bench is an example of cyber-physical application,

which requires co-simulation algorithms to orchestrate the information exchange between its different elements. Furthermore, its virtual components are required to achieve RT performance in the selected hardware and software implementation, i.e., it is mandatory to guarantee a RT co-simulation under any circumstances. In addition, the test bench can also be used to evaluate the ability of PISE algorithms to fuse simulation results with sensor readings to estimation relevant system magnitudes during runtime.

Moreover, the availability of an MBST test bench opens up the possibility to acquire in-depth knowledge about e-powertrain operation, especially about the electric motors used in vehicles, such as the PMSM widely used in automotive applications.

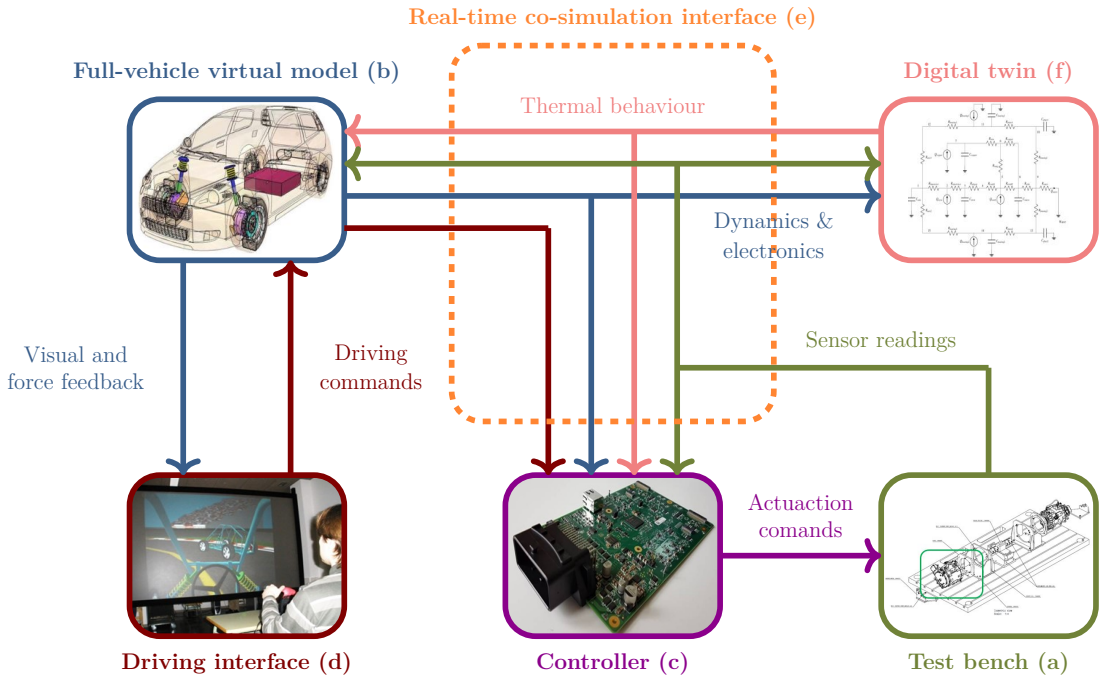


Figure 6.1: Elements of the proposed cyber-physical test bench for e-powertrain motors.

Figure 6.1 shows a conceptual overview of the proposed cyber-physical test bench for e-powertrain motors. The motor under test is interfaced to a full-vehicle MBS dynamics simulation, which may reproduce a pre-defined manoeuvre or follow the commands of a human or virtual driver. The simulation determines the loads that the motor under test would have to bear during operation, and these are exerted by means of a second electric motor, in a back-to-back configuration [177]. Moreover, a DT of the tested motor is used to gain insight into the information provided by the sensors mounted on the system and monitor its behaviour beyond directly available measurements. The information gathered by the DT is processed to develop a more accurate model of the e-motor, which can be used to take into consideration effects like component ageing or damages undergone during operation.

The back-to-back configuration has been presented in the literature [27, 42, 177, 178] as an efficient manner to test e-powertrains of vehicles. It consists in a two-motor assembly, in which one is the motor being tested, and the second acts as

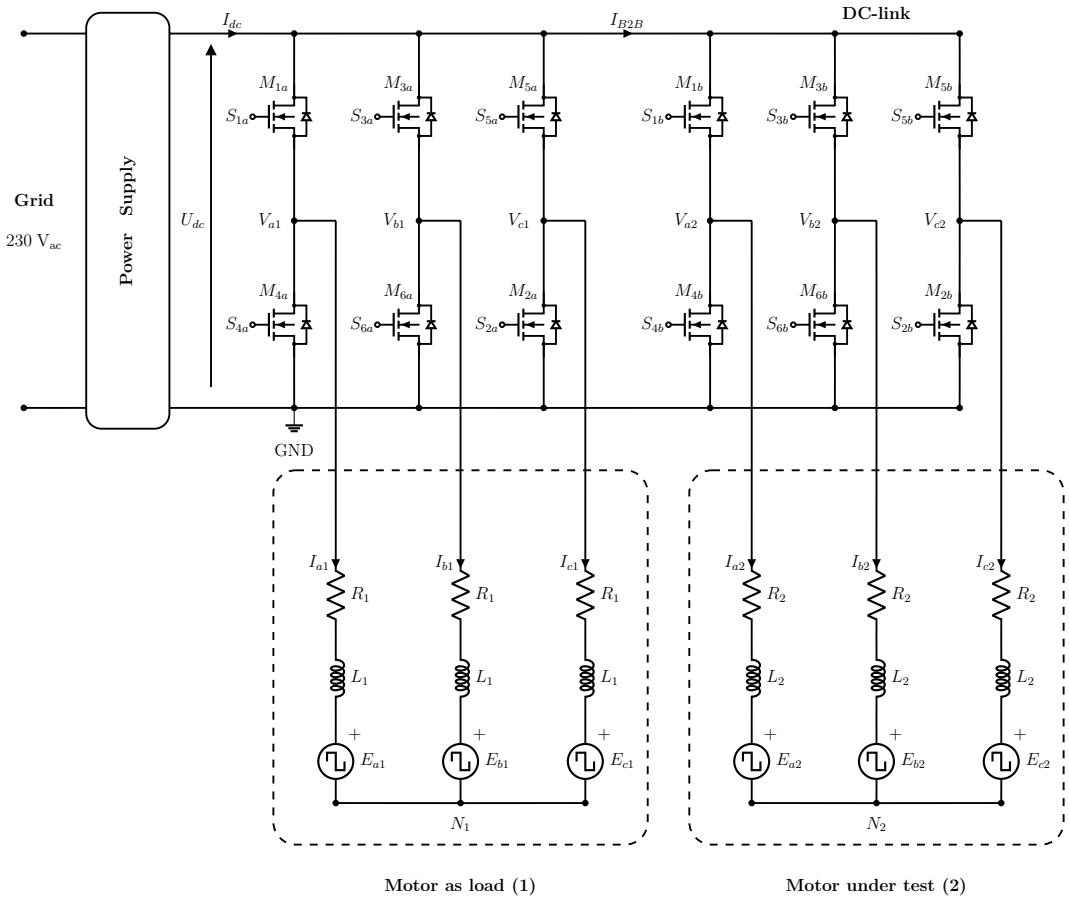


Figure 6.2: Example of back-to-back configuration for two motors in a test bench.

a regulator to keep the first in the desired operation regime defined as a pair of torque-speed values (τ, ω), where τ denotes the torque developed by the motors, and ω the angular speed of the shaft. This is achieved through a physical coupling between the shafts of both motors, in order to transmit mechanical power through it. The configuration is energy-efficient, because one of the motors behaves as a generator that provides part of the power required by the other. The electrical grid only needs to provide the power necessary to modify the operation conditions, e.g., during acceleration, and to compensate the losses of the components in the setup. Depending on the operation point, the generation may occur at any of the motors in the assembly.

Figure 6.2 represents an example of the electrical scheme of a test bench following a back-to-back configuration. In particular, the figure shows an operation point in which the motor on the left (*motor as load*) operates in a generator regime providing power to the DC-link. The motor on the right (*motor under test*), in turn, consumes the current from the DC-link provided by the power supply and the other motor. Another point of operation, in which the power evacuated by the motor as load is larger than the consumed by the motor under test, is also possible. This excess power needs to be dissipated somehow, for instance by means of a *braking resistor* or injected back to the electrical grid.

Designing and building a full-size test bench for electric motors is a complex

project that involves aspects from mechanical, electrical, thermal, and control engineering. The use of automotive-grade motors, besides, brings in at least three big issues. First, the electric motors operate at energy levels that may cause damage to the rest of physical components in the setup or even the operators; they can be rightly considered dangerous equipment. Second, the installation of all the systems required for their operation is a rather involved procedure, in which some issues may only become apparent in later stages of the building process. Third, the algorithms developed during this work, which are necessary to connect real and virtual subsystems and keep track of relevant magnitudes, require a tuning process during which malfunctions are relatively likely to occur. These reasons made it advisable to build a secondary bench prototype for low-power motors, that mimics the configuration and operation principles of the main facility, and can be used as preliminary test and learning platform. The results obtained with the prototype can be used to gain experience and increase the safety level of the operation of its full-size counterpart.

6.2 Prototype test bench

The layout of the prototype test bench is the same as that of the full-size testing facility, shown in Fig. 6.3; it uses a back-to-back configuration as well. Besides its role as preliminary test platform, this reduced-scale bench can also be considered a demonstrator and learning tool, to train new users in the operation of CPS testing equipment.

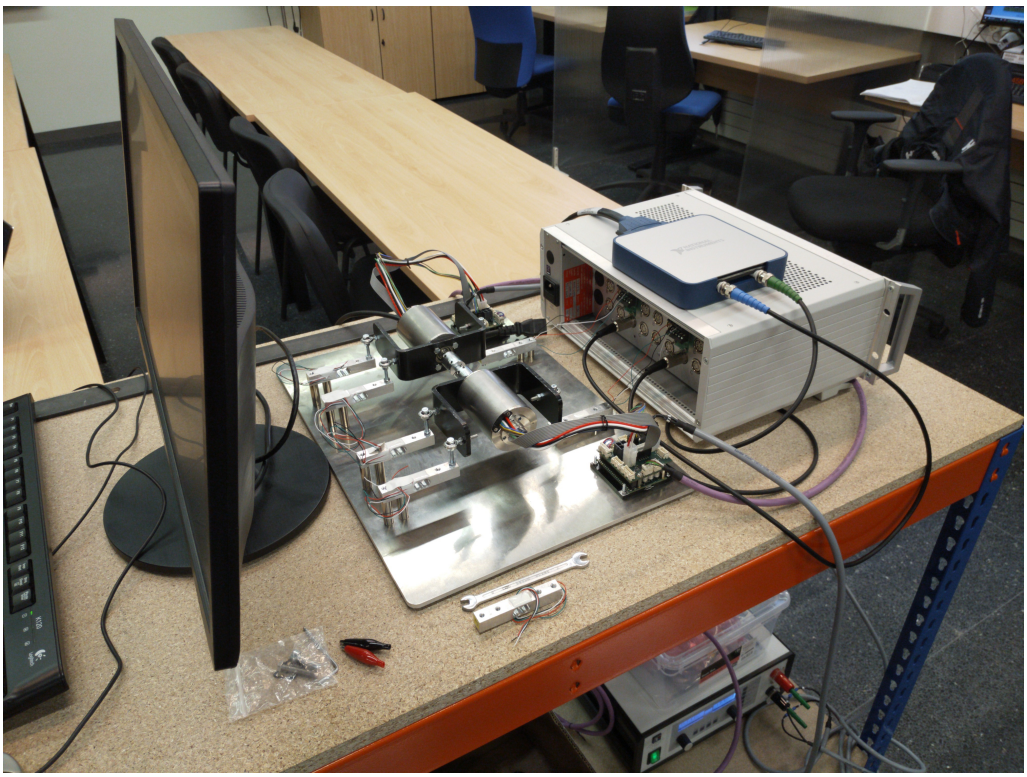


Figure 6.3: Prototype test bench for low-power motors.

6. Implementation: building a model-based test bench for electric motors

In its current configuration, the prototype bench mounts two Maxon EC-i 52[1] motors, whose rotational motion is mechanically coupled at their shafts. Potential misalignments between the two axes of rotation are corrected using two flexible couplings to prevent the appearance of high-frequency vibrations during operation. The motors are commanded by two ECUs Maxon EPOS4 Module 50/15[2] controllers, connected through CAN. The motors can be operated in four different modes: position, speed, torque, and current. The aforementioned components are powered by a programmable power supply [3]. The main electrical and mechanical parameters of the motors are summarized in Table 6.1.

Table 6.1: Electrical and mechanical parameters of the motors in the prototype bench.

DC-link voltage	U_{dc}	35 V
Nominal power	P_{nom}	180 W
Nominal speed	ω_{nom}	4200 rpm
Nominal current	I_{nom}	8.81 A
Maximum speed	ω_{max}	4680 rpm
No load current	I_0	726 mA
Phase resistance	R	44.7 mΩ
Phase inductance	L	61 μH
Voltage constant	k_B	5.076 mV/rpm
Torque constant	k_τ	48.6 mNm/A
Rotor inertia	J_R	170 gcm^2
Pair of poles	p_p	8

Sensor measurements in the prototype include current, voltage, position, angular speed, and torque readings. The input voltage is directly measured and returned by the power source. This value is intended to remain constant during operation. The ECUs implement current, position, and angular speed sensors to provide the feedback to the control loop. These measurements are retrieved to determine the operation point of the motors. The motor torques can, in principle, be calculated from the motor parameters and equations. However, in order to retrieve the torque of the operation point in a way that is closer to the one used in the full-size test bench, torque sensors for each motor were built using load cells. This required modifications in the design of the motor brackets, which are connected to the ground by means of ball and universal joints to ensure that only shear efforts are transmitted to the load

[1] https://www.maxongroup.com/maxon/view/product/motor/ecmotor/EC-i/516068

[2] https://www.maxongroup.com/maxon/view/product/control/Positionierung/520886

[3] https://elektroautomatik.com/shop/en/products/programmable-dc-laboratory-power-supplies/discontinued-series/series-ps-8000-dt-br-320-w-up-to-1-5-kw/684/laboratory-power-supply-0..80v/0..40a

cells. The electric signals from the cells are sent to a signal conditioning system[4] and later passed to a data acquisition card NI PCIe-6363[5], which in turn feeds them as digital inputs to a PC with a RT Linux OS.

6.2.1 Communications and control scheme

The prototype test bench described in this Section can be considered a CPS in a HiL configuration. Figure 6.4 illustrates the communication layout used in this application. An equivalent scheme is used in the full-size bench, although the nature and number of the sensors used differ.

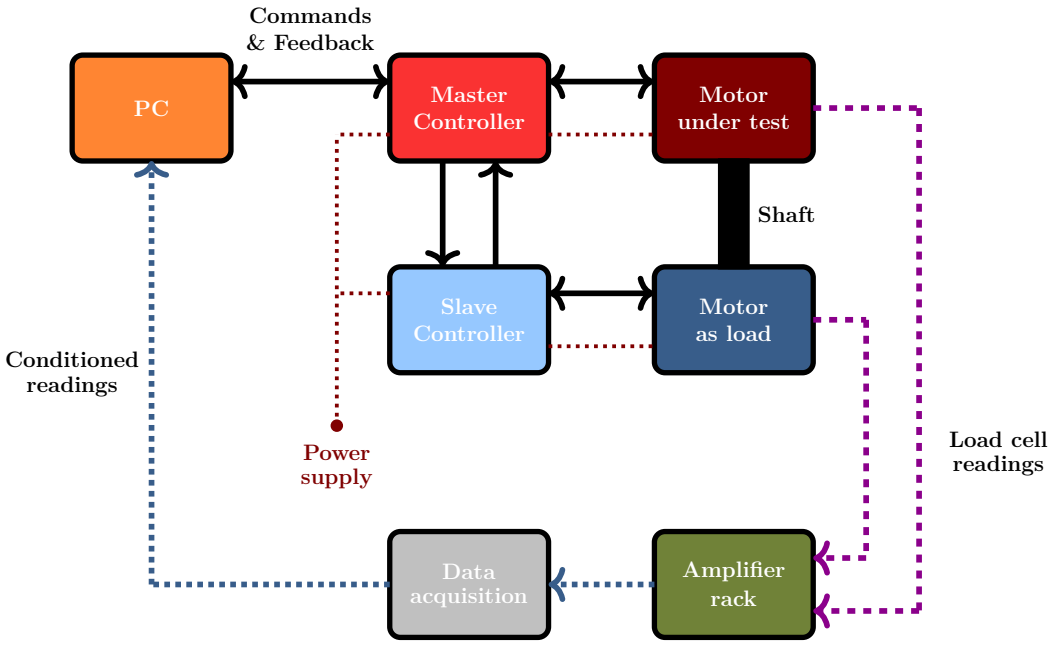


Figure 6.4: Communications layout of the prototype test bench.

The motors in the prototype bench are interfaced to a computer simulation of a system of interest, which runs on a PC. This computer is also used to control the physical motors by means of the exchange of commands via CAN bus. The motor controllers adapt the output voltage, current, and frequency that the motors receive depending on their requested working points, e.g., (τ, ω) or (I, ω). The angular position and speed of the motor shafts are measured by the motor encoders. The controllers also equip Hall effect sensors to measure the current delivered to the motors. All these measurements are sent to the PC with the rest of commands through CAN bus, and they are made accessible to simulation in the virtual environment. The torque readings, however, are evaluated by the load cells through the indirect measurement of shear efforts at the motor supports. These readings are gathered by the amplifier rack and introduced through the data acquisition system into the PC. The load cells deliver a noisy voltage signal within the range $[0, 5]$ V. The amplifier rack filters the input signals and conditions the output signal to the voltage level

[4]https://www.rdpe.com/ex/m600.htm
[5]https://www.ni.com/en-us/support/model.pcie-6363.html

of the data acquisition system before entering the PC through a PCI-Express port. The angular speed, current, and torque readings from the bench are used as feedback in the control loop of the motors, to make them match the dynamics calculated by the computer simulation. The bi-directional data exchange between the computer and the hardware components in the bench has to comply with RT constraints, i.e., the dataflow through the CAN bus has to be synchronized with the control orders from the simulation. If a misalignment between the physical components and their virtual environment were to occur, it may cause the malfunction of the test bench in extreme cases. More generally, the reliability of the obtained results would be questionable; the study of the severity of the issues caused by data transfer problems, such as noise and delays, is a topic that needs to be studied in the future.

6.2.2 Software

Throughout this research, several computational methods and solutions have been put forward to address diverse issues that arise in the context of MBST, and which were discussed in Chapters 3 – 5 of this thesis. The use of these algorithms and techniques requires their implementation as software libraries. Ideally, these should be reusable in different applications. Furthermore, they are required to be as portable as possible across hardware and software platforms, and efficient so as to comply with RT constraints if the application that uses them requires so.

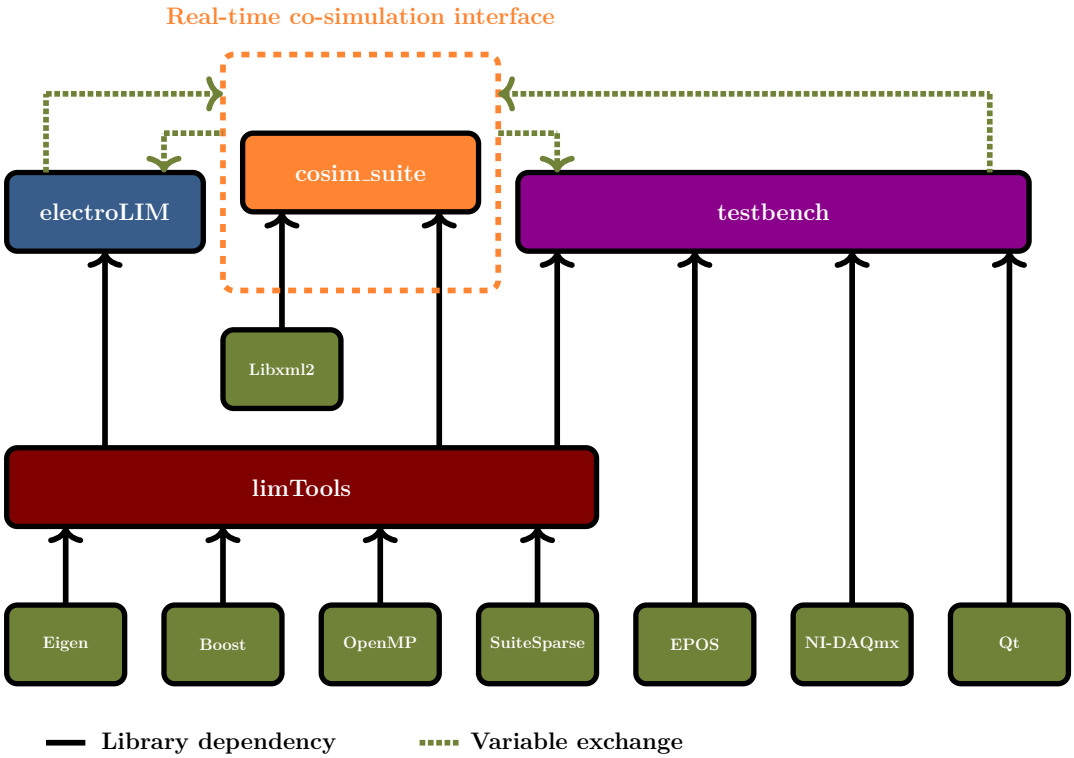


Figure 6.5: Software tools employed in the test bench.

Figure 6.5 presents an overview of the software tools used to operate the test bench, and also shows how the libraries developed during this thesis come together

in a cyber-physical application. The *electroLIM* library permits the systematic building and benchmarking of electric and thermal models of e-powertrain parts for their use in the RT models required by the DT of the physical components and the virtual environment of the bench. The co-simulation methods to couple the different physical and virtual subsystems are included in *cosim_suite*, which contains the implementation of co-simulation manager schemes and the definition of subsystem wrappers according to the FMI standard.

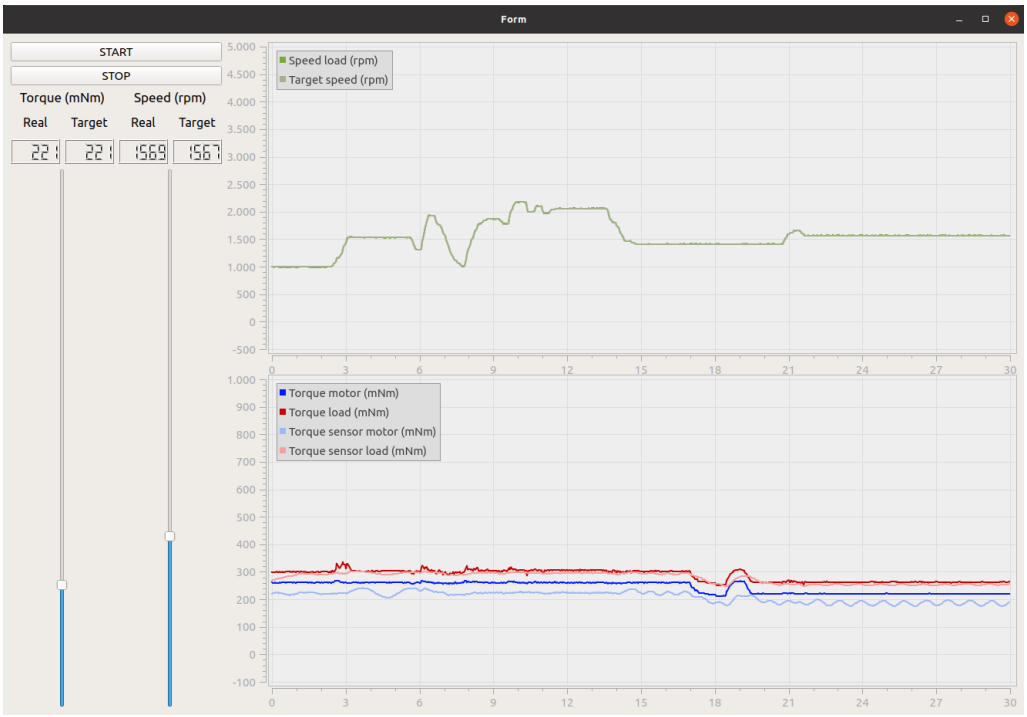


Figure 6.6: Graphical interface of the test bench included in the *testbench* software library.

The communication between the simulation environment and the bench motors is performed by the *testbench* library, which receives data from the sensors on the physical components and sends commands to the actuators, complying with the RT requirements for cyber-physical applications. The control of the motors can be performed using the EPOS library provided by Maxon, or directly via CAN commands. Besides providing a gateway between real-world components and their virtual environment, it also features a graphical interface for the monitoring and calibration of the system, shown in Fig. 6.6. This interface permits the control of the operation point of the bench and calibration of its sensors prior to its use in experimental runs.

Additional information about the above mentioned libraries and third-party software used in this project can be found in Appendix A.

6.2.3 Benchmark problems

Simple benchmark problems were put forward to perform initial tests with the prototype bench described in this Section.

The first series of tests is intended to enable the initial calibration and verification of the device. These will be performed using the bench to represent a constant rotational inertia that the motor under test will move starting from an initial zero velocity. Different values of the inertia will be tested, and results will be compared to the analytical solution of the problem. This series will also serve to verify the effect of communication aspects, such as the sampling step-size used in the virtual environment.

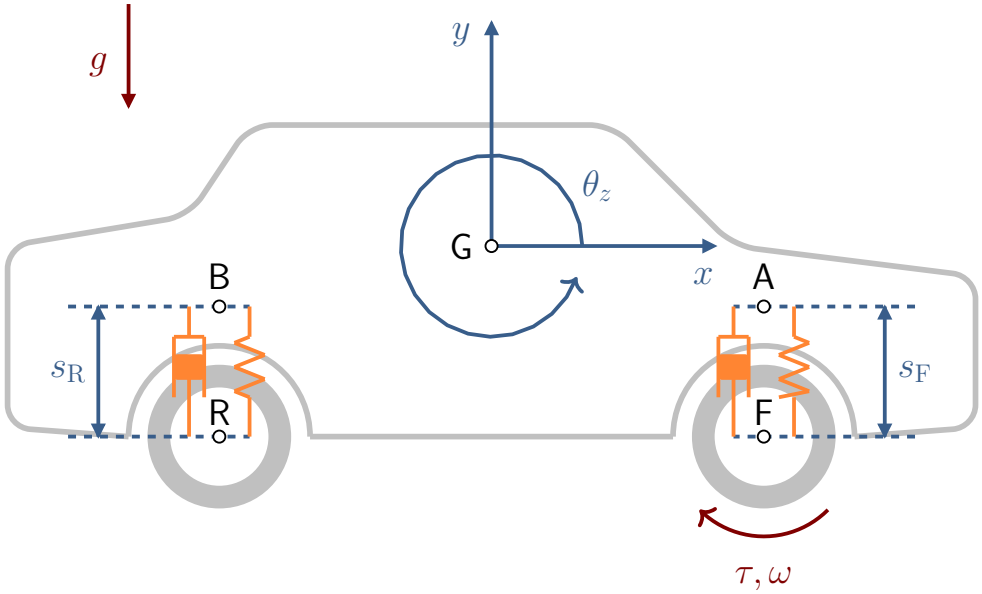


Figure 6.7: Benchmark e-Car model.

A second benchmark problem consists in a planar, three-degree-of-freedom multi-body model of a radiocontrol vehicle, actuated by the Maxon motors mounted on the bench, shown in Fig. 6.7. The MBS dynamics simulation of the vehicle is to be conducted in the virtual environment, providing the angular velocity of the wheels that serves as input for the motor that represents the load in the bench. The motor under test, which represents the motor that powers the front wheels of the vehicle, is controlled by a throttle signal from the driver and exerts a torque on the load that is fed back to virtual environment to close the simulation loop. The driver can be a human, a virtual driver algorithm, or a pre-recorded manoeuvre.

The vehicle motion can be described by means of five generalized coordinates

$$\mathbf{x} = \begin{bmatrix} x_\mathrm{G} & y_\mathrm{G} & \theta_z & s_\mathrm{F} & s_\mathrm{R} \end{bmatrix}^\mathrm{T} \tag{6.1}$$

where x_G, y_G are the global coordinates of the centre of mass G of the vehicle, θ_z stands for the rotation of the car body with respect to the horizontal x-axis, and s_F and s_R are the length of the front and rear suspension, respectively. Points F and R represent the centres of the wheels and they are assumed to remain at a constant distance from the ground, i.e., the wheel radius r. A continuously maintained rolling contact is assumed between the wheels and the ground. The vertical distance between G and the upper ends A and B of the suspensions is h. The centre of mass G is assumed to be located halfway along the vehicle x-axis between points A and

B. Moreover, points A and B always remain vertically aligned with F and R, so changes in the suspension lengths s_{F} and s_{R} only affect the vertical coordinate of these points. Under these assumptions, the vehicle has three degrees of freedom, so two kinematic constraints need to be imposed on $\mathbf{x}$ to correctly represent the vehicle motion:

$$\mathbf{\Phi} = \begin{bmatrix} s_{\mathrm{F}} - s_{\mathrm{R}} - l_{\mathsf{AB}} \sin\theta_z \\ s_{\mathrm{R}} + l_{\mathsf{BG}} \sin(\theta_F + \theta_z) - y_{\mathsf{G}} \end{bmatrix} \tag{6.2}$$

where l_{AB} and l_{BG} stand for the distances between points A – B and B – G, respectively, and θ_F is a constant value that represents the angle between the local x-axis of the vehicle and line B–G. Table 6.2 contains the parameters of the scaled e-Car benchmark problem.

Table 6.2: Mechanical parameters of the e-Car.

Distance between wheel centres	l_{AB}	180 mm
Local y distance from G to A and B	h	10 mm
Wheel radius	r	40 mm
Mass of car body	m	2 kg
Gravity	g	9.81 ms^{-2}
Suspension stiffness, front	k_{F}	$3 \cdot 10^3$ Nm^{-1}
Suspension stiffness, rear	k_{R}	$3 \cdot 10^3$ Nm^{-1}
Suspension damping, front	c_{F}	10^2 Nsm^{-1}
Suspension damping, rear	c_{R}	10^2 Nsm^{-1}
Natural length of spring, front	s_{F0}	40 mm
Natural length of spring, rear	s_{R0}	40 mm
Rotational inertia of main body	J_z	$m l_{\mathsf{BG}}{}^2$ kgm^2

The dynamics equations of this system can be formulated by means of conventional MBS dynamics formulations, e.g., an augmented Lagrangian method [146, 179], and integrated with explicit or implicit integration formulas. The size and complexity of the model does not pose a problem to achieve RT performance in most computational platforms of interest.

In a future stage, realistic three-dimensional vehicle models, e.g., [180], will be coupled to the bench hardware to deliver a HiL driving simulator.

6.2.3.1 Coupling scheme

From the point of view of co-simulation, the test bench follows the coupling scheme shown in Fig. 6.8. The virtual environment sends the angular velocity ω that results from the simultion of the multibody system dynamics to the electric motors in the test bench through the co-simulation framework. Specifically, the motor that acts as load is the one that receives this velocity command. The motor

under test sends back the torque τ that it exerts to the co-simulation environment. It is worth mentioning that the motor under test also receives a throttle command, which determines the torque that it exerts on the load that it is moving.

Figure 6.8: Co-simulation diagram of the prototype test bench, arranged following a torque-angular speed coupling scheme.

In Fig. 6.8, $\mathcal{M}$ and $\mathcal{E}$ stand for the multibody virtual model and the physical electric motors mounted on the bench, respectively. In this co-simulation scheme, the product of the coupling variables has power units, therefore the energy correction method described in Section 5.6 could be used to regulate the energy level of the experimental setup.

6.3 Automotive-grade test bench for e-powertrain components

The prototype test bench in Section 6.2 is a scaled-down model of the full-size facility presented at the beginning of this Chapter, aimed at the MBST evaluation of automotive-grade electric motors and other e-powertrain components. It is currently under construction, as shown in Fig. 6.9; upon completion, it will be used to test, in a back-to-back configuration, permanent-magnet three-phase motors of up to 200 kW of power, 550 A_{rms} at each phase, maximum angular speeds of 18,000 rpm, and able to deliver an output torques of 400 Nm.

Figure 6.9: Test bench (in progress) for automotive-grade motors and other e-powertrain components.

The motors for this application have been provided by GKN Driveline Zumaia[6] and are compatible with the limits of this test bench. These will be commanded with

[6]https://www.gknautomotive.com/

inverters designed and manufactured by Cascadia Motion, in particular model PM 150DX[7]. Motors and inverters are instrumented and are able to provide information about the temperatures in some internal points, although not in crucial points of the motors, such as magnets or stator winding ends. These measurements, however, will be of help to validate the LPTM models of these e-powertrain components. Additional sensor readings include a commercial torque sensor that will be installed at the connection between the shafts of the motors, and a wattmeter. These will provide precise measurements of the torque at the mechanical shaft and the power consumed by the motor under test and its inverter, thus enabling the determination of the motor performance at every tested working condition.

Figure 6.10 provides a simplified illustration of the proposed communications layout for this bench, in which the aforementioned sensors are connected to the central PC by means of a data acquisition system.

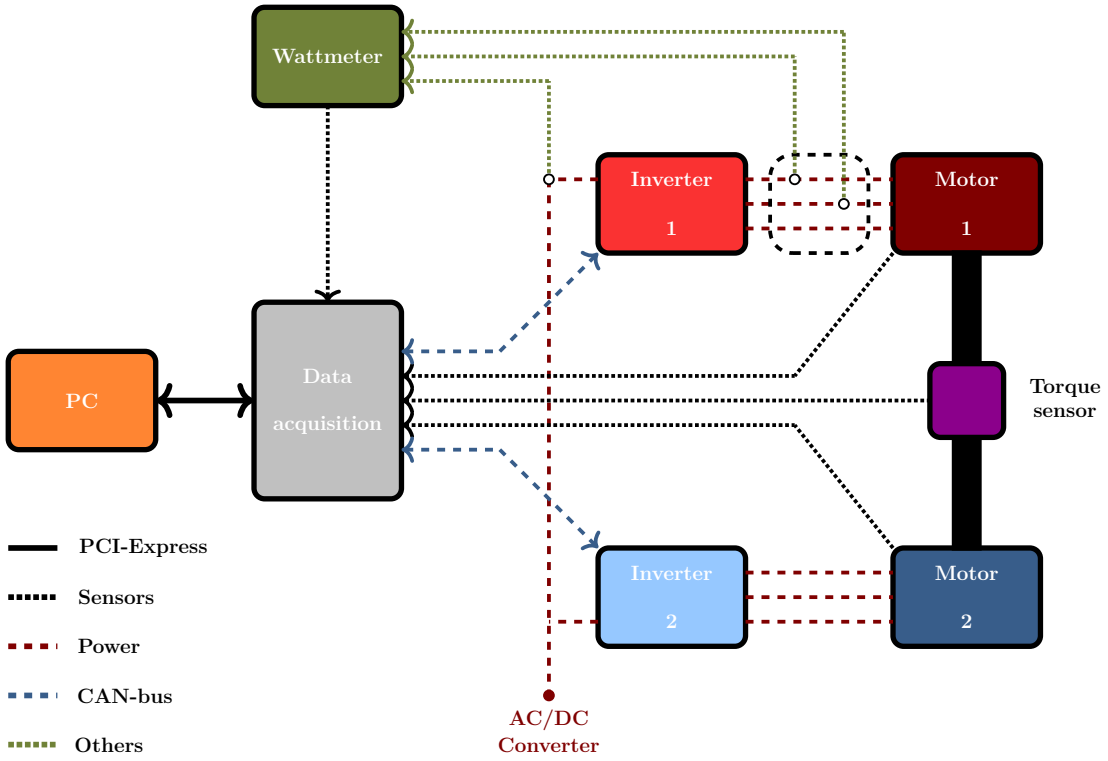


Figure 6.10: Communications layout of the automotive test bench.

The power supply of the assembly is provided, in this case, by an AC/DC converter manufactured by Nidec, model M700[8]. This converter is connected to the grid by means of a 54-kW wye-wye transformer, which provides the adequate input voltage level for the desired working voltage range between 270 and 400 V. Due to electrical grid restrictions, it is not possible to inject the excess power again into the grid. Thus, this surplus will be dissipated in a 3-kW braking resistor. The electric power consumption of this test bench has made it necessary to adapt the present

[7]https://www.cascadiamotion.com/productlist/8-inverters/pm-inverters/2-pm150

[8]https://acim.nidec.com/es-es/drives/control-techniques/products/ac-drives/unidrive-m/unidrive-m700

electrical installation of the building to accommodate the power requirements of the operation of this high-power test bench. Figure 6.11 contains a schematic diagram of the electrical installation.

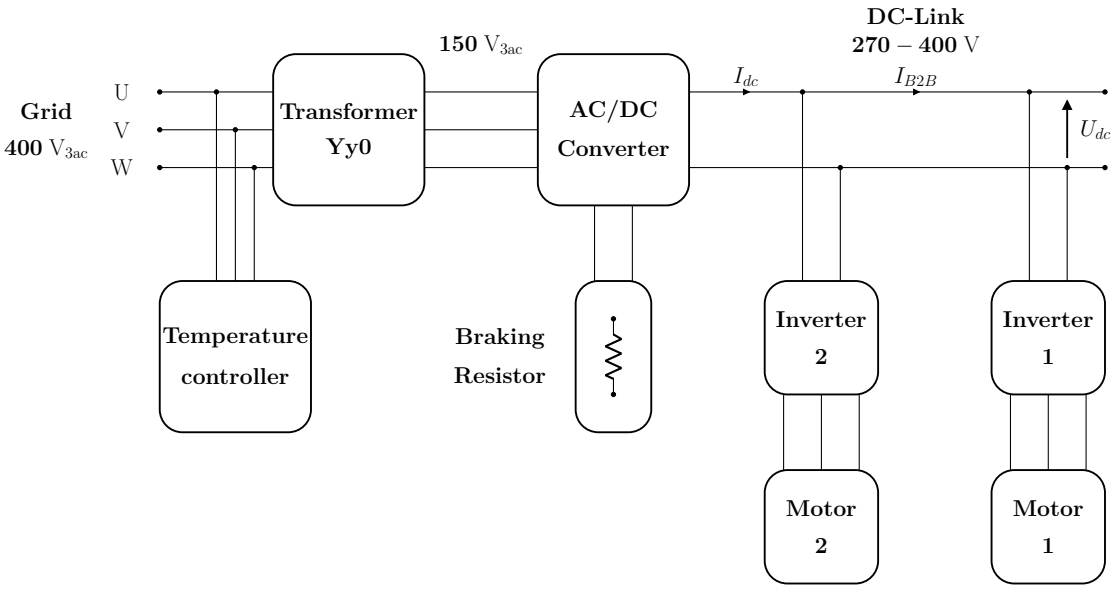


Figure 6.11: Layout of the automotive test bench power supply.

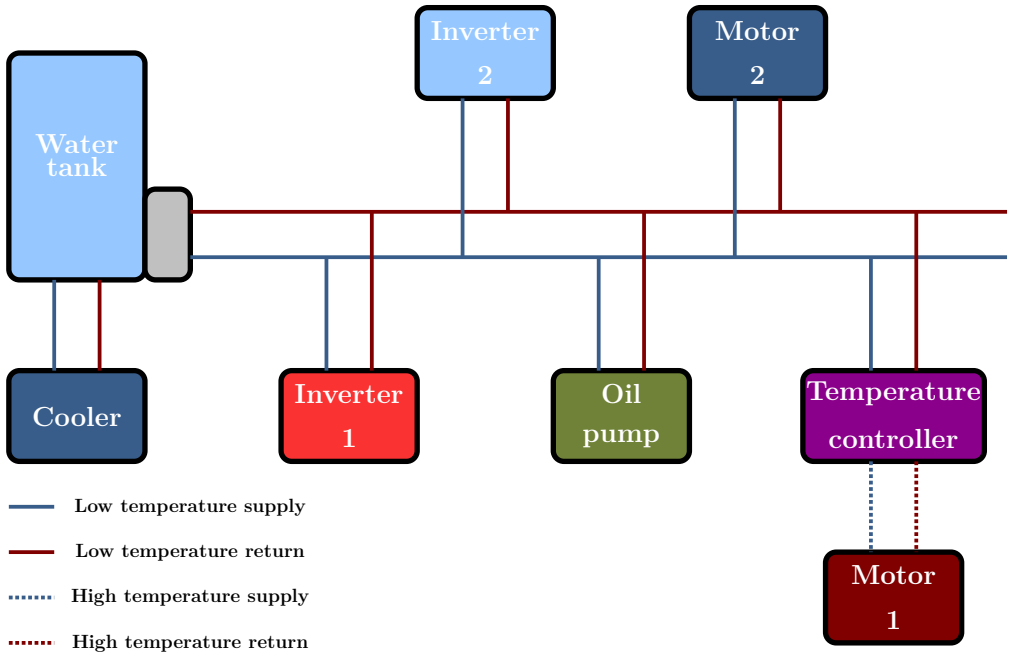


Figure 6.12: Schematic layout of the cooling system for the automotive test bench.

Refrigeration is an additional system that the prototype test bench did not need but its full-size counterpart requires. Electric motors used in automotive applications tend to feature reduced Joule effect losses in a wide range of their working conditions. However, their compactness hinders an efficient refrigeration of the most

critical parts; only a reduced and superficial area of the motor housing is able to dissipate the generated heat to its environment. Thus, the cooling system has to be able to keep the motor and inverter temperatures within an appropriate range during the test cycle, which sometimes represents a challenging task. This cooling is necessary to prevent motors from suffering permanent damages derived from excessive temperature, such as demagnetization. Figure 6.12 represents the cooling system for this test bench. It consists of a 2000-litre water tank with a heat exchanger and accessories, a 9-kW indoor temperature controller, a 10-kW outdoor cooler, and the pipe connections (supply and return), for the purpose of refrigerating both inverters, motors and an oil pump responsible for the lubrication of the motor bearings. The temperature of the refrigerant is often kept in the range $295 - 305$ K. However, the motor under test needs to perform in thermal conditions that are representative of its service operation. For this reason, it is cooled through a temperature controller that serves as heat exchanger between the motor 1 cooling subsystem and the rest of the cooling system. The temperature controller is intended to keep the same temperature than in a car cooling system, usually in the range $330 - 350$ K.

6.4 Future work

At the moment of concluding the present thesis, the industrial test bench described in the previous Section, is still under construction and, for this reason, it has not been possible to present results associated with it. However, upon the completion of its assembly and an initial calibration and validation stage, this facility is expected to be used to carry out diverse test campaigns aimed at determining the thermal behaviour of automotive motors.

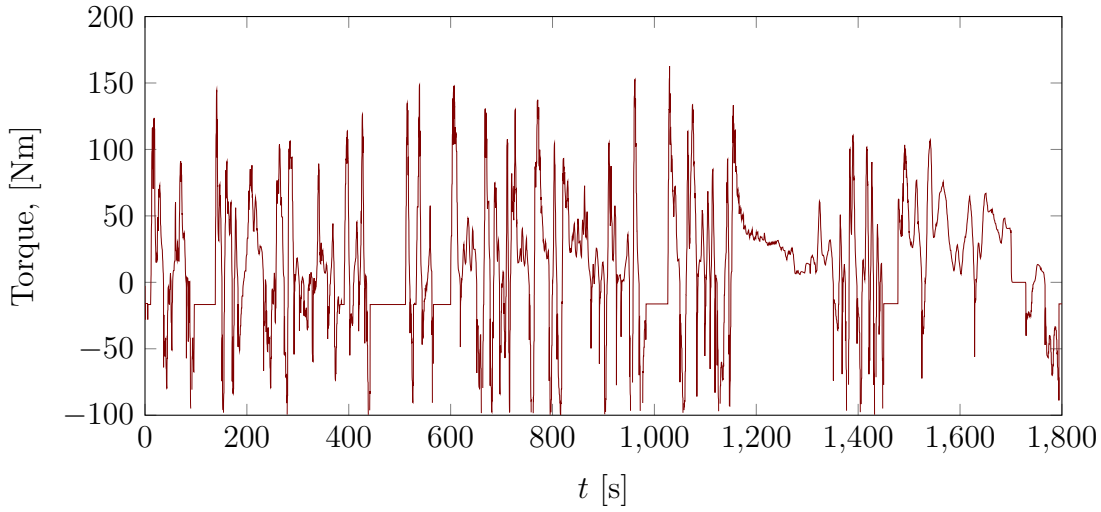
Two main test programmes are currently envisioned:

- *Standard driving cycle:* this is a harmonized test to ratify motor consumptions. Nowadays, the World harmonized Light-duty vehicles Test Procedure (WLTP) cycle [181] is the official standard in Europe. It consists in a 30-minute cycle during which the motor under test operates in both load and motor regimes. Figure 6.13 shows a speed profile that can be employed in the test, together with the torque necessary to attain it with a particular vehicle.

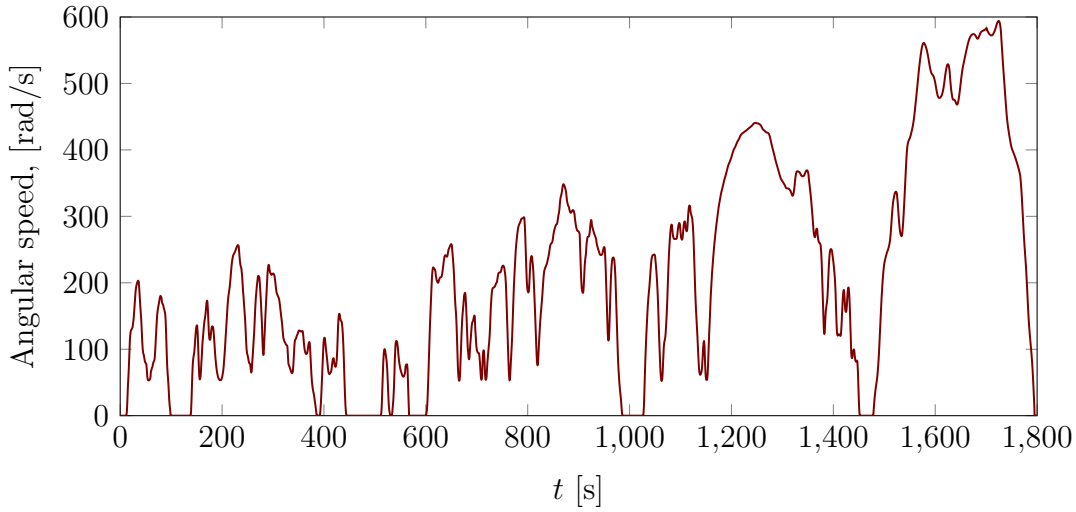
- *Continuous duty:* this is a short test in which the motor is made to operate under a specific working point defined by the angular speed of the shaft and the motor torque. These trials are commonly demanding for the motors, although their duration is limited to 5-10 minutes to prevent them from suffering permanent damages.

These tests are intended to provide details about how the operation history of the electric motors affects the evolution of their temperatures. At the same time, they will serve to validate the simplified thermal models (LPTMs) used to represent them in RT applications, and the estimation algorithms employed to approximate the temperatures at critical locations.

Besides, the proposed tests will also allow to step through a deeper understanding of the issues addressed in this thesis by means of the application of the methods



(a) Torque.



(b) Angular speed.

Figure 6.13: Example of a standard driving cycle for automotive electric motors.

that have been put forward to a cyber-physical testing facility, which represents a challenging and still open research task of industrial interest.

Chapter 7

Conclusions

This Chapter summarizes the work developed during this thesis and presents its conclusions. In addition, open lines for future research are also introduced.

7.1 Conclusions

MBST is an emerging product development cycle paradigm in the manufacturing industry that is gaining popularity, especially in automotive applications. A key element of this paradigm is the mutual interaction between physical experimentation and computer simulation. This thesis focused on three technologies related to MBST: RT system modelling and simulation, generation of DTs and use of PISE methodologies, and co-simulation, especially in cyber-physical applications.

The use of electric, electronic, and thermal models in the simulation of automotive components has contributed to a better understanding of nonmechanical phenomena in automotive components. This knowledge becomes more complete when the interactions between different components are considered from a system-level point of view. This requires the definition of multiphysics models; moreover, in MBST environments, such models need to interact in RT with physical components. This makes it necessary to evaluate their RT performance in multiplatform frameworks. This task was addressed in Chapter 3, in which a framework was developed to evaluate and benchmark the RT capabilities of implementations of electric, electronic, and thermal circuits. Besides, two new solution methods for these problems were put forward, and numerical issues with commonly used integration formulas were identified and addressed.

Chapter 4 focused on the concept of DTs and a possible implementation and usage of this technology in MBST developments. DTs consist of a physical system, its virtual counterpart, and a two-way communication to each other that makes it possible the processing of the information exchanged between them. In this Chapter, a procedure for building DTs was presented and tested by means of two different benchmark examples. The use of PISE techniques based on KFs was incorporated to the DT definition, in order to estimate the model inputs, parameters, and states both at the initialization stage and during runtime, using to this end the available measurements from the real part of the DT. Thus, through the combination of PISE techniques and DTs, it is possible to determine magnitudes which could not

be measured directly by sensors located in the real system. Along these lines, the initial scope of the framework developed in Chapter 3 was extended to fit this novel method in it.

Co-simulation is also an essential technology in MBST that allows one to split a given system into diverse subsystems that are integrated separately. At discrete points in time these systems exchange a reduced set of variables, used to synchronize the integration processes inside each subsystem. Cyber-physical applications, in which real and virtual components are part of the same system, require the synchronization of the dynamics of their different elements through a discrete-time interface as well, and can be considered a particular case of RT co-simulation. In practical applications, synchronizing the integration of the dynamics of subsystems with a wide variety of physical behaviours and time-scales is a challenging task, which can be addressed by means of a large array of schemes and implementations. In general, however, explicit coupling schemes are used. These may modify the real behaviour of the system, introducing artificial energy or damping into the system. This can renders the numerical integration unstable and requires to implement and use correction algorithms to keep track of co-simulation errors and eliminate them. Chapter 5 discussed and compared several widely used co-simulation configurations for RT and non-RT applications. A multiplatform co-simulation framework was also implemented to perform FMI-compliant co-simulation under RT constraints for virtual and cyber-physical applications. In order to monitor and correct the errors introduced by discrete-time interfaces in co-simulation schemes, a power error indicator and a new energy correction method were presented in this Chapter and added to the above mentioned framework.

Finally, Chapter 6 describes the design and building of a cyber-physical test bench for automotive electrical motors, under construction at the moment of writing this thesis. Automotive test benches are an interesting and challenging example of a cyber-physical application, in which some physical components of the vehicle are interfaced to a computer simulation. As a preliminary development, a low-power test bench prototype was assembled in a first stage to evaluate and validate the methods proposed in this thesis.

7.2 Research outlook

The research developed in this thesis falls within the active projects of the Mechanical Engineering Laboratory - LIM at the UDC. The aforementioned conclusions obtained in this thesis have been added to the know–how of the group, and therefore they will be used in upcoming projects. Several future work lines can identified at the conclusion of this thesis.

First, the dynamics formulation and the benchmark framework presented in Chapter 3 could serve as a platform for developing RT-capable models which will be able to run embedded in low-consumption platforms. Model-order reduction techniques are a possible avenue to improve their RT capabilities, especially in microcontrollers, e.g., those that belong to the ARM Cortex-M family.

Second, the initial tuning stage of the LPTM representation for thermal systems described in Chapter 4 features some drawbacks. The initial set of parameters

was calculated by means of an EKF, which does not always guarantee the convergence of the system parameters to a globally optimal solution for nonlinear LPTM circuits. Furthermore, the proposed tuning procedure is highly dependent of the system topology. A global optimization of the parameters by means of a sensitivity analysis could be a possible way to address the initial tuning of the LPTM model and arrive at optimal solution parameter sets.

Third, the co-simulation methods described in Chapter 5 can be applied to the simulation of both virtual-only systems and CPSs. In this thesis, explicit schemes have been used because physical components cannot come back in time to retake their steps; moreover, iterative algorithms may be incompatible with RT performance constraints. It would be interesting, however, to explore additional coupling schemes that were not discussed in detail, such as explicit Gauss-Seidel schemes, to determine their capacity to improve co-simulation accuracy and stability. The proposed energy-correction method will need to be adapted to these co-simulation setups. In fact, it would be desirable to develop a general-purpose procedure for the automatic tuning of the method coefficients. It is possible that this can be achieved via the use of techniques based on control theory. Additional open lines of research in hybrid RT co-simulation include developing corrective actions to take care of communication errors and delays between physical components, and the automatic selection of accurate and stable co-simulation schemes, a task that could be tackled using artificial intelligence algorithms.

Finally, the building and setup of the automotive test bench described in Chapter 6 is still in progress. This CPS testing facility will serve to gain insight into the thermal behaviour of automotive PMSMs by means of the use of PISE techniques. It will also serve as test platform for the theoretical developments presented in this thesis, and also future ones.

Bibliography

[1] F. L. M. dos Santos, R. Pastorino, B. Peeters, C. Faria, W. Desmet, L. C. S. Góes, and H. V. D. Auweraer, "Model based system testing: Bringing testing and simulation close together," in *Structural Health Monitoring, Damage Detection & Mechatronics, Volume 7*, pp. 91–97, Springer International Publishing, 2016.

[2] F. Tao, Q. Qi, L. Wang, and A. Nee, "Digital twins and cyber–physical systems toward smart manufacturing and Industry 4.0: correlation and comparison," *Engineering*, vol. 5, no. 4, pp. 653–661, 2019.

[3] R. Pastorino, "Model-based system testing as enabling technology for model-based product development - keynote lecture," in *ASME IDETC-CIE Multibody Systems and Nonlinear Dynamics and Control Conference*, (Online event), 2020.

[4] R. Pastorino, F. Cosco, F. Naets, W. Desmet, and J. Cuadrado, "Hard real-time multibody simulations using ARM-based embedded systems," *Multibody System Dynamics*, vol. 37, no. 1, pp. 127–143, 2016.

[5] M. Grieves, "Digital twin: Manufacturing excellence through virtual factory replication," tech. rep., White paper, 2014.

[6] F. Naets, J. Croes, and W. Desmet, "An online coupled state/input/parameter estimation approach for structural dynamics," *Computer Methods in Applied Mechanics and Engineering*, vol. 283, pp. 1167–1188, 2015.

[7] C. Gomes, C. Thule, D. Broman, P. G. Larsen, and H. Vangheluwe, "Co-simulation: A survey," *ACM Computing Surveys*, vol. 51, no. 3, pp. 1–33, 2018.

[8] G. Stettinger, M. Benedikt, M. Tranninger, M. Horn, and J. Zehetner, "Recursive FIR-filter design for fault-tolerant real-time co-simulation," in *25th Mediterranean Conference on Control and Automation (MED)*, (Valletta, Malta), pp. 461–466, 2017.

[9] T. Blockwitz, M. Otter, J. Akesson, M. Arnold, C. Clauss, H. Elmqvist, M. Friedrich, A. Junghanns, J. Mauss, D. Neumerkel, H. Olsson, and A. Viel, "Functional Mockup Interface 2.0: The standard for tool independent exchange of simulation models," in *Proceedings of the 9th International MODELICA Conference,*, (Munich, Germany), 2012.

[10] R. Pastorino, *Experimental Validation of a Multibody Model for a Vehicle Prototype and its Application to Automotive State Observers*. PhD thesis, University of A Coruña, 2012.

[11] E. Sanjurjo, *State Observers Based on Detailed Multibody Models Applied to an Automobile*. PhD thesis, University of A Coruña, 2016.

[12] E. Sanjurjo, D. Dopico, A. Luaces, and M. Á. Naya, "State and force observers based on multibody models and the indirect Kalman filter," *Mechanical Systems and Signal Processing*, vol. 106, pp. 210–228, 2018.

[13] F. González, M. González, and A. Mikkola, "Efficient coupling of multibody software with numerical computing environments and block diagram simulators," *Multibody System Dynamics*, vol. 24, no. 3, pp. 237–253, 2010.

[14] F. J. González, *Efficient Implementations and Co-simulation Techniques in Multibody System Dynamics*. PhD thesis, University of A Coruña, 2010.

[15] A. Peiret, F. González, J. Kövecses, M. Teichmann, and A. Enzenhoefer, "Model-based coupling for co-simulation of robotic contact tasks," *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 5756–5763, 2020.

[16] A. J. Rodríguez, R. Pastorino, Á. Carro-Lagoa, K. Janssens, and M. Á. Naya, "Hardware acceleration of multibody simulations for real-time embedded applications," *Multibody System Dynamics*, vol. 51, no. 4, pp. 455–473, 2020.

[17] A. J. Rodríguez, *Implementation in Embedded Systems of State Observers Based on Multibody Dynamics*. PhD thesis, University of A Coruña, 2020.

[18] S. D. I. Software, "Model-based system testing," tech. rep., Siemens Digital Industries Software, 2019.

[19] F. L. Marques dos Santos, J. Anthonis, and H. van der Auweraer, "Multiphysics thermal and NVH modeling: Integrated simulation of a switched reluctance motor drivetrain for an electric vehicle," in *IEEE International Electric Vehicle Conference*, (Greenville, SC, USA), 2012.

[20] C. T. Faria, G. Pulvirenti, and T. Geluk, "Modeling and nonlinear parameter identification of an electric-power steering system," in *Topics in Modal Analysis & Testing*, vol. 10, pp. 127–135, Springer International Publishing, 2017.

[21] Wilhelmus H. A. Schilders, Henk A. van der Vorst, and J. Rommes, *Model Order Reduction: Theory, Research Aspects and Applications*. Springer Berlin Heidelberg, 2008.

[22] J. Lee, K. Yi, D. Lee, B. Jang, M. Kim, and S. Hwang, "Haptic control of steer-by-wire systems for tracking of target steering feedback torque," *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering*, vol. 234, no. 5, pp. 1389–1401, 2019.

[23] P. E. McKenney, "'Real time' vs. 'real fast': How to choose?," in *Proceedings of the Ottawa Linux Symposium*, (Ottawa, Ontario, Canada), 2008.

[24] N. Vun, H. Hor, and J. Chao, "Real-time enhancements for embedded linux," in *14th IEEE International Conference on Parallel and Distributed Systems*, (Melbourne, Australia), 2008.

[25] J. Wu, C. Dufour, and L. Sun, "Hardware-in-the-loop testing of hybrid vehicle motor drives at ford motor company," in *IEEE Vehicle Power and Propulsion Conference*, (Lille, France), 2010.

[26] S. A. Asadollah, R. Inam, and H. Hansson, "A survey on testing for cyber physical system," in *Testing Software and Systems* (K. El-Fakih, G. Barlas, and N. Yevtushenko, eds.), pp. 194–207, Springer, 2015.

[27] Z. Wang, H. Lv, X. Zhou, Z. Chen, and Y. Yang, "Design and modeling of a test bench for dual-motor electric drive tracked vehicles based on a dynamic load emulation method," *Sensors*, vol. 18, no. 7, p. 1993, 2018.

[28] H. Sellami, L. Cazenille, T. Fujii, M. Hagiya, N. Aubert-Kato, and A. J. Genot, "Accelerating the finite-element method for reaction-diffusion simulations on GPUs with CUDA," *Micromachines*, vol. 11, no. 9, p. 881, 2020.

[29] D. Negrut, A. Tasora, H. Mazhar, T. Heyn, and P. Hahn, "Leveraging parallel computing in multibody dynamics," *Multibody System Dynamics*, vol. 27, no. 1, pp. 95–117, 2011.

[30] X. Li, H. Jin, and S. dong Mei, "GPU-accelerated method for real-time shadow generation," in *ICCEA - 2nd International Conference on Computer Engineering and Technology*, vol. 6, (Bali Island, Indonesia), pp. 656–659, 2010.

[31] E. Wu, Y. Liu, and X. Liu, "An improved study of real-time fluid simulation on GPU," *Computer Animation and Virtual Worlds*, vol. 15, 2004.

[32] W. Mucha, "Real-time finite element simulations on ARM microcontroller," *Journal of Applied Mathematics and Computational Mechanics*, vol. 16, pp. 109–116, 2017.

[33] R. Pastorino, F. Cosco, F. Naets, W. Desmet, and J. Cuadrado, "Hard real-time multibody simulations using ARM-based embedded systems," *Multibody System Dynamics*, vol. 37, no. 1, pp. 127–143, 2016.

[34] J. Nunez-Yanez, S. Amiri, M. Hosseinabady, A. Rodríguez, R. Asenjo, A. Navarro, D. Suarez, and R. Gran, "Simultaneous multiprocessing in a software-defined heterogeneous FPGA," *The Journal of Supercomputing*, vol. 75, no. 8, pp. 4078–4095, 2018.

[35] A. Helali, H. Ameur, J. M. Górriz, J. Ramírez, and H. Maaref, "Hardware implementation of real-time pedestrian detection system," *Neural Computing and Applications*, vol. 32, no. 16, pp. 12859–12871, 2020.

[36] Y.-K. Choi, J. Cong, Z. Fang, Y. Hao, G. Reinman, and P. Wei, "In-depth analysis on microarchitectures of modern heterogeneous CPU-FPGA platforms," *ACM Transactions on Reconfigurable Technology and Systems*, vol. 12, no. 1, pp. 1–20, 2019.

[37] R. Featherstone, *Robot dynamics algorithms*. Kluwer Academic Publishers, 1987.

[38] M. González, F. González, D. Dopico, and A. Luaces, "On the effect of linear algebra implementations in real-time multibody system dynamics," *Computational Mechanics*, vol. 41, no. 4, pp. 607–615, 2008.

[39] A. Gupta, "Recent advances in direct methods for solving unsymmetric sparse systems of linear equations," *ACM Transactions on Mathematical Software*, vol. 28, no. 3, pp. 301–324, 2002.

[40] N. I. M. Gould, J. A. Scott, and Y. Hu, "A numerical evaluation of sparse direct solvers for the solution of large sparse symmetric linear systems of equations," *ACM Transactions on Mathematical Software*, vol. 33, no. 2, p. 10, 2007.

[41] S. Turek, C. Becker, and A. Runge, "The FEAST indices—realistic evaluation of modern software components and processor technologies," *Computers & Mathematics with Applications*, vol. 41, no. 10-11, pp. 1431–1464, 2001.

[42] R. M. Schupbach and J. C. Balda, "A versatile laboratory test bench for developing powertrains of electric vehicles," in *Proceedings of the IEEE 56th Vehicular Technology Conference*, vol. 3, (Vancouver, BC, Canada), pp. 1666–1670, 2002.

[43] M. González, D. Dopico, U. Lugrís, and J. Cuadrado, "A benchmarking system for MBS simulation software: Problem standardization and performance measurement," *Multibody System Dynamics*, vol. 16, no. 2, pp. 179–190, 2006.

[44] M. González, F. González, A. Luaces, and J. Cuadrado, "A collaborative benchmarking framework for multibody system dynamics," *Engineering with Computers*, vol. 26, no. 1, pp. 1–9, 2010.

[45] O. A. Bauchau, P. Betsch, A. Cardona, J. Gerstmayr, B. Jonker, P. Masarati, and V. Sonneville, "Validation of flexible multibody dynamics beam formulations using benchmark problems," *Multibody System Dynamics*, vol. 37, no. 1, pp. 29–48, 2016.

[46] B. Kaminska, K. Arabi, I. Bell, P. Goteti, J. Huertas, B. Kim, A. Rueda, and M. Soma, "Analog and mixed-signal benchmark circuits-first release," in *Proceedings of the International Test Conference*, (Washington, DC, USA), 1997.

[47] N. Navet, *Automotive embedded systems handbook*. Boca Raton: CRC Press, 2009.

[48] H. Guo, D. Cao, H. Chen, C. Lv, H. Wang, and S. Yang, "Vehicle dynamic state estimation: state of the art schemes and perspectives," *IEEE/CAA Journal of Automatica Sinica*, vol. 5, no. 2, pp. 418–431, 2018.

[49] S. Kabadayi, A. Pridgen, and C. Julien, "Virtual sensors: Abstracting data from physical sensors," in *International Symposium on a World of Wireless, Mobile and Multimedia Networks(WoWMoM'06)*, (Buffalo-Niagara Falls, NY, USA), 2006.

[50] L. Liu, S. M. Kuo, and M. Zhou, "Virtual sensing techniques and their applications," in *International Conference on Networking, Sensing and Control*, (Okayama, Japan), 2009.

[51] R. L. Neitzel, N. S. Seixas, and K. K. Ren, "A review of crane safety in the construction industry," *Applied Occupational and Environmental Hygiene*, vol. 16, pp. 1106–1117, dec 2001.

[52] N. Raveendranathan, S. Galzarano, V. Loseu, R. Gravina, R. Giannantonio, M. Sgroi, R. Jafari, and G. Fortino, "From modeling to implementation of virtual sensors in body sensor networks," *IEEE Sensors Journal*, vol. 12, no. 3, pp. 583–593, 2012.

[53] S. Moschini, K. Gryllias, W. Desmet, and B. Pluymers, "Virtual sensing for rotordynamics," in *Volume 6: Ceramics; Controls, Diagnostics and Instrumentation; Education; Manufacturing Materials and Metallurgy*, American Society of Mechanical Engineers, 2016.

[54] N. Wiener, *The Extrapolation, Interpolation and Smoothing of Stationary Time Series*. New York: John Wiley & Sons, Inc, 1949.

[55] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Journal of Basic Engineering*, vol. 82, no. 1, pp. 35–45, 1960.

[56] J. F. Bellantoni and K. W. Dodge, "A square root formulation of the Kalman-Schmidt filter," *AIAA Journal*, vol. 5, no. 7, pp. 1309–1314, 1967.

[57] R. Van der Merwe, *Sigma-Point Kalman Filters for Probabilistic Inference in Dynamic State-space Models*. OGI School of Science & Engineering at OHSU, 2004.

[58] T. Lefebvre, H. Bruyninckx, and J. D. Schuller, "Comment on "a new method for the nonlinear transformation of means and covariances in filters and estimators"," *IEEE Transactions on Automatic Control*, vol. 47, no. 8, pp. 1406–1409, 2002.

[59] S. Julier, J. Uhlmann, and H. Durrant-Whyte, "A new approach for filtering nonlinear systems," in *Proceedings of 1995 American Control Conference - ACC'95*, American Autom Control Council, 1995.

[60] S. J. Julier, "Skewed approach to filtering," in *Signal and Data Processing of Small Targets 1998* (O. E. Drummond, ed.), SPIE, 1998.

[61] D. Simon, *Optimal State Estimation*. Hoboken, NJ, USA: John Wiley & Sons, 2006.

[62] T. Li, Y. Zhang, Y. Liang, Q. Ai, and H. Dou, "Multiphysics analysis of an axial-flux in-wheel motor with an amorphous alloy stator," *IEEE Access*, vol. 8, pp. 27414–27425, 2020.

[63] H. F. Grip, L. Imsland, T. A. Johansen, T. I. Fossen, J. C. Kalkkuhl, and A. Suissa, "Nonlinear vehicle velocity observer with road-tire friction adaptation," in *Proceedings of the 45th IEEE Conference on Decision and Control*, IEEE, 2006.

[64] P. A. C. Lopes, J. A. B. Gerald, and M. S. Piedade, "The random walk model kalman filter in multichannel active noise control," *IEEE Signal Processing Letters*, vol. 22, no. 12, pp. 2244–2248, 2015.

[65] A. J. Rodríguez, E. Sanjurjo, R. Pastorino, and M. Á. Naya, "State, parameter and input observers based on multibody models and kalman filters for vehicle dynamics," *Mechanical Systems and Signal Processing*, vol. 155, p. 107544, 2021.

[66] P. D. Groves, *Principles of GNSS, Inertial, and Multisensor Integrated Navigation Systems*. Boston, MA, USA: Artech House, 2008.

[67] Q. Lou, F. González, and J. Kövecses, "Kinematic modeling and state estimation of exploration rovers," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 1311–1318, 2019.

[68] A. Onat, "A novel and computationally efficient joint unscented Kalman filtering scheme for parameter estimation of a class of nonlinear systems," *IEEE Access*, vol. 7, pp. 31634–31655, 2019.

[69] X. Ding, Z. Wang, L. Zhang, and C. Wang, "Longitudinal vehicle speed estimation for four-wheel-independently-actuated electric vehicles based on multi-sensor fusion," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 11, pp. 12797–12806, 2020.

[70] A. Rahmoun, H. Biechl, and A. Rosin, "SOC estimation for li-ion batteries based on equivalent circuit diagrams and the application of a Kalman filter," in *2012 Electric Power Quality and Supply Reliability*, IEEE, 2012.

[71] S. M. Azizi, "Fault diagnosis in three-phase power inverters using multiple-model Kalman filter," in *2019 IEEE International Electric Machines & Drives Conference (IEMDC)*, (San Diego, CA, USA), IEEE, 2019.

[72] B.-H. Lee, K.-S. Kim, J.-W. Jung, J.-P. Hong, and Y.-K. Kim, "Temperature estimation of IPMSM using thermal equivalent circuit," *IEEE Transactions on Magnetics*, vol. 48, no. 11, pp. 2949–2952, 2012.

[73] Q. Chen, Z. Zou, and B. Cao, "Lumped-parameter thermal network model and experimental research of interior PMSM for electric vehicle," *CES Transactions on Electrical Machines and Systems*, vol. 1, no. 4, pp. 367–374, 2017.

[74] B. Rodríguez, F. González, M. Á. Naya, and J. Cuadrado, "Assessment of methods for the real-time simulation of electronic and thermal circuits," *Energies*, vol. 13, no. 6, p. 1354, 2020.

[75] Y.-C. Lo, Y.-C. Hu, and P.-Z. Chang, "Parameter estimation of the thermal network model of a machine tool spindle by self-made bluetooth temperature sensor module," *Sensors*, vol. 18, no. 2, p. paper 656, 2018.

[76] M. A. Eleffendi and C. M. Johnson, "Application of Kalman filter to estimate junction temperature in IGBT power modules," *IEEE Transactions on Power Electronics*, vol. 31, no. 2, pp. 1576–1587, 2016.

[77] M. Musallam and C. M. Johnson, "Real-time compact thermal models for health management of power electronics," *IEEE Transactions on Power Electronics*, vol. 25, no. 6, pp. 1416–1425, 2010.

[78] X. Du, J. Zhang, S. Zheng, and H.-M. Tai, "Thermal network parameter estimation using cooling curve of IGBT module," *IEEE Transactions on Power Electronics*, vol. 34, no. 8, pp. 7957–7971, 2019.

[79] Z. Wang, W. Qiao, B. Tian, and L. Qu, "An effective heat propagation path-based online adaptive thermal model for IGBT modules," in *2014 IEEE Applied Power Electronics Conference and Exposition - APEC 2014*, (Fort Worth, TX, USA), pp. 513–518, 2014.

[80] M. S. Grewal and A. P. Andrews, *Kalman Filtering*. Hoboken, NJ, USA: John Wiley & Sons, 2014.

[81] Y. Dou, "An improved prediction model of IGBT junction temperature based on backpropagation neural network and Kalman filter," *Complexity*, vol. 2021, pp. 1–10, 2021.

[82] X. Wang, A. Castellazzi, and P. Zanchetta, "Observer based temperature control for reduced thermal cycling in power electronic cooling," *Applied Thermal Engineering*, vol. 64, no. 1-2, pp. 10–18, 2014.

[83] X. Dong, A. Griffo, D. Hewitt, and J. Wang, "Reduced-order thermal observer for power modules temperature estimation," *IEEE Transactions on Industrial Electronics*, vol. 67, no. 12, pp. 10085–10094, 2020.

[84] P. Milanfar and J. Lang, "Monitoring the thermal condition of permanent-magnet synchronous motors," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 32, no. 4, pp. 1421–1429, 1996.

[85] D. E. Gaona Erazo, O. Wallscheid, and J. Bocker, "Improved fusion of permanent magnet temperature estimation techniques for synchronous motors using

a Kalman filter," *IEEE Transactions on Industrial Electronics*, vol. 67, no. 3, pp. 1708–1717, 2020.

[86] J. C. Samin, O. Brüls, J. F. Collard, L. Sass, and P. Fisette, "Multiphysics modeling and optimization of mechatronic multibody systems," *Multibody System Dynamics*, vol. 18, no. 3, pp. 345–373, 2007.

[87] M. Naya, J. Cuadrado, D. Dopico, and U. Lugris, "An efficient unified method for the combined simulation of multibody and hydraulic dynamics: Comparison with simplified and co-integration approaches," *Archive of Mechanical Engineering*, vol. 58, no. 2, 2011.

[88] J. Rahikainen, M. Kiani, J. Sopanen, P. Jalali, and A. Mikkola, "Computationally efficient approach for simulation of multibody and hydraulic dynamics," *Mechanism and Machine Theory*, vol. 130, pp. 435–446, 2018.

[89] C. Lacoursière and T. Härdin, "FMI go! a simulation runtime environment with a client server architecture over multiple protocols," in *Proceedings of the 12th International Modelica Conference*, (Prague, Czech Republic), Linköping University Electronic Press, 2017.

[90] S. Sadjina, L. T. Kyllingstad, M. Rindarøy, S. Skjong, V. Æsøy, and E. Pedersen, "Distributed co-simulation of maritime systems and operations," *Journal of Offshore Mechanics and Arctic Engineering*, vol. 141, no. 1, 2019.

[91] O. Vaculin, W. Krüger, and M. Valasek, "Overview of coupling of multibody and control engineering tools," *Vehicle System Dynamics*, vol. 41, pp. 415–429, 2004.

[92] "FMI - Functional Mock-up Interface, https://fmi-standard.org/."

[93] C. Andersson, *Methods and Tools for Co-Simulation of Dynamic Systems with the Functional Mock-up Interface*. PhD thesis, Lund University, 2016.

[94] O. Oberschelp and H. Vöcking, "Multirate simulation of mechatronic systems," in *Proceedings of the IEEE International Conference on Mechatronics, 2004.*, (Istanbul, Turkey), IEEE, 2004.

[95] J. Rahikainen, F. González, and M. Á. Naya, "An automated methodology to select functional co-simulation configurations," *Multibody System Dynamics*, vol. 48, no. 1, pp. 79–103, 2020.

[96] A. Peiret, F. González, J. Kövecses, and M. Teichmann, "Multibody system dynamics interface modelling for stable multirate co-simulation of multiphysics systems," *Mechanism and Machine Theory*, vol. 127, pp. 52–72, 2018.

[97] B. Hannaford and J.-H. Ryu, "Time-domain passivity control of haptic interfaces," *IEEE Transactions on Robotics and Automation*, vol. 18, no. 1, pp. 1–10, 2002.

[98] H. Vangheluwe, J. Lara, and P. Mosterman, "An introduction to multi-paradigm modelling and simulation," in *Proceedings of the AIS'2002 Conference (AI, Simulation and Planning in High Autonomy Systems)*, (Lisbon, Portugal), 2002.

[99] H. V. der Auweraer, J. Anthonis, S. D. Bruyne, and J. Leuridan, "Virtual engineering at work: the challenges for designing mechatronic products," *Engineering with Computers*, vol. 29, no. 3, pp. 389–408, 2012.

[100] F. Cremona, M. Lohstroh, D. Broman, E. A. Lee, M. Masin, and S. Tripakis, "Hybrid co-simulation: it's about time," *Software & Systems Modeling*, vol. 18, no. 3, pp. 1655–1679, 2019.

[101] H. Kopetz, *Real-Time Systems*. Springer-Verlag GmbH, 2011.

[102] B. Schweizer, P. Li, and D. Lu, "Explicit and implicit cosimulation methods: Stability and convergence analysis for different solver coupling approaches," *Journal of Computational and Nonlinear Dynamics*, vol. 10, no. 5, 2015.

[103] B. Schweizer, P. Li, D. Lu, and T. Meyer, "Stabilized implicit co-simulation methods: solver coupling based on constitutive laws," *Archive of Applied Mechanics*, vol. 85, no. 11, pp. 1559–1594, 2015.

[104] R. Kübler and W. Schiehlen, "Modular simulation in multibody system dynamics," *Multibody System Dynamics*, vol. 4, no. 2, pp. 107–127, 2000.

[105] N. R. Tavana and V. Dinavahi, "Real-time nonlinear magnetic equivalent circuit model of induction machine on FPGA for hardware-in-the-loop simulation," *IEEE Transactions on Energy Conversion*, vol. 31, no. 2, pp. 520–530, 2016.

[106] T. Duan, T. Cheng, and V. Dinavahi, "Heterogeneous real-time co-emulation for communication-enabled global control of AC/DC grid integrated with renewable energy," *IEEE Open Journal of the Industrial Electronics Society*, vol. 1, pp. 261–270, 2020.

[107] C. W. Gear and D. R. Wells, "Multirate linear multistep methods," *BIT*, vol. 24, no. 4, pp. 484–502, 1984.

[108] F. González, M. Á. Naya, A. Luaces, and M. González, "On the effect of multirate co-simulation techniques in the efficiency and accuracy of multibody system dynamics," *Multibody System Dynamics*, vol. 25, no. 4, pp. 461–483, 2011.

[109] F. González, S. Arbatani, A. Mohtat, and J. Kövecses, "Energy-leak monitoring and correction to enhance stability in the co-simulation of mechanical systems," *Mechanism and Machine Theory*, vol. 131, pp. 172–188, 2019.

[110] J. Rahikainen, F. González, M. Á. Naya, J. Sopanen, and A. Mikkola, "On the cosimulation of multibody systems and hydraulic dynamics," *Multibody System Dynamics*, vol. 50, no. 2, pp. 143–167, 2020.

[111] M. E. Latoschik and H. Tramberend, "Short paper: Engineering realtime interactive systems: Coupling and cohesion of architecture mechanisms," 2010.

[112] M. Busch, "Continuous approximation techniques for co-simulation methods: Analysis of numerical stability and local error," *Journal of Applied Mathematics and Mechanics / Zeitschrift für Angewandte Mathematik und Mechanik*, vol. 96, no. 9, pp. 1061–1081, 2016.

[113] A. Ben Khaled-El Feki, L. Duval, C. Faure, D. Simon, and M. Ben Gaid, "CHOPtrey: contextual online polynomial extrapolation for enhanced multi-core co-simulation of complex systems," *Simulation*, vol. 93, no. 3, pp. 185–200, 2017.

[114] M. Benedikt, D. Watzenig, and A. Hofer, "Modelling and analysis of the non-iterative coupling process for co-simulation," *Mathematical and Computer Modelling of Dynamical Systems*, vol. 19, no. 5, pp. 451–470, 2013.

[115] T. Meyer, P. Li, D. Lu, and B. Schweizer, "Implicit co-simulation method for constraint coupling with improved stability behavior," *Multibody System Dynamics*, vol. 44, no. 2, pp. 135–161, 2018.

[116] T. Haid, G. Stettinger, D. Watzenig, and M. Benedikt, "A model-based corrector approach for explicit co-simulation using subspace identification," in *Proceedings of the 5th Joint International Conference on Multibody System Dynamics, Lisbon, Portugal*, 2018.

[117] S. Sadjina, L. T. Kyllingstad, S. Skjong, and E. Pedersen, "Energy conservation and power bonds in co-simulations: Non-iterative adaptive step size control and error estimation," *Engineering with Computers*, vol. 33, no. 3, pp. 607–620, 2017.

[118] S. Sadjina and E. Pedersen, "Energy conservation and coupling error reduction in non-iterative co-simulations," *Engineering with Computers*, vol. 36, no. 4, pp. 1579–1587, 2020.

[119] M. Benedikt, D. Watzenig, J. Zehetner, and A. Hofer, "A nearly energy-preserving coupling element for holistic weak-coupled system co-simulations," in *NAFEMS World Congress 2013*, (Salzburg, Austria), 2013.

[120] V. Nguyen, Y. Besanger, Q. Tran, and T. Nguyen, "On conceptual structuration and coupling methods of co-simulation frameworks in cyber-physical energy system validation," *Energies*, vol. 10, no. 12, p. 1977, 2017.

[121] L. W. Nagel and D. O. Pederson, "Simulation program with integrated circuit emphasis," in *Proceedings of the sixteenth Midwest symposium on circuit theory*, (Waterloo, ON, Canada), 1973.

[122] L. O. Chua and P.-M. Lin, *Computer-Aided Analysis of Electronic Circuits : Algorithms and Computational Techniques*. Englewood Cliffs, N.J: Prentice-Hall, 1975.

[123] C. Dufour, S. Abourida, and J. Bélanger, "Real-time simulation of electrical vehicle motor drives on a PC cluster," in *Proceedings of the 10th European Conference on Power Electronics and Applications (EPE-2003)*, (Toulouse, France), 2003.

[124] J. Almaguer, V. Cárdenas, J. Espinoza, A. Aganza-Torres, and M. González, "Performance and control strategy of real-time simulation of a three-phase solid-state transformer," *Applied Sciences*, vol. 9, no. 4, p. 789, 2019.

[125] B. Zhang, X. Jin, S. Tu, Z. Jin, and J. Zhang, "A new FPGA-based real-time digital solver for power system simulation," *Energies*, vol. 12, no. 24, p. 4666, 2019.

[126] S. Iwata, M. Takamatsu, and C. Tischendorf, "Tractability index of hybrid equations for circuit simulation," *Math. Comput.*, vol. 81, pp. 923–939, 2012.

[127] N. M. Newmark, "A method of computation for structural dynamics," *Journal of the Engineering Mechanics Division, ASCE*, vol. 85, no. EM3, pp. 67–94, 1959.

[128] E. Hairer, S. P. Nørsett, and G. Wanner, *Solving Ordinary Differential Equations I.* Berlin, Germany: Springer, 1993.

[129] G. Hachtel, R. Brayton, and F. Gustavson, "The sparse tableau approach to network analysis and design," *IEEE Transactions on Circuit Theory*, vol. 18, no. 1, pp. 101–113, 1971.

[130] C. Gear, "Simultaneous numerical solution of differential-algebraic equations," *IEEE Transactions on Circuit Theory*, vol. 18, no. 1, pp. 89–95, 1971.

[131] D. Calahan, "Numerical considerations for implementation of a nonlinear transient circuit analysis program," *IEEE Transactions on Circuit Theory*, vol. 18, no. 1, pp. 66–73, 1971.

[132] S. Iwata and M. Takamatsu, "Index minimization of differential-algebraic equations in hybrid analysis for circuit simulation," *Mathematical Programming*, vol. 121, pp. 105–121, 2010.

[133] I. Hajj, *Computational Methods in Circuit Simulation.* United States: Ibrahim N. Hajj, 2016.

[134] J. G. Fijnvandraat, S. H. M. J. Houben, E. J. W. ter Maten, and J. M. F. Peters, "Time domain analog circuit simulation," *Journal of Computational and Applied Mathematics*, vol. 185, no. 2, pp. 441–459, 2006.

[135] P. Maffezzoni, L. Codecasa, and D. D'Amore, "Time-domain simulation of nonlinear circuits through implicit Runge-Kutta methods," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 54, no. 2, pp. 391–400, 2007.

[136] F. Yuan and A. Opal, "Computer methods for switched circuits," *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, vol. 50, no. 8, pp. 1013–1024, 2003.

[137] V. Acary, O. Bonnefon, and B. Brogliato, "Time-stepping numerical simulation of switched circuits within the nonsmooth dynamical systems approach," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 29, no. 7, pp. 1042–1055, 2010.

[138] F. Ebert and S. Bächle, "Element-based index reduction in electrical circuit simulation," *PAMM*, vol. 6, no. 1, pp. 731–732, 2006.

[139] A. E. Jaramillo, P. Boulanger, and F. Prieto, "On-line 3-d system for the inspection of deformable parts," *The International Journal of Advanced Manufacturing Technology*, vol. 57, no. 9-12, pp. 1053–1063, 2011.

[140] A. Steinbrecher and T. Stykel, "Model order reduction of electrical circuits with nonlinear elements," in *Mathematics in Industry*, pp. 169–177, Springer Berlin Heidelberg, 2012.

[141] A. Davoudi, J. Jatskevich, P. L. Chapman, and A. Bidram, "Multi-resolution modeling of power electronics circuits using model-order reduction techniques," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 60, no. 3, pp. 810–823, 2013.

[142] Z. Hu, M. Du, K. Wei, and W. G. Hurley, "An adaptive thermal equivalent circuit model for estimating the junction temperature of IGBTs," *IEEE Journal of Emerging and Selected Topics in Power Electronics*, vol. 7, no. 1, pp. 392–403, 2019.

[143] J. Vlach and K. Singhal, *Computer Methods for Circuit Analysis and Design*. Van Nostrand Reinhold, 1983.

[144] C.-W. Ho, A. Ruehli, and P. Brennan, "The modified nodal approach to network analysis," *IEEE Transactions on Circuits and Systems*, vol. 22, no. 6, pp. 504–509, 1975.

[145] E. Bayo and R. Ledesma, "Augmented Lagrangian and mass–orthogonal projection methods for constrained multibody dynamics," *Nonlinear Dynamics*, vol. 9, no. 1-2, pp. 113–130, 1996.

[146] D. Dopico, F. González, J. Cuadrado, and J. Kövecses, "Determination of holonomic and nonholonomic constraint reactions in an index-3 augmented Lagrangian formulation with velocity and acceleration projections," *Journal of Computational and Nonlinear Dynamics*, vol. 9, no. 4, 2014.

[147] T. A. Davis and E. P. Natarajan, "Algorithm 907: KLU, a direct sparse solver for circuit simulation problems," *ACM Transactions on Mathematical Software*, vol. 37, no. 3, pp. 1–17, 2010.

[148] C. Sah, R. Noyce, and W. Shockley, "Carrier generation and recombination in P-N junctions and P-N junction characteristics," *Proceedings of the IRE*, vol. 45, no. 9, pp. 1228–1243, 1957.

[149] S. Touhami, Y. Bertin, Y. Lefèvre, J. F. Llibre, C. Henaux, and M. Fénot, "Lumped parameter thermal model of permanent magnet synchronous machines," tech. rep., LAPLACE - LAboratoire PLasma et Conversion d'Energie, 2017.

[150] J. Salvador Iborra, *A Contribution to the Global Modeling of Heat Transfer Processes in Diesel Engines*. PhD thesis, Universitat Politècnica de València, 2020.

[151] Nexperia, "RC thermal models," Tech. Rep. AN11261, Nexperia, 2020.

[152] O. Wallscheid and J. Böcker, "Global identification of a low-order lumped-parameter thermal network for permanent magnet synchronous motors," *IEEE Transactions on Energy Conversion*, vol. 31, no. 1, pp. 354–365, 2016.

[153] D. Simon, "Kalman filtering with state constraints: A survey of linear and nonlinear algorithms," *IET Control Theory & Applications*, vol. 4, no. 8, pp. 1303–1318, 2010.

[154] J. Schulz-Harder, "Advantages and new development of direct bonded copper substrates," *Microelectronics Reliability*, vol. 43, no. 3, pp. 359–365, 2003.

[155] D. Murdock, J. Torres, J. Connors, and R. Lorenz, "Active thermal control of power electronic modules," *IEEE Transactions on Industry Applications*, vol. 42, no. 2, pp. 552–558, 2006.

[156] M. Andresen, G. Buticchi, and M. Liserre, "Study of reliability-efficiency tradeoff of active thermal control for power electronic systems," *Microelectronics Reliability*, vol. 58, pp. 119–125, 2016.

[157] F. P. Incropera, D. P. Dewitt, T. L. Bergman, and A. S. Lavine, *Fundamentals of Heat and Mass Transfer*. John Wiley & Sons, 2007.

[158] D. Dopico, A. Luaces, M. Gonzalez, and J. Cuadrado, "Dealing with multiple contacts in a human-in-the-loop application," *Multibody System Dynamics*, vol. 25, no. 2, pp. 167–183, 2010.

[159] J. García de Jalón, "Twenty-five years of natural coordinates," *Multibody System Dynamics*, vol. 18, no. 1, pp. 15–33, 2007.

[160] R. Perry and K. Arora, "Using PSPICE to simulate the photoresponse of ideal CMOS integrated circuit photodiodes," in *Proceedings of SOUTHEASTCON '96*, (Tampa, FL, USA), IEEE, 1996.

[161] J. Faiz, M. Ojaghi, and A. Keyhani, "PSPICE simulation of single-phase induction motors," *IEEE Transactions on Energy Conversion*, vol. 14, no. 1, pp. 86–92, 1999.

[162] J. Rahikainen, A. Mikkola, J. Sopanen, and J. Gerstmayr, "Combined semi-recursive formulation and lumped fluid method for monolithic simulation of multibody and hydraulic dynamics," *Multibody System Dynamics*, vol. 44, no. 3, pp. 293–311, 2018.

[163] I. Hafner and N. Popper, "On the terminology and structuring of co-simulation methods," in *Proceedings of the 8th International Workshop on Equation-Based Object-Oriented Modeling Languages and Tools*, (Weßling, Germany), 2017.

[164] T. Meyer, J. Kraft, and B. Schweizer, "Co-simulation: Error estimation and macro-step size control," *Journal of Computational and Nonlinear Dynamics*, vol. 16, no. 4, p. 041002, 2021.

[165] B. Gu and H. H. Asada, "Co-simulation of algebraically coupled dynamic subsystems without disclosure of proprietary subsystem models," *Journal of Dynamic Systems, Measurement, and Control*, vol. 126, no. 1, pp. 1–13, 2004.

[166] B. Schweizer and D. Lu, "Semi-implicit co-simulation approach for solver coupling," *Archive of Applied Mechanics*, vol. 84, no. 12, pp. 1739–1769, 2014.

[167] B. Schweizer and D. Lu, "Stabilized index-2 co-simulation approach for solver coupling with algebraic constraints," *Multibody System Dynamics*, vol. 34, no. 2, pp. 129–161, 2015.

[168] M. Arnold, C. Clauss, and T. Schierz, "Error analysis and error estimates for co-simulation in FMI for model exchange and co-simulation v2.0," *Archive of Mechanical Engineering*, vol. 60, no. 1, pp. 75–94, 2013.

[169] M. Burger and S. Steidel, "Local extrapolation and linear-implicit stabilization in a parallel coupling scheme," in *IUTAM Symposium on Solver-Coupling and Co-Simulation*, pp. 43–56, Berlin, Germany: Springer International Publishing, 2019.

[170] "Modelica, https://www.modelica.org/," 2019.

[171] B. Schweizer, P. Li, D. Lu, and T. Meyer, "Stabilized implicit cosimulation method: Solver coupling with algebraic constraints for multibody systems," *Journal of Computational and Nonlinear Dynamics*, vol. 11, no. 2, p. 021002, 2016.

[172] J. García de Jalón and E. Bayo, *Kinematic and Dynamic Simulation of Multibody Systems. The Real–Time Challenge*. New York, USA: Springer–Verlag, 1994.

[173] R. Zhang, H. Zhang, A. Zanoni, Q. Wang, and P. Masarati, "A tight coupling scheme for smooth/non-smooth multibody co-simulation of a particle damper," *Mechanism and Machine Theory*, vol. 161, p. 104181, 2021.

[174] M. Arnold, "How to verify worst case asymptotic error bounds for co-simulation algorithms," in *Proceedings of COSIM2021, an International Symposium on Co-simulation and Solver Coupling in Dynamics* (F. González and J. Cuadrado, eds.), (Ferrol, Spain), pp. 48–49, 2021.

[175] J.-H. Ryu, C. Preusche, B. Hannaford, and G. Hirzinger, "Time domain passivity control with reference energy following," *IEEE Transactions on Control Systems Technology*, vol. 13, no. 5, pp. 737–742, 2005.

[176] E. Hesla, "Electric propulsion [history]," *IEEE Industry Applications Magazine*, vol. 15, no. 4, pp. 10–13, 2009.

[177] E. V. Beyerleyn and P. V. Tyuteva, "Energy efficiency of back-to-back method for induction traction motors testing," in *2014 15th International Conference of Young Specialists on Micro/Nanotechnologies and Electron Devices (EDM) Novosibirsk, Russia*, IEEE, 2014.

[178] H. He, F. chun Sun, and J. Xing, "Dynamic simulation and experiment of electric drive system on test bench," in *2007 IEEE International Conference on Vehicular Electronics and Safety*, (Beijing, China), IEEE, 2007.

[179] E. Bayo, J. García de Jalón, and M. A. Serna, "A modified Lagrangian formulation for the dynamic analysis of constrained mechanical systems," *Computer Methods in Applied Mechanics and Engineering*, vol. 71, no. 2, pp. 183–195, 1988.

[180] A. Parra, A. J. Rodriguez, A. Zubizarreta, and J. Perez, "Validation of a real-time capable multibody vehicle dynamics formulation for automotive testing frameworks based on simulation," *IEEE Access*, vol. 8, pp. 213253–213265, 2020.

[181] "Global technical regulation no. 15, worldwide harmonized light vehicles test procedure, ece/trans/180/add.15," tech. rep., United Nations, 2014.

[182] J. F. Mora, *Circuitos Eléctricos*. Madrid: Pearson, 2012.

[183] K. Ogata, *Modern Control Engineering*. Boston, MA, USA: Prentice-Hall, 2010.

Appendices

Appendix A

Software

During this research, four software libraries have been developed to test the proposed formulation and algorithms. These libraries must fulfil several requisites, such as:

- *portability* between operating systems (Windows and Linux-based ones), just as hardware platforms (x86, x86-64 and ARM).
- *modularity* allows to be partitioned and reused for similar applications.
- *efficiency* to achieve RT capabilities independently of the operating system and hardware platform.

To this end, a general-purpose and object-oriented programming language as C++ has been selected for developing the codes, which satisfies the aforementioned criteria.

A.1 In-house developed software

Before making the decision of developing an own software for this research, some commercial software was tested. However, the RT capabilities or the difficulties to adapt the software to the necessities of this work have forced to make the decision of developing an in-house software.

This software is composed of a set of libraries, which contain the algorithms described in this thesis. The main features are summarized below:

- *limTools:* performs a wide variety of tasks that support the functionality of every other library developed in this project.
- *electroLIM:* is used for describing, formulating and solving electric circuits as described in Chapter 3. This library also contains a module called *thermoLIM* that is the extension for simulating thermal circuits described as LPTM. Both types of circuits can be simulated with this library under RT constraints. Input, parameter and state estimation described in Chapter 4 can be also carried out with a specific solver included in this library.

- *cosim_suite:* runs co-simulations using the FMI standard. This library implements a simple way to compute co-simulations among several subsystems defined through FMI standard. The necessity of this library stems from customizing the different co-simulation setups explained in Chapter 5 in an efficient and easy manner.

- *testbench:* commands electric motors via USB or CAN-bus and reads dynamic variables as speed, torque or electric parameters. This data is read from the data acquisition system and processed to show useful information to the user. This library is used for the SitL application of the Chapter 6.

A.2 Third-party software

The aforementioned libraries require the use of different algorithms for performing a wide variety of tasks, e.g., string sorting, matrix and vector manipulations or linear algebra solvers. These algorithms have been published on the Internet under free license, in order to ease other software developments. The use of these published algorithms instead of developing a new version of them, is justified due to the complexity and high performance that these libraries have already reached in the last releases. These software codes are listed below:

- *Eigen*[1]*:* is a high-level library written in C++ that contains the templates for linear algebra: matrices, vectors, solvers, and related algorithms.

- *Boost*[2]*:* is a set of libraries written in the C++ language that implements a broad spectrum of applications.

- *SuiteSparse*[3]*:* is a suite of sparse matrix algorithms written in C and C++. It contains very efficient sparse solvers for the linear problems.

- *Libxml2*[4]*:* contains a XML parser written in the C language, in order to load subsystem model descriptions under a FMI standard.

- *OpenMP*[5]*:* or Open Multi-Processing is a multi-platform Application Programming Interface (API) available in different languages, such as Fortran, C or C++. It is widely used for developing easily multi-core and multi-thread algorithms.

- *Qt*[6]*:* is a widget toolkit for Graphical User Interface (GUI) creation. It is written in the C++ language and its developments are cross-platform, such as for PC or embedded systems.

[1] http://eigen.tuxfamily.org/index.php?title=Main_Page
[2] https://www.boost.org/
[3] https://people.engr.tamu.edu/davis/suitesparse.html
[4] http://xmlsoft.org/index.html
[5] https://www.openmp.org/
[6] https://www.qt.io/

- *EPOS*[7]*:* is a library written in the C language by Maxon Motor AG for commanding via USB or CAN its own electric motors. It is also able to receive the data from the available sensors.
- *NI DAQmx*[8]*:* is a library written in C language by National Instruments Corporation to communicate its data acquisition systems with a PC.

[7]https://www.maxongroup.es/medias/sys_master/root/8839888044062/EPOS-2-4-IDX-Setup.zip

[8]https://www.ni.com/es-es/support/downloads/drivers/download.ni-daqmx.html#348669

Appendix B

Airgap resistor correlation

The value of the airgap resistance in the PMSM in Section 3.4.4 is obtained from the following relation

$$R_{\text{airgap}} = \frac{1}{h_{tc}\, S_a} \tag{B.1}$$

where h_{tc} is the convection coefficient at the airgap and S_a is the area of the common surface between stator and rotor. The convection coefficient h_{tc} is a function of the Nusselt number, Nu

$$Nu = \frac{h_{tc}\, \kappa_{\text{air}}}{2e} \tag{B.2}$$

where κ_{air} is the air conductivity and e is the air gap thickness. The Nusselt number can be evaluated using the correlation presented in [149] as

$$\begin{cases} Nu = 2 & \text{if } T_{am} < 1700 \\ Nu = 0.128\, T_{am}^{0.367} & \text{if } 1700 < T_{am} < 10^4 \\ Nu = 0.409\, T_{am}^{0.241} & \text{if } T_{am} > 10^4 \end{cases} . \tag{B.3}$$

where the term T_{am} is defined as

$$T_{am} = \frac{T_a}{F_g} \tag{B.4}$$

$$T_a = \frac{\omega^2\, R_a\, e^3}{\nu_{\text{air}}^2} \tag{B.5}$$

$$F_g = \frac{\pi^4}{P} \frac{1}{1697 \left(1 - \dfrac{e}{2R_a}\right)^2} \tag{B.6}$$

$$P = 0.0571\,(1 - 0.625x) + \frac{0.00056}{1 - 0.625x} \tag{B.7}$$

$$x = \frac{e/R_a}{1 - e/2R_a} \tag{B.8}$$

in the previous equations, ω is the angular speed of the motor, R_a is the mean radius of the rotor and the stator at the airgap and ν_{air} is the kinematic viscosity of the

air. The physical parameters of the air, such as viscosity or conductivity, can be found in [157].

Appendix C

Kirchhoff's Laws and component equations

An electric circuit can be considered as a system composed of lumped elements in such a way that dynamics fulfils component, KCL and KVL system of equations [182]. In the following section, component constitutive equations and circuit equations will be detailed.

C.1 Kirchhoff's Current Laws

This law stems from the conservation of the charge, which establishes that the incoming and outgoing currents are balanced at every moment, or *the algebraic sum of currents in a network of conductors meeting at a point is zero.*

$$\sum_{\forall \tilde{\alpha}} I_{\tilde{\alpha}} = 0 \tag{C.1}$$

In the example illustrated in Fig. C.1, the sum of incoming currents I_G and I_C are equal to the sum of outgoing currents I_R and I_L or $I_G + I_C - I_R - I_L = 0$.

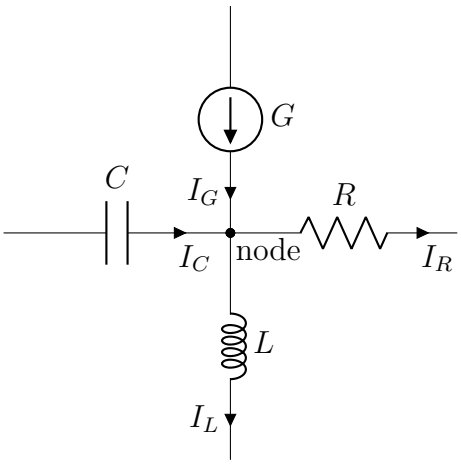


Figure C.1: Currents meeting at a node.

C.2 Kirchhoff's Voltage Laws

This law stems from the conservation of the energy, which establishes the voltage drops around a closed loop are balanced at every moment, or *the directed sum of the voltage differences around any closed loop is zero.*

$$\sum_{\forall \tilde{\alpha}} V_{\tilde{\alpha}} = 0 \tag{C.2}$$

In the example shown in Fig. C.2, the sum of the voltage drops of a closed loop is zero or $E + V_R + V_L + V_C = 0$.

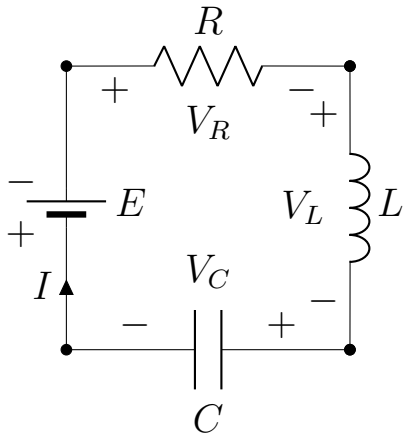


Figure C.2: Voltage drops around a closed loop.

C.3 Component equations

Components can be divided into two groups: active and passive. Active elements are the components that provide power to the circuit, such as sources. Passive elements, on the contrary, dissipate or store power coming from the rest of the system.

C.4 Active component equations

Active and ideal components can be classified into two large groups: voltage and currents sources, depending on which variable is fixed in the component.

C.4.1 Voltage sources

A voltage source is an element that provides a voltage E between its two nodes, regardless of the current that passes through it. Fig. C.3 represents an example of a constant voltage source. According to the value of E, sources could be classified in three subcategories:

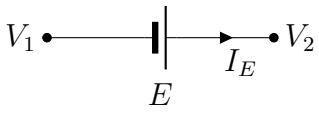


Figure C.3: A voltage source.

- *Constant:* E remains constant along time.
- *Variable:* The value of E is considered to be a function of time.
- *Dependent:* E is a function of other variables of the system, e.g., a current through a resistor.

Every voltage source imposes only one algebraic constraint given by

$$V_2 - V_1 - E = 0 \tag{C.3}$$

C.4.2 Current sources

A current source, on the contrary, fixes the current G that passes through it, regardless of the voltage between its nodes. Figure C.4 represents an example of a constant current source. Current sources can be classified with the same criterion

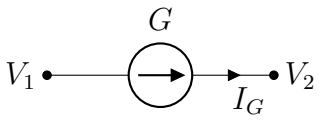


Figure C.4: A current source.

than voltage sources in Section C.4.1.

Every current source imposes only one algebraic constraint given by

$$I_G - G = 0 \tag{C.4}$$

C.5 Passive component equations

Passive ideal elements can be classified into three categories: resistors, inductors and capacitors.

C.5.1 Resistors

A resistor is an element that dissipates a heat power when a current flows through it. The parameter R called resistance is the ratio between the voltage and the current. Figure C.5 represents an example of a resistor. Every resistor imposes only one algebraic constraint (also known as Ohm's Law) given by

$$V_1 - V_2 - I_R R = 0 \tag{C.5}$$

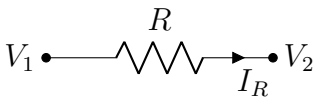


Figure C.5: A resistor.

C.5.2 Inductors

An inductor is an electrical element that stores power in a magnetic field when a current flows through its coils. The parameter L called inductance is the ratio of the voltage variation and the current. Figure C.6 represents an example of an inductor.

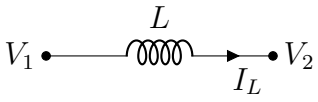


Figure C.6: An inductor.

Every inductor imposes a single differential equation given by

$$V_1 - V_2 - \dot{I}_R L = 0 \tag{C.6}$$

C.5.3 Capacitors

A capacitor is an electrical element that stores power in an electric field when a voltage is applied between its nodes. The parameter C called capacitance is the ratio of the current ad voltage variation. Figure C.7 represents an example of a capacitor. Every capacitor imposes only one differential equation given by

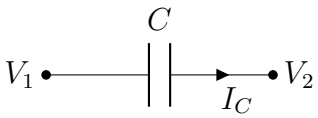


Figure C.7: A capacitor.

$$C\left(\dot{V}_1 - \dot{V}_2\right) - I_C = 0 \tag{C.7}$$

Appendix D

Backward Differentiation Formula coefficients

BDF is a multi-step method, which uses already computed states to calculate derivatives and states at the next time step. BDF is classified according to the number of already calculated time steps used (ξ-order uses ξ already calculated steps and the values at time t_{k+1}).

In this method, derivatives are a sub-product and they are not used to calculate the states. In spite of using derivatives, they are replaced by a specific formula using only computed states and t_{k+1} state. The $\xi + 1$ coefficients of the ξ-order BDF can be obtained as [133]

$$\begin{bmatrix} 1 & 1 & \dots & 1 \\ 0 & \frac{t_{k+1} - t_k}{h} & \dots & \frac{t_{k+1} - t_{k-\xi+1}}{h} \\ \vdots & \vdots & \dots & \vdots \\ 0 & \left(\frac{t_{k+1} - t_k}{h}\right)^{\xi} & \dots & \left(\frac{t_{k+1} - t_{k-\xi+1}}{h}\right)^{\xi} \end{bmatrix} \begin{bmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_\xi \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ \vdots \\ 0 \end{bmatrix} \tag{D.1}$$

where $\beta_{\tilde{\alpha}}$ are the coefficients and ξ is the order of the BDF.

Appendix E

State-space representation

A state-space representation is a mathematical model of a linear physical system by means of a first-order differential equations, in which the derivatives and outputs can be expressed using the system states and inputs [183]. The most general representation for time-continuous systems is

$$\dot{\mathbf{x}} = \mathbf{F}_\mathrm{c}\mathbf{x} + \mathbf{G}_\mathrm{c}\mathbf{u} \tag{E.1}$$

$$\mathbf{y} = \mathbf{H}\mathbf{x} + \mathbf{N}\mathbf{u} \tag{E.2}$$

where $\mathbf{x}$ is the vector of state variables, $\mathbf{F}$ is the state matrix, $\mathbf{G}$ is the input matrix, $\mathbf{H}$ is the output matrix, $\mathbf{N}$ is the feedthrough matrix, $\mathbf{u}$ is the vector of inputs and $\mathbf{y}$ is the vector or outputs. Subscript $()_\mathrm{c}$ denotes the time-continuous form of the matrices.

The dynamical systems are described through differential equations, which are subsequently discretized to simulate and implement control strategies on electronic devices, among other tasks. This process requires a transformation from a continuous system into a discrete one. The most general representation for time-discrete systems is

$$\mathbf{x}_{k+1} = \mathbf{F}_\mathrm{d}\mathbf{x}_k + \mathbf{G}_\mathrm{d}\mathbf{u}_k \tag{E.3}$$

where k is the current step and subscript $()_\mathrm{d}$ denotes time-discrete form of the terms.

E.1 Discretization process

Discrete state-space matrices can be easily transformed from their continuous form by means of the following expressions [80]

$$\mathbf{F}_\mathrm{d} = e^{\mathbf{F}_\mathrm{c}h} = \sum_{\tilde{\alpha}=0}^{\infty} \frac{(\mathbf{F}_\mathrm{c}h)^{\tilde{\alpha}}}{\tilde{\alpha}!} \approx \mathbf{I} + \mathbf{F}_\mathrm{c}h \tag{E.4}$$

$$\mathbf{G}_\mathrm{d} = \mathbf{F}_\mathrm{c}^{-1}\left(\mathbf{F}_\mathrm{d} - \mathbf{I}\right)\mathbf{G}_\mathrm{c} \approx \mathbf{G}_\mathrm{c}h \tag{E.5}$$

where h is the step-size and $\mathbf{I}$ is an identity matrix.

However, the previous expressions can only be used if $\mathbf{F}_\mathrm{c}$ is regular, although it is possible to compute the transformations even if $\mathbf{F}_\mathrm{c}$ is singular as

$$\begin{bmatrix} \mathbf{F}_\mathrm{d} & \mathbf{G}_\mathrm{d} \\ \mathbf{0} & \mathbf{I} \end{bmatrix} = e^{\begin{bmatrix} \mathbf{F}_\mathrm{c} & \mathbf{G}_\mathrm{c} \\ \mathbf{0} & \mathbf{0} \end{bmatrix}h} \tag{E.6}$$

E.2 Controllability

A system which is described in the state-space form and defined by a vector of with n variables, it is controllable only if its controllability matrix $\mathbf{Co}$ has full column rank. The controllability test is formulated as

$$\mathbf{Co} = \begin{bmatrix} \mathbf{G}_\mathrm{d} \\ \mathbf{F}_\mathrm{d}\mathbf{G}_\mathrm{d} \\ \vdots \\ \mathbf{F}_\mathrm{d}^{n-1}\mathbf{G}_\mathrm{d} \end{bmatrix} \tag{E.7}$$

In [61] other definitions of controllability are described for continuous and discrete systems as follows:

A continuous system: "is controllable if for any initial state $\mathbf{x}(0)$ and any final time $t > 0$ there exists a control that transfers the state to any desired value at time t".

A discrete system: "is controllable if for any initial state $\mathbf{x}(0)$ and some final time k there exists a control that transfers the state to any desired value at time k"

E.3 Observability

A system which is described in the state-space form and defined by a vector of with n variables, it is observable only if its observability matrix $\mathbf{O}$ has full row rank. The observability test is formulated as

$$\mathbf{O} = \begin{bmatrix} \mathbf{F}_\mathrm{d} \\ \mathbf{H}\mathbf{F}_\mathrm{d} \\ \vdots \\ \mathbf{H}^{n-1}\mathbf{F}_\mathrm{d} \end{bmatrix} \tag{E.8}$$

In [61] other definitions of observability are described for continuous and discrete systems as follows:

A continuous system: "is observable if for any initial state $\mathbf{x}(0)$ and any final time $t > 0$ the initial state $\mathbf{x}(0)$ can be uniquely determined by knowledge of the input $\mathbf{u}(\tau)$ and $\mathbf{y}(\tau)$ for all $\tau \in [0, t]$".

A discrete system: "is observable if for any initial state $\mathbf{x}(0)$ and some final time k the initial state $\mathbf{x}(0)$ can be uniquely determined by knowledge of the $\mathbf{u}_{\tilde{\alpha}}$ and output $\mathbf{y}_{\tilde{\alpha}}$ for all $\tilde{\alpha} \in [0, k]$".

Appendix F

Calculation of covariance matrices

F.1 White Gaussian noise

According to [61], a noise can be classified into two large categories:

- *White Gaussian Noise:* the noise values for all pairs $[t_1, t_2]$ are independent if $t_1 \neq t_2$.
- *Coloured Noise:* the noise values are not completely independent for all pairs in the interval $[t_1, t_2]$.

In practice, WGN is a vector, in which their values are uncorrelated (covariance matrix is diagonal), following a normal distribution with mean equal to 0 and a finite standard deviation.

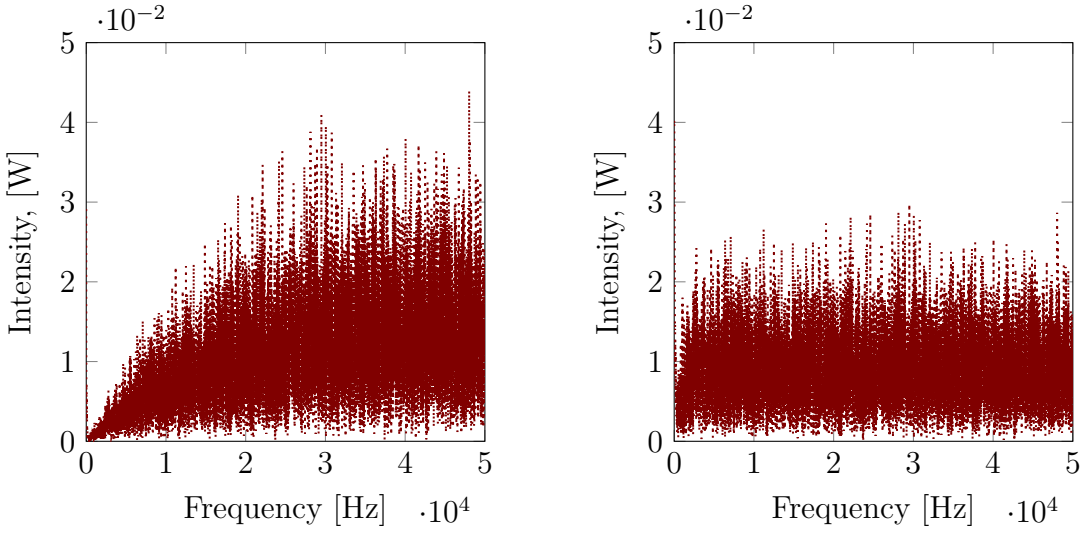
F.2 Power spectral density applied to a Kalman filter

PSD is intended to be a measure of the normalized power associated with a signal at every frequency. It can be applied to a KF, in order to check if the innovation $\mathbf{o} - \mathbf{h}$ converges to a WGN. If so, PSD has to converge to a constant power intensity at every frequency, where no trend could be appreciated.

F.3 State covariance matrix P

Term $\mathbf{P}$ is associated with the uncertainties of the vector of the n states at every step. It is assumed that errors of the states are uncorrelated, and initial value of each state $\mathbf{x}_{\tilde{\alpha}}$ equal to $P^0_{\tilde{\alpha}}$.

$$\mathbf{P}_0 = \begin{bmatrix} P_1^0 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & P_n^0 \end{bmatrix} \tag{F.1}$$



(a) Incorrect selection of values in $\mathbf{Q}$

(b) Correct selection of values in $\mathbf{Q}$

Figure F.1: PSD applied to the innovation for two different values of the plant error covariances $\mathbf{Q}$.

F.4 Plant covariance matrix Q

Term $\mathbf{Q}$ represents the uncertainties of the plant. It is positive-defined and commonly diagonal matrix that is selected to obtain a constant PSD. For instance, Fig. F.1 illustrates two different selection of covariances, in which the impact on the PSD is clearly appreciated. In the particular case, in which the state vector also contains the derivatives, the matrix $\mathbf{Q}$ can be calculated as [80]

- *Vector of states also contains first derivatives:* These covariances can be calculated using Eq. (F.2).
- *Vector of states also contains first and second derivatives:* Covariances are obtained through Eq. (F.3).

$$\mathbf{Q} \propto \begin{bmatrix} \frac{h^3}{3} & \frac{h^2}{2} \\ \frac{h^2}{2} & h \end{bmatrix} \tag{F.2}$$

$$\mathbf{Q} \propto \begin{bmatrix} \frac{h^5}{20} & \frac{h^4}{8} & \frac{h^3}{6} \\ \frac{h^4}{8} & \frac{h^3}{3} & \frac{h^2}{2} \\ \frac{h^3}{6} & \frac{h^2}{2} & h \end{bmatrix} \tag{F.3}$$

F.5 Sensor covariance matrix R

Term $\mathbf{R}$ is defined as the matrix with the covariances of the s sensors located in the system. Every sensor has a error that is assumed normally distributed with mean equal to 0 and a standard deviation σ. It is considered that sensor errors are uncorrelated to each other, thus $\mathbf{R}$ is a diagonal matrix defined as

$$\mathbf{R} = \begin{bmatrix} \sigma_1^2 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & \sigma_s^2 \end{bmatrix} \tag{F.4}$$

Appendix G

Lumped-parameter thermal models

A LPTM is a simplified model that describes the behaviour of a thermal system, where its topology consists of a set of discrete elements, such as thermal resistors, thermal capacitors and heat sources. In a LPTM requires that KCL and component equations were imposed. KCL were described in Sec. C.1 and the component equations are described next.

G.1 Thermal components

G.1.1 Heat source

A heat source represents the power heat losses or dissipation, expressed in Watts [W], due to a system component. A possible analogy with an electric current source may be considered, in which currents are replaced with heat flows, and voltages with temperatures. A possible representation of a heat source is given in Fig. G.1.

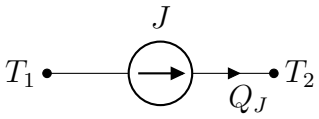


Figure G.1: A current source.

Every heat source imposes only one algebraic constraint given by

$$Q_J - J = 0 \tag{G.1}$$

G.1.2 Thermal resistor

A thermal resistor represents the resistance of material, expressed in Kelvin per Watt [K/W], how much a component resists a heat flow. A possible representation of a thermal resistor is given in Fig. G.2.

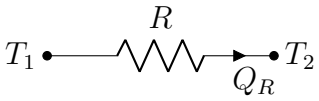


Figure G.2: A thermal resistor.

The calculation of the thermal resistance is widely explained in the bibliography [157]. For the specific case of an isotropic flat plate the expression is

$$R = \frac{e}{\kappa A} \tag{G.2}$$

where e is the thickness of the plate, κ is the material conductivity and A is the cross section of the plate. Every thermal resistor imposes one algebraic constraint given by

$$T_1 - T_2 - Q_R R = 0 \tag{G.3}$$

G.1.3 Thermal capacitor

A thermal capacitor represents the inertia of material, expressed in Joule per Kelvin [J/K], which enables to store energy, and providing an inertia against temperature transients. A possible representation of a thermal capacitor is given in Fig. G.3.

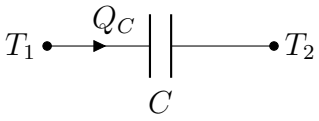


Figure G.3: A thermal capacitor.

The calculation of the thermal inertia of a material can be obtained as

$$C = mc_{\mathrm{p}} \tag{G.4}$$

where m is the mass and c_{p} is the specific heat. Every capacitor imposes a single differential equation of the form

$$C\dot{T}_1 - Q_C = 0 \tag{G.5}$$

Stored or transferred energy in a thermal capacitor can be calculated as

$$E = C\Delta T = mc_{\mathrm{p}}\Delta T \tag{G.6}$$

Appendix H

List of publications

This thesis has been financed by the HiPERFORM project which has received funding from the ECSEL Joint Undertaking under grant agreement No. 783174. Several conference papers have been presented as a result of the work made in this thesis. Moreover, two journal papers related with the Chapters 3 and 4 have been published. In addition, another journal paper presenting the results of the Chapter 5 was under preparation at the time of writing this thesis. The list of publications is exposed hereafter.

Journal papers

B. Rodríguez, F. González, M. Á. Naya, and J. Cuadrado. Assessment of Methods for the Real-Time Simulation of Electronic and Thermal Circuits. *Energies* (2020). DOI: 10.3390/en13061354.

B. Rodríguez, E. Sanjurjo, M. Tranchero, C. Romano and F. González. Thermal Parameter and State Estimation for Digital Twins of e-Powertrain Components. *IEEE Access* (2021). DOI: 10.1109/ACCESS.2021.3094312.

Upcoming journal papers

B. Rodríguez, A. J. Rodríguez, B. Sputh, R. Pastorino, M. Á. Naya, and F. González. Energy-based monitoring and correction to enhance accuracy and stability of explicit co-simulation schemes.

Conference communications

B. Rodríguez, F. González, M. Á. Naya, and J. Cuadrado. A Test Framework for the Co-simulation of Electric Powertrains and Vehicle Dynamics. In *9th EC-COMAS Thematic Conference on Multibody Dynamics*, Duisburg, Germany, July 2019.

B. Rodríguez, A. Zar, F. González, M. Á. Naya and J. Cuadrado. Use of Energy Indicators in the Explicit Co-simulation of Multibody Systems. In *Proceedings of the ASME 2020 International Design Engineering Technical Conferences &*

Computers and Information in Engineering Conference, Saint Louis, USA, August 2020.

B. Rodríguez, A. Zar, B. Sputh, M. Á. Naya, F. González and R. Pastorino. Evaluation of Indicators for the Accuracy and Stability of Explicit Co-simulation Schemes. In *COSIM 2021 - International Symposium on Co-Simulation and Solver Coupling in Dynamics*, Ferrol, Spain, May 2021.

A. Zar, F. González, B. Rodríguez, A. Luaces, M. Á. Naya and J. Cuadrado. Benchmark problems for co-simulation methods. In *COSIM 2021 - International Symposium on Co-Simulation and Solver Coupling in Dynamics*, Ferrol, Spain, May 2021.

Submitted conference communications

B. Rodríguez, A. J. Rodríguez, D. Maceira, E. Sanjurjo, U. Lugrís, M. Á. Naya, F. González and J. Cuadrado. Cyber-Physical Test Benches for Model-Based System Testing of Electric Motors. In *International Conference on Machine Design 2021*, Porto, Portugal, September 2021.

B. Rodríguez, A. J. Rodríguez, D. Maceira, F. Bottero, E. Sanjurjo, U. Lugrís, M. Á. Naya, F. González and J. Cuadrado. Development of a Cyber-Physical Test Bench for E-Powertrain Components. In *10th ECCOMAS Multibody Conference 2021*, Budapest, Hungary, December 2021.

G. Boschetti, F. González, G. Piva, D. Richiedei, B. Rodríguez and A. Trevisani. Synthesis of an Extended Kalman Filter for Cable-Driven Parallel Robots. In *10th ECCOMAS Multibody Conference 2021*, Budapest, Hungary, December 2021.

B. Rodríguez, A. Zar, F. González, M. Á. Naya and J. Cuadrado. Monitoring Energy Errors in Explicit Co-Simulation Setups. In *6th Joint International Conference on Multibody System Dynamics and 10th Asian Conference on Multibody System Dynamics*, New Delhi, India, October 2022.

Appendix I

Resumen extendido

Los ensayos de sistemas basados en modelos emergen como un nuevo paradigma de desarrollo que actualmente está ganando popularidad, especialmente en la industria automotriz. Este nuevo enfoque se centra en combinar la simulación por ordenador con la experimentación para desplazar la mayor parte de la detección de problemas y rediseños hacia las fases tempranas del desarrollo de producto. De esta forma, los ensayos de sistemas basados en modelos se centran en disminuir la cantidad de recursos invertidos en estas tareas y habilitar la identificación temprana de errores de diseño y problemas durante la operación, incluso antes de que los prototipos del vehículo completo estén disponibles. Sin embargo, el uso de esta estrategia requiere implementar algunas tecnologías críticas, tres de las cuales se abordan en esta tesis.

La primera tecnología de interés consiste en el desarrollo de herramientas de software para aplicaciones de tiempo real. En el contexto de esta tarea se ha llevado a cabo la definición e implementación de un entorno de trabajo para la evaluación de las capacidades de tiempo real, eficiencia y precisión de los métodos de simulación para circuitos eléctricos, electrónicos y térmicos. Las conclusiones obtenidas ayudan a definir los requerimientos necesarios para la simulación en tiempo real; al mismo tiempo, el desarrollo de herramientas para la simulación de circuitos es de interés para su uso en bancos de ensayos ciberfísicos de motores eléctricos. Esta propuesta de entorno de trabajo consiste en una serie de ejemplos, criterios y procedimientos para evaluar los resultados de una simulación. Los problemas seleccionados fueron definidos de una forma clara y sencilla para facilitar su replicación, y diseñados para incluir al menos un aspecto exigente de la simulación de circuitos. En el capítulo 3 se presentan nuevas formulaciones para la resolución de circuitos y se evalúa su idoneidad para tiempo real por medio del entorno de trabajo propuesto. Se emplearon distintas plataformas de software y hardware para comparar los resultados en términos de eficiencia y precisión con diferentes niveles de tolerancia, formulaciones e integradores.

En segundo lugar, las técnicas de estimación basadas en filtros de Kalman son de gran interés en aplicaciones que siguen el paradigma de ensayos basados en modelos. Estas técnicas, descritas en el capítulo 4, permiten la estimación de estados, parámetros y entradas a partir de las lecturas de sensores montados sobre el sistema físico y de su modelo digital. La estimación es de ayuda en una amplia variedad de

aplicaciones; un ejemplo es la generación y mejora de gemelos digitales de sistemas físicos. En el caso de las transmisiones de vehículos eléctricos, estos gemelos operarán posteriormente en condiciones de trabajo que requieren una predicción muy precisa del punto de operación, al tratarse de elementos críticos del automóvil, cuyo rendimiento además se ve afectado considerablemente por su estado térmico. El correcto funcionamiento de las transmisiones eléctricas requiere que las temperaturas de los puntos críticos estén controladas y se mantengan por debajo de ciertos umbrales para evitar caídas de rendimiento e incluso el fallo del sistema completo. Sin embargo, no todas las magnitudes de interés se pueden medir directamente mediante sensores colocados exactamente en las zonas más relevantes del conjunto monitorizado. Esta dificultad se puede abordar mediante el uso de sensorización virtual, es decir, empleando modelos de sistemas computacionales que tratan de representar el comportamiento de su homólogo físico haciendo uso de un estimador, por ejemplo, un filtro de Kalman, para fusionar los resultados de la simulación y las lecturas de los sensores montados en los componentes físicos. Por tanto, la estimación es un elemento clave para evitar que la simulación se desvíe de las dinámicas reales de los componentes. Este capítulo detalla un método para generar gemelos digitales de sistemas térmicos lineales, comenzando con un modelo en forma de sistema de ecuaciones algebraico-diferenciales y transformándolo en una formulación en un sistema de ecuaciones diferenciales que es más adecuado para su uso con algoritmos de estimación. La estimación obtenida se puede utilizar para realizar el ajuste inicial de los parámetros del modelo, así como las correcciones de estos parámetros, y las entradas y estados del sistema en tiempo de ejecución. El método propuesto se ha evaluado empleando un circuito térmico sencillo y el modelo térmico de un inversor trifásico. Las pruebas fueron realizadas en diversas plataformas de hardware y software, para demostrar las capacidades de tiempo real del método, necesarias cuando se utilizan gemelos digitales en aplicaciones en las que están presenten componentes reales, por ejemplo si los resultados de simulación se usan para tomar decisiones y aplicar correcciones al funcionamiento del sistema real.

En tercer lugar, el desarrollo de algoritmos de co-simulación para entornos en tiempo real es necesario para poder sincronizar y mantener estable la ejecución de sistemas ciberfísicos y se aborda en el capítulo 5. Seguir un enfoque de co-simulación implica dividir el sistema global bajo estudio en varios subsistemas y usar un algoritmo de gestión para sincronizar su ejecución y coordinar el intercambio de datos entre ellos. Desde el punto de vista del algoritmo de control, cada subsistema se comporta como una caja negra, cuyos detalles internos no quedan expuestos al resto de subsistemas, por tanto la interacción con el resto de su entorno se reduce al intercambio de un conjunto limitado de datos en momentos concretos. Por otro lado, esta estrategia permite adaptar el integrador de cada subsistema a su dinámica particular, escala de tiempo y nivel de detalle con el que se necesita realizar el modelo. Este enfoque es mucho más modular y flexible que usar una misma estrategia de resolución para todo el ensamblaje. En el contexto de esta tesis, se ha desarrollado una plataforma de software de co-simulación para probar y validar diversos algoritmos, que se ha adaptado posteriormente para permitir la interacción entre subsistemas físicos y virtuales siguiendo las especificaciones del estándar FMI. Durante la investigación llevada a cabo se han identificado tres requisitos que afectan

a los entornos de co-simulación cuando contienen componentes reales: las capacidades de tiempo real tienen que estar garantizadas para todos los subsistemas, el uso de pasos de tiempo constantes para la integración de la dinámica de los subsistemas es una necesidad práctica en la mayoría de las aplicaciones y es necesario el uso de esquemas de co-simulación explícitos, ya que no es posible repetir los pasos de integración para los componentes físicos. El uso de esquemas de cosimulación explícitos a menudo da lugar a la introducción de energía artificial en la dinámica del sistema debido al efecto que tiene la comunicación en tiempo discreto entre los subsistemas y el algoritmo coordinador de la cosimulación. Este exceso de energía degrada la precisión de la cosimulación y finalmente puede comprometer la estabilidad del proceso de integración general, a menos que sea manejado adecuadamente por el esquema de cosimulación. Con el fin de abordar este problema, se presentó un indicador para realizar un seguimiento del exceso de energía y monitorizar la precisión y estabilidad de los resultados de la cosimulación. También se implementó en la plataforma de cosimulación un nuevo método para eliminar el exceso de energía, basado en extender el propósito original del indicador de evaluación del error mencionado anteriormente. La capacidad de este método para ofrecer resultados de cosimulación explícitos estables y precisos se probó con varios ejemplos mecánicos y multifísicos.

Este trabajo de investigación concluye con un ejemplo de aplicación ciberfísica, que combina componentes reales con la simulación por ordenador de entornos virtuales: un banco de ensayos para motores eléctricos de automoción. Esta aplicación ilustra el empleo conjunto de las tecnologías abordadas en esta tesis según el enfoque de ensayos de sistemas basados en modelos. Los componentes virtuales han de ser construidos a partir de modelos capaces de ser simulados en tiempo real, las técnicas de estimación también son necesarias para el ajuste y correcta ejecución de los modelos, y la cosimulación permite una sincronización y comunicación bidireccional entre los componentes físicos y virtuales a través de un intercambio reducido de información en puntos concretos del tiempo.

El banco de pruebas propuesto ha sido diseñado para la evaluación de componentes de los trenes de potencia en automoción, en particular para motores eléctricos. La experimentación de estos componentes debe llevarse a cabo en un entorno controlado y reproducible, por ello esta instalación facilita la recopilación sistemática de datos experimentales de un modo seguro, lo que permite, por ejemplo, ensayar los componentes en situaciones peligrosas como maniobras de emergencia. Este enfoque no sería simple, ni siquiera factible de ejecutar, si se siguiera una estrategia de diseño a base de prototipos. Para el caso particular de los motores eléctricos, la evaluación de su comportamiento térmico es de gran interés para el diseño y mejora de algoritmos de control eficientes que aprovechen al máximo las capacidades de los componentes sin comprometer su funcionamiento seguro, evitando temperaturas excesivas que puedan dañarlos.

I.1 Estructura de la tesis y contribuciones

Los contenidos de estas tesis han sido organizados en siete capítulos, como se describe a continuación:

El **Capítulo 1** presenta un resumen de esta tesis y su motivación. Los objetivos de la tesis y sus principales contribuciones también se describen brevemente en este capítulo.

El **Capítulo 2** resume el estado del arte de las principales tecnologías necesarias para el paradigma de ensayos de sistemas basados en modelos, y detalla cómo este concepto encaja con las necesidades de la industria.

El **Capítulo 3** trata la evaluación de las capacidades de tiempo real del software de simulación para aplicaciones multifísicas. La principal contribución de este capítulo es la definición e implementación de un entorno multiplataforma para comparar el rendimiento de tiempo real de distintos modelos y métodos de simulación. Se desarrolló una nueva formulación para problemas eléctricos, electrónicos y térmicos que fue propuesta en este capítulo y posteriormente evaluada con la metodología propuesta.

El **Capítulo 4** introduce el concepto de gemelo digital en un sistema ciberfísico y describe como se pueden estimar las entradas, estados y parámetros en una aplicación en tiempo real usando la información proporcionada por los sensores instalados en el sistema físico. Este capítulo muestra la aplicación de este concepto utilizando un filtro de Kalman para realizar estimaciones del comportamiento térmico de los componentes de trenes de potencia eléctricos.

El **Capítulo 5** analiza los esquemas y configuraciones de la cosimulación para su uso en aplicaciones prácticas, especialmente aquellas sujetas a restricciones de tiempo real, junto con la necesidad de métodos para estabilizar y corregir las cosimulaciones explícitas. Las principales contribuciones de este capítulo son la implementación de una plataforma de cosimulación para tiempo real compatible con el estándar FMI y el uso de indicadores del error en energía para monitorizar la calidad de la cosimulación. También se presentan aquí nuevos métodos para eliminar o reducir las desviaciones de energía ocasionadas por la comunicación en tiempo discreto en la interfaz de cosimulación.

El **Capítulo 6** ilustra cómo los conceptos tratados en los capítulos 3, 4, y 5 pueden combinarse para aplicar las tecnologías del paradigma de ensayos de sistemas basados en modelos a la experimentación en bancada de componentes de trenes de potencia eléctricos. Se describe el diseño y construcción de dos bancos de ensayos para motores eléctricos siguiendo el enfoque descrito en los capítulos previos de esta tesis. Una vez estén finalizados, estos bancos servirán para la evaluación experimental de los métodos y algoritmos presentados durante esta tesis.

El **Capítulo 7** resume los principales resultados y conclusiones de esta tesis y propone posibles líneas de investigación de trabajos futuros.

I.2 Conclusiones

El ensayo de sistemas basado en modelos es un nuevo paradigma de desarrollo de producto que está ganando popularidad, especialmente en el ámbito de la automoción. Un elemento central de este nuevo paradigma es la interacción mutua entre la experimentación física y la simulación por ordenador. Este tesis se ha centrado en tres de sus tecnologías necesarias: modelado de sistemas para su simulación en tiempo real, generación de gemelos digitales y uso de técnicas de estimación, y cosimulación en aplicaciones ciberfísicas.

El uso de modelos eléctricos, electrónicos y térmicos en la simulación de componentes en automoción ha contribuido a una mejor comprensión de los fenómenos, tanto mecánicos como no mecánicos, que afectan a estos componentes. Considerar además las interacciones entre componentes desde el punto de vista del sistema en su conjunto proporciona un conocimiento todavía más completo de su funcionamiento. Esto requiere la definición de modelos multifísicos de los componentes, capaces de interactuar en tiempo real con elementos físicos del mundo real. De esta forma, se hace necesario evaluar el rendimiento de las implementaciones de software, empleando para ello diferentes plataformas de hardware y software. Esta tarea se llevó a cabo en el capitulo 3, en el que se desarrolló un software para evaluar y comparar el rendimiento de métodos e implementaciones para simular circuitos eléctricos, electrónicos y térmicos. Además, se han propuesto dos nuevos métodos para resolver circuitos, y se identificaron y trataron problemas numéricos que suelen aparecer en la integración numérica de estos sistemas.

El capítulo 4 trata sobre el concepto de gemelo digital y presenta una posible implementación y uso de esta tecnología en el desarrollo de producto a través del ensayo de sistemas basado en modelos. Los gemelos digitales constan de un sistema físico, su homólogo virtual y una comunicación bidireccional entre ambos, que hace posible el procesado de la información que intercambian. En este capítulo, se presentó un procedimiento para crear gemelos digitales que representan los efectos térmicos en componentes electrónicos. Se incorporaron técnicas de estimación basadas en filtros de Kalman a la definición del gemelo digital, pudiendo de esta manera estimar entradas, estados y parámetros del sistema durante la inicialización y ejecución, usando para ello las medidas proporcionadas por los sensores colocados sobre el sistema real. Así, a través de la combinación de técnicas de estimación y gemelos digitales, es posible determinar magnitudes que no podrían ser medidas directamente por los sensores localizados en el sistema real. De esta forma, el alcance inicial del entorno de pruebas realizado en el capítulo 3 fue ampliado para encajar este nuevo método en él.

La cosimulación es también una tecnología esencial en los ensayos de sistemas basados en modelos que permite dividir el sistema completo en diversos subsistemas que serán integrados de forma separada. En instantes concretos, estos subsistemas intercambian una cantidad limitada de información, usada por los subsistemas para sincronizar su proceso de integración numérica. En aplicaciones ciberfísicas, donde componentes reales y virtuales forman parte del mismo sistema, esto requiere la sincronización de la dinámica de sus diferentes elementos a través de una interfaz de tiempo discreto. En este sentido, puede considerarse que son un caso particular

de cosimulación, sujeta a restricciones de tiempo real. En aplicaciones prácticas, sincronizar la integración de la dinámica de los subsistemas con un gran variedad de escalas de tiempos y respuestas dinámicas es una tarea exigente, que puede abordarse con una gran cantidad de esquemas e implementaciones. Sin embargo, por lo general se utilizan esquemas de acoplamiento explícitos. Estos pueden llegar a modificar el comportamiento real de los sistemas, introduciendo o eliminando energía del sistema. Así, este fenómeno puede inestabilizar la integración numérica y hacer necesario el uso de algoritmos de corrección que permitan monitorizar y eliminar estos errores. El capítulo 5 trató y comparó diversas configuraciones de cosimulación, tanto para aplicaciones de tiempo real como para aplicaciones que no están sujetas a estas restricciones. Se desarrolló también una herramienta de software para cosimulación, multiplataforma y compatible con el estándar FMI, para coordinar la ejecución de sistemas virtuales y ciberfísicos. En cuanto a la monitorización de los errores de cosimulación, se ha incluido en esta herramienta de software un indicador para cuantificar los errores de potencia, así como un nuevo método para corregir estas desviaciones.

Por último, el capítulo 6 describe el diseño y montaje de un banco de ensayos ciberfísico para motores eléctricos. En el momento de escribir este tesis, esta instalación se encuentra aún en construcción. Los bancos de ensayos para automoción son un interesante y complejo ejemplo de aplicación ciberfísica, en la que algunos componentes físicos del vehículo se conectan a una simulación por ordenador. Como desarrollo preliminar, un banco de ensayos con motores de baja potencia fue ensamblado en primer lugar para evaluar y validar los métodos propuestos en esta tesis.

I.3 Futuras líneas de investigación

La investigación realizada en esta tesis se inscribe dentro de los proyectos activos del Laboratorio de Ingeniería Mecánica - LIM en la Universidade da Coruña (UDC). La experiencia obtenida en esta tesis se ha añadido al conocimiento del grupo y podrá aplicarse en proyectos futuros. Al término de esta tesis, se habían identificado algunas líneas posibles para continuar el trabajo realizado.

En primer lugar, la formulación dinámica y la metodología de evaluación presentados en el capítulo 3 podrían servir como plataforma de desarrollo para modelos de tiempo real, que puedan ejecutarse en plataformas de bajo consumo. Las técnicas de reducción de orden de modelos son un posible camino para mejorar sus capacidades de tiempo real, especialmente en microcontroladores, por ejemplo aquellos que pertenecen a la familia ARM Cortex-M.

En segundo lugar, la fase inicial de ajuste de los modelos LPTM para sistemas térmicos descrita en el capítulo 4 presenta algunos inconvenientes. El conjunto inicial de parámetros se determina empleando un filtro de Kalman extendido, que no siempre garantiza la convergencia de los parámetros del sistema a una solución óptima global para LPTMs de sistemas no lineales. Además, el proceso de ajuste propuesto es altamente dependiente de la topología del sistema. Una optimización global de los parámetros por medio de un análisis de sensibilidad podría ser un posible camino para el ajuste inicial del LPTM y así alcanzar un conjunto de parámetros

óptimos.

En tercer lugar, los métodos de cosimulación descritos en el capítulo 5 pueden aplicarse tanto a sistemas virtuales como a sistemas ciberfísicos. En esta tesis, se han usado esquemas explícitos debido a que los componentes físicos no pueden volver atrás en el tiempo y repetir la integración de sus pasos de tiempo. Además, los algoritmos iterativos podrían ser incompatibles con las restricciones de rendimiento de tiempo real. Sin embargo, sería interesante explorar otros esquemas de acoplamiento que no fueron estudiados en detalle, como los esquemas Gauss-Seidel explícitos, para determinar sus capacidades de mejorar la precisión y estabilidad de la cosimulación. El método propuesto de corrección de la energía necesitará adaptaciones para funcionar también con estos esquemas. De hecho, sería deseable desarrollar un método que pudiese seleccionar automáticamente una configuración de cosimulación óptima; una posible vía para esto podría ser el uso de técnicas de teoría de control o algoritmos de inteligencia artificial. También será necesario en numerosas aplicaciones ciber-físicas realizar acciones correctivas para corregir errores de comunicación y retrasos entre los componentes físicos.

Por último, el montaje y configuración del banco de ensayos para automoción descrito en el capítulo 6 es todavía un trabajo en curso. Los resultados obtenidos con esta instalación servirán para adquirir conocimiento sobre el comportamiento térmico de motores eléctricos, así como para validar los desarrollos teóricos presentados en esta tesis y los que se obtendrán en trabajos futuros.

I.4 Trabajos derivados de la realización de esta tesis

Esta tesis ha sido financiada por el proyecto HiPERFORM, que ha recibido financiación del ECSEL mediante el contrato 783174. Diversas publicaciones en congreso han sido presentadas como resultado del trabajo realizado en esta tesis. Además, se han publicado en revista dos artículos relacionados con los capítulos 3 y 4. Por último, otra publicación de revista que presenta los resultados del capítulo 5 está en preparación en el momento de escribir esta tesis. La lista de los trabajos derivados de esta tesis se expone a continuación.

Artículos de revista

B. Rodríguez, F. González, M. Á. Naya, and J. Cuadrado. Assessment of Methods for the Real-Time Simulation of Electronic and Thermal Circuits. *Energies* (2020). DOI: 10.3390/en13061354.

B. Rodríguez, E. Sanjurjo, M. Tranchero, C. Romano and F. González. Thermal Parameter and State Estimation for Digital Twins of e-Powertrain Components. *IEEE Access* (2021). DOI: 10.1109/ACCESS.2021.3094312.

Artículos de revista en preparación

B. Rodríguez, A. J. Rodríguez, B. Sputh, R. Pastorino, M. Á. Naya, and F. González. Energy-based monitoring and correction to enhance accuracy and stability of explicit co-simulation schemes.

Comunicaciones en congresos

B. Rodríguez, F. González, M. Á. Naya, and J. Cuadrado. A Test Framework for the Co-simulation of Electric Powertrains and Vehicle Dynamics. En *9th EC-COMAS Thematic Conference on Multibody Dynamics*, Duisburg, Alemania, Julio 2019.

B. Rodríguez, A. Zar, F. González, M. Á. Naya and J. Cuadrado. Use of Energy Indicators in the Explicit Co-simulation of Multibody Systems. En *Proceedings of the ASME 2020 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference*, San Luis, Estados Unidos, Agosto 2020.

B. Rodríguez, A. Zar, B. Sputh, M. Á. Naya, F. González and R. Pastorino. Evaluation of Indicators for the Accuracy and Stability of Explicit Co-simulation Schemes. En *COSIM 2021 - International Symposium on Co-Simulation and Solver Coupling in Dynamics*, Ferrol, España, Mayo 2021.

A. Zar, F. González, B. Rodríguez, A. Luaces, M. Á. Naya and J. Cuadrado. Benchmark problems for co-simulation methods. In *COSIM 2021 - International Symposium on Co-Simulation and Solver Coupling in Dynamics*, Ferrol, España, Mayo 2021.

Comunicaciones en congresos aceptadas

B. Rodríguez, A. J. Rodríguez, D. Maceira, E. Sanjurjo, U. Lugrís, M. Á. Naya, F. González and J. Cuadrado. Cyber-Physical Test Benches for Model-Based System Testing of Electric Motors. En *International Conference on Machine Design 2021*, Oporto, Portugal, Septiembre 2021.

B. Rodríguez, A. J. Rodríguez, D. Maceira, F. Bottero, E. Sanjurjo, U. Lugrís, M. Á. Naya, F. González and J. Cuadrado. Development of a Cyber-Physical Test Bench for E-Powertrain Components. En *10th ECCOMAS Multibody Conference 2021*, Budapest, Hungría, Diciembre 2021.

G. Boschetti, F. González, G. Piva, D. Richiedei, B. Rodríguez and A. Trevisani. Synthesis of an Extended Kalman Filter for Cable-Driven Parallel Robots. En *10th ECCOMAS Multibody Conference 2021*, Budapest, Hungría, Diciembre 2021.

B. Rodríguez, A. Zar, F. González, M. Á. Naya and J. Cuadrado. Monitoring Energy Errors in Explicit Co-Simulation Setups. En *6th Joint International*

Conference on Multibody System Dynamics and 10th Asian Conference on Multibody System Dynamics, Nueva Delhi, La India, Octubre 2022.