SWANSEA UNIVERSITY

Digital twinning of thermal problems in fusion energy systems

by

Wiera Bielajewa

A thesis submitted in partial fulfillment for the
degree of Doctor of Philosophy

in the

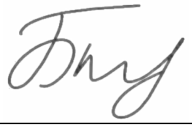
Faculty of Science and Engineering
School of Aerospace, Civil, Electrical and Mechanical Engineering

February 2026

Declaration of Authorship

I, Wiera Bielajewa, declare that this thesis titled, 'Digital twinning of thermal problems in fusion energy systems' and the work presented in it are my own. I confirm that:

- This work has not previously been accepted in substance for any degree and is not being concurrently submitted in candidature for any degree.
- This thesis is the result of my own investigations, except where otherwise stated. Other sources are acknowledged by footnotes giving explicit references. A bibliography is appended.
- I hereby give consent for my thesis, if accepted, to be available for electronic sharing.
- The University's ethical procedures have been followed and, where appropriate, that ethical approval has been granted.
- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed: 

Date: 01/03/2026

SWANSEA UNIVERSITY

Abstract

Faculty of Science and Engineering
School of Aerospace, Civil, Electrical and Mechanical Engineering

Doctor of Philosophy

by Wiera Bielajewa

Digital Twinning (DT) technology is in the process of becoming an essential instrument for optimising efficiency, ensuring safety, and enhancing research productivity of the fusion energy experimental facilities. A digital twin represents a virtual counterpart of a physical system that maintains real-time synchronisation through the sensor-derived data. Within fusion energy experimental facilities, DT technology enables enhanced information extraction from the experimental data and facilitates optimal control. This work presents two distinct methodologies facilitating DT control of a sample.

The first DT approach, Finite Element-based Digital Twinning (FE DT) is a dual-component control system comprising of a full-solution construction from limited data and a control mechanism. The solution reconstruction operates in near-real-time for small scale problems through a novel Finite Element (FE)-based data integration approach that transforms sparse measurements into comprehensive solutions for non-linear systems. This method integrates FE discretisation with a loss function minimisation to generate complete solutions from limited measurement data. Jacobian matrices are computed analytically rather than through Automatic Differentiation (AD). The control component consists of a digital (discrete) Proportional–Integral–Derivative (PID) controller that regulates the cooling water temperature of the test specimens. The modification of the solution construction approach involves its coupling with the thermal eigenvalue-based Reduced Order Modelling (ROM). This allows for the speed-up of the solution construction process whilst keeping the overall procedure the same.

The second DT approach, Physics-Driven Machine Learning-based Digital Twinning (PD-ML DT), is rooted in Machine Learning (ML). It involves training two Neural Network (NN) using steady-state data. A two-part NN system is employed, forming a control loop. The first part, the heat flux NN, addresses the challenge of estimating thermal conditions. Its aim is to construct a steady-state equivalent of the heat flux based on the provided temperature measurements. The second part, the coolant

NN, provides a solution for active thermal management. In this work, it determines the necessary coolant velocity to maintain the maximum sample temperature below a specified threshold. The additional adjustments are made to ensure the system's effectiveness and robustness when dealing with dynamic system. These include extrapolating the constructed heat flux and selecting the maximum value from a range of past and extrapolated data points. These adjustments are critical as they account for the fact that both NNs are trained on steady-state data, allowing them to operate effectively in non-steady-state conditions. This combined approach enables real-time temperature monitoring and control addressing a challenge in thermal system management.

The performance of two DT systems is demonstrated through the cooling control analysis of samples which were previously evaluated in a fusion energy experimental facility. Various system responses are generated under their control, and they are compared based on the accuracy, speed, and the resistance to measurement noise. The results show that they display advantages and disadvantages based on the information which could be provided during the experiment. Primarily, FE DT requires more measurements than PD-ML DT in the presence of noise; however, it has the ability to provide more information compared with PD-ML DT in the form of the full temperature field. As opposed with PD-ML DT, FE DT does not have instantaneous inference time; however, by employing ROM as part of the workflow it could be reduced.

The reported development of DT process for thermal systems, powered by FE combined with PID controller or a dual NN control loop, holds significant implications for the future of fusion energy research and beyond. By enabling real-time monitoring and active thermal management, it provides a crucial tool for optimising the performance and ensuring the safety of experimental fusion facilities. This approach moves the progress closer to autonomous, self-correcting systems that can operate with greater efficiency and reliability. The methodology's application extends beyond fusion, offering a template for managing thermal systems in a wide range of engineering fields where sparse measurements are involved.

Acknowledgements

Completing this thesis has been a long and, at times, challenging journey, but also a rewarding one. I am deeply grateful to have been surrounded by inspiring mentors, a supportive family, and caring friends who have walked alongside me throughout.

I would like to express my heartfelt gratitude to my supervisors, Professor Perumal Nithiarasu and Michelle Baxter. Their guidance, encouragement, and patience have been invaluable throughout the course of my research. They not only provided the academic support and expertise I needed to carry out this work, but also challenged me to think critically, to ask deeper questions, and to grow as both a researcher and an individual. I am truly grateful for their time and for the trust they placed in me to pursue my own ideas while ensuring I remained on the right path.

I would also like to extend my deepest appreciation to the teams at UKAEA, particularly Doctor Lloyd Fletcher, Doctor Adel Tayeb, and Doctor Jonathan Horne-Jones, for the fruitful discussions regarding the physical testing of the heat exchange components and the diagnostic tools employed during the experiments. Their willingness to share expertise was invaluable, and I am especially grateful for the provision of experimental data, which formed an essential part of this research.

I am also deeply indebted to my family for their unwavering love and support. To my parents, who instilled in me the values of perseverance and curiosity, thank you for always believing in me and encouraging me to pursue my passions, even when the path was not easy. To my brother and his partner, their words of encouragement kept me motivated during the most challenging moments. I have greatly cherished the time spent with them during the holidays, which provided comfort and joy along the way. I am also deeply thankful to both of them for inspiring me to draw again, re-discovering that passion has brought me balance and delight. I am also grateful to my grandparents for their support and reassurance.

To my friends and peers from the PhD journey, I owe a special thank you. They have been a source of laughter, strength, and perspective throughout this process. Whether through late-night conversations, words of reassurance, or simply being there when I needed a break, their support made this journey far more enjoyable. The shared experiences, camaraderie, and kindness kept me grounded and positive.

I would also like to thank my friends outside of the PhD for their constant support and friendship. Their presence has been a welcome reminder of life beyond research, offering balance and perspective when I needed it most. Lastly, I would like to express my deep gratitude to Anya. Her relentless pursuit of her architecture degree has been a

true inspiration to me, and our conversations about literature and art often provided a much-needed respite during busy times.

This work has been part funded by the EPSRC Energy Programme [grant number EP/W006839/1]. I would like to acknowledge the support of Supercomputing Wales and AccelerateAI projects, which is part-funded by the European Regional Development Fund (ERDF) via the Welsh Government. Furthermore, I am grateful to NVIDIA for supporting my research by providing NVIDIA RTX 6000 Ada GPU grant. This thesis would not have been possible without these invaluable contributions.

Such days may meet you just before the springtime:
Beneath the snow the meadow lies in peace;
The treetops wobble with a dainty rhythm;
Benevolent and balmy is the breeze.
And you can feel the lightness in your body,
And you do not quite recognize your home,
And eagerly you find yourself intoning
That song you thought you'd tired of long ago.

— Spring, 1915, Anna Akhmatova (translated by Robin Kallsen)

Contents

List of Figures

List of Tables

Abbreviations

AC Alternating Current

AD Automatic Differentiation

AE Autoencoder

AH Åström–Hägglund

BC Boundary Condition

CB Craig–Bampton

CFD Computational Fluid Dynamics

CG Conjugate-Gradient

CHIMERA Combined Heating and Magnetic Research Apparatus

CPU Central Processing Unit

DNN Deep Neural Network

DOFs Degrees of Freedom

DT Digital Twinning

EM Electromagnetics

FDM Finite Difference Method

FE Finite Element

FE DT Finite Element-based Digital Twinning

FEM Finite Element Method

GPU Graphical Processing Unit

GT Ground Truth

HHF High Heat Flux

HIVE Heat by Induction to Verify Extremes

HM Hybrid Modelling

HMC Hamiltonian Monte Carlo

IC Initial Condition

ITER International Thermonuclear Experimental Reactor

JET Joint European Torus

LHFM Low to High Fidelity Modelling

LSTM Long Short-Term Memory

MAPE Mean Absolute Percentage Error

MCMC Markov Chain Monte Carlo

ML Machine Learning

MLP Multi-Layer Perception

MSE Mean Squared Error

NF Normalising Flow

NLP Natural Language Processing

NN Neural Network

NRMSE Normalised Root Mean Squared Error

ODIL Optimising a Discrete Loss

ONB Onset of Nucleate Boiling

PD-ML DT Physics-Driven Machine Learning-based Digital Twinning

PDEs Partial Differential Equations

PFC Plasma-Facing Components

PID Proportional–Integral–Derivative

PINNs Physics-Informed Neural Networks

POD Proper Orthogonal Decomposition

PSO Particle Swarm Optimisation

Pyvale Python Validation Engine

RBF Radial Basis Function

RL Reinforcement Learning

RMSE Root Mean Squared Error

RNN Recurrent Neural Network

ROM Reduced Order Modelling

SOL Scrape-Off Layer

TC Thermocouple

UK United Kingdom

UKAEA UK Atomic Energy Authority

VAE Variational Autoencoder

VL VirtualLab

ZN Ziegler–Nichols

List of Symbols

D Deuterium

G Heat source

He Helium

M Set of mesh nodes where temperature measurements are recorded

N_i Shape (interpolation) function

Pr Prandtl number

Re Reynolds number

T Temperature

T_M Temperature values vector at mesh nodes belonging to a measurement set M

T_a Ambient temperature

T_i Temperature value at mesh node

T_{ONB} Temperature during onset of nucleate boiling

T_{sat} Saturation temperature

T_{wall} Pipe wall temperature

Tr Tritium

Γ_h Convection domain boundary

Γ_q Heat flux domain boundary

Ω Problem domain

$[\boldsymbol{I}]$ Identity matrix

$[\boldsymbol{J}]$ Jacobian matrix

$[\boldsymbol{K}]$ Global stiffness matrix

$[\boldsymbol{M}]$ Global capacitance matrix

$[\boldsymbol{R}]$ Reduction matrix

$[\boldsymbol{\Phi}]$ Matrix containing eigenmodes

$\{\boldsymbol{L}\}$ Loss vector

$\{\boldsymbol{N}\}$ Global shape (interpolation) function vector

$\{\boldsymbol{f}\}$ Global loading vector

$\{\boldsymbol{f}\}_q$ Part of the global loading vector $\{\boldsymbol{f}\}$ corresponding to the applied heat flux q

$\{\boldsymbol{f}\}_{g-q}$ Part of the global loading vector $\{\boldsymbol{f}\}$ excluding $\{\boldsymbol{f}\}_q$

$\{\boldsymbol{n}\}$ Vector normal to Γ_q

$\{\boldsymbol{v}\}$ Vector of unknown parameters

$\{\boldsymbol{T}\}$ Temperature distribution vector on the nodes of a mesh

$\{\boldsymbol{T}\}_{con}$ Temperature distribution vector constructed from temperature measurements

μ_0 Vacuum's magnetic permeability equal to $4\pi \times 10^{-7} NA^{-2}$

μ_r Conductor's relative magnetic permeability

ρ Density

θ Time discretisation scheme parameter

$\widetilde{\{\boldsymbol{T}\}}$ Reduced temperature distribution vector obtained from $\{\boldsymbol{T}\}$

c_p Specifc heat

d_{pipe} Pipe's diameter

f Electric current's frequency

h Convection heat transfer coefficient

k_x Thermal conductivity in x direction

k_y Thermal conductivity in y direction

k_z Thermal conductivity in z direction

l_c x direction cosine

l_{pipe} Pipe's length

m_c y direction cosine

n n^{th} the time step (instance)

n_c z direction cosine

n_n Number of mesh nodes

ne Neutron

q Heat flux

q_{con} Heat flux on Γ_q constructed from $\{\boldsymbol{T}\}_{con}$

t Time

x First spacial direction

y Second spacial direction

z Third spacial direction

Chapter 1

Introduction

1.1 Fusion energy

Sustainable and dependable energy sources are fundamental to the world's continued advancement. Addressing the dual challenges of the increased energy demand and climate change, exacerbated by fossil fuel usage, requires a diverse mixture of strategies. Controlled thermonuclear fusion offers several critical benefits on the way to addressing these challenges. It could contribute towards sustainability, safety, and energy security, and it could become a stable electricity provider, contributing reliably to other energy systems [1].

Fusion energy differs significantly from nuclear fission, which is the technology used in current nuclear power plants. Fission relies on a self-sustaining chain reaction that can lead to runaway scenarios if not carefully controlled. In contrast, fusion does not involve chain reactions. Continuous external input is required in order to maintain the conditions for fusion, and deviations from the optimal conditions causes the plasma to cool and the reaction to halt naturally. Furthermore, fusion reactions produce significantly less lasting radioactive waste and do not involve materials bearing resemblance to the military-grade ones (e.g. Plutonium and low-enriched Uranium), thus enhancing both its environmental and safety profiles. Finally, its fuel sources are abundant and effectively inexhaustible.

Unlike fission process which is based on the splitting of heavy atomic nucleus into two lighter ones, fusion energy is produced by fusing two light nuclei into one heavier nucleus [8]. Typically, in a fusion reactor hydrogen isotope, deuterium D, reacts with

tritium T, which is a heavier hydrogen isotope, to produce a neutron n, helium He, and approximately 14.1 MeV of energy from the neutron [1, 9]:

$$D + T \rightarrow He^4\ (3.5MeV) + n\ (14.1MeV) \tag{1.1}$$

The isotopes are combined under conditions of extreme temperature to form a heavier nucleus. The majority of the produced energy is carried away by the neutron. The energy of these neutrons is then absorbed by the blanket material surrounding the plasma, where it is converted to heat and subsequently used to generate electricity via standard thermodynamic cycles [1].

A fusion reactor is an integrated system designed to achieve and sustain thermonuclear conditions conducive to net energy gain. The most mature concept is the tokamak. It is a toroidal magnetic confinement device that employs strong toroidal and poloidal magnetic fields used to confine and stabilise a plasma with temperatures exceeding $10^8\ ^\circ C$ [9]. The alternative for the traditional toroidal arrangement is spherical tokamak, which has a lower aspect ratio and could lead to improved stability [10].

Figure 1.1 show the example of future fusion plant configuration. A deuterium-tritium fuel mixture is injected into a vacuum chamber in gaseous form or as frozen pellets, where confinement and heating systems bring it to the plasma state and sustain continuous fusion reactions [11]. The plasma produces energy in the form of energetic particles, radiation, and high-energy neutrons, as well as helium ash, which is extracted and processed outside the reactor. Charged particles and radiation deposit their energy in the first wall surrounding the plasma, while neutrons travel beyond the first wall into the blanket. Most of the neutron energy is deposited in the blanket, where it is converted into thermal power and removed by a coolant circulating within the structure. This heat is transferred to steam generators to produce steam that drives a conventional turbine-generator system for electricity production. A fraction of the generated electrical power is recirculated to sustain plasma operation and supply auxiliary systems. The first wall, blanket, and vacuum vessel are actively cooled by the heat extraction system to enable steady-state operation under fusion-relevant conditions.

A fusion reactor consists of several key systems that work together to achieve and sustain nuclear fusion. The plasma confinement system uses strong toroidal magnetic fields produced by superconducting magnets to contain the hot plasma and prevent it from

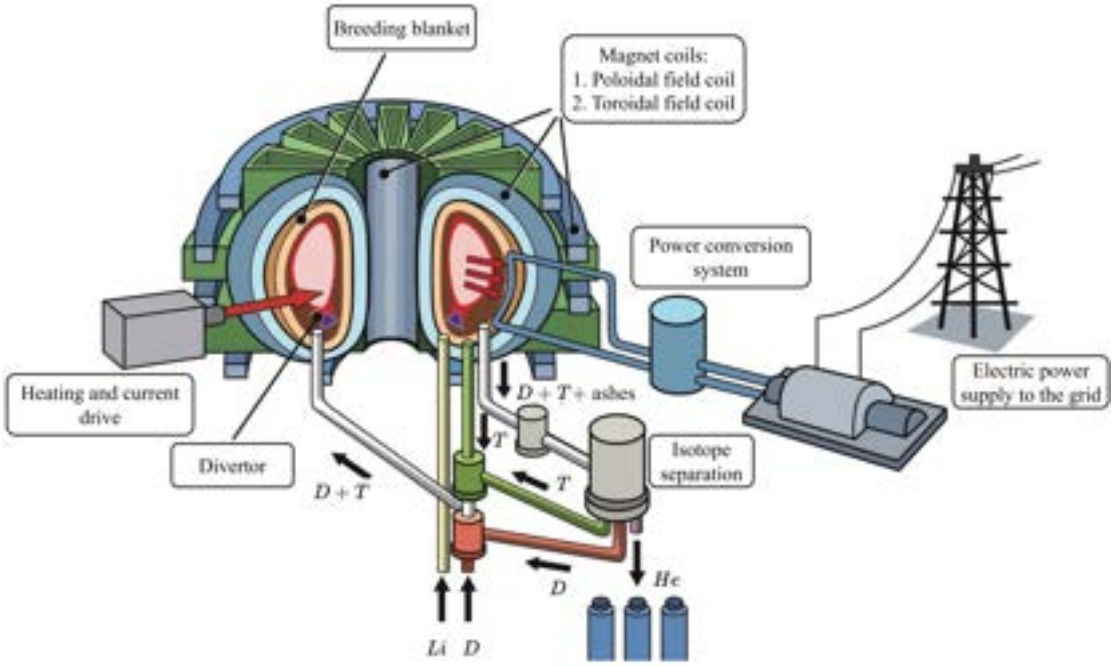


FIGURE 1.1: Fusion power plant example configuration depicted with some auxiliary systems.

coming into contact with the reactor walls. This system ensures that the plasma remains at the high pressures and temperatures needed for fusion. To heat the plasma to the necessary fusion conditions, dedicated heating systems such as neutral-beam injectors are employed. These systems raise the energy of the particles and enable collisions strong enough to overcome the Coulomb barrier. Meanwhile, the fuel injection system introduces deuterium and tritium into the plasma, keeping the fuel supply steady and the plasma density within optimal limits. Deuterium and lithium are abundant resources. Deuterium can be extracted from any type of water and is regularly manufactured for use in scientific and industrial fields.

Other components include the blanket and shielding, which capture high-energy neutrons produced during fusion. Deuterium and tritium fuel undergo nuclear fusion at extremely high temperatures, releasing energy in the form of charged particles, neutrons, X-rays, and ultraviolet radiation. These neutrons interact with lithium within the blanket to breed more tritium, sustaining the fuel cycle. The blanket also absorbs the energy from the neutrons, converting it into heat that can be used to generate electricity. The divertor is positioned at the bottom of the vacuum vessel. It defines the outer boundary of the confined plasma, or plasma edge, and serves several essential functions. Chief among these is the removal of waste products, particularly helium ash produced during fusion,

as well as other impurities that can accumulate in the plasma. By maintaining plasma purity and regulating excess heat, the divertor ensures continuous and stable fusion conditions. The area allocated for the divertor cavity also accommodates other in-vessel systems not covered by the blanket. Lastly, the cooling system regulates the intense heat generated by the fusion process through coolant loops and heat exchangers. This system not only facilitates the transfer of thermal energy to electricity-generating systems but also helps maintain the reactor's structural integrity and efficiency by cooling magnets and other key components.

Figure 1.2 shows the distribution and management of heat loads within the fusion reactor, using EFDA DEMO 2050 as an example [1]. The reactor is designed to produce 2GW of fusion power. An additional 50MW is supplied by external heating systems, bringing the total power that should be exhausted as heat by the reactor systems to 2.05GW. Of this, 78% (1.6 GW) comes from the kinetic energy of neutrons that leave the plasma core without interacting further. This neutron energy is split between the reactor's blanket structures, which absorb 80% of it, and the divertor region, which receives the remaining 20%. The other 22% (0.45GW) of the total power is associated with the energy of alpha particles (400MW) and the power supplied by external heating systems (50MW). This energy eventually exhausts from the plasma. 240MW is released as radiation in the form of high-energy photons, while the rest is carried by the plasma flow to the Scrape-Off Layer (SOL). Specifically, 210MW transported by the high-energy protons are absorbed by Plasma-Facing Componentss (PFCs), resulting in an average heat flux of 0.175 MW per square meter. The remaining energy interacts with localised areas of the first wall or is deposited onto the divertor, which is designed to handle concentrated heat and impurities.

The development of the commercial fusion reactor capable of producing a net energy gain requires extensive validation of its components, plasma control techniques, and materials capable of withstanding extreme thermal loads. Several large-scale experimental facilities have been constructed or are under development to advance this goal. The Joint European Torus (JET) in the United Kingdom (UK) has achieved a record for the highest energy output from a controlled fusion reaction during its final experiments in 2024 [12]. The Spherical Tokamak for Energy Production (STEP) seeks to develop a UK prototype fusion power plant and establish a pathway toward the fusion energy's commercialisation [13]. Its goals include demonstrating net energy gain, self-sufficiency

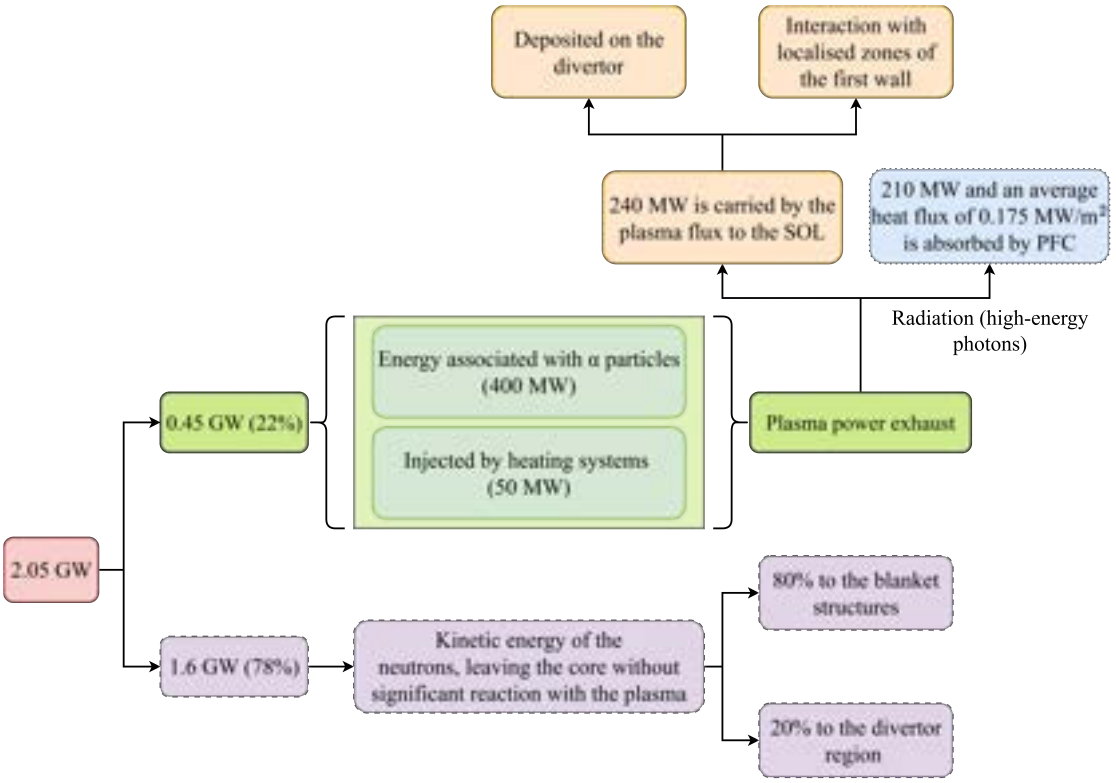


FIGURE 1.2: Heat load distribution within the fusion reactor based on EFDA DEMO 2050 [1].

of fuel, and practical approaches to plant maintenance. International Thermonuclear Experimental Reactor (ITER), under construction in France, is the largest fusion experiment to date. It aims to demonstrate a gain factor of at least 10, meaning it should produce ten times more fusion power than the power used to keep the plasma in a steady state [14]. In parallel, inertial confinement is being explored at the National Ignition Facility (NIF) in the United States, which achieved energy gain in 2022 [15]. Experimental Advanced Superconducting Tokamak (EAST) was established in China [16], and Burning Plasma Experimental Tokamak (BEST) is under development [17]. Finally, companies such as Tokamak Energy and First Light Fusion are rapidly developing in the UK private sector.

1.2 Fusion energy experimental facilities and Digital Twinning

The research into fusion energy technology depends on various fusion energy experimental facilities. While large-scale projects like ITER and JET focus on plasma confinement

and energy production, a number of facilities aim to address the engineering and material performance challenges associated with operating a fusion power plant. Among these, HIVE [18–20] and Combined Heating and Magnetic Research Apparatus (CHIMERA) [21] facilities are designed to investigate the reactor component prototypes, particularly the ones designed to perform a plasma-facing function (such as divertor components). HIVE is capable of subjecting the sample to high heat fluxes and temperatures in vacuum conditions. While CHIMERA will be able to subject the samples to high heat fluxes combined with magnetic loads under air, vacuum, or inert gas environments.

DT has emerged as a vital tool aiming to maximise the efficiency, safety, and research output of such facilities. A digital twin is a virtual replica of a physical system with a real-time synchronisation achieved through the data collected from the sensors [22]. In the context of fusion energy experimental facilities, DT can be used to extract more information from the experimental measurements and control the physical asset in an optimal manner. Furthermore, DT might help operators regulate test conditions and observe safety constraints.

1.3 HIVE facility and samples

The HIVE facility is a high heat flux experimental system developed at the UKAEA site in Culham, Oxfordshire to support research and development of plasma-facing components. It operates under vacuum and uses induction heating generated by an induction coil to reproduce the thermal conditions experienced during steady-state fusion reactor operation. Surface heat fluxes of up to $20 MW/m^2$ are applied to the sample. Excess heat is removed via an actively cooled, pressurised water loop connected to the test sample.

The experimental setup is housed within a cylindrical vacuum vessel approximately 500mm in height and diameter, featuring multiple ports for vacuum pumping, diagnostics, and optical access. The sample is mechanically clamped at the centre of the vessel. A range of sample materials has been tested, including tungsten-copper and stainless steel configurations. Sample length is 50mm, while the height is 25mm for the tungsten-copper sample and 35mm for the stainless steel one.

Heating is achieved through electromagnetic induction. An alternating current in the coil generates a time-varying magnetic field, inducing eddy currents within the electrically conductive sample. Resistive losses associated with these currents lead to Joule heating. Although induction heating is volumetric in principle, the induced currents are concentrated near the surface of the conductor due to the skin effect. The characteristic skin depth depends on the material resistivity, magnetic permeability, and excitation frequency, and decreases with increasing frequency.

HIVE typically operates in the 50-150 kHz range, around 100 kHz, corresponding to a skin depth of less than 0.5mm. As a result, the thermal loading can be approximated as a surface heat flux, closely replicating fusion-relevant conditions. The induction system can draw up to 45kW of electrical power. Although, the actual heat transferred to the sample depends on the coupling efficiency between the coil and the conductor, which decreases with increasing coil-sample separation.

The HIVE facility was selected as a representative platform for developing and assessing DT workflows due to its ability to reproduce key thermal conditions relevant to PFCs, such as those found in divertors. The application of high heat fluxes combined with active cooling to the samples closely mirrors the operational environment and material challenges encountered in fusion systems. Moreover, these extreme thermal conditions make HIVE particularly well suited for DT approaches. Experimental measurements in such environments are often spatially and temporally limited, while direct access to the sample is constrained. A DT enables the extraction of additional information from available measurements, improving state estimation, interpretation of experimental data, and ultimately control of the heating process. This makes HIVE an ideal test-bed for demonstrating the value of DT in fusion-relevant thermal experiments.

1.4 Aims and objectives

The primary aim of this thesis is to develop and evaluate DT methodologies for monitoring and controlling the behaviour of samples under experimental conditions within the HIVE facility.

The objectives of this thesis are as follows:

1. To construct and demonstrate operation of a FE DT of a representative component prototype, capturing thermal responses under various loading scenarios and enabling physics-based temperature monitoring and control.

2. To develop and demonstrate operation of a PD-ML DT of a representative component prototype, enabling data-driven temperature monitoring and control.

3. To compare the performance of both DT approaches with respect to accuracy (control error), computational efficiency (real-time or near real-time operational speed), and robustness to uncertainties (sensitivity to noise).

1.5 Thesis outline

This thesis is structured into six chapters, including this introduction. They aim to provide a logical progression from the theoretical foundations to the implementation and evaluation of aforementioned DT methodologies.

Chapter 2: Background information and literature review

This chapter provides the necessary background on inverse modelling and reviews various available DT techniques.

2.1 Inverse modelling methods: A review of approaches for constructing system states from measurements and inverse material properties derivation, with emphasis on thermal problems.

2.2 DT methods for thermal systems: A survey of existing DT frameworks, with comparison between physics-based and data-driven approaches. The focus of the review is on various thermal systems.

Chapter 3: FE and ML-based DT

This chapter outlines the two DT frameworks developed in this work. It describes the mathematical formulations and workflows allowing for temperature control within the sample.

3.1 FEM: Theory behind FEM for thermal simulations.

3.2 FE DT: Construction of a physics-based DT model using FE analysis and a PID controller.

3.3 PD-ML DT: An approach using two ML models trained on FE data.

Chapter 4: Solution and thermal conductivity construction using FE-based approach

This chapter focuses on the implementation and validation of the FE-based construction process.

4.1 Forward model: Description of the forward FE model used to test the solution and thermal conductivity construction process, the calibrated DT control loops, and to generate training and validation datasets for PD-ML DT.

4.2 Solution construction using computational data: Testing and analysis of the FE-based solution construction process using computational data.

4.3 Thermal conductivity construction using computational data: Testing and analysis of the FE-based thermal conductivity construction process using computational data.

4.4 Solution construction using experimental data: Testing and analysis of the FE-based solution construction process using experimental data.

Chapter 5: Temperature monitoring and control

This chapter focuses on the calibration and testing of two DT control loops. It also presents the application of both DT control loops in the context of temperature monitoring and control. It evaluates their accuracy, computational performance, and sensitivity to noise. Comparative results between the FE DT and PD-ML DT approaches are discussed in detail.

5.1 Finite Element-based Digital Twinning control loop tuning: Calibration process of FE DT. It employs two methods, ZN and AH.

5.2 Physics-Driven Machine Learning-based Digital Twinning control loop: Training and calibration processes for PD-ML DT. It includes the training of two NNs and the tuning of two hyperparameters used in the assembled PD-ML DT.

5.3 Testing parameters for Digital Twinning control loops: Description of the testing parameters used to generate the control results. It includes the various noise and applied heat flux settings.

5.4 Temperature control results analysis: The analysis of the results achieved using the testing parameters defined in the previous section.

Chapter 6: Conclusions and future work

This final chapter draws conclusions and discusses future work.

Chapter 2

Background information and literature review

Digital twins and the process of DT are gaining widespread adoption across many engineering disciplines. This growth is driven by the need to develop responsive virtual models of physical systems that can extract valuable real-time insights from physical assets. A DT refers to a dynamic, continuously updated virtual model that mirrors the behaviour of its real-world counterpart using real-time data [22]. In contrast, DT encompasses the broader methodology of creating and maintaining these models, especially important when active control of the physical system is involved.

In the context of experimental facilities, DT serves two primary roles: enhancing the limited measurements obtained from diagnostic sensors (i.e., system monitoring) and supporting the control of experiments to reach desired conditions efficiently (i.e., system control). Achieving both of these goals requires a robust inverse analysis framework capable of integrating experimental data into simulations in real time to reconstruct a system state representation. This is particularly critical in scenarios where essential quantities, such as the maximum temperature within a test sample, cannot be directly measured from sparse data alone.

This chapter presents an overview of inverse modelling techniques and DT approaches.

2.1 Inverse modelling methods

In general terms, a forward problem involves applying a known physical model to determine the outcomes resulting from specified inputs or causes. In the case of transient problems, solving the forward problem requires specifying BCs, Initial Conditions (ICs), material properties, and potentially other system parameters. In contrast, inverse problems generally fall into two main categories: (1) identifying system parameters based on observed inputs and outputs, and (2) reconstructing system causes from observed outcomes or effects.

The first category reflects the classical definition of inverse problems, where unknown parameters are inferred [23]. The second category, on the other hand, focuses on reconstructing BCs and thus the complete solution field within a domain using limited or sparse measurements, assuming that the system parameters are already known. This type of inverse problem often arises in physical experiments, which are the primary source of sparse observational data. Historically, inverse problems have primarily centred on parameter estimation within the context of Partial Differential Equations (PDEs) [24–26].

From an engineering standpoint, data assimilation and inverse modelling, specifically of the second type, are closely related processes. Both involve integrating sparse observational or experimental data into a numerical model to enhance the available information using established physical laws. These observed data may consist of material properties or measurable variables such as velocity, pressure, displacement, or temperature. The objective may be to determine unknown material parameters and/or reconstruct the complete spatial field of a physical variable. Direct measurement of material properties is uncommon, with the exception of certain cases like mechanical deformation, where stress–strain data can be directly used to infer material characteristics. More typically, measurable physical variables are recorded, and the goal is to infer the associated unknown or uncertain properties of the material. Data assimilation and inverse modelling may differ slightly in terms of their underlying assumptions. Inverse modelling often presumes parameters are completely unknown, and data assimilation may treat them as uncertain but partially known. Data assimilation could be aimed at reducing uncertainty by refining model parameters with measurement data. In this work, the terms data assimilation and inverse modelling are used interchangeably.

The primary challenge of inverse problems lies in their ill-posedness, meaning that solutions may not be unique. This ill-posedness necessitates the use of regularisation techniques and careful selection of inversion methods. Given that there are often multiple possible ways for incorporating data into a model, some form of regularisation or prior information is typically necessary to constrain the solution space. Regularisation can be explicit, such as by introducing assumptions about the unknown parameters in the form of prior, or it could be implicit. Implicit regularisation may be due to using, for example, a set of carefully selected forward simulation cases as training data for a ML model. Some authors argue that purely data-driven methods relying solely on data without explicit assumptions eliminate the need for prior knowledge [27]. However, this viewpoint overlooks the fact that large volumes of high-quality data can themselves act as a form of regularisation. In effect, the data implicitly encode prior information by constraining the solution space and filtering out implausible outcomes.

2.1.1 Methods without Machine Learning usage

Classical approaches to solving these problems include functional analytic and statistical regularisation techniques [23, 24], with Bayesian inversion being one of the most prominent examples of statistical regularisation [24]. The posterior distribution over the unknown parameters is found through the process of updating the prior distribution with the measured data. Since the posterior distribution can rarely be calculated explicitly, posterior sampling techniques based on the repeated forward evaluations are usually used, such as Markov Chain Monte Carlo (MCMC) [28], Hamiltonian Monte Carlo (HMC) [29], Variational Inference (VI) [30], and Normalising Flows (NFs) [30]. Girolami et al. [31] adopt a Bayesian view of the inverse problem with MCMC being used to sample the posterior distributions. The parametrised probabilities, associated with the uncertainties present in the model definition and measured data, are derived. Then the measured data can be used to determine or learn the values of parameters defining these probabilities.

Kalman Filter together with its variations could be used for data assimilation as well. Duffin et al. [32] attempted to reduce the computational complexity of the method introduced by Girolami et al. [31]. It was done by utilising some approximations and

applying the extended Kalman filter instead of more computationally demanding algorithms such as MCMC. Habibi et al. [33] employs a combination of sequential Kalman filter and Dynamic Mode Decomposition (DMD) ROM technique to assimilate blood flow data into the computational model.

Another category of solutions involves search and optimisation algorithms, such as Particle Swarm Optimisation (PSO) method [34]. Comprehensive reviews of inverse problem-solving strategies can be found in the works of Tamaddon-Jahromi et al. [35] and Arridge et al. [24]. However, many of these traditional methods tend to be computationally intensive and often lack the flexibility required for modern engineering applications.

Optimising a Discrete Loss (ODIL) approach [36, 37] proved to be much more efficient than Physics-Informed Neural Networkss (PINNss) which are described in the subsequent sub-section. ODIL combines discretisation methods like finite volume (FV) with ML optimisation. It uses AD for Jacobian matrices and represents unknown material properties with NNs. This work, therefore, modifies ODIL for the FEM to be applied as a component for FE DT. A new, FEM-specific regularisation term is introduced into the loss function, which is used alongside Neumann BCs. The loss function gradients are derived analytically, and the material properties are modelled as a piecewise linear function.

2.1.2 Methods with Machine Learning usage

ML has gained significant traction across a range of engineering disciplines, from aerospace [38, 39] to manufacturing [40, 41]. ML techniques offer several advantages over the approaches described in the previous sub-section, with efficiency, scalability, and accuracy being among them [42]. These benefits make them attractive for a wide variety of problems.

Purely data-driven ML models typically require large datasets to achieve acceptable accuracy and sufficient generalisation. While models such as Gaussian Process Regression (GPR) [43] can perform well with less data compared to NNs [44, 45], Long Short-Term Memory (LSTM) networks [5], or Transformers [46, 47], the data requirements for complex engineering problems still remain high. Appendices A and B compare Transformer-based ML models with LSTM for transient thermal solution construction.

Data acquisition in such contexts is often difficult, and experimental data tend to be sparse and insufficient for robust training. One workaround involves using numerical simulations, such as FE models, to generate synthetic training data [19, 35, 46, 48, 49]. Pawar et al. [50] and Kim et al. [51] used Autoencoder (AE) - Generative Adversarial Network (GAN) combination [50] and Gappy AE [51], respectively, to directly perform a solution reconstruction utilising purely data-driven ML. However, this approach hinges on performing a high number of simulations, which could be time-consuming. Moreover, training ML models on selective simulation results may inadvertently introduce bias, potentially leading to discrepancies between predictions and real-world measurements. To address some of these issues, physics-based and hybrid ML approaches have been developed.

PINNs [52, 53] do not rely on pre-existing datasets but are instead trained using the governing PDEs. This helps avoid the bias associated with purely data-driven models. However, PINNs come with their own drawbacks. One of the key advantages of ML in DT is fast inference once the model is trained, making it suitable for real-time applications. PINNs, on the other hand, require continuous training as new sensor data are introduced, which significantly slows down real-time performance. Integrating PINNs into traditional industrial simulation pipelines, which are often based on conventional FEM software, is also not straightforward and presents a practical challenge. Nevertheless, significant progress is being achieved in this realm. Gao et al. [54] develop a discrete PINNs by performing conventional FE discretisation, merging it with Graph Convolutional Network (GCN), and then minimising the objective function similarly to the standard PINNs. Wang et al. [55] present another variation of discretised PINNs. The domain is discretised using a FE mesh ensuring that the predicted values align to the FEM constraints; the objective function limited by the FE mesh and the NN parameters are obtained.

Various Low to High Fidelity Modelling (LHFM) and Hybrid Modelling (HM) are an alternative to the aforementioned methods. LHFM generally means creating a link between what is considered to be a low fidelity data and what is considered to be a high fidelity data. For example, low-fidelity data can be simulations, while high-fidelity data is experimental data. Alternatively, low-fidelity data might be 2D simulations, with high-fidelity data being 3D simulations. On the other hand, during HM not only the model parameters related to the material properties or variables are updated based on

measured data, but the whole model structure is modified. ROM is generally intertwined with the notions of LHFM as well as HM. Xu et al. [28] created a ROM using Variational Autoencoder (VAE) and then employed Bayesian inversion coupled with MCMC sampling to inversely determine material properties. While Wu et al. [30] and Li et al. [29] created a ROM using Proper Orthogonal Decomposition (POD) combined with a NN and performed Bayesian inversion using NFs and HMC, respectively. Shahzadi and Soulaimani [56] created a ROM using NN and then used various optimisation algorithms, such as PSO, genetic algorithm, and differential evolution, to minimise the objective function related to the measured data. The studies conducted by Xu et al. [57] and De et al. [58] are examples of ROM, which at the same time is LHFM and HM. Low-fidelity data is FEM simulations, while high-fidelity data is measured data. One NN is trained offline using FEM data first; and then the second NN, built on top of the first one, is trained online using only measured data, while the parameters of the first NN remain unchanged.

The aforementioned LHFM and HM approaches use FEM simulations to create a training dataset for supervised ML models. However, there are also several works in which FEM is used more interactively within the ML framework. Li et al. [59] used a FE system of equations as a filter after AE. The equivalent load vector outputted by AE is inputted into FE system of equations which then calculate the deflection. The calculated deflection can subsequently be compared with the measured data during unsupervised learning process. The study by Askari et al. [60] is an example of HM that is not a ROM at the same time as well. Similar to Li et al. [59], the FE system of equations acts as a filter for the output of NN or recurrent NN. The difference is that in this case the ML model predicts the values for some unknown material property as well as some unknown physics, which makes it HM. The authors state that the training process is supervised; however, similar to Li et al. [59], it perhaps can also be classified as unsupervised as only the measured data is used during the training. Analogous to the works by Li et al. [59] and Askari et al. [60], Meethal et al. [61] introduce a ROM workflow where FE system of equations acts as a filter for a NN output, and the model is also trained in essentially an unsupervised manner. This approach can also be described as HM, as various unknowns apart from material properties can be parametrised and incorporated into the training process.

FEM has conventionally been limited to forward problems, where all BCs are known.

It is often regarded as difficult to apply directly to inverse problems due to missing or unknown boundary information. However, the FE-based inverse analysis framework developed in this work demonstrates that the standard FE workflow can be adapted to handle inverse tasks effectively, allowing for accurate reconstruction of system states even in the absence of complete input data.

2.2 Digital Twinning methods for thermal systems

What primarily sets a DT apart from a stand-alone simulation running alongside a physical process is the ongoing exchange of information between the virtual model and the physical asset. This interaction occurs through measurement data collected by sensors, enabling the digital twin to accurately reflect the current state of the physical system. This real-time feedback not only improves the fidelity of the model but also supports more effective system control. Given its capacity to continuously mirror the behaviour of a physical asset, a digital twin can be particularly valuable for monitoring and managing complex thermal systems. These may include experimental platforms designed to investigate heat transfer under extreme conditions or complete energy systems like power plants. In practical applications, the ability to track and regulate factors such as peak temperatures, mechanical strain, or cooling rates is crucial to prevent material degradation and ensure safe, efficient operation. Various inverse modelling and ROM approaches described in the previous section play a significant role in enabling the creation of the DT process.

The concept of DT has been established for several decades and has been widely implemented in fields such as healthcare [62, 63], as well as manufacturing [64, 65] and aerospace [66, 67] engineering. However, the use of DT technology in heat transfer problems is a much more recent advancement. This field is still in its early stages, with many technical and methodological challenges yet to be resolved [68]. In several studies, a heat transfer digital twin is not conceived as a fully integrated system involving continuous data exchange between the physical asset and its virtual counterpart, which is a central requirement of the definition adopted in this work. Instead, these studies focus on numerical simulations that emulate experimental conditions without establishing a bidirectional or continuous connection with the physical system. Moreover, the

simulations are not designed to operate at real-time or near real-time speeds, which further differentiates them from the digital twin framework considered here. This is shown further down in this section. The DT framework proposed in this work also enables the virtual model to exert control over the physical system, thereby forming a closed-loop interaction, a capability that is not addressed in a lot of the previously reported studies.

Spateri et al. [70] proposed an electrothermal digital twin for radiation-based thermoforming processes. It integrates lumped-parameter heater modelling, a FDM thermal diffusion model of the polymer sheet, and data-driven identification of radiation view factors. The method decomposes the heating process into electric power generation, radiative transfer, and through-thickness heat diffusion. It then couples physics-based models with experimentally calibrated parameters obtained via constrained optimisation. Validation on a laboratory thermoforming test bench demonstrates good agreement between predicted and measured surface temperatures, while also enabling the estimation of internal bulk temperatures. This DT offers reduced computational complexity and modularity, potentially allowing it to function in real time. However, real-time execution was not demonstrated bySpateri et al. [70]. The model parameters (e.g. radiation view factors) were identified using experimental measurements, but this was performed offline during calibration rather than continuously during operation. Spateri et al. [70] discussed control of the physical system only as a future application.

Chen et al. [69] presented two segmented algorithms (a linear equations method and an iterative method) for modelling 1D temperature and pressure fields of a plate heat exchanger within a digital twin context. By accounting for spatial variations in fluid properties along the flow direction, their approach improved accuracy over non-segmented models while maintaining low computational cost. Their model predicts key thermal and hydraulic quantities, including heat transfer rates, wall temperatures, and fluid temperature, and pressure distributions, given known BCs. Validation against the commercial software showed sufficient agreement under multiple test conditions. Chen et al. [69] did not demonstrate continuous measurement integration, as the proposed digital twin relied on predefined BCs and physics-based algorithms rather than real-time sensor data or online model updating. The method exhibits fast convergence and reduced computational cost indicating potential suitability for near-real-time applications. Furthermore, Chen et al. [69] did not implement control over the physical system. Instead, they focused on accurate calculation of temperature and pressure fields, without incorporating

feedback mechanisms, control strategies, or actuation based on the model outputs.

Cui et al. [71] proposed a data-driven digital twin framework for early detection of metal temperature anomalies in a 600MW supercritical coal-fired boiler operating under flexible load conditions. The authors developed a PSO-optimised extreme gradient boosting ML model to predict tube metal temperature using selected operational variables identified via grey relational analysis. To improve robustness and reduce false or missed alarms, predicted temperatures are transformed into confidence-interval-based ranges derived from statistical distributions. The load-dependent anomaly benchmarks are established using histogram analysis and sliding window detection. Validated with plant data, Cui et al. [71] showed that this framework significantly reduced prediction error and successfully detected overheating events in advance. It demonstrated practical applicability for real-time anomaly detection and boiler tube protection. Cui et al. [71] demonstrated continuous measurement integration by acquiring real-time operational data from the plant's system, indicating ongoing data-stream integration rather than offline batch analysis. This framework was designed to operate in near real time. The prediction and anomaly detection computations require only a few seconds, and the model forecasts metal temperature 6min ahead, providing actionable early warnings during operation. However, control is not directly exerted autonomously on the physical system. Instead, the digital twin issues warnings and visual alerts to operators, who should then manually implement corrective actions (e.g., adjusting spray attemperation). This means that this framework supports decision-making rather than closed-loop automatic control.

A number of works [72, 73] focused on optimising pseudo-steady thermal systems using a Reinforcement Learning (RL) open loop. Renault et al. [72] investigated the application of RL to optimise gas furnace control using high-fidelity Computational Fluid Dynamics (CFD) simulations of conjugate heat transfer governed by the coupled Navier–Stokes and heat equations. The authors implemented a single-step (open-loop) variant of Proximal Policy Optimisation (PPO) to optimise control parameters. The control parameters included burner flow distribution, workpiece position, and orientation in both 2D and 3D furnace configurations. Virtual temperature sensors are used to define reward functions targeting heating homogeneity and efficiency, including multi-objective trade-offs. Results demonstrated that RL could effectively explore high-dimensional control spaces,

achieving significant improvement in temperature homogeneity without additional energy input. Continuous measurement integration was not demonstrated. Renault et al. [72] used CFD simulations with virtual sensors, and control was applied in a single-step (open-loop) manner per episode rather than through continuous feedback from real-time measurements. The approach does not operate in real time or near real time, as each training campaign requires numerous high-fidelity CFD simulations, making it suitable for offline optimisation rather than live deployment.

Similar problem was addressed byHachem et al. [73]. Their work investigated the use of RL for controlling conjugate heat transfer problems governed by the coupled Navier–Stokes and heat equations. A modified PPO algorithm was introduced for open-loop optimisation and is coupled with a high-fidelity CFD framework. Through multiple 2D and 3D natural and forced convection test cases, the method is shown to effectively reduce heat transfer enhancement and improve temperature uniformity. Hachem et al. [73] did not demonstrate continuous measurement integration, as the control strategy is formulated in an open-loop optimisation setting rather than relying on real-time state feedback.

In their study, Meglio et al. [74] investigated two methods for controlling heat transfer systems through the use of digital twins, focusing particularly on transient thermal systems. The research involved creating a digital model of a physical system, specifically a 2D square subject to an inward, transient heat flux. The goal was to keep the maximum temperature within a set limit by adjusting the convective cooling. One method used a NN trained on steady-state data, while the other applied an interactive RL algorithm. The findings showed that both approaches effectively controlled the system's thermal behaviour. However, the RL-based method provided greater adaptability to new scenarios, though it came with the trade-off of higher computational requirements. This was due to the need for interactive learning combined with unsteady FE simulations during the training, validation, and testing stages. Similar to the studies by Hachem et al. [73] and Renault et al. [72], this approach did not include sensor measurement integration during the system's online operation.

The current body of research on DT for thermal systems reveals a significant gap in the integration of continuous real-time measurements and the closed-loop control of physical systems. While several studies have explored the use of digital twins in thermal

applications, many rely on offline model calibration or simulations that do not support real-time data exchange between the virtual and physical counterparts. Some models also lack the capability to operate in real-time or near real-time, while others fail to exert control over the physical system. For example, anomaly detection frameworks show promise in providing actionable insights but rely on operator intervention based on alerts, rather than offering autonomous control. The novelty of the current work lies in its development of a digital twin framework that addresses these limitations by enabling continuous real-time sensor data integration, operating in near real-time, and implementing closed-loop control to directly influence the physical thermal system. This approach offers a more dynamic and comprehensive method for managing thermal systems, paving the way for enhanced performance, reliability, and efficiency, particularly in complex, high-stakes environments such as fusion energy experimental facilities.

Chapter 3

Finite Element- and Machine Learning-based Digital Twinning

3.1 Finite Element Method

The strong form of the non-linear 3D transient heat conduction equation is as follows [75]:

$$\frac{\partial}{\partial x}\left(k_x\frac{\partial T}{\partial x}\right)+\frac{\partial}{\partial y}\left(k_y\frac{\partial T}{\partial y}\right)+\frac{\partial}{\partial z}\left(k_z\frac{\partial T}{\partial z}\right)+G=\rho c_p\frac{\partial T}{\partial t} \tag{3.1}$$

where k_x, k_y, and k_z are the temperature-dependent thermal conductivities in x, y, and z directions, respectively. ρ and c_p are temperature-dependent density and specific heat, respectively, while T is the temperature. Finally, G is a heat source, and t is time.

The expressions describing Neumann BCs are:

$$\begin{aligned} k_x\frac{\partial T}{\partial x}l_c+k_y\frac{\partial T}{\partial y}m_c+k_z\frac{\partial T}{\partial z}n_c+q=0 \quad &\text{on} \quad \Gamma_q \\ k_x\frac{\partial T}{\partial x}l_c+k_y\frac{\partial T}{\partial y}m_c+k_z\frac{\partial T}{\partial z}n_c+h(T-T_a)=0 \quad &\text{on} \quad \Gamma_h \end{aligned} \tag{3.2}$$

where Γ_q and Γ_h are the boundary surfaces where these heat flux and convection BCs are applied. h is a convection heat transfer coefficient, T_a is ambient temperature, and q is the time-dependent heat flux. l_c, m_c, and n_c are the direction cosines of the surface normals.

Eqs. 3.1 and 3.2 are discretised in space by employing the Galerkin weighted residual method [75]:

$$\int_{\Omega} N_i \left[\frac{\partial}{\partial x}\left(k_x \frac{\partial T}{\partial x}\right) + \frac{\partial}{\partial y}\left(k_y \frac{\partial T}{\partial y}\right) + \frac{\partial}{\partial z}\left(k_z \frac{\partial T}{\partial z}\right) + G - \rho c_p \frac{\partial T}{\partial t} \right] \mathrm{d}\Omega = 0 \tag{3.3}$$

Temperature T could be approximated in space and time as:

$$T(x, y, z, t) = \sum_{i=1}^{n_n} N_i(x, y, z, t) T_i \tag{3.4}$$

where Ω is the whole problem domain, N_i are the shape (interpolation) functions, and n_n is the number of nodes. T_i are the temperature values at the nodes of a mesh.

Eq. 3.3 is integrated by parts:

$$\begin{aligned} -\int_{\Omega} \left[\frac{\partial N_i}{\partial x}\left(k_x \frac{\partial T}{\partial x}\right) + \frac{\partial N_i}{\partial y}\left(k_y \frac{\partial T}{\partial y}\right) + \frac{\partial N_i}{\partial z}\left(k_z \frac{\partial T}{\partial z}\right) - N_i G + N_i \left(\rho c_p \frac{\partial T}{\partial t}\right) \right] \mathrm{d}\Omega + \\ + \int_{\Gamma_{q+h}} N_i \left[\left(k_x \frac{\partial T}{\partial x} l_c\right) + \left(k_y \frac{\partial T}{\partial y} m_c\right) + \left(k_z \frac{\partial T}{\partial z} n_c\right) \right] d\Gamma = 0 \end{aligned} \tag{3.5}$$

Eq. 3.2 is integrated as:

$$\begin{aligned} \int_{\Gamma_q} N_i \left[\left(k_x \frac{\partial T}{\partial x} l_c\right) + \left(k_y \frac{\partial T}{\partial y} m_c\right) + \left(k_z \frac{\partial T}{\partial z} n_c\right) \right] d\Gamma = -\int_{\Gamma_q} N_i q d\Gamma \\ \int_{\Gamma_h} N_i \left[\left(k_x \frac{\partial T}{\partial x} l_c\right) + \left(k_y \frac{\partial T}{\partial y} m_c\right) + \left(k_z \frac{\partial T}{\partial z} n_c\right) \right] d\Gamma = -\int_{\Gamma_h} N_i h (T - T_a) d\Gamma \end{aligned} \tag{3.6}$$

Eqs. 3.4 and 3.6 is substituted into Eq. 3.5 to obtain the global system of equations:

$$\begin{aligned} -\int_{\Omega} \left[\frac{\partial N_i}{\partial x}\left(k_x \frac{\partial N_j}{\partial x} T_j\right) + \frac{\partial N_i}{\partial y}\left(k_y \frac{\partial N_j}{\partial y}\right) T_j + \frac{\partial N_i}{\partial z}\left(k_z \frac{\partial N_j}{\partial z} T_j\right) \right] \mathrm{d}\Omega + \\ + \int_{\Omega} \left[N_i G - N_i \left(\rho c_p \frac{\partial N_j}{\partial t} T_j\right) \right] - \int_{\Gamma_q} N_i q d\Gamma - \int_{\Gamma_h} N_i h (T - T_a) d\Gamma = 0 \end{aligned} \tag{3.7}$$

Eq. 3.7 can be re-written as:

$$[\boldsymbol{M}] \left\{ \frac{d\{\boldsymbol{T}\}}{dt} \right\} + [\boldsymbol{K}] \{\boldsymbol{T}\} = \{\boldsymbol{f}\} \tag{3.8}$$

where $\{\boldsymbol{T}\}$ is the temperature vector. Global capacitance matrix $[\boldsymbol{M}]$, stiffness matrix $[\boldsymbol{K}]$, and loading vector $\{\boldsymbol{f}\}$ being equal to:

$$[\boldsymbol{M}] = \int_{\Omega} \rho c_p \, [\boldsymbol{N}]^T \, [\boldsymbol{N}] \, \mathrm{d}\Omega \tag{3.9}$$

$$[\boldsymbol{K}] = \int_{\Omega} [\boldsymbol{B}]^T \, [\boldsymbol{D}] \, [\boldsymbol{B}] \, \mathrm{d}\Omega + \int_{\Gamma_h} h \, [\boldsymbol{N}]^T \, [\boldsymbol{N}] \, \mathrm{d}\Gamma \tag{3.10}$$

$$\{\boldsymbol{f}\} = \int_{\Omega} G \, [\boldsymbol{N}]^T \, \mathrm{d}\Omega - \int_{\Gamma_q} q \, [\boldsymbol{N}]^T \, \mathrm{d}\Gamma + \int_{\Gamma_h} h T_a \, [\boldsymbol{N}]^T \, \mathrm{d}\Gamma \tag{3.11}$$

$$[\boldsymbol{N}]^T = \{\boldsymbol{N}\} = \begin{Bmatrix} N_1 \\ N_2 \\ \cdots \\ N_n \end{Bmatrix} \quad \text{and} \quad [\boldsymbol{D}] = \begin{bmatrix} k_{xx} & k_{xy} & k_{xz} \\ k_{yx} & k_{yy} & k_{yz} \\ k_{zx} & k_{zy} & k_{zz} \end{bmatrix}$$

$$[\boldsymbol{B}] = \begin{bmatrix} \dfrac{\partial N_1}{\partial x} & \dfrac{\partial N_2}{\partial x} & \cdots & \dfrac{\partial N_n}{\partial x} \\ \dfrac{\partial N_1}{\partial y} & \dfrac{\partial N_2}{\partial y} & \cdots & \dfrac{\partial N_n}{\partial y} \\ \dfrac{\partial N_1}{\partial z} & \dfrac{\partial N_2}{\partial z} & \cdots & \dfrac{\partial N_n}{\partial z} \end{bmatrix} \tag{3.12}$$

where $\{\boldsymbol{N}\}$ is the shape function vector. The absence of a heat source G transforms Eq. 3.11 into the following:

$$\{\boldsymbol{f}\} = -\int_{\Gamma_q} q \, [\boldsymbol{N}]^T \, \mathrm{d}\Gamma + \int_{\Gamma_h} h T_a \, [\boldsymbol{N}]^T \, \mathrm{d}\Gamma \tag{3.13}$$

FDM is used for time discretisation, which is derived using Taylor series. Temperature can be expanded at the $(n+1)^{\text{th}}$ time instance:

$$T^{n+1} = T^n + \Delta t \frac{dT^n}{dt} + \frac{\Delta t^2}{2} \frac{d^2 T^n}{dt^2} + \ldots \tag{3.14}$$

where Δt is a time step size. When the higher-order terms are removed, the following forward difference approximation can be achieved:

$$\frac{dT^n}{dt} \approx \frac{T^{n+1} - T^n}{\Delta t} + O(\Delta t) \tag{3.15}$$

The parameter θ is introduced:

$$\{\boldsymbol{T}\}^{n+\theta} = \theta \{\boldsymbol{T}\}^{n+1} + (1-\theta) \{\boldsymbol{T}\}^n \tag{3.16}$$

where superscript n denotes the n^{th} time step. θ varies from 0 to 1: 1 corresponds to the fully implicit scheme, 0 to fully explicit one, and 0.5 to the semi-implicit one.

The system of equations is transformed by substituting Eqs. 3.16 and 3.15 into Eq. 3.8:

$$[\boldsymbol{M}] \left\{ \frac{\{\boldsymbol{T}\}^{n+1} - \{\boldsymbol{T}\}^n}{\Delta t} \right\} + [\boldsymbol{K}] \{\boldsymbol{T}\}^{n+\theta} = \{\boldsymbol{f}\}^{n+\theta} \tag{3.17}$$

$$[\boldsymbol{M}] \left\{ \frac{\{\boldsymbol{T}\}^{n+1} - \{\boldsymbol{T}\}^n}{\Delta t} \right\} + [\boldsymbol{K}] \left\{ \theta \{\boldsymbol{T}\}^{n+1} + (1-\theta) \{\boldsymbol{T}\}^n \right\} = \\ = \theta \{\boldsymbol{f}\}^{n+1} + (1-\theta) \{\boldsymbol{f}\}^n \tag{3.18}$$

With θ equal to 1, Eq. 3.18 becomes:

$$[\boldsymbol{M}] \left\{ \frac{\{\boldsymbol{T}\}^{n+1} - \{\boldsymbol{T}\}^n}{\Delta t} \right\} + [\boldsymbol{K}] \{\boldsymbol{T}\}^{n+1} = \{\boldsymbol{f}\}^{n+1} \tag{3.19}$$

3.2 HIVE sample

The HIVE facility was developed to support the research and development of PFCs and is situated at the UKAEA site in Culham, Oxfordshire, UK [18]. It is a High Heat Flux (HHF) experimental setup designed to simulate the thermal conditions experienced by components during steady-state operation of a fusion reactor. Operating under vacuum, it employs induction heating to apply heat fluxes of up to $20 MW/m^2$ to test samples, while a pressurised coolant system removes the excess thermal energy.

Figure 3.1 presents the HIVE's vacuum vessel. It measures 500mm in both height and diameter and features six ports arranged around its circumference. Two of these ports are dedicated to vacuum pumping and system monitoring, while the remaining four are equipped with viewing windows that offer multiple observational angles.

FIGURE 3.1: HIVE's vacuum vessel (photo courtesy of UKAEA).

Figure 3.2 shows the test setup used in the HIVE facility, illustrating the main components. At the centre is the test sample, clamped in place and surrounded by an induction heating coil that delivers thermal loads. The sample is connected to a pressurised water feed loop, allowing active water cooling during testing. The entire assembly is held securely within a support frame, which ensures proper alignment and mechanical stability throughout the experiment.

Figures 3.3 and 3.4 show two HIVE samples: Sample 1 consisting of tungsten and copper and Sample 2 consisting of stainless steel. Figure 3.5 shows the sample dimensions.

Induction heating involves raising the temperature of electrically conductive materials using Electromagnetics (EM) induction. When an Alternating Current (AC) flows through a coil, it creates a rapidly fluctuating magnetic field. This changing field induces circulating electric currents within any nearby conductive object, such as the sample in HIVE. These electric currents are know as eddy currents. As eddy currents flow through the material, they encounter electrical resistance, which causes the energy to dissipate as heat. This mechanism is known as Joule (or resistive) heating.

Electrical resistivity is a property that describes how much a material resists the flow of electric current. Higher resistivity results in more heat being produced when eddy

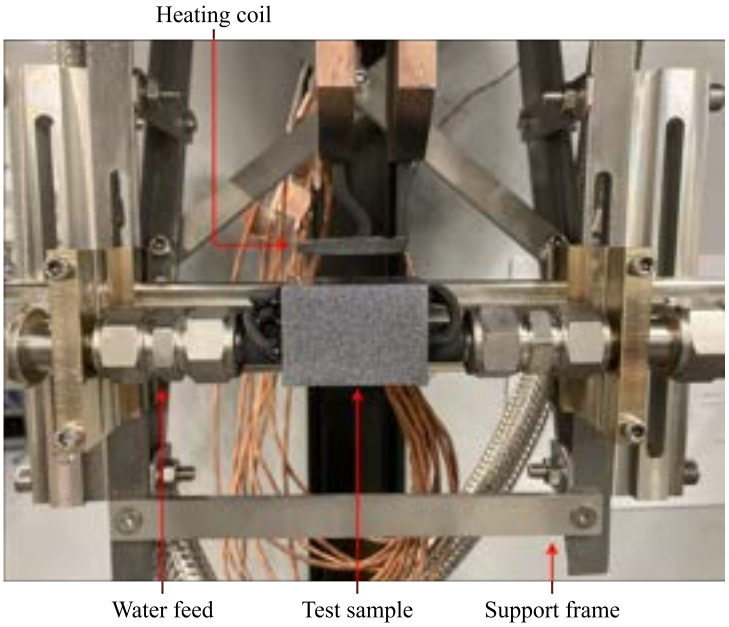


FIGURE 3.2: HIVE's setup inside the vacuum vessel (photo courtesy of UKAEA).

FIGURE 3.3: Sample 1 for HIVE (photo courtesy of UKAEA).

currents are induced. Although electromagnetic induction generates heat throughout the volume of a material (volumetric heating), components in fusion devices primarily experience heat as a surface flux. The induced currents are concentrated near the surface of the conductor. Specifically to HIVE, they are concentrated between the surface closest

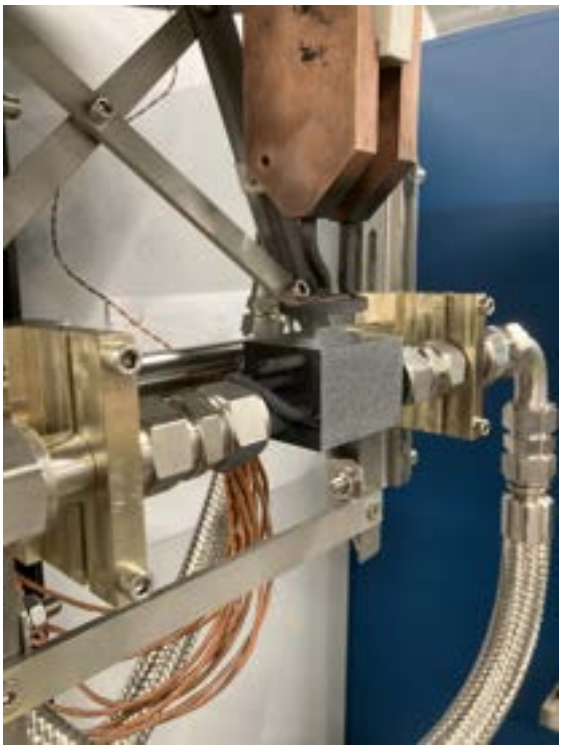
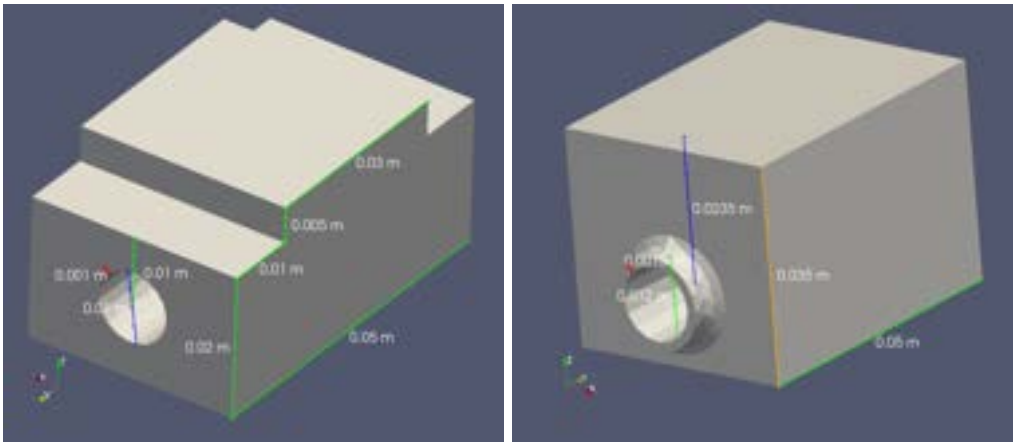


FIGURE 3.4: Sample 2 for HIVE (photo courtesy of UKAEA).



(a) Sample 1 (b) Sample 2

FIGURE 3.5: The dimensions for Samples 1 and 2 (Figures 3.3 and 3.4).

to the coil and a depth known as the skin depth d, which is defined as [76]:

$$d = \sqrt{\frac{\rho}{\pi f \mu_0 \mu_r}}$$
$$\mu_0 = 4\pi \times 10^{-7} N A^{-2} \tag{3.20}$$

where μ_0 and μ_r are vacuum's and conductor's relative magnetic permeabilities, respectively. f and ρ are the electric current's frequency and conductor's electrical resistivity,

respectively. The intensity of eddy currents diminishes exponentially with distance from the surface facing the coil. Additionally, because skin depth decreases as f increases, operating at higher frequencies results in heating that is concentrated closer to the surface of the sample. HIVE operates within a frequency range of $50kHz$ to $150kHz$, typically around $100kHz$. At this frequency, the skin depth is expected to be less than $0.5mm$, which enables the induced heating to be approximated as a heat flux applied to the surface. The HIVE's induction system could deliver up to $45kW$ of power. This upper limit refers to the electrical power drawn from the power supply, not the thermal power transferred to the sample. The amount of heat delivered to the component depends on the coupling efficiency of the induction coil. This efficiency decreases with increasing distance between the coil and the conductor, being inversely proportional to the square root of that distance.

The effect of the coolant on the sample's temperature is approximated using a 1D coolant model, which is described in Sub-section 4.1.2.

3.3 Finite Element-based Digital Twinning

3.3.1 Workflow

Figure 3.6 shows the assembled FE DT control loop. The rest of this section describes each component in detail.

This control loop combines solution construction using FEM and ROM with a digital PID controller. The main goal of this framework is to monitor and regulate the maximum temperature within the HIVE sample by adjusting the coolant flow rate based on a limited number of temperature measurements. The framework comprises three main components: (1) the digital PID controller, (2) a FE model that imitates the physical experiment, and (3) the FE-based solution reconstruction module with eigenvalue-based ROM. These components work in an integrated manner to enable near real-time temperature control and monitoring.

The process begins with a user-defined temperature set point T_{sp_n}, which corresponds to the desired maximum temperature within the HIVE sample at the current control step n. This set point is compared against the actual maximum temperature T_{max_n}

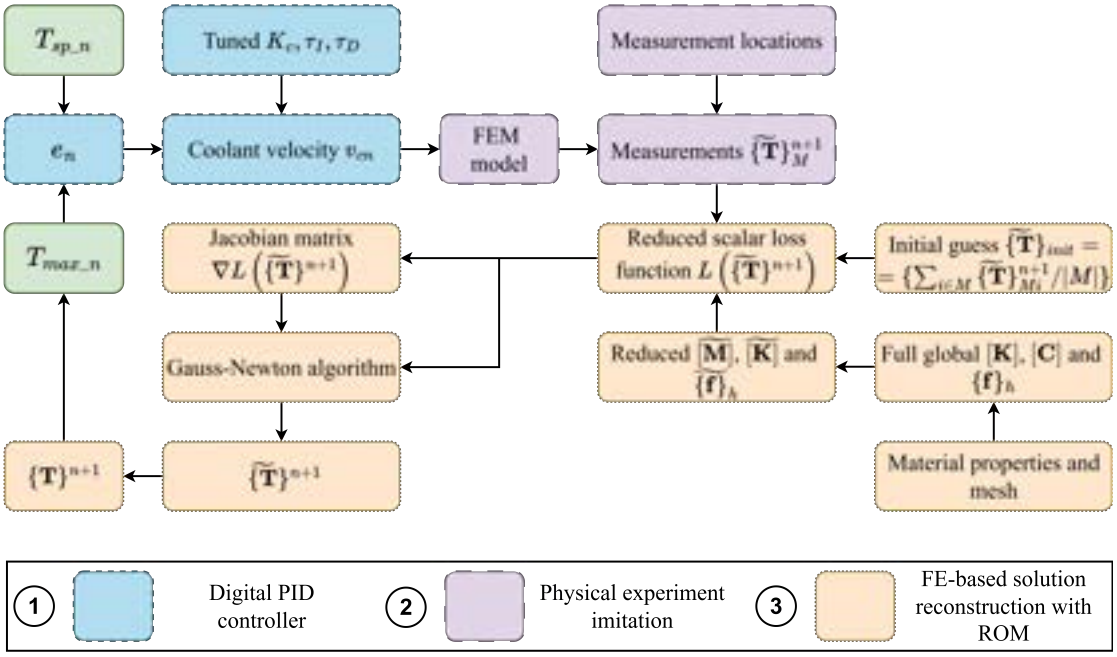


FIGURE 3.6: FE DT loop combining FE-based solution construction with digital PID controller. The temperature measurements shown here are pseudo-measurements generated using the forward FE model rather than experimental data. These synthetic measurements are used to validate the proposed methodology. In future applications, the same framework aims to process experimentally measured temperature data.

estimated by the FE-based solution construction process. The difference between these two values constitutes the control error e_n, which is passed to the digital PID controller. The PID controller uses three tuned parameters: proportional gain K_C, integral time τ_I, and derivative time τ_D. It computes the appropriate control signal. Sub-section 3.3.4 provides more information about PID controller. In this application, the control output is the coolant velocity v_{cn}, which is adjusted to minimise the temperature deviation from the set point.

The updated coolant velocity v_{cn} serves as an input to the FE model, which simulates the thermal response of the HIVE sample under the given cooling condition. This model represents the physical system in a virtual environment and provides synthetic temperature measurements at selected measurement locations. These simulated measurements, denoted by $\{\widetilde{\mathbf{T}}\}_M^{n+1}$, represent the temperature values obtained at a sparse set of sensor locations on the domain boundary.

The next stage involves reconstructing the full temperature field $\{\widetilde{\mathbf{T}}\}^{n+1}$ across the entire domain using these measurements. This is achieved using a reduced-order FE-based solution construction method. The construction process starts with an initial

guess $\widetilde{\{\mathbf{T}\}}_{init}$, calculated as the measurements' average.

Using this initial guess, a reduced scalar loss function $L\left(\widetilde{\{\mathbf{T}\}}^{n+1}\right)$ is evaluated, which quantifies the discrepancy between the current estimate and the measured data together with selected FE equations. To evaluate this loss function efficiently, the method uses global reduced FE matrices and vectors: a reduced stiffness matrix $\widetilde{[\mathbf{K}]}$, a reduced capacity matrix $\widetilde{[\mathbf{M}]}$, and a reduced convective loading vector $\widetilde{\{\mathbf{f}\}}_h$. These matrices are derived from the full global FE matrices and vectors based on the material properties and mesh, which are known a priori. Sub-section 3.3.3 provides more information about system reduction used in this work. The minimisation of the loss function is carried out iteratively using the Gauss–Newton algorithm. At each iteration, the Jacobian matrix $\nabla L\left(\widetilde{\{\mathbf{T}\}}^{n+1}\right)$ is computed. The Gauss–Newton method uses this Jacobian to update the estimate of the temperature field and converge toward the reduced temperature distribution $\widetilde{\{\mathbf{T}\}}^{n+1}$.

Once the full temperature distribution is reconstructed, the maximum temperature value within the HIVE sample is extracted from the result. This value is then fed back to the digital PID controller to form the control loop.

3.3.2 Solution and material property construction

For convenience, Eq. 3.19 is written as:

$$[\boldsymbol{M}]\left\{\frac{\Delta \boldsymbol{T}}{\Delta t}\right\} + [\boldsymbol{K}]\{\boldsymbol{T}\}^{n+1} = \{\boldsymbol{f}\}^{n+1} \tag{3.21}$$

where $\Delta \boldsymbol{T} = \{\boldsymbol{T}\}^{n+1} - \{\boldsymbol{T}\}^{n}$. The decomposition of the global loading vector $\{\boldsymbol{f}\}$ into $\{\boldsymbol{f}\}_q$ and $\{\boldsymbol{f}\}_{g-q}$ is performed. Vector $\{\boldsymbol{f}\}_q$ denotes the elements of $\{\boldsymbol{f}\}$ corresponding to the applied heat flux q, and vector $\{\boldsymbol{f}\}_{g-q}$ corresponds to the rest of $\{\boldsymbol{f}\}$ including the convection. It is assumed that the heat flux BC is unknown.

Then Eq. 3.21 could be re-arranged into two groups as follows:

$$\begin{bmatrix}[\boldsymbol{M}]_q \\ [\boldsymbol{M}]_{g-q}\end{bmatrix}\left\{\frac{\Delta \boldsymbol{T}}{\Delta t}\right\} + \begin{bmatrix}[\boldsymbol{K}]_q \\ [\boldsymbol{K}]_{g-q}\end{bmatrix}\{\boldsymbol{T}\}^{n+1} - \begin{Bmatrix}\{\boldsymbol{f}\}_q \\ \{\boldsymbol{f}\}_{g-q}\end{Bmatrix}^{n+1} = 0 \tag{3.22}$$

Subscript q refers to the set of equations in Eq. 3.21 associated with the nodes located on Γ_q. The subscript $g-q$ denotes the equations corresponding to nodes located in $\Omega \setminus \Gamma_q$, that is, the whole computational domain except for nodes on Γ_q where the heat flux BC is imposed.

The second group of Eq. 3.22 encompasses all known BCs, such as convection. This formulation enables the determination of temperature throughout the domain, including on boundaries where heat fluxes are unknown.

The heat flux applied at each time step is not imposed directly but reconstructed from the estimated temperature field. The temperature distribution $\{\boldsymbol{T}\}_{con}$ is obtained by constructing the full spatial field from sparse temperature measurements using the FE-based solution construction method described in this sub-section. The corresponding heat flux on Γ_q is then computed as:

$$q_{con}^{n} = \frac{\int_{\Gamma_q} \{\boldsymbol{n}\}\, k(T) \nabla T_{con}^{n}}{\int_{\Gamma_q} 1} \tag{3.23}$$

where q_{con} denotes the heat flux on Γ_q constructed or inferred from $\{\boldsymbol{T}\}_{con}$, $\{\boldsymbol{T}\}_{con}$ is the reconstructed temperature field at time step n, and $\{\boldsymbol{n}\}$ is the outward unit normal vector to Γ_q.

If temperature measurements are available at selected locations and certain BCs are not specified, the problem transforms into a minimisation task involving the points with known temperature values along with the second group in Eq. 3.22. This procedure is commonly referred to as the inverse estimation of BCs. In such an inverse formulation, it is assumed that the ambient temperature is known, as well as the temperature values $\{\boldsymbol{T}\}_M$ at points that are part of the measurement set M. The applied heat flux q is treated as an unknown. Under these assumptions, Eq. 3.22 can be re-formulated as follows:

$$\begin{Bmatrix} \{\boldsymbol{f}\}_q \\ \{\boldsymbol{R}\}_{g-q} \end{Bmatrix}^{n+1} = \begin{bmatrix} [\boldsymbol{M}]_q \\ [\boldsymbol{M}]_{g-q} \end{bmatrix} \left\{ \frac{\Delta \boldsymbol{T}}{\Delta t} \right\} + \begin{bmatrix} [\boldsymbol{K}]_q \\ [\boldsymbol{K}]_{g-q} \end{bmatrix} \{\boldsymbol{T}\}^{n+1} - \begin{Bmatrix} 0 \\ \{\boldsymbol{f}\}_{g-q} \end{Bmatrix}^{n+1} \tag{3.24}$$

Prior to evaluating the heat flux Γ_q, it is essential to determine the vector containing the unknown nodal temperatures and material properties. These unknown quantities are collectively represented by the vector $\{\boldsymbol{v}\}$. In order to obtain $\{\boldsymbol{v}\}$, the magnitude of

a loss vector $\{\boldsymbol{L}\}$, which is a function of $\{\boldsymbol{v}\}$ should be minimised. The definition of the loss vector is:

$$\{\boldsymbol{L}\} = \begin{Bmatrix} \{\boldsymbol{L}\}_R \\ \{\boldsymbol{L}\}_M \\ \{\boldsymbol{L}\}_{REG} \end{Bmatrix} \tag{3.25}$$

The residual component introduced in Eq. 3.25 is computed using Eq. 3.24 as:

$$\{\boldsymbol{L}\}_R = \{\boldsymbol{R}\}_{g-q} \tag{3.26}$$

The measurement component is:

$$\begin{aligned} \{\boldsymbol{L}\}_M &= \{\boldsymbol{s}\}_M \odot (\{\boldsymbol{T}\}_i - \{\boldsymbol{T}\}_{Mi}) \\ \{\boldsymbol{s}\}_{Mi} &= \begin{cases} 1, & \text{if } i \in M \\ 0, & \text{if } i \notin M \end{cases} \end{aligned} \tag{3.27}$$

where $\odot$ represents Hadamard (element-wise) product. For two matrices (or vectors), $\boldsymbol{A}$ and $\boldsymbol{B}$, of the same dimensions, their Hadamard product $\boldsymbol{C} = \boldsymbol{A} \odot \boldsymbol{B}$ is defined component-wise as:

$$C_{ij} = A_{ij} B_{ij} \tag{3.28}$$

At each iteration, the regularisation component, used for smoothing, is computed using the first group of equations (related to $\{\boldsymbol{f}\}_q$) in Eq. 3.24 as:

$$
\begin{aligned}
\{\boldsymbol{L}\}_{REG} &= \begin{Bmatrix} \{\boldsymbol{s}\}_1 \odot \left(\{\boldsymbol{f}\}_q - \frac{\sum_{i \in \Gamma_q \setminus \Gamma_{edge}} \{\boldsymbol{f}\}_{qi}}{|\Gamma_q \setminus \Gamma_{edge}|} \right) \\ \{\boldsymbol{s}\}_2 \odot \left(\{\boldsymbol{f}\}_q - \frac{\sum_{i \in \Gamma_{edge} \setminus \Gamma_{corners}} \{\boldsymbol{f}\}_{qi}}{|\Gamma_{edge} \setminus \Gamma_{corners}|} \right) \\ \{\boldsymbol{s}\}_3 \odot \left(\{\boldsymbol{f}\}_q - \frac{\sum_{i \in \Gamma_{corners}} \{\boldsymbol{f}\}_{qi}}{|\Gamma_{corners}|} \right) \end{Bmatrix} \\
\{\boldsymbol{s}\}_{1i} &= \begin{cases} 1, & \text{if } i \in \Gamma_q \setminus \Gamma_{edge} \\ 0, & \text{if } i \notin \Gamma_q \setminus \Gamma_{edge} \end{cases} \\
\{\boldsymbol{s}\}_{2i} &= \begin{cases} 1, & \text{if } i \in \Gamma_{edge} \setminus \Gamma_{corners} \\ 0, & \text{if } i \notin \Gamma_{edge} \setminus \Gamma_{corners} \end{cases} \\
\{\boldsymbol{s}\}_{3i} &= \begin{cases} 1, & \text{if } i \in \Gamma_{corners} \\ 0, & \text{if } i \notin \Gamma_{corners} \end{cases}
\end{aligned} \tag{3.29}
$$

Γ_{edge} denotes the three-dimensional edge of the surface on which the heat flux q is applied. Sharp geometrical features along Γ_{edge} that are captured using a single finite element are denoted by $\Gamma_{corners}$. For the steady-state solution, the time-dependent terms in Eqs. 3.21, 3.22, and 3.23 can be neglected.

The magnitude of $\{\boldsymbol{L}\}$, $\|\{\boldsymbol{L}\}\|$, could be minimised using various gradient-based optimisation approaches. In particular, Gauss-Newton method is employed in this work [77, 78]. Gauss-Newton method is well suited to this optimisation problem because each iteration reduces to the solution of a linearised system, allowing efficient use of various solvers or preconditioners. Previous work has shown that Gauss-Newton method converges significantly faster than gradient-based approaches, such as Adam optimiser, provided the linearised system remains sufficiently sparse [36]. The latter exhibit slower convergence due to their local update nature, where parameter updates rely primarily on first-order gradient information and limited history.

The term $\|\{\boldsymbol{L}\}\|^2$ represents the sum of the squared components of the loss vector, specifically $\|\{\boldsymbol{L}\}\|^2 = \sum l_i^2$. In each iteration of the minimization process, it is necessary to compute the Jacobian matrix $[\boldsymbol{J}]_{ij} = \partial l_i / \partial \{\boldsymbol{v}\}_j$, which contains the partial derivatives of each loss term with respect to the unknown parameters. When the loss vector originates from the FE equations, the Jacobian $[\boldsymbol{J}]$ can be determined analytically. The

norm $\|\{\boldsymbol{L}\}\|$ reduced by evaluating both $[\boldsymbol{J}]$ and $\{\boldsymbol{L}\}^s$ at each iteration s, and then solving the following system of equations to update the temperature increment $\Delta\{\boldsymbol{T}\}$:

$$\begin{aligned}[\boldsymbol{J}]^T[\boldsymbol{J}]\Delta\{\boldsymbol{T}\} &= -[\boldsymbol{J}]^T\{\boldsymbol{L}\}^s \\ \{\boldsymbol{T}\}^{s+1} &= \{\boldsymbol{T}\}^s + \Delta\{\boldsymbol{T}\}\end{aligned} \tag{3.30}$$

$\begin{Bmatrix}\{\boldsymbol{f}\}_q \\ \{\boldsymbol{R}\}_{g-q}\end{Bmatrix}$ is annotated as $\{\boldsymbol{r}\}$. This vector is derived from the heat conduction equation as shown in Eq. 3.24. Then, for the FE equations, $[\boldsymbol{J}]$ can be calculated analytically following the steps:

1. Jacobian matrix of $\{\boldsymbol{r}\}$, $[\boldsymbol{J}]_r$, can be analytically calculated for a given vector $\{\boldsymbol{v}\}$.
2. Jacobian matrix of the residual term is

$$[\boldsymbol{J}]_R = \{[\boldsymbol{J}]_{ri} : i \in \Omega \setminus \Gamma_q\} \tag{3.31}$$

3. Jacobian matrix of the measurement term is

$$[\boldsymbol{J}]_M = \{[\boldsymbol{I}_n]_i : i \in M\} \tag{3.32}$$

4. Jacobian matrix of the regularisation term is

$$\begin{aligned}[\boldsymbol{J}]_{REG} = \{[\boldsymbol{J}]_{ri} : i \in \Gamma_q \setminus \Gamma_{edge}\} - \mu_{\Gamma_q \setminus \Gamma_{edge}} + \\ + \{[\boldsymbol{J}]_{ri} : i \in \Gamma_{edge} \setminus \Gamma_{corners}\} - \mu_{\Gamma_{edge} \setminus \Gamma_{corners}} + \\ + \{[\boldsymbol{J}]_{ri} : i \in \Gamma_{corners}\} - \mu_{\Gamma_{corners}}\end{aligned} \tag{3.33}$$

   where $\mu_{\Gamma_q \setminus \Gamma_{edge}}$ are expressed by Eq. 3.34

5. $[\boldsymbol{J}]$ is obtained by vertically concatenating $[\boldsymbol{J}]_R$, $[\boldsymbol{J}]_M$, and $[\boldsymbol{J}]_{REG}$.

Subscripts i and j denote the i^{th} row and j^{th} column of the matrix, respectively; $[\boldsymbol{I}]$ is the identity matrix of size equal to the number of nodes in a mesh (subscript n).

$$\mu_{\Gamma_q \setminus \Gamma_{edge}} = \frac{1}{|\Gamma_q \setminus \Gamma_{edge}|} \left[\sum_{i \in \Gamma_q \setminus \Gamma_{edge}} [\boldsymbol{J}]_{ri1}, \sum_{i \in \Gamma_q \setminus \Gamma_{edge}} [\boldsymbol{J}]_{ri2}, \cdots, \sum_{i \in \Gamma_q \setminus \Gamma_{edge}} [\boldsymbol{J}]_{rin} \right]$$
$$\mu_{\Gamma_{edge} \setminus \Gamma_{corners}} = \frac{1}{|\Gamma_{edge} \setminus \Gamma_{corners}|} \left[\sum_{i \in \Gamma_{edge} \setminus \Gamma_{corners}} [\boldsymbol{J}]_{ri1}, \cdots, \sum_{i \in \Gamma_{edge} \setminus \Gamma_{corners}} [\boldsymbol{J}]_{rin} \right]$$
$$\mu_{\Gamma_{corners}} = \frac{1}{|\Gamma_{corners}|} \left[\sum_{i \in \Gamma_{corners}} [\boldsymbol{J}]_{ri1}, \sum_{i \in \Gamma_{corners}} [\boldsymbol{J}]_{ri2}, \cdots, \sum_{i \in \Gamma_{corners}} [\boldsymbol{J}]_{rin} \right] \tag{3.34}$$

3.3.2.1 Solution construction

When reconstructing the solution using sparse temperature measurements under the assumption that the material properties are known, the set of unknowns is represented by the following vector:

$$\{\boldsymbol{v}\} = \{\boldsymbol{T}\} = [T_1, T_2, \ldots, T_N]^T \tag{3.35}$$

For this case, $[\boldsymbol{J}]$ is equal to $[\boldsymbol{K}]$ at every Gauss-Newton iteration.

3.3.2.2 Linear material properties

This subsection outlines the procedure for determining linear material properties using limited temperature measurements within the domain. Under the assumption that the domain is composed of a single linear material characterized by thermal conductivity k, the resulting vector of unknowns is defined as follows:

$$\{\boldsymbol{v}\} = [T_1, T_2, \ldots, T_n, k]^T \tag{3.36}$$

Additional column is added to $[\boldsymbol{J}]$, it corresponds to $\partial l_i / \partial k$, as it is shown in Figure 3.7. $[\boldsymbol{K}]$ is calculated for the values of $\{\boldsymbol{v}\}$ at every Gauss-Newton iteration. The derivative of the residual with respect to the thermal conductivity is obtained using an analytical sensitivity approach, exploiting the linear dependence of the conductivity matrix on k. Since the stiffness matrix can be written as $[\boldsymbol{K}](k) = k\,[\boldsymbol{K}]_{conn}$, its derivative is constant, $\partial [\boldsymbol{K}] / \partial k = [\boldsymbol{K}]_{conn}$, which leads directly to the expression illustrated in Figure 3.7.

Stiffness connectivity matrix $[\boldsymbol{K}]_{conn}$ is calculated as the stiffness matrix when k is equal to 1.

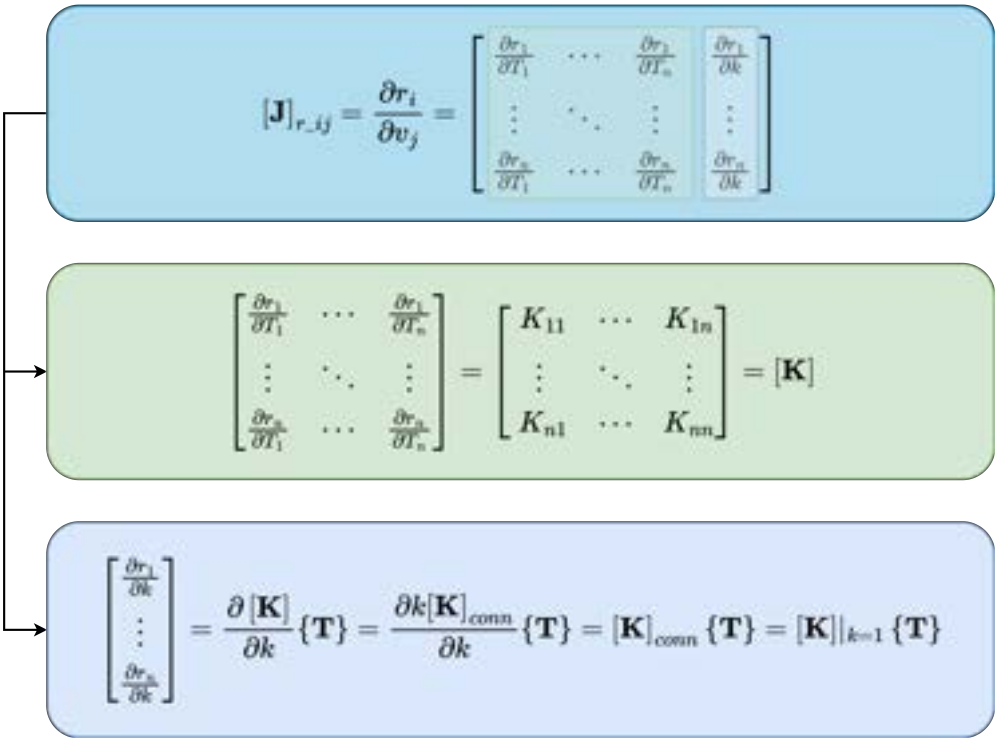


FIGURE 3.7: Calculation of Jacobian (tangent) matrix of $\{\boldsymbol{r}\}$, $[\boldsymbol{J}]_r$, for a given vector $\{\boldsymbol{v}\}$ for one material and temperature-independent thermal conductivity.

In cases where the domain is made up of two distinct materials with thermal conductivities k_1 and k_2, the corresponding vector of unknown variables is expressed as:

$$\{\boldsymbol{v}\} = [T_1, T_2, \ldots, T_n, k_1, k_2]^T \tag{3.37}$$

Two additional columns are added to $[\boldsymbol{J}]$, they correspond to $\partial l_i/\partial k_1$ and $\partial l_i/\partial k_2$, as it is shown in Figure 3.8. Stiffness connectivity matrix $[\boldsymbol{K}]_{conn_1}$ for k_1 is calculated as the stiffness matrix when k_1 is equal to 1 and k_2 is equal to 0. Stiffness connectivity matrix $[\boldsymbol{K}]_{conn_2}$ for k_2 is calculated as the stiffness matrix when k_1 is equal to 0 and k_2 is equal to 1.

3.3.2.3 Nonlinear material properties

A similar approach could be used to construct the non-linear material properties from the temperature measurements.

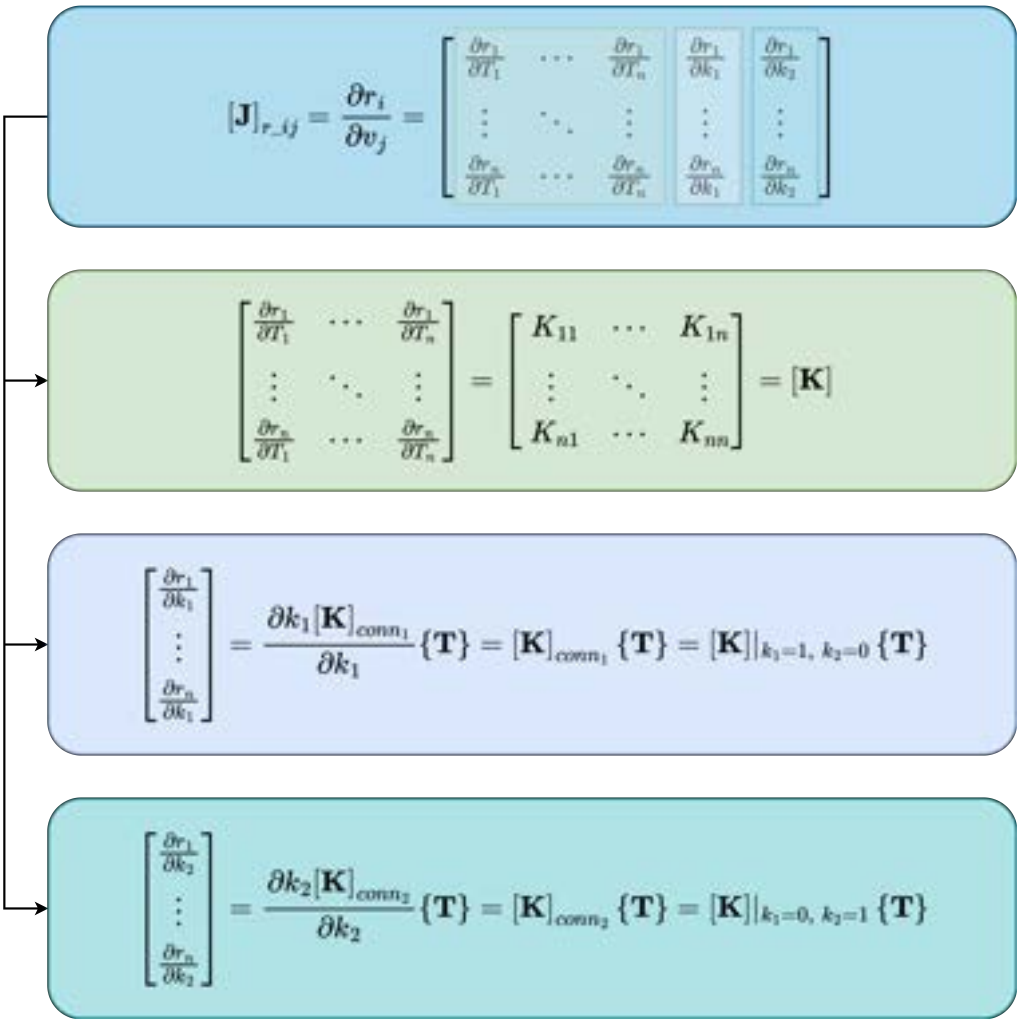


FIGURE 3.8: Calculation of Jacobian (tangent) matrix of $\{\boldsymbol{r}\}$, $[\boldsymbol{J}]_r$, for a given vector $\{\boldsymbol{v}\}$ for two materials and temperature-independent thermal conductivity.

Firstly, the domain is assumed to be made of a single material, with the thermal conductivity exhibiting a linear dependence on temperature as expressed by the following relationship:

$$k(T) = aT + b \tag{3.38}$$

Additionally, it is assumed that the thermal conductivity is known to be k_{known} at a specific temperature T_{known}. As a result, Eq. 3.38 can be re-formulated as follows:

$$k(T) = aT + (k_{known} - aT_{known}) \tag{3.39}$$

Consequently, the unknowns vector transforms to the following:

$$\{\boldsymbol{v}\} = [T_1, T_2, \ldots, T_n, a]^T \tag{3.40}$$

Additional column is added to $[\boldsymbol{J}]$. It relates to $\partial l_i/\partial a$ and is calculated as:

$$\frac{\partial k}{\partial a} = T - T_{known} \tag{3.41}$$

Eq 3.41 follows from Eq 3.39. Figure 3.9 shows the Jacobian calculation process. $[\boldsymbol{K}]$ is calculated for the values of $\{\boldsymbol{v}\}$ at every Gauss-Newton iteration. Furthermore, temperature values at Gauss points $\{\boldsymbol{T}\}_{gp}^{iter}$ are calculated at every iteration from $\{\boldsymbol{T}\}$. Then, the stiffness connectivity matrix $[\boldsymbol{K}]_{conn}$ is calculated as the stiffness matrix for when the temperature values at Gauss points are equal to $\{\boldsymbol{T}\}_{gp}^{iter} - T_{known}$.

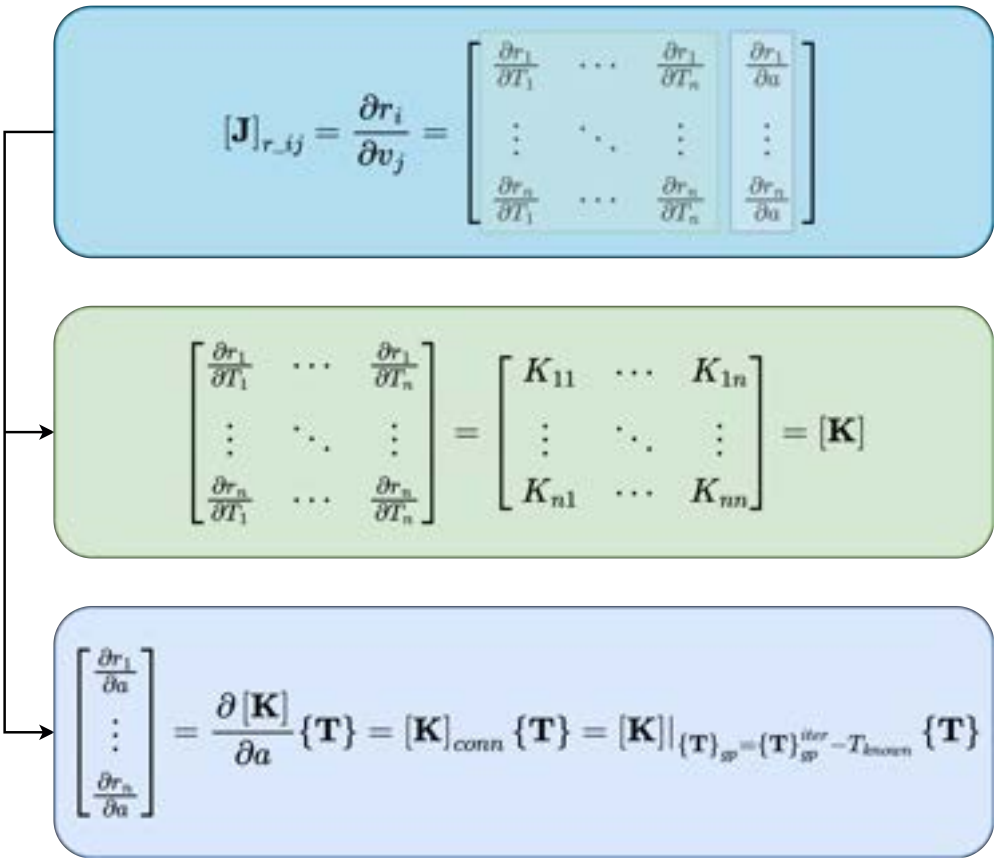


FIGURE 3.9: Calculation of Jacobian (tangent) matrix of $\{\boldsymbol{r}\}$, $[\boldsymbol{J}]_r$, for a given vector $\{\boldsymbol{v}\}$ for one material, linear $k(T)$, and k equal to k_{known} at T_{known}. $\{\boldsymbol{T}\}_{gp}$ is a vector of temperature values at Gauss points.

If the thermal conductivity $k(T)$ is known for all temperatures less than or equal to T_r, then Eq.3.41 can be employed to describe $k(T)$ for temperatures greater than T_r:

$$k(T) = aT + (k_r - aT_r) \tag{3.42}$$

Since $k(T)$ is already defined for $T \leq T_r$, the corresponding conductivity value k_r at temperature T_r is known. This allows for the determination of the parameter a, which can then be used to estimate $k(T)$ for $T > T_r$ using Eq.3.42. This process is demonstrated in Figure 3.10.

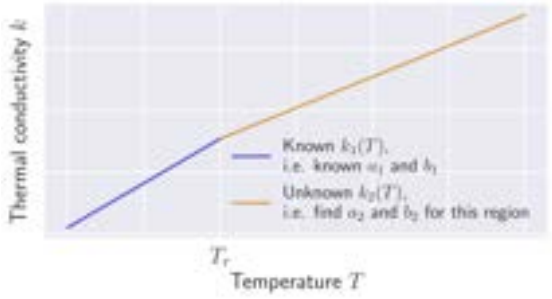


FIGURE 3.10: Linear $k(T)$ construction for $T \leq T_r$ with known $k(T)$.

Figure 3.11 shows the Jacobian calculation process. $[\boldsymbol{K}]$ is calculated for the values of $\{\boldsymbol{v}\}$ at every Gauss-Newton iteration. Temperature values at Gauss points $\{\boldsymbol{T}\}_{gp}^{iter}$ are calculated at every iteration. Then, the stiffness connectivity matrix $[\boldsymbol{K}]_{conn}$ is calculated by employing a Heaviside step function H. It is a stiffness matrix for when the temperature values at Gauss points are equal to $H\left(\{\mathbf{T}\}_{gp}^{iter} - T_r\right)\left(\{\mathbf{T}\}_{gp}^{iter} - T_r\right)$.

To summarise, two methods for constructing a linear thermal conductivity profile $k(T)$ were discussed: one using a single known point (T_{known}, k_{known}), and the other based on known values of $k(T)$ for $T \leq T_r$. These two methods can be integrated into a unified approach. This allows for the derivation of a piecewise linear relationship between k and T, originating from just one known reference point (T_{known}, k_{known}). The procedure, illustrated in Figures 3.12, 3.13, and 3.14, can be applied dynamically as the test piece is subjected to increasing temperatures. The non-linear thermal conductivity $k(T)$ is defined as a series of linear segments, with transition temperatures denoted as T_{r1}, T_{r2}, T_{r3}, and so on. Each linear segment covers a temperature interval between consecutive reference points, such as segment 1 spanning T_{r1} to T_{r2}, and segment 2 covering the range from T_{r2} to T_{r3}.

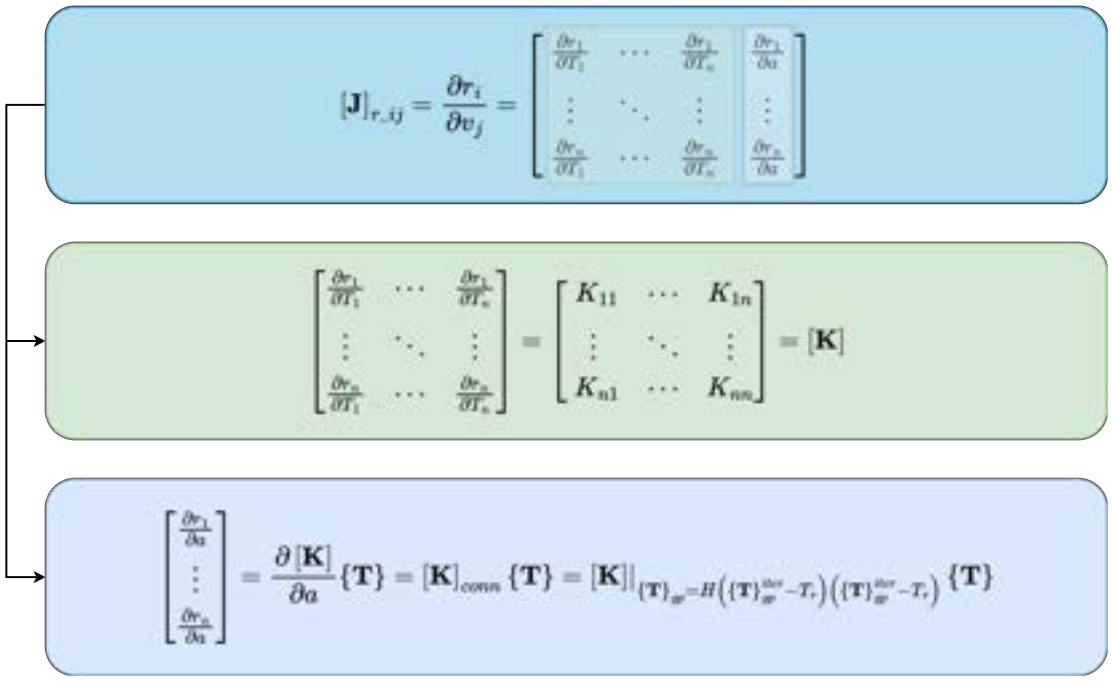


FIGURE 3.11: Calculation of Jacobian (tangent) matrix of $\{\boldsymbol{r}\}$, $[\boldsymbol{J}]_r$, for a given vector $\{\boldsymbol{v}\}$ for one material, linear $k(T)$, and known $k(T)$ for $T \leq T_{known}$. H represents Heaviside step function, while $\{\boldsymbol{T}\}_{gp}$ is a vector of temperature values at Gauss points.

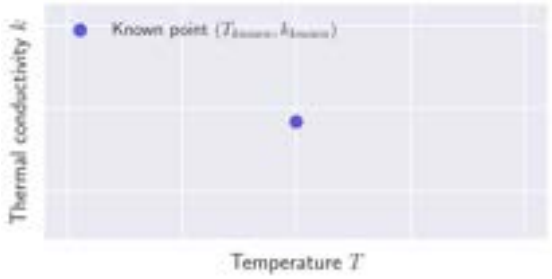


FIGURE 3.12: Step 1: Known point (T_{known}, k_{known}).

3.3.3 Thermal eigenvalue Reduced Order Modelling

The aim of ROM in this sub-section is to convert the full set of equations represented by Eq. 3.21 into a set with the smaller number of Degrees of Freedom (DOFs). This is achieved by producing a reduction matrix $[\boldsymbol{R}]$, so that the following relationship is fulfilled:

$$\{\boldsymbol{T}\} \approx [\boldsymbol{R}] \widetilde{\{\boldsymbol{T}\}} \tag{3.43}$$

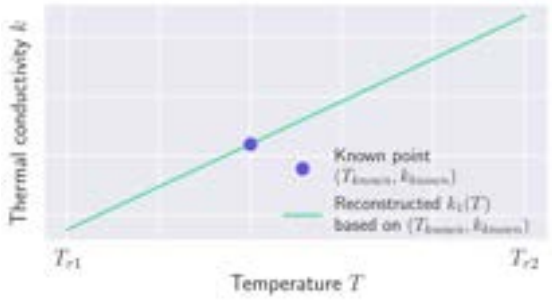


FIGURE 3.13: Step 2: T_{r1} and T_{r2} are start and end temperatures for the first linear piece, respectively. They are pre-defined. The first linear piece $k_1(T)$ is constructed using (T_{known}, k_{known}) is used to construct the first linear piece.

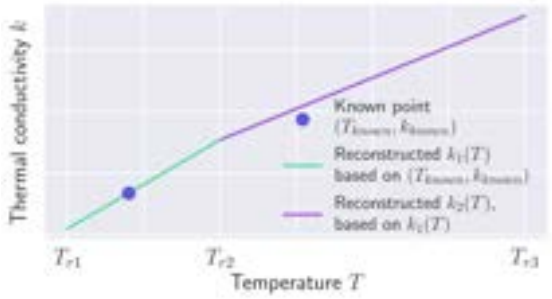


FIGURE 3.14: Step 3: T_{r2} and T_{r3} are the start and end temperatures for the second linear piece, respectively. They are pre-defined. $k_1(T)$ is used to construct the second linear piece $k_2(T)$.

Then the reduced system is obtained by multiplying Eq. 3.21 by a matrix $[\boldsymbol{P}]$:

$$\left([\boldsymbol{P}]^T [\boldsymbol{M}] [\boldsymbol{P}]\right) \left\{\frac{\Delta \boldsymbol{T}}{\Delta t}\right\} + \left([\boldsymbol{P}]^T [\boldsymbol{K}] [\boldsymbol{P}]\right) \{\boldsymbol{T}\}^{n+1} = [\boldsymbol{P}]^T \{\boldsymbol{f}\}^{n+1} \tag{3.44}$$

There are various way to define $[\boldsymbol{R}]$ and $[\boldsymbol{P}]$ [79]. The Craig–Bampton (CB) reduction technique is employed for the present work [80]. CB falls into the category of component mode synthesis methods. These methods approximate the system's response within a subspace spanned by selected linear eigenmodes, which, in the heat conduction context, correspond to the eigenfunctions of the linear heat conduction operator. These thermal eigenmodes represent independent spatial temperature distribution patterns, each associated with a characteristic decay rate governing the transient heat transfer behaviour.

In particular, CB is known for its ability to maintain the system's stability characteristics throughout the reduction process, as well as for its numerical robustness [80]. CB involves classifying the DOFs into internal and boundary categories. The internal DOFs

are approximated using a chosen set of mode shapes $[\boldsymbol{R}]$, whereas the boundary DOFs are retained directly in the transformation in Eq. 3.43.

Eq. 3.21 is re-arranged as:

$$\begin{bmatrix} [\boldsymbol{M}]_{bb} & [\boldsymbol{M}]_{bi} \\ [\boldsymbol{M}]_{ib} & [\boldsymbol{M}]_{ii} \end{bmatrix} \begin{Bmatrix} \{\Delta \boldsymbol{T}/\Delta t\}_b \\ \{\Delta \boldsymbol{T}/\Delta t\}_i \end{Bmatrix} + \begin{bmatrix} [\boldsymbol{K}]_{bb} & [\boldsymbol{K}]_{bi} \\ [\boldsymbol{K}]_{ib} & [\boldsymbol{K}]_{ii} \end{bmatrix} \begin{Bmatrix} \{\boldsymbol{T}\}_b \\ \{\boldsymbol{T}\}_i \end{Bmatrix}^{n+1} = \begin{Bmatrix} \{\boldsymbol{f}\}_b \\ \{\boldsymbol{f}\}_i \end{Bmatrix}^{n+1} \tag{3.45}$$

where internal DOFs are notated by subscript i, and boundary DOFs by subscript b. The following eigenproblem is solved to calculate fixed-boundary eigenmodes:

$$\left(\lambda \left[\boldsymbol{M}\right]_{ii} + \left[\boldsymbol{K}\right]_{ii}\right) \phi = 0 \tag{3.46}$$

Then a matrix $[\boldsymbol{\Phi}]_{im}$ containing eigenmodes correlated with a set of lowest eigenvalues is assembled.

The reduction matrix also incorporates the static response of the full system to specified boundary loads in the form of Guyan static constraint modes. They account for the quasi-static response of the internal DOFs induced by prescribed boundary temperatures. Each Guyan mode represents the steady-state temperature distribution of the internal DOFs resulting from a unit temperature applied at a single boundary Degree of Freedom (DOF), while all other boundary DOFss are held fixed. These modes are obtained by statically condensing the internal DOFs and ensure that the reduced model accurately reproduces the static heat conduction behaviour associated with boundary loading. The i^{th} Guyan mode describes how the internal DOFs respond when a unit temperature is applied at the i^{th} boundary node, while all other boundary nodes are held at zero temperature.

Finally, the reduction matrix could be constructed as follows:

$$[\boldsymbol{R}] = \begin{bmatrix} [\boldsymbol{\Phi}]_{static} & [\boldsymbol{\Phi}]_{dynamic} \end{bmatrix} = \begin{bmatrix} [\boldsymbol{I}]_{bb} & [\boldsymbol{0}]_{bm} \\ -\left[\boldsymbol{K}\right]_{ii}^{-1} \left[\boldsymbol{K}\right]_{ib} & [\boldsymbol{\Phi}]_{im} \end{bmatrix} \tag{3.47}$$

where $[\boldsymbol{I}]$ is the identity matrix. In Eq. 3.44 $[\boldsymbol{P}]$ is set to be equal to $[\boldsymbol{R}]$. Following from Eq. 3.44, the reduced global capacitance and stiffness matrices and loading vector

are equal to:

$$\begin{aligned}\widetilde{[\boldsymbol{M}]} &= [\boldsymbol{R}]^T [\boldsymbol{M}] [\boldsymbol{R}] \\ \widetilde{[\boldsymbol{K}]} &= [\boldsymbol{R}]^T [\boldsymbol{K}] [\boldsymbol{R}] \\ \widetilde{\{\boldsymbol{f}\}} &= [\boldsymbol{R}]^T \{\boldsymbol{f}\}\end{aligned} \tag{3.48}$$

This separation of the nodes into boundary and internal ones becomes especially important in problems involving contact, for example. CB ensures that the contact forces act directly on the boundary DOFs of the reduced system. Similarly, this feature of the reduction approach benefits the solution constriction process introduced in Subsection 3.3.2.1, as it allows the process to stay unchanged. The nodes used in the regularisation component as well as the measurement nodes are grouped into the boundary node set and thus stay unreduced.

The reduced loss vector is defined as:

$$\widetilde{\{\boldsymbol{L}\}} = \begin{Bmatrix} \widetilde{\{\boldsymbol{L}\}}_R \\ \widetilde{\{\boldsymbol{L}\}}_M \\ \widetilde{\{\boldsymbol{L}\}}_{REG} \end{Bmatrix} \tag{3.49}$$

The reduced components of $\widetilde{\{\boldsymbol{L}\}}$ are:

$$\begin{aligned}\widetilde{\{\boldsymbol{L}\}}_R &= \widetilde{\{\boldsymbol{R}\}}_{g-q} \\ \widetilde{\{\boldsymbol{L}\}}_M &= \{\boldsymbol{s}\}_M \odot \left(\widetilde{\{\boldsymbol{T}\}}_i - \widetilde{\{\boldsymbol{T}\}}_{Mi}\right) \\ \widetilde{\{\boldsymbol{L}\}}_{REG} &= \begin{Bmatrix} \{\boldsymbol{s}\}_1 \odot \left(\widetilde{\{\boldsymbol{f}\}}_q - \frac{\sum_{i\in\Gamma_q\setminus\Gamma_{edge}} \widetilde{\{\boldsymbol{f}\}}_{qi}}{|\Gamma_q\setminus\Gamma_{edge}|}\right) \\ \{\boldsymbol{s}\}_2 \odot \left(\widetilde{\{\boldsymbol{f}\}}_q - \frac{\sum_{i\in\Gamma_{edge}\setminus\Gamma_{corners}} \widetilde{\{\boldsymbol{f}\}}_{qi}}{|\Gamma_{edge}\setminus\Gamma_{corners}|}\right) \\ \{\boldsymbol{s}\}_3 \odot \left(\widetilde{\{\boldsymbol{f}\}}_q - \frac{\sum_{i\in\Gamma_{corners}} \widetilde{\{\boldsymbol{f}\}}_{qi}}{|\Gamma_{corners}|}\right) \end{Bmatrix}\end{aligned} \tag{3.50}$$

Overall, the reduction in computational time stems from the system reduction, i.e., calculating only a limited number of eigenvectors. The time can be further decreased by updating the matrices and vectors corresponding to coolant convection not at every iteration and time step, but only at every second time step, using the temperature distribution from the preceding step.

3.3.4 Proportional–Integral–Derivative controller

PID control is a widely used and well-established feedback control strategy commonly used in various dynamic systems in engineering. Its popularity stems from its relative simplicity and straightforwardness of implementation, as well as its remarkable effectiveness in a broad range of applications [81, 82]. They range from motor speed control and robotics [83] to chemical process regulation [84].

The FE-based reconstruction approach allows for the determination of the full temperature field within the HIVE sample using only a limited number of temperature measurements. A key objective of the DT is to facilitate real-time monitoring and regulation of system parameters, for example ensuring the temperature remains below a critical threshold. In this work, this objective is accomplished by adjusting the velocity of the cooling water. Although regulating the applied heat flux is also a viable control strategy, the focus here is solely on manipulating water flow velocity. The objective of the feedback control is to minimise a control error, defined as follows:

$$e(t) = y_{sp}(t) - y_m(t) \tag{3.51}$$

where $y_{sp}(t)$ is a set point or target for the controlled variable and $y_m(t)$ is a controlled variable measurement. The control error can potentially vary with time, but it normally stays constant for significant time periods. Proportional control mode ensures that the controller output is proportional to the control error (Eq. 3.51):

$$u(t) = \overline{u} + K_C e(t) \tag{3.52}$$

where $u(t)$ and $\overline{u}$ are controller output and its nominal value, respectively. K_C is dimensionless controller gain. There are two fundamental principles of proportional control. Namely, K_C can be tuned to make the controller's response to differences between $y_{sp}(t)$ and $y_m(t)$ as responsive as needed, and the sign of K_C can be selected so that the $u(t)$ either increases or decreases as $e(t)$ grows.

Integral control mode ensures that the controller output is dependent on the control error's integral over time:

$$u(t) = \overline{u} + \frac{1}{\tau_I} \int_0^t e(t^*) dt^* \tag{3.53}$$

where τ_D is integral time (or reset time), and it has units of time.

The integral eliminates steady-state offset, which is a significant benefit. For the process to reach a steady state, $u(t)$ should remain constant so that $y_m(t)$ also stays fixed. However, Eq. 3.53 shows that $u(t)$ continues to change over time unless the error $e(t)$ becomes zero. Therefore, with integral action, $u(t)$ gradually adjusts until it reaches a value that eliminates the steady-state error. This favourable outcome typically occurs unless u saturated, i.e. when it reaches its operational limit. During saturation $u(t)$ can no longer influence the system to bring $y_m(t)$ back to the desired $y_{sp}(t)$. Such saturation can happen when a disturbance or the change in$y_{sp}(t)$ is too large for the available control range. Increase in the integral mode after saturation is refereed to as reset (or integral) windup.

Finally, the derivative control mode is represented as:

$$u(t) = \overline{u} + \tau_D \frac{de(t)}{dt} \tag{3.54}$$

where τ_D is the derivative time, and, similarly to integral mode it has units of time. Derivative mode works by predicting the future trend of $e(t)$ based on how quickly it is changing. This predictive capability is something that proportional and integral modes alone cannot provide. A proportional mode only responds to the current size of $e(t)$, without accounting for how rapidly $e(t)$ is changing over time. Similarly, integral control reacts to the duration of $e(t)$, making it slow to respond to sudden changes. However, when $e(t)$ remains constant, the derivative mode's output stays unchanged. For this reason, it is never used by itself. Additionally, by offering anticipatory behaviour, derivative mode helps stabilise the system, making it particularly effective at offsetting the destabilising effects of integral mode. Finally, the derivative mode improves the system's dynamic performance by reducing the time it takes for the output to settle into a steady state (settling time).

Proportional, integral, and derivative control modes are combined to form the PID controller. Its parallel configuration is defined as follows [81]:

$$u(t) = \overline{u} + K_C \left[e(t) + \frac{1}{\tau_I} \int_0^t e(t^*)dt^* + \tau_D \frac{de(t)}{dt} \right] \tag{3.55}$$

To convert Eq 3.55 into the Laplace domain, the Laplace transform is applied to each term. The Laplace transform of $e(t)$ is $E(s)$, where s the complex frequency variable used in the Laplace transform domain. The Laplace transform of $de(t)/dt$ is $sE(s)$. Consequently, the Laplace transform of Eq 3.55 is as follows:

$$U'(s) = K_C \left[E(s) + \frac{1}{\tau_I} E(s) + \tau_D s E(s) \right] \tag{3.56}$$

Then, the transfer function corresponding to Eq. 3.55 is equal to:

$$\frac{U'(s)}{E(s)} = K_C \left[1 + \frac{1}{\tau_I s} + \tau_D s \right] \tag{3.57}$$

Hence, PID controller in its parallel form could be represented as three distinct blocks functioning in parallel (Figure 3.15).

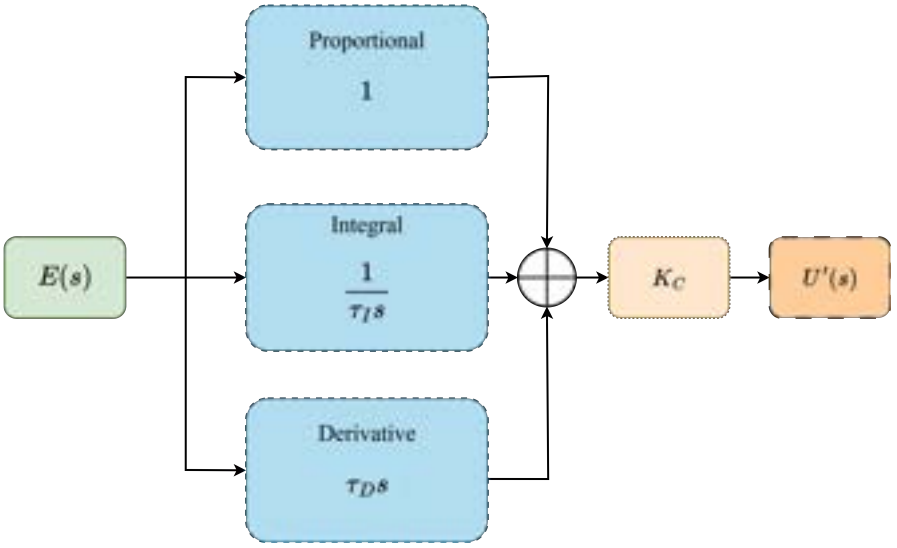


FIGURE 3.15: PID controller in its parallel configuration.

Eq. 3.55 could be represented in the expanded form as follows:

$$u(t) = \overline{u} + K_C e(t) + K_I \int_0^t e(t^*)dt^* + K_D \frac{de(t)}{dt} \tag{3.58}$$

where K_I and K_D are the integral and derivative gains, respectively. This configuration may seem ideal for controller tuning since each gain affects only a single mode of control independently. However, the common tuning methods used in this work were specifically designed for the series and parallel configurations. Consequently, Eq. 3.55 could be used.

Eq. 3.55 and 3.58 considers the controller's input and output signals to be continuous over time. However, digital control systems are far more prevalent due to their adaptability, processing capabilities, and cost efficiency. In a digital feedback control setup, both the input and output signals are discrete (or digital) rather than continuous (analog). To enable this, the analog signal from the sensor or transmitter is sampled and transformed into a digital signal using an analog-to-digital converter (ADC). A digital control algorithm then processes this signal to generate a digital output. For this work the digital controller is necessary since the temperature measurements are gathered discretely over time.

The parallel PID configuration represented by Eq. 3.55 can be discretised in time using the rectangular and forward difference approximations:

$$\begin{aligned} \int_0^t e(t^*)dt^* &\approx \sum_{j=1}^{k} e_j \Delta t \\ \frac{de(t)}{dt} &\approx \frac{e_n - e_{n-1}}{\Delta t} \end{aligned} \tag{3.59}$$

where the discrete control error is:

$$e_n = y_{sp\_n} - y_{m\_n} \tag{3.60}$$

Eq. 3.59 is substituted into Eq. 3.55, and a position form is obtained:

$$u_n = \overline{u} + K_C \left[e_n + \frac{\Delta t}{\tau_I} \sum_{j=1}^{n} e_j + \frac{\tau_D}{\Delta t} \left(e_n - e_{n-1} \right) \right] \tag{3.61}$$

Alternatively, using a velocity form, a change in $u(t)$ could be computed by re-writing Eq. 3.61 for $(n+1)^{\text{th}}$ time instant:

$$u_{n-1} = \overline{u} + K_C \left[e_{n-1} + \frac{\Delta t}{\tau_I} \sum_{j=1}^{n-1} e_j + \frac{\tau_D}{\Delta t} \left(e_{n-1} - e_{n-2} \right) \right] \tag{3.62}$$

The velocity form is derived by combining Eq. 3.61 and Eq. 3.62:

$$\Delta u_n = u_n - u_{n-1} = K_C \left[(e_n - e_{n-1}) + \frac{\Delta t}{\tau_I} e_n + \frac{\tau_D}{\Delta t} (e_n - 2e_{n-1} + e_{n-2}) \right] \tag{3.63}$$

This form can be beneficial due to the fact that it does not employ the errors' sum. It make this for windup resistant.

The parameters K_c, τ_I, and τ_D should be adjusted appropriately to ensure effective control performance. Thus, the procedure for tuning these values is addressed later.

Since the objective is to regulate the maximum temperature within the HIVE sample (Section 3.2), a discrete, problem-specific control error is calculated as

$$e_n = T_{sp\_n} - T_{max\_n} \tag{3.64}$$

where T_{sp_n} denotes the desired maximum temperature (set point) and T_{max_n} corresponds to the maximum temperature observed in the HIVE sample at the n^{th} time step.

In this setup, $u(t)$ is assigned to be the coolant velocity v_c. During the tuning phase the position form of the controller (Eq.3.61) is employed. Once tuning is completed, the system transitions to using the velocity form (Eq.3.63) for normal operation. The tuning process is described in Section 5.1. A direct-acting controller with $K_c < 0$ is selected, as an increase in T_{max_k} should lead to a corresponding increase in v_c.

3.4 Physics-Driven Machine Learning-based Digital Twinning

3.4.1 Workflow

Figure 3.16 shows the assembled PD-ML DT control loop. The rest of this section describes each component in detail.

This framework is an alternative to the previously introduced FE DT approach (Subsection 3.3.1). Its goal is also to regulate and control the maximum temperature within the HIVE sample during the experiment. While FE DT and PD-ML DT frameworks

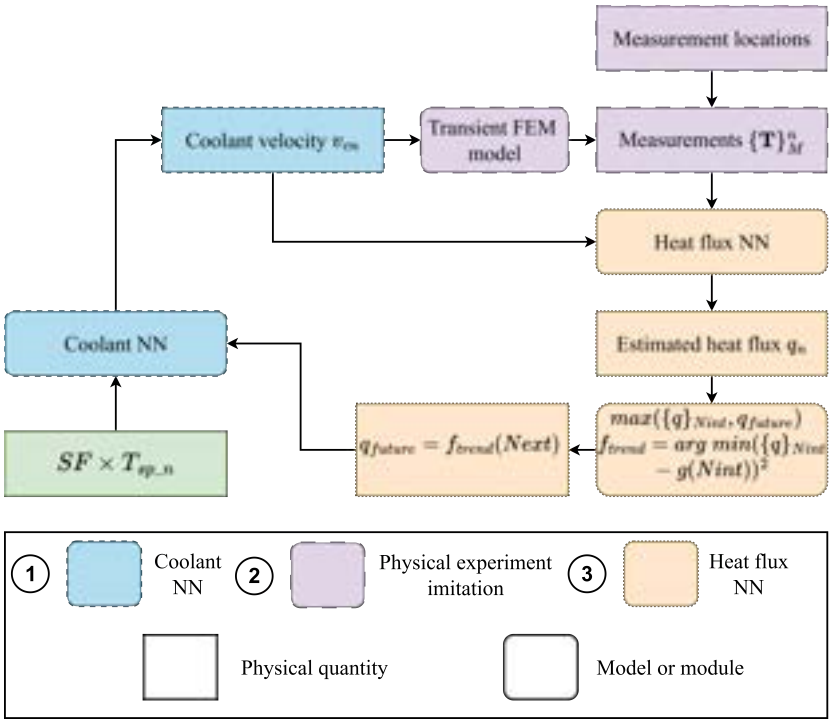


FIGURE 3.16: PD-ML DT loop combining two ML models.

serve the same thermal management objective, they embody fundamentally different philosophies. The former is rooted in physics-based modelling with classical control theory, and the latter is based on data-driven ML. PD-ML DT employs two NNs, which are referred to as coolant NN and the heat flux NN.

The loop begins with a scaled temperature set point, defined as $SF \times T_{sp\_n}$, where SF is a user-defined safety factor that accounts for prediction margins and operating tolerances. In previous work [68], SF was chosen to be greater than 1.0 to compensate for the absence of extrapolation in the process, as discussed further. In this work, however, SF is set to 1.0, since the use of heat-flux extrapolation already provides predictive capability, making an additional safety margin unnecessary in this context. Although SF could also be incorporated into the FE DT framework as well, the aim here is to evaluate the pure control achieved by the two DT approaches. It should be noted that an additional safety factor can always be introduced later, depending on the level of measurement noise or uncertainty in the system.

This scaled set point is passed as an input to the coolant NN, which is trained to predict the necessary coolant velocity v_{cn} required to maintain system temperatures below the target threshold. The coolant NN is trained offline using steady-state simulation data,

and is capable of approximating nonlinear relationships between the desired maximum temperature and cooling demand.

The updated coolant velocity v_{cn} is then applied to the forward FE model, which simulates the thermal response of the HIVE sample under the given cooling condition. The forward FE model produces a set of pseudo temperature measurements $\{\mathbf{T}\}_M^{n+1}$. The temperature measurements are passed to the heat flux NN, which is trained on the steady-state simulation data to estimate the surface heat flux q_n from them. This NN is effectively trained to learn the inverse mapping between sparse measurements and the thermal loading conditions that produced them.

The next part of the process is a heat flux prediction module. This module analyses a sequence of recently estimated flux values, denoted $\{q\}_{Nint}$, over an interpolation time window $Nint$. A function f_{trend} is created by interpolating $\{q\}_{Nint}$, then the future heat flux value q_{future} is estimated by evaluating f_{trend} at extrapolation time step $\{q\}_{Next}$. The maximum heat flux value is selected out of $max(\{q\}_{Nint}, q_{future})$. This module is necessary to account for the fact that the steady-state data is used to train two NNs. It should be noted that this extrapolation remains valid as long as the change in heat flux is not too abrupt relative to the time step size used for control.

The output from the heat flux trend prediction feeds back into the coolant NN, enabling it to adapt its output coolant velocity based not only on the present estimated heat flux but also on anticipated future heat loads.

3.4.2 Neural Network

NNs are widely used and highly effective ML tools suitable for addressing a broad range of engineering tasks. A NN consists of interconnected units known as nodes or neurons, drawing inspiration from the structure and functionality of biological NNs found in the brains of humans and animals. The nature and strength of the connections between these neurons vary depending on the specific type of NN architecture employed.

Among the various types of NNs, the MLP is one of the most commonly used architectures. MLPs consist of one or more hidden layers situated between the input and output layers, as shown in Figure 3.17. They are often called Deep Neural Networks (DNNs) due to the multiple layers that make up their architecture. In contrast to the

simple perceptron, where inputs are directly connected to outputs, the MLP enables more complex mappings by utilising these intermediate layers. Each neuron in a given layer is linked to every neuron in the next layer through weighted connections. These weights are learned through the training process, allowing the network to adjust and optimise its predictions based on input-output data relationships.

Each layer in the network is capable of learning and extracting distinct features from the input data. As the depth of the network increases, the model gains the ability to recognise increasingly complex and abstract patterns. The input layer contains a number of neurons equal to the number of input variables i, while the output layer consists of nodes corresponding to the number of outputs j the model is expected to produce. The following equations represent the relationship shown in Figure 3.17, i.e. the forward propagation process:

$$a_k^1 = g_1\left(\sum_{l=1}^{i} x_l W_{kl}^1\right), \quad k = 1, \ldots, n_1 \tag{3.65}$$

$$a_k^p = g_p\left(\sum_{l=1}^{n_{p-1}} a_l^{p-1} W_{kl}^p\right), \quad k = 1, \ldots, n_p, \quad p = 2, \ldots, L \tag{3.66}$$

$$y_k = g_o\left(\sum_{l=1}^{n_L} a_l^L W_{kl}^{L+1}\right), \quad k = 1, \ldots, j \tag{3.67}$$

The process begins with Eq. 3.65, where each neuron a_k^1 in the first hidden layer computes a weighted sum of the input features x_l using weights W_{kl}^1, and applies a non-linear activation function g_1 to the result. Eq. 3.66 generalises this operation for deeper hidden layers, where the activation a_k^p of a neuron in layer p is computed from the activations a_k^{p-1} of the previous layer using weights W_{kl}^p and the activation function g_p. Finally, Eq. 3.67 calculates the network outputs y_k by applying the activation function g_o to the weighted sum of the last hidden layer's activations a_l^L using weights W_{kl}^{L+1}.

Activation functions play a crucial role in DNNs because they introduce non-linearity into the network, allowing it to model complex relationships. It is not necessary to apply the same activation function in every layer, as it is done in Eqs. 3.65-3.67. Popular

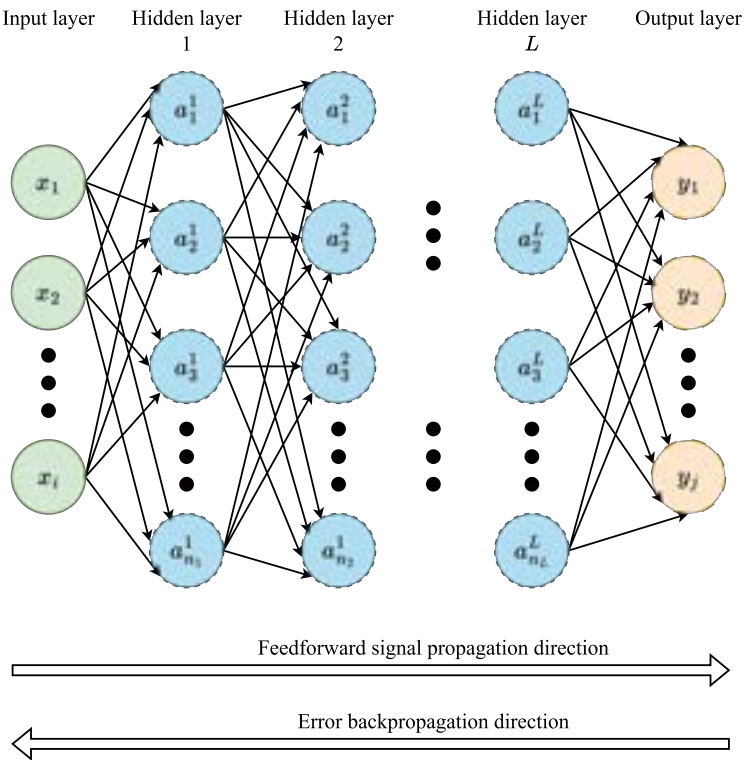


FIGURE 3.17: MLP NN diagram.

activation functions, tanh, sigmoid, leaky ReLU, and swish, are defined as follows:

$$
\begin{aligned}
g_{tanh}(z) &= \frac{(e^{z} - e^{-z})}{(e^{z} + e^{-z})} \\
g_{sigmoid}(z) &= \frac{1}{(1 + e^{-z})} \\
g_{ReLU}(z) &= \begin{cases} 0.01x & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases} \\
g_{swish}(z) &= \frac{z}{(1 + e^{-z})}
\end{aligned}
\tag{3.68}
$$

The initial stage in developing a DNN involves selecting its architecture, which includes determining how many hidden layers the network have and how many neurons are in each layer. This architectural choice is closely linked to other hyperparameters, meaning adjustments in one area can influence overall model performance. As a result, there is no universal rule for selecting an optimal design. A typical strategy is to begin with a configuration that has worked well for similar tasks. From there, the architecture can be fine-tuned by modifying the number of layers or neurons, and by experimenting with different activation functions, until a suitable model is achieved. However, since each

of these elements can vary widely, the number of possible model combinations becomes enormous, making the training process potentially very time-consuming, particularly for larger DNNs.

The most widely used loss function used during the training is the Mean Squared Error (MSE) computed across the training dataset. The standard approach for adjusting the DNN's parameters is the backpropagation algorithm [42]. This process begins by computing the gradient of the loss function with respect to the weights in the output layer, and then uses these gradients to determine the gradients in the preceding layers, moving backward through the DNN. Once all the gradients are obtained, the weights are updated. Because DNNs typically involve a large number of parameters to train, first-order optimisation methods are commonly employed to carry out the updates efficiently. Some of the examples include Stochastic Gradient Descent (SGD) and Adam [42, 44].

When the gradients are computed using the whole training dataset, the method is known as batch gradient descent. In this approach, the gradient of the loss function is computed using all training samples, and the model parameters are updated once per iteration based on this global gradient. Although this approach is straightforward, it can be computationally expensive for large datasets and may lead to suboptimal outcomes by converging to sharp local minima [85]. A widely adopted alternative is the mini-batch gradient descent, where gradients are computed using smaller subsets of the data. After computing the gradients for one mini-batch, the model weights are updated before proceeding to the subsequent batch. Introducing this randomness into the optimisation process helps the model avoid sharp minima, thus improving both performance and generalisation. Typically, mini-batch sizes range from 16 to 512 samples. The method is called stochastic gradient descent, when only one data point is used per update.

Due to the typically large number of trainable parameters, overfitting can become a significant concern. Therefore, several strategies have been developed to mitigate this issue. One widely used method is dropout, which involves randomly setting the output of certain neurons to zero during the training process. The likelihood of each neuron being dropped is specified before training begins. This technique can be applied to multiple layers in the DNN. Dropout helps prevent the model from becoming overly reliant on specific neurons, thereby enhancing its ability to generalise to unseen data. Another common approach is to track the DNN's performance on both the training and

validation datasets. By monitoring the loss values during training, the process could be stopped early once the validation loss begins to rise, indicating potential overfitting. Additionally, because models with many parameters often have complex loss landscapes, they tend to contain numerous local minima. The final solution reached during training can therefore depend heavily on the initial random weight configuration.

The architecture of DNNs allows them to produce multiple outputs, making them suitable for directly modelling a full-field FE surrogate model. However, building such a model would necessitate a DNN of considerable size, involving a vast number of trainable parameters. This increased complexity results in longer training durations and significantly raises the risk of overfitting, as the abundance of parameters may cause the model to capture noise in the training data rather than general patterns. Consequently, the construction of the full-field FE surrogate model is bypassed in this work.

3.4.3 Coolant Neural Network

Figure 3.18 illustrates the training procedure for the coolant NN, which forms a key component of the broader PD-ML DT framework shown previously in Figure 3.16. This coolant NN is designed to avoid constructing a full-field surrogate model, which would require high-dimensional input and output data, such as spatially distributed temperatures or heat fluxes across the entire domain. This approach significantly reduces complexity by focusing only on a few scalar quantities. Specifically, the coolant NN is trained to predict the coolant velocity v_{cNN} required to maintain a desired maximum temperature.

During the training process, different combinations of heat flux q and coolant velocity v_c are fed into the steady-state forward FE model to simulate the resulting temperature distribution. From each simulation, only the maximum temperature $max\{\mathbf{T}\}$ is extracted and paired with the corresponding coolant velocity. Thus, the training, validation, and testing datasets are created. The NN receives the maximum temperature value from the steady-state FEM simulations together with q used to produce it as its inputs. The output for NN is v_{cNN}, which is compared with v_c during the training and validation processes. These input–output pairs are then used to train the NN to approximate the inverse mapping going from maximum temperature to coolant velocity.

Once the NN is trained and tested, it can receive the maximum temperature set point $SF \times T_{sp\_n}$ and current estimated heat flux q as its inputs and output coolant velocity which should be used at the next time step of the DT loop.

This approach greatly simplifies the model by reducing the number of required training parameters and avoiding the need to model full-field spatial responses. As a result, the coolant NN can be efficiently integrated into the PD-ML DT loop for real-time inference. It is able to contribute to the control of the maximum temperature within the HIVE sample without the computational burden of training or evaluating a full surrogate FE model.

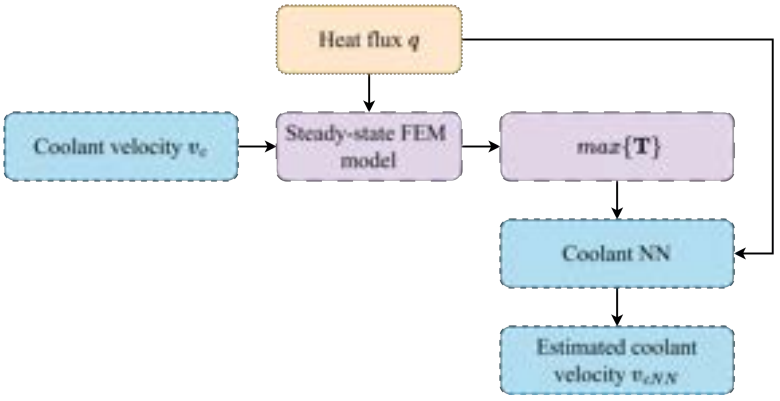


FIGURE 3.18: Training process of coolant NN.

3.4.4 Heat flux Neural Network

Figure 3.19 depicts the training workflow for the heat flux NN, which forms another integral component of the PD-ML DT framework. The goal of this NN is to estimate the applied heat flux q based on a number of temperature measurements $\{\mathbf{T}\}_M$ at predefined sensor locations. This inverse modelling approach avoids the need to infer high-resolution thermal fields or detailed boundary conditions and instead focuses on learning a low-dimensional mapping from sensor data to a scalar heat input.

The training procedure begins by running multiple steady-state FEM simulations using different values of coolant velocity v_c and heat flux q as inputs. For each simulation, the resulting temperature distribution is sampled at fixed measurement locations to extract the measurement vector $\{\mathbf{T}\}_M$. It emulates the sensor data available in the real system. These vectors, paired with their corresponding input heat flux values and coolant velocities, are used to train the heat flux NN.

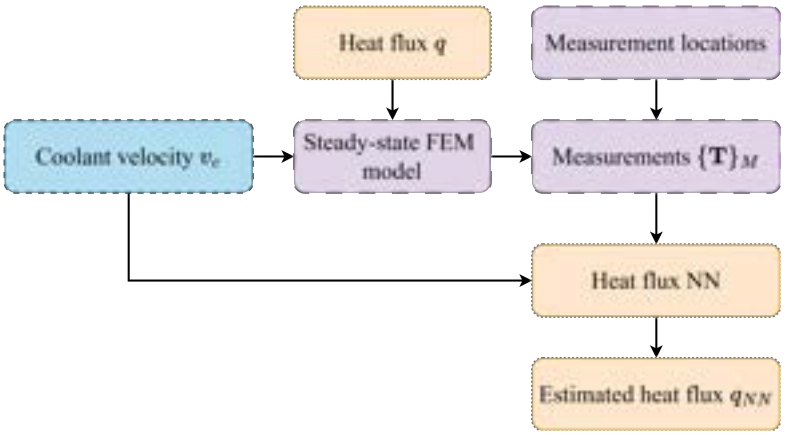


FIGURE 3.19: Training process of heat flux NN.

In doing so, the model learns to infer the unknown thermal loading conditions from sparse temperature data, enabling real-time estimation of the heat flux during operation. As with the coolant NN, this strategy offers a significant reduction in model dimensionality compared to full-field surrogate models. It enhances computational efficiency and making the approach well suited for integration into real-time DT control schemes.

Chapter 4

Solution and thermal conductivity construction using FE-based approach

4.1 Forward model

This section describes the forward FE model used in this work. The purpose of this model is threefold. Firstly, it is used as an imitation of the physical experiment used to collect the pseudo measurements in order to imitate the DT process. Secondly, it serves as a reference used to compare with the solution constructed from measurements in the case of FE DT approach. Thirdly, it is used to calculate the FE matrices and vectors to perform a FE-based solution construction for FE DT.

4.1.1 Finite Element solver

The VL platform was created for HIVE simulations in previous work [19]. However, this platform relies on Salome Meca / Code_Aster software [7], which has a number of limitations restricting its application to this work. Primarily, it is challenging to couple it with the Python code in a manner that is suitable for the simulation of DT process. In particular, it is not possible apply a temperature distribution array on the mesh nodes as IC, which is essential when the BC are time-variant and the precise values

are not known a priori before control, step is complete. Furthermore, it was found that Code_Aster is not very well suitable for non-linear FE matrix and vector extraction, which is crucial for FE DT. Code_Aster only allows such extraction for temperature independent material properties and BC.

Consequently, Netgen/NGSolve, high-performance Python-based open-source FE solver, was selected to perform FE simulations in this work [86]. It is widely used for various analysis types, including thermal simulations, CFD, and EM [87–89]. It is highly flexible, which is evidenced by the ease of FE matrix and vector extraction and IC application in the form of temperature distribution array. Moreover, parallel computing is fully developed in NGSolve as opposed to other Python-based FE solvers, such as Sfepy [90], which have it only partially implemented.

4.1.2 1D coolant model

It is necessary to model the effect pressurised water flow has on the sample's temperature (Section 3.2). Fully 3D CFD model of the water pipe is computationally expensive, impractical, and unnecessary for this application. Previous research into the cooling pipes have shown that their heat transfer could be simplified and represented using a 1D model [91] defined by a boiling curve. This curve provides the dependency of heat flux q between the pipe and coolant on the pipe wall temperature, T_{wall}. A typical water boiling curve is illustrated in Figure 4.1.

Marshall et al. [91] developed a 1D coolant model predicting the divertor thermal behaviour in fusion reactors, which included a representation of heat transfer over the full range of operating conditions. Previous correlations for the boiling curve addressed isolated regimes, such as forced convection, nucleate boiling, or Critical Heat Flux (CHF), but lacked a unified, physically consistent framework. The work by Marshall et al. [91] presented an integrated heat transfer model that captured all regimes of the boiling curve relevant to water-cooled fusion divertor channels. Their coolant model includes pre- and post- CHF regimes. It combines regime-specific correlations into a comprehensive formulation applicable to fusion-relevant heat fluxes and flow conditions. Furthermore, it was validated against experimental data. Consequently, it is selected for the present work.

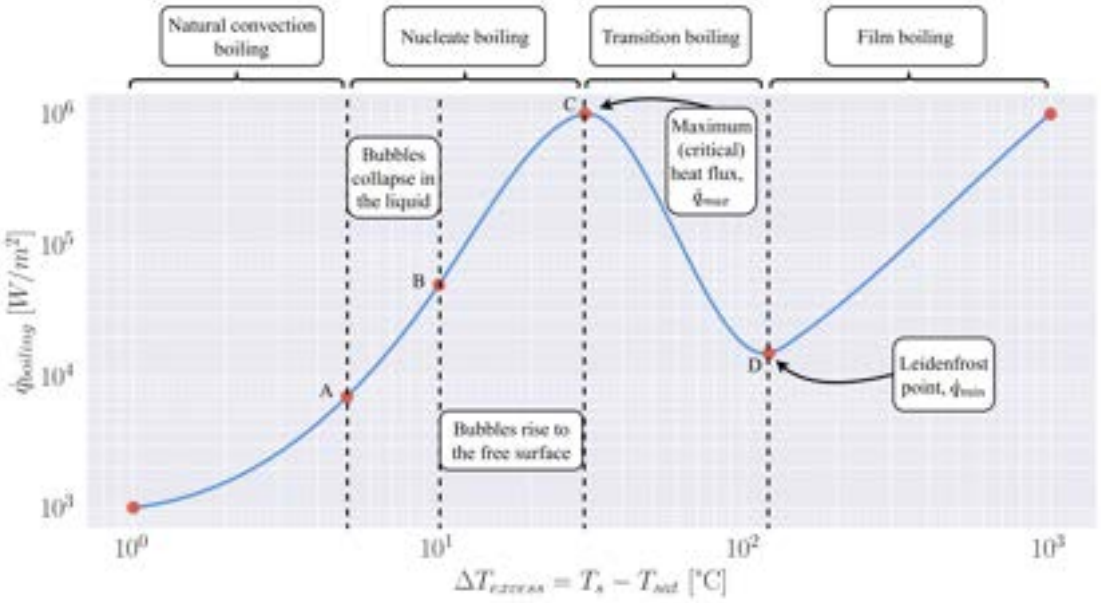


FIGURE 4.1: Example of water boiling curve at the pressure of 1atm, the data points used to create this figure were taken from the book by Cengel [2]. Here, T_s is the surface temperature of the heated solid, i.e. the wall in contact with the liquid.

IAPWS97 Python package [92], based on the information provided by the International Association for the Properties of Water and Steam (IAPWS) [93], is used to calculate water properties for the defined velocity, temperature, and pressure. This range of water proprieties is used during the calculations performed at various regimes of the boiling curve.

For the optimal heat transfer efficiency, the pipe wall temperature T_{wall} should be maintained below the CHF threshold. Exceeding this limit pushes the system into transition (film boiling) stages where a vapour layer blankets the surface. Because vapour acts as an insulator, it severely inhibits the exchange of thermal energy between the wall and the fluid. Consequently, the following correlations focus on the convection and nucleate boiling phases, where heat transfer remains most effective.

When T_{wall} remains below the saturation temperature T_{sat} of the coolant, the system operates within the single-phase regime. Heat transfer in this state occurs via two primary mechanisms: natural and forced convection. During the former, the fluid movement is generated spontaneously by internal forces, such as density gradients and buoyancy. During the latter, the fluid flow is powered by external hardware such as fans or pumps. Forced convection is relevant to HIVE.

The water is in the forced convection regime when T_{wall} is lower than saturation temperature T_{sat}. During this regime, q is calculated as follows:

$$q = h(T_{wall} - T_{water}) \tag{4.1}$$

where h is the heat transfer coefficient and T_{water} is the temperature of the water. h could be estimated as:

$$h = Nu\frac{k_{water}}{l_{pipe}} \tag{4.2}$$

where Nu is a Nusselt number, k is the water's thermal conductivity, and l_{pipe} is the pipe's length, which is known. k_{water} is derived from the aforementioned IAPWS97 package.

Nu could be modelled using Sieder-Tate correlation [91, 94], which is an extension of Dittus-Boelter correlation [95]:

$$\begin{gathered} Nu = 0.027 Re_{d_{pipe}}^{4/5} Pr^{1/3} \left(\frac{\mu_{water}}{\mu_{wall}}\right)^{0.14} \\ 0.7 \leq Pr \leq 16700\ ,\ Re_{d_{pipe}} \geq 10,000\ ,\ \frac{l_{pipe}}{d_{pipe}} \geq 10 \end{gathered} \tag{4.3}$$

where Re is the Reynolds number, Pr is the Prandtl number, and d_{pipe} is the pipe's diameter. Finally, μ_{water} and μ_{wall} are the bulk water and wall water viscosities, respectively. This correlation is adopted due to its strong agreement with prior heat transfer experiments conducted at Sandia National Laboratories [91].

The water starts to transition into the nucleate boiling regime once it reaches T_{sat}. In the nucleate boiling regime, the process begins with fluid evaporation at the pipe's surface. When the difference between wall and saturation temperatures ΔT_{sat} is slightly above zero, heat transfer still follows the single-phase correlations. As the temperature rises, steam bubbles begin to form. These bubbles enhance heat transfer efficiency by physically mixing the fluid as they move. Initially, these bubbles are small and collapse shortly after leaving the surface. However, as ΔT_{sat} continues to grow these individual bubbles merge into continuous vapour jets.

To model nucleate boiling effectively, the specific conditions that trigger it need to be determined. This threshold, known as the Onset of Nucleate Boiling (ONB), can

be estimated using Bergles and Rohsenow correlation, which estimates the heat flux required for ONB [96]:

$$q_{ONB} = 1082(10P_{water})^{1.156}(1.8(T_{wall} - T_{ONB}))^{\frac{2.16}{(10P_{water})^{0.0234}}}$$
$$0.1 \leq P_{water} \leq 13.8 \tag{4.4}$$

where P_{water} is the water pressure, and T_{ONB} is the temperature during ONB. The wall temperature necessary for the ONB is determined by setting the ONB correlation equal to the forced convection correlation. If the wall temperature exceeds this value, the flow enters the nucleate boiling regime and requires a different modelling approach.

After the water fully transitions into the nucleate boiling regime the following Japan Atomic Energy Research Institute (JAERI) correlation could be used: [97]:

$$q_{NB} = 10^6 e^{\frac{P_{water}}{8.6}} \left(\frac{T_{wall} - T_{ONB}}{25.72} \right)^3$$
$$30 \leq T_{water} < 80 \ , \ 0.5 \leq P_{water} \leq 1.6 \tag{4.5}$$

An asymptotic model is used in order to ensure a smooth transition between two regimes, forced convection and nucleate boiling [96]. This model account for heat transfer by combining the effects of forced convection q_{FC} and nucleate boiling q_{NB}:

$$q = q_{FC} \left(1 + \frac{q_{NB}}{q_{FC}} \left(1 - \frac{q_{ONB}}{q_{NB}} \right)^2 \right)^{1/2} \tag{4.6}$$

4.1.3 Forward NGSolve model

The 1D coolant model is used to calculate the temperature-dependent sample-coolant convection coefficient $h_c(T)$, which is applied as a BC together with the heat flux q. The problem is non-linear as the material properties are temperature-dependent, hence Newton's iterative approach is used to solve the FE system of equations. For each time-step (or in the beginning of the solution process for a steady-state problem), the initial guess is given as $\{\boldsymbol{T}\}^0$. The discretised system of equations could be written as follows:

$$[\boldsymbol{A}]\{\boldsymbol{T}\} = \{\boldsymbol{f}\} \tag{4.7}$$

Then for each iteration s, the residual is computed as:

$$\{\boldsymbol{r}\}^s = [\boldsymbol{A}]^s \{\boldsymbol{T}\}^s - \{\boldsymbol{f}\}^s \tag{4.8}$$

The Jacobian of $[\boldsymbol{A}]^s$ is defined and linearised as:

$$[\boldsymbol{J}]^s = \lim_{\epsilon \to 0} \frac{[\boldsymbol{A}]\left(\{\boldsymbol{T}\}^s + \epsilon \{\boldsymbol{T}\}\right) - [\boldsymbol{A}]\left(\{\boldsymbol{T}\}^s\right)}{\epsilon} \approx [\boldsymbol{A}]^s \tag{4.9}$$

The temperature update $\Delta \{\boldsymbol{T}\}^s$ that drives $\{\boldsymbol{r}\}^s$ to zero is found by solving the following system of equations:

$$[\boldsymbol{J}]^s \Delta \{\boldsymbol{T}\}^s = - \{\boldsymbol{r}\}^s \tag{4.10}$$

Eq. 4.10 is solved by using the Conjugate-Gradient (CG) method with the Jacobi preconditioner. CG aims to solve the system of equations by minimising the residual's energy norm.

Finally, the solution is updated:

$$\{\boldsymbol{T}\}^{s+1} = \{\boldsymbol{T}\}^s + \Delta \{\boldsymbol{T}\}^s \tag{4.11}$$

4.1.4 NGSolve model verification

To ensure the reliability and correctness of the NGSolve simulations performed in this work, the forward model implemented in NGSolve is verified against solutions generated using Code_Aster. This verification establishes confidence in the NGSolve implementation before it is used in DT construction and ML modelling.

A copper-tungsten HIVE sample, shown in Figure 3.3, is used for this verification. Its geometry is presented in Figure 4.2 and Table 4.1. The coolant parameters T_c, p_c, and v_c stay at 30°C, 1atm, and 10m/s, respectively. The uniform heat flux $q(t)$ applied to the tile's top surface is equal to $500 \times 10^4 W/m^2$.

TABLE 4.1: HIVE sample 1 BCs and materials; the labels are displayed in Figure 4.2.

Surface	**BC**
ABCD	Uniform heat flux $q(t)$
Pipe surface	Sample-coolant convection h_c
Set	**Location**
Γ_q	Surface ABCD
Γ_{edge}	Edge ABCD
$\Gamma_{corners}$	Nodes A, B, C, D
Volume	**Material**
Tile ABCDHEFG	Tungsten
Block IJKLPMNO	Copper
Pipe	Copper

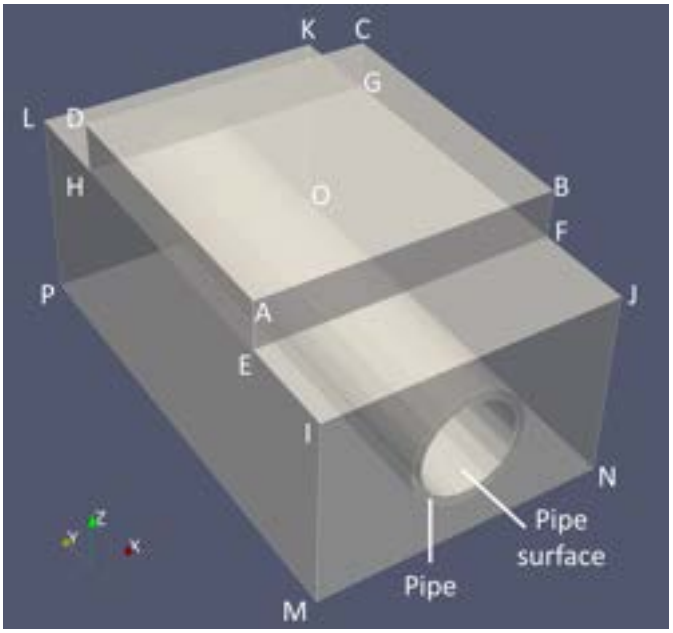


FIGURE 4.2: HIVE sample 1 diagram (Figure 3.3 and Table 4.1).

The steady-state solutions are generated using VL and NGSolve. Figures 4.3 and 4.4 show the relative errors between the two solutions. It can be seen that the errors remain within the order of 10^{-3}% with the highest errors located on the pipe's surface.

The transient solutions are generated in VL and NGSolve using the time step Δt equal to 0.5s with initial temperature distribution being 30°C everywhere. The solutions are generated up to a duration of 30s. Figures 4.5 and 4.6 show the relative and absolute errors for the two solutions. Similar to the steady-state solutions, it can be seen that the errors stay within the order of 10^{-3}% with the highest errors located on the pipe's surface.

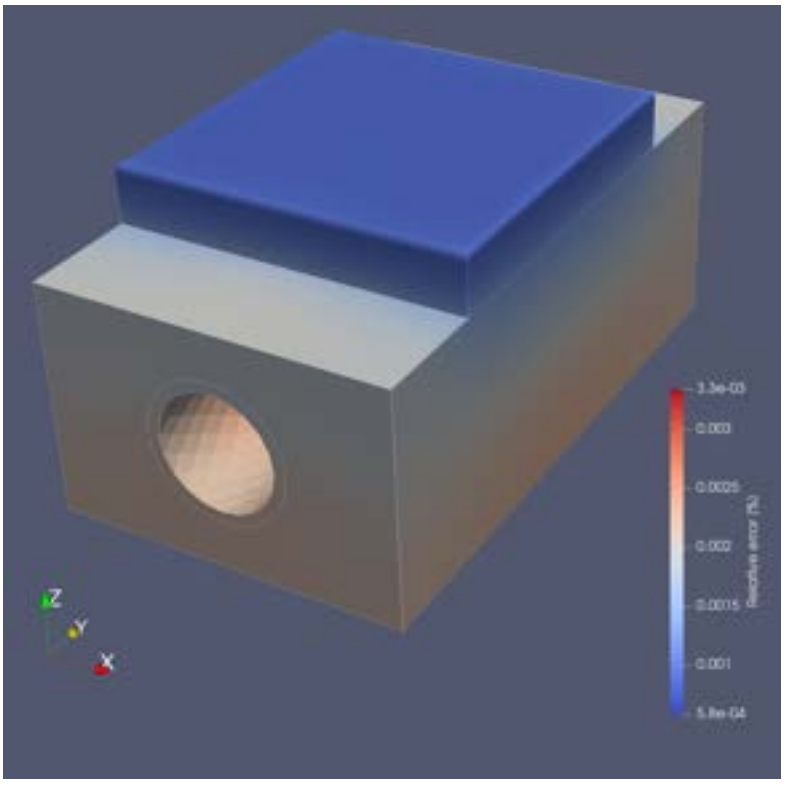


FIGURE 4.3: Relative error between VL and NGSolve steady-state solutions.

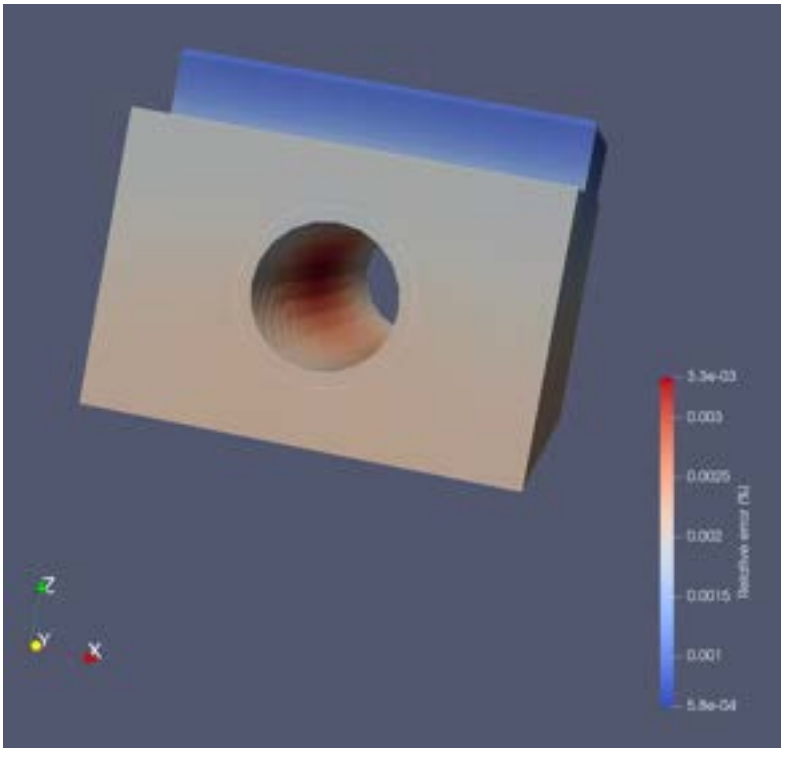


FIGURE 4.4: Relative error between VL and NGSolve steady-state solutions, showing the area with the highest error.

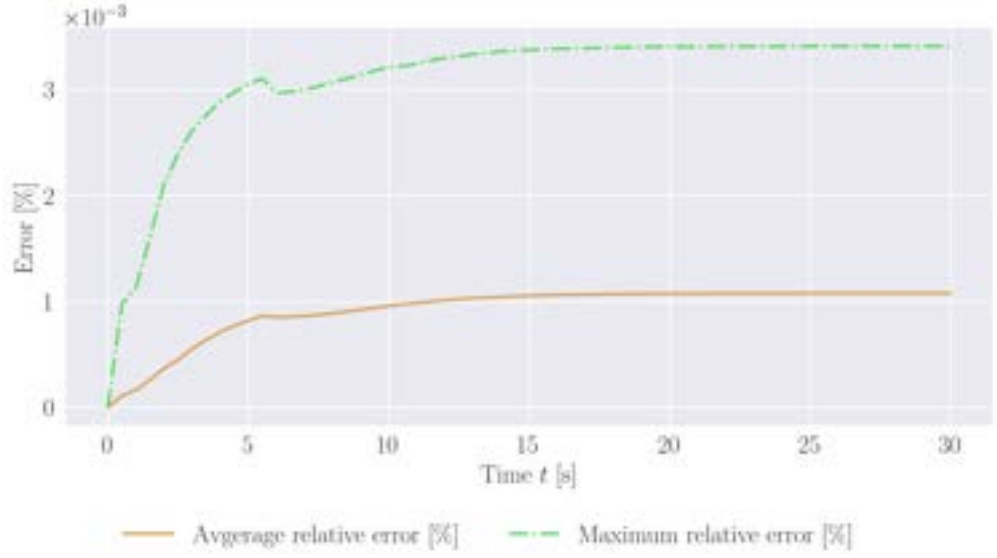


FIGURE 4.5: Relative error between VL and NGSolve transient solutions.

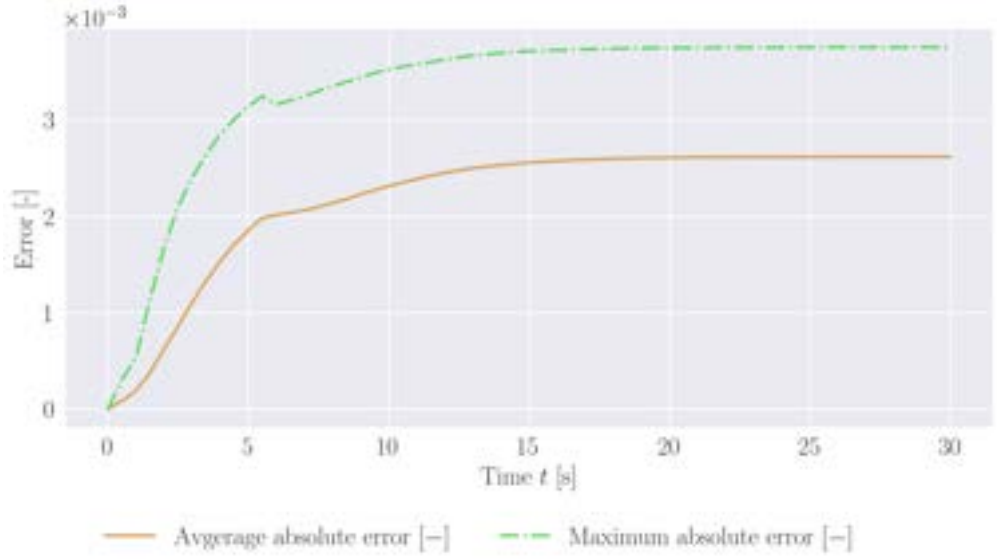


FIGURE 4.6: Absolute error between VL and NGSolve transient solutions.

Based on the comparison between the two solutions, the relative error remains consistently low. This level of agreement indicates excellent consistency between the implementations. Therefore, the solver is considered verified with respect to the reference solution.

4.2 Solution construction using computational data

4.2.1 Testing parameters

In this section, the solution construction process is tested by applying a linearly increasing heat flux, represented by Eq. 4.12, to the HIVE Sample 1 (Figure 4.2). The applied heat flux ramp in Eq. 4.12 is selected to remain within the range of experimentally relevant heat fluxes for the HIVE facility [19]. The maximum imposed value is consistent with the operational envelope and applicability limits.

$$q(t) = \frac{21 \times 10^5}{80} t = 26250t \tag{4.12}$$

Figure 4.7 shows 4, 8, 11, and 17 measurement locations used to generate results presented in this section.

Furthermore, the solution setup is tested with added noise in the temperature measurements to assess the robustness of the process under such conditions. The noise consists of random values drawn from a zero-mean Gaussian distribution where the standard deviation corresponds to the noise level (the measurement precision). The noise level, i.e. the standard deviation definition for the Gaussian distribution, is stated as a percentage of the true temperature value. The investigated noise levels are listed in Table 4.2.

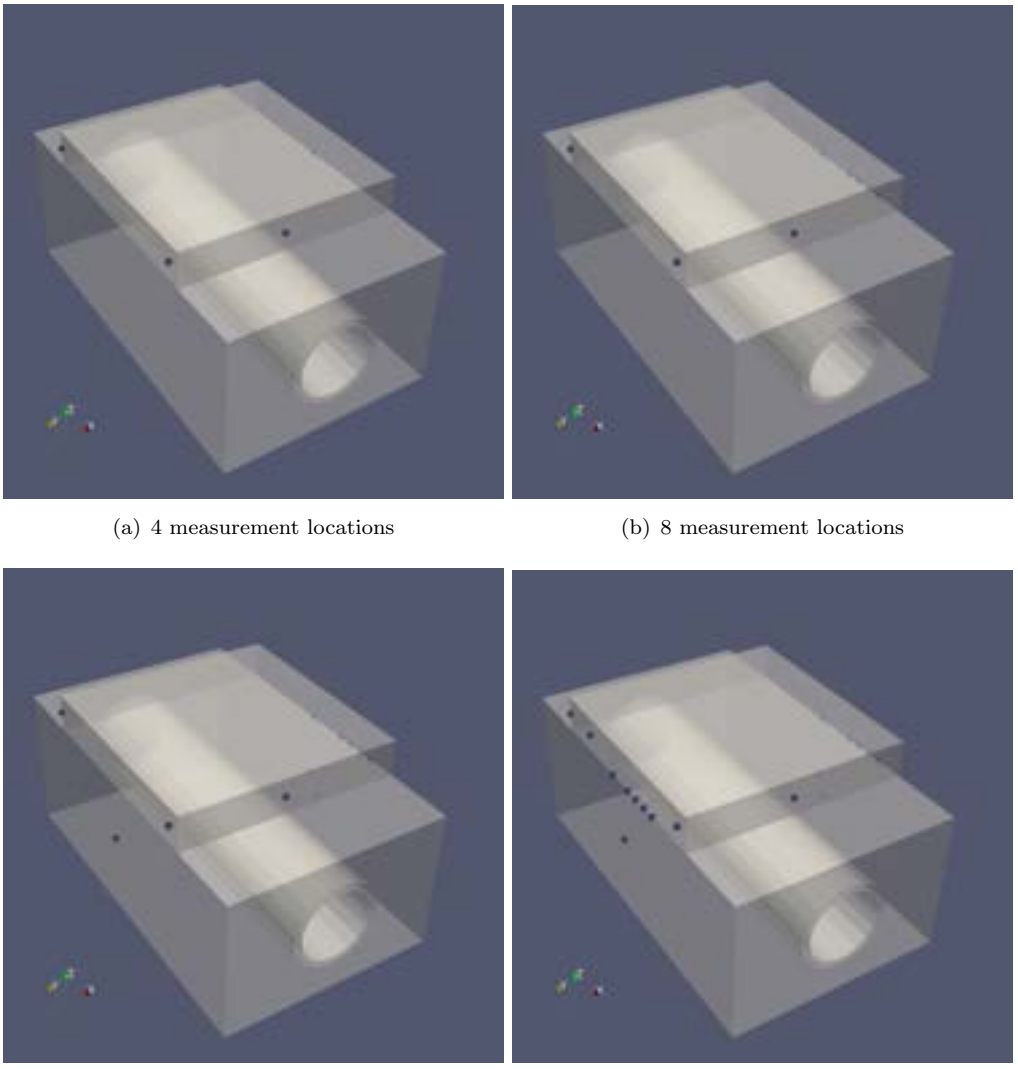
TABLE 4.2: Selected noise settings as a percentage of the true temperature values.

Noise setting No.	**Noise level (measurement precision) σ [%]**
1	0.0
2	0.2
3	1.0
4	$\sigma(T)$

The temperature-dependent overall precision characteristic $\sigma(T)$ of Type K TC measurements is illustrated in Figure 4.8, displaying values in both °C and as percentages of the actual temperature reading [3].

4.2.2 Solution construction results analysis

The initial temperature field is selected to be a uniform distribution with the temperature of 30°C, representing the initial steady state matching the coolant temperature. The



(a) 4 measurement locations (b) 8 measurement locations

(c) 11 measurement locations (d) 17 measurement locations

FIGURE 4.7: Measurement locations used for Sample 1.

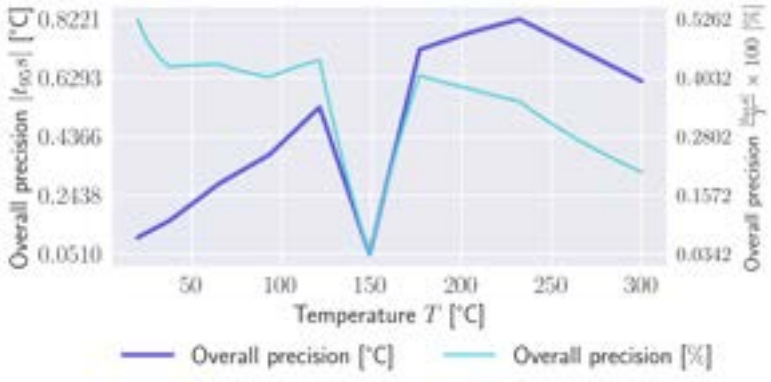


FIGURE 4.8: True temperature-dependent precision of a standard type K TC $\sigma(T)$ [3].

time step size used to generate the reference solution is equal to 0.4s, where as the time step size used to construct the solution from the measurements is equal to 2.8s.

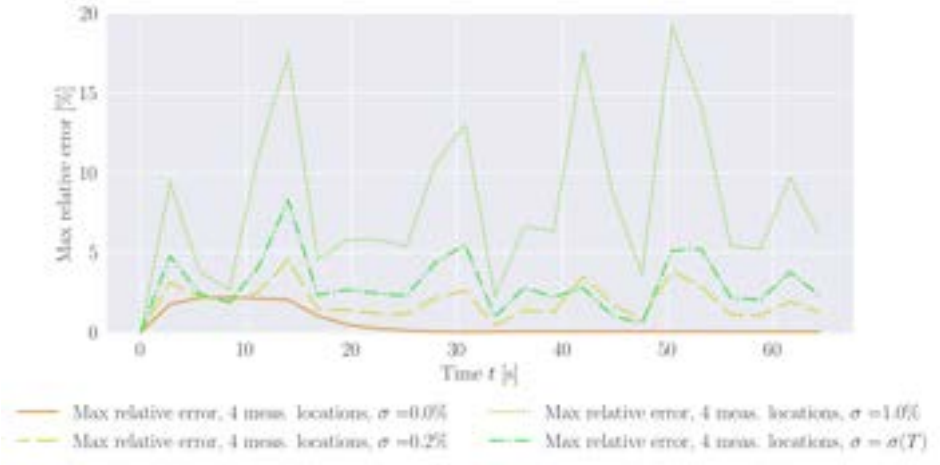
Table 4.3 presents the solution construction errors and time step runtimes achieved using FE-based solution construction process. The average and maximum errors presented in this table are calculated across the space and time.

TABLE 4.3: Solution construction errors and time step runtimes achieved using FE-based solution construction process.

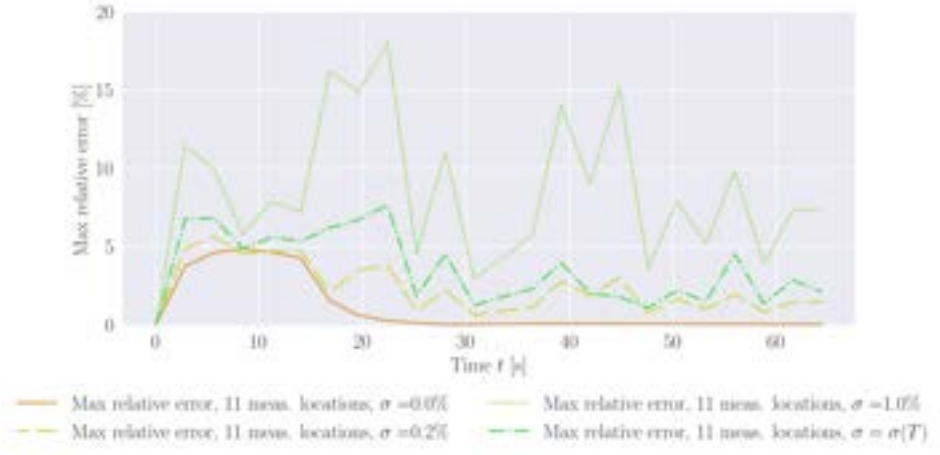
		Sol. rec. time		Sol. rec. errors	
No. of measurement locations	Noise level σ [%]	Avg. time step runtime [s] [a]	Max. time step runtime [s] [a]	Avg. relative error [%]	Max. relative error [%]
4	0.0	2.306e+00	3.038e+00	0.282	2.196
4	0.2	2.347e+00	2.495e+00	0.353	4.55
4	1.0	2.278e+00	2.433e+00	0.734	19.222
4	$\sigma(T)$	2.265e+00	2.376e+00	0.421	8.267
8	0.0	2.297e+00	2.515e+00	0.263	2.423
8	0.2	2.244e+00	2.379e+00	0.303	6.173
8	1.0	2.299e+00	2.411e+00	0.553	21.164
8	$\sigma(T)$	2.276e+00	2.463e+00	0.342	10.433
11	0.0	2.279e+00	2.419e+00	0.177	4.858
11	0.2	2.253e+00	2.441e+00	0.229	5.629
11	1.0	2.261e+00	2.387e+00	0.514	18.008
11	$\sigma(T)$	2.303e+00	2.615e+00	0.277	7.655
17	0.0	2.311e+00	2.452e+00	0.174	5.584
17	0.2	2.259e+00	2.429e+00	0.23	7.321
17	1.0	2.286e+00	2.404e+00	0.522	23.951
17	$\sigma(T)$	2.277e+00	2.432e+00	0.283	9.316

[a] Times are given for AMD Ryzen 7 5800X 8-Core Central Processing Unit (CPU).

As anticipated, for a fixed number of measurement locations, the relative construction errors rise with increasing noise levels. This observation is further supported by Figures 4.9 and 4.10 showing the dependency of maximum and average relative error on time. In these figures, the average relative error is calculated at every time instance by comparing the constructed temperature distribution against the reference forward solution. The errors increase in the beginning of the simulation reaching the maximum at the time instance of 8.4s. This can be attributed to the initial transition of the temperature field from the uniform distribution of 30°C to a profile shaped by the applied heat flux. Due to the rate of heat flux increase, this initial transition is characterized by rapid changes in the temperature distribution over both space and time. Consequently, the



(a) 4 measurement locations

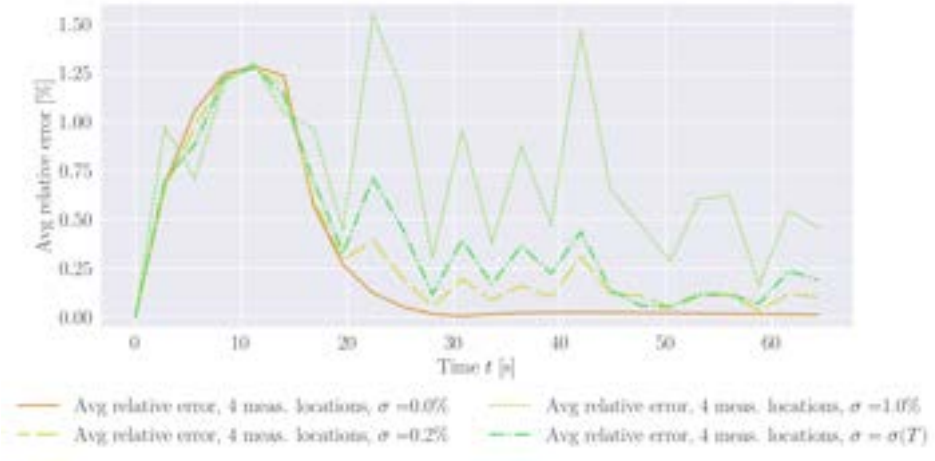


(b) 11 measurement locations

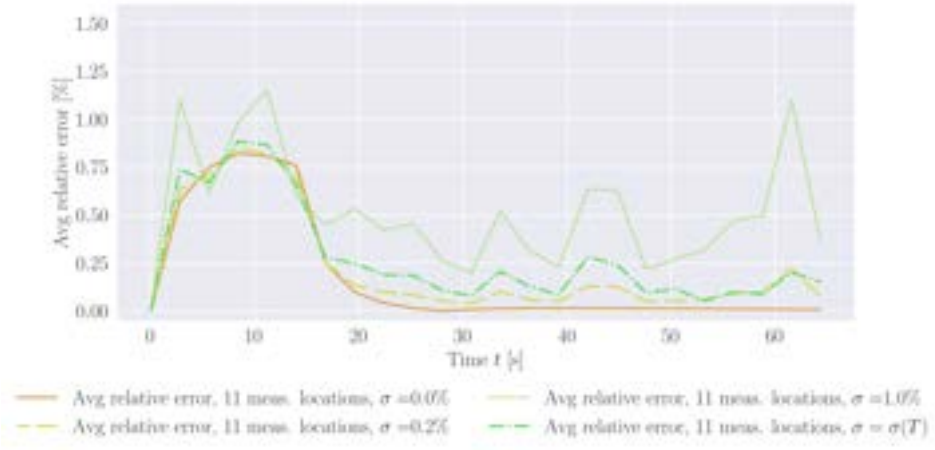
FIGURE 4.9: The progress of maximum relative errors with time for 4 and 11 measurement locations (Figure 4.7).

larger time step chosen for the solution construction limits its responsiveness, leading to the initial spike in errors. However, the errors decrease after 20s.

The noise induces an increasingly abrupt fluctuations in the maximum and average relative errors. This suggests that additional measurement points are required to mitigate the effect. Indeed, as the number of measurement locations grows, the average relative construction error generally decreases (Table 4.3 and Figure 4.10), initial temperature transition period before 20s with the highest errors. In spite of this, the reduction in maximum relative error does not show a direct correlation with the number of measurement locations. As the time progresses, the maximum relative errors tend to oscillate around the same value or even increase for some explored measurement locations with the same noise levels (Figure 4.9).



(a) 4 measurement locations

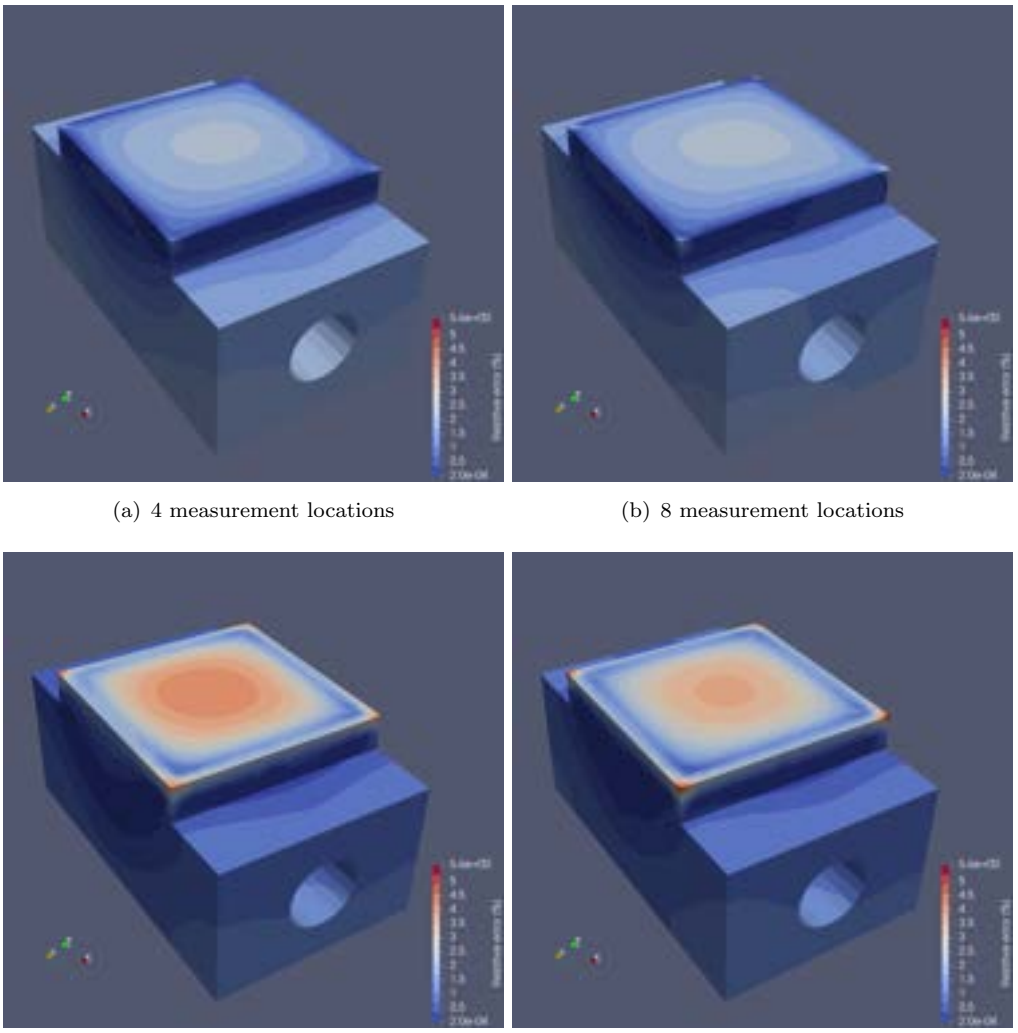


(b) 11 measurement locations

FIGURE 4.10: The progress of average relative errors with time for 4 and 11 measurement locations (Figure 4.7).

.

The analyses of the error distributions in space provides insight into this phenomenon. Figures 4.11 and 4.12 show the relative error distributions at two time instances, 8.4s and 50.4s. 8.4s is time instance where maximum spatial relative errors are observed (Figure 4.9, lines corresponding to the zero-noise scenario), while 50.4s is an example of the time instance where minimum relative errors are observed Figure 4.9, lines corresponding to the zero-noise scenario). The common tendencies can be noted between these two figures. When only 4 or 8 measurement locations are used, the error distribution is relatively uniform and concentrated at low values, with most of the domain showing small errors. However, as the number of measurement locations increases to 11 and 17, the error patterns change noticeably. In these cases, regions of higher relative error appear near the top surface of the tile, particularly at corners, while the errors within



(a) 4 measurement locations

(b) 8 measurement locations

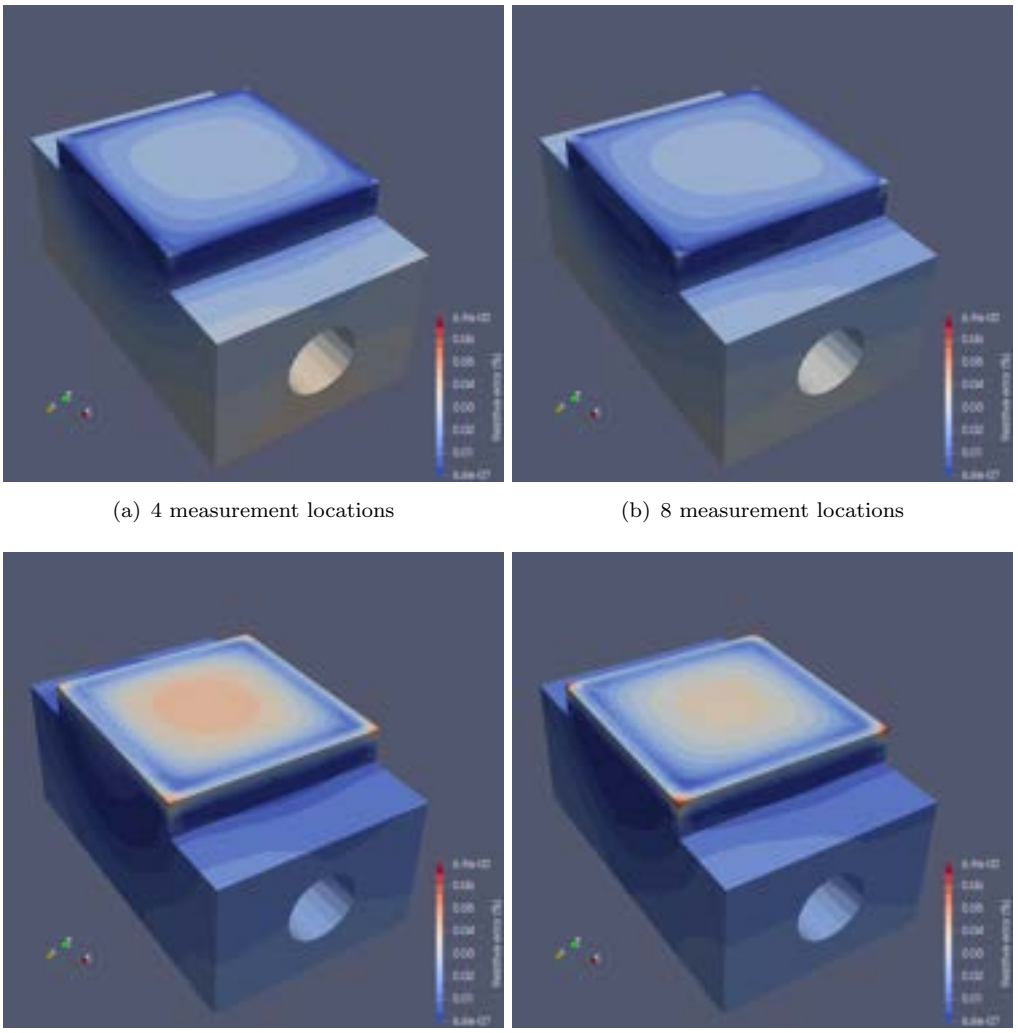
(c) 11 measurement locations

(d) 17 measurement locations

FIGURE 4.11: Relative error distributions for $t = 8.4s$; the measurement locations are shown in Figure 4.7.

the block, pipe, and the majority of the tile decrease. While increasing measurement numbers generally reduces average errors, the spatial error distribution does not improve uniformly across the domain. This suggests that although the average construction error decreases with additional measurement points, localised regions of higher error may still persist.

The temperature distribution constructed from the measurements can be used to calculate the constructed heat flux (Eq. 3.23) and compare it with the reference applied heat flux. Figure 4.13 shows the dependency of the constructed heat flux and the corresponding relative heat flux error on time. The increased average relative errors during the aforementioned initial transition period cause the spike in the relative heat flux error at



(a) 4 measurement locations

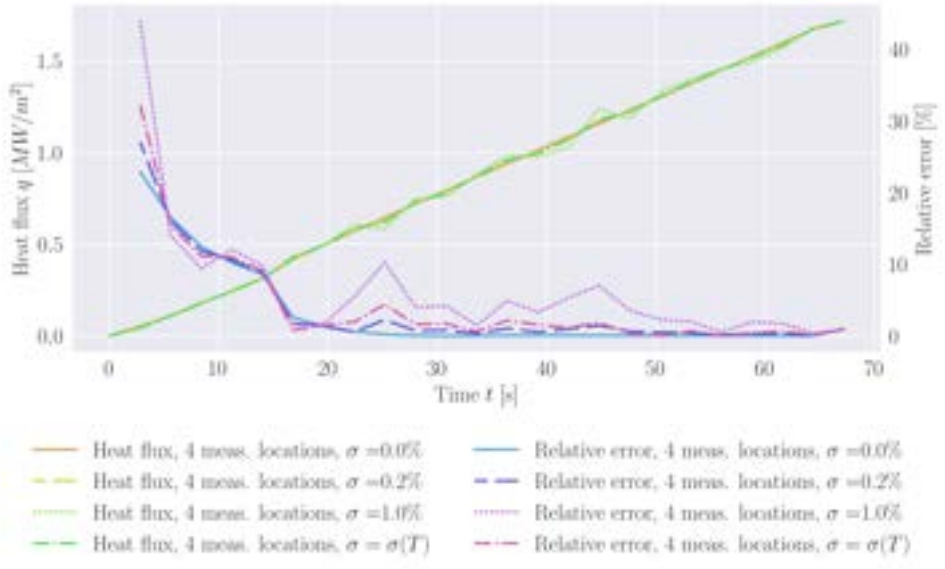
(b) 8 measurement locations

(c) 11 measurement locations

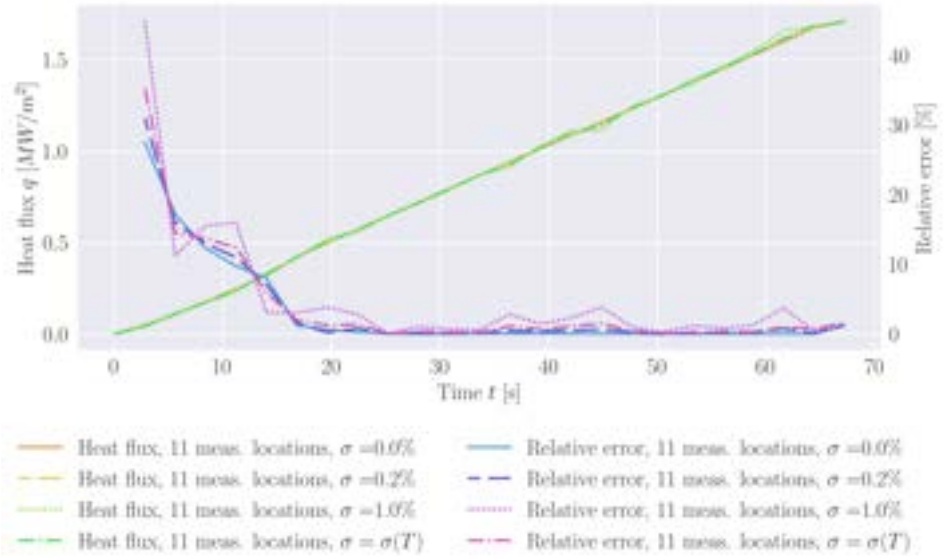
(d) 17 measurement locations

FIGURE 4.12: Relative error distributions for $t = 50.4s$; the measurement locations are shown in Figure 4.7.

the beginning of the simulation. However, it rapidly declines as the time approaches 20s. Figure 4.14 shows the average and maximum relative heat flux error recorded after 20s. It readily illustrates the benefit more measurement locations and hence lower average relative solution construction errors can provide, particularly for higher noise scenarios. Both average and maximum relative heat flux errors tend to decrease as the number of measurements increase. This trend can also be observed in Figure 4.13, where the osculation amplitude of the heat flux error markedly decrease for the higher number of measurement locations.



(a) 4 measurement locations

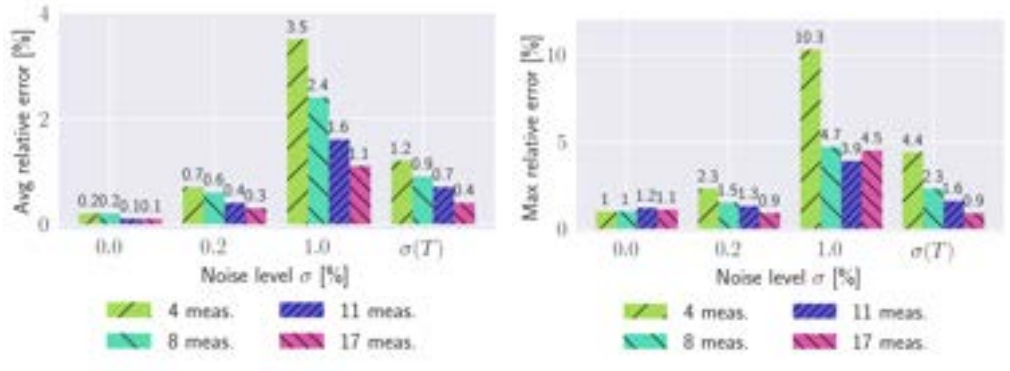


(b) 11 measurement locations

FIGURE 4.13: The progress of constructed heat flux with time for 4 and 11 measurement locations (Figure 4.7).

4.3 Thermal conductivity construction using computational data

Linear and nonlinear thermal conductivities are constructed from the sparse measurements using a FE-based approach. The geometry of Sample 1 is used as an example.



(a) Average relative heat flux error (b) Maximum relative heat flux error

FIGURE 4.14: Relative heat flux errors after 20s.

4.3.1 Linear thermal conductivity construction

Two scenarios are examined. The initial scenario assumes a single-material sample with thermal conductivity k_1; 4 measurements are used for this case. The second scenario considers a two-material sample with thermal conductivity k_1 for the block and k_2 for the tile; 5 measurements are used for this case. The selection of 4 and 5 measurement locations aligns well with the standard quantity of TCs typically employed for HIVE samples. For the two-material configuration, an extra measurement location is added to capture temperature data specifically for the block. The applied heat flux q is established at 210000 $[W/m^2]$; coolant velocity v_c is fixed at 1m/s.

Figure 4.15 demonstrates parameter convergence behaviour. The iteration process terminates when Δk between consecutive iterations falls below 0.01%. Table 4.4 presents the outcomes, indicating successful construction of linear thermal conductivity and corresponding solutions.

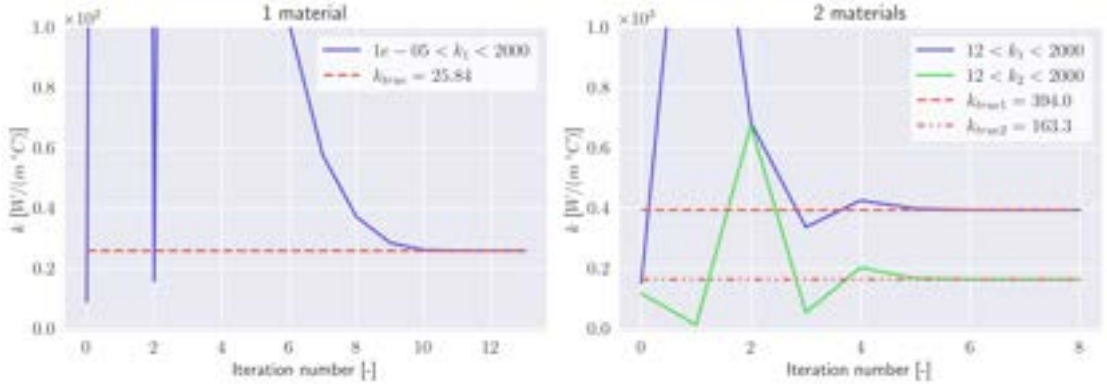


FIGURE 4.15: Convergence of linear thermal conductivities.

TABLE 4.4: Linear thermal conductivity construction results.

Case	k_1 error [%]	k_2 error [%]	Avg. $\{T\}$ error [%]	Max. $\{T\}$ error [%]	Number of iterations	Runtime [s]
1 material	2.13E-03	-	9.7E-05	4.2E-04	13	7.115
2 materials	2.39E-06	3.33E-05	2.3E-06	4.0E-05	8	4.831

4.3.2 Nonlinear thermal conductivity construction

4.3.2.1 Testing parameters

More detailed analysis is performed for the nonlinear thermal conductivity construction. This analysis includes exploring the influence of the measurement location number (Figure 4.7) and noise level (Table 4.2) on the construction process performance. The goal is to construct the piecewise linear thermal conductivity depicted in Figure 4.16. Table 4.5 specifies the four linear segments of the temperature-dependent relationship to be established, it additionally presents the temperature boundaries and BCs employed for constructing each linear segment. The parameters a and b correspond to the mathematical definition of each linear segment as demonstrated in Eq.3.38 and Figure3.10.

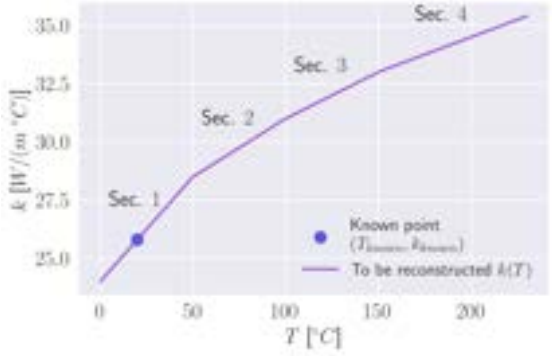


FIGURE 4.16: Piecewise linear thermal conductivity consisting of four sections and with T_{known} and k_{known} equal to 20.0°C and 25.8 $W/(m°C)$, respectively.

TABLE 4.5: Four linear sections of the temperature dependency to be constructed, the temperature limits, and BCs used to construct each of the four linear sections (Figure 4.16).

Section	a $[W/m°C^2]$	b $[W/m°C]$	Limit 1	Limit 2	v_c $[m/s]$	q $[W/m^2]$
1	0.09	24.0	$T_{r1} = 0°C$	$T_{r2} = 50°C$	1.0	33000
2	0.05	26.0	$T_{r2} = 50°C$	$T_{r3} = 100°C$	2.0	130000
3	0.04	27.0	$T_{r3} = 100°C$	$T_{r4} = 150°C$	2.0	240000
4	0.03	28.5	$T_{r4} = 150°C$	$T_{r5} = 230°C$	2.0	422000

4.3.2.2 Conductivity construction results analysis

Table 4.6 lists the results, from which it can be seen that the non-linear thermal conductivity and the corresponding solutions are successfully constructed for a certain number of measurement locations. This table includes the relative errors between the constructed and reference thermal conductivities k, the average and maximum values for 4 linear sections are displayed. Moreover, this table shows the relative errors between the constructed and reference temperature distributions, the average and maximum values for four linear sections (Figure 4.16) are also shown.

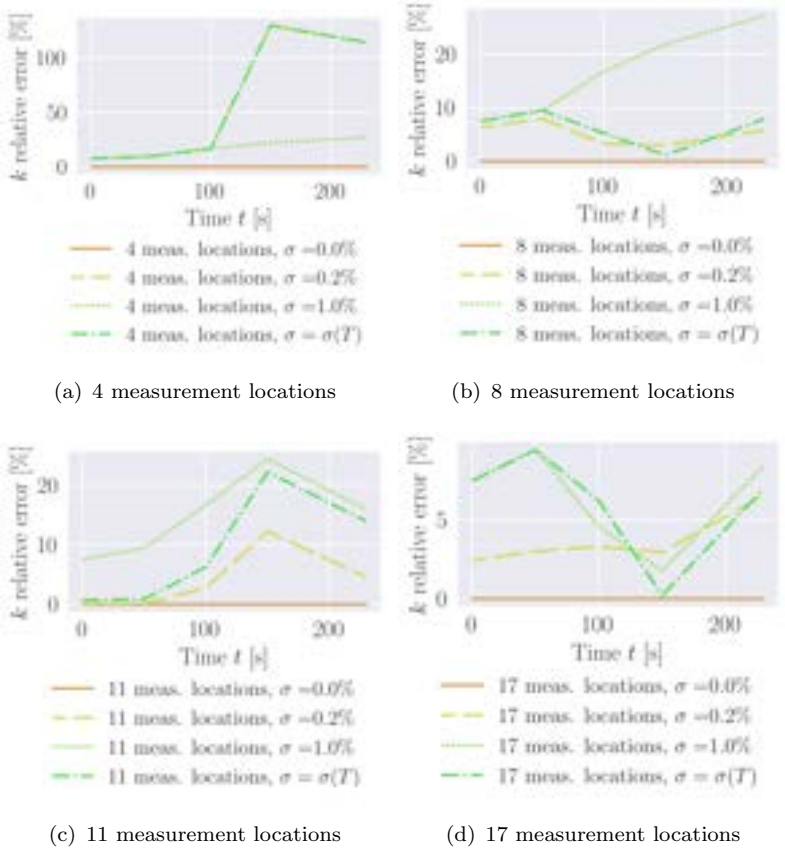
TABLE 4.6: Nonlinear thermal conductivity construction results (Figure 4.16).

		k relative errors		$\{\boldsymbol{T}\}$ relative errors	
No. of measurement locations	Noise level σ [%]	Avg. k error [%]	Max. k error [%]	Avg. $\{\boldsymbol{T}\}$ error [%]	Max. $\{\boldsymbol{T}\}$ error [%]
4	0.0	0.006	0.016	0.0001	0.0002
4	0.2	55.514	129.697	12.109	28.021
4	1.0	16.537	27.119	15.837	181.358
4	$\sigma(T)$	55.514	129.697	15.511	28.672
8	0.0	4.829e-05	9.568e-05	1.29e-06	9.92e-06
8	0.2	5.252	7.964	0.51	1.122
8	1.0	16.537	27.119	4.027	16.329
8	$\sigma(T)$	6.241	9.474	0.953	2.986
11	0.0	3.751e-05	1.458e-04	3.5e-07	9.73e-06
11	0.2	3.976	12.241	0.107	1.04
11	1.0	14.822	24.45	1.796	4.375
11	$\sigma(T)$	8.807	22.251	2.237	34.25
17	0.0	6.8e-06	2.177e-05	3.4e-07	6.14e-06
17	0.2	3.646	6.545	0.195	2.614
17	1.0	6.348	9.474	1.067	8.411
17	$\sigma(T)$	6.063	9.444	0.839	3.289

As expected, when the number of measurement locations is fixed, the relative construction errors generally increase with higher noise levels. This trend is further illustrated in Figure 4.17, which display the time-dependent behaviour of the relative thermal conductivity errors. This is also the case for the stand-alone solution construction previously discussed in Section 4.2.

The measurement noise is modelled as zero-mean Gaussian noise, where the standard deviation represents the measurement precision. For the case $\sigma(T)$, the noise level depends on the local temperature and therefore varies in time and space. Importantly, $\sigma(T)$ does

not correspond to the highest noise level at the highest temperatures, nor does it uniformly exceed the constant noise cases. As a result, the temperature-dependent noise does not necessarily produce the largest relative error at all times. The reconstruction error is governed by the combined effects of the noise realisation and the number of measurement locations. Consequently, depending on the measurement configuration and time interval, $\sigma(T)$ may lead to either higher or lower errors compared to constant noise levels, as observed in Figure 4.17.



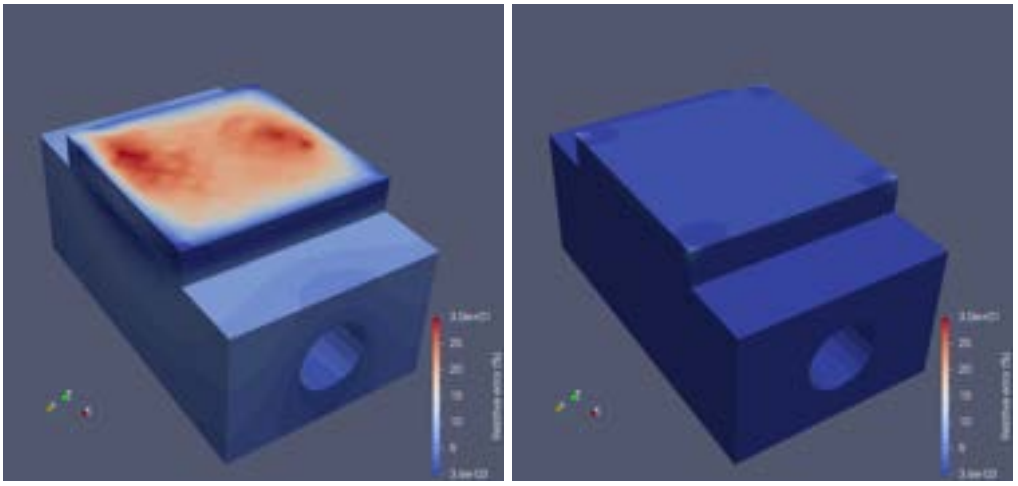
(a) 4 measurement locations (b) 8 measurement locations

(c) 11 measurement locations (d) 17 measurement locations

FIGURE 4.17: The progress of relative thermal conductivity error with time for various measurement locations (Figure 4.7).

All cases with zero noise achieve excellent results, with errors of the order of 10^{-3} or less. However, the performance deteriorates rapidly for a certain number of measurements once noise is introduced. The cases with 4 measurement locations fail to adequately deal with noise even at 0.2%, with errors oftentimes exceeding 100%, which render the constructed thermal conductivity and solution meaningless. On the other hand, the performance of the construction process noticeably improves when more measurement locations are introduced. The best results are achieved for 17 measurement locations, with errors staying below 10% for all considered noise levels. These observations are

supported by the spatial distributions of relative $\{\boldsymbol{T}\}$ error, which are displayed in Figure 4.18. These error distributions bear some similarity to the distributions presented in Section 4.2 (Figures 4.11 and 4.12). Particularly, as the number of measurement locations increase the average relative $\{\boldsymbol{T}\}$ errors decrease with the maximum relative errors starting to concentrate around the corners of the tile's top surface.



(a) 4 measurement locations (b) 17 measurement locations

FIGURE 4.18: The spatial distribution of relative $\{\boldsymbol{T}\}$ error for 4 and 17 measurement locations (Figure 4.7) for the TC-specific noise level.

4.4 Solution construction using experimental data

FE-based solution construction is trialled using experimental data collected for the stainless-steel Sample 2 (Figure 3.3). This step is essential in the context of FE DT development, as it tests its ability to assimilate real measurement data and construct the underlying temperature field. By using experimental inputs rather than purely simulated ones, the approach is validated under more realistic conditions where various measurement limitations are present.

The experimental steady-state temperature values were collected at six locations shown in Figure 4.20a and Table 4.8. The collected experimental values are listed in Table 4.7.

TABLE 4.7: Average and standard deviation of the experimental temperature measurements.

TC number	**1**	**2**	**3**	**4**	**5**	**6**
Average $[^{\circ}C]$	233.9	173.4	163.7	163.7	236.4	161.8
Standard deviation $[^{\circ}C]$	1.259	0.157	1.466	0.202	1.238	0.158

TABLE 4.8: Locations of the TCs used to collect the experimental measurements.

TC number	**1**	**2**	**3**	**4**	**5**	**6**
x [m]	0.0116	0.0138	0.0110	-0.0105	-0.018	-0.018
y [m]	-0.0245	-0.0245	0.0245	0.0245	-0.0006	-0.004
z [m]	0.0194	0.0013	0.0031	-0.005	0.0164	-0.0085

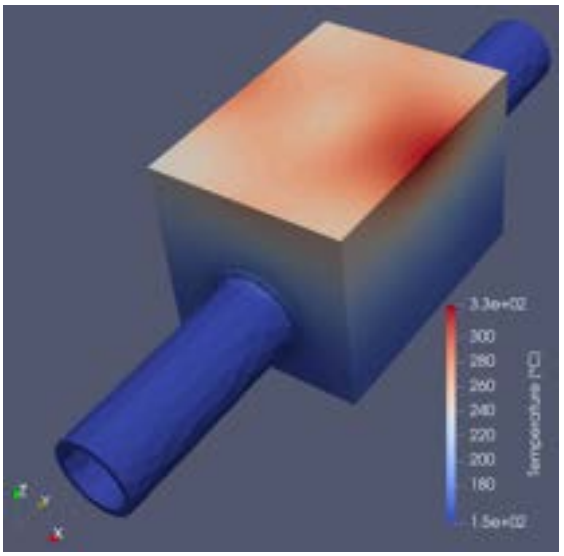
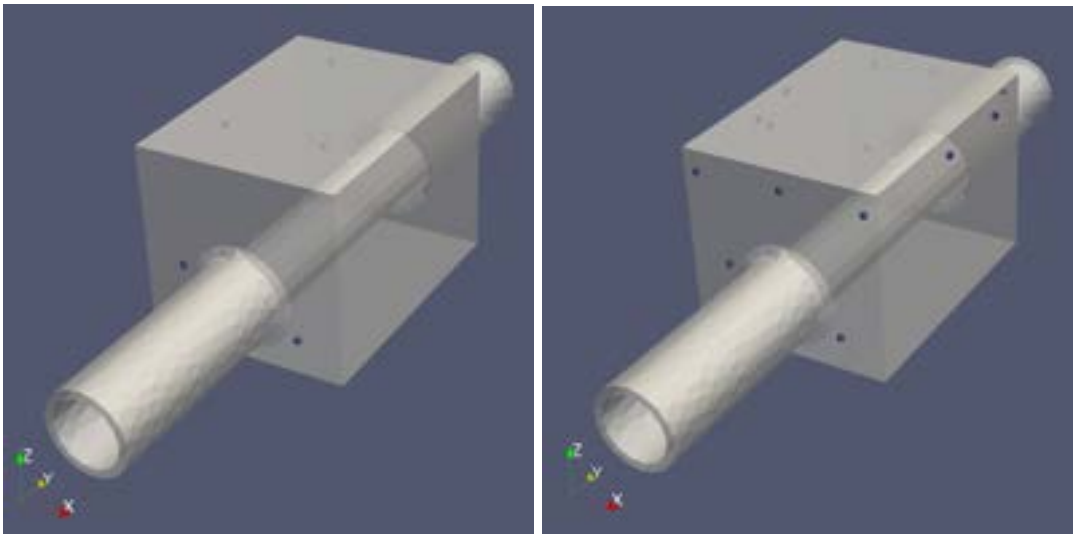


FIGURE 4.19: Reference solution generated using COMSOL [4]. This is the solution used to augment the experimental data and consequently calculate some relative errors.

TABLE 4.9: Solution construction errors for Sample 2 (Figure 3.3).

No. of measurement locations	**Avg. relative error [%]**	**Max. relative error [%]**	**Max. temperature relative error [%]**
6	7.213	32.634	15.991
9	1.887	16.480	1.287
12	1.436	7.967	1.149
15	1.005	6.252	1.073

The experimental data is augmented with the thermal solution generated using coil-induced EM heating in COMSOL Multi-physics software [4]. Tetrahedral mesh is used within COMSOL, therefore the modification of the regularisation component of a loss function is required in Eq 3.29. The regularisation used in Eq 3.29 is suitable for a surface of a hexahedral mesh as generally on such a surface each node is connected to the constant number of nodes; consequently, an averaging function could be used to encourage smoothness. However, in the case of a surface of a tetrahedral mesh, the number of nodes the specific node is connected with could vary, which makes the regularisation function in Eq 3.29 less effective.



(a) 6 measurement locations. (b) 15 measurement locations.

FIGURE 4.20: Measurement locations used for Sample 2 (Figure 3.3).

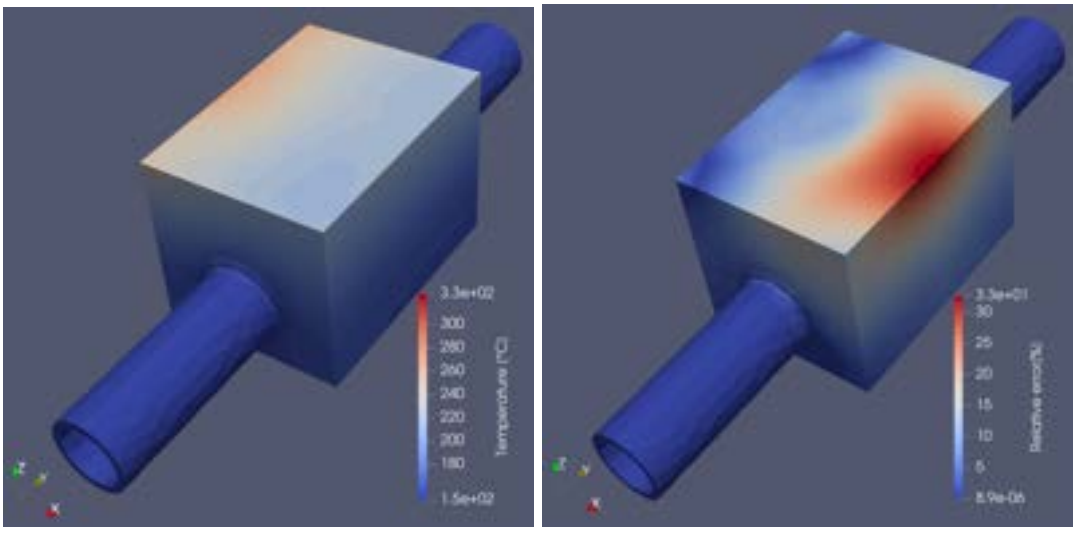
In order to enable the method to work with a tetrahedral mesh, a more flexible regularisation component is formulated as follows:

$$\{\boldsymbol{L}\}_{REG} = \{\boldsymbol{f}\}_q - \mathcal{S}(\{\boldsymbol{x}\}_q) \tag{4.13}$$

where $\mathcal{S}$ is an spatial interpolation function constructed using $\{\boldsymbol{f}\}_q$ values, and $\{\boldsymbol{x}\}_q$ contains x, y, and z coordinates of $\{\boldsymbol{f}\}_q$. $\mathcal{S}$ is defined as Radial Basis Function (RBF) interpolation with linear kernel. The regularisation is ensured through penalising deviations from a smoothed surface interpolated using the values of $\{\boldsymbol{f}\}_q$ at the current iteration. The gradients for the Jacobian matrix are approximated by assuming that the values of $\mathcal{S}$ are constant for the current iteration.

The reference solution used to assess the solution construction results is displayed in Figure 4.19.

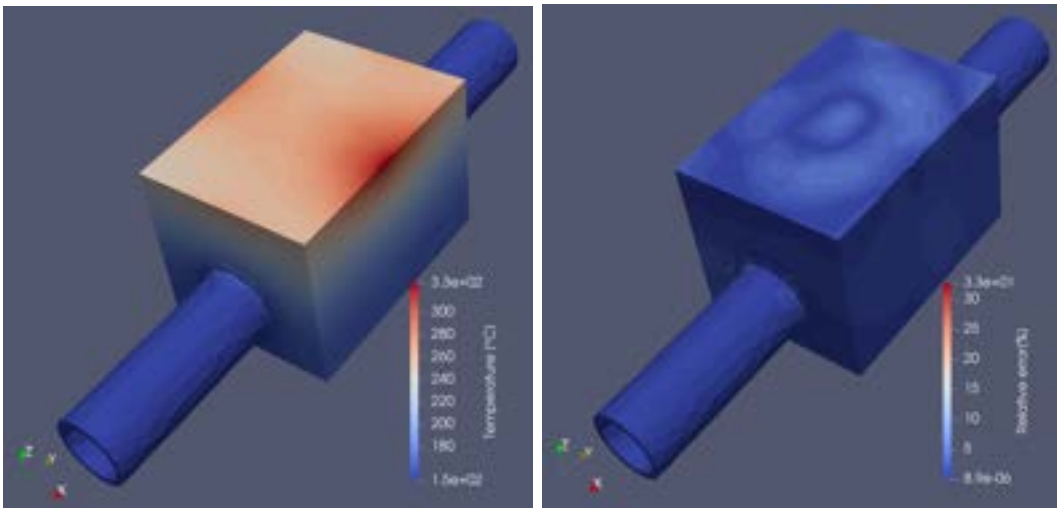
The TC measurement locations used during the experiment are shown in Figure 4.20(a), there are 6 locations. Initially, only these locations are used for the solution construction. Figure 4.21 shows the constructed temperature distributions together with the relative errors. It can be seen that 6 measurements are not sufficient to capture the complexity of the heat flux distribution produced by the coil. Particularly, the solution construction process struggles to estimate the temperature in the region where the maximum temperature is observed. This could be attributed to the lack of TCs located in



(a) Constructed temperature using 6 measurement locations.

(b) Relative error achieved 6 measurement locations.

FIGURE 4.21: Results achieved for 6 measurement locations used for Sample 2 (Figure 3.3), the maximum relative error is 32.634%.



(a) Constructed temperature using 15 measurement locations.

(b) Relative error achieved 6 measurement locations.

FIGURE 4.22: Results achieved for 15 measurement locations used for Sample 2 (Figure 3.3), the maximum relative error is 6.252%.

the vicinity of this region.

This issue is resolved by including some measurements sampled from the COMSOL solution and adding the temperature-dependent noise associated with typical TCs (Figure 4.8). The number of measurements are increased from 6 to 15 with Figure 4.20(b) displaying 15 measurement locations. Table 4.9 lists the relative solution construction errors including the errors for the maximum temperature within the domain. As expected

the errors decrease with the increase in the number of measurements. Figure 4.21 shows the constructed temperature distributions together with the relative errors for 15 measurement locations shown in Figure 4.20(b). For this case, the constructed temperature distribution bears significantly more similarity to the reference solution than the distribution achieved for the initial 6 measurements. And, indeed, considerable improvement could be observed in the relative errors with maximum relative error being around 6% as opposed to around 33% for 6 measurements.

4.5 Summary

This chapter presents a FE-based framework for temperature field and thermal conductivity construction in the context of DT for high heat flux experiments conducted at the HIVE facility. A detailed forward FE model is developed to represent the experimental setup, incorporating surface heating, temperature-dependent material properties, and a 1D coolant model. The coolant model captures forced convection and nucleate boiling regimes. The framework is implemented in NGSolve to enable efficient nonlinear matrix extraction, flexible BC handling, and seamless integration with the DT workflow. Verification against Code_Aster, which was used in the previous work, demonstrated excellent agreement for both steady-state and transient simulations, confirming the reliability of the implementation.

The FE-based solution construction approach is then evaluated using computational data under varying number of measurement locations and noise levels. The results show that accurate reconstruction of the temperature field is achievable in near real time, with average relative errors remaining low for moderate noise levels. While increasing the number of measurement locations consistently reduced average errors, localised regions of higher error persisted, particularly near the surface corners. The reconstructed temperature fields enabled reliable estimation of the applied heat flux, with accuracy improving markedly as additional measurements are introduced, especially under noisy conditions.

The framework is further extended to thermal conductivity construction. Both linear and nonlinear (i.e. temperature-dependent) conductivities are successfully reconstructed from sparse measurements. For noise-free data, excellent accuracy is achieved even with

a limited number of sensors. However, the nonlinear reconstruction proves sensitive to measurement noise when few measurements are available. Increasing the number of measurement locations significantly improved robustness, enabling accurate recovery of both conductivity profiles and temperature fields across all considered noise levels.

Finally, the FE-based solution construction is applied to experimental data from a stainless-steel HIVE sample. This demonstrates the framework's ability to assimilate experimental measurements and construct physically consistent temperature fields under realistic conditions. The results confirm that sparse experimental measurements alone may be insufficient to capture complex heating patterns, but that solution quality improves substantially as additional measurement information is incorporated.

Overall, the chapter establishes the FE-based solution and material property construction as a robust and flexible foundation for FE-driven DT, suitable for both computational and experimental data assimilation.

Chapter 5

Temperature monitoring and control

The methodology behind FE DT and PD-ML DT control loops are described in Sections 3.3 and 3.4, respectively. In this chapter the assembled FE DT loop with and without ROM and PD-ML DT loop are tuned and tested, and the results are discussed. The forward FE model is described in Section 4.1 in the previous chapter, and it is used again in this chapter as a physical experiment imitation. The solution construction discussed in the previous chapter form an essential part of FE DT.

5.1 Finite Element-based Digital Twinning control loop tuning

In this work, the PID controller is adjusted using two traditional techniques: the ZN and AH methods [81, 82]. This sub-section outlines the steps taken during the tuning process. The controller operates with the same time step as the solution procedure, specifically $\Delta t = 2.8s$. The choice of this time step is dictated by the requirements of near real-time solution construction within the DT framework. The FE-based solution construction should operate fast enough to provide timely temperature estimates for control purposes. Smaller time steps would compromise real-time feasibility, as the solution construction code would not be able to deliver results at the required rate. As demonstrated previously, the solution construction retains sufficient accuracy at this

time step. Since the PID controller relies directly on the constructed solution to compute control actions, it is therefore designed to operate at the same temporal resolution. Moreover, the PID gains are explicitly tuned using this time step, ensuring consistent and stable closed-loop behaviour despite the relatively coarse discretisation.

Although recent advancements have introduced RL-based methods for PID tuning [98, 99], these approaches often involve long training periods and require fine-tuning of the RL model itself, which may introduce unnecessary complexity.

5.1.1 Ziegler–Nichols method

The ZN approach involves a two-step procedure [81, 82]. In the first step, the integral and derivative components of the controller are disabled by assigning τ_I to ∞ and τ_D to 0. The coolant flow velocity is maintained at its nominal value $\overline{v_c}$ equal to 5 m/s. Under a constant heat flux with a linear ramp-up period, specified in Eq 5.2, the system is allowed to reach a steady state, where the peak temperature equals T_{ss_sp}. Following this, a brief disturbance is introduced by altering the set point to:

$$T_{dist\_sp} = 1.1 T_{ss\_sp} \tag{5.1}$$

$$q(t) = \begin{cases} \frac{21\times10^5}{80} t & \text{if } t \leq 80 \\ 21 \times 10^5 & \text{if } t > 80 \end{cases} \tag{5.2}$$

The controller gain's magnitude, $|K_C|$, previously discussed in Sub-section 3.3.4, is gradually increased until the system exhibits stable, repeating oscillations, referred to as continuous cycling. This behaviour is illustrated in Figure 5.1, which shows that such oscillations begin when K_C is equal to -0.430. Therefore, the ultimate gain K_{cu} is determined to be -0.258, and the ultimate period P_u, representing the duration of one full oscillation cycle, is found to be 16.8s. The obtained values of K_{cu} and P_u are then used to determine K_c, τ_I, and τ_D, as presented in Table 5.1.

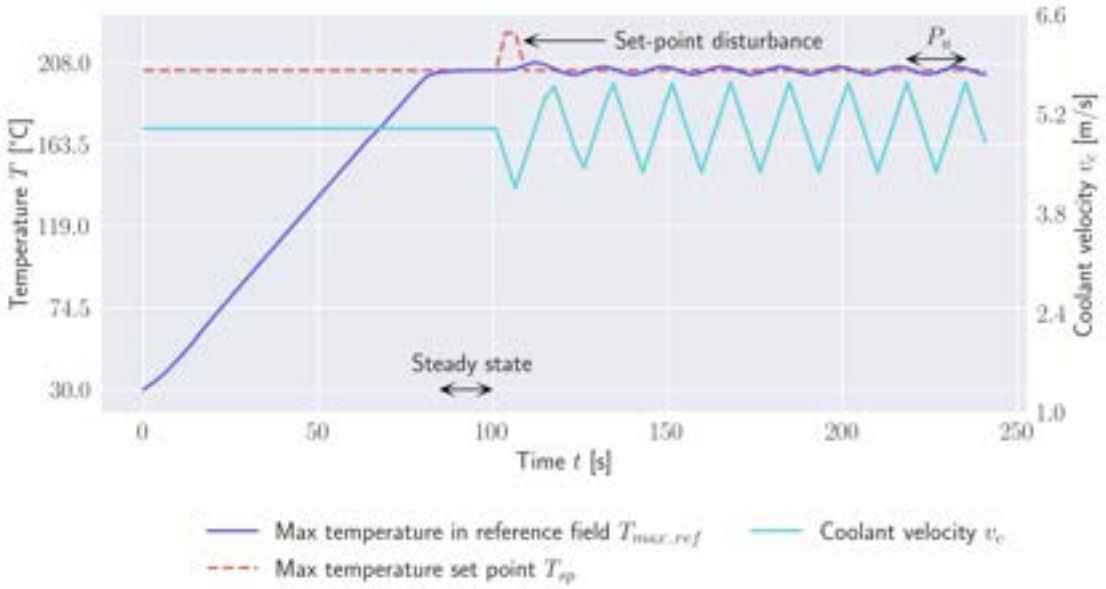


FIGURE 5.1: Continuous oscillations achieved for $K_c = -0.430$, $\tau_I = \infty$, and $\tau_D = 0$. It is a response to the set point disturbance introduced by Eq. 5.1. The steady state is reached by applying heat flux in the form of a constant with linear ramp-up period (Eq 5.2).

TABLE 5.1: PID controller parameter values calibrated using ZN and AH approaches.

Standard					
Method	K_{cu}	P_u	$K_c = 0.600K_{cu}$	$\tau_I = 0.500P_u$	$\tau_D = 0.125P_u$
ZN	-0.430	16.800	-0.258	8.400	2.100
AH	-0.444	11.200	-0.266	5.600	1.400
Modified τ_I					
Method	K_{cu}	P_u	$K_c = 0.600K_{cu}$	$\tau_I = 3.000P_u$	$\tau_D = 0.125P_u$
ZN	-0.430	16.800	-0.258	50.400	2.100
AH	-0.444	11.200	-0.266	33.600	1.400

5.1.2 Åström–Hägglund method

Another widely used method for tuning a PID controller is the AH technique [81, 82]. In this method, the PID controller is temporarily replaced with a relay (on–off) controller, where the output is constrained to two fixed values: a minimum and a maximum. In the present work, the relay controller parameters are set to $v_{c_min} = 4.85$ m/s and $v_{c_max} = 5.15$ m/s. The same set–point disturbance applied in the previously described ZN method is used here as well. This relay configuration induces regular oscillations in the controlled variable T_{max}, as illustrated in Figure 5.2. The ultimate gain K_{cu} can

then be estimated as:

$$|K_{cu}| = \frac{4d}{\pi a} \tag{5.3}$$

d denotes the relay amplitude, which in this case is 0.15, and a represents the amplitude of the controlled variable, measured as 0.43. The value of P_u corresponds to the oscillation period of the controlled variable, found to be 11.2 s.

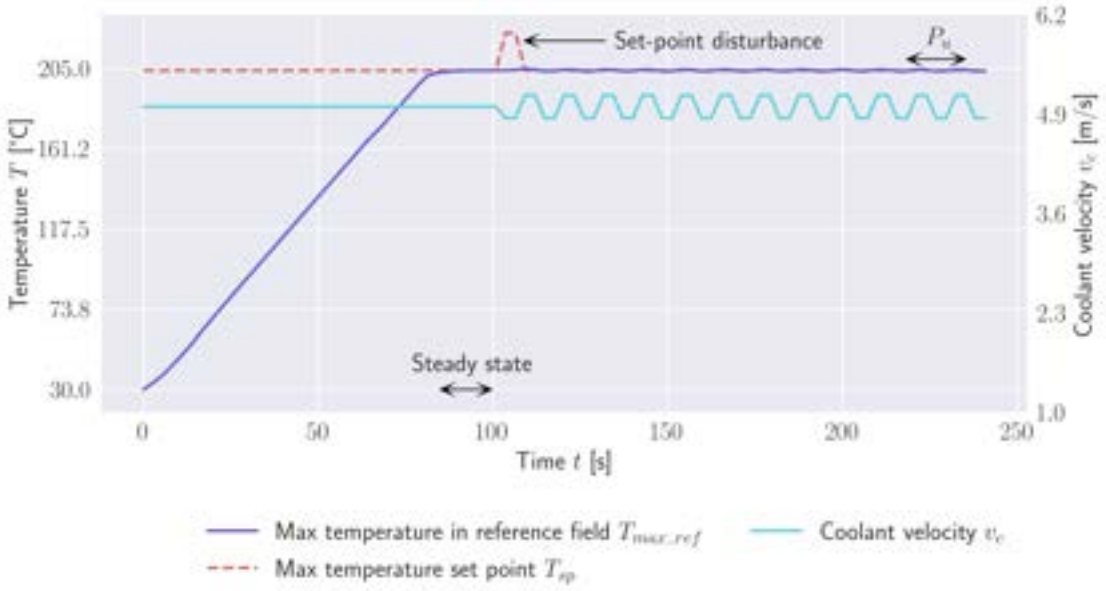


FIGURE 5.2: Continuous oscillations achieved using relay (on–off) controller with v_{c_min} equal to 4.85 m/s and v_{c_max} equal to 5.15 m/s, as a response to the set point disturbance represented by Eq. 5.1. The applied heat flux defined by Eq. 5.2.

5.1.3 Modified tuned Proportional–Integral–Derivative parameters

Figures 5.3 and 5.4 present a comparison of system responses to a constant heat flux $q(t)$ (Eq. 5.2). The corresponding performance metrics are summarised in Table 5.2, where the overshoot quantifies how much the peak temperature T_{max} exceeds the set point T_{sp}. In this work, T_{sp} is taken as 200°C for all scenarios, except in the ZN and AH tuning cases, where it is set to the steady–state maximum temperature. 200°C is a reasonable temperature limit for the HIVE maximum heat fluxes and sample materials, as it ensures the system operates within a safe thermal range. This limit accounts for the thermal gradients present between various sections of a sample, maintaining internal stresses within safe margins and preventing degradation of the multi-material assembly.

For ZN tuning under constant $q(t)$, the overshoot is close to 7%, whereas for AH tuning it is close to 4%. Furthermore, the values of τ_I and τ_D for both cases are insufficient to fully suppress oscillations induced by the proportional term. To address this, the standard ZN relations are adjusted by increasing τ_I, yielding a more conservative controller response. The updated PID parameters are listed in Table 5.1. The improved responses, shown in Figures 5.3 and 5.4, confirm that the modification eliminates or reduces both overshoot and oscillations (Table 5.2). The revised PID formulation triggers coolant velocity adjustments earlier, which is advantageous given the imposed acceleration limit on the coolant; this restriction prevents the velocity from changing as rapidly as required by the standard ZN or AH tuned parameters. The modified ZN tuned parameters yield a more conservative response with lower overshoot and no coolant velocity oscillations. Hence, they are adopted for generating the remaining results.

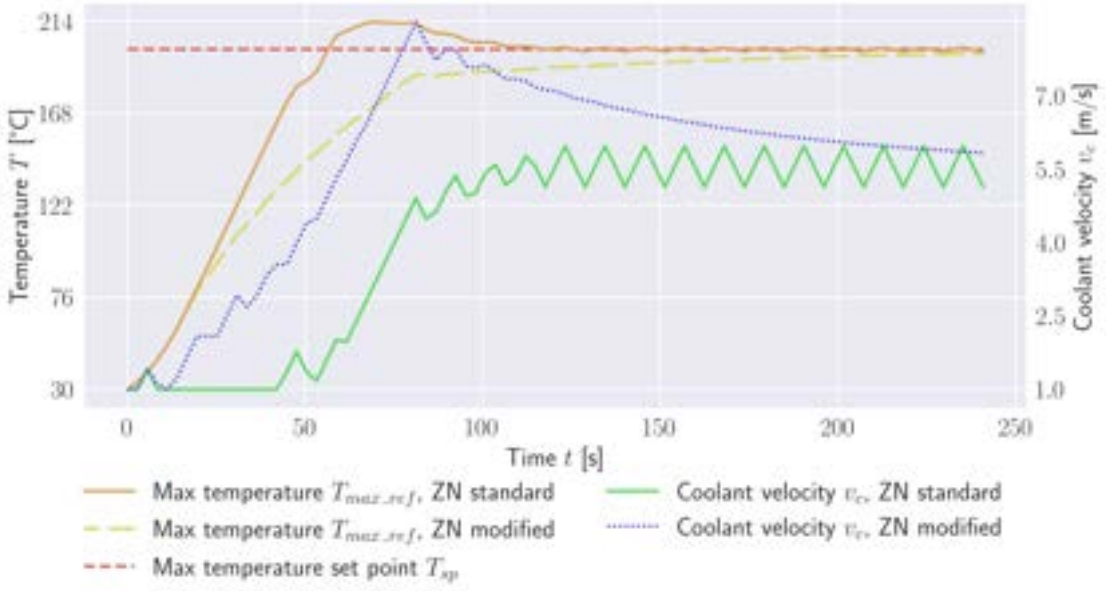


FIGURE 5.3: System response to $q(t)$ presented by Eq. 5.2 with standard and modified ZN-tuned PID parameters (Table 5.1).

5.2 Physics-Driven Machine Learning-based Digital Twinning control loop

5.2.1 Heat flux Neural Network training and testing

The dataset for training, testing, and validation is generated by performing 10000 steady-state simulations using Sample 1 geometry (Figure 4.2). Coolant velocity is randomly

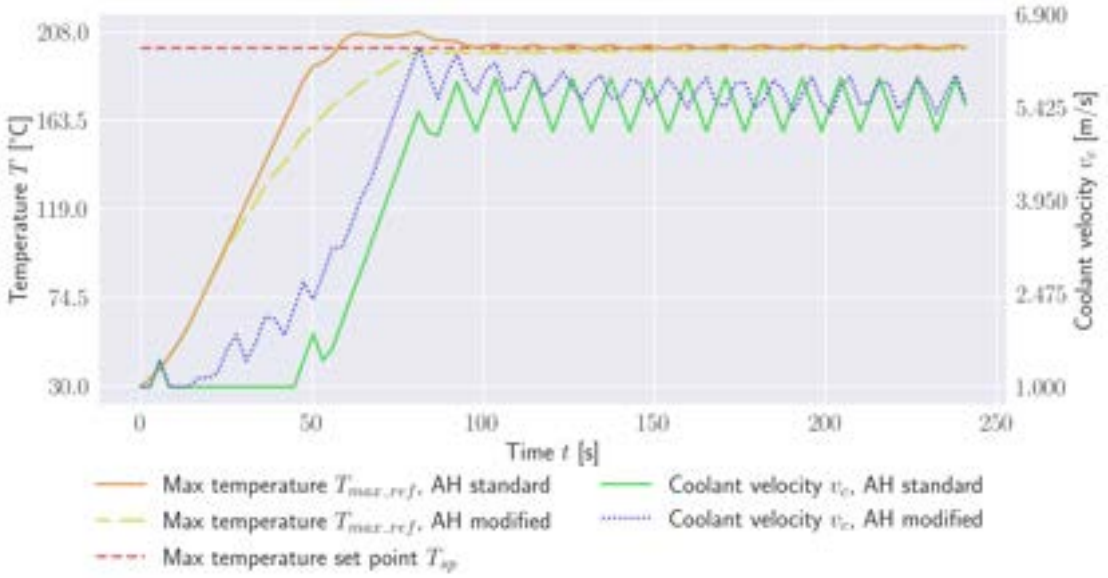


FIGURE 5.4: System response to $q(t)$ presented by Eq. 5.2 with standard and modified AH-tuned PID parameters (Table 5.1).

TABLE 5.2: Average and maximum relative solution construction errors; Table 5.9 and Figure 5.21 detail the applied heat flux $q(t)$ options.

	Sol. rec.		
Case	Avg. relative error [%]	Max. relative error [%]	Overshoot [%]
ZN tuning	0.25	1.04	2.15
AH tuning	0.12	1.04	0.37
Constant (ZN standard)	0.35	2.36	6.81
Constant (AH standard)	0.35	2.36	3.90
Constant (ZN modified)	0.12	2.19	-1.09
Constant (AH modified)	0.26	2.36	-0.03

varied between 1 and 10m/s, whilst the uniform heat flux is randomly varied between 0 and $2300 kW/m^2$. Corresponding Reynolds numbers *Re* are between 12,477 and 124,768 thus making the flow highly turbulent. 90% of this dataset is used for testing and 10% for training and validation. Each input and output is normalised using Min-Max normalisation as follows:

$$x_{norm} = \frac{x - x_{min}}{x_{max} - x_{min}} \tag{5.4}$$

Then the normalised NN output could be converted back to the original range as:

$$x = x_{norm} \times (x_{max} - x_{min}) + x_{min} \tag{5.5}$$

In order to mitigate overfitting, k-fold cross-validation is used to assess the NNs. The training dataset is divided into k equal-sized subsets (or folds). The NN model is trained k times, each time using a different fold as the validation set and the remaining $k-1$ folds as the training set. This ensures that every data point is used once for validation and $k-1$ times for training. 4 subsets are used in this work. Three error types are used during training, validation, and testing, RMSE, MAPE, and MSE:

$$RMSE(y_{pred}, y_{true}) = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_{true,i} - y_{pred,i})^2} \tag{5.6}$$

$$MAPE(y_{pred}, y_{true}) = \frac{100}{n}\sum_{i=1}^{n}\left|\frac{y_{true,i} - y_{pred,i}}{y_{true,i}}\right| \tag{5.7}$$

$$MSE(y_{pred}, y_{true}) = \frac{1}{n}\sum_{i=1}^{n}(y_{true,i} - y_{pred,i})^2 \tag{5.8}$$

where y_{pred} and y_{true} are the value predicted by the NN and the true value from the dataset, respectively.

Due to its relatively low memory requirements and good performance across a wide range of ML problems, Adam optimiser is often the default choice for MLtraining. Hence, it is used in this work. The training dataset is split into 1064 batches and MSE loss is used for backpropagation. After the training of a model for each fold is complete, the NNs are assessed based on validation RMSE and MAPE. The learning rate is set to 0.001. The hyperparameters explored are listed in Table 5.3. *Sigmoid* activation function is selected for the output layer since the normalised output is limited to the range $[0.0, 1.0]$. NVIDIA RTX 6000 Ada Graphical Processing Unit (GPU) is used for training all NNs in this work.

The best performing hyperparameter configuration is selected by comparing their average MAPEs achieved as shown in Table 5.4. Consequently, 64-32-16 NN configuration

TABLE 5.3: NN hyperparameters explored for the heat flux and coolant NNs.

Layer No.	**Output dimension**							**Activation function**
1	8	32	64	16	8	16	32	*tanh*
2	4	8	32	8	8	16	32	*tanh*
3	2	4	16	4	8	16	32	*tanh*
4	1	1	1	1	1	1	1	*sigmoid*

is selected.

TABLE 5.4: Hyperparameter configuration results for the heat flux NNs.

Hyperparameter configuration	**Mean MAPE [%]**	**Max. MAPE [%]**	**Min. MAPE [%]**
8-4-2	4.791	8.386	2.692
32-8-4	3.519	6.917	1.084
64-32-16	1.943	3.462	1.106
16-8-4	2.827	5.286	1.204
8-8-8	2.543	4.431	1.296
16-16-16	2.165	4.282	1.090
32-32-32	2.309	4.171	1.350

Figures 5.5 and 5.6 display the error convergence during training and validation for the best performing fold.

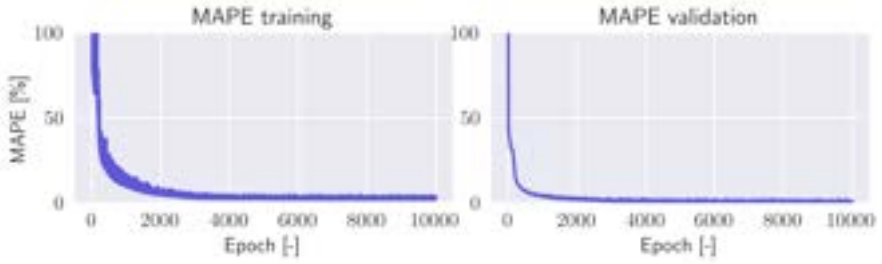


FIGURE 5.5: Training process of heat flux NN - MAPE errors.

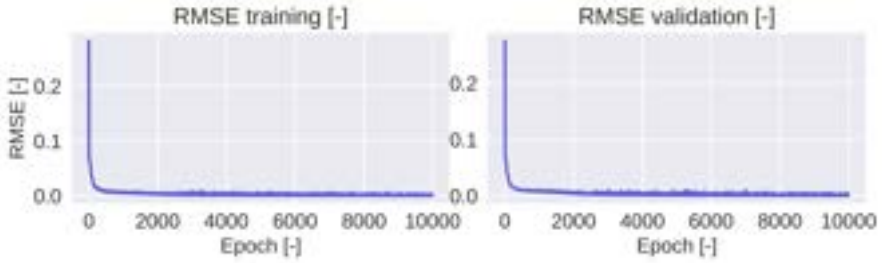


FIGURE 5.6: Training process of heat flux NN - RMSE errors.

During the testing process, the accuracy is assessed by computing MAPE errors for the testing dataset and classifying them into the following categories:

$$\text{True Positive (TP)} = \sum_{i=1}^{N} 1\left(e_i > 0 \;\text{ and }\; |e_i| < 2.5\%\right) \tag{5.9}$$

$$\text{True Negative (TN)} = \sum_{i=1}^{N} 1\left(e_i < 0 \;\text{ and }\; |e_i| < 2.5\%\right) \tag{5.10}$$

$$\text{False Positive (FP)} = \sum_{i=1}^{N} 1\left(e_i > 0 \;\text{ and }\; |e_i| \geq 2.5\%\right) \tag{5.11}$$

$$\text{False Negative (FN)} = \sum_{i=1}^{N} 1\left(e_i < 0 \;\text{ and }\; |e_i| \geq 2.5\%\right) \tag{5.12}$$

where $1(\cdot)$ is the indicator function, which returns 1 if the condition is true, and 0 otherwise. The accuracy is estimated by using the following formula:

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + +\text{FP} + \text{FN}} \tag{5.13}$$

Figure 5.7 shows the testing error distribution, while Table 5.5 lists the number of errors in each category. The overall accuracy is 0.976 with only 24 errors out of 1000 lying outside of 2.5% limit.

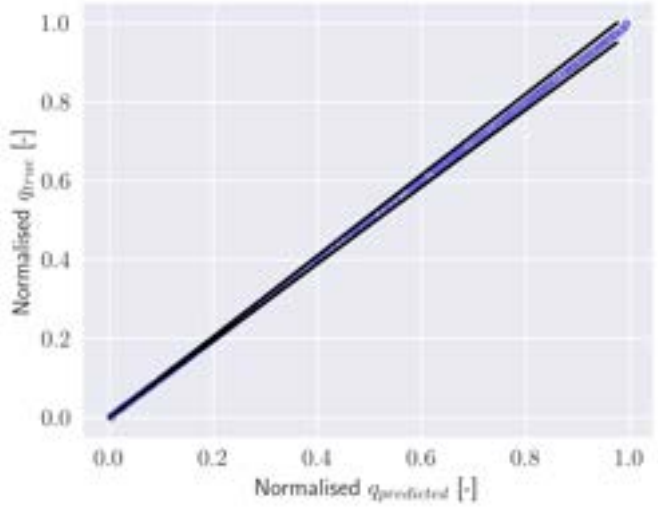


FIGURE 5.7: Testing accuracy of heat flux NN - 0.976 accuracy.

TABLE 5.5: MAPE error classification for the heat flux NNs.

MAPE error classification	**Number of errors**
TP	932
TN	44
FP	17
FN	7

5.2.2 Coolant Neural Network training and testing

The process for training, validating, and testing a coolant NN follows the same procedure as for the heat flux NN described in the previous sub-section. The hyperparameter options explored are listed in Table 5.3. The corresponding training and validation results are presented in Table 5.6. 32-32-32 NN is the best-performing configuration based on average MAPE, therefore it is selected.

TABLE 5.6: Hyperparameter configuration results for the coolant NNs.

Hyperparameter configuration	**Mean MAPE [%]**	**Max. MAPE [%]**	**Min. MAPE [%]**
8-4-2	20.665	36.144	8.079
32-8-4	16.254	29.227	4.960
64-32-16	14.51	30.774	2.928
16-8-4	14.931	28.625	6.076
8-8-8	14.568	25.011	4.307
16-16-16	13.280	27.309	2.818
32-32-32	10.209	17.566	2.092

Figures 5.8 and 5.9 display the error convergence during training and validation for the best performing fold.

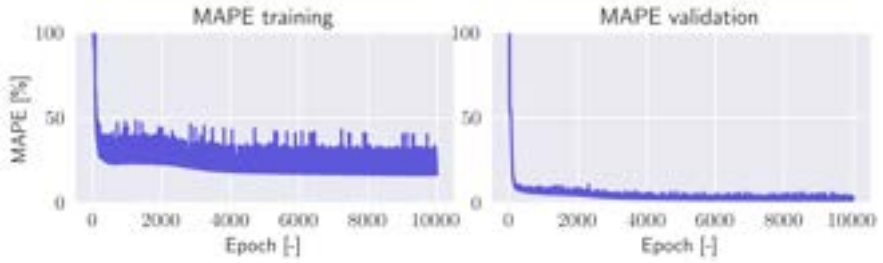


FIGURE 5.8: Training process of heat flux NN - MAPE errors.

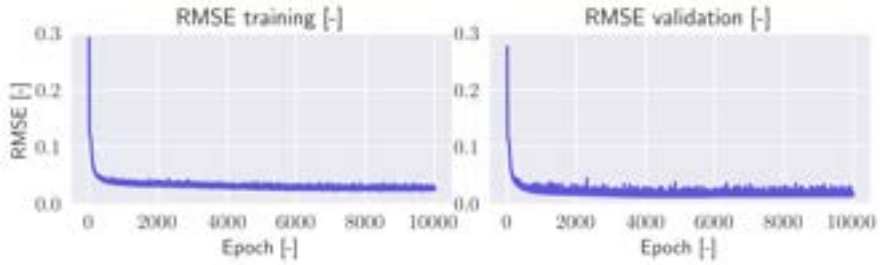


FIGURE 5.9: Training process of heat flux NN - RMSE errors.

Figure 5.10 shows the testing error distribution, while Table 5.7 lists the number of errors in each category. The overall accuracy is 0.902 with 98 out of 1000 errors lying outside of 2.5% limit. The accuracy of coolant NN is lower than the accuracy of the heat flux NN; however, it is still above 90%.

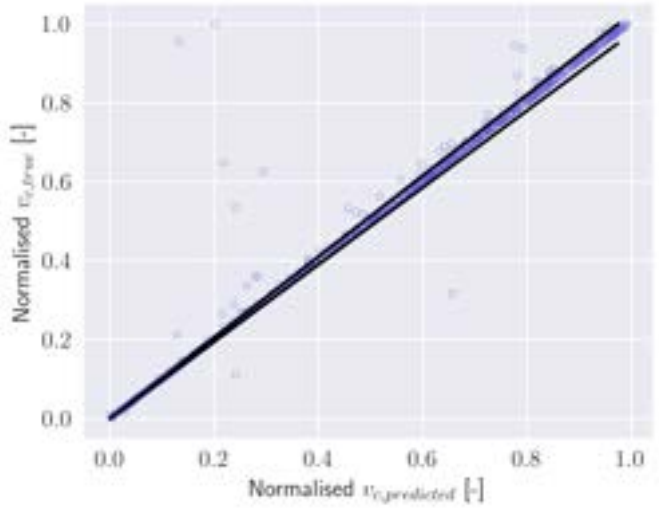


FIGURE 5.10: Testing accuracy of coolant NN - 0.902 accuracy.

TABLE 5.7: MAPE error classification for the coolant NNs.

MAPE error classification	**Number of errors**
TP	80
TN	822
FP	18
FN	80

5.2.3 Physics-Driven Machine Learning-based Digital Twinning control tuning

Once the two NNs are trained, validated, and tested, the whole PD-ML DT needs to be tuned and the effect of the control parameters assessed. Primarily, there are two control parameters, N_{int} and N_{ext}. These parameters are introduced in Section 3.4. These parameters define the heat flux trend prediction module. It examines a sequence of recently estimated flux values, q_{Nint}, where $Nint$ specifies the length of the interpolation window used to capture recent history. From these values, an interpolation function f_{trend} is constructed. This function is then evaluated at a future point determined by $Next$, the extrapolation horizon, to obtain the predicted heat flux value q_{future}. Finally, the maximum is selected between q_{future} and the most recent values in q_{Nint}.

A range of N_{int} and N_{ext} parameters are explored on a constant heat flux scenario expressed by Eq. 5.2. The performance is assessed based on the overshoot defined as:

$$\text{Overshoot} = \frac{T_{max\_ref} - T_{sp}}{T_{sp}} \times 100\% \tag{5.14}$$

Firstly, N_{ext} is set to 0, and just N_{int} is varied. Regardless of N_{int} selected when $N_{ext} = 0$, the overshoot stays constant at 2.42%. However, the difference between these cases could be seen when analysing the system response, shown in Figure 5.11. The coolant velocity stabilises and reaches steady state for all N_{int} values. The cases exhibit exactly the same behaviour when the maximum temperature just reaches the temperature set point. The difference between the cases lies in the fact that increasing N_{int} increases the time it takes for the velocity to reach the steady state value.

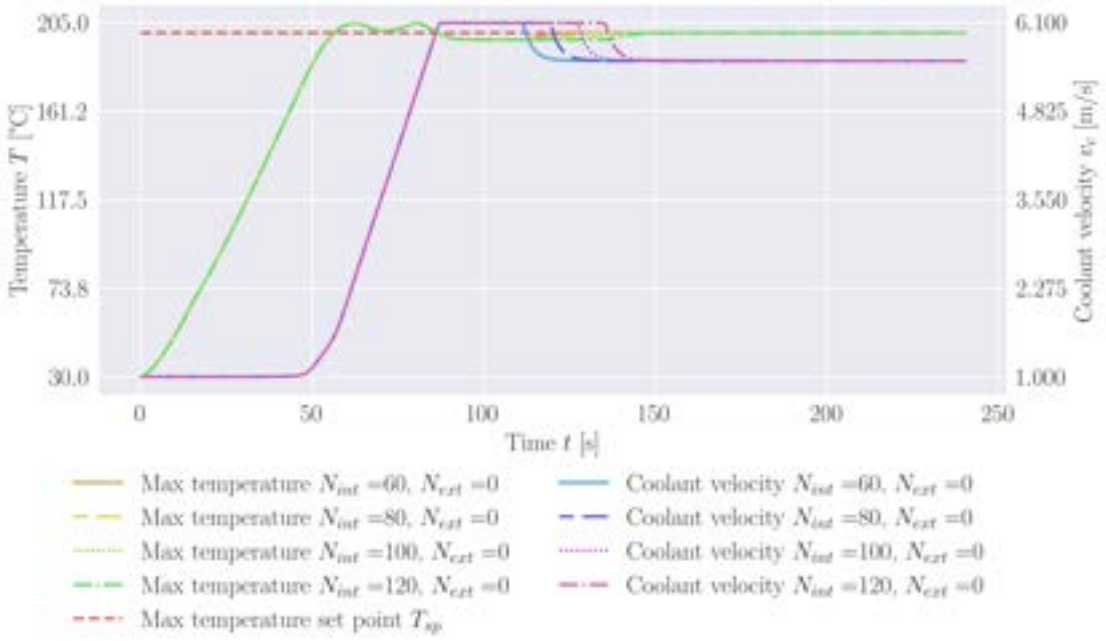


FIGURE 5.11: System response constant heat flux with linear transition (Eq. 5.2) for various N_{int} values and N_{ext} equal to 0.

Figure 5.12 shows the heat flux constructed by the heat flux NN and compares it with the transient applied heat flux. This constructed heat flux can be referred to as the steady-state equivalent heat flux. During the linear transition in the beginning of the simulation, the constructed heat flux generally follows the true heat flux; however, at one point in the beginning it deviates by 83.01%. the constructed heat flux approaches the true heat flux as the sample reaches steady state.

Next, the system response is trialled for various non-zero combinations of N_{int} and N_{ext} values. Figure 5.13 shows the obtained overshoots. All achieved overshoots are below zero, meaning that the maximum temperature always stays below the set point temperature. The lowest overshoot is achieved for N_{int} equal to 100 and N_{ext} equal to

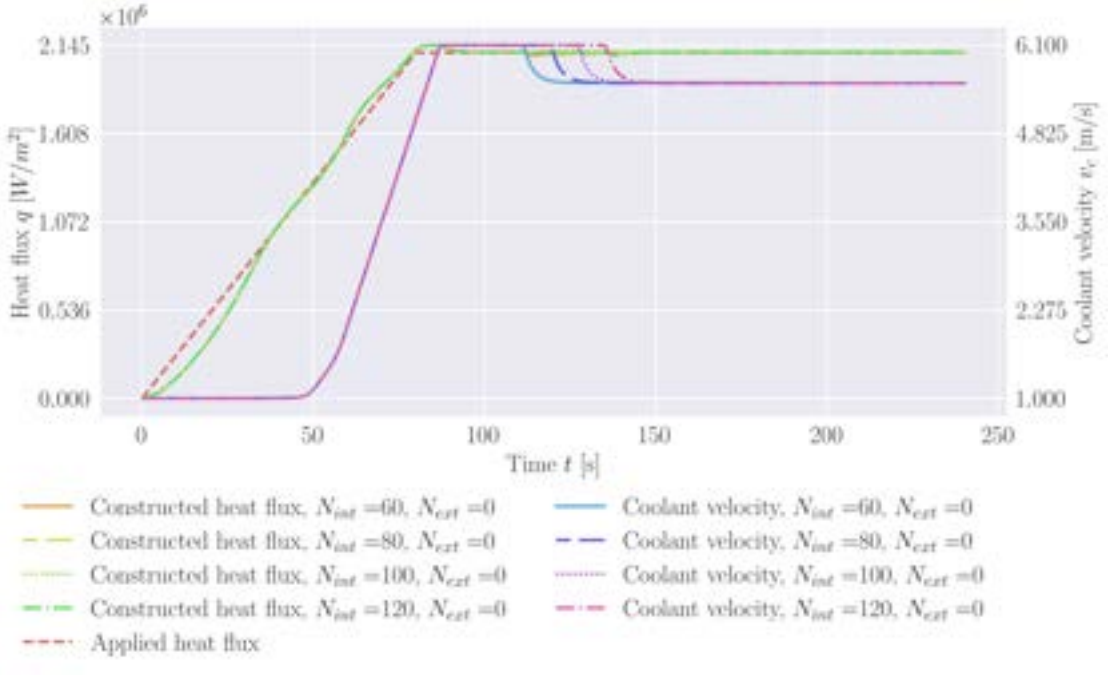


FIGURE 5.12: Heat flux constructed by the heat flux NN as a response to the heat flux with linear transition (Eq. 5.2) for various N_{int} values and N_{ext} equal to 0.

120, while the overshoot closest to zero is achieved for N_{int} equal to 60 and N_{ext} equal to 120.

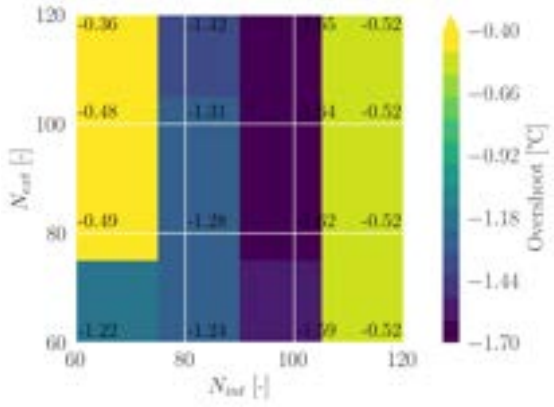


FIGURE 5.13: Overshoots achieved for various non-zero N_{int} and N_{ext} values.

The difference in system response between various cases with N_{ext} equal to 60 is shown in Figure 5.14. Figure 5.15 shows the corresponding constructed heat flux. The extrapolation assists in reducing the overshoot and keeping it negative; however, in all cases it causes the constructed heat flux to fluctuate, which leads to the coolant velocity oscillating around its steady state value. In the cases with $N_{ext} = 0$, the coolant velocity essentially consists of five main sections: 1. constant at 0m/s, 2. gradual increase to the value above the steady state, 3. constant at the value above steady state, 4. gradual

decrease to the steady state, 5. constant at the steady state. On the other hand, this is not observed for the cases with $N_{ext} = 60$. For these cases, the coolant velocity consists of four main sections: 1. constant at 0m/s, 2. rapid increase to the value above the steady state, 3. rapid decrease to the value below steady state, 4. continuous oscillations around the steady state point. Overall, the coolant velocity starts to increase earlier and the changes are more rapid. It allows the control to react faster as it attempts to estimate the future constructed heat flux. However, the extrapolation leads to the constructed heat flux fluctuation, which in turn leads to coolant velocity oscillations. Consequently, the temperature converges to the approximation of the steady state.

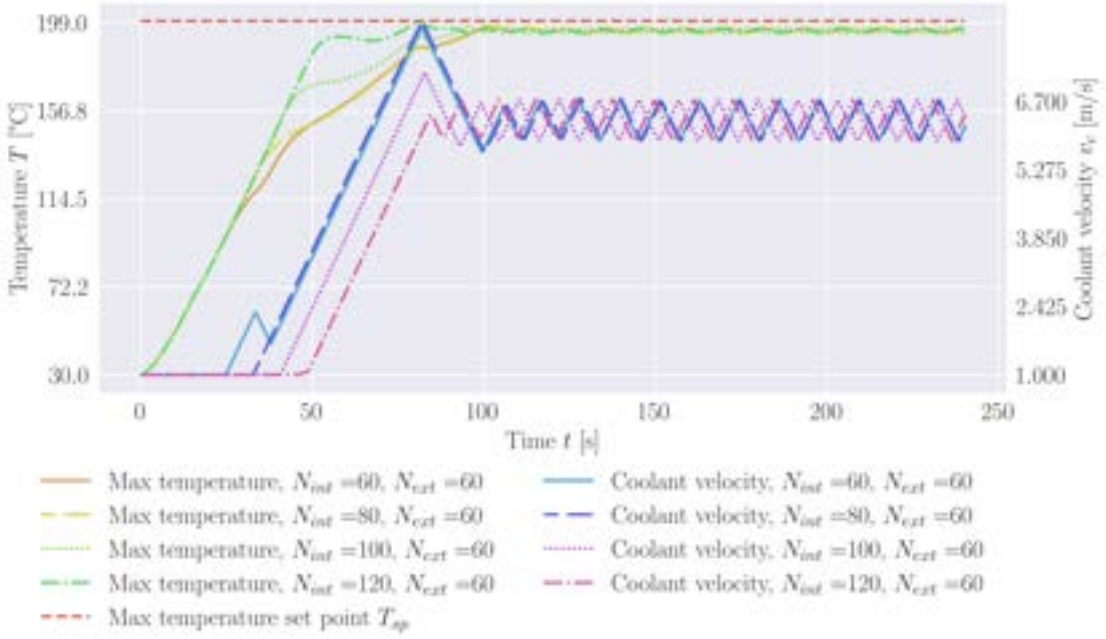


FIGURE 5.14: System response constant heat flux with linear transition (Eq. 5.2) for various N_{int} values and N_{ext} equal to 60.

Similar system responses could be observed when considering the cases with $N_{ext} = 80$, as displayed in Figures 5.16 and 5.17. The main difference lies in the case for when $N_{int} = 60$. After the coolant velocity reaches its peak, it overcompensates, decreases to the value below the steady state and only then it find its equilibrium fluctuating around the steady state point.

An attempt could be made in reconciliation the cases with N_{ext} equal to 0 and the cases with non-zero N_{ext}. The extrapolation could be switched off when the difference between the two subsequent constructed heat flux values becomes less than a certain

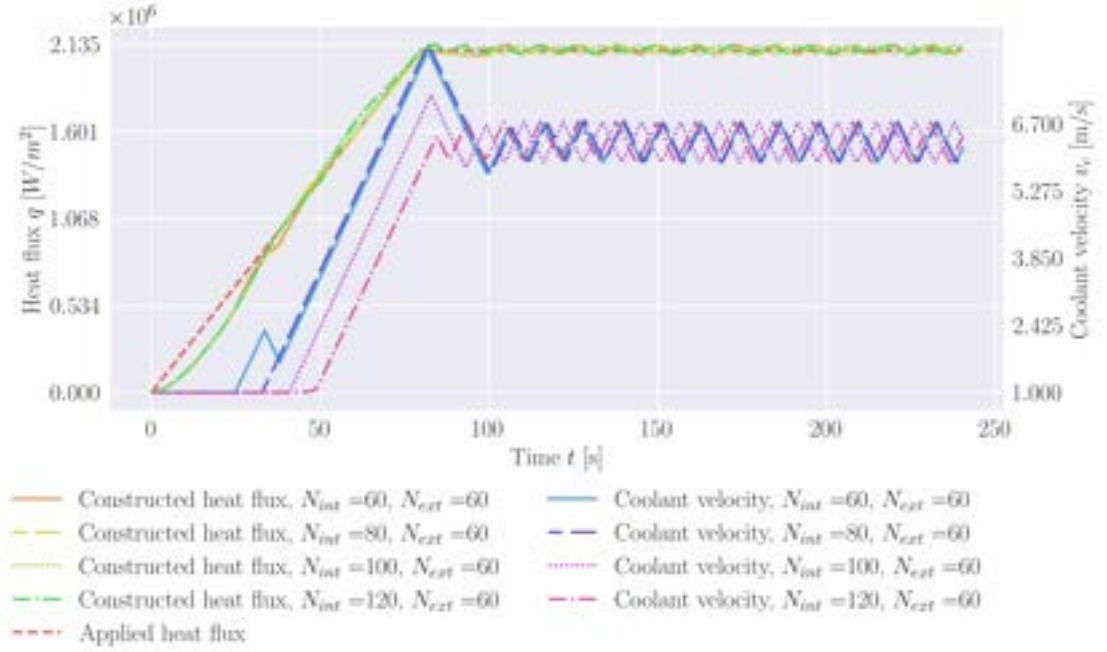


FIGURE 5.15: Heat flux constructed by the heat flux NN as a response to the heat flux with linear transition (Eq. 5.2) for various N_{int} values and N_{ext} equal to 60.

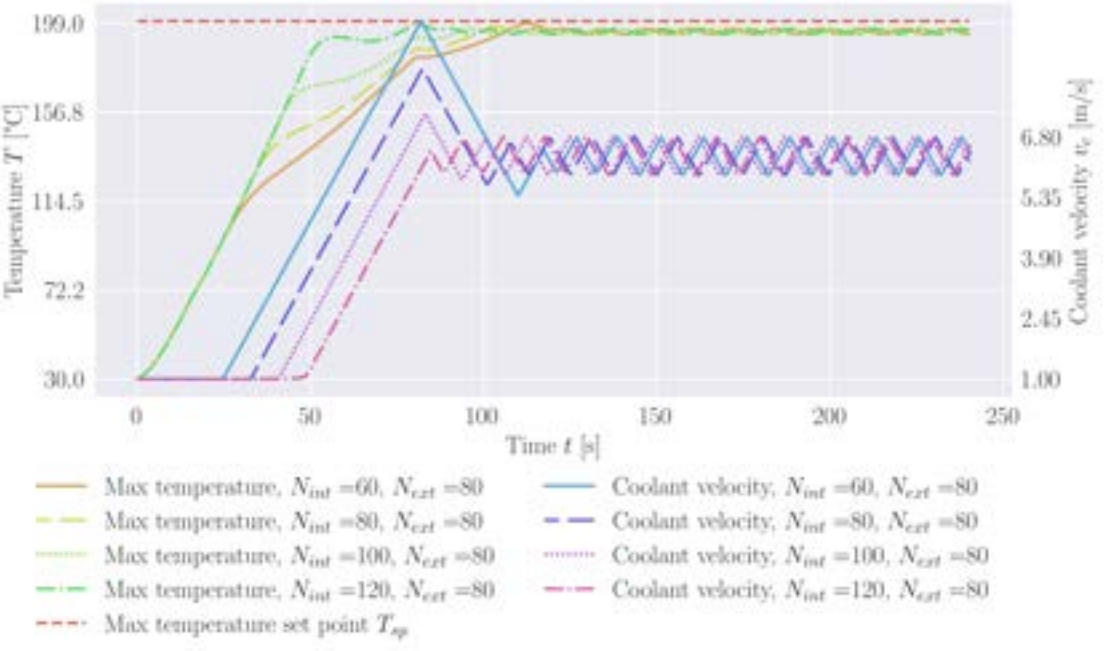


FIGURE 5.16: System response constant heat flux with linear transition (Eq. 5.2) for various N_{int} values and N_{ext} equal to 80.

limit:

$$q_{change} = \frac{\left|q_{rec}^{n-1} - q_{rec}^{n}\right|}{q_{rec}^{n}} \times 100\%$$

$$q_{change} < q_{change,lim} \tag{5.15}$$

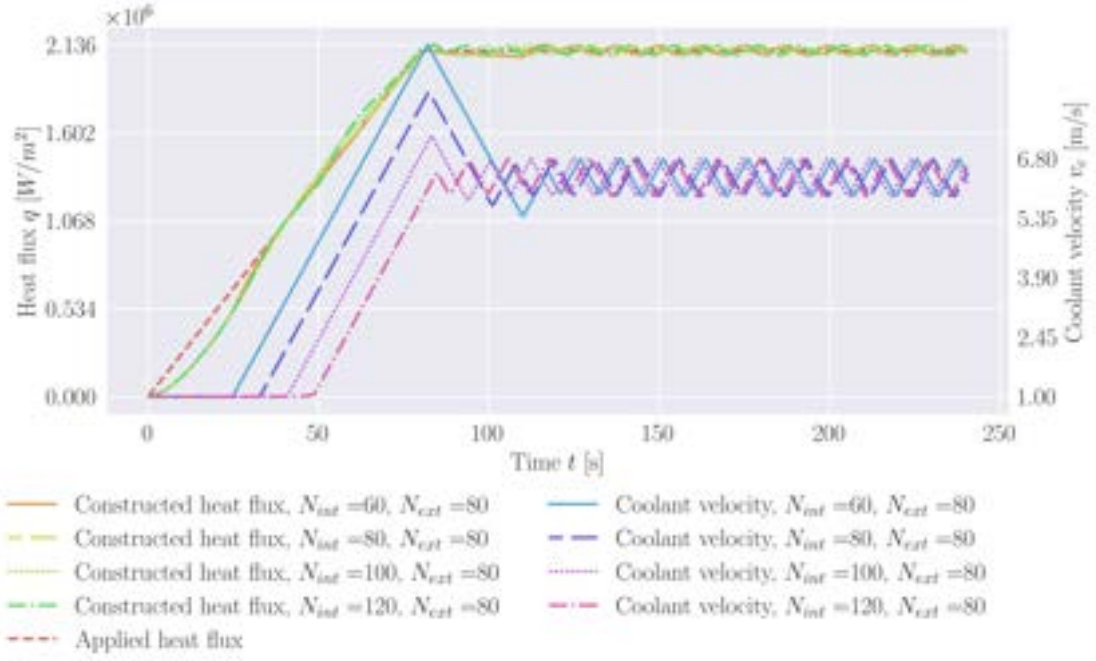


FIGURE 5.17: Heat flux constructed by the heat flux NN as a response to the heat flux with linear transition (Eq. 5.2) for various N_{int} values and N_{ext} equal to 80.

For example, the cases for when this limit is set to 1% could be considered. Figure 5.18 shows the corresponding overshoots. It is evident that the introduction of $q_{change,lim}$ does not improve the system response, but in fact makes it worse with all overshoots being above 2%.

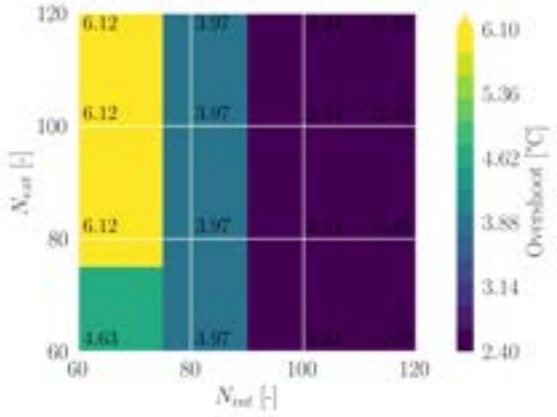


FIGURE 5.18: Overshoots achieved for various non-zero N_{int} and N_{ext} values when $q_{change,lim} = 1\%$.

Figures 5.19 and 5.20 provide more insight into the deterioration of the system response. The coolant velocity starts to increase rapidly in the beginning which bears similarity to the cases with non-zero N_{ext}. However, after the rapid increase it exhibits a downturn

which prevents it from increasing fast enough to effectively limit the maximum temperature as it approaches the set point. Overall, the system starts to behave more similar to the cases with zero N_{ext} closer to the set point.

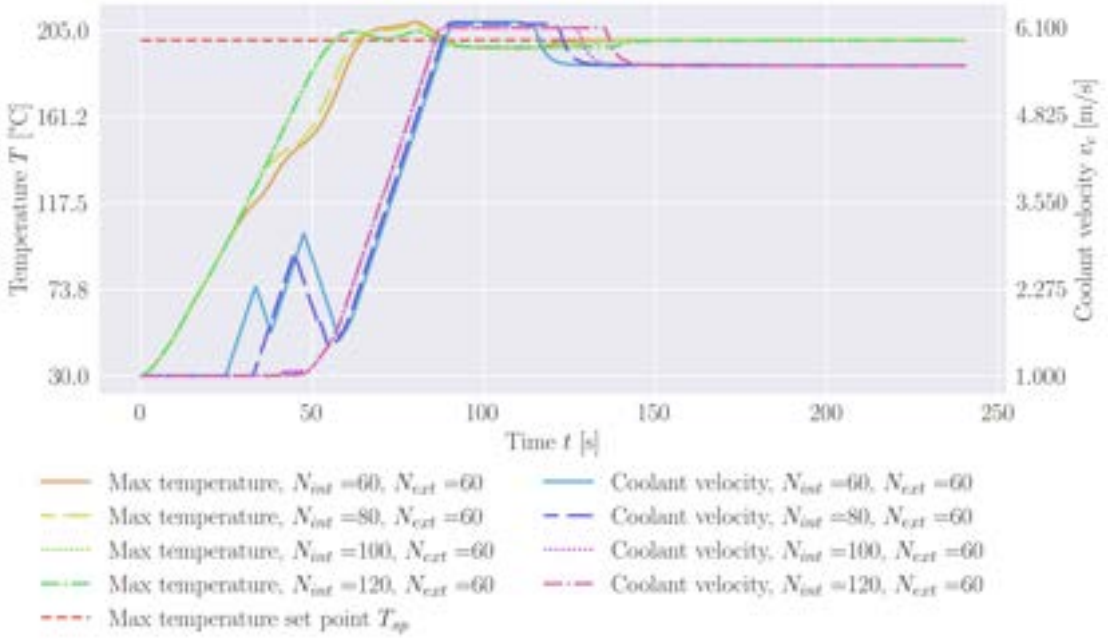


FIGURE 5.19: System response to constant heat flux with linear transition (Eq. 5.2) for various N_{int} values and N_{ext} equal to 80 when $q_{change,lim} = 1\%$.

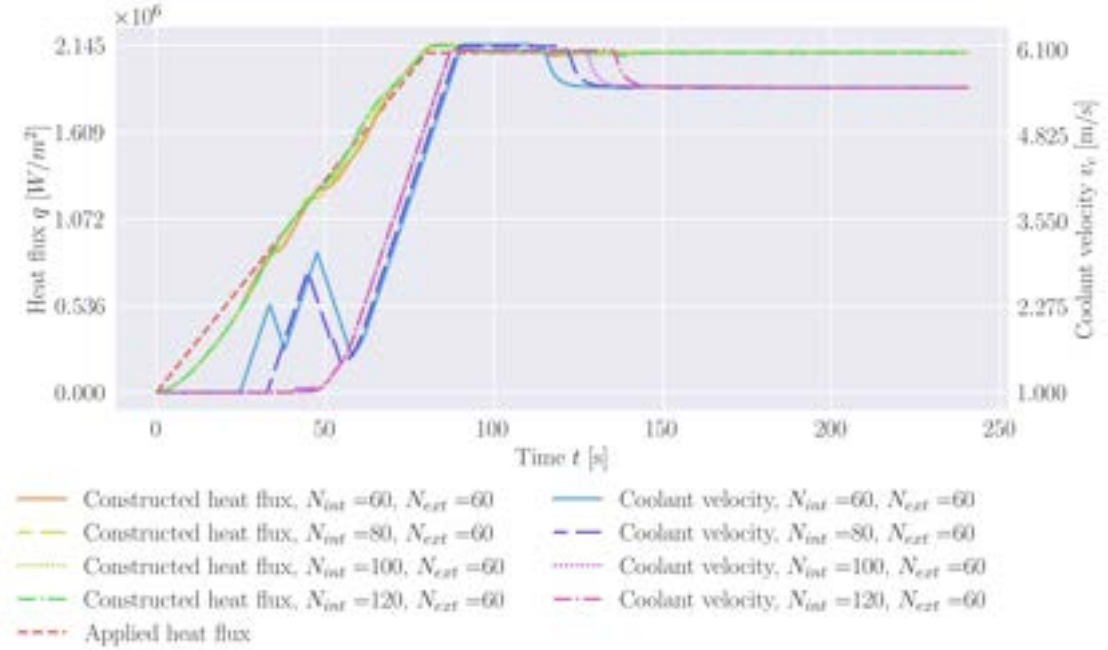


FIGURE 5.20: Heat flux constructed by the heat flux NN as a response to the heat flux with linear transition (Eq. 5.2) for various N_{int} values and N_{ext} equal to 80.

Consequently, the parameters are selected for further analysis are shown in Table 5.8.

TABLE 5.8: Selected parameter sets for PD-ML DT control.

Parameter set No.	N_{int} [−]	N_{ext} [−]	$q_{change,lim}$ [%]	**Reason for selection**
1	60	0	-	Converges to steady state
2	120	0	-	Converges to steady state
3	60	120	0.0	Closest to set point
4	100	120	0.0	Minimum overshoot

5.3 Testing parameters for Digital Twinning control loops

The control is tested using the heat flux options listed in Table 5.9 and Figure 5.21. The constant q is used in the previous sections of this chapter in order to tune the appropriate DT parameters. It is used to assess whether the controllers can bring the system to a stable equilibrium and hold the maximum temperature at the desired set point without significant oscillations or drift. Two sinusoidal waves are used as a standard test in control engineering. They check the ability of the control loops to handle smooth, periodic variations in input, they also reveals how quickly and accurately the controllers can track changes without. The triangular wave has constant ramp-up and ramp-down slopes, hence it stresses the control loops with alternating linearly increasing and decreasing inputs. It is used to check their ability to keep control during continuous ramp conditions and whether the controller can minimise overshoots when faced with sustained increases. Finally, the filtered Gaussian noise verifies the controllers' robustness to unpredictable disturbances and checks how well they suppress temperature fluctuations caused by random input variation.

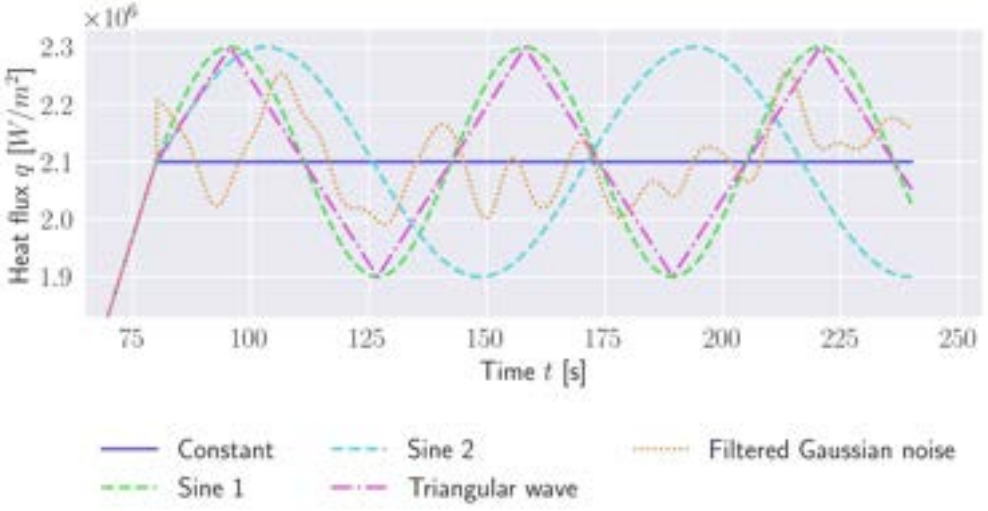


FIGURE 5.21: Applied heat flux functions used for testing, defined in Table 5.9.

TABLE 5.9: Applied heat flux functions used for testing, shown in Figure 5.21.

Function name	Function definition
Constant	$q(t) = \begin{cases} \frac{21\times10^5}{80}t & \text{if } t \leq 80 \\ 2100000 & \text{if } t > 80 \end{cases}$
Sine 1	$q(t) = \begin{cases} \frac{21\times10^5}{80}t & \text{if } t \leq 80 \\ 21\times10^5 + 2\times10^5 \sin\left(0.032\pi(t-17.5)\right) & \text{if } t > 80 \end{cases}$
Sine 2	$q(t) = \begin{cases} \frac{21\times10^5}{80}t & \text{if } t \leq 80 \\ 21\times10^5 + 2\times10^5 \sin\left(0.022\pi(t+10.5)\right) & \text{if } t > 80 \end{cases}$
Triangular wave	$q(t) = \begin{cases} \frac{21\times10^5}{80}t & \text{if } t \leq 80 \\ 21\times10^5 + 2\times10^5 \frac{2}{\pi} \arcsin\left[\sin\left(0.032\pi(t+45)\right)\right] & \text{if } t > 80 \end{cases}$
Gaussian noise	$q(t) = \begin{cases} \frac{21\times10^5}{80}t & \text{if } t \leq 80 \\ \text{Gaussian noise with } \mu = 21\times10^5,\ \sigma = 8\times10^5 & \\ \text{filtered using 1D Gaussian filter with } \sigma = 30 & \text{if } t > 80 \end{cases}$

In addition, the control is also tested by adding noise to the temperature measurements in order to verify how robust the control stays under such conditions. The noise levels are listed in Table 4.2 and Figure 4.8. They are described in Section 4.2.

5.4 Temperature control results analysis

The results are generated for three control strategies: 1. FE DT without ROM, 1. FE DT with ROM, and 3. PD-ML DT. The results are grouped by the type of applied heat flux as listed in Tables 5.9. The responses of each control strategy to each heat flux type are produced utilising varying degrees of measurement noise (Table 4.2 and Figure 4.8). Additionally, for FE DT, 4, 8, 11, and 17 measurement locations are used, whereas for PD-ML DT the results generated from 4 parameter sets are analysed (Table 5.8). The selected results are analysed in this section, the full results are available in Appendix C.

5.4.1 Best overshoots overview

Two types of overshoots are presented, primary and secondary. Primary overshoot signifies the first (initial) overshoot achieved, whilst the secondary overshoot is defined

as an overshoot occurring after the primary (first) overshoot. In this work, the primary overshoot is defined as overshoot before $t = 125s$, and secondary overshoot is defined as overshoot after $t = 125s$. This particular time is selected as it is located in the vicinity of transition from the linear section of the applied heat flux. The purpose of this split is illustrate the convergence of the maximum temperature towards the set point; moreover, the primary overshoot occurs when the transition from the linear heat flux increase to a different pattern happens.

Table 5.10 summarises the best (i.e. minimum) primary overshoots achieved across 3 DT approaches for each heat flux type and noise level. The primary overshoots for 3 approaches are compared revealing clear differences in performance, the best and worst ones between three approaches for each heat flux type and noise level are highlighted.

TABLE 5.10: Minimum primary overshoots [%] achieved across 3 DT approaches for each noise level and heat flux type (Table 5.9). The best results for each noise level and heat flux type (row-wise) are highlighted in **green bold**, while the worst results are highlighted with a red underline.

Applied heat flux type	Noise level σ [%]	FE DT without ROM	FE DT with ROM	PD-ML DT
Sine 1 wave	0.0	0.649	**-1.856**	-1.446
Sine 1 wave	0.2	1.939	**0.584**	4.709
Sine 1 wave	1.0	7.918	10.82	**4.846**
Sine 1 wave	$\sigma(T)$	3.375	**2.67**	4.627
Sine 2 wave	0.0	1.357	**-1.749**	-0.31
Sine 2 wave	0.2	**0.388**	1.54	3.202
Sine 2 wave	1.0	5.65	12.194	**3.191**
Sine 2 wave	$\sigma(T)$	2.973	3.94	**2.942**
Triangular wave	0.0	2.976	-1.987	**-2.38**
Triangular wave	0.2	0.992	**0.243**	3.629
Triangular wave	1.0	6.028	11.618	**3.66**
Triangular wave	$\sigma(T)$	3.161	**2.701**	3.478
Filtered Gaussian noise	0.0	-0.219	**-1.596**	-0.098
Filtered Gaussian noise	0.2	0.933	**0.672**	5.666
Filtered Gaussian noise	1.0	**4.47**	10.569	5.637
Filtered Gaussian noise	$\sigma(T)$	**2.597**	2.912	5.399

Overall, FE DT with ROM demonstrates the best behaviour with 8 out of 16 overshoots being minimum within approaches considered, whereas PD-ML DT frequently exhibits the largest overshoots (8 out of 16). FE DT without ROM tends to fall in between (10

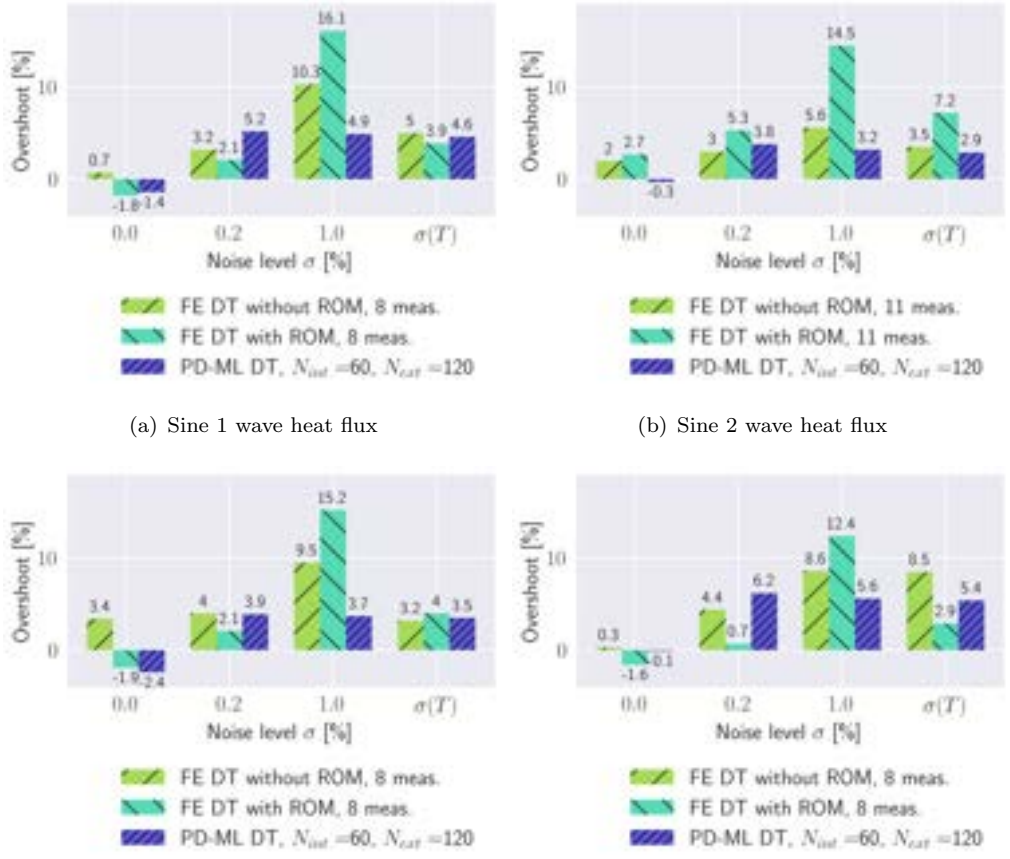
out of 16 overshoots are neither the best nor the worst), meaning that the introduction of ROM generally improves the behaviour of FE DT model.

FE DT with and without ROM show clear sensitivity to noise as primary overshoots increase markedly as the noise level grows, reaching 7.9% and 12.1% for FE DT without and with ROM, respectively. The results in Table 5.10 also indicate that ROM usage makes FE DT more susceptible to the influence of noise. It seems that based on the results from Table 5.10 FE DT with ROM behaves better than FE DT without ROM for lower noise levels, even achieving undershoots (negative overshoots), while the opposite is true for higher noise levels. However, for the TC-specific noise level $\sigma(T)$ both FE DT variations achieve overshoots around 3%. In contrast, PD-ML DT displays limited effect from the increasing noise level. Initially, at zero noise level PD-ML DT exhibits undershoots. However, for all other explored noise levels, including TC-specific one, the overshoots remain stable for a certain heat flux type, for example they fluctuate around 3% for the Triangular wave.

TABLE 5.11: Secondary overshoots [%] achieved across 3 DT approaches for each noise level and heat flux type (Table 5.9). They are observed in the same cases for which the primary overshoots listed in Table 5.10 are minimal. The best results for each noise level and heat flux type (row-wise) are highlighted in **green bold**, while the worst results are highlighted with a red underline.

Applied heat flux type	Noise level σ [%]	FE DT without ROM	FE DT with ROM	PD-ML DT
Sine 1 wave	0.0	0.649	0.82	**-1.446**
Sine 1 wave	0.2	**1.939**	3.566	4.347
Sine 1 wave	1.0	7.918	11.806	**-0.463**
Sine 1 wave	$\sigma(T)$	3.375	3.661	**-0.759**
Sine 2 wave	0.0	1.357	1.493	**-0.31**
Sine 2 wave	0.2	**0.294**	1.109	2.361
Sine 2 wave	1.0	3.358	10.82	**-0.894**
Sine 2 wave	$\sigma(T)$	2.973	4.406	**0.51**
Triangular wave	0.0	2.976	2.835	**-2.383**
Triangular wave	0.2	**0.992**	4.205	1.689
Triangular wave	1.0	5.29	12.168	**-0.88**
Triangular wave	$\sigma(T)$	2.8	3.723	**-0.592**
Filtered Gaussian noise	0.0	**-0.219**	1.165	-0.098
Filtered Gaussian noise	0.2	**0.933**	2.365	3.614
Filtered Gaussian noise	1.0	4.304	10.085	**-2.56**
Filtered Gaussian noise	$\sigma(T)$	2.597	4.608	**-1.312**

Distinct behaviours could be discerned when analysing the secondary overshoots observed in the same cases for which the primary overshoots listed in Table 5.10 are minimal. FE DT without ROM either maintains a constant overshoot (i.e. the primary and secondary overshoots are equal) or exhibits a gradual decrease in overshoot over time. This gradual decline is particularly noticeable at higher noise levels. Table 5.11 presents a more pessimistic picture for FE DT with ROM compared to Table 5.10. In particular, in a significant number of cases the secondary overshoot exceeds the primary overshoot, which may indicate that FE DT with ROM requires more time than FE DT without ROM to achieve convergence of the maximum temperature to the set point. Furthermore, the secondary overshoots produced by FE DT with ROM are almost always the largest among the three DT models considered.



(a) Sine 1 wave heat flux

(b) Sine 2 wave heat flux

(c) Triangular wave heat flux

(d) Filtered Gaussian noise heat flux

FIGURE 5.22: Overshoot's dependency on noise level for 3 DT approaches.

Finally, in spite of PD-ML DT's tendency to frequently exhibit the largest primary overshoots out of 3 DT models (Table 5.10), PD-ML DT oftentimes displays the lowest secondary overshoots (11 out of 16 cases). This might mean that this model manages to

achieve the fastest convergence of the maximum temperature to the the set point and has the swiftest rate of response.

Figure 5.22 shows the example of overshoots' dependency on the noise level for 3 DT models. The results generated using 8 or 11 measurement locations are used for FE DT approaches, while the results from cases with N_{int} equal to 60 and N_{ext} equal to 120 are used for PD-ML DT. The cases shown in Figure 5.22 do not directly correspond with the best overshoots achieved, previously shown in Table 5.10. This is due to the fact that there is no setting for FE DT (number of measurement locations) and PD-ML DT(N_{int} and N_{ext}) that produce best overshoots consistently for all noise levels and applied heat flux types. However, Figure 5.22 serves as an illustration of the common patters observed between the models. The trends seen in this figure confirm the observations made in the previous section. The overshoots associated with PD-ML DT tend to fluctuate slightly around a certain value for all noise levels apart from zero. On the other hand, FE DT's overshoots increase with the noise level.

More insights into the models' behaviour are provided in the subsequent sub-sections.

5.4.2 Finite Element-based Digital Twinning control without Reduced Order Modelling

The results for FE DT without ROM are generated by varying the number of measurement locations, thus its influence on the control is explored. Table 5.12 lists primary and secondary overshoots as well as relative errors under the applied heat flux in the form of Sine wave 1. These results indicate the general trend for the data obtained for the other heat fluxes.

As expected, for the same number of measurement locations the relative construction errors increase with the noise level. Therefore, the results indicate that more measurement locations are needed to combat this increase. And, indeed, as the number of measurement locations increase the average relative solution construction errors tend to decrease. However, the maximum relative error improvement does not directly correlate with the increase in measurement locations. The similar trend could be noticed with the primary overshoot not always improving with the measurement location increase. Consequently, Figure 5.23 shows the correlation between the primary overshoot and maximum relative

TABLE 5.12: System response under FE DT control without ROM and average and maximum time step runtimes; applied heat flux $q(t)$ is in the form of Sine wave 1 (Table 5.9 and Figure 5.21).

		Overshoot [%]		Sol. rec. + Control time		Sol. rec.	
No. of measurement locations	Noise level σ [%]	Primary	Secondary	Avg. time step runtime [s] [a]	Max. time step runtime [s] [a]	Avg. relative error [%]	Max. relative error [%]
4	0.0	0.649	0.649	2.221e+00	2.454e+00	0.184	2.192
4	0.2	1.939	1.939	2.224e+00	2.438e+00	0.244	4.769
4	1.0	15.956	14.242	2.216e+00	2.390e+00	0.697	24.52
4	$\sigma(T)$	4.722	4.722	2.238e+00	2.575e+00	0.33	8.109
8	0.0	0.736	0.736	2.272e+00	2.528e+00	0.177	2.352
8	0.2	3.223	3.223	2.289e+00	2.610e+00	0.247	5.825
8	1.0	10.305	8.678	2.288e+00	2.523e+00	0.563	25.958
8	$\sigma(T)$	4.995	4.995	2.273e+00	2.513e+00	0.301	10.072
11	0.0	1.392	1.392	2.293e+00	2.553e+00	0.148	4.871
11	0.2	2.189	2.189	2.291e+00	2.547e+00	0.199	5.629
11	1.0	7.918	7.918	2.293e+00	2.621e+00	0.468	22.276
11	$\sigma(T)$	6.021	6.021	2.270e+00	2.487e+00	0.263	7.248
17	0.0	0.816	0.816	2.268e+00	2.482e+00	0.141	5.629
17	0.2	2.49	2.49	2.288e+00	2.596e+00	0.187	6.21
17	1.0	13.024	10.159	2.286e+00	2.535e+00	0.48	21.744
17	$\sigma(T)$	3.375	3.375	2.291e+00	2.500e+00	0.227	8.223

[a] Times are given for AMD Ryzen 7 5800X 8-Core CPU.

error. The linear trendline with coefficient of determination R^2 equal to 0.783 illustrates a reasonable fit between these two variables. The linear fit is better for the cases with lower noise including the TC-specific σ, and it noticeable deviates for the noise level of 1%. Overall, the increase in the number of measurement locations improves relative errors during the solution construction causing the average relative errors to decrease. However, at the same time, this measurement location increase might negatively affect the localised maximum relative errors, which causes the control deterioration and thus the overshoot increase.

Figures 5.24a, 5.25a, 5.26a, and 5.27a show the system response examples to the various heat fluxes including the maximum temperature and the coolant velocity change over time. Maximum temperature is a variable to be controlled by adjusting the coolant velocity, hence they are plotted on the same chart to show how the coolant velocity changes influence the corresponding maximum temperature.

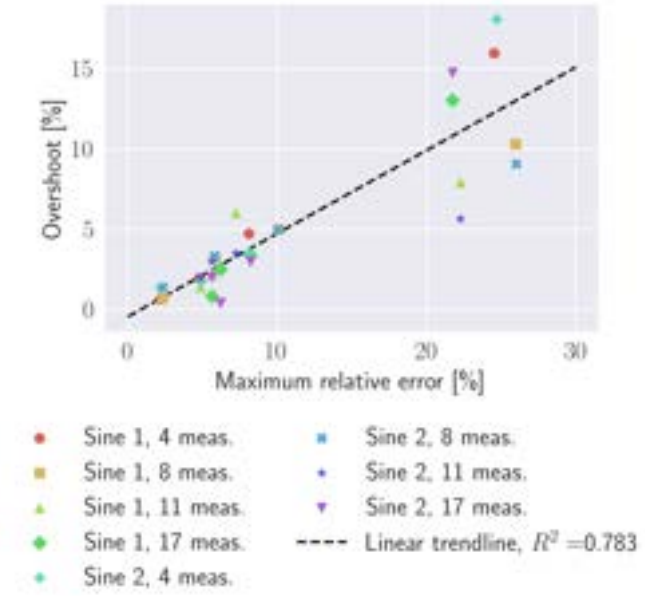
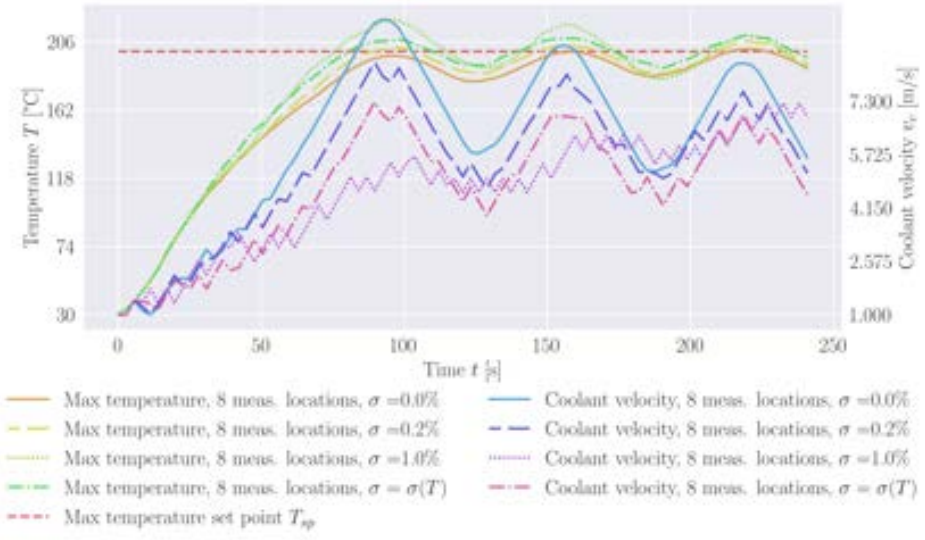


FIGURE 5.23: The dependency between the primary overshoot and the maximum relative error for FE DT without ROM; the heat flux is in the forms of Sine 1 an 2 waves.

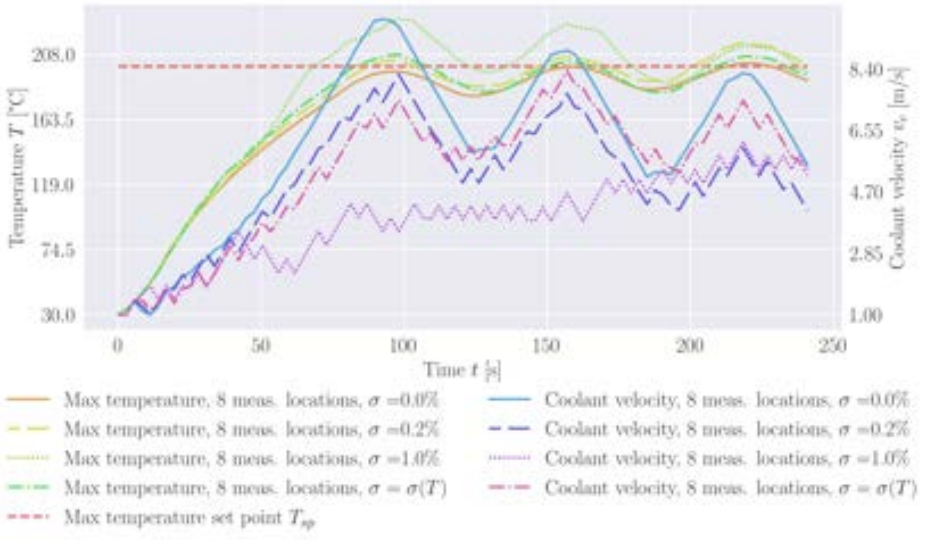
The introduction of inaccuracies to the constructed solution mainly caused by the presence of noise, from which the maximum temperature is derived, alters the obtained coolant velocity values and hence the overshoots. The coolant velocity transitions tend to be smooth when there is no noise in the measurements; however, the noise induces more abrupt changes in the coolant velocity limited by the imposed coolant acceleration. In spite of this, FE DT still successfully maintains the control of the system.

The maximum temperature convergence to the set point occurs slower than the time observed for PD-ML DT, as it is noted in Sub-section 5.4.4. It could be explained by the fact that for PD-ML DT the constructed heat flux value, defining the coolant velocity, is selected out of the range of values (previous and future), which assists in mitigating the effect the noise has on the accuracy of the constructed heat flux. However, in the case of FE DT only one maximum temperature is used to define the velocity; consequently, the accuracy of the solution construction at one particular time step plays a higher role in the control strategy.

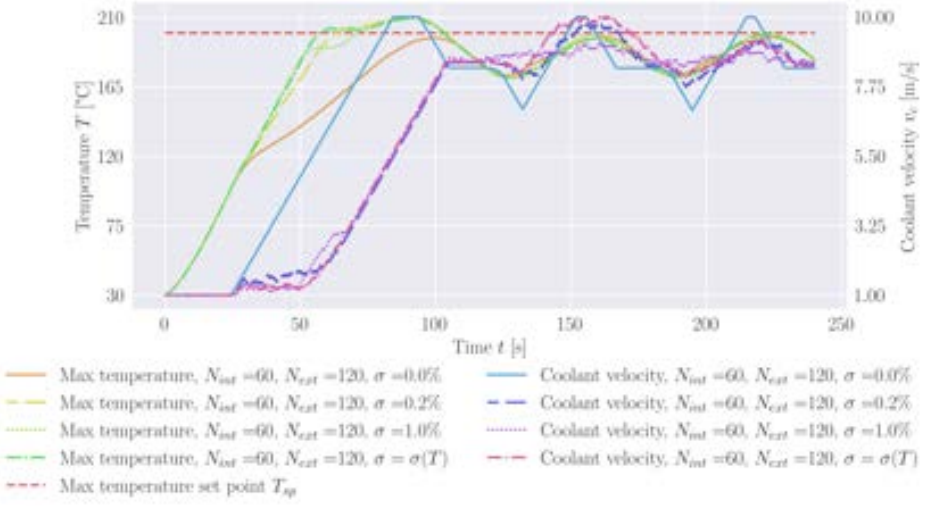
Finally, the additional information derived from the measurements comes at the cost of longer run times, as listed in Table 5.12. The time step runtimes are comprised of two parts: the time it takes to construct the solution from the measurements and the time it takes for the PID controller to adjust the coolant velocity. The control part is a



(a) FE DT without ROM, 8 measurement locations

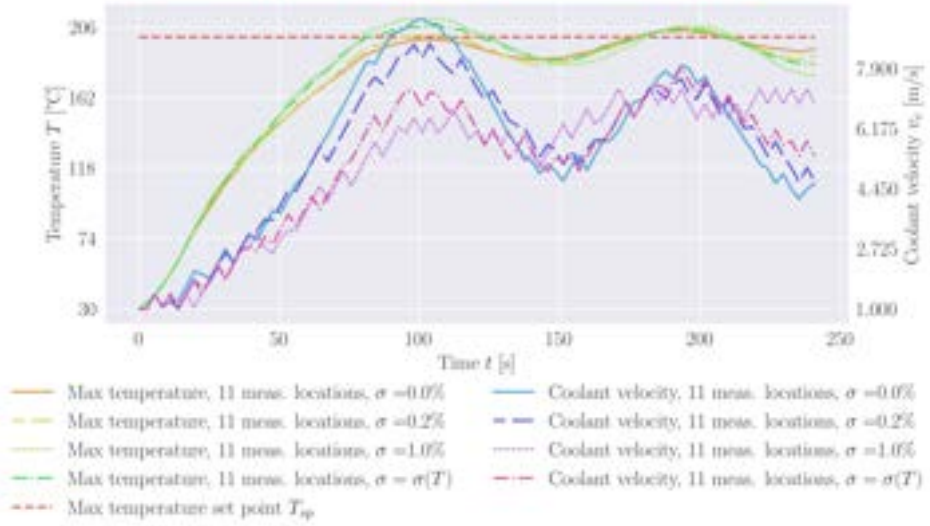


(b) FE DT with ROM, 8 measurement locations

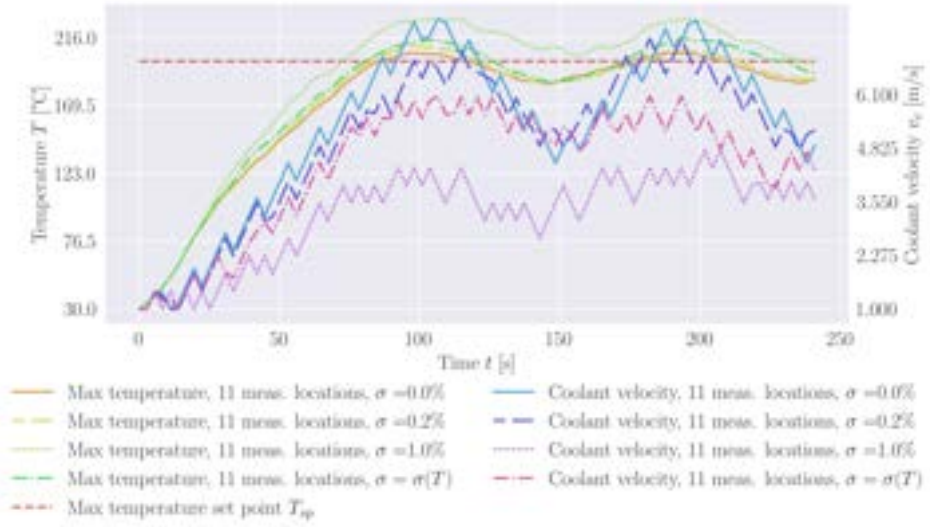


(c) PD-ML DT, N_{int} equal to 60 and N_{ext} equal to 120

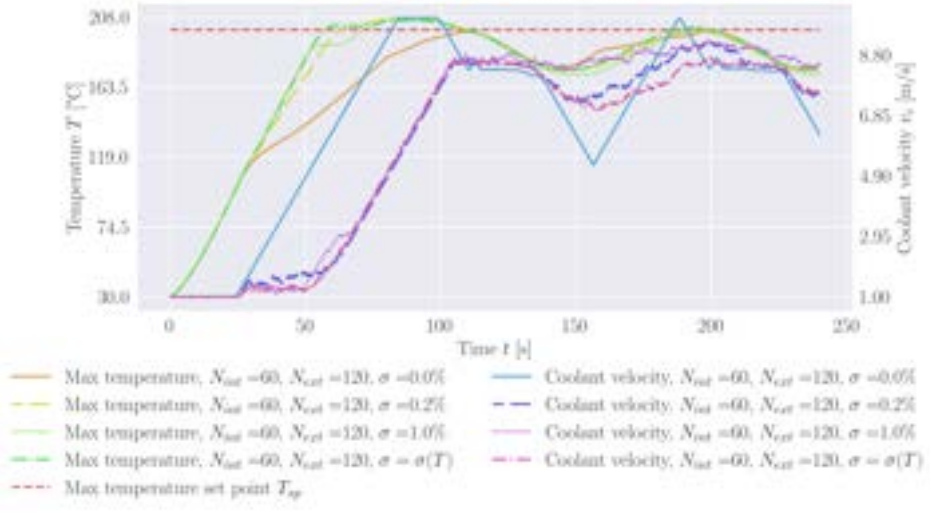
FIGURE 5.24: System responses to Sine wave 1 heat flux (Table 5.9 and Figure 5.21).



(a) FE DT without ROM, 11 measurement locations

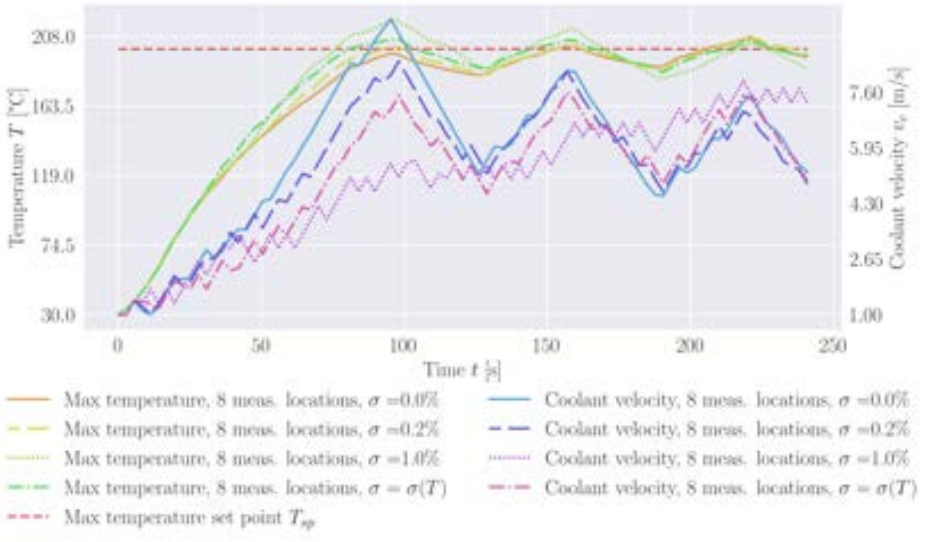


(b) FE DT with ROM, 11 measurement locations

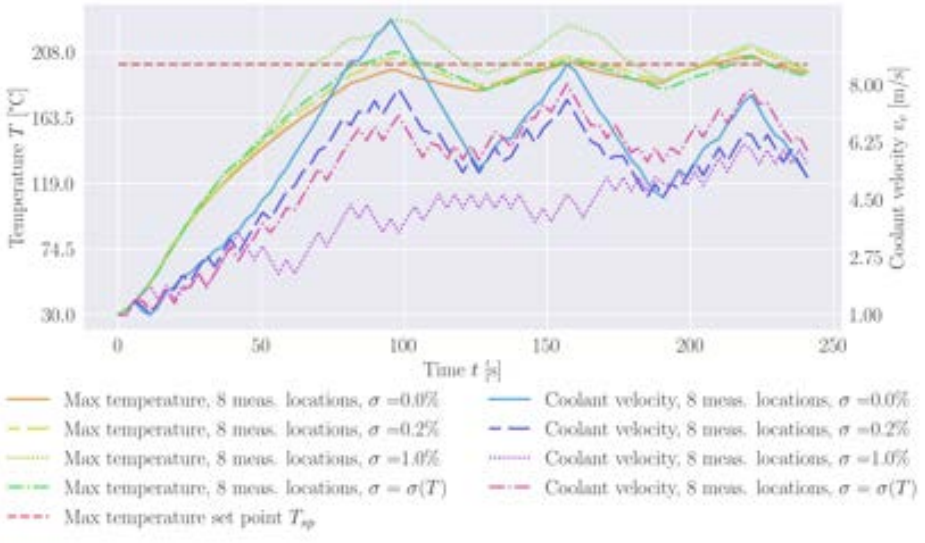


(c) PD-ML DT, N_{int} equal to 60 and N_{ext} equal to 120

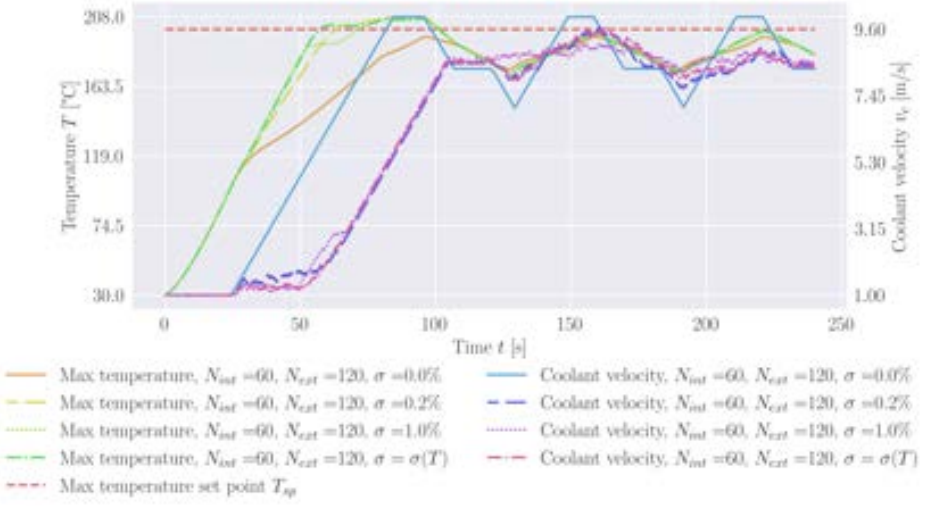
FIGURE 5.25: System responses to Sine wave 2 heat flux (Table 5.9 and Figure 5.21).



(a) FE DT without ROM, 8 measurement locations

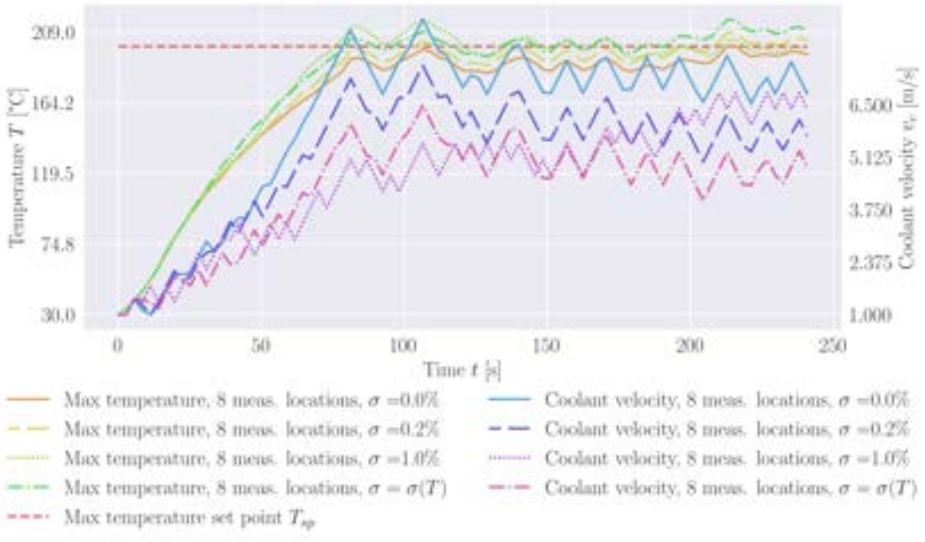


(b) FE DT with ROM, 8 measurement locations

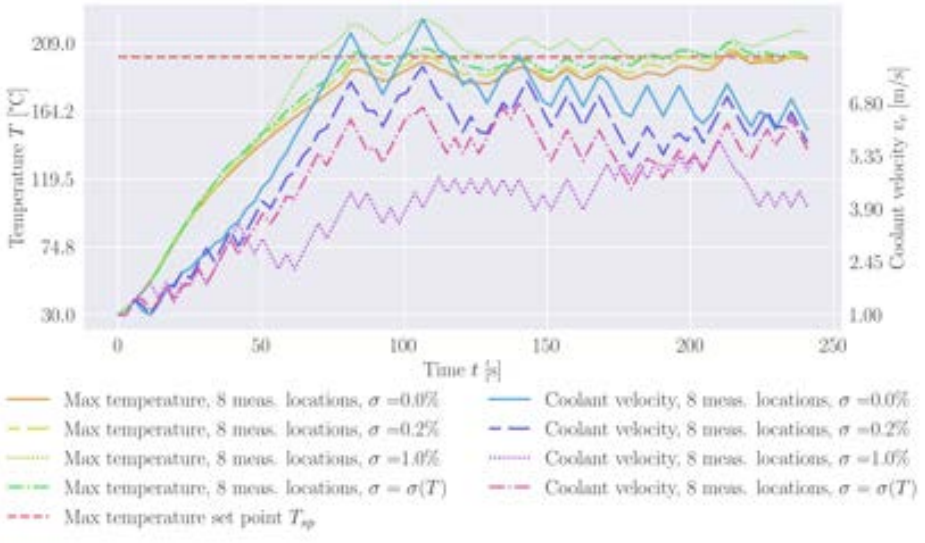


(c) PD-ML DT, N_{int} equal to 60 and N_{ext} equal to 120

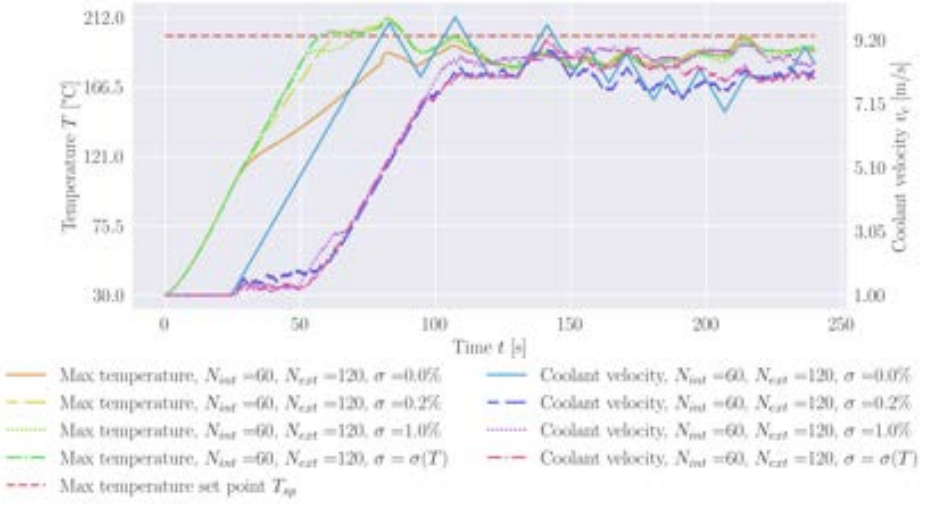
FIGURE 5.26: System responses to Triangular wave heat flux (Table 5.9 and Figure 5.21).



(a) FE DT without ROM, 8 measurement locations



(b) FE DT with ROM, 8 measurement locations



(c) PD-ML DT, N_{int} equal to 60 and N_{ext} equal to 120

FIGURE 5.27: System responses to Filtered Gaussian noise heat flux (Table 5.9 and Figure 5.21).

negligible part of the time step runtime, as the majority of time is taken by the solution contraction process. Consequently, the solution construction time should be should be taken into account when selecting the appropriate time step size used for the solution construction process and PID controller. It should be less or equal than time step size. This is necessary in order to ensure that the control process happens in near real time with a constant delay. Moreover, the mesh size might altogether limit the application of FE DT to a particular problem or geometry.

5.4.3 Finite Element-based Digital Twinning control with Reduced Order Modelling

As with FE DT without ROM discussed in the previous sub-section, the results for FE DT without ROM are generated by varying the number of measurement locations, thus its influence on the control is explored. Table 5.13 lists primary and secondary overshoots as well as relative errors under the applied heat flux in the form of Sine wave 1. These results indicate the general trend for the data obtained for the other heat fluxes.

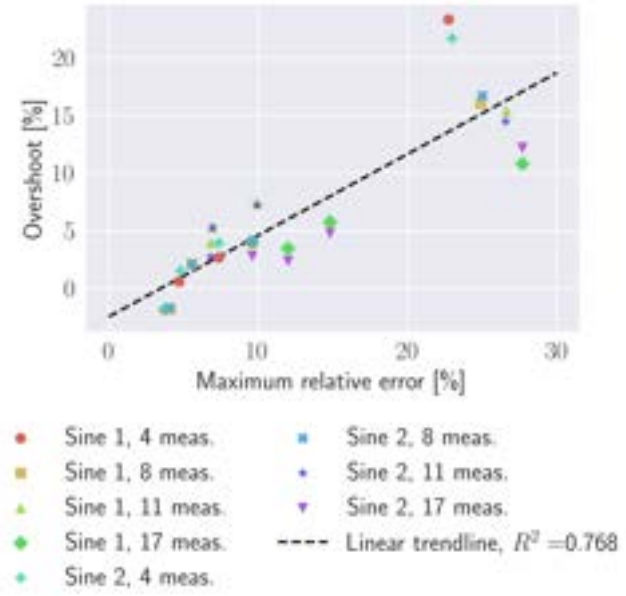


FIGURE 5.28: The dependency between the primary overshoot and the maximum relative error for FE DT with ROM; the heat flux is in the forms of Sine 1 an 2 waves.

As expected, the solution construction accuracy has deteriorated as compared with FE DT without ROM (Table 5.12) due to the inaccuracies introduced by using only a limited amount of eigenvectors and keeping the convective FE components fixed during the

TABLE 5.13: System response under FE DT control with ROM and average and maximum time step runtimes; applied heat flux $q(t)$ is in the form of Sine wave 1 (Table 5.9 and Figure 5.21).

		Overshoot [%]		Sol. rec. + Control time		Sol. rec.	
No. of measurement locations	Noise level σ [%]	Primary	Secondary	Avg. time step runtime [s] [a]	Max. time step runtime [s] [a]	Avg. relative error [%]	Max. relative error [%]
4	0.0	-1.856	0.82	3.123e-01	4.268e-01	0.75	3.768
4	0.2	0.584	3.566	3.150e-01	4.134e-01	0.839	4.706
4	1.0	23.335	16.868	3.129e-01	4.148e-01	1.241	22.735
4	$\sigma(T)$	2.67	3.661	3.163e-01	3.825e-01	0.875	7.354
8	0.0	-1.786	1.1	3.024e-01	4.409e-01	0.694	4.123
8	0.2	2.122	8.072	3.151e-01	4.003e-01	0.768	5.558
8	1.0	16.052	14.287	3.030e-01	3.852e-01	1.008	24.897
8	$\sigma(T)$	3.941	3.432	3.119e-01	4.595e-01	0.788	9.573
11	0.0	3.944	5.213	2.225e-01	2.912e-01	0.622	6.815
11	0.2	5.31	3.824	2.575e-01	3.904e-01	0.615	6.945
11	1.0	15.396	13.529	2.377e-01	3.769e-01	0.928	26.62
11	$\sigma(T)$	7.428	4.731	2.712e-01	3.916e-01	0.651	9.917
17	0.0	4.019	5.093	3.162e-01	4.118e-01	0.652	9.607
17	0.2	3.472	9.748	3.157e-01	4.272e-01	0.679	11.998
17	1.0	10.82	11.806	3.171e-01	4.073e-01	0.909	27.681
17	$\sigma(T)$	5.759	11.241	3.108e-01	3.892e-01	0.717	14.806

[a] Times are given for AMD Ryzen 7 5800X 8-Core CPU.

iterative solution process. For a fixed number of measurement points, higher noise levels lead to larger relative errors. Increasing the number of measurement points generally reduces the average relative error.

However, similarly to FE with ROM (Sub-section 5.4.2), the reduction in the maximum relative error does not scale directly with the number of measurement points. The overshoot seems to be correlated with the maximum relative error, as it is displayed in Figure 5.28. The linear trendline with a coefficient of determination of $R^2 = 0.768$ indicates a reasonable fit between the two variables, R^2 is lower than the value achieved for FE DT without ROM (Sub-section 5.4.2). The fit is stronger for lower noise levels, including the TC-specific σ but shows noticeable deviation at a 1% noise level. In general, increasing the number of measurement locations reduces the average relative error in the solution. However, this increase can also lead to higher localised maximum relative errors, which in turn deteriorates control performance and contributes to overshoot. Furthermore, the slope of the trendline shown in Figure 5.28 is higher than the slope of

the trendline in Figure 5.23, meaning that the overshoot tends to increase at a higher rate when ROM is introduced.

Figures 5.24b, 5.25b, 5.26b, and 5.27b show the system response examples to the various heat fluxes including the maximum temperature and the coolant velocity change over time. Overall, the coolant velocity patterns are similar to the ones displayed for FE DT without ROM (Sub-section 5.4.2). In the absence of measurement noise, coolant velocity changes smoothly, the presence of noise introduces sharper fluctuations, constrained by the imposed coolant acceleration. However, the inaccuracies associated with the introduction of ROM can induce some abrupt coolant velocity changes even in the scenarios with no noise (Figure 5.25b).

As it is noted in Sub-section 5.4.1, the secondary overshoots tend to be higher than the primary overshoots, which could potentially indicate a loss of control. Nevertheless, the system responses displayed in Figures 5.24b and 5.26b prove this assumption incorrect, as it is evident that the maximum temperature converges to the set point albeit at a lower pace than the one displayed for FE DT without ROM (Figures 5.24a and 5.26a).

Finally, Table 5.13 lists time step runtimes. The runtimes are significantly decreased when compared with runtimes for FE DT without ROM (Table 5.12).

This is achieved by reducing the system (calculating 100 eigenvectors) and updating the matrices and vectors corresponding to convection from the coolant every second time step using the temperature distribution from the previous time step. The run times when using ROM in conjunction with solution construction are 6-9 times shorter than when not applying ROM. However, these improved computational times still cannot compete with near instantaneous speeds achieved by PD-ML DT with the trained NNs, as it is discussed in the subsequent sub-section.

5.4.4 Physics-Driven Machine Learning-based Digital Twinning control

The PD-ML DT results are generated by varying N_{int} and N_{ext}, thus their influence on the control is explored. Table 5.14 lists primary and secondary overshoots under the applied heat flux in the form of Sine wave 1. These results indicate the general trend for the data obtained for the other heat fluxes.

TABLE 5.14: PD-ML DT system response under PD-ML DT control and average and maximum time step runtimes; applied heat flux $q(t)$ is in the form of Sine wave 1 (Table 5.9 and Figure 5.21).

Parameter set			Overshoot [%]		Control time	
N_{int}	N_{ext}	Noise level σ [%]	Primary	Secondary	Avg. time step runtime [s] [a]	Max. time step runtime [s] [a]
60	0	0.0	4.476	4.289	3.643e-04	5.991e-04
60	0	0.2	4.709	4.347	3.830e-04	9.775e-04
60	0	1.0	5.078	3.822	3.630e-04	8.943e-04
60	0	$\sigma(T)$	4.743	4.320	3.742e-04	8.216e-04
120	0	0.0	4.476	1.078	3.734e-04	1.022e-03
120	0	0.2	4.812	0.880	3.647e-04	6.154e-04
120	0	1.0	5.202	0.527	3.651e-04	5.813e-04
120	0	$\sigma(T)$	4.743	0.983	3.690e-04	7.968e-04
60	120	0.0	-1.446	-1.446	1.063e-03	1.828e-03
60	120	0.2	5.179	-0.708	1.065e-03	1.820e-03
60	120	1.0	4.851	-0.158	1.031e-03	1.807e-03
60	120	$\sigma(T)$	4.627	-0.759	1.107e-03	1.565e-03
100	120	0.0	-0.037	-1.647	1.786e-03	3.057e-03
100	120	0.2	4.958	-1.574	2.268e-03	3.416e-03
100	120	1.0	4.846	-0.463	1.978e-03	3.006e-03
100	120	$\sigma(T)$	5.019	-1.181	1.861e-03	3.154e-03

[a] Times are given for AMD Ryzen 7 5800X 8-Core CPU.

Comparing the first two parameter sets, where N_{ext} is set to 0, an increase in N_{int} from 60 to 120 generally leads to a marked reduction in secondary overshoot. The primary overshoot stays similar both for $N_{int} = 60$ and $N_{int} = 120$. And, indeed, when analysing the system responses shown in Figures 5.29 and 5.30, it is evident that for a regular heat flux pattern $N_{int} = 60$ cases maintain a constant overshoot for each cycle. Whereas $N_{int} = 120$ cases induce the reduction in overshoot at each cycle thus steadily bringing the maximum temperature closer to the set point. This suggests that reasonably higher N_{int} value contributes to better system stability. On the other hand, the system responses to heat flux in the form of the Filtered Gaussian noise prove to be an exception to this observation (Figures C.42 and C.43). The unexpected heat flux increase after $t = 200$ induces similar system responses from both $N_{int} = 60$ and $N_{int} = 120$ cases.

A notable difference in the system response could be observed when analysing cases involving non-zero N_{ext}. With the added extrapolation the PD-ML DT is able to react quicker to the changes in the heat flux and hence reduce the overshoot faster. The secondary overshoot becomes negative for non-zero N_{ext} (Table 5.14), meaning that the

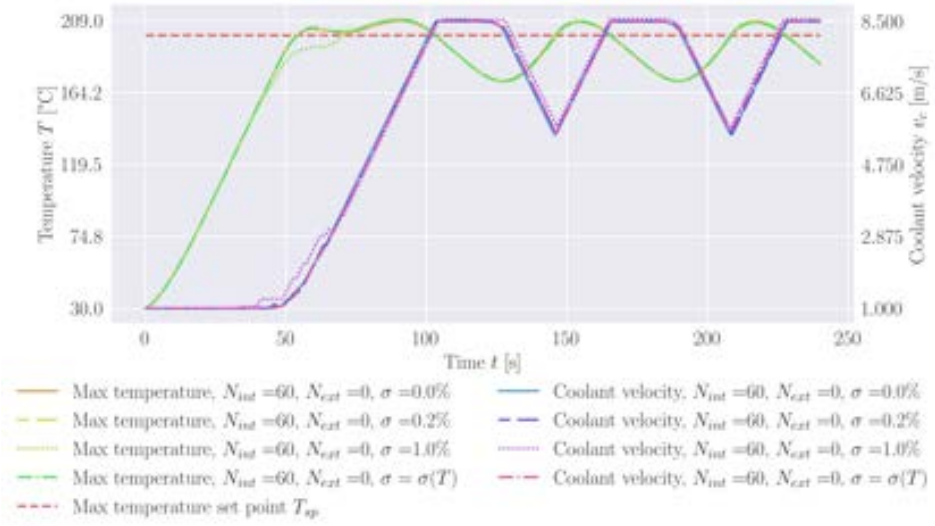


FIGURE 5.29: PD-ML DT system response Sine wave 1 (Table 5.9 and Figure 5.21) for N_{int} equal to 60 and N_{ext} equal to 0.

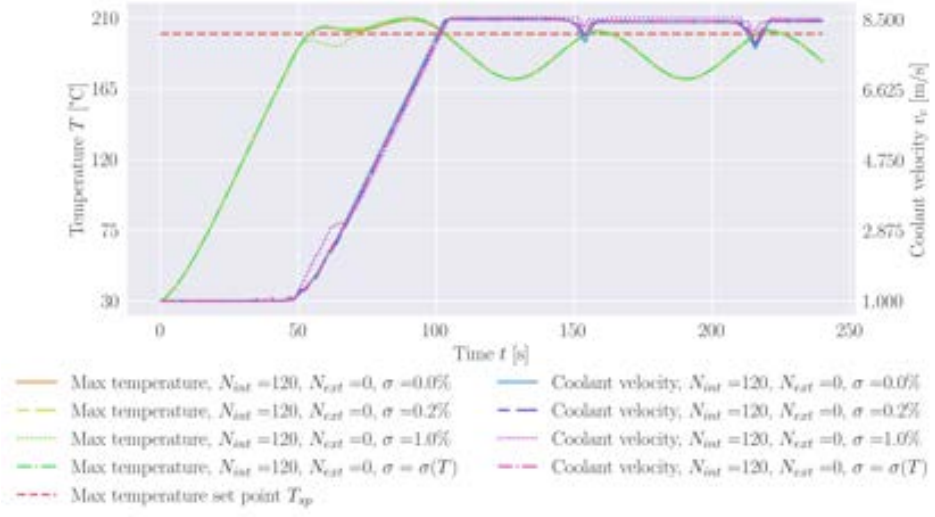


FIGURE 5.30: PD-ML DT system response Sine wave 1 (Table 5.9 and Figure 5.21) for N_{int} equal to 120 and N_{ext} equal to 0.

maximum temperature stays below the set point at all the times after the initial heat flux transition from linearity. This observation is supported by the system responses shown in Figures 5.24c, 5.25c, 5.26c, and 5.27c. Most importantly, unlike for cases with zero N_{ext}, this is true even for the Filtered Gaussian noise heat flux, which means that the N_{ext} introduction improves the control when encountering random heat flux changes (Figure 5.27c). On the contrary, there is no significant improvement in the primary overshoot when analysing cases with non-zero noise percentage.

For PD-ML DT, the noise-induced disruption to the coolant velocity is noticeable, especially when comparing the results from the non-zero noise cases with the zero noise

scenarios (Figures 5.24c, 5.25c, 5.26c, and 5.27c). When there is no noise, the coolant velocity changes are sharp, only limited by the imposed coolant acceleration. Noise substantially smooths out the velocity shape making the transitions more gradual. At zero noise level, both primary and secondary overshoots tend to be negative, whereas the primary overshoots tend to become positive once the noise is introduced. On the other hand, the secondary overshoots are influenced by the noise level only slightly.

However, the impact of noise has a different effect on PD-ML DT when N_{ext} is equal to 0, i.e. when there no extrapolation is present, as it is shown in Figure 5.30. The coolant velocity pattern over time stays approximately the same, for these cases, the noise induces approximately parallel translation to the velocity's shape. The noise does not introduce significant changes neither in the primary nor in the secondary overshoots.

Finally, Table 5.14 lists the typical time step runtimes achieved by PD-ML DT. Unlike with FE DT, there is no solution construction stage, hence the run time is comprised only of the time it takes to adjust the coolant velocity, i.e. control time. The time step run time is vanishingly small as is common for the trained NNs models.

5.5 Summary

This chapter investigates temperature monitoring and control using three DT approaches: a FE DT, with and without ROM, and a PD-ML DT. The forward FE model is employed as a virtual experiment to ensure a consistent and physically grounded basis for controller tuning and performance evaluation. Particular emphasis is placed on achieving near real-time feasibility while maintaining sufficient accuracy for closed-loop temperature control.

For the FE DT frameworks, classical PID control is adopted and tuned using the ZN and AH methods. The controller is designed to operate at a relatively coarse time step, dictated by the computational requirements of real-time FE-based solution construction. It is demonstrated that this time step preserves sufficient solution accuracy and that stable closed-loop control can be achieved when the PID gains are explicitly tuned at this temporal resolution. Standard ZN- and AH-tuned parameters are found to induce oscillatory responses and relatively large overshoots under constant heat flux conditions. By modifying the integral time constant to produce a more conservative

controller, overshoot and oscillations are significantly reduced or eliminated, while respecting practical actuator constraints such as coolant acceleration limits. The modified ZN-tuned controller is found to provide the most robust and stable performance and is therefore adopted for subsequent analyses.

The PD-ML DT approach replaces the explicit solution construction step with NN-based estimators for heat flux and coolant velocity. Extensive datasets generated from steady-state FE simulations enable accurate training and validation of both NNs. The heat flux network achieves high predictive accuracy, while the coolant network exhibits lower but still acceptable performance. These results confirm that the learned models are sufficiently reliable to support closed-loop control decisions within the PD-ML DT framework.

Control tuning for the PD-ML DT focuses on the parameters governing heat flux trend estimation, namely the interpolation window length N_{int} and the extrapolation horizon N_{ext}. When extrapolation is disabled $N_{ext} = 0$, system overshoot remained positive but insensitive to the choice of N_{int}, with differences primarily affecting the speed at which the coolant velocity reached steady state. Introducing extrapolation led to consistently negative overshoots, ensuring that the maximum temperature remained below the set point. However, this benefit came at the cost of increased oscillations in the constructed heat flux and coolant velocity, particularly for larger extrapolation horizons. While extrapolation enabled earlier and faster control action, it also introduced fluctuations that prevented smooth convergence to steady state. Attempts to reconcile extrapolated and non-extrapolated behaviour by conditionally disabling extrapolation based on heat flux variation did not improve performance and instead resulted in larger overshoots.

A comparative analysis of overshoot behaviour reveals clear differences between the investigated approaches. FE DT with ROM often achieves the lowest primary overshoots at low to moderate noise levels and can even produce undershoot. However, it is more susceptible to noise, with secondary overshoots frequently exceeding primary ones, indicating slower convergence to the temperature set point. In contrast, FE DT without ROM exhibits more stable and predictable behaviour, typically maintaining similar primary and secondary overshoots and showing gradual convergence over time, albeit at the cost of significantly longer runtimes that may hinder real-time applicability.

The inclusion of ROM substantially reduces computational cost, with runtimes decreasing by a factor of approximately 6-9 compared to the full FE DT. This efficiency gain is achieved through system size reduction and less frequent updates of convection-related terms, but it also increases sensitivity to noise, leading to larger localised errors and stronger overshoot growth under noisy conditions. Consequently, the ROM-based approach presents a clear trade-off between computational efficiency and robustness.

The PD-ML DT demonstrates superior responsiveness and robustness. While it may exhibit larger primary overshoots in some cases, it consistently achieves the lowest secondary overshoots and the fastest convergence to the set point across most scenarios. Its performance remains relatively insensitive to noise, and the inclusion of extrapolation enables anticipatory control, further reducing overshoot under irregular and stochastic heat flux variations.

From a computational standpoint, the PD-ML DT offers near-instantaneous runtimes after training, as control actions rely solely on NN inference rather than solution reconstruction, making it well suited for real-time applications. However, this advantage comes with substantial upfront costs for data generation and training, and retraining is required if the control objective changes. By contrast, FE DT provides a detailed physics-based representation without prior training and allows flexible adaptation to alternative control objectives, while remaining compatible with a wide range of control strategies beyond the PID formulation used in this work.

Chapter 6

Conclusions and future work

6.1 Conclusions

This thesis set out to develop and evaluate DT methodologies for monitoring and controlling the thermal behaviour of samples in the HIVE fusion energy experimental facility. Two distinct approaches are investigated. The first is a FE DT, which captured thermal responses under different loading scenarios and enabled FE-driven temperature monitoring and control. The second is a PD-ML DT, designed to provide temperature monitoring and control through the application of NNs. A comparative assessment of these methods is carried out, focusing on accuracy in terms of control error, computational efficiency with respect to real-time or near-real-time performance, and robustness under uncertainties such as measurement noise. Through this study, valuable insights are gained into the trade-offs between FE DT and PD-ML DT, offering guidance for selecting appropriate strategies for future DT applications in thermal systems.

FE DT is founded upon the FE-based solution construction process, where a full thermal solution is constructed from a sparse number of measurements. This solution construction process is coupled with a PID controller in order to enable the control of temperature field using the information recovered from the constructed solution. The solution is constructed by minimising a loss vector consisting of three components: the residual, measurement, and regularisation (smoothing). Two variations of FE DT are explored. The first one does not use any ROM, and the second one combines the solution construction process with the thermal eigenvalue-based ROM.

PD-ML DT employs a combination of two NNs in order to achieve control. The first one is the heat flux NN, the goal of which is to estimate the heat flux from the provided temperature measurements and the current value of the coolant velocity. The second NN is called the coolant NN; as an input it takes the estimated heat flux and the set point temperature and as an output it provides the coolant velocity for the future time step. Both NNs are trained using a steady-state data, which significantly reduces the training and validation time and simplifies the whole process. In order to compensate for the fact the transient data is not used for training, the heat flux extrapolation is added to the algorithm in order to estimate the future heat flux value. This future value is used as a part of the range of estimated heat fluxes to choose the maximum value from; the maximum value is used as an input to the coolant NN.

The analysis of overshoot behaviour reveals clear differences between the methods. FE DT with ROM frequently achieves the lowest primary overshoots, particularly at low or moderate noise levels. It even produces undershoots in certain cases. However, it is also the most susceptible to degradation at higher noise levels, with secondary overshoots often exceeding primary ones, suggesting slower convergence to the maximum temperature set point. By contrast, FE DT without ROM demonstrates more stable and predictable performance, generally maintaining overshoots at comparable levels between primary and secondary phases and showing gradual convergence to the set point over time. Although this method avoids the excessive secondary overshoots observed with ROM, its runtimes are significantly longer, which may hinder real-time applicability.

The introduction of ROM provides a substantial reduction in computational cost, with runtimes reducing by a factor of 6 to 9 compared with the full FE DT. This efficiency gain is achieved by reducing the system size and updating convection-related terms less frequently. However, the simplifications introduce also increased sensitivity to noise, leading to higher localised errors and a stronger correlation between noise level and overshoot growth. This trade-off highlights an important limitation of the ROM-based approach: while computationally efficient, its robustness is compromised under noisy measurement conditions.

The PD-ML DT approach displays distinct advantages in terms of responsiveness and robustness. While it tends to exhibit larger primary overshoots than FE DT in some cases, its secondary overshoots are consistently the lowest of the three approaches, with

11 out of 16 cases showing the best performance. This indicates that PD-ML DT achieves the fastest convergence of the maximum temperature to the set point and maintains the swiftest rate of response. Importantly, the model proved less sensitive to noise than the FE DT approaches, with overshoot values fluctuating only slightly around stable levels for most noise conditions. The inclusion of extrapolation improves its performance, allowing it to anticipate changes in the applied heat flux and thereby reduce overshoots, even under irregular and random loading scenarios such as Gaussian noise.

From a perspective of computational speed, PD-ML DT offered a decisive advantage. As no solution reconstruction is required, runtimes after training are vanishingly small, consisting only of the control adjustments executed by trained NNs. This near-instantaneous performance makes PD-ML DT particularly well-suited for real-time applications where both accuracy and speed are critical. On the other hand, prior to the experiment, a significant time needs to be allocated for data generation together with NNs training and validation. Moreover, if the objective of the experiment shifts from controlling the maximum temperature to regulating another property of the temperature field, the NNs would need to be retrained.

Contrary to PD-ML DT, FE DT is able to provide an valuable detailed physics-based representation throughout the experiment. It does not require time-consuming data generation or training. Furthermore, once the solution construction process is set up, changing the experiment's objective is straightforward, as a different property of the temperature field can simply be extracted from the constructed solution. Also, it should be noted that although a standard PID controller is used in conjunction with the FE-based solution construction in this work, it is not intrinsically integrated into the overall FE DT workflow. This means it can be relatively easily substituted with other forms of control, such as RL [74], or with hybrid approaches that combine PID control and RL [98].

Overall, the aims and objectives set out at the beginning of this thesis have been successfully achieved. The key outcomes of the work can be directly linked to the stated objectives as follows:

1. **FE DT development.** A FE DT of a representative component prototype was successfully constructed and demonstrated. The framework enabled physics-based

temperature field construction, heat flux estimation, and closed-loop thermal control under a wide range of loading scenarios. This is representative of experiments conducted at the HIVE facility. Near real-time operation was achieved through appropriate temporal discretisation and, where applicable, ROM, while maintaining sufficient accuracy for control purposes.

2. **PD-ML DT development.** A PD-ML DT was developed and validated to enable data-driven temperature monitoring and control. Trained NNs provided accurate heat flux and coolant velocity estimates, allowing control actions to be computed without explicit solution reconstruction. After training, the approach achieved near-instantaneous runtimes and demonstrated strong responsiveness and robustness to measurement noise. Particularly, it exhibited rapid convergence of the maximum temperature to the set point.

3. **Comparative evaluation of DT approaches.** The FE DT with and without ROM as well as PD-ML DT were systematically compared in terms of control accuracy, computational efficiency, and robustness to uncertainty. The FE DT without ROM exhibited the most stable and predictable behaviour but incurred higher computational cost. The inclusion of ROM significantly improved efficiency at the expense of increased noise sensitivity. The PD-ML DT delivered the fastest convergence and highest computational efficiency, though at the cost of substantial upfront data generation and retraining requirements when control objectives change.

Together, these achievements demonstrate the feasibility of both methodologies for DT of thermal systems and provide a solid foundation for future work aimed at extending their capabilities and integration into more complex experimental scenarios.

6.2 Future work

Future work should primarily focus on broadening the scope of both DT approaches beyond purely thermal behaviour. A natural next step is to extend the methodology to simulations involving thermo-mechanical responses, where stresses arise as a result of thermal loads. This would enable the monitoring and control of both temperature

fields and stress distributions within the sample, providing a more comprehensive representation of system behaviour. Achieving this will require incorporating displacement measurements alongside temperature measurements and developing control strategies that act on both variables simultaneously.

Integrating the DT approaches with physical experiments can be a complex process, but alternative avenues for their evaluation exist. The Python Validation Engine (Pyvale) project [100] seeks to create a virtual engineering laboratory that facilitates sensor setup analysis, optimises experimental design, and supports the calibration and validation of simulations. One of its primary objectives is to enable the planning of validation experiments without requiring direct access to experimental facilities. Although Pyvale cannot fully replace real experimental data, it can generate computational measurements that closely replicate experimental observations, including data from various imaging sensors such as digital image correlation and infrared thermography. These simulated measurements can provide invaluable support for evaluating the DT approaches developed in this work.

In addition to thermo-mechanical coupling, future studies should also consider the integration of these approaches into more complex facilities. Whereas the HIVE facility was used as a representative case in this work, the CHIMERA facility will introduce new challenges, including pulsed magnetic loads and magnetohydrodynamic effects. This presents an opportunity to test whether the proposed DT frameworks can be adapted to account for transient electromagnetic phenomena. Assessing the suitability and limitations of these approaches under such multi-physics conditions will provide valuable insights into their generalisability and robustness.

Finally, as it is mentioned previously, although a standard PID controller is employed in conjunction with the FE-based solution construction in this work, it is not intrinsically embedded within the FE DT workflow. As such, future research could explore replacing it with alternative control strategies, such as previously mentioned RL [74], or various hybrid methods [98].

Appendix A

Transformer- and Long Short-Term Memory-based Machine Learning methods

A.1 Introduction

Computational engineering encompasses transient inverse analysis as a specialized research domain focused on resolving time-dependent inverse problems, particularly solution reconstruction from sparse measurements. Inverse problems fundamentally contrast with conventional forward problems. Transient forward problems typically exhibit well-posed characteristics, enabling numerical solution computation with specified accuracy when provided with suitable initial and boundary conditions [23]. Conversely, certain transient inverse problems require reconstructing complete domain data from available sparse observations or measurements. In contrast to standard forward problems, inverse problems are ill-posed [23]. Thus, solution non-uniqueness and inherent instability create significant modelling challenges that often render problems computationally intractable, particularly for complex scenarios.

To address limitations inherent in traditional transient solution reconstruction approaches, this study investigates two ML model types for obtaining solutions within acceptable uncertainty bounds. The emergence of advanced GPUs has popularised ML across scientific and engineering disciplines due to its adaptability and capacity for processing

large datasets within practical time frames. Recently, Transformer-based ML models have gained prominence through exceptional performance in Natural Language Processing (NLP) applications, exemplified by the renowned ChatGPT chatbot [101], and various time-series analysis challenges [102–106]. These achievements in temporal sequence transformation suggest their optimal suitability for transient inverse analysis applications. This study evaluates novel complex Transformer-based models against simple LSTM models [5], which represent the traditional ML approach for sequential data processing [107], for reconstructing transient 1D and 2D thermal fields.

This research demonstrates the applicability of increasingly prevalent Transformer-based models for transient solution reconstruction. This is a challenge common in industrial applications. They are compared with the more established ML models such as LSTM.

A.2 Background

A.2.1 Transient inverse problem

A transient forward problem can be broadly defined as calculating time-dependent consequences of specified causes through an appropriate physical system model. Solving transient forward problems to achieve complete domain solutions via conventional methods requires prescribed system parameters, boundary conditions, and initial system conditions. In contrast, transient inverse problems encompass two distinct categories. The first one is identifying system parameters based on observed causes and effects. This represents the classical inverse problem definition [23]. The second category is identifying causes from observed time-dependent effects. This essentially involves utilizing available sparse domain data (observations or measurements) to reconstruct complete solution datasets.

Numerous methodologies have been developed to address inverse problems throughout the years; nevertheless, historically, greater emphasis has been placed on first-type inverse problems. Traditional approaches encompass functional analytic regularisation and statistical regularisation, with Bayesian inversion serving as the most recognised example [23, 24]. Search-and-optimization methodologies represent an alternative approach for obtaining transient problem solutions. For example, [108] addressed the inverse 2D

natural convection problem in steady state employing a PSO algorithm [109]. Meanwhile, [24] and [35] offer comprehensive surveys of diverse methodologies employed for inverse problem resolution.

This research concentrates on second-type transient inverse problems, specifically transient thermal field reconstruction. Physical experimental data likely represents the most prevalent source of sparse domain observations. Fusion energy research facilities, engineered to evaluate component performance under extreme fusion reactor conditions, routinely face transient inverse problems arising from sparse experimental datasets. The HIVE experimental facility [18] exemplifies this challenge: inverse analysis must be conducted to reconstruct complete temperature fields from temperature measurements obtained by limited thermocouples.

A.2.2 Long Short-Term Memory

Sequential data processing applications widely employ RNNs across numerous tasks [107]. RNNs are generally composed of multiple standard recurrent cells (Figure A.1), whose mathematical formulation can be expressed as:

$$\begin{aligned} h_t &= \sigma\left(W_h h_{t-1} + W_x x_t + b\right) \\ y_t &= h_t \end{aligned} \tag{A.1}$$

where y, x, and h represent the output, input, and the hidden state (containing the recurrent information), respectively. W_x and W_h are the cell weights, and b is the cell bias.

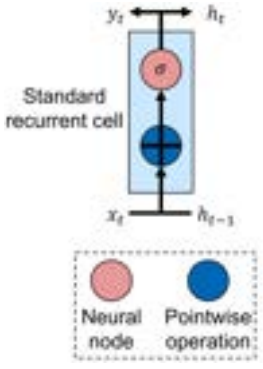


FIGURE A.1: RNN standard cell.

Nevertheless, RNNs constructed with standard recurrent cells commonly encounter training challenges due to vanishing or exploding gradient problems between temporally distant inputs [110]. To resolve these long-term dependency issues, the LSTM cell, an RNN variant, was introduced over twenty years ago [5] and has been successfully implemented across diverse sequential applications, including speech recognition [111], trajectory prediction [112], and remaining useful life prediction [113], among others. The initial LSTM cell architecture incorporates only input and output gates (Figure A.2) and can be described by the following mathematical expressions:

$$
\begin{aligned}
i_t &= \sigma\left(W_{ih} h_{t-1} + W_{ix} x_t + b_i\right) \\
\tilde{c}_t &= \tanh\left(W_{\tilde{c}h} h_{t-1} + W_{\tilde{c}x} x_t + b_{\tilde{c}}\right) \\
c_t &= c_{t-1} + i_t \odot \tilde{c}_t \\
o_t &= \sigma\left(W_{oh} h_{t-1} + W_{ox} x_t + b_0\right) \\
h_t &= o_t \odot \tanh\left(c_t\right)
\end{aligned}
\tag{A.2}
$$

where c is a cell state. W_{ih}, W_{ix}, $W_{\tilde{c}h}$, $W_{\tilde{c}x}$, W_{oh}, W_{ox} are the weights, b_i, $b_{\tilde{c}}$, b_o are the biases. The input gate controls which information gets stored within the cell state, while the output gate selects which information from the cell state should be used for the output.

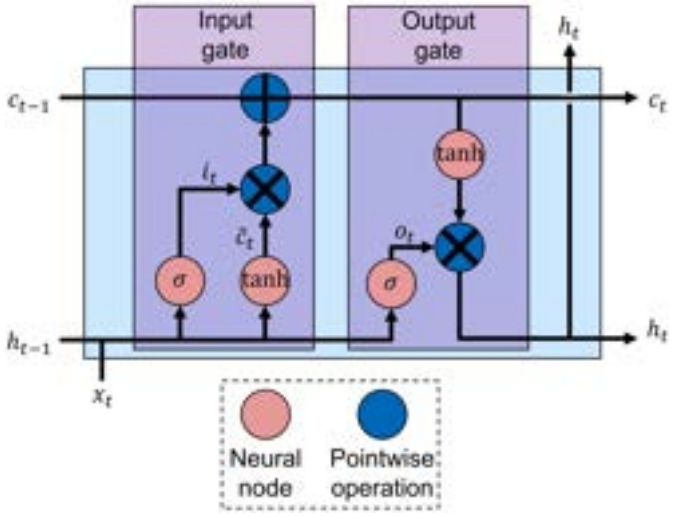


FIGURE A.2: Initial LSTM cell [5].

This work employs an LSTM network utilising a modified variant of the original LSTM cells. The modification includes a forget gate that determines which information should be removed from the cell state [114]. Figure A.3 illustrates the modified LSTM cell,

with the cell represented by the subsequent equations:

$$
\begin{aligned}
f_t &= \sigma\left(W_{fh}h_{t-1} + W_{fx}x_t + b_f\right) \\
i_t &= \sigma\left(W_{ih}h_{t-1} + W_{ix}x_t + b_i\right) \\
\tilde{c}_t &= \tanh\left(W_{\tilde{c}h}h_{t-1} + W_{\tilde{c}x}x_t + b_{\tilde{c}}\right) \\
c_t &= f_t \odot c_{t-1} + i_t \odot \tilde{c}_t \\
o_t &= \sigma\left(W_{oh}h_{t-1} + W_{ox}x_t + b_0\right) \\
h_t &= o_t \odot \tanh\left(c_t\right)
\end{aligned} \tag{A.3}
$$

where f_t is a forget gate value.

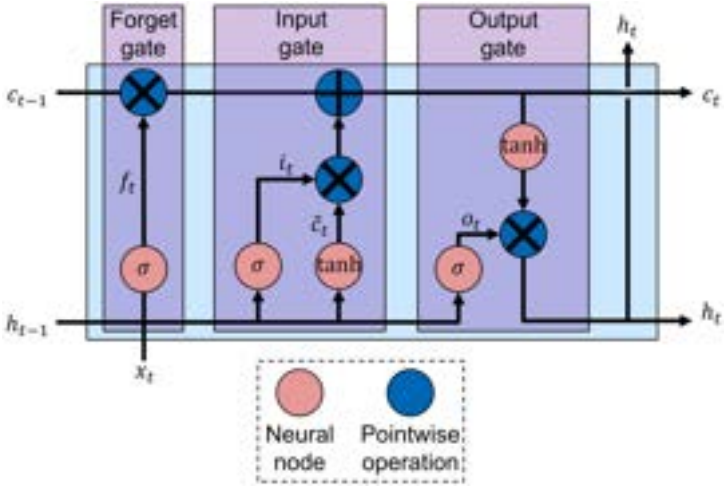


FIGURE A.3: LSTM cell modified with forget gate.

It is important to note that within this sub-section, t represents the t^{th} time step in a sequence, following standard RNN notation; conversely, t denotes the total sequence length in following subsections.

A.2.3 Transformers

Google's research team developed the classic Transformer model in 2017, successfully implementing it for NLP applications including natural language generation and machine translation [6]. Currently, Transformers are regarded as the optimal models for diverse NLP tasks [115], with Chat Generative Pre-trained Transformer (ChatGPT) [101], an

Artificial Intelligence (AI) chatbot, representing perhaps the most renowned demonstration of Transformer capabilities. Since NLP fundamentally constitutes sequence-to-sequence transformation tasks, where model inputs and outputs typically consist of ordered element series, Transformers prove well-suited for time-series prediction applications. Recently, numerous Transformer-based models have been effectively applied to various temporal tasks, including weather, electricity consumption, and exchange rate forecasting. Wen et al. [116] examined state-of-the-art Transformer-based models for time-series analysis, while Lim and Zohren [106] surveyed Deep Learning (DL) methodologies for time-series forecasting. These surveys may not be entirely comprehensive since Transformers' growing popularity drives annual creation of new Transformer-based models and variations of existing architectures. Consequently, tracking all developments proves challenging, particularly as advances occur across diverse research domains, from weather prediction [117] to computer vision [118]. This cross-domain development complicates model comparison processes, as testing occurs on research-specific datasets, making optimal engineering application selection less apparent.

Transformer-based models' principal benefit stems from eliminating recurrent connections present in RNNs [5]. Removing recurrent connections substantially reduces training duration and enhances parallelisation capabilities, advantageous for GPU-based model training. Additionally, these models excel at identifying long- and short-term sequence dependencies, potentially improving accuracy [6].

Figure A.4 presents a simplified transformer block representation. This paper omits detailed Transformer operational explanations. Readers should consult Bloem [119] and Vaswani et al. [6] for comprehensive self-attention and classic Transformer introductions. However, the following subsection provides a concise classic self-attention overview.

A.2.4 Self-attention

Self-attention operations, or their variants, form the fundamental component of all Transformer-based models. Figure A.5 illustrates the classic self-attention mechanism employed in the original Transformer [6]. The system receives t input vectors of dimension K and produces a distinct set of t output vectors with identical dimensions. These input vectors are utilised to compute queries, keys, and values through corresponding

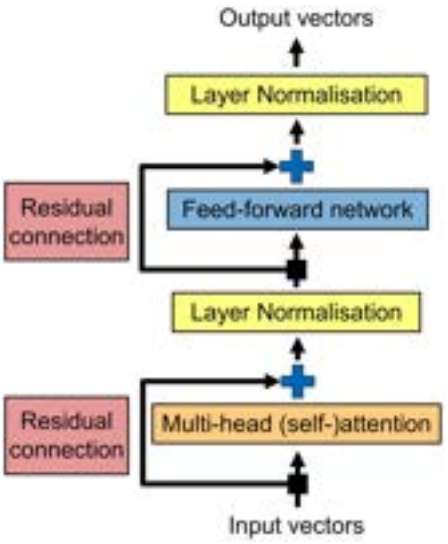


FIGURE A.4: General transformer block.

query, key, and value weight matrices:

$$\begin{aligned} \{\boldsymbol{q_i}\} &= [\boldsymbol{W_q}]\,\{\boldsymbol{x_i}\} \\ \{\boldsymbol{k_i}\} &= [\boldsymbol{W_k}]\,\{\boldsymbol{x_i}\} \\ \{\boldsymbol{v_i}\} &= [\boldsymbol{W_v}]\,\{\boldsymbol{x_i}\} \end{aligned} \tag{A.4}$$

where $\{\boldsymbol{k_i}\}$, $\{\boldsymbol{q_i}\}$, $\{\boldsymbol{v_i}\}$, and $\{\boldsymbol{x_i}\}$ are i^{th} key, query, value, and input vectors, respectively. $[\boldsymbol{W_k}]$, $[\boldsymbol{W_q}]$, and $[\boldsymbol{W_v}]$ are the key, query, and value weight matrices, respectively.

Key $[\boldsymbol{K}]$, query $[\boldsymbol{Q}]$, and value $[\boldsymbol{V}]$ matrices could be derived by concatenating all key, query, and value vectors. The scaled $[\boldsymbol{W}]'$ is as follows:

$$[\boldsymbol{W}]' = \frac{[\boldsymbol{Q}]\,[\boldsymbol{K}]^T}{\sqrt{K}} \tag{A.5}$$

It is normalised as follows:

$$[\boldsymbol{W}] = \text{softmax}\left([\boldsymbol{W}]'\right) \tag{A.6}$$

where $softmax$ is a softmax function [42].

Lastly, the output $[\boldsymbol{Y}]$ is obtained:

$$[\boldsymbol{Y}] = [\boldsymbol{W}]\,[\boldsymbol{V}] \tag{A.7}$$

$[\boldsymbol{Y}]$ consists of output vectors $\{\boldsymbol{y_i}\}$.

Key, query, and value weight matrices represent learnable parameters. Transformer-based architectures typically utilise multiple parallel self-attention operations (Figure A.6), enabling more effective feature extraction from time series data. Self-attention mechanisms consist of matrix multiplication operations, which proves advantageous since these can be executed using highly optimised and efficient matrix multiplication implementations.

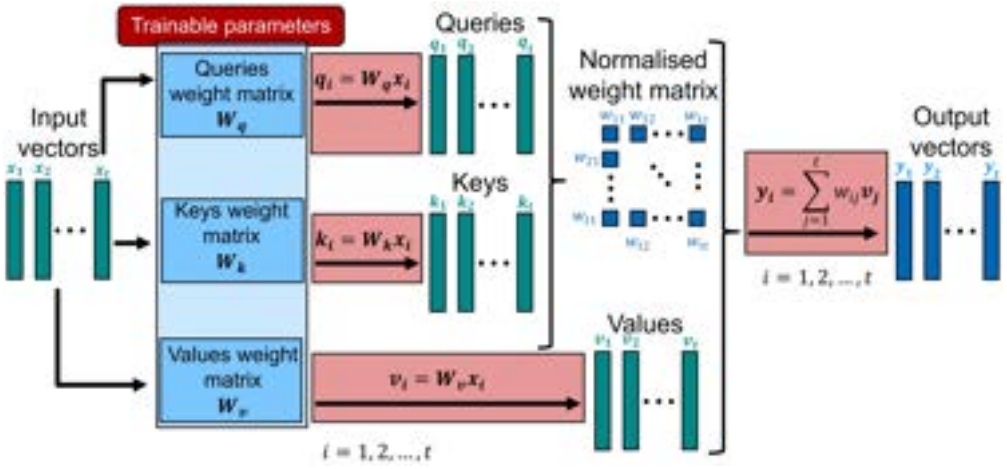


FIGURE A.5: Conventional self-attention[6].

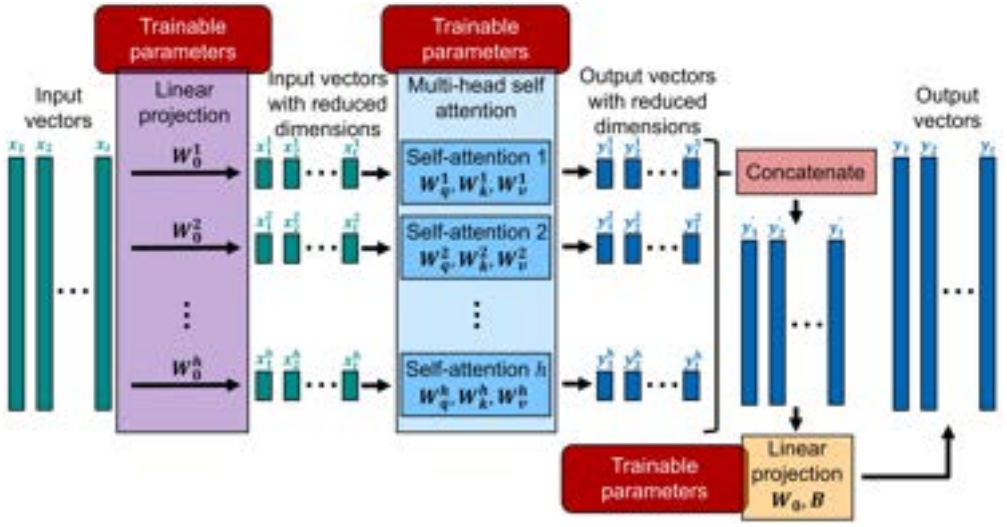


FIGURE A.6: Conventional multi-head (self-)attention operation.

A.3 Methodology

A.3.1 Selected models

This work applies an LSTM model alongside four Transformer-based architectures to 1D and 2D heat conduction problems. Table A.1 summarises these models, with the classic Transformer subsequently referred to simply as the Transformer. The self-attention mechanism employed in the Transformer [6] is detailed in Sub-section A.2.4. Informer [103], Autoformer [104], and FEDformer [105] were designed to enhance Transformer efficiency through complexity reduction and architecture optimization specifically for time-series applications (Table A.1).

TABLE A.1: All models analysed in this work.

Model type	LSTM	Transformer	Informer	Autoformer	FEDformer
Original purpose	Sequential data	Linguistic data	Temporal data	Temporal data	Temporal data
Self-attention type	N/A	Classic self-attention	Sparse self-attention	Auto-Correlation	Discrete Fourier Transform (DFT)

A.3.2 Model structure

Table A.2 displays the hyperparameters employed for Transformer-based models in this work. The architecture utilises three layers comprising two encoder layers and one decoder layer; input and output time-series length (sequence length hyperparameter) is evaluated at three values: 25, 50, and 100. For enhanced clarity, sequence length l is designated as the prediction window size, where the prediction window represents the temporal interval over which model predictions are generated. Hyperparameters No. 4-9 adopt values established in existing literature [6, 103–105].

Table A.3 shows the hyperparameters applied to the LSTM model in this research. A single layer configuration is implemented with model dimension fixed at 512 to align with Transformer-based architectures. Prediction window size l is examined at three values: 25, 50, and 100. Additionally, a feed-forward layer follows the LSTM layer to reshape outputs and enable direct temperature prediction across multiple time steps within a single inference operation.

TABLE A.2: Transformer-based model hyperparameters.

No.	Hyperparameter	Value(s)
1.	Encoder layers	2
2.	Decoder layers	1
3.	Prediction window size (sequence length) l	25, 50, and 100
4.	Model dimension d_{model}	512
5.	Multi-head (self-)attention heads h	8
6.	Feed-forward network dimension	2048
7.	Dropout rate	0.05
8.	Activation function	GELU
9.	Attention factor	3

TABLE A.3: LSTM model hyperparameters.

No.	Hyperparameter	Value(s)
1.	LSTM layers	1
2.	Prediction window size (sequence length) l	25, 50, and 100
3.	Model dimension d_{model}	512
4.	Dropout rate	0.05

A.3.3 Training

Adam optimiser [120] is employed for training all Transformer-based models. Furthermore, a warm-up phase is implemented for the learning rate since it has been demonstrated to enhance the training process of Transformer-based architectures [121]. The learning rate throughout the warm-up phase is defined by:

$$lr(it) = \frac{it}{T_{warmup}} lr_{max} \quad \text{for} \quad it \leq T_{warmup} \tag{A.8}$$

where it and lr are iteration number and learning rate, respectively.

The learning rate with the warm-up is calculated as follows:

$$lr(it) = \frac{lr(it-1)\sqrt{T_{warmup}}}{\sqrt{it}} \quad \text{for} \quad it > T_{warmup} \tag{A.9}$$

Training of Transformer-based models employs lr_{max} and T_{warmup} values specified in Table A.4 over 500 epochs with a batch size of 32. Subsequently, the optimal configuration is chosen for each model type presented in Table A.1 based on Normalised Root

Mean Squared Error (NRMSE) as the evaluation metric:

$$NRMSE = \frac{RMSE}{u_{true\_mean}} \quad \text{and} \quad RMSE = \sqrt{\frac{\sum_{k=1}^{N_{total}} (u_{true\_k} - u_{pred\_k})^2}{N_{total}}} \tag{A.10}$$

where u_{true_mean} and u_{pred_k} are the true temperature mean and the k^{th} predicted temperature value, respectively. u_{true_k} and N_{total} are the k^{th} true temperature value and total data point number, respectively.

TABLE A.4: Learning rates used to train the models.

Option No.	Constant lr or with warm-up	lr_{max}	T_{warmup}
1.	Constant $lr = 1e^{-4}$	N/A	N/A
2.	With warm-up	$1e^{-3}$	4000
3.	With warm-up	$1e^{-3}$	2000
4.	With warm-up	$1e^{-3}$	500
5.	With warm-up	$5e^{-4}$	4000
6.	With warm-up	$5e^{-4}$	2000
7.	With warm-up	$5e^{-4}$	500

LSTM model training employs only the Adam optimiser [120] across 500 epochs. Training of all ML models discussed in this study utilises an NVIDIA A100 40GB GPU. Figure A.7 illustrates training convergence, displaying optimal configurations for Transformer, Informer, Autoformer, and FEDformer.

Fixed random seed initialisation is applied to all models to ensure result reproducibility. This methodology ensures consistent initial weights and stochastic processes within training algorithms across multiple runs. Therefore, performance variability from random initialisation is eliminated, enabling fair architectural comparison between models.

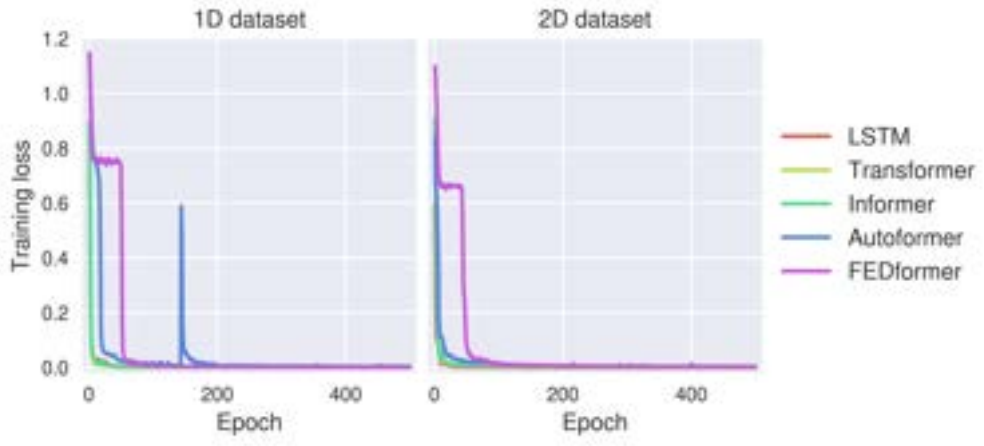


FIGURE A.7: Model convergence during the training process.

A.4 Results and discussion

A.4.1 1D transient heat conduction

The 1D transient heat conduction is defined as:

$$\frac{\partial u(x,t)}{\partial t} = \alpha \frac{\partial^2 u(x,t)}{\partial x^2} \tag{A.11}$$

where u and α are temperature and thermal diffusivity, respectively. α is equal to $8.6e^{-4}m^2/s$. ICs and BCs are:

$$\begin{aligned} &\text{Boundary Conditions: } u(x = x_A, t) = 255.372K \quad \text{and} \quad \frac{\partial u(x = x_B, t)}{\partial x} = 0 \\ &\text{Initial Conditions: } u(x, t = 0) = 272.039K \end{aligned} \tag{A.12}$$

The GT solution is generated using the FDM, which is implemented with the PyPDE Python package [122]. This GT is created on the grid displayed in Figure A.8 and is shown in Figure A.9. The simulation itself ran for a total of 1000s, with temperature readings taken every second. This dataset is strategically partitioned for the ML model: the first 700s are allocated for training, the next 100s for validation, and the final 200s for testing. The model's six input channels, which are randomly selected (Figure A.8), are used to provide temperature data. The model's task is to then predict the temperature values for the remaining 194 output channels (Figure A.10).

For training, validation, and testing, a single input-output sample is created by advancing the prediction window by one second. If the prediction window size, denoted by l, is 50s, the first window for the 200s used in testing would be from 1s to 50s. The next window would be from 2s to 51s, and so on. As a result, when $l = 50$, there are $200 - 50 + 1 = 151$ prediction windows for testing. For each of these windows, a predicted temperature matrix, $[\boldsymbol{U_{pred}}]$, and a true temperature matrix, $[\boldsymbol{U_{true}}]$, are created. Both matrices have dimensions of $((50 \cdot 151) \times N_{out})$, where$N_{out}$ represents the number of output channels, which is 194.

Therefore, for Eq. A.10, N_{total} can be calculated as:

$$N_{total} = (50 \cdot 151) \cdot N_{out} = (50 \cdot 151) \cdot 194 \tag{A.13}$$

Similar procedures are applied for the calculation for other values of l and N_{out}.

FIGURE A.8: The grid employed to produce the GT for 1D heat conduction.

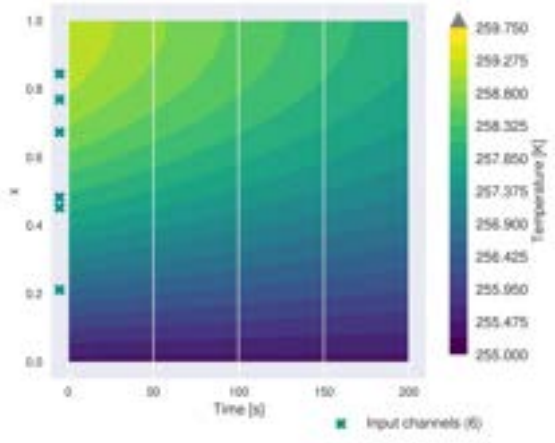


FIGURE A.9: The GT for 1D heat conduction produced utilising the FDM.

FIGURE A.10: ML model outline employed for the transient thermal field reconstruction.

Table A.5 presents the NRMSE for testing, which were calculated using Eq. A.10 with Eq. A.13, along with the training times. To visualise how these errors are distributed over time and space, four consecutive prediction windows are selected out of the 151 total windows for a window size of 50. These windows span from 1s to 50s, 51s to 100s, 101s to 150s, and 151s to 200s within the 200s used for testing. The top row of Figure A.11 shows the prediction error distribution defined by Eq. A.14, while the bottom row displays the prediction errors averaged at each time step using Eq. A.15.

$$PE_{i,n} = \frac{|u_{pred\_i,n} - u_{true\_i,n}|}{u_{true\_i,n}} \tag{A.14}$$

$$PE_i = \frac{\sum_{n=1}^{N_{out}} PE_{i,n}}{N_{out}} \tag{A.15}$$

where PE_i is a prediction error averaged at time step i; $u_{pred\_i,n}$, $PE_{i,n}$ is a prediction error at node n at time step i. $u_{pred\_i,n}$ and $u_{true\_i,n}$ are the predicted and true temperatures, respectively.

According to Table A.5, the LSTM model demonstrated the most efficient training, requiring the least amount of time for all three l values, in contrast to the FEDformer model, which took the longest to train. In terms of accuracy, the LSTM model had the lowest NRMSE for window sizes $l = 25$ and $l = 50$, a result corroborated by the error distributions in Figure A.11. While the Transformer model performed slightly better for $l = 100$ with the lowest NRMSE, the LSTM's result was very close, only 0.004% higher. The Autoformer model consistently underperformed, with an NRMSE over two times greater than the LSTM model's.

TABLE A.5: Testing errors and training times for the 1D heat conduction (Eq. A.10). The best results are highlighted in **green bold**, while the worst results are highlighted with a red underline.

Model type	LSTM	Transformer	Informer	Autoformer	FEDformer
Prediction window size $l = 25$					
NRMSE (Eq. A.10) [%]	**0.188**	0.189	0.192	0.497	0.362
Training time [min]	**35.9**	39.1	40.2	40.9	46.4
Prediction window size $l = 50$					
NRMSE (Eq. A.10) [%]	**0.189**	0.190	0.190	0.493	0.365
Training time [min]	**36.0**	38.5	39.8	42.4	52.3
Prediction window size $l = 100$					
NRMSE (Eq. A.10) [%]	0.196	**0.192**	0.195	0.468	0.371
Training time [min]	**36.3**	38.5	40.9	43.8	65.2

A.4.2 2D transient heat conduction

The linear 2D transient heat conduction is defined as:

$$\frac{\partial u(x,y,t)}{\partial t} = \alpha_x \frac{\partial^2 u(x,y,t)}{\partial x^2} + \alpha_y \frac{\partial^2 u(x,y,t)}{\partial y^2} \tag{A.16}$$

where α_x and α_y are the thermal diffusivities in x and y directions. α_x is equal to $13.9e^{-4}m^2/s$, α_y is equal to $3.3e^{-4}m^2/s$. The BCs and ICs are defined by Eq. A.17. Figure A.12 displays the domain's boundaries.

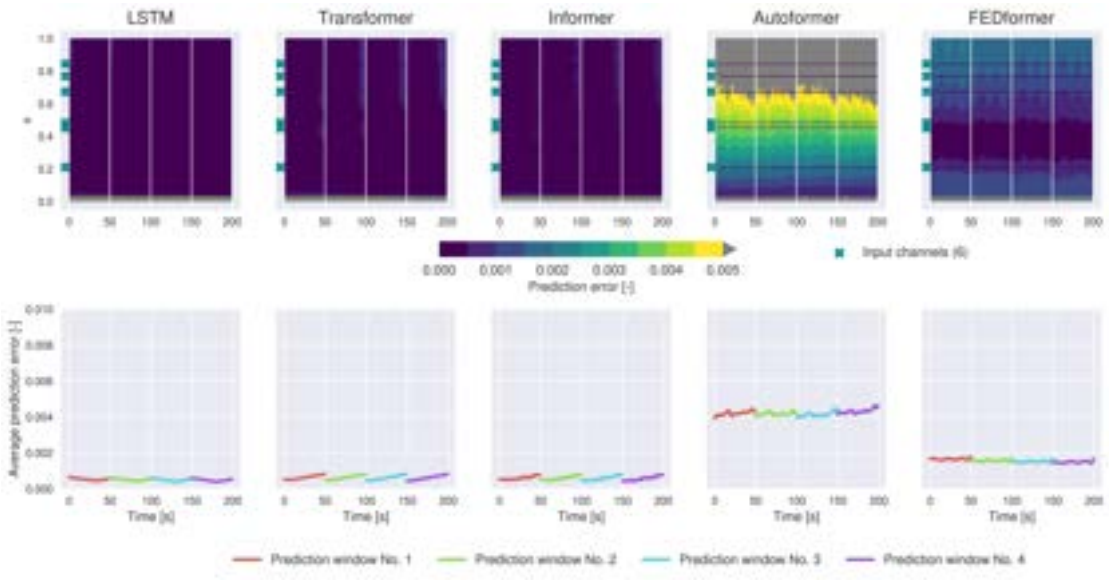


FIGURE A.11: Error distribution (Eq. A.14) and errors averaged at each time step (Eq. A.15) for the 1D heat conduction for models with $l = 50$. Four consecutive prediction windows are shown; the green crosses show input channels.

Boundary Conditions:

1. $u(x, y, t) = 255.372K \quad \text{for} \quad x, y \in [AB] \cup [BC]$
2. $\nabla u(x, y, t) = 0 \quad \text{for} \quad x, y \in [CD] \cup [DA]$

Initial Conditions: $u(x, y, t = 0) = 272.039K$ (A.17)

To establish GT, FEM is used within the open-source software Code_Aster [7]. The GT is created using the mesh in Figure A.12 and is presented in Figure A.13 for model testing. The simulation runs for 1000s, with temperature readings taken every second. This dataset is partitioned for ML: the first 700s for training, the following 100s for validation, and the last 200s for testing. The model receives temperature data from twelve randomly selected input channels and then predicts temperatures for 1142 output channels (Figures A.10 and A.12). The process for generating input-output samples by shifting the prediction window by one second is the same as in the 1D case (Sub-section A.4.1), as are the calculations for the total number of prediction windows and NRMSEs.

Table A.6 summarises the testing NRMSEs, derived from Eq. A.10 and Eq. A.13, and the corresponding training times. The prediction error distributions for a prediction window size of $l = 50$ are visualised in a manner similar to the 1D case (Sub-section A.4.1). The

top row of Figure A.14 displays the time-averaged error distribution using Eq. A.18, while the bottom row shows the errors averaged at each time step using Eq. A.15. It is important to note that the scales in Figure A.14 are different from those in Figure A.11.

$$PE_n = \frac{\sum_{i=1}^{200} PE_{i,n}}{200} \tag{A.18}$$

where PE_n is a error averaged at node n. The error distribution change with time is shown in Figure A.15.

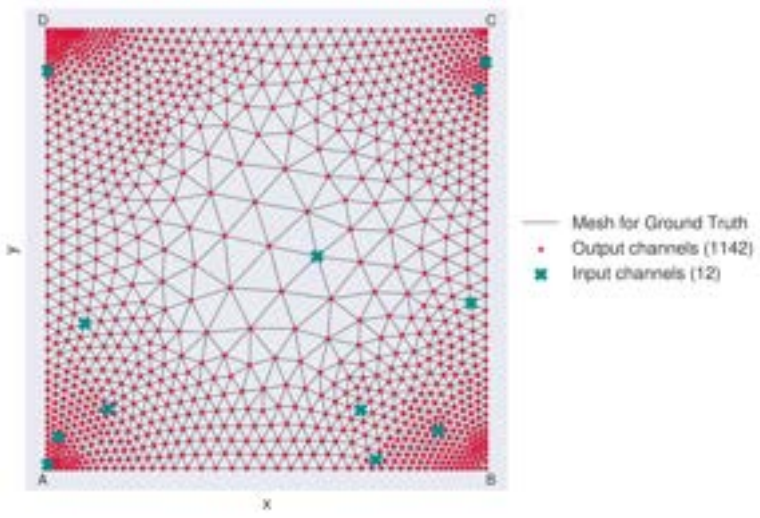


FIGURE A.12: The 2D mesh utilised to produce the GT for 2D transient heat conduction.

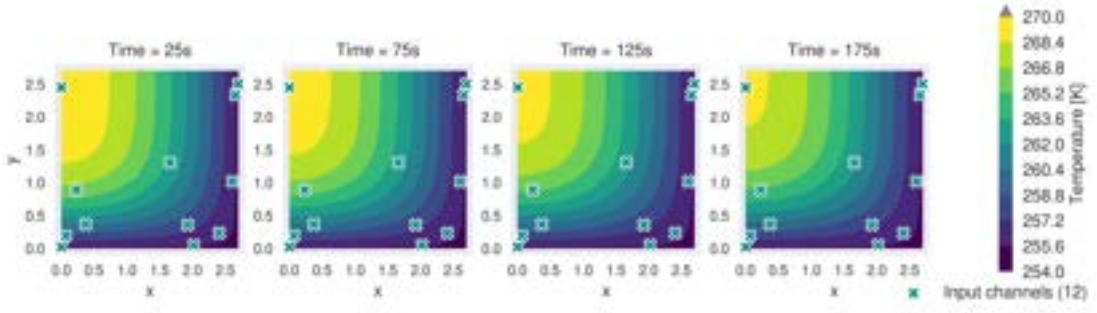


FIGURE A.13: The GT for 2D transient heat conduction problem produced utilising the FEM implemented in Code_Aster [7]. .

According to Table A.6, the FEDformer model had the best performance in terms of NRMSEs for all three l values, but this came at the cost of the longest training times. The LSTM model offered a more balanced trade-off, with NRMSEs only 0.1-0.2% higher than the FEDformer, but with a training speed that was 20-41% faster. The LSTM's

one-layer architecture also makes it simpler and more manageable for troubleshooting compared to the more complex FEDformer structure.

Surprisingly, all models, regardless of their architecture, produced similar prediction error distribution patterns, as shown in Figures A.14 and A.15. While this might be expected among the self-attention-based Transformer models, the fact that the LSTM model, which lacks this mechanism, also follows a similar pattern is unexpected. This observation could challenge the perceived superiority of Transformer-based models.

TABLE A.6: Testing errors and training times for the 2D heat conduction (Eq. A.10). The best results are highlighted in **green bold**, while the worst results are highlighted with a red underline.

Model type	LSTM	Transformer	Informer	Autoformer	FEDformer
Prediction window size $l = 25$					
NRMSE (Eq. A.10) [%]	2.197	2.208	2.218	2.543	**2.015**
Training time [min]	**37.9**	38.3	40.9	42.7	47.5
Prediction window size $l = 50$					
NRMSE (Eq. A.10) [%]	2.170	2.191	2.193	2.415	**2.015**
Training time [min]	39.7	**39.6**	41.5	44.2	67.0
Prediction window size $l = 100$					
NRMSE (Eq. A.10) [%]	2.127	2.155	2.157	2.430	**1.979**
Training time [min]	**42.0**	**42.0**	43.6	48.1	66.9

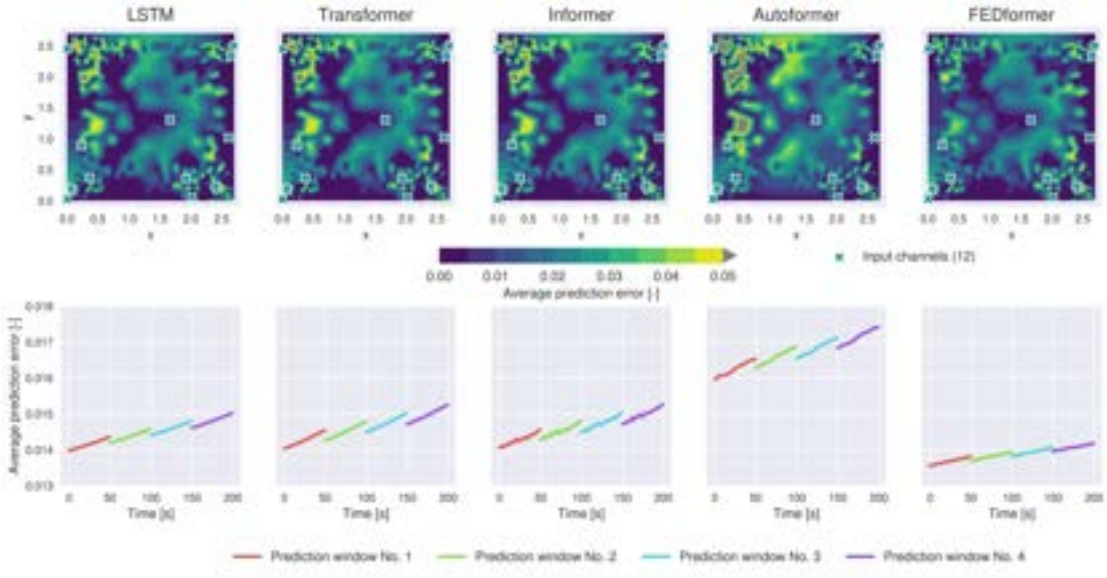


FIGURE A.14: Time-averaged prediction error distribution (Eq A.18) and prediction errors averaged at each time step (Eq. A.15) for the 2D heat conduction for five models with $l = 50$. Four consecutive prediction windows are chosen. The twelve green crosses show the input channels.

A.5 Conclusions

In this work, popular Transformer-based ML models were compared against simple one-layer LSTM models for transient thermal field reconstruction. Traditionally, Transformer-based models are considered advantageous over RNNs like LSTM for sequence processing for three main reasons [6]. The first one is their superior accuracy: their self-attention mechanism is expected to be better at capturing long-range dependencies, leading to higher accuracy. The second reason is the ability to train faster: the absence of recurrent connections makes them more parallelisable, which should result in shorter training times. The third and final reason is the enhanced interpretability: attention maps (Appendix B) are thought to make them less of a "black box" [123], thus increasing confidence in their predictions.

However, the results of this research challenge these assumptions for the thermal problems examined, offering the following counterarguments. Firstly, the Transformer-based models showed either lower or comparable prediction accuracy compared to the simple LSTM model. Secondly, transformer-based models had training times that were either longer than or similar to the simple LSTM model. Thirdly, the interpretability advantage is negated. While attention maps are useful for tasks like NLP [6] and computer vision [124], where the relationships between words or image parts are intuitive, this is not the case for purely temporal data. The relationships between values at different time steps are not as easily interpreted, especially for problems based on differential equations. Furthermore, LSTM layer weights can be visualised similarly (Appendix B), providing a similarly low level of interpretability for these problems.

Overall, the work concludes that there is no significant benefit to using complex Transformer-based ML models over simpler alternatives like the classic LSTM network for solving transient thermal field reconstruction problems. The application of these models to practical problems (such as those in Sub-section A.2.1) would require either a sufficient number of reliable forward simulations or a large amount of experimental data, which is often difficult to obtain. Additionally, the computational and memory demands increase with the length of the input time sequence (Table A.7), meaning that for more complex scenarios, more time is needed for training, and GPU memory (VRAM) must be carefully managed through batching.

TABLE A.7: Models' computational complexity and memory usage.

Model type	LSTM	Transformer	Informer	Autoformer	FEDformer
Computation complexity (training)	$O(L)$	$O(L^2)$	$O(LlogL)$	$O(LlogL)$	$O(L)$
Memory usage (training)	$O(L)$	$O(L^2)$	$O(LlogL)$	$O(LlogL)$	$O(L)$

* L denotes the input time sequence length.

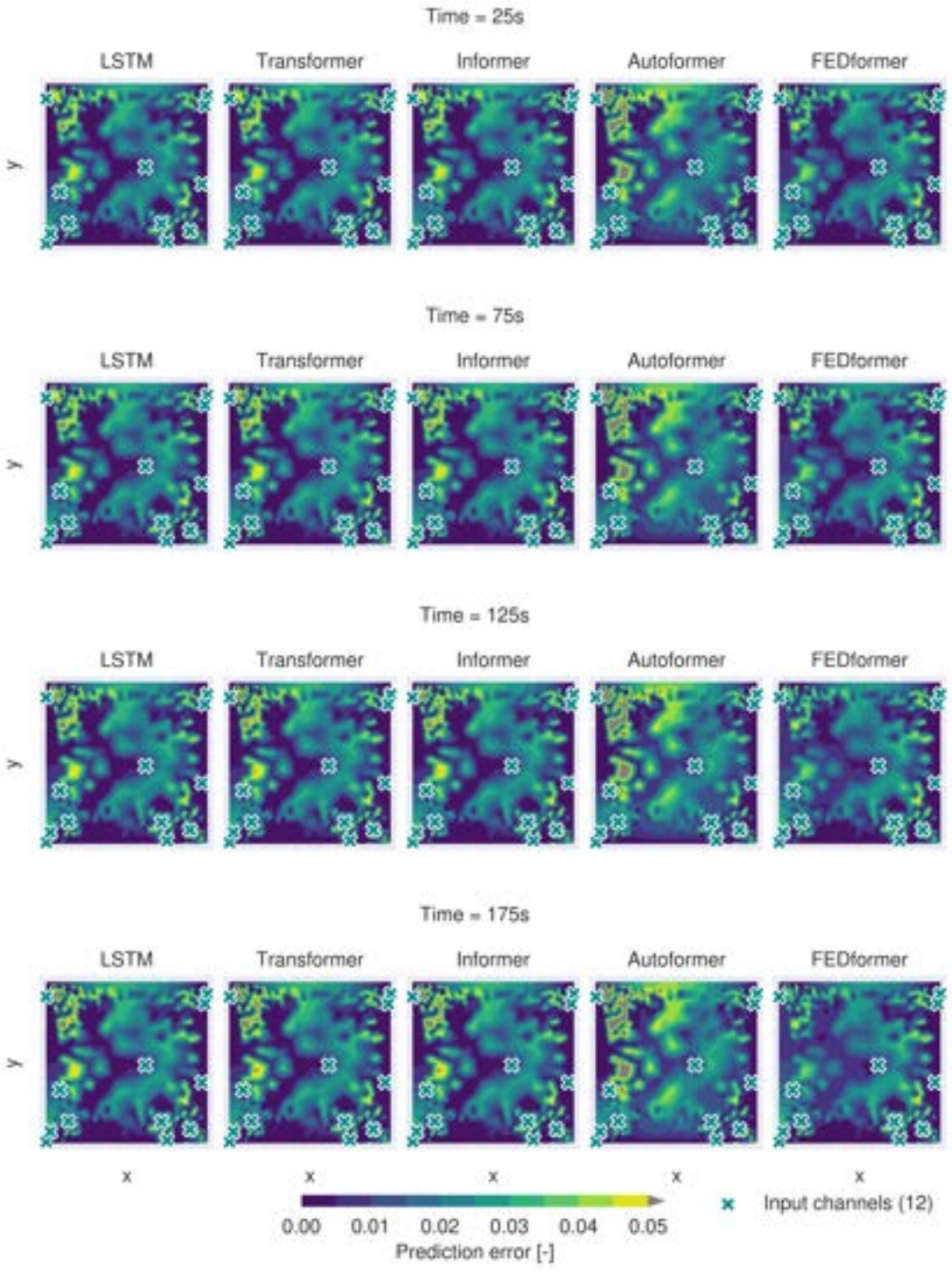


FIGURE A.15: Prediction error distribution dependence on time for the 2D heat conduction for five models with the prediction window size $l = 50$.

Appendix B

Attention maps and weight visualisations

B.1 Attention maps for Transformer-based models

The self-attention mechanism, or its variants, enables a model to focus on specific parts of an input vector sequence to produce an output vector sequence [6]. This means the model gives greater importance to selected input vectors it considers more relevant when generating a particular output vector. As noted in Sub-section A.2.4, a weight matrix, referred to as an attention matrix, is created by combining queries and keys produced from weight matrices. Each element of this attention matrix, an attention score, represents its contribution to a specific output vector.

To illustrate this, the NLP example in Figure B.1 could be considered [119]. Here, the key, representing a book's qualities, is matched with the query, representing a reader's preferences, using a dot product. The resulting attention score indicates how well the book aligns with the reader's preferences. In general, an attention score measures the relevance between a key and a query, showing the degree to which a particular output vector is influenced by a particular input vector.

For the transient thermal field reconstruction problems discussed in this work, an input vector i contains information from the input channels at the i^{th} time step, while an output vector i holds information from the output channels at the same time step (Figures

A.8 and A.12). With a classic Transformer model, the attention score that shows how much output vector i is influenced by input vector j is calculated using the dot product of query i and key j (Eq. B.1).

$$Attention\_score\_(i,j) = \{\boldsymbol{q_i}\}^T \cdot \{\boldsymbol{k_j}\} \quad i,j = 1,2,...,t \tag{B.1}$$

where $\{\boldsymbol{k_j}\}$ is the key vector j corresponding to the input vector j. $\{\boldsymbol{q_i}\}$ is the query vector corresponding to the output vector i. t is a sequence length (Table A.2).

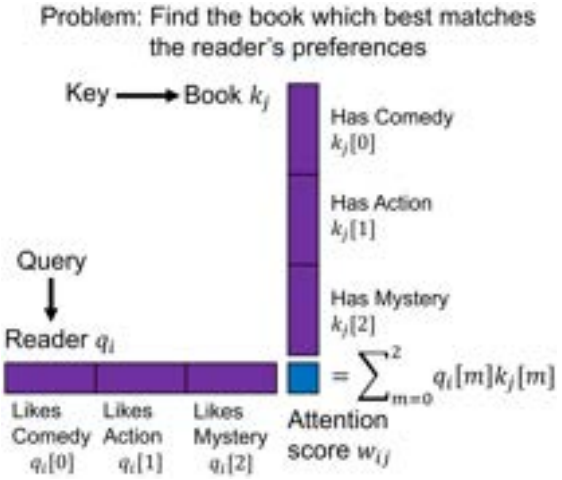


FIGURE B.1: In the book example, the self-attention operation works by matching a key (representing the book's qualities) with a query (representing the reader's preferences) using a dot product. The result is an attention score that indicates how relevant the book is to the reader's preferences. More broadly, the attention score measures the degree of relevance between a key and a query, showing how much a specific output vector is influenced by a particular input vector. For the transient thermal problems in this paper, an input vector i contains data from the input channels at time step i, while an output vector i contains data from the output channels at the same time step (Figures A.8 and A.12).

An attention matrix, which holds the attention scores, can be visualised as an image where each cell (i,j) corresponds to the attention score calculated for query vector i and key vector j. As an example, Figure B.2 shows a single attention score (i,j) on such a map.

The Transformer-based models in this study use multi-head attention (Figure A.6), which performs several self-attention operations in parallel (Sub-section A.2.4). This means that each multi-head attention operation generates multiple attention maps, one for each attention head. This work uses eight attention heads (Table A.2), so eight maps are produced per operation. It is crucial to note that attention scores should only

be compared within a single map, not across different heads. The size of the attention matrix varies by model: for Transformer [6] and Informer [103], it is $l \times l$, where l is the prediction window size (Table A.2). For Autoformer [104] and FEDformer [105], it is sequence length by d_{model}/h because keys are projected in the sequence length direction before scores are calculated.

Figures B.3 and B.4 display four of the eight attention maps from the first layer of each model for $l = 50$. The scores in each map are scaled between 0 and 1, with the scaling done independently for each map. These attention maps reveal some distinct characteristics. The maps for Transformer and Informer show diversity, with different heads focusing on different parts of the input sequence. This diversity is beneficial as it suggests the heads are extracting unique features from the data. The Transformer's maps are smoother than the Informer's, likely because Informer uses a sparse version of the self-attention mechanism. In contrast, Autoformer's attention maps look very similar despite minor differences. This is undesirable, as it suggests the heads are redundantly attempting to extract the same features. This similarity in attention maps could be a reason for Autoformer's lower performance compared to the other models.

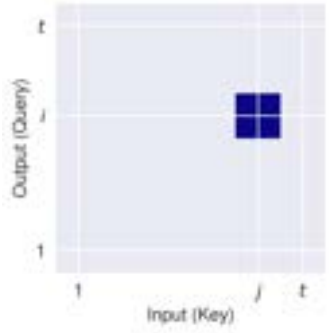


FIGURE B.2: Attention score example. The attention matrix comprised of the attention scores can be visualised as a map. Each cell (i, j) corresponds to the attention score computed for query vector i and key vector j.

The ability to generate attention maps makes Transformer-based models more interpretable than other ML models. These maps visually demonstrate how the model focuses on different sections of the input vector sequence. By visualising the attention scores in this way, it is possible to see which parts of the input sequence the model "pays attention" to at each time step.

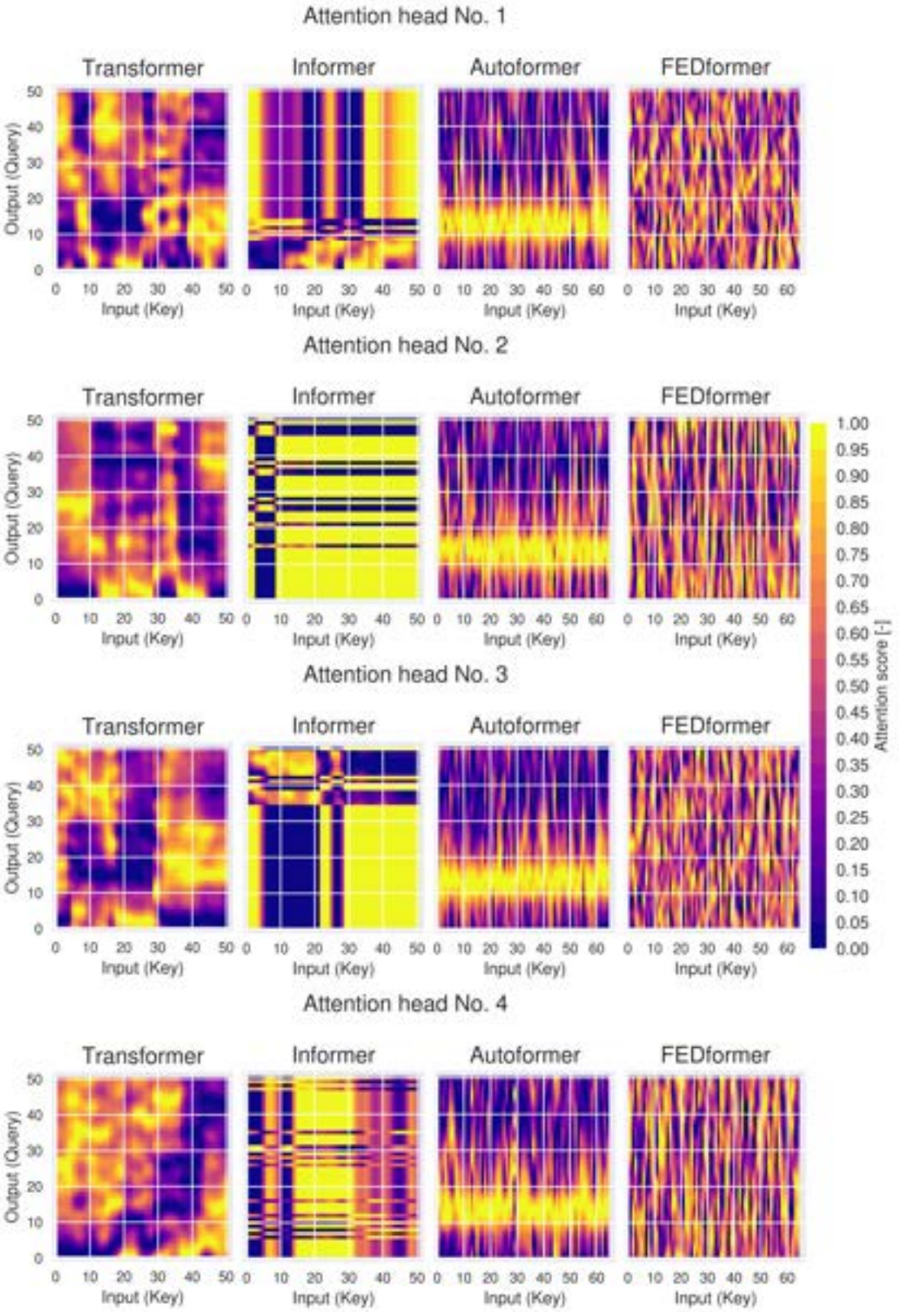


FIGURE B.3: Attention maps for Encoder layer No. 1 for 1D heat conduction for $l = 50$. Four out of eight attention maps for multi-head attention in the first layer of each model are shown. The attention scores are normalised to be between 0 and 1.

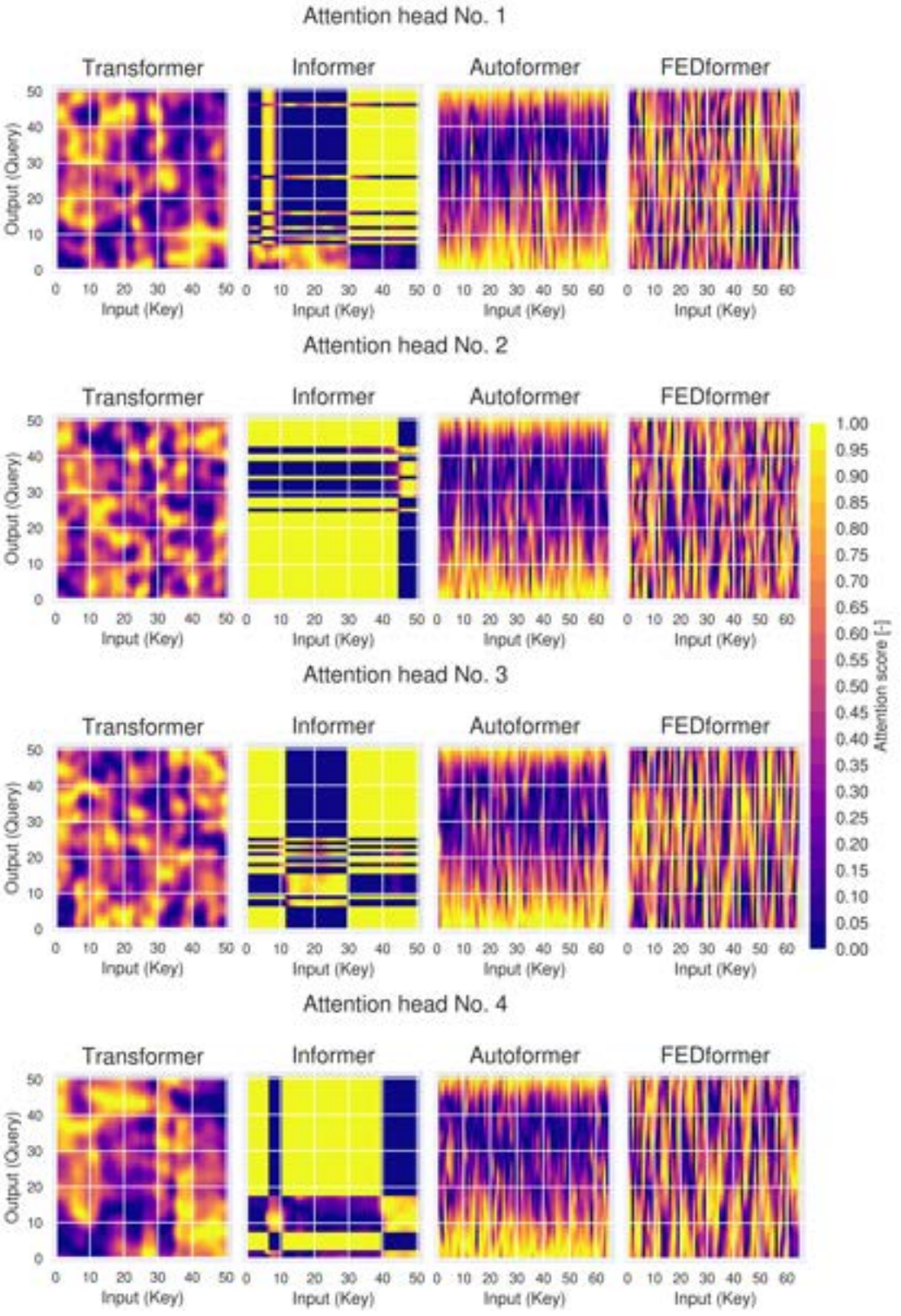


FIGURE B.4: Attention maps for Encoder layer No. 1 for 2D heat conduction $l = 50$. Four out of eight attention maps for multi-head attention in the first layer of each model are shown. The attention scores are normalised to be between 0 and 1.

B.2 Weight visualisations for Long Short-Term Memory models

The purpose of this section is to demonstrate that a weight visualisation similar to an attention map can be created for a one-layer LSTM model. For the transient thermal field reconstruction problems examined in this paper, these LSTM visualisations offer the same low level of intuitive interpretability as those for the Transformer-based models. Figure B.5 displays the values of h_t for each time step l (Eq. A.2) for LSTM models with a prediction window size of 50. These values, referred to as "attention scores" in Figure B.5 for consistency, are individually scaled between 0 and 1 for each visualisation.

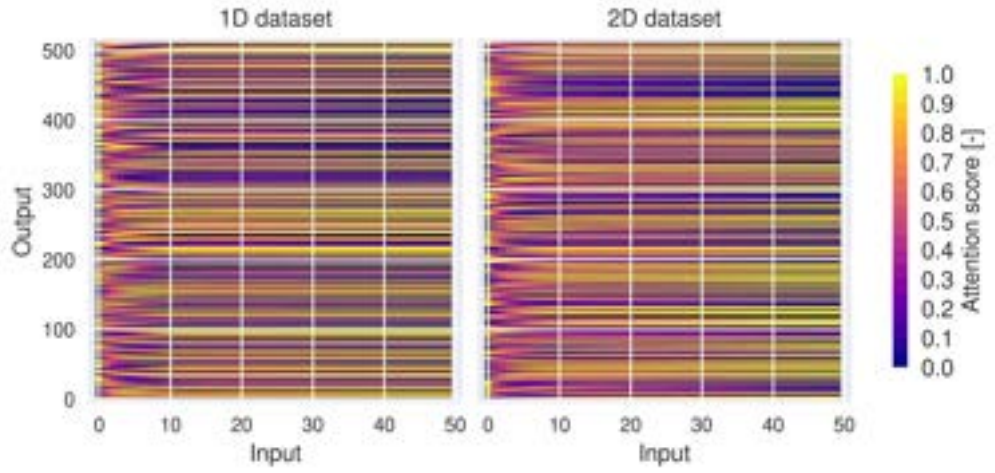


FIGURE B.5: LSTM model weight visualisations for 1D and 2D heat conduction for $l = 50$. h_t value for each time step l (Eq. A.2) is shown. The attention scores are normalised to be between 0 and 1.

Appendix C

Extended temperature control results

This appendix provides extended tables and figures related to Chapter 5. The results are grouped by the heat flux type (Table 5.9) and 3 DT approaches.

C.1 Heat flux in the form of Sine wave 1

C.1.1 Finite Element-based Digital Twinning control without Reduced Order Modelling

Table C.1 shows the overshoots and control run times as a response to Sine wave 1 heat flux shape. Figures C.1, C.2, C.3, and C.4 display the corresponding system responses.

C.1.2 Finite Element-based Digital Twinning control with Reduced Order Modelling

Table C.2 shows the overshoots and control run times as a response to Sine wave 1 heat flux shape. Figures C.5, C.6, C.7, and C.8 display the corresponding system responses.

TABLE C.1: System response under FE DT control without ROM and average and maximum time step runtimes; applied heat flux $q(t)$ is in the form of Sine wave 1 (Table 5.9 and Figure 5.21).

		Overshoot [%]		Sol. rec. + Control time		Sol. rec.	
No. of measurement locations	Noise level σ [%]	Primary	Secondary	Avg. time step runtime [s] [a]	Max. time step runtime [s] [a]	Avg. relative error [%]	Max. relative error [%]
4	0.0	0.649	0.649	2.221e+00	2.454e+00	0.184	2.192
4	0.2	1.939	1.939	2.224e+00	2.438e+00	0.244	4.769
4	1.0	15.956	14.242	2.216e+00	2.390e+00	0.697	24.52
4	$\sigma(T)$	4.722	4.722	2.238e+00	2.575e+00	0.33	8.109
8	0.0	0.736	0.736	2.272e+00	2.528e+00	0.177	2.352
8	0.2	3.223	3.223	2.289e+00	2.610e+00	0.247	5.825
8	1.0	10.305	8.678	2.288e+00	2.523e+00	0.563	25.958
8	$\sigma(T)$	4.995	4.995	2.273e+00	2.513e+00	0.301	10.072
11	0.0	1.392	1.392	2.293e+00	2.553e+00	0.148	4.871
11	0.2	2.189	2.189	2.291e+00	2.547e+00	0.199	5.629
11	1.0	7.918	7.918	2.293e+00	2.621e+00	0.468	22.276
11	$\sigma(T)$	6.021	6.021	2.270e+00	2.487e+00	0.263	7.248
17	0.0	0.816	0.816	2.268e+00	2.482e+00	0.141	5.629
17	0.2	2.49	2.49	2.288e+00	2.596e+00	0.187	6.21
17	1.0	13.024	10.159	2.286e+00	2.535e+00	0.48	21.744
17	$\sigma(T)$	3.375	3.375	2.291e+00	2.500e+00	0.227	8.223

[a] Times are given for AMD Ryzen 7 5800X 8-Core CPU.

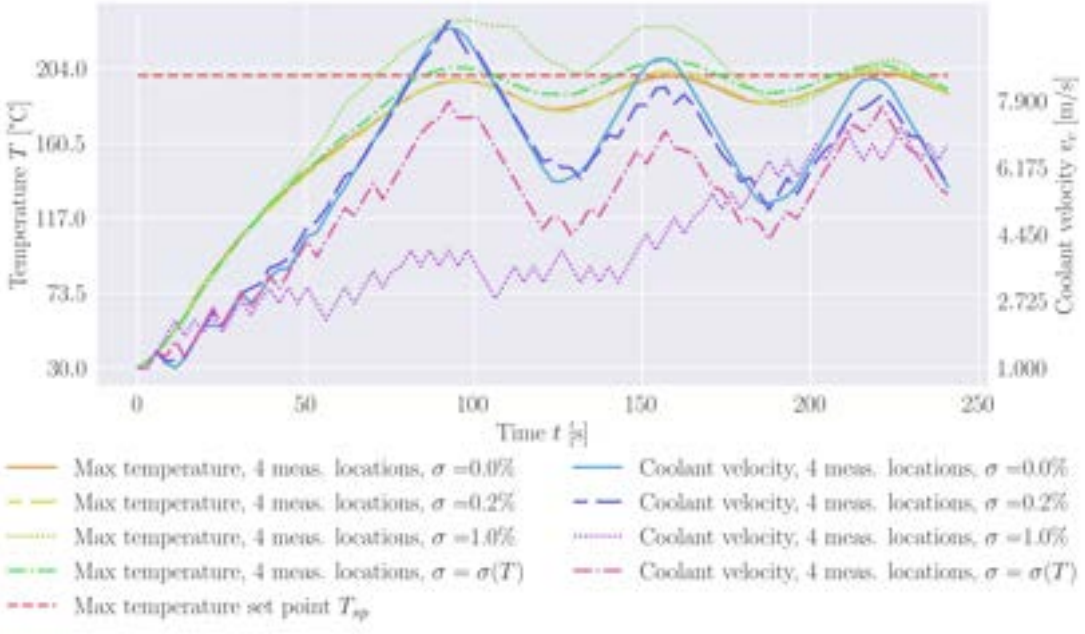


FIGURE C.1: FE DT without ROM system response Sine wave 1 (Table 5.9 and Figure 5.21) for 4 measurement locations; the overshoots are listed in Table C.1.

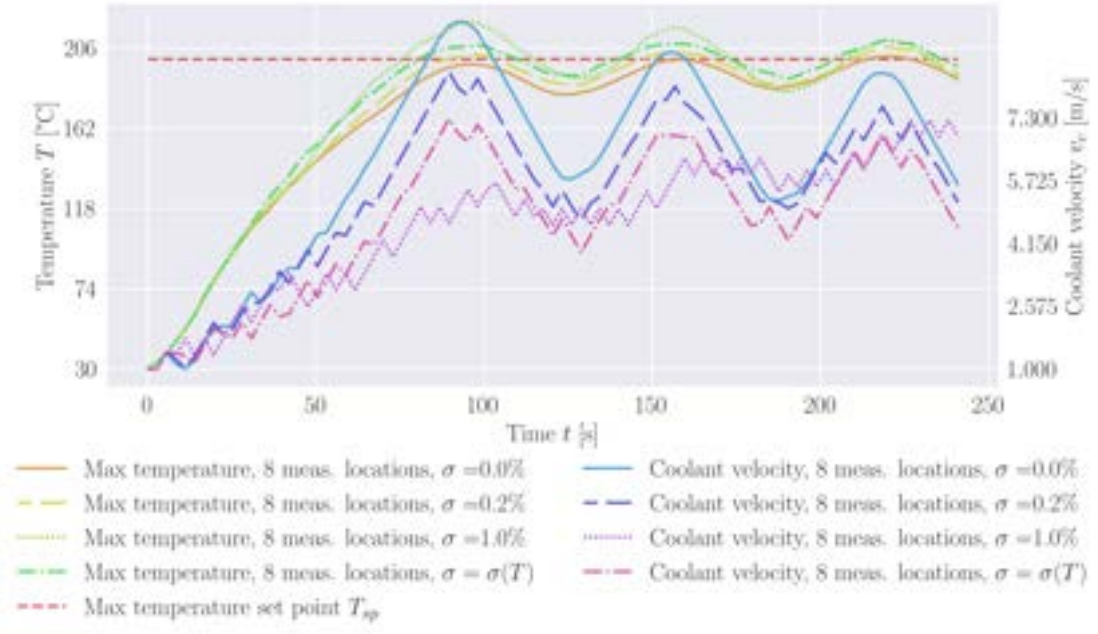


FIGURE C.2: FE DT without ROM system response Sine wave 1 (Table 5.9 and Figure 5.21) for 8 measurement locations; the overshoots are listed in Table C.1.

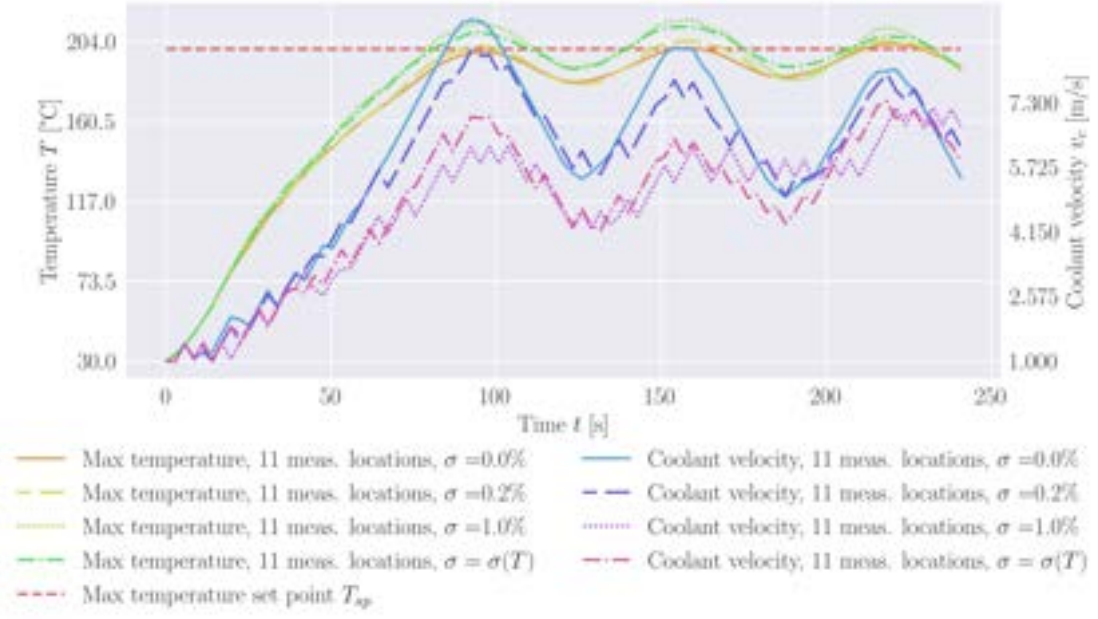


FIGURE C.3: FE DT without ROM system response Sine wave 1 (Table 5.9 and Figure 5.21) for 11 measurement locations; the overshoots are listed in Table C.1.

C.1.3 Physics-Driven Machine Learning-based Digital Twinning control

Table C.3 shows the overshoots and control run times as a response to Sine wave 1 heat flux shape. Figures C.9, C.10, C.11, and C.12 display the corresponding system

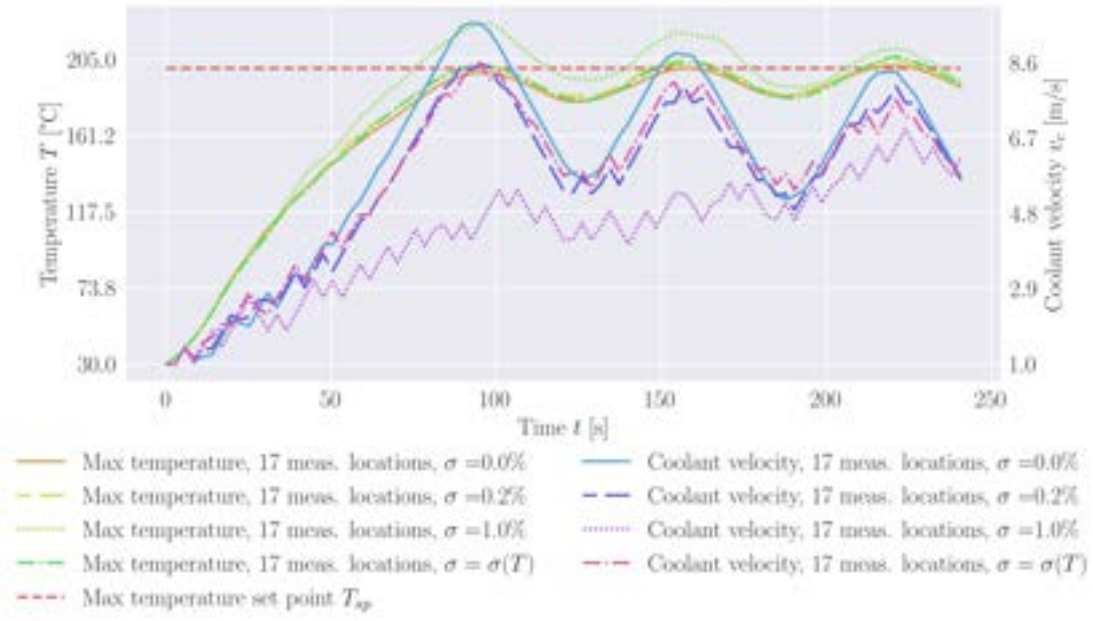


FIGURE C.4: FE DT without ROM system response Sine wave 1 (Table 5.9 and Figure 5.21) for 17 measurement locations; the overshoots are listed in Table C.1.

TABLE C.2: System response under FE DT control with ROM and average and maximum time step runtimes; applied heat flux $q(t)$ is in the form of Sine wave 1 (Table 5.9 and Figure 5.21).

		Overshoot [%]		Sol. rec. + Control time		Sol. rec.	
No. of measurement locations	Noise level σ [%]	Primary	Secondary	Avg. time step runtime [s] [a]	Max. time step runtime [s] [a]	Avg. relative error [%]	Max. relative error [%]
4	0.0	-1.856	0.82	3.123e-01	4.268e-01	0.75	3.768
4	0.2	0.584	3.566	3.150e-01	4.134e-01	0.839	4.706
4	1.0	23.335	16.868	3.129e-01	4.148e-01	1.241	22.735
4	$\sigma(T)$	2.67	3.661	3.163e-01	3.825e-01	0.875	7.354
8	0.0	-1.786	1.1	3.024e-01	4.409e-01	0.694	4.123
8	0.2	2.122	8.072	3.151e-01	4.003e-01	0.768	5.558
8	1.0	16.052	14.287	3.030e-01	3.852e-01	1.008	24.897
8	$\sigma(T)$	3.941	3.432	3.119e-01	4.595e-01	0.788	9.573
11	0.0	3.944	5.213	2.225e-01	2.912e-01	0.622	6.815
11	0.2	5.31	3.824	2.575e-01	3.904e-01	0.615	6.945
11	1.0	15.396	13.529	2.377e-01	3.769e-01	0.928	26.62
11	$\sigma(T)$	7.428	4.731	2.712e-01	3.916e-01	0.651	9.917
17	0.0	4.019	5.093	3.162e-01	4.118e-01	0.652	9.607
17	0.2	3.472	9.748	3.157e-01	4.272e-01	0.679	11.998
17	1.0	10.82	11.806	3.171e-01	4.073e-01	0.909	27.681
17	$\sigma(T)$	5.759	11.241	3.108e-01	3.892e-01	0.717	14.806

[a] Times are given for AMD Ryzen 7 5800X 8-Core CPU.

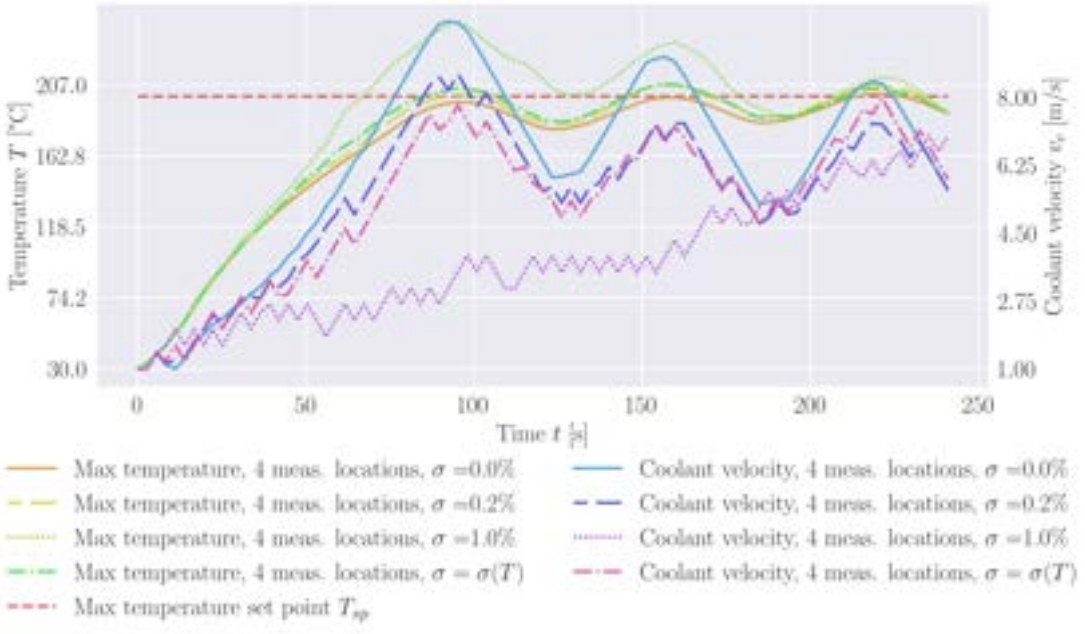


FIGURE C.5: FE DT with ROM system response Sine wave 1 (Table 5.9 and Figure 5.21) for 4 measurement locations; the overshoots are listed in Table C.1.

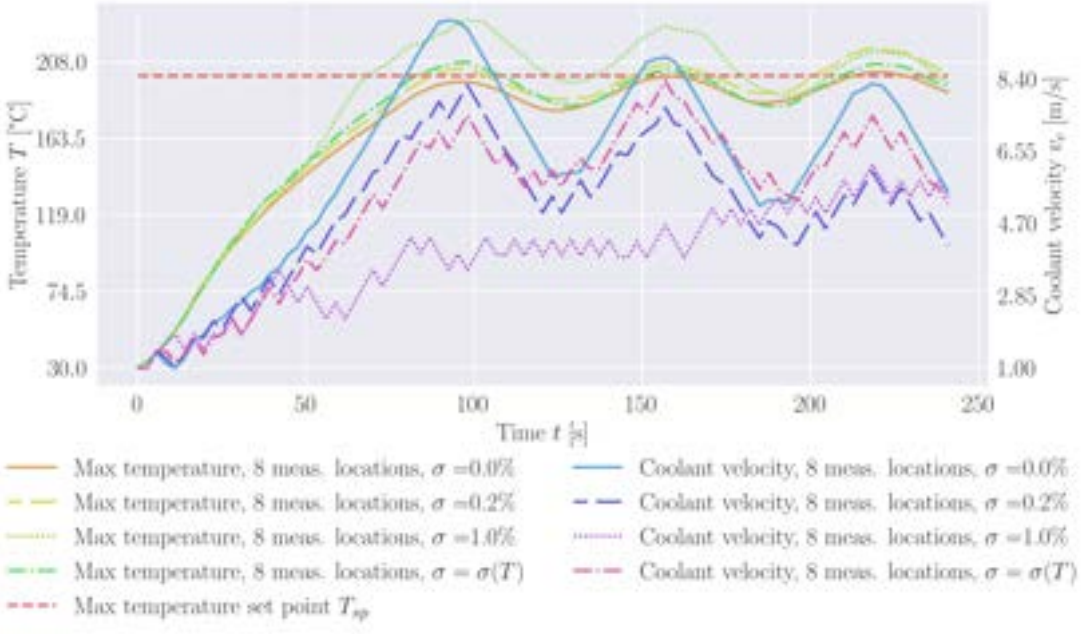


FIGURE C.6: FE DT with ROM system response Sine wave 1 (Table 5.9 and Figure 5.21) for 8 measurement locations; the overshoots are listed in Table C.1.

responses.

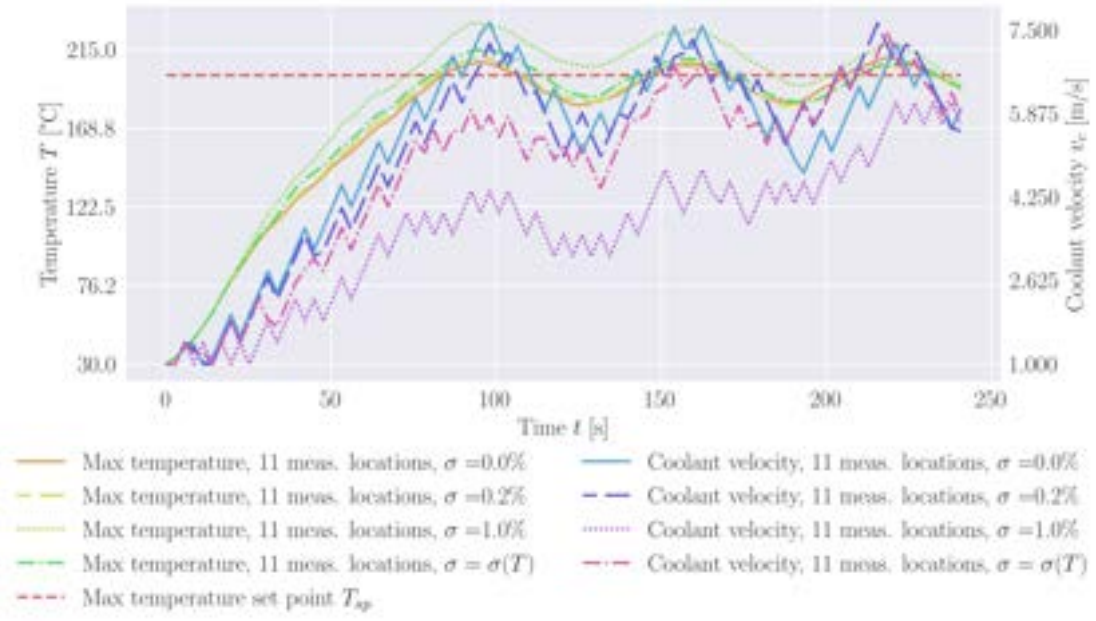


FIGURE C.7: FE DT with ROM system response Sine wave 1 (Table 5.9 and Figure 5.21) for 11 measurement locations; the overshoots are listed in Table C.1.

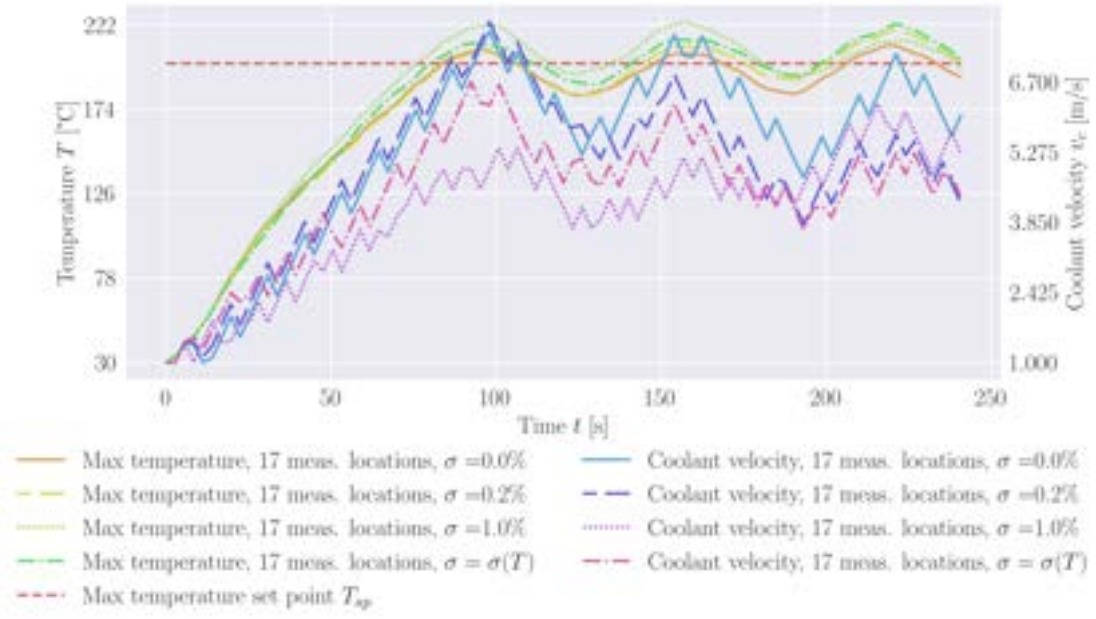


FIGURE C.8: FE DT with ROM system response Sine wave 1 (Table 5.9 and Figure 5.21) for 17 measurement locations; the overshoots are listed in Table C.1.

TABLE C.3: PD-ML DT system response under PD-ML DT control and average and maximum time step runtimes; applied heat flux $q(t)$ is in the form of Sine wave 1 (Table 5.9 and Figure 5.21).

Parameter set			Overshoot [%]		Control time	
N_{int}	N_{ext}	Noise level σ [%]	Primary	Secondary	Avg. time step runtime [s] [a]	Max. time step runtime [s] [a]
60	0	0.0	4.476	4.289	3.643e-04	5.991e-04
60	0	0.2	4.709	4.347	3.830e-04	9.775e-04
60	0	1.0	5.078	3.822	3.630e-04	8.943e-04
60	0	$\sigma(T)$	4.743	4.320	3.742e-04	8.216e-04
120	0	0.0	4.476	1.078	3.734e-04	1.022e-03
120	0	0.2	4.812	0.880	3.647e-04	6.154e-04
120	0	1.0	5.202	0.527	3.651e-04	5.813e-04
120	0	$\sigma(T)$	4.743	0.983	3.690e-04	7.968e-04
60	120	0.0	-1.446	-1.446	1.063e-03	1.828e-03
60	120	0.2	5.179	-0.708	1.065e-03	1.820e-03
60	120	1.0	4.851	-0.158	1.031e-03	1.807e-03
60	120	$\sigma(T)$	4.627	-0.759	1.107e-03	1.565e-03
100	120	0.0	-0.037	-1.647	1.786e-03	3.057e-03
100	120	0.2	4.958	-1.574	2.268e-03	3.416e-03
100	120	1.0	4.846	-0.463	1.978e-03	3.006e-03
100	120	$\sigma(T)$	5.019	-1.181	1.861e-03	3.154e-03

[a] Times are given for AMD Ryzen 7 5800X 8-Core CPU.

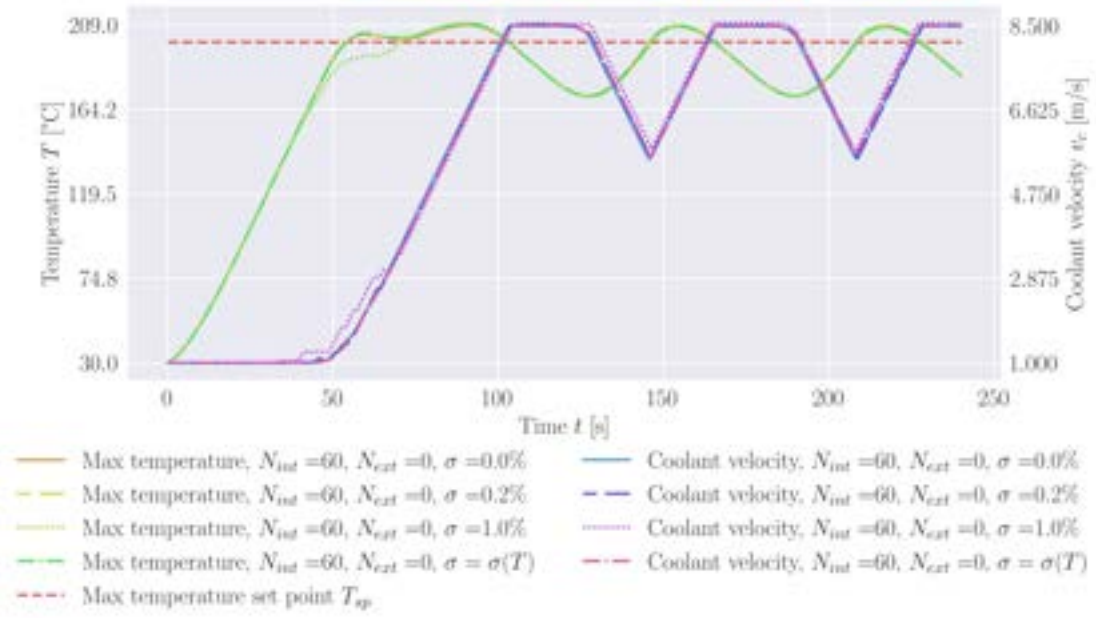


FIGURE C.9: PD-ML DT system response Sine wave 1 (Table 5.9 and Figure 5.21) for N_{int} equal to 60 and N_{ext} equal to 0; the overshoots are listed in Table C.3.

C.2 Heat flux in the form of Sine wave 2

C.2.1 Finite Element-based Digital Twinning control without Reduced Order Modelling

Table C.4 shows the overshoots and control run times as a response to Sine wave 2 heat flux shape. Figures C.13, C.14, and C.15 display the corresponding system responses.

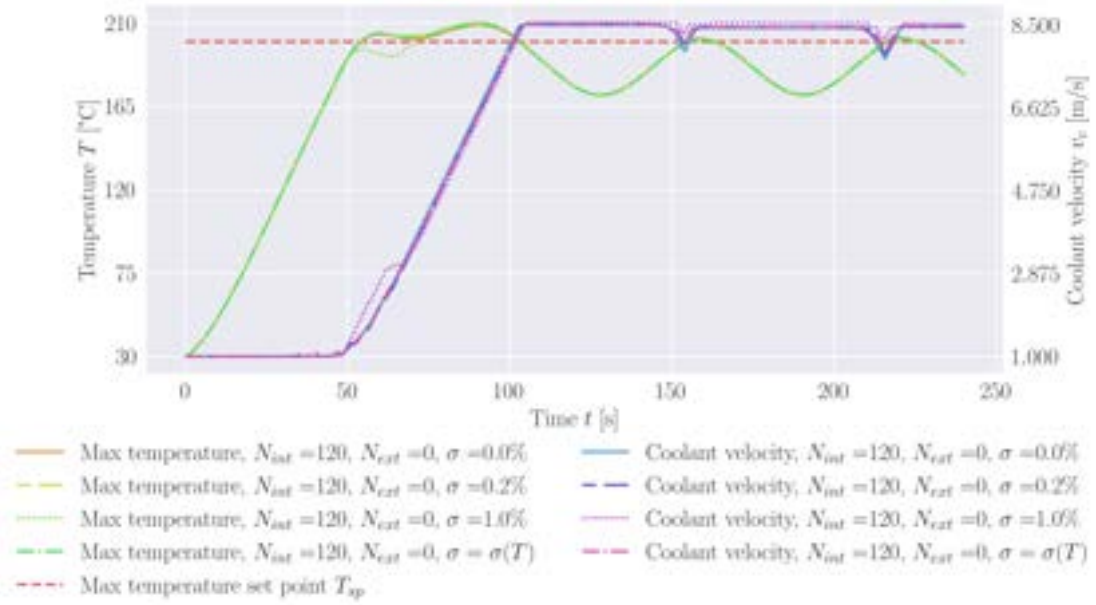


FIGURE C.10: PD-ML DT system response Sine wave 1 (Table 5.9 and Figure 5.21) for N_{int} equal to 120 and N_{ext} equal to 0; the overshoots are listed in Table C.3.

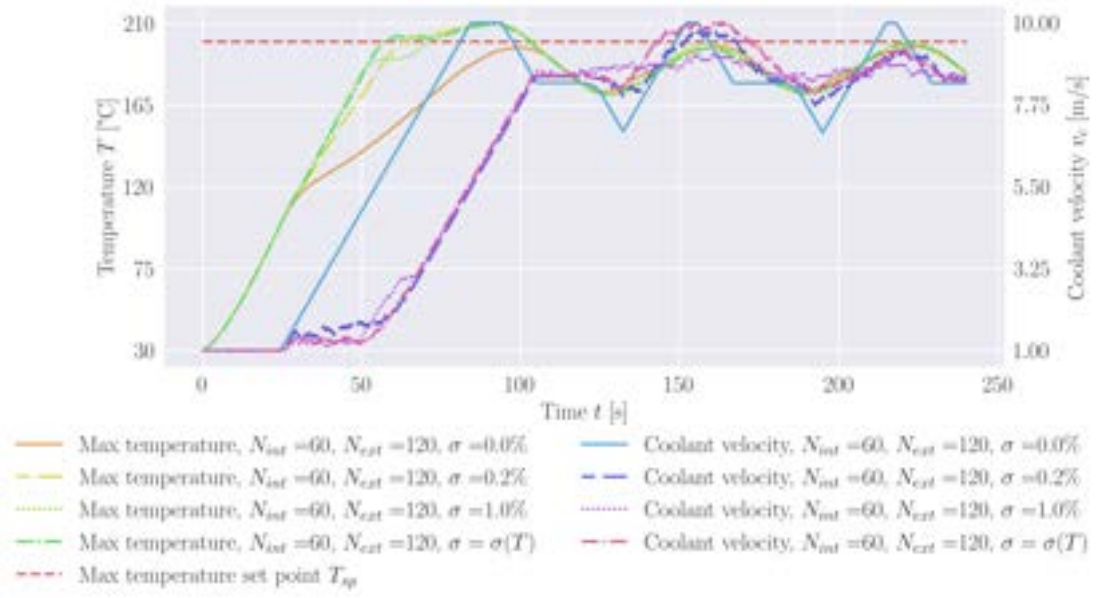


FIGURE C.11: PD-ML DT system response Sine wave 1 (Table 5.9 and Figure 5.21) for N_{int} equal to 60 and N_{ext} equal to 120; the overshoots are listed in Table C.3.

C.2.2 Finite Element-based Digital Twinning control with Reduced Order Modelling

Table C.5 shows the overshoots and control run times as a response to Sine wave 2 heat flux shape. Figures C.16, C.17, and C.18 display the corresponding system responses.

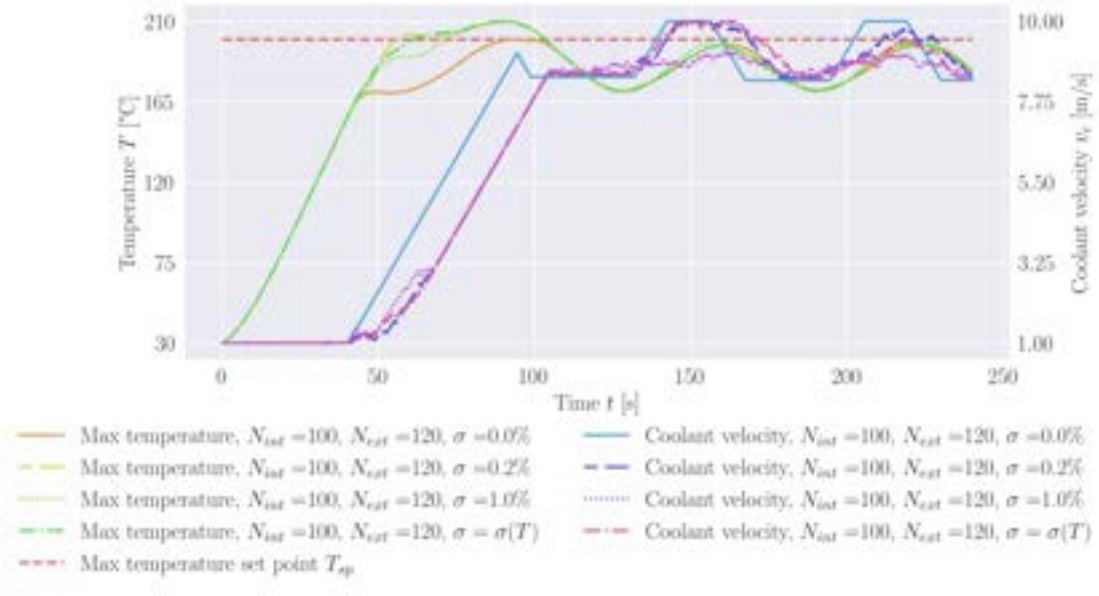


FIGURE C.12: PD-ML DT system response Sine wave 1 (Table 5.9 and Figure 5.21) for N_{int} equal to 100 and N_{ext} equal to 120; the overshoots are listed in Table C.3.

TABLE C.4: System response under FE DT control without ROM and average and maximum time step runtimes; applied heat flux $q(t)$ is in the form of Sine wave 2 (Table 5.9 and Figure 5.21).

		Overshoot [%]		Sol. rec. + Control time		Sol. rec.	
No. of measurement locations	Noise level σ [%]	Primary	Secondary	Avg. time step runtime [s] [a]	Max. time step runtime [s] [a]	Avg. relative error [%]	Max. relative error [%]
4	0.0	1.357	1.357	2.286e+00	2.469e+00	0.173	2.192
4	0.2	1.755	1.755	2.289e+00	2.526e+00	0.246	4.928
4	1.0	18.061	11.795	2.297e+00	2.595e+00	0.693	24.693
4	$\sigma(T)$	3.497	3.497	2.295e+00	2.635e+00	0.331	8.109
8	0.0	1.391	1.391	2.283e+00	2.534e+00	0.166	2.352
8	0.2	3.32	3.32	2.282e+00	2.493e+00	0.253	5.825
8	1.0	9.053	4.787	2.296e+00	2.512e+00	0.562	25.999
8	$\sigma(T)$	4.962	4.962	2.306e+00	2.494e+00	0.306	10.072
11	0.0	1.97	1.97	2.282e+00	2.466e+00	0.166	4.871
11	0.2	2.99	2.99	2.290e+00	2.538e+00	0.21	5.629
11	1.0	5.65	3.358	2.277e+00	2.502e+00	0.452	22.267
11	$\sigma(T)$	3.503	2.604	2.289e+00	2.548e+00	0.261	7.248
17	0.0	1.983	1.983	2.278e+00	2.505e+00	0.143	5.629
17	0.2	0.388	0.294	2.269e+00	2.507e+00	0.176	6.21
17	1.0	14.728	10.579	2.279e+00	2.542e+00	0.491	21.744
17	$\sigma(T)$	2.973	2.973	2.286e+00	2.556e+00	0.227	8.223

[a] Times are given for AMD Ryzen 7 5800X 8-Core CPU.

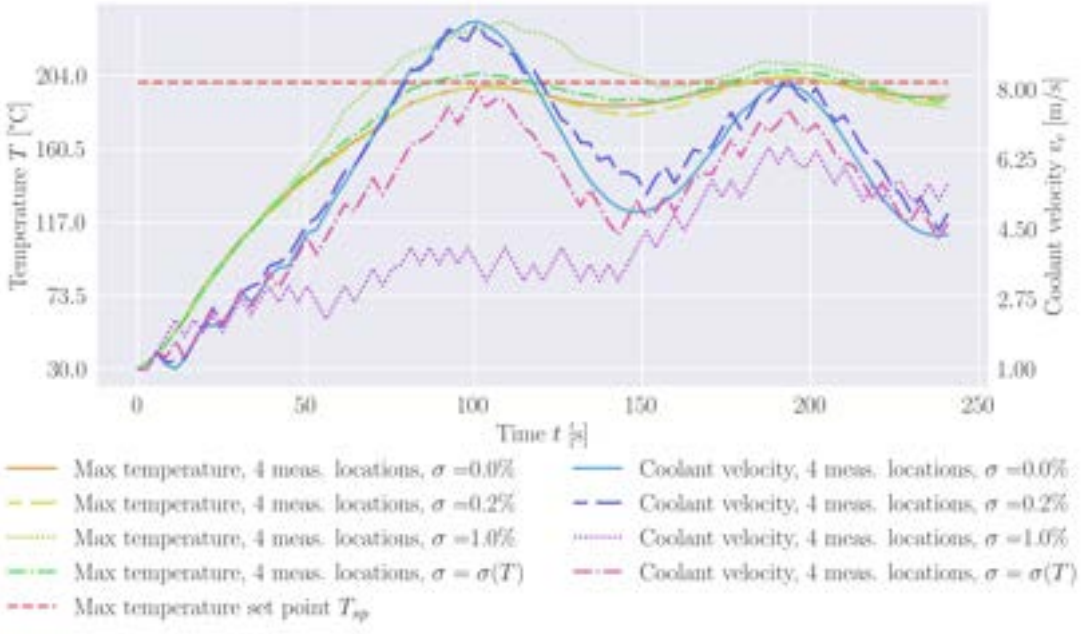


FIGURE C.13: FE DT without ROM system response Sine wave 2 (Table 5.9 and Figure 5.21) for 4 measurement locations; the overshoots are listed in Table C.1.

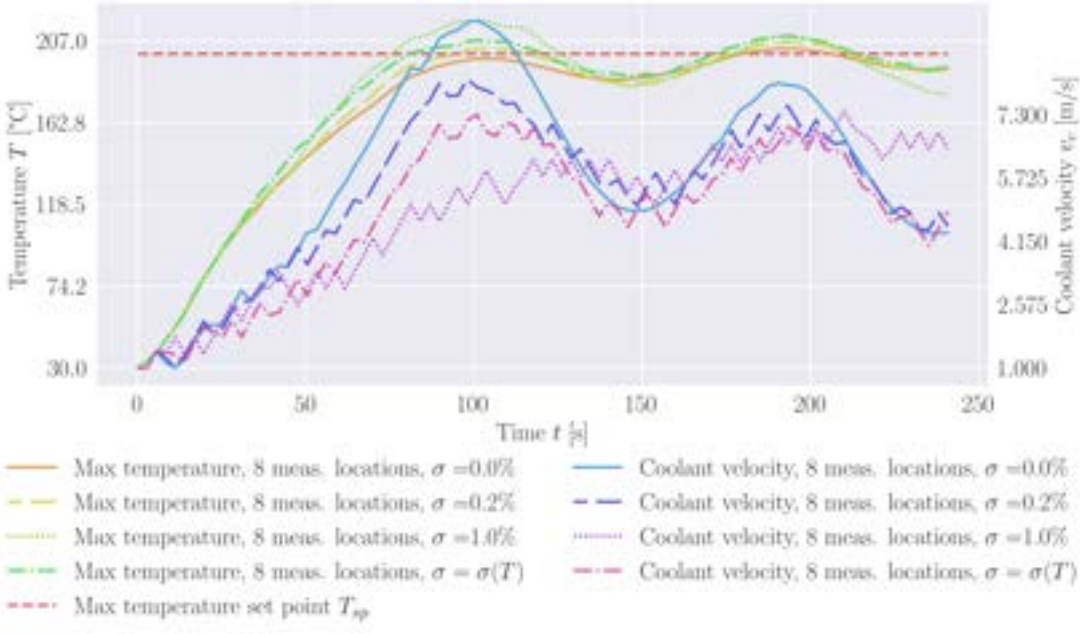


FIGURE C.14: FE DT without ROM system response Sine wave 2 (Table 5.9 and Figure 5.21) for 8 measurement locations; the overshoots are listed in Table C.1.

C.2.3 Physics-Driven Machine Learning-based Digital Twinning control

Table C.6 shows the overshoots and control run times as a response to Sine wave 2 heat flux shape. Figures C.19, C.20, and C.21 display the corresponding system responses.

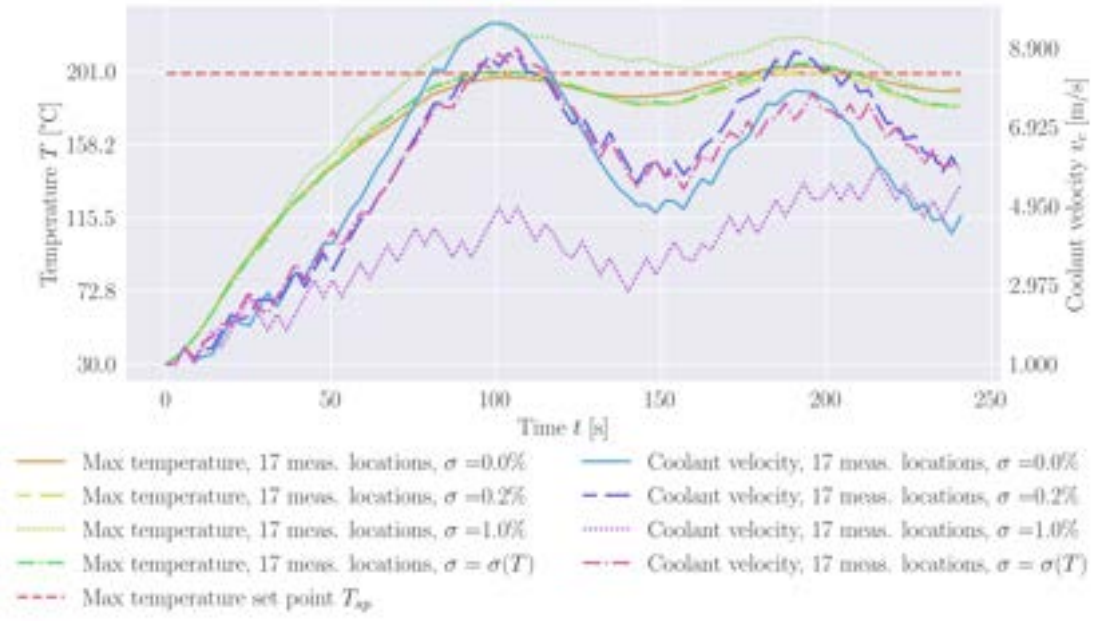


FIGURE C.15: FE DT without ROM system response Sine wave 2 (Table 5.9 and Figure 5.21) for 17 measurement locations; the overshoots are listed in Table C.1.

TABLE C.5: System response under FE DT control with ROM and average and maximum time step runtimes; applied heat flux $q(t)$ is in the form of Sine wave 2 (Table 5.9 and Figure 5.21).

		Overshoot [%]		Sol. rec. + Control time		Sol. rec.	
No. of measurement locations	Noise level σ [%]	Primary	Secondary	Avg. time step runtime [s] [a]	Max. time step runtime [s] [a]	Avg. relative error [%]	Max. relative error [%]
4	0.0	-1.749	1.493	3.086e-01	4.190e-01	0.723	3.696
4	0.2	1.54	1.109	3.126e-01	4.387e-01	0.822	4.801
4	1.0	21.699	10.437	3.135e-01	3.995e-01	1.208	22.976
4	$\sigma(T)$	3.94	4.406	3.142e-01	3.975e-01	0.879	7.354
8	0.0	-1.694	1.545	3.086e-01	3.914e-01	0.661	4.064
8	0.2	2.103	2.354	3.171e-01	4.110e-01	0.743	5.558
8	1.0	16.722	7.348	3.143e-01	4.017e-01	0.962	25.009
8	$\sigma(T)$	4.103	3.836	3.098e-01	4.242e-01	0.802	9.573
11	0.0	2.738	2.835	2.568e-01	4.131e-01	0.628	6.815
11	0.2	5.271	3.79	3.130e-01	3.862e-01	0.629	6.945
11	1.0	14.487	14.352	2.899e-01	3.946e-01	0.935	26.555
11	$\sigma(T)$	7.211	7.919	2.750e-01	3.789e-01	0.692	9.917
17	0.0	2.809	3.034	3.131e-01	4.296e-01	0.658	9.607
17	0.2	2.362	1.684	3.201e-01	4.443e-01	0.649	11.998
17	1.0	12.194	10.82	3.093e-01	4.154e-01	0.907	27.681
17	$\sigma(T)$	4.775	4.236	3.123e-01	4.001e-01	0.671	14.806

[a] Times are given for AMD Ryzen 7 5800X 8-Core CPU.

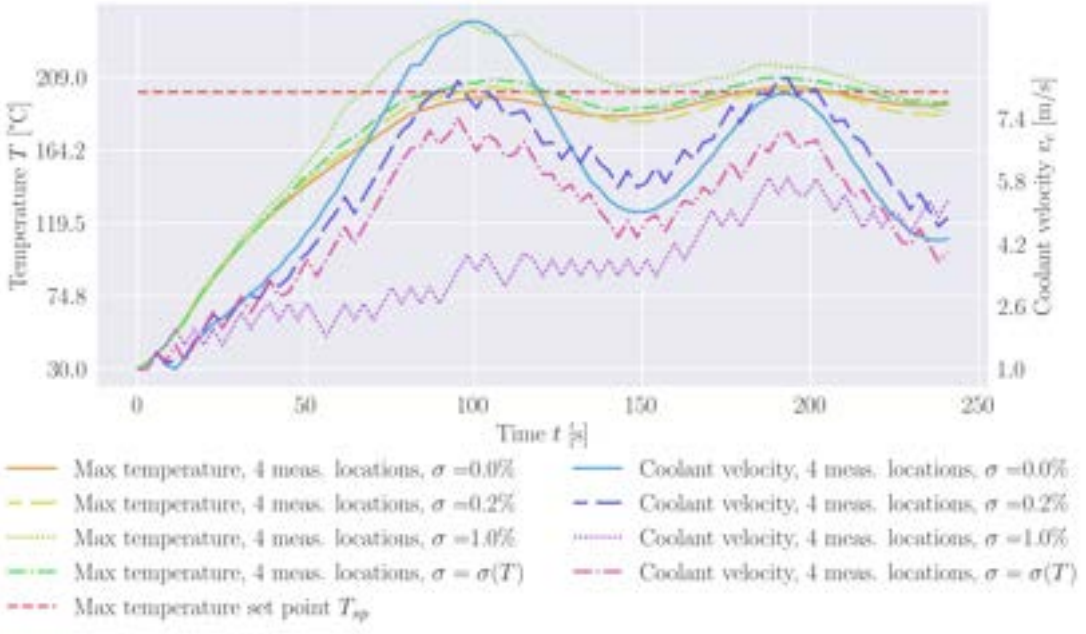


FIGURE C.16: FE DT with ROM system response Sine wave 2 (Table 5.9 and Figure 5.21) for 4 measurement locations; the overshoots are listed in Table C.1.

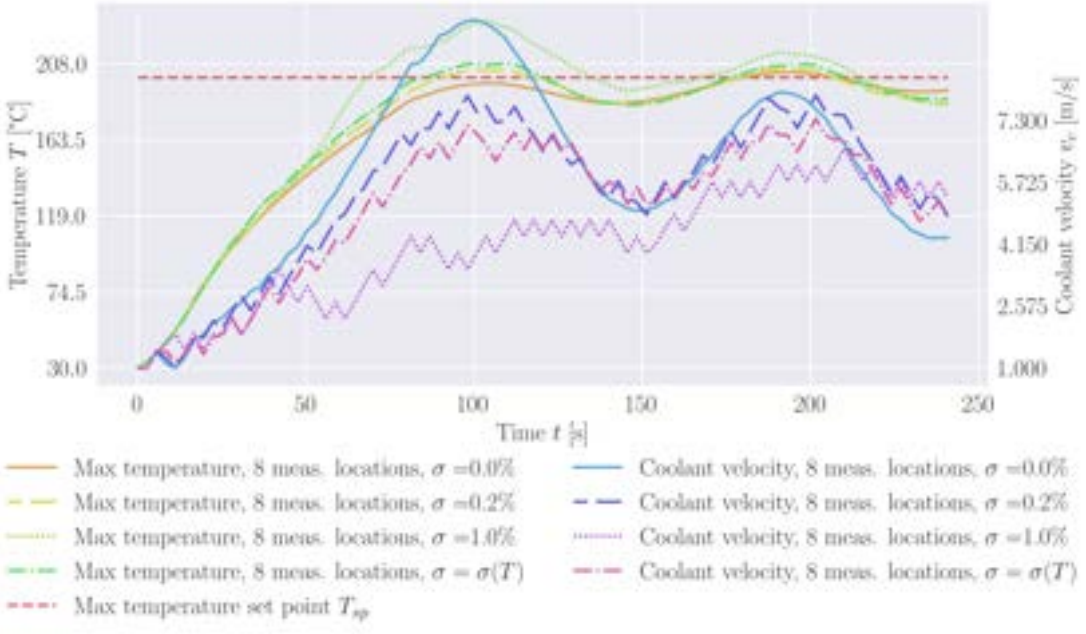


FIGURE C.17: FE DT with ROM system response Sine wave 2 (Table 5.9 and Figure 5.21) for 8 measurement locations; the overshoots are listed in Table C.1.

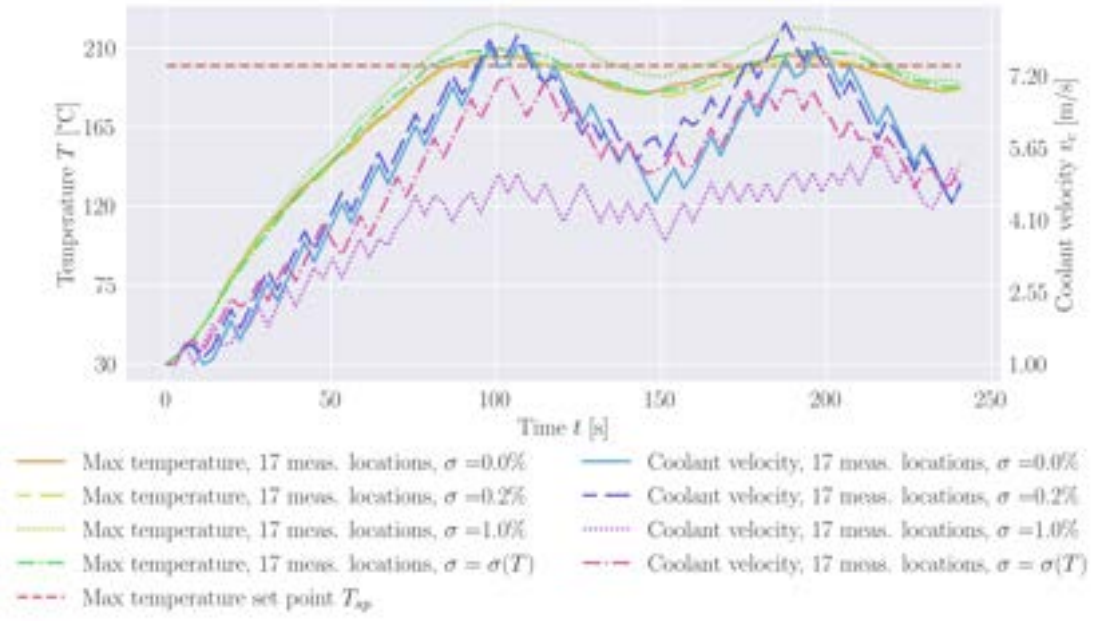


FIGURE C.18: FE DT with ROM system response Sine wave 2 (Table 5.9 and Figure 5.21) for 17 measurement locations; the overshoots are listed in Table C.1.

TABLE C.6: PD-ML DT system response under PD-ML DT control and average and maximum time step runtimes; applied heat flux $q(t)$ is in the form of Sine wave 2 (Table 5.9 and Figure 5.21).

Parameter set			Overshoot [%]		Control time	
N_{int}	N_{ext}	Noise level σ [%]	Primary	Secondary	Avg. time step runtime [s] [a]	Max. time step runtime [s] [a]
60	0	0.0	3.477	3.477	3.674e-04	5.908e-04
60	0	0.2	3.415	3.415	3.851e-04	9.041e-04
60	0	1.0	3.550	3.550	3.679e-04	8.273e-04
60	0	$\sigma(T)$	3.773	3.773	3.606e-04	5.920e-04
120	0	0.0	2.862	2.862	3.709e-04	5.944e-04
120	0	0.2	3.202	2.361	3.637e-04	5.896e-04
120	0	1.0	3.644	1.149	3.627e-04	5.894e-04
120	0	$\sigma(T)$	3.096	2.466	3.646e-04	5.996e-04
60	120	0.0	-0.310	-0.310	1.003e-03	1.980e-03
60	120	0.2	3.750	-0.429	1.058e-03	1.781e-03
60	120	1.0	3.197	-0.507	1.062e-03	1.660e-03
60	120	$\sigma(T)$	2.942	0.510	1.062e-03	1.539e-03
100	120	0.0	-0.041	-0.628	1.979e-03	3.058e-03
100	120	0.2	3.391	-1.279	2.227e-03	3.266e-03
100	120	1.0	3.191	-0.894	2.073e-03	3.753e-03
100	120	$\sigma(T)$	3.516	-1.241	2.015e-03	3.147e-03

[a] Times are given for AMD Ryzen 7 5800X 8-Core CPU.

C.3 Heat flux in the form of Triangular wave

C.3.1 Finite Element-based Digital Twinning control without Reduced Order Modelling

Table C.7 shows the overshoots and control run times as a response to Triangular heat flux shape. Figures C.22, C.23, C.24, and C.25 display the corresponding system re-

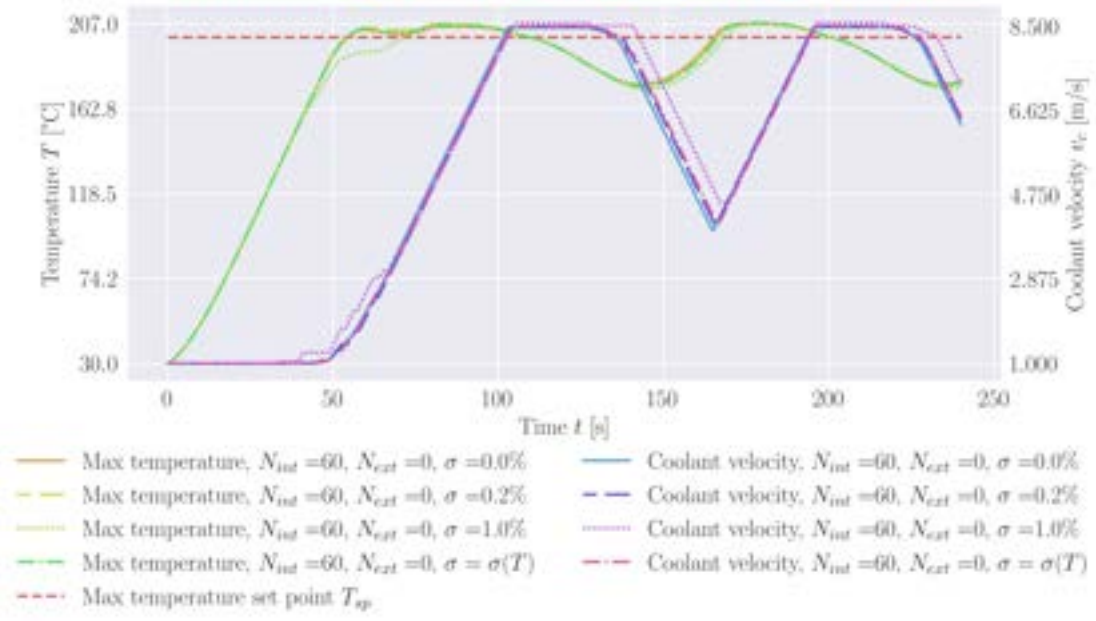


FIGURE C.19: PD-ML DT system response Sine wave 2 (Table 5.9 and Figure 5.21) for N_{int} equal to 60 and N_{ext} equal to 0; the overshoots are listed in Table C.6.

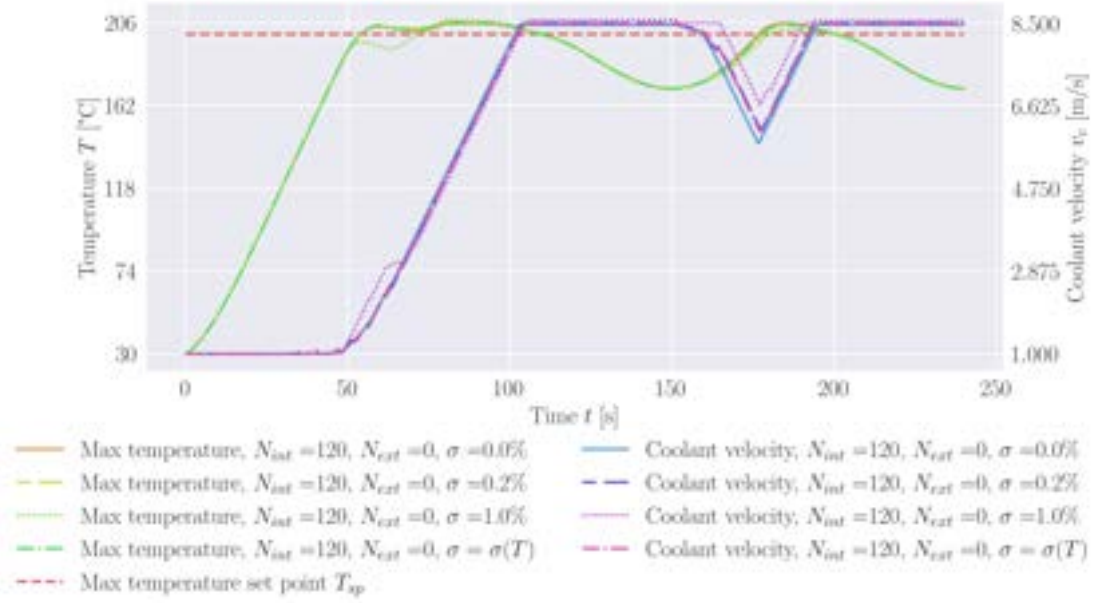


FIGURE C.20: PD-ML DT system response Sine wave 2 (Table 5.9 and Figure 5.21) for N_{int} equal to 120 and N_{ext} equal to 0; the overshoots are listed in Table C.6.

C.3.2 Finite Element-based Digital Twinning control with Reduced Order Modelling

Table C.8 shows the overshoots and control run times as a response to Triangular wave heat flux shape. Figures C.26, C.27, C.28, and C.29 display the corresponding system responses.

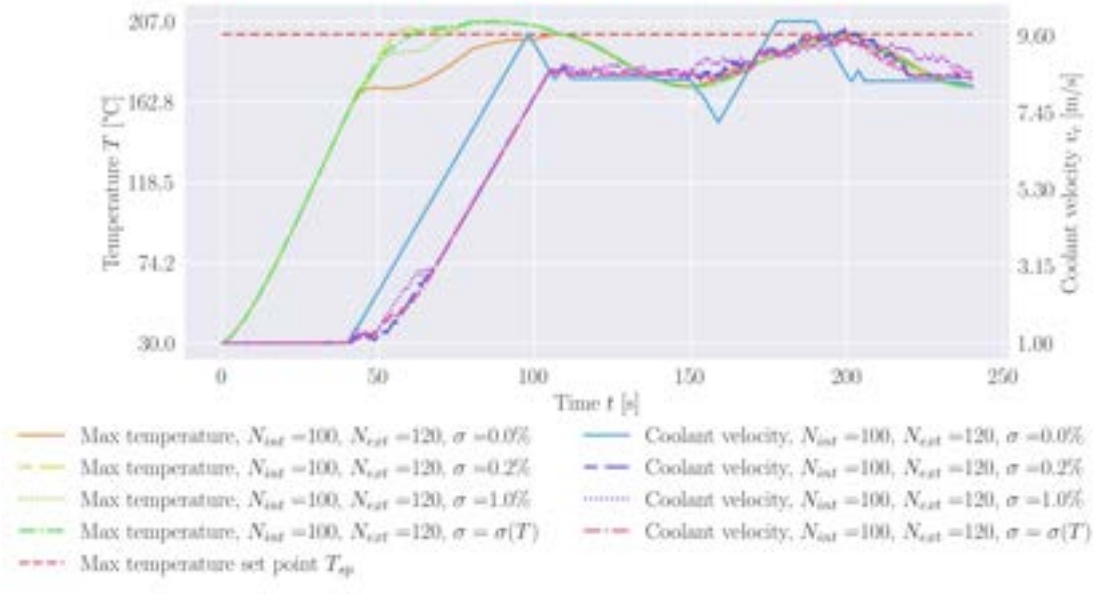


FIGURE C.21: PD-ML DT system response Sine wave 2 (Table 5.9 and Figure 5.21) for N_{int} equal to 100 and N_{ext} equal to 120; the overshoots are listed in Table C.6.

TABLE C.7: System response under FE DT control without ROM and average and maximum time step runtimes; applied heat flux $q(t)$ is in the form of Triangular wave (Table 5.9 and Figure 5.21).

		Overshoot [%]		Sol. rec. + Control time		Sol. rec.	
No. of measurement locations	Noise level σ [%]	Primary	Secondary	Avg. time step runtime [s] [a]	Max. time step runtime [s] [a]	Avg. relative error [%]	Max. relative error [%]
4	0.0	3.158	3.158	2.289e+00	2.554e+00	0.212	2.192
4	0.2	2.881	2.881	2.283e+00	2.470e+00	0.261	4.855
4	1.0	15.125	12.898	2.287e+00	2.675e+00	0.697	24.51
4	$\sigma(T)$	4.515	4.515	2.283e+00	2.650e+00	0.328	8.109
8	0.0	3.437	3.437	2.284e+00	2.563e+00	0.205	2.352
8	0.2	3.996	3.996	2.290e+00	2.531e+00	0.245	5.825
8	1.0	9.49	6.22	2.285e+00	2.481e+00	0.552	25.842
8	$\sigma(T)$	3.161	2.8	2.288e+00	2.471e+00	0.292	10.072
11	0.0	3.714	3.714	2.289e+00	2.575e+00	0.173	4.871
11	0.2	0.992	0.992	2.307e+00	2.495e+00	0.195	5.629
11	1.0	6.028	5.29	2.283e+00	2.522e+00	0.46	22.262
11	$\sigma(T)$	3.789	3.45	2.288e+00	2.594e+00	0.26	7.248
17	0.0	2.976	2.976	2.283e+00	2.474e+00	0.164	5.629
17	0.2	4.35	4.35	2.287e+00	2.500e+00	0.196	6.21
17	1.0	14.324	10.078	2.290e+00	2.509e+00	0.489	21.744
17	$\sigma(T)$	4.843	4.843	2.279e+00	2.532e+00	0.239	8.223

[a] Times are given for AMD Ryzen 7 5800X 8-Core CPU.

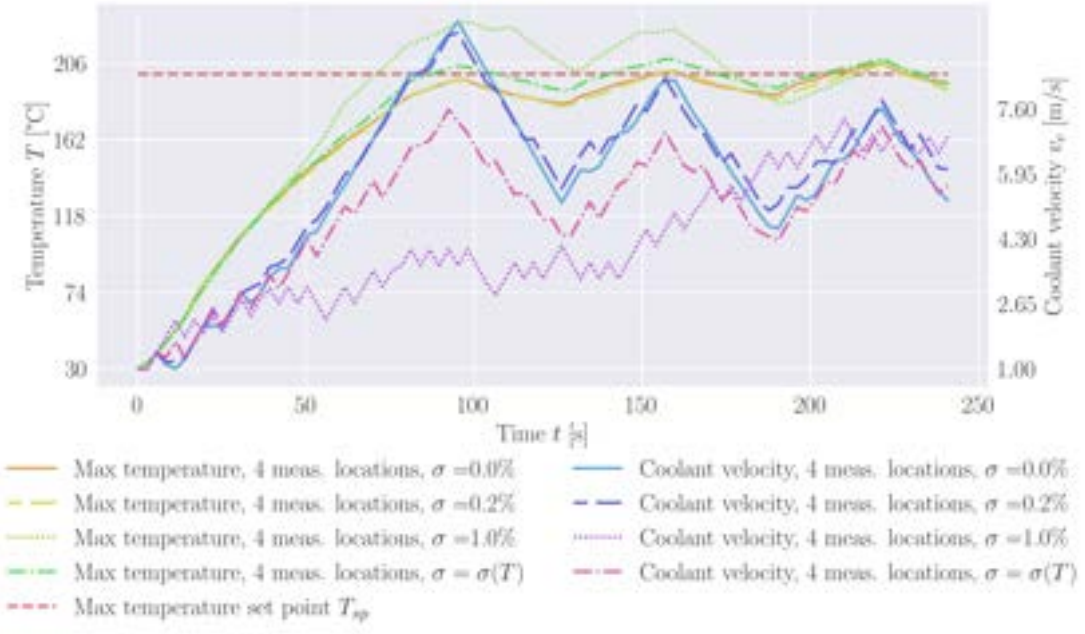


FIGURE C.22: FE DT without ROM system response Triangular wave (Table 5.9 and Figure 5.21) for 4 measurement locations; the overshoots are listed in Table C.1.

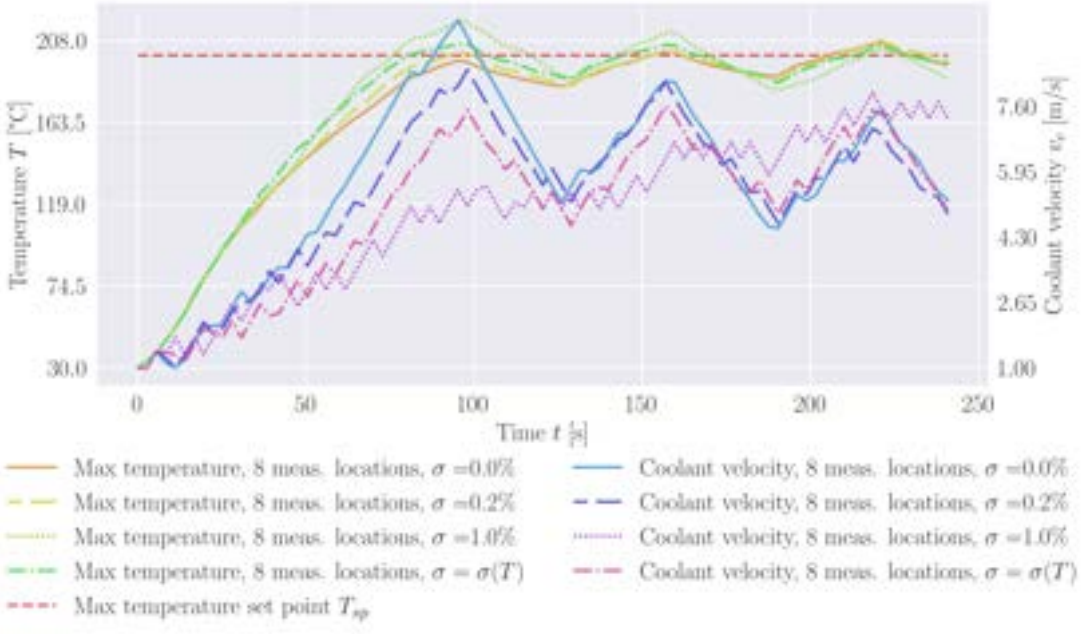


FIGURE C.23: FE DT without ROM system response Triangular wave (Table 5.9 and Figure 5.21) for 8 measurement locations; the overshoots are listed in Table C.1.

C.3.3 Physics-Driven Machine Learning-based Digital Twinning control

Table C.9 shows the overshoots and control run times as a response to Sine wave 2 heat flux shape. Figures C.30, C.31, C.32, and C.33 display the corresponding system

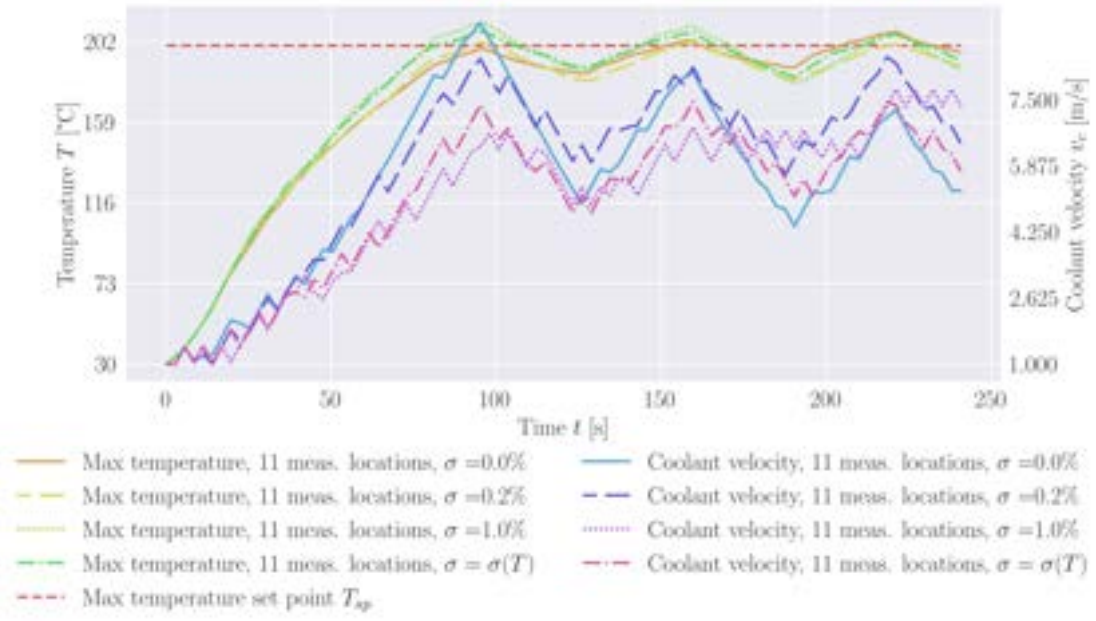


FIGURE C.24: FE DT without ROM system response Triangular wave (Table 5.9 and Figure 5.21) for 11 measurement locations; the overshoots are listed in Table C.1.

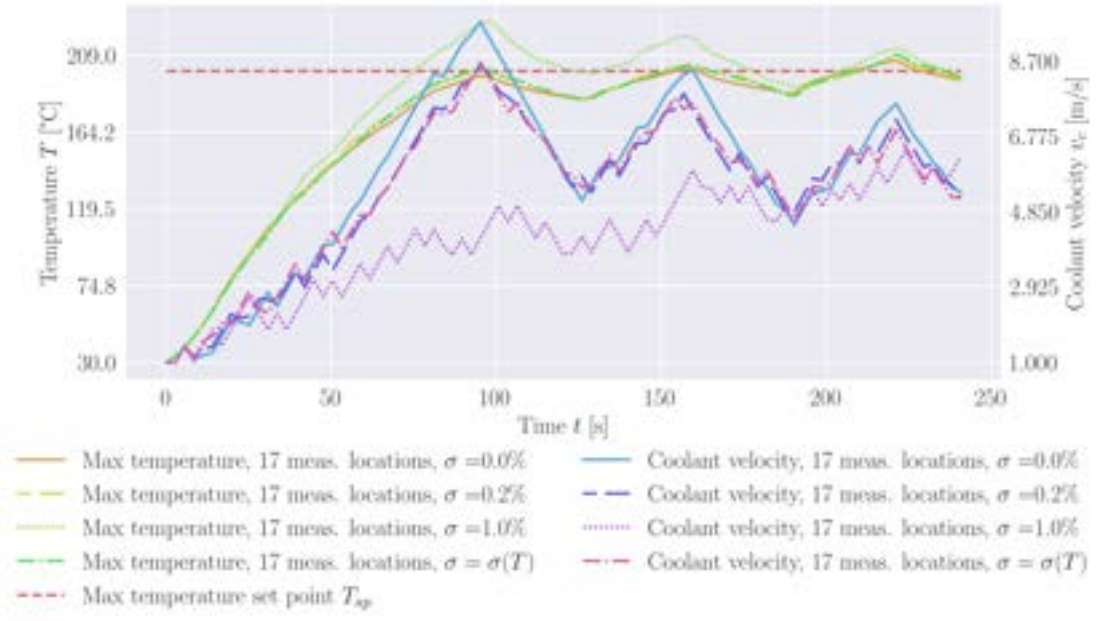


FIGURE C.25: FE DT without ROM system response Triangular wave (Table 5.9 and Figure 5.21) for 17 measurement locations; the overshoots are listed in Table C.1.

responses.

TABLE C.8: System response under FE DT control with ROM and average and maximum time step runtimes; applied heat flux $q(t)$ is in the form of Triangular wave (Table 5.9 and Figure 5.21).

		Overshoot [%]		Sol. rec. + Control time		Sol. rec.	
No. of measurement locations	Noise level σ [%]	Primary	Secondary	Avg. time step runtime [s] [a]	Max. time step runtime [s] [a]	Avg. relative error [%]	Max. relative error [%]
4	0.0	-1.987	2.835	3.035e-01	3.973e-01	0.765	3.875
4	0.2	0.243	4.205	2.897e-01	3.718e-01	0.826	4.737
4	1.0	22.387	19.609	3.114e-01	4.018e-01	1.266	22.903
4	$\sigma(T)$	2.701	3.723	2.931e-01	4.219e-01	0.869	7.354
8	0.0	-1.928	2.964	2.933e-01	4.004e-01	0.709	4.24
8	0.2	2.086	5.524	3.045e-01	3.989e-01	0.761	5.558
8	1.0	15.219	12.97	3.137e-01	4.571e-01	0.98	24.815
8	$\sigma(T)$	4.023	2.396	3.141e-01	4.085e-01	0.783	9.573
11	0.0	3.205	3.724	2.874e-01	4.325e-01	0.623	6.815
11	0.2	6.053	4.84	3.204e-01	4.293e-01	0.623	6.945
11	1.0	14.336	12.18	3.126e-01	4.184e-01	0.936	26.552
11	$\sigma(T)$	6.492	6.068	3.183e-01	4.418e-01	0.663	9.917
17	0.0	3.279	3.634	3.139e-01	4.187e-01	0.652	9.607
17	0.2	2.657	7.568	3.161e-01	4.680e-01	0.671	11.998
17	1.0	11.618	12.168	3.107e-01	3.947e-01	0.9	27.681
17	$\sigma(T)$	4.732	7.029	3.198e-01	4.257e-01	0.706	14.806

[a] Times are given for AMD Ryzen 7 5800X 8-Core CPU.

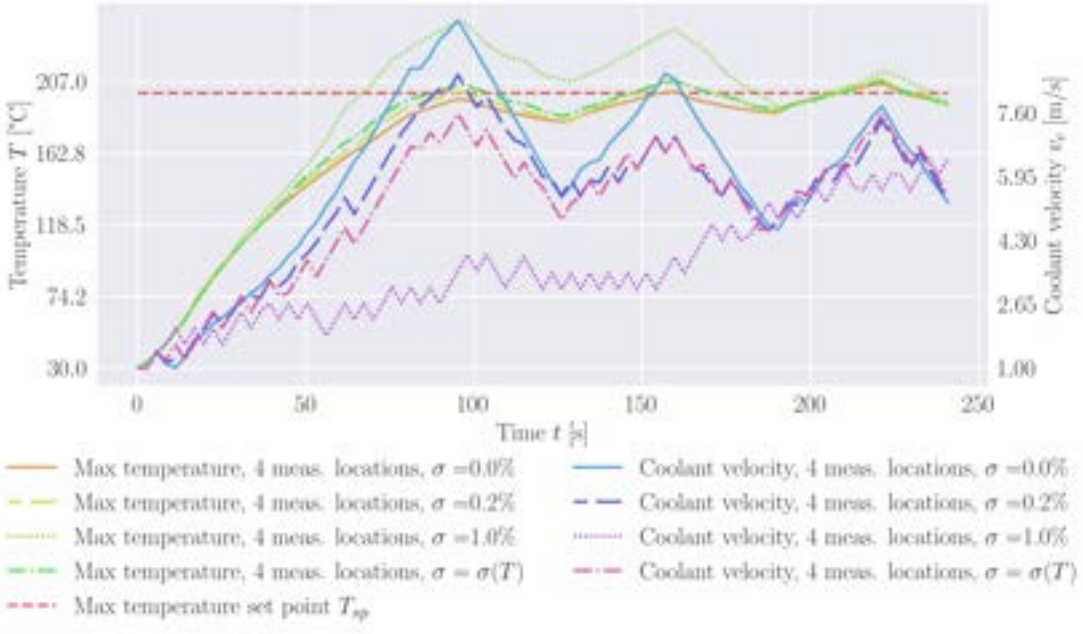


FIGURE C.26: FE DT with ROM system response Triangular wave (Table 5.9 and Figure 5.21) for 4 measurement locations; the overshoots are listed in Table C.1.

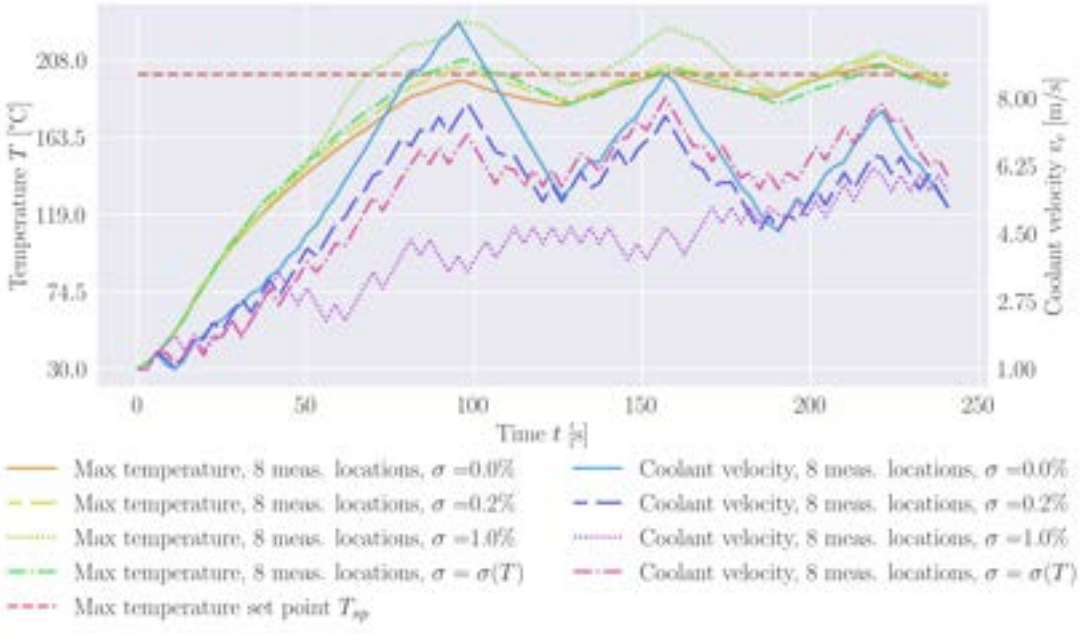


FIGURE C.27: FE DT with ROM system response Triangular wave (Table 5.9 and Figure 5.21) for 8 measurement locations; the overshoots are listed in Table C.1.

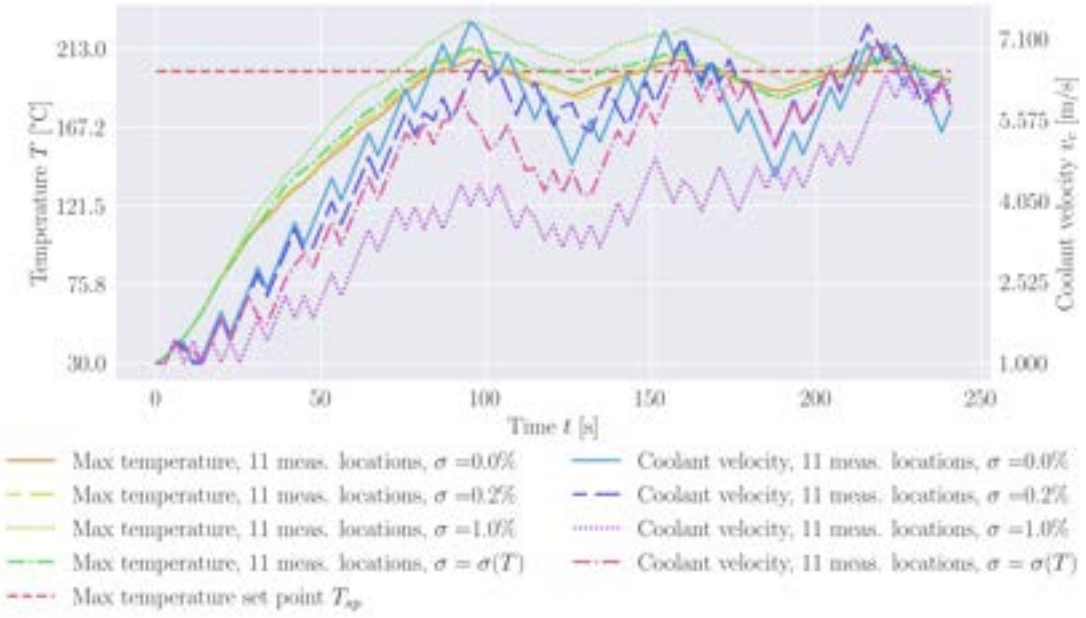


FIGURE C.28: FE DT with ROM system response Triangular wave (Table 5.9 and Figure 5.21) for 11 measurement locations; the overshoots are listed in Table C.1.

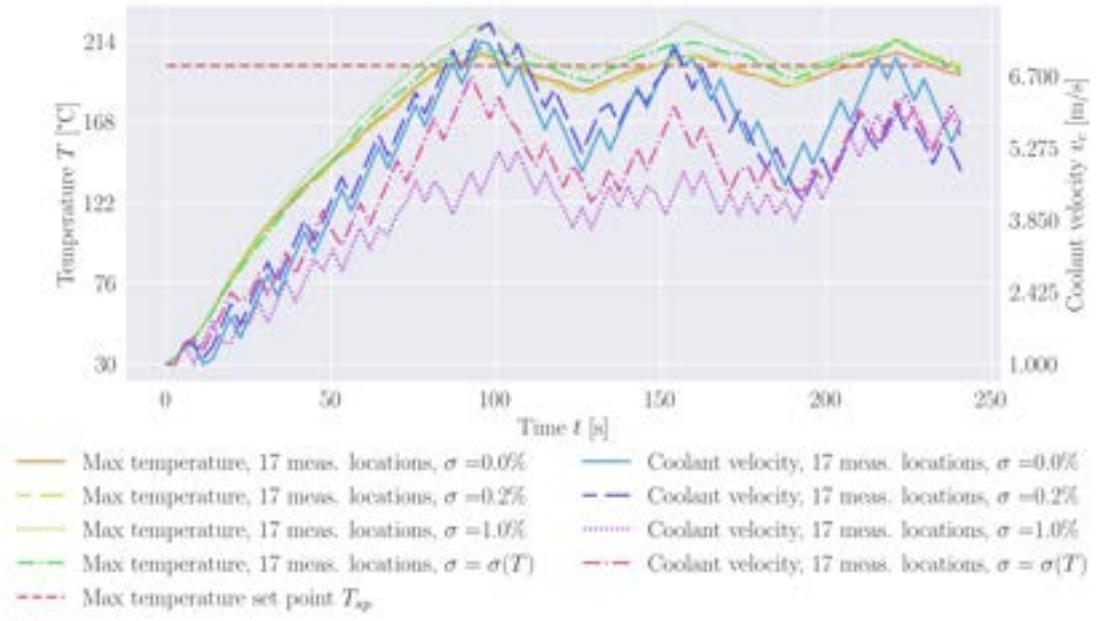


FIGURE C.29: FE DT with ROM system response Triangular wave (Table 5.9 and Figure 5.21) for 17 measurement locations; the overshoots are listed in Table C.1.

TABLE C.9: PD-ML DT system response under PD-ML DT control and average and maximum time step runtimes; applied heat flux $q(t)$ is in the form of Triangular wave (Table 5.9 and Figure 5.21).

Parameter set			Overshoot [%]		Control time	
N_{int}	N_{ext}	Noise level σ [%]	Primary	Secondary	Avg. time step runtime [s] [a]	Max. time step runtime [s] [a]
60	0	0.0	3.897	3.897	3.701e-04	5.932e-04
60	0	0.2	3.968	3.968	3.830e-04	8.121e-04
60	0	1.0	3.851	3.159	3.596e-04	6.039e-04
60	0	$\sigma(T)$	3.91	3.91	3.613e-04	7.958e-04
120	0	0.0	3.352	1.866	3.694e-04	5.851e-04
120	0	0.2	3.629	1.689	3.622e-04	6.032e-04
120	0	1.0	3.952	0.805	3.633e-04	6.013e-04
120	0	$\sigma(T)$	3.573	1.843	3.615e-04	5.848e-04
60	120	0.0	-2.38	-2.383	1.045e-03	1.546e-03
60	120	0.2	3.929	-0.488	1.067e-03	1.554e-03
60	120	1.0	3.664	-0.563	1.037e-03	1.513e-03
60	120	$\sigma(T)$	3.478	-0.592	1.093e-03	1.562e-03
100	120	0.0	-0.65	-2.392	1.889e-03	3.437e-03
100	120	0.2	3.75	-1.617	2.258e-03	3.479e-03
100	120	1.0	3.66	-0.88	2.050e-03	3.130e-03
100	120	$\sigma(T)$	3.799	-1.368	1.840e-03	3.182e-03

[a] Times are given for AMD Ryzen 7 5800X 8-Core CPU.

C.4 Heat flux in the form of Filtered Gaussian noise

C.4.1 Finite Element-based Digital Twinning control without Reduced Order Modelling

Table C.10 shows the overshoots and control run times as a response to Filtered Gaussian noise heat flux shape. Figures C.34, C.35, C.36, and C.37 display the corresponding

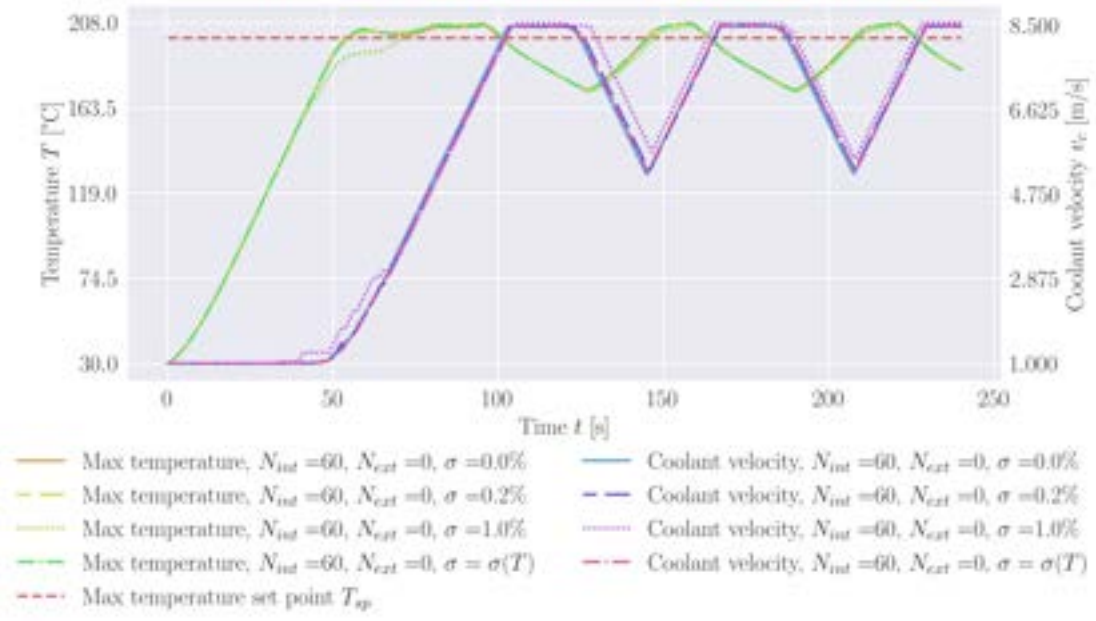


FIGURE C.30: PD-ML DT system response Triangular wave (Table 5.9 and Figure 5.21) for N_{int} equal to 60 and N_{ext} equal to 0; the overshoots are listed in Table C.9.

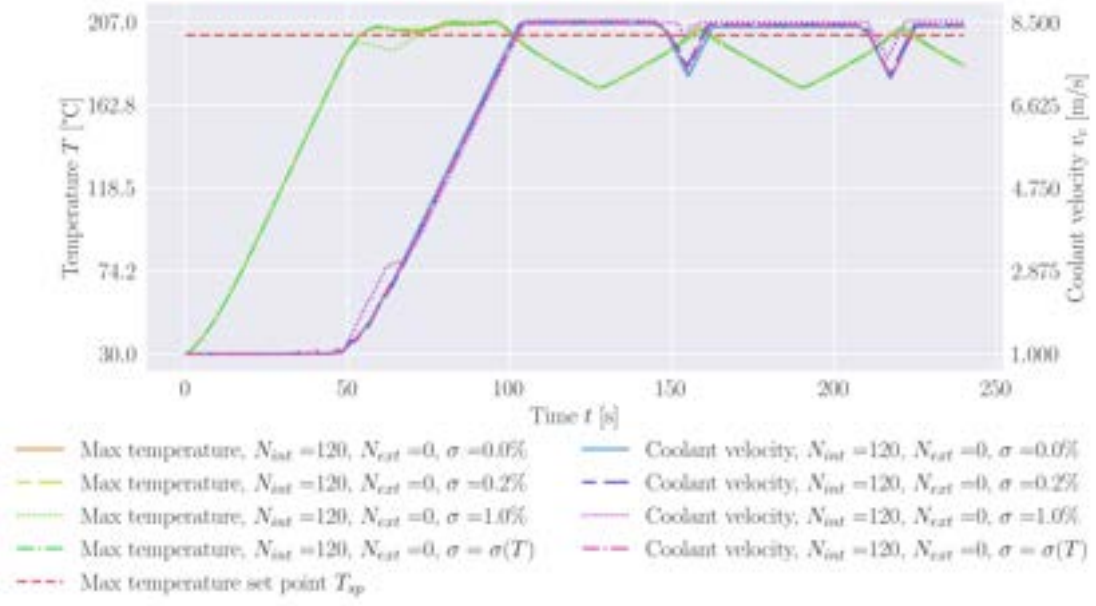


FIGURE C.31: PD-ML DT system response Triangular wave (Table 5.9 and Figure 5.21) for N_{int} equal to 120 and N_{ext} equal to 0; the overshoots are listed in Table C.9.

system responses.

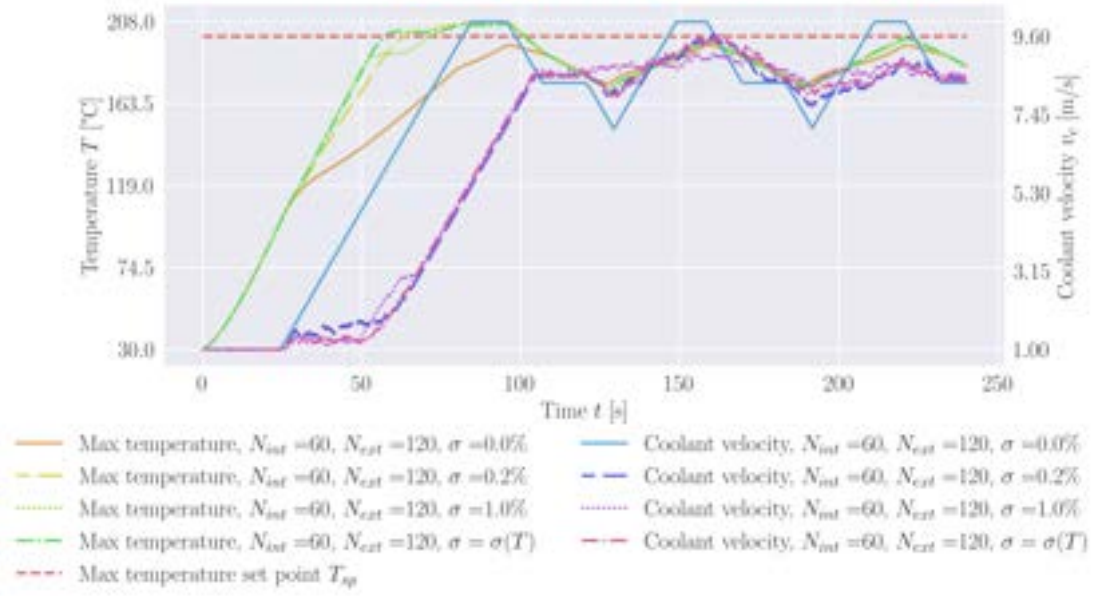


FIGURE C.32: PD-ML DT system response Triangular wave (Table 5.9 and Figure 5.21) for N_{int} equal to 60 and N_{ext} equal to 120; the overshoots are listed in Table C.9.

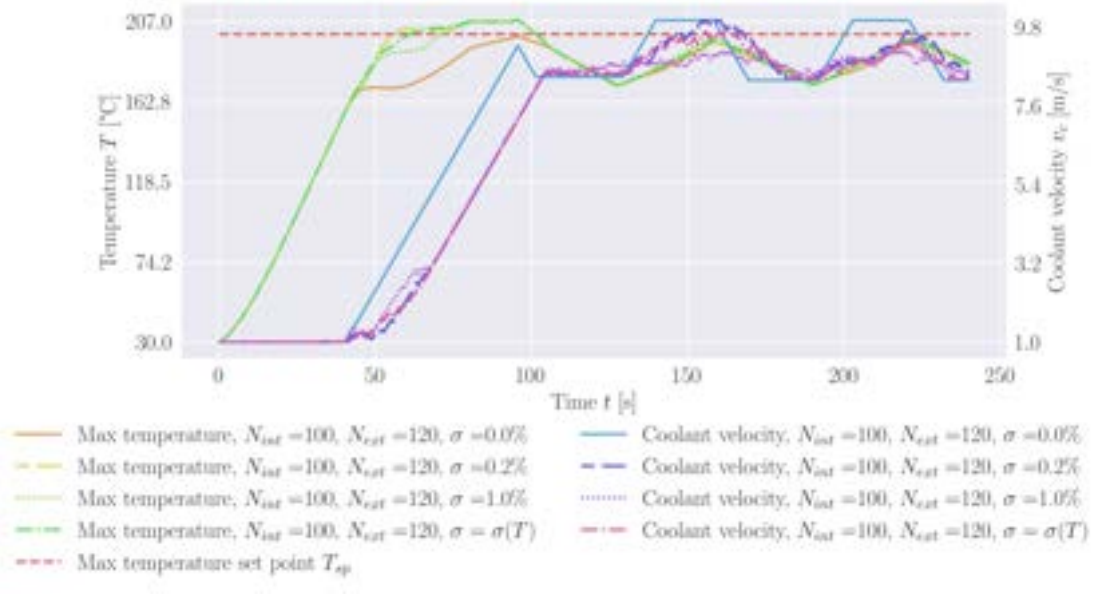


FIGURE C.33: PD-ML DT system response Triangular wave (Table 5.9 and Figure 5.21) for N_{int} equal to 100 and N_{ext} equal to 120; the overshoots are listed in Table C.9.

C.4.2 Finite Element-based Digital Twinning control with Reduced Order Modelling

Table C.11 shows the overshoots and control run times as a response to Filtered Gaussian noise heat flux shape. Figures C.38, C.39, C.40, and C.41 display the corresponding

TABLE C.10: System response under FE DT control without ROM and average and maximum time step runtimes; applied heat flux $q(t)$ is in the form of Filtered Gaussian noise (Table 5.9 and Figure 5.21).

		Overshoot [%]		Sol. rec. + Control time		Sol. rec.	
No. of measurement locations	Noise level σ [%]	Primary	Secondary	Avg. time step runtime [s] [a]	Max. time step runtime [s] [a]	Avg. relative error [%]	Max. relative error [%]
4	0.0	0.276	0.276	2.279e+00	2.525e+00	0.224	2.192
4	0.2	3.954	3.954	2.270e+00	2.560e+00	0.279	5.055
4	1.0	12.903	11.937	2.299e+00	2.544e+00	0.7	25.083
4	$\sigma(T)$	7.771	7.771	2.307e+00	2.585e+00	0.364	8.109
8	0.0	0.297	0.297	2.285e+00	2.423e+00	0.216	2.352
8	0.2	4.404	4.404	2.270e+00	2.540e+00	0.285	5.825
8	1.0	8.566	2.929	2.281e+00	2.514e+00	0.563	25.853
8	$\sigma(T)$	8.471	8.471	2.284e+00	2.461e+00	0.34	10.072
11	0.0	-0.115	-0.115	2.282e+00	2.485e+00	0.188	4.871
11	0.2	0.933	0.933	2.276e+00	2.602e+00	0.202	5.629
11	1.0	4.47	4.304	2.285e+00	2.513e+00	0.466	21.915
11	$\sigma(T)$	4.122	4.122	2.270e+00	2.463e+00	0.269	7.248
17	0.0	-0.219	-0.219	2.280e+00	2.498e+00	0.18	5.629
17	0.2	2.203	2.203	2.281e+00	2.490e+00	0.215	6.21
17	1.0	9.523	6.815	2.293e+00	2.468e+00	0.491	21.744
17	$\sigma(T)$	2.597	2.597	2.280e+00	2.520e+00	0.251	8.223

[a] Times are given for AMD Ryzen 7 5800X 8-Core CPU.

system responses.

C.4.3 Physics-Driven Machine Learning-based Digital Twinning control

Table C.12 shows the overshoots and control run times as a response to Filtered Gaussian noise heat flux shape. Figures C.42, C.31, C.44, and C.45 display the corresponding system responses.

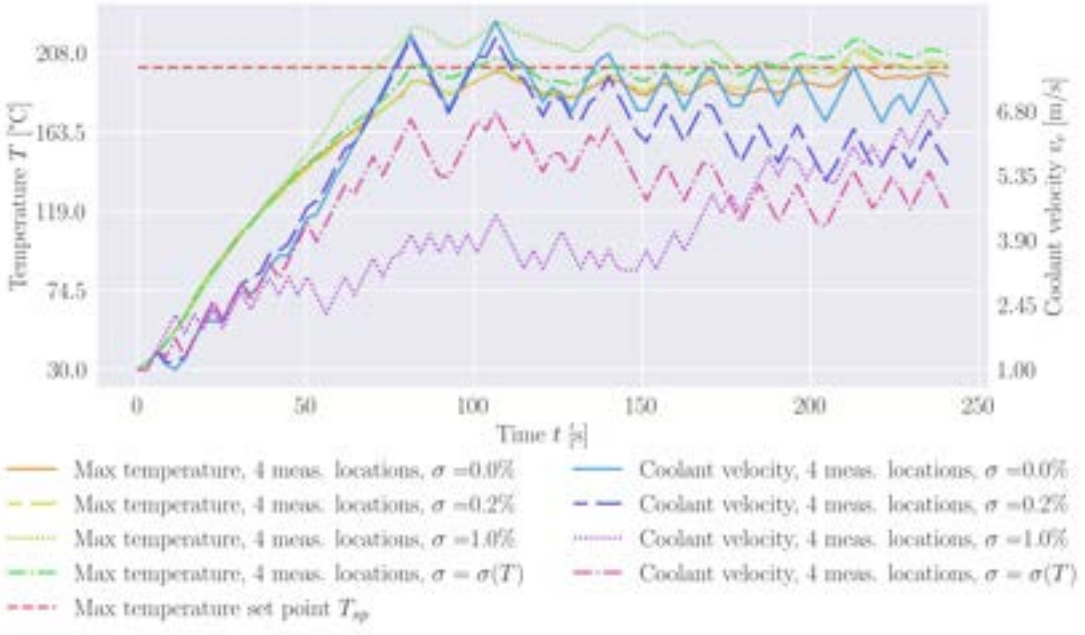


FIGURE C.34: FE DT without ROM system response Filtered Gaussian noise (Table 5.9 and Figure 5.21) for 4 measurement locations; the overshoots are listed in Table C.1.

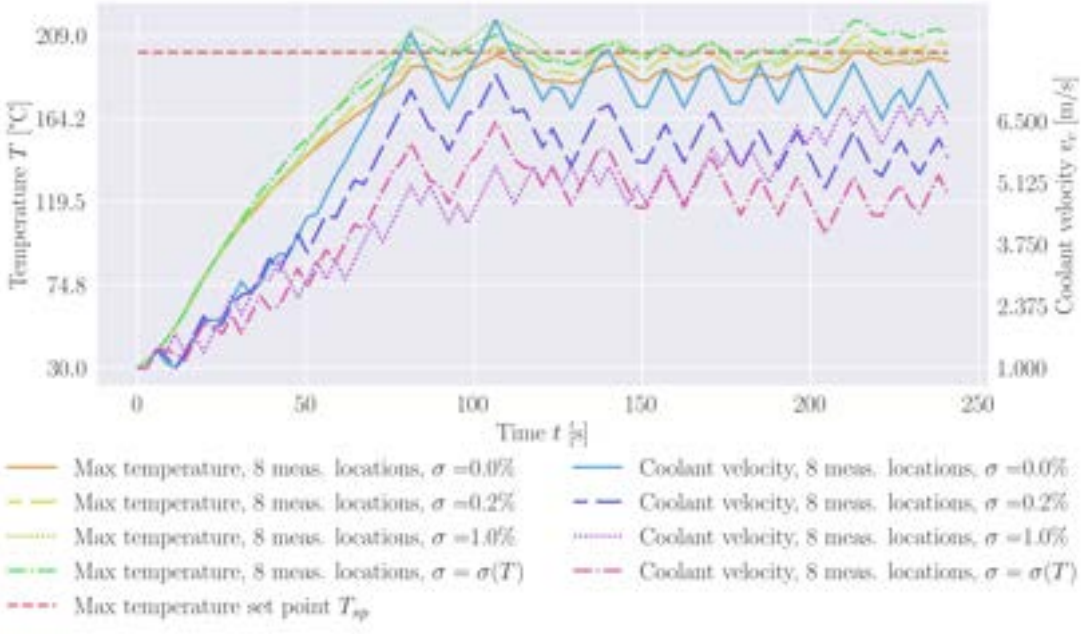


FIGURE C.35: FE DT without ROM system response Filtered Gaussian noise (Table 5.9 and Figure 5.21) for 8 measurement locations; the overshoots are listed in Table C.1.

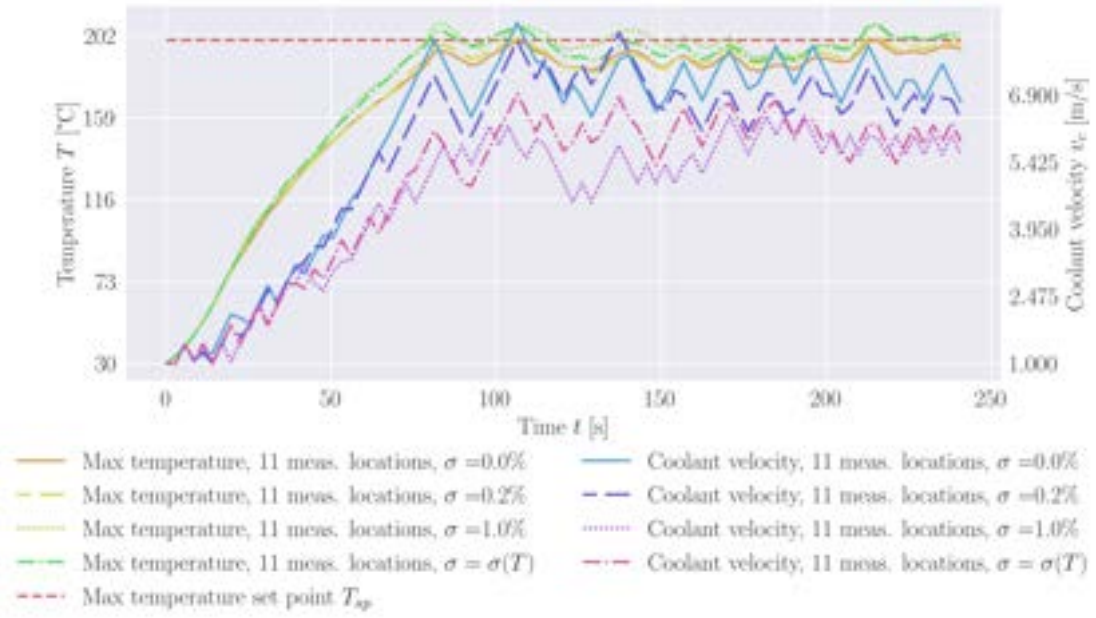


FIGURE C.36: FE DT without ROM system response Filtered Gaussian noise (Table 5.9 and Figure 5.21) for 11 measurement locations; the overshoots are listed in Table C.1.

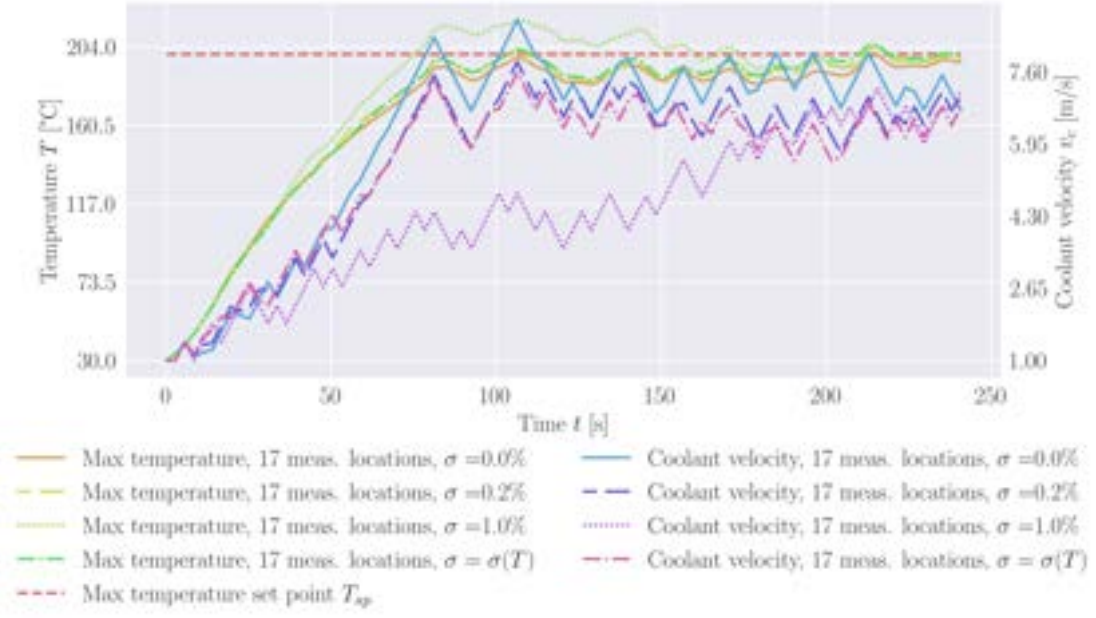


FIGURE C.37: FE DT without ROM system response Filtered Gaussian noise (Table 5.9 and Figure 5.21) for 17 measurement locations; the overshoots are listed in Table C.1.

TABLE C.11: System response under FE DT control with ROM and average and maximum time step runtimes; applied heat flux $q(t)$ is in the form of Filtered Gaussian noise (Table 5.9 and Figure 5.21).

		Overshoot [%]		Sol. rec. + Control time		Sol. rec.	
No. of measurement locations	Noise level σ [%]	Primary	Secondary	Avg. time step runtime [s] [a]	Max. time step runtime [s] [a]	Avg. relative error [%]	Max. relative error [%]
4	0.0	-1.596	1.165	3.043e-01	3.912e-01	0.782	4.259
4	0.2	1.814	4.954	3.116e-01	4.068e-01	0.873	4.955
4	1.0	17.416	12.263	3.194e-01	3.966e-01	1.24	23.084
4	$\sigma(T)$	17.416	12.263	3.194e-01	3.966e-01	1.24	23.084
8	0.0	-1.583	1.354	2.808e-01	3.560e-01	0.718	4.693
8	0.2	0.672	2.365	2.419e-01	3.629e-01	0.761	5.558
8	1.0	12.397	8.485	2.840e-01	3.733e-01	0.981	24.96
8	$\sigma(T)$	2.912	4.608	2.257e-01	2.828e-01	0.858	9.573
11	0.0	4.805	5.154	3.141e-01	3.936e-01	0.65	6.815
11	0.2	5.303	7.981	3.117e-01	3.832e-01	0.66	6.945
11	1.0	10.569	10.085	3.150e-01	4.110e-01	0.892	25.365
11	$\sigma(T)$	5.92	3.523	3.153e-01	4.148e-01	0.69	9.917
17	0.0	4.518	4.499	3.164e-01	4.088e-01	0.677	9.607
17	0.2	3.466	4.68	3.185e-01	3.971e-01	0.674	11.998
17	1.0	10.952	7.774	3.195e-01	4.416e-01	0.901	27.681
17	$\sigma(T)$	3.663	8.505	3.180e-01	4.690e-01	0.721	14.806

[a] Times are given for AMD Ryzen 7 5800X 8-Core CPU.

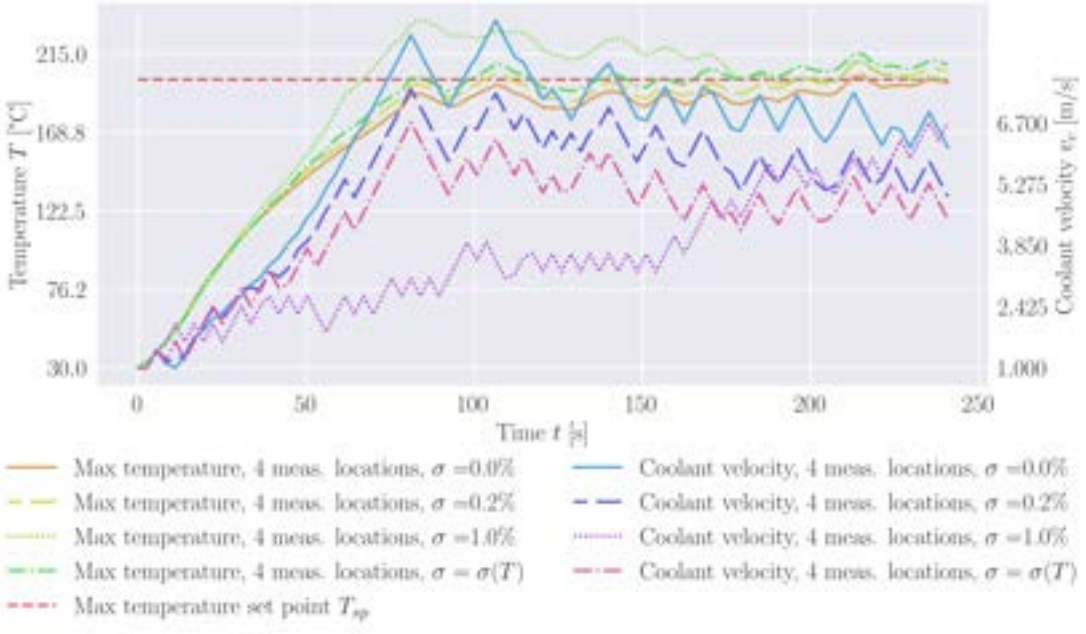


FIGURE C.38: FE DT with ROM system response Filtered Gaussian noise (Table 5.9 and Figure 5.21) for 4 measurement locations; the overshoots are listed in Table C.1.

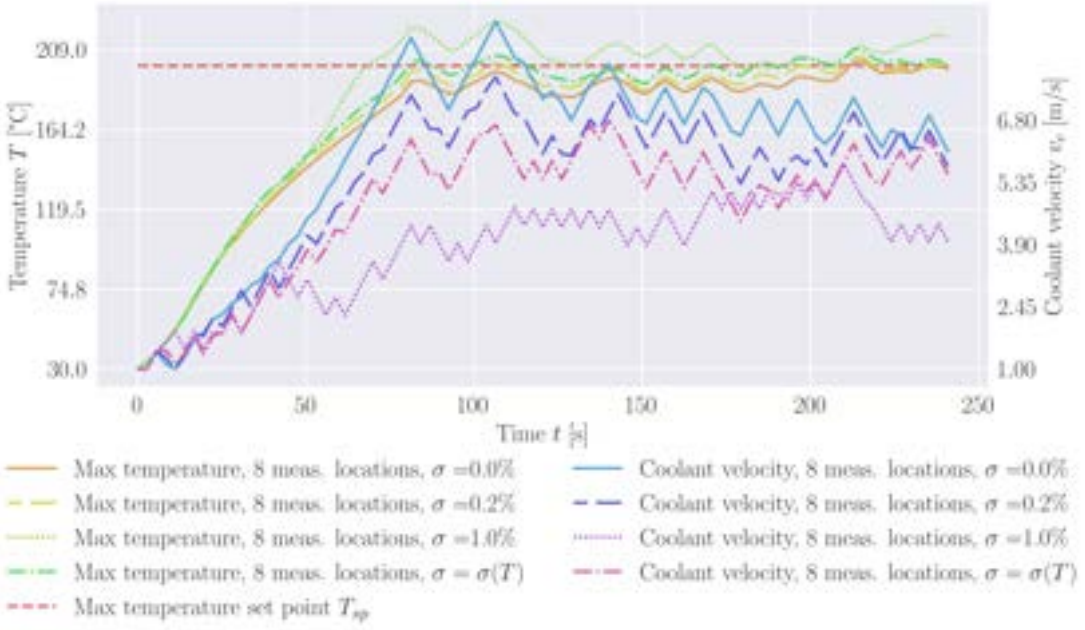


FIGURE C.39: FE DT with ROM system response Filtered Gaussian noise (Table 5.9 and Figure 5.21) for 8 measurement locations; the overshoots are listed in Table C.1.

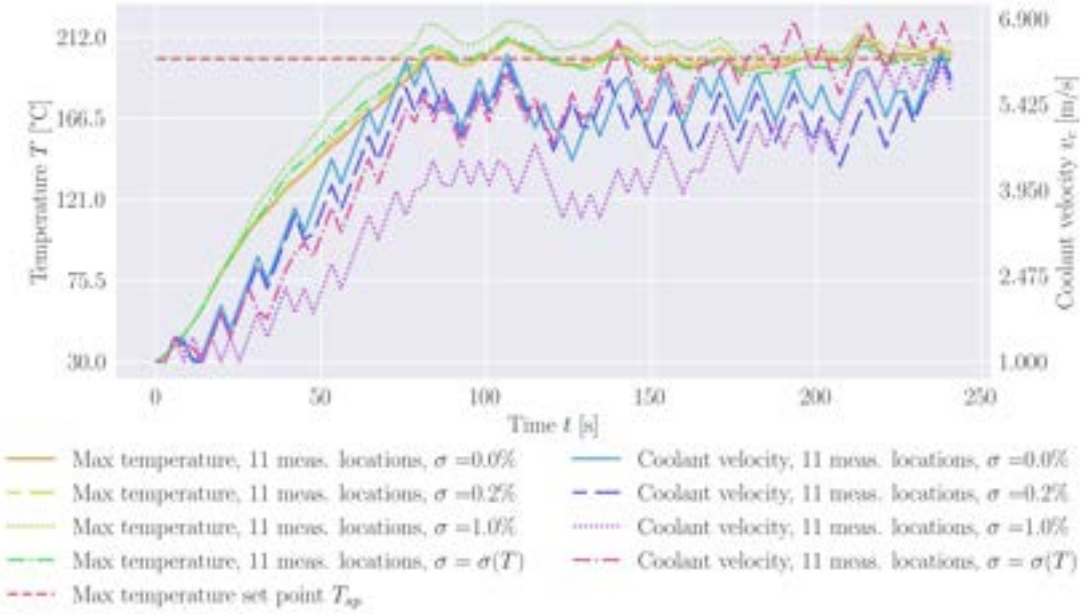


FIGURE C.40: FE DT with ROM system response Filtered Gaussian noise (Table 5.9 and Figure 5.21) for 11 measurement locations; the overshoots are listed in Table C.1.

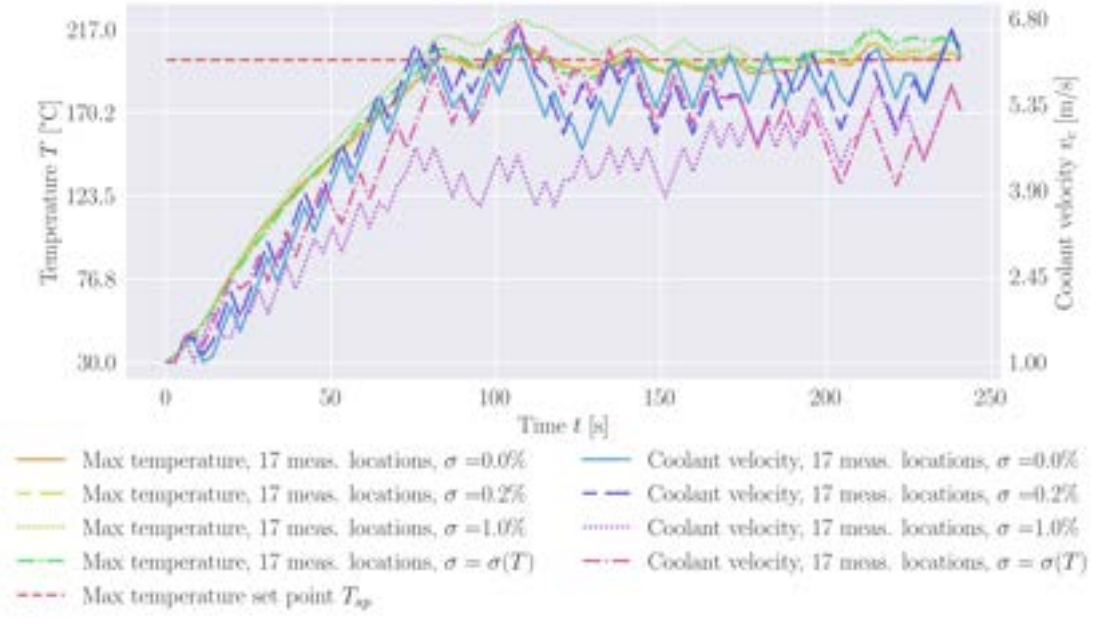


FIGURE C.41: FE DT with ROM system response Filtered Gaussian noise (Table 5.9 and Figure 5.21) for 17 measurement locations; the overshoots are listed in Table C.1.

TABLE C.12: PD-ML DT system response under PD-ML DT control and average and maximum time step runtimes; applied heat flux $q(t)$ is in the form of Filtered Gaussian noise (Table 5.9 and Figure 5.21).

Parameter set			Overshoot [%]		Control time	
N_{int}	N_{ext}	Noise level σ [%]	Primary	Secondary	Avg. time step runtime [s] [a]	Max. time step runtime [s] [a]
60	0	0.0	5.183	3.73	3.710e-04	6.251e-04
60	0	0.2	5.666	3.632	3.825e-04	9.189e-04
60	0	1.0	5.948	1.897	3.559e-04	5.898e-04
60	0	$\sigma(T)$	5.559	3.482	3.723e-04	5.958e-04
120	0	0.0	5.183	3.831	3.808e-04	6.118e-04
120	0	0.2	5.666	3.614	3.636e-04	5.972e-04
120	0	1.0	6.106	1.901	3.635e-04	6.015e-04
120	0	$\sigma(T)$	5.56	3.536	3.681e-04	6.106e-04
60	120	0.0	-0.074	-0.074	1.006e-03	1.528e-03
60	120	0.2	6.185	-0.87	1.103e-03	1.727e-03
60	120	1.0	5.643	-2.56	1.015e-03	1.714e-03
60	120	$\sigma(T)$	5.399	-1.312	1.166e-03	1.559e-03
100	120	0.0	-0.098	-0.098	1.860e-03	3.108e-03
100	120	0.2	5.855	-0.872	2.165e-03	3.344e-03
100	120	1.0	5.637	-2.56	2.135e-03	3.193e-03
100	120	$\sigma(T)$	5.962	-1.299	1.817e-03	3.124e-03

[a] Times are given for AMD Ryzen 7 5800X 8-Core CPU.

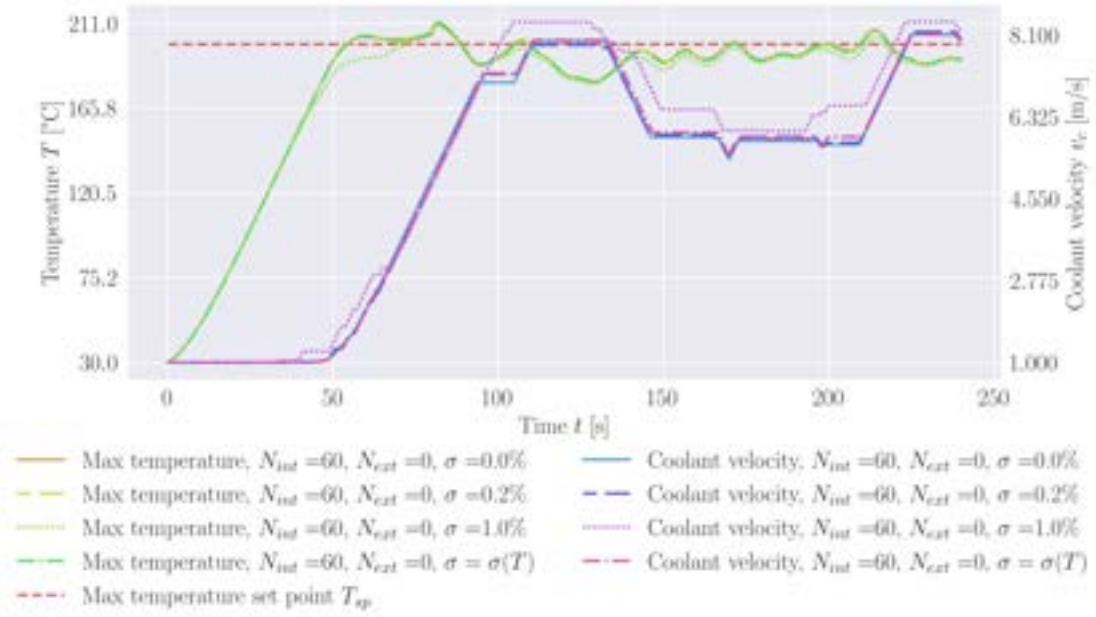


FIGURE C.42: PD-ML DT system response Filtered Gaussian noise (Table 5.9 and Figure 5.21) for N_{int} equal to 60 and N_{ext} equal to 0; the overshoots are listed in Table C.12.

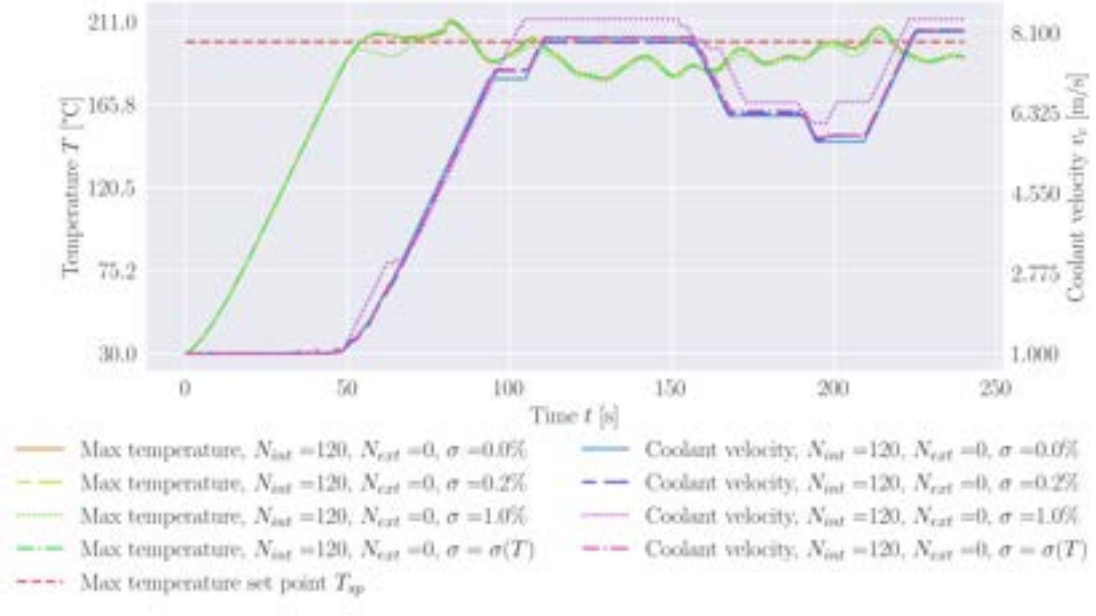


FIGURE C.43: PD-ML DT system response Filtered Gaussian noise (Table 5.9 and Figure 5.21) for N_{int} equal to 120 and N_{ext} equal to 0; the overshoots are listed in Table C.12.

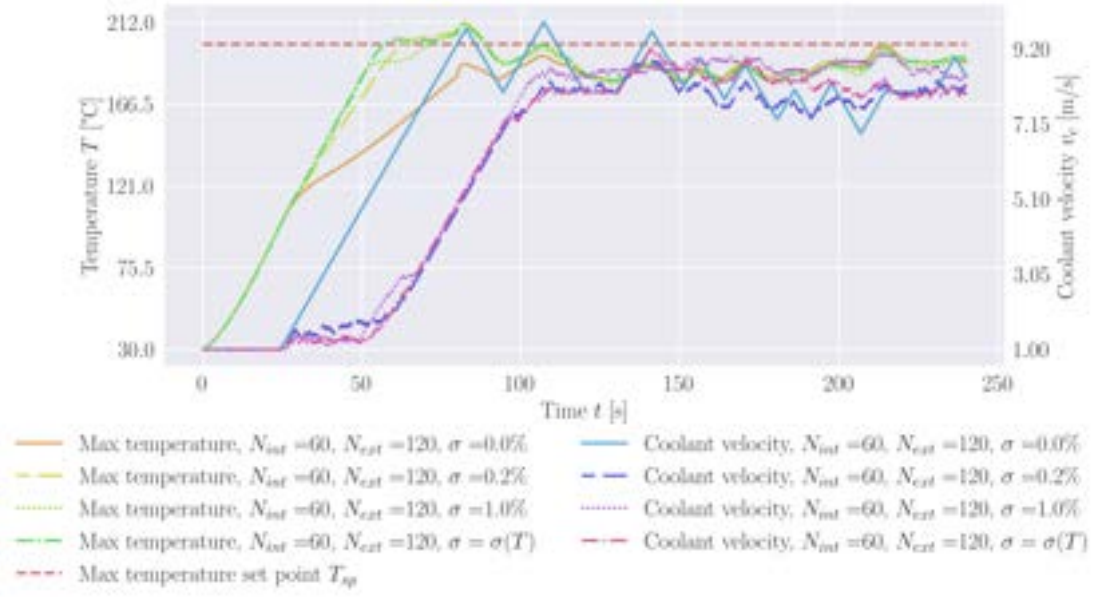


FIGURE C.44: PD-ML DT system response Filtered Gaussian noise (Table 5.9 and Figure 5.21) for N_{int} equal to 60 and N_{ext} equal to 120; the overshoots are listed in Table C.12.

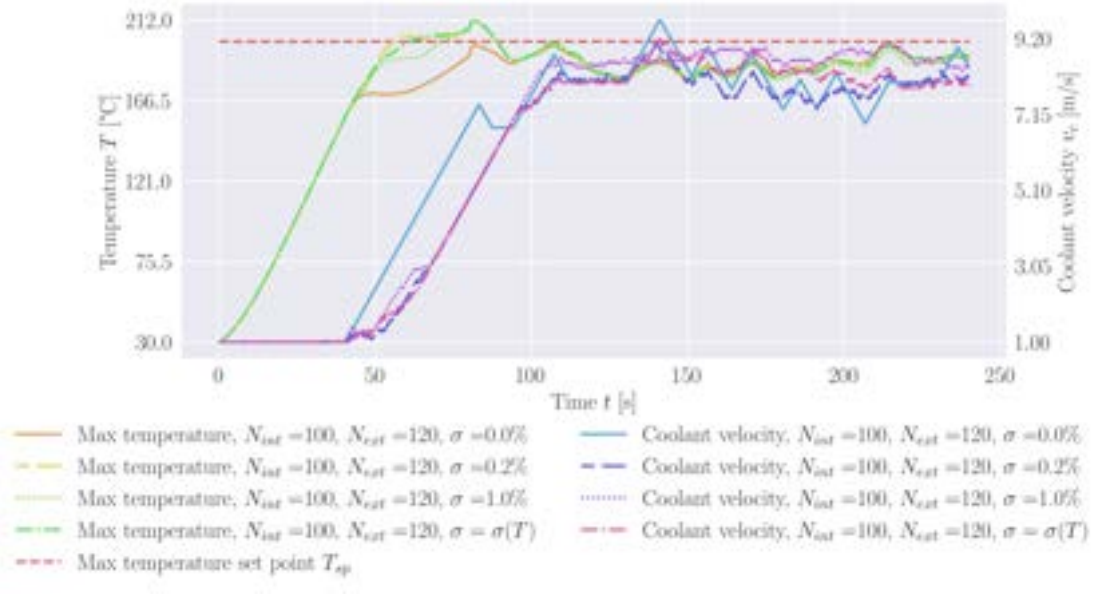


FIGURE C.45: PD-ML DT system response Filtered Gaussian noise (Table 5.9 and Figure 5.21) for N_{int} equal to 100 and N_{ext} equal to 120; the overshoots are listed in Table C.12.

Bibliography

[1] Guido Van Oost, editor. *Fundamentals of Magnetic Fusion Technology*. Non-serial Publications. International Atomic Energy Agency, Vienna, 2023. ISBN 978-92-0-110721-3.

[2] Yunus Cengel. *Heat and mass transfer: A practical approach*. Boston: McGraw-Hill, third edition, 2007.

[3] R. M. Park, editor. *Manual on the Use of Thermocouples in Temperature Measurement*. ASTM Committee E20, 4th edition, 1993. ISBN 0-8031-1466-4.

[4] Inc. COMSOL. Comsol multi-physics. Available at https://www.comsol.com/comsol-multiphysics, 2025. Accessed: 19/05/2025.

[5] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9:1735–80, 12 1997. doi: 10.1162/neco.1997.9.8.1735.

[6] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[7] EDF. Code_aster software. Available at https://code-aster.org/spip.php?rubrique2, 2025. Accessed: 19/05/2025.

[8] Richard A Dunlap. *Energy from Nuclear Fusion*. 2053-2563. IOP Publishing, 2021. ISBN 978-0-7503-3307-8. doi: 10.1088/978-0-7503-3307-8.

[9] John Wesson. *The Science of Jet*. JET Joint Undertaking, second edition, 2006.

[10] William Morris, J. R. Harrison, A. Kirk, B. Lipschultz, F. Militello, D. Moulton, and N. R. Walkden. Mast upgrade divertor facility: A test bed for novel divertor

solutions. *IEEE Transactions on Plasma Science*, 46(5):1217–1226, 2018. doi: 10.1109/TPS.2018.2815283.

[11] Cutler J. Cleveland, editor. *Encyclopedia of Energy*. Elsevier Science, 2004. ISBN 978-0-12-176480-7.

[12] UK Atomic Energy Authority. Jet's final tritium experiments yield new fusion energy record, 2024. URL https://www.gov.uk/government/news/jets-final-tritium-experiments-yield-new-fusion-energy-record. Accessed: 19/05/2025.

[13] Stuart I. Muldrew, Chris Harrington, Jonathan Keep, Chris Waldon, Christopher Ashe, Rhian Chapman, Charles Griesel, Alexander J. Pearce, Francis Casson, Stephen P. Marsden, and Emmi Tholerus. Conceptual design workflow for the step prototype powerplant. *Fusion Engineering and Design*, 201:114238, 2024. ISSN 0920-3796. doi: https://doi.org/10.1016/j.fusengdes.2024.114238. URL https://www.sciencedirect.com/science/article/pii/S0920379624000917.

[14] Sergio Orlandi. Iter project: International cooperation and energy investment. In Luciano Maiani, Raymond Jeanloz, Micah Lowenthal, and Wolfango Plastino, editors, *International Cooperation for Enhancing Nuclear Safety, Security, Safeguards and Non-proliferation*, pages 169–191, Cham, 2020. Springer International Publishing. ISBN 978-3-030-42913-3.

[15] National Ignition Facility. National ignition facility - 2022 annual report, 2022. URL https://annual.llnl.gov/fy-2022/national-ignition-facility-2022. Accessed: 19/05/2025.

[16] Jinxing Zheng, Jinggang Qin, Kun Lu, Min Xu, Xuru Duan, Guosheng Xu, Jiansheng Hu, Xianzu Gong, Qing Zang, Zhihong Liu, Liang Wang, Rui Ding, Jiming Chen, Pengyuan Li, Lei Xue, Lijun Cai, and Yuntao Song. Recent progress in chinese fusion research based on superconducting tokamak configuration. *The Innovation*, 3(4):100269, 2022. ISSN 2666-6758. doi: https://doi.org/10.1016/j.xinn.2022.100269.

[17] Tamás Szabolics. Eurofusion and asipp begins collaboration on best research plan, 2024. URL https://euro-fusion.org/eurofusion-news/

eurofusion-and-asipp-begins-collaboration-on-best-research-plan/. Accessed: 27/08/2025.

[18] David Hancock. *Employing Additive Manufacturing for Fusion High Heat Flux Structures*. Phd thesis, University of Sheffield, 2018.

[19] Rhydian Lewis. *Simulation driven machine learning methods to optimise design of physical experiments and enhance data analysis for testing of fusion energy heat exchanger components*. Phd thesis, Swansea University, 2023.

[20] Jonathan Pearl, James Paterson, Kieran Flinders, Nicolas Mantel, and Jeong-Ha You. Cyclic medium heat flux testing of a wta lattice structure on the hive facility. *Fusion Engineering and Design*, 194:113699, 2023. ISSN 0920-3796. doi: https://doi.org/10.1016/j.fusengdes.2023.113699.

[21] T.R. Barrett, M. Bamford, B. Chuilon, T. Deighan, P. Efthymiou, L. Fletcher, M. Gorley, T. Grant, T. Hall, D. Horsley, M. Kovari, and M. Tindall. The chimera facility development programme. *Fusion Engineering and Design*, 194:113689, 2023. ISSN 0920-3796. doi: https://doi.org/10.1016/j.fusengdes.2023.113689.

[22] Jun-Feng Yao, Yong Yang, Xue-Cheng Wang, and Xiao-Peng Zhang. Systematic review of digital twin technology and applications. *Visual Computing for Industry, Biomedicine, and Art*, 6(1):10, 5 2023. ISSN 2524-4442. doi: 10.1186/s42492-023-00137-4.

[23] Albert Tarantola. *Inverse Problem Theory and Methods for Model Parameter Estimation*. Society for Industrial and Applied Mathematics, 2004. ISBN 0898715725.

[24] Simon Arridge, Peter Maass, Ozan Öktem, and Carola-Bibiane Schönlieb. Solving inverse problems using data-driven models. *Acta Numerica*, 28:1–174, 2019. ISSN 0962-4929. doi: 10.1017/S0962492919000059.

[25] David Lehký, Zbyněk Keršner, and Drahomír Novák. Framepid-3pb software for material parameter identification using fracture tests and inverse analysis. *Advances in Engineering Software*, 72:147–154, 2014. ISSN 0965-9978. doi: https://doi.org/10.1016/j.advengsoft.2013.10.001. Special Issue dedicated to Professor Zdeněk Bittnar on the occasion of his Seventieth Birthday: Part 2.

[26] Marek Paruch, Alicja Piasecka-Belkhayat, and Anna Korczak. Identification of the ultra-short laser parameters during irradiation of thin metal films using the interval lattice boltzmann method and evolutionary algorithm. *Advances in Engineering Software*, 180:103456, 2023. ISSN 0965-9978. doi: https://doi.org/10.1016/j.advengsoft.2023.103456.

[27] Z. Wu, H. Ye, H. Zhang, and Y. Zheng. Seq-svf: An unsupervised data-driven method for automatically identifying hidden governing equations. *Computer Physics Communications*, 292, 2023. doi: 10.1016/j.cpc.2023.108887.

[28] Z. Xu, Y. Xia, and Q. Liao. A domain-decomposed vae method for bayesian inverse problems. *International Journal for Uncertainty Quantification*, 14(3):67–95, 2024. doi: 10.1615/Int.J.UncertaintyQuantification.2023047236.

[29] S. Li, C. Zhang, Z. Zhang, and H. Zhao. A data-driven and model-based accelerated hamiltonian monte carlo method for bayesian elliptic inverse problems. *Statistics and Computing*, 33(4), 2023. doi: 10.1007/s11222-023-10262-y.

[30] Z. Wu, C. Zhang, and Z. Zhang. Reduced-order model-based variational inference with normalizing flows for bayesian elliptic inverse problems. *Journal of Computational and Applied Mathematics*, 441, 2024. doi: 10.1016/j.cam.2023.115659.

[31] Mark Girolami, Eky Febrianto, Ge Yin, and Fehmi Cirak. The statistical finite element method (statfem) for coherent synthesis of observation data and model predictions. *Computer Methods in Applied Mechanics and Engineering*, 375:113533, 2021. ISSN 0045-7825. doi: https://doi.org/10.1016/j.cma.2020.113533.

[32] C. Duffin, E. Cripps, T. Stemler, and M. Girolami. Low-rank statistical finite elements for scalable model-data synthesis. *Journal of Computational Physics*, 463, 2022. doi: 10.1016/j.jcp.2022.111261.

[33] M. Habibi, R. M. D'Souza, S. T. M. Dawson, and A. Arzani. Integrating multi-fidelity blood flow data with reduced-order data assimilation. *Computers in Biology and Medicine*, 135, 2021. doi: 10.1016/j.compbiomed.2021.104566.

[34] Ardeshir Bangian-Tabrizi and Yogesh Jaluria. An optimization strategy for the inverse solution of a convection heat transfer problem. *International Journal of Heat and Mass Transfer*, 124:1147 – 1155, 2018. doi: 10.1016/j.ijheatmasstransfer.2018.04.053.

[35] Hamid Reza Tamaddon-Jahromi, Neeraj Kavan Chakshu, Igor Sazonov, Llion M. Evans, Hywel Thomas, and Perumal Nithiarasu. Data-driven inverse modelling through neural network (deep learning) and computational heat transfer. *Computer Methods in Applied Mechanics and Engineering*, 369:113217, 2020. ISSN 0045-7825. doi: https://doi.org/10.1016/j.cma.2020.113217.

[36] Petr Karnakov, Sergey Litvinov, and Petros Koumoutsakos. Flow reconstruction by multiresolution optimization of a discrete loss with automatic differentiation. *The European Physical Journal E*, 46(7):59, 2023. ISSN 1292-895X. doi: 10.1140/epje/s10189-023-00313-7.

[37] Petr Karnakov, Sergey Litvinov, and Petros Koumoutsakos. Solving inverse problems in physics by optimizing a discrete loss: Fast and accurate learning without neural networks. *PNAS Nexus*, 3(1):pgae005, 01 2024. ISSN 2752-6542. doi: 10.1093/pnasnexus/pgae005.

[38] Soledad Le Clainche, Esteban Ferrer, Sam Gibson, Elisabeth Cross, Alessandro Parente, and Ricardo Vinuesa. Improving aircraft performance using machine learning: A review. *Aerospace Science and Technology*, 138:108354, 2023. ISSN 1270-9638. doi: https://doi.org/10.1016/j.ast.2023.108354.

[39] Slawomir Szrama and Tomasz Lodygowski. Aircraft engine remaining useful life prediction using neural networks and real-life engine operational data. *Advances in Engineering Software*, 192:103645, 2024. ISSN 0965-9978. doi: https://doi.org/10.1016/j.advengsoft.2024.103645.

[40] Hasan Tercan and Tobias Meisen. Machine learning and deep learning based predictive quality in manufacturing: a systematic review. *Journal of Intelligent Manufacturing*, 33(7):1879–1905, 2022. ISSN 1572-8145. doi: 10.1007/s10845-022-01963-8.

[41] Hung-Wei Chiu and Ching-Hung Lee. Prediction of machining accuracy and surface quality for cnc machine tools using data driven approach. *Advances in Engineering Software*, 114:246–257, 2017. ISSN 0965-9978. doi: https://doi.org/10.1016/j.advengsoft.2017.07.008.

[42] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. The MIT Press, 2016. ISBN 9780262035613.

[43] C.E. Rasmussen and C.K.I. Williams. *Gaussian Processes for Machine Learning*. Adaptive Computation and Machine Learning series. MIT Press, 2005. ISBN 9780262182539.

[44] Bernhard Mehlig. *Machine Learning with Neural Networks: An Introduction for Scientists and Engineers*. Cambridge University Press, 2021. doi: 10.1017/9781108860604.

[45] G. Yagawa and H. Okuda. Neural networks in computational mechanics. *Archives of Computational Methods in Engineering*, 3(4):435 – 512, 1996. doi: 10.1007/BF02818935.

[46] Wiera Bielajewa, Michelle Tindall, and P. Nithiarasu. Comparative study of transformer- and lstm-based machine learning methods for transient thermal field reconstruction. *Computational Thermal Sciences: An International Journal*, 16, 2024. doi: 10.1615/ComputThermalScien.2023049936.

[47] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[48] Yunyang Zhang, Zhiqiang Gong, Weien Zhou, Xiaoyu Zhao, Xiaohu Zheng, and Wen Yao. Multi-fidelity surrogate modeling for temperature field prediction using deep convolution neural network. *Engineering Applications of Artificial Intelligence*, 123:106354, 2023. ISSN 0952-1976. doi: https://doi.org/10.1016/j.engappai.2023.106354.

[49] Feiding Zhu, Jincheng Chen, Dengfeng Ren, and Yuge Han. Transient temperature fields of the tank vehicle with various parameters using deep learning method. *Applied Thermal Engineering*, 230:120697, 2023. ISSN 1359-4311. doi: https://doi.org/10.1016/j.applthermaleng.2023.120697.

[50] N. M. Pawar, R. Soltanmohammadi, S. Faroughi, and S. A. Faroughi. Geo-guided deep learning for spatial downscaling of solute transport in heterogeneous porous media. *Computers and Geosciences*, 188, 2024. doi: 10.1016/j.cageo.2024.105599.

[51] Y. Kim, Y. Choi, and B. Yoo. Gappy ae: A nonlinear approach for gappy data reconstruction using auto-encoder. *Computer Methods in Applied Mechanics and Engineering*, 426, 2024. doi: 10.1016/j.cma.2024.116978.

[52] M. Raissi, P. Perdikaris, and G. E. Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378: 686–707, 2019. ISSN 0021-9991. doi: https://doi.org/10.1016/j.jcp.2018.10.045.

[53] Prakhar Sharma, Llion Evans, Michelle Tindall, and Perumal Nithiarasu. Stiff-pdes and physics-informed neural networks. *Archives of Computational Methods in Engineering*, 30(5):2929–2958, 6 2023. ISSN 1886-1784. doi: 10.1007/s11831-023-09890-4.

[54] Han Gao, Matthew J. Zahr, and Jian-Xun Wang. Physics-informed graph neural galerkin networks: A unified framework for solving pde-governed forward and inverse problems. *Computer Methods in Applied Mechanics and Engineering*, 390: 114502, 2022. ISSN 0045-7825. doi: https://doi.org/10.1016/j.cma.2021.114502.

[55] Lu Wang, Guangyan Liu, Guanglun Wang, and Kai Zhang. M-pinn: A mesh-based physics-informed neural network for linear elastic problems in solid mechanics. *International Journal for Numerical Methods in Engineering*, 125(9):e7444, 2024. doi: https://doi.org/10.1002/nme.7444.

[56] G. Shahzadi and A. Soulaimani. Deep neural network-based inverse analysis with application to a rockfill dam. *KSCE Journal of Civil Engineering*, 28(1):155–168, 2024. doi: 10.1007/s12205-023-0355-y.

[57] C. Xu, B. T. Cao, Y. Yuan, and G. Meschke. A multi-fidelity deep operator network (deeponet) for fusing simulation and monitoring data: Application to real-time settlement prediction during tunnel construction. *Engineering Applications of Artificial Intelligence*, 133, 2024. doi: 10.1016/j.engappai.2024.108156.

[58] S. De, M. Reynolds, M. Hassanaly, R. N. King, and A. Doostan. Bi-fidelity modeling of uncertain and partially unknown systems using deeponets. *Computational Mechanics*, 71(6):1251–1267, 2023. doi: 10.1007/s00466-023-02272-4.

[59] Y. Li, P. Ni, L. Sun, and Y. Xia. Finite element model-informed deep learning for equivalent force estimation and full-field response calculation. *Mechanical Systems and Signal Processing*, 206, 2024. doi: 10.1016/j.ymssp.2023.110892.

[60] E. Askari, D. Gorgoretti, and G. Crevecoeur. Hybrid modeling of multibody vehicles with partially known physics: discovering complex behaviors of tires. *Multibody System Dynamics*, 2024. doi: 10.1007/s11044-024-09983-3.

[61] R. E. Meethal, A. Kodakkal, M. Khalil, A. Ghantasala, B. Obst, K. U. Bletzinger, and R. Wüchner. Finite element method-enhanced neural network for forward and inverse problems. *Advanced Modeling and Simulation in Engineering Sciences*, 10 (1), 2023. doi: 10.1186/s40323-023-00243-1.

[62] Neeraj Kavan Chakshu, Igor Sazonov, and Perumal Nithiarasu. Towards enabling a cardiovascular digital twin for human systemic circulation using inverse analysis. *Biomechanics and Modeling in Mechanobiology*, 20(2):449–465, 4 2021. ISSN 1617-7940. doi: 10.1007/s10237-020-01393-6.

[63] Neeraj Kavan Chakshu, Jason Carson, Igor Sazonov, and Perumal Nithiarasu. A semi-active human digital twin model for detecting severity of carotid stenoses from head vibration—a coupled computational mechanics and computer vision method. *International Journal for Numerical Methods in Biomedical Engineering*, 35(5), 2019. doi: 10.1002/cnm.3180.

[64] A. Abadías Llamas, N. J. Bartie, M. Heibeck, M. Stelter, and M. A. Reuter. Simulation-based exergy analysis of large circular economy systems: Zinc production coupled to cdte photovoltaic module life cycle. *Journal of Sustainable Metallurgy*, 6(1):34–67, 3 2020. ISSN 2199-3831. doi: 10.1007/s40831-019-00255-5.

[65] Claudio Mandolla, Antonio Messeni Petruzzelli, Gianluca Percoco, and Andrea Urbinati. Building a digital twin for additive manufacturing through the exploitation of blockchain: A case analysis of the aircraft industry. *Computers in Industry*, 109:134–152, 2019. ISSN 0166-3615. doi: https://doi.org/10.1016/j.compind.2019.04.011.

[66] Harry Millwater, Juan Ocampo, and Nathan Crosby. Probabilistic methods for risk assessment of airframe digital twin structures. *Engineering Fracture Mechanics*,

221:106674, 2019. ISSN 0013-7944. doi: https://doi.org/10.1016/j.engfracmech.2019.106674.

[67] Chenzhao Li, Sankaran Mahadevan, You Ling, Sergio Choze, and Liping Wang. Dynamic bayesian network for aircraft wing health monitoring digital twin. *AIAA Journal*, 55(3):930–941, 2017. doi: 10.2514/1.J055201.

[68] Armando Di Meglio, Nicola Massarotti, and Perumal Nithiarasu. A physics-driven and machine learning-based digital twinning approach to transient thermal systems. *International Journal of Numerical Methods for Heat & Fluid Flow*, 02 2024. doi: https://doi.org/10.1108/HFF-10-2023-0616.

[69] Lei Chen, Kunfeng Zhao, and Wen-Quan Tao. Research on one-dimensional digital twin algorithm of plate heat exchanger. *Numerical Heat Transfer, Part A: Applications*, 85(15):2419–2438, 2024. doi: 10.1080/10407782.2023.2222906.

[70] Enrico Spateri, Fredy Ruiz, and Giambattista Gruosso. An electrothermal digital twin for design and management of radiation heating in industrial processes. *IEEE Transactions on Industry Applications*, 59(5):5784–5791, Sep. 2023. ISSN 1939-9367. doi: 10.1109/TIA.2023.3287817.

[71] Zhipeng Cui, Jing Xu, Wenhao Liu, Guanjia Zhao, and Suxia Ma. Data-driven modeling-based digital twin of supercritical coal-fired boiler for metal temperature anomaly detection. *Energy*, 278:127959, 2023. ISSN 0360-5442. doi: https://doi.org/10.1016/j.energy.2023.127959.

[72] M. Renault, J. Viquerat, P. Meliga, G.-A. Grandin, N. Meynet, and E. Hachem. Investigating gas furnace control practices with reinforcement learning. *International Journal of Heat and Mass Transfer*, 209:124147, 2023. ISSN 0017-9310. doi: https://doi.org/10.1016/j.ijheatmasstransfer.2023.124147.

[73] E. Hachem, H. Ghraieb, J. Viquerat, A. Larcher, and P. Meliga. Deep reinforcement learning for the control of conjugate heat transfer. *Journal of Computational Physics*, 436:110317, 2021. ISSN 0021-9991. doi: https://doi.org/10.1016/j.jcp.2021.110317.

[74] Armando Di Meglio, Nicola Massarotti, and Perumal Nithiarasu. Digital twins of thermal systems: a comparison between supervised and reinforcement learning.

Computational Thermal Sciences: An International Journal, 17(3):39–46, 2025. doi: 10.1615/ComputThermalScien.2025057435.

[75] Perumal Nithiarasu, Roland W. Lewis, and Kankanhalli N. Seetharamu. *Fundamentals of the Finite Element Method for Heat and Mass Transfer*. Wiley, 2 edition, 2016. ISBN 9780470756256.

[76] S. Zinn, S.L. Semiatin, E.P.R. Institute, and B.M.I.C. Laboratories. *Elements of Induction Heating: Design, Control, and Applications*. ASM International, 1988. ISBN 9781615031986.

[77] Åke Björck. *Numerical Methods for Least Squares Problems*. Society for Industrial and Applied Mathematics, 1996. doi: 10.1137/1.9781611971484.

[78] S. Gratton, A. S. Lawless, and N. K. Nichols. Approximate gauss–newton methods for nonlinear least squares problems. *SIAM Journal on Optimization*, 18(1):106–132, 2007. doi: 10.1137/050624935.

[79] B. Besselink, U. Tabak, A. Lutowska, N. van de Wouw, H. Nijmeijer, D.J. Rixen, M.E. Hochstenbach, and W.H.A. Schilders. A comparison of model reduction techniques from structural dynamics, numerical mathematics and systems and control. *Journal of Sound and Vibration*, 332(19):4403–4422, 2013. ISSN 0022-460X. doi: https://doi.org/10.1016/j.jsv.2013.03.025.

[80] Nicolas Guérin, Anders Thorin, Fabrice Thouverez, Mathias Legrand, and Patricio Almeida. Thermomechanical model reduction for efficient simulations of rotor-stator contact interaction. *Journal of Engineering for Gas Turbines and Power*, 141(2):022501, 09 2018. ISSN 0742-4795. doi: 10.1115/1.4040858.

[81] Dale E. Seborg, Francis J. Doyle III, Duncan A. Mellichamp, and Thomas F. Edgar. *Process Dynamics and Control*. John Wiley & Sons, Inc., 4th edition, 2016. ISBN 978-1-119-28591-5.

[82] Gene F. Franklin, J. David Powell, and Abbas Emami-Naeini. *Feedback Control of Dynamic Systems*. Addison-Wesley Publishing Company, 3rd edition, 1994. ISBN 0-201-52747-2.

[83] R. Roberto Carmona, Hong Gun Sung, Young Shik Kim, and H. Alberto Vazquez. Stable pid control for mobile robots. In *2018 15th International Conference on*

Control, Automation, Robotics and Vision (ICARCV), pages 1891–1896, 2018. doi: 10.1109/ICARCV.2018.8581132.

[84] Oscar Camacho, Sebastian Vega, Marco Herrera, Antonio Di Teodoro, and Juan J. Gude. A fractional order pid-based sliding mode controller approach for chemical processes. *Results in Control and Optimization*, 20:100592, 2025. ISSN 2666-7207. doi: https://doi.org/10.1016/j.rico.2025.100592.

[85] Sunwoo Lee, Chaoyang He, and Salman Avestimehr. Achieving small-batch accuracy with large-batch scalability via hessian-aware learning rate adjustment. *Neural Networks*, 158:1–14, 2023. ISSN 0893-6080. doi: https://doi.org/10.1016/j.neunet.2022.11.007.

[86] Joachim Schoberl. C++11 implementation of finite elements in ngsolve. Technical report, Institute for Analysis and Scientific Computing, Vienna University of Technology, 2014.

[87] Peter Gangl, Kevin Sturm, Michael Neunteufel, and Joachim Schöberl. Fully and semi-automated shape differentiation in ngsolve. *Structural & Multidisciplinary Optimization*, 63(3):1579–1607, 2021. ISSN 1615-1488. doi: https://doi.org/10.1007/s00158-020-02742-w.

[88] Sabine Zaglmayr. *High Order Finite Elements for Electromagnetic Field Computation*. Phd thesis, Johannes Kepler University Linz, 2006.

[89] Philipp W. Schroeder, Volker John, Philip L. Lederer, Christoph Lehrenfeld, Gert Lube, and Joachim Schöberl. On reference solutions and the sensitivity of the 2d kelvin–helmholtz instability problem. *Computers and Mathematics with Applications*, 77(4):1010–1028, 2019. ISSN 0898-1221. doi: https://doi.org/10.1016/j.camwa.2018.10.030.

[90] Robert Cimrman, Vladimír Lukeš, and Eduard Rohan. Multiscale finite element calculations in python using sfepy. *Advances in Computational Mathematics*, 2019. ISSN 1572-9044. doi: 10.1007/s10444-019-09666-0.

[91] Theron D. Marshall, Dennis L. Youchison, and Lee C. Cadwallader. Modeling the nukiyama curve for water-cooled fusion divertor channels. *Fusion Technol.*, 35: 8–16, 2001. URL http://film2000.free.fr/TOFE.pdf.

[92] W. Wagner, J. R. Cooper, A. Dittmann, J. Kijima, H.-J. Kretzschmar, A. Kruse, R. Mares˘, K. Oguchi, H. Sato, I. Sto¨cker, O. S˘ifner, Y. Takaishi, I. Tanishita, J. Tru¨benbach, and Th. Willkommen. The iapws industrial formulation 1997 for the thermodynamic properties of water and steam. *Journal of Engineering for Gas Turbines and Power*, 122(1):150–184, 01 2000. ISSN 0742-4795. doi: 10.1115/1.483186.

[93] International Association for the Properties of Water and Steam, 2025. URL https://www.iapws.org/. Accessed: 19/05/2025.

[94] E. N. Sieder and G. E. Tate. Heat transfer and pressure drop of liquids in tubes. *Industrial & Engineering Chemistry*, 28(12):1429–1435, 1936. doi: 10.1021/ie50324a027.

[95] Florian Seibold, Phillip Ligrani, and Bernhard Weigand. Flow and heat transfer in swirl tubes — a review. *International Journal of Heat and Mass Transfer*, 187: 122455, 2022. ISSN 0017-9310. doi: https://doi.org/10.1016/j.ijheatmasstransfer.2021.122455.

[96] A. E. Bergles and W. M. Rohsenow. The determination of forced-convection surface-boiling heat transfer. *Journal of Heat Transfer*, 86(3):365–372, 08 1964. ISSN 0022-1481. doi: 10.1115/1.3688697.

[97] Masanori Araki, Masuro Ogawa, Tomoaki Kunugi, Kazuyoshi Satoh, and Satoshi Suzuki. Experiments on heat transfer of smooth and swirl tubes under one-sided heating conditions. *International Journal of Heat and Mass Transfer*, 39(14):3045–3055, 1996. ISSN 0017-9310. doi: https://doi.org/10.1016/0017-9310(95)00344-4.

[98] Myisha A. Chowdhury, Saif S.S. Al-Wahaibi, and Qiugang Lu. Entropy-maximizing td3-based reinforcement learning for adaptive pid control of dynamical systems. *Computers & Chemical Engineering*, 178:108393, 2023. ISSN 0098-1354. doi: https://doi.org/10.1016/j.compchemeng.2023.108393.

[99] Oguzhan Dogru, Kirubakaran Velswamy, Fadi Ibrahim, Yuqi Wu, Arun Senthil Sundaramoorthy, Biao Huang, Shu Xu, Mark Nixon, and Noel Bell. Reinforcement learning approach to autonomous pid tuning. *Computers & Chemical Engineering*, 161:107760, 2022. ISSN 0098-1354. doi: https://doi.org/10.1016/j.compchemeng.2022.107760.

[100] Lloyd Fletcher, Joel Hirst, Lorna Sibson, Megan Sampson, Adel Tayeb, Alex Marsh, Rory Spencer, and John Charlton. The python validation engine, 2025. URL https://computer-aided-validation-laboratory.github.io/pyvale/. Accessed: 10/09/2025.

[101] Chengwei Qin, Aston Zhang, Zhuosheng Zhang, Jiaao Chen, Michihiro Yasunaga, and Diyi Yang. Is chatgpt a general-purpose natural language processing task solver?, 2023.

[102] Qingsong Wen, Tian Zhou, Chaoli Zhang, Weiqi Chen, Ziqing Ma, Junchi Yan, and Liang Sun. Transformers in time series: A survey, 2023.

[103] Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. Informer: Beyond efficient transformer for long sequence time-series forecasting. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(12):11106–11115, May 2021. doi: 10.1609/aaai.v35i12.17325.

[104] Haixu Wu, Jiehui Xu, Jianmin Wang, and Mingsheng Long. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. In *Neural Information Processing Systems*, 2021.

[105] Tian Zhou, Ziqing Ma, Qingsong Wen, Xue Wang, Liang Sun, and Rong Jin. FEDformer: Frequency enhanced decomposed transformer for long-term series forecasting. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 27268–27286. PMLR, 17–23 Jul 2022.

[106] Bryan Lim and Stefan Zohren. Time-series forecasting with deep learning: a survey. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 379(2194):20200209, 2021. doi: 10.1098/rsta.2020.0209.

[107] Yong Yu, Xiaosheng Si, Changhua Hu, and Jianxun Zhang. A review of recurrent neural networks: Lstm cells and network architectures. *Neural Computation*, 31 (7):1235–1270, 2019. doi: 10.1162/neco_a_01199.

[108] Ardeshir Bangian-Tabrizi and Yogesh Jaluria. An optimization strategy for the inverse solution of a convection heat transfer problem. *International Journal of*

Heat and Mass Transfer, 124:1147 – 1155, 2018. doi: 10.1016/j.ijheatmasstransfer.2018.04.053.

[109] Yudong Zhang, Shuihua Wang, Genlin Ji, et al. A comprehensive survey on particle swarm optimization algorithm and its applications. *Mathematical problems in engineering*, 2015, 2015.

[110] Y. Bengio, P. Simard, and P. Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, 5(2):157–166, 1994. doi: 10.1109/72.279181.

[111] Wei-Ning Hsu, Yu Zhang, Ann Lee, and James R. Glass. Exploiting depth and highway connections in convolutional recurrent deep neural networks for speech recognition. In *Interspeech*, 2016.

[112] Florent Altché and Arnaud de La Fortelle. An lstm network for highway trajectory prediction. *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, pages 353–359, 2017.

[113] Lei Ren, Jiabao Dong, Xiaokang Wang, Zihao Meng, Li Zhao, and M. Jamal Deen. A data-driven auto-cnn-lstm prediction model for lithium-ion battery remaining useful life. *IEEE Transactions on Industrial Informatics*, 17(5):3478–3487, 2021. doi: 10.1109/TII.2020.3008223.

[114] Felix A. Gers, Jürgen Schmidhuber, and Fred Cummins. Learning to forget: Continual prediction with lstm. *Neural Computation*, 12(10):2451–2471, 2000. doi: 10.1162/089976600300015015.

[115] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online, October 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-demos.6.

[116] Qingsong Wen, Tian Zhou, Chaoli Zhang, Weiqi Chen, Ziqing Ma, Junchi Yan, and Liang Sun. Transformers in time series: A survey, 2023.

[117] Binrong Wu, Lin Wang, and Yu-Rong Zeng. Interpretable wind speed prediction with multivariate time series and temporal fusion transformers. *Energy*, 252: 123990, 2022. ISSN 0360-5442. doi: https://doi.org/10.1016/j.energy.2022.123990.

[118] Boris Ivanovic and Marco Pavone. The trajectron: Probabilistic multi-agent trajectory modeling with dynamic spatiotemporal graphs. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.

[119] Peter Bloem. Transformers from scratch, August 2019. URL https://peterbloem.nl/blog/transformers. Accessed on: March 29, 2023.

[120] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017.

[121] Ruibin Xiong, Yunchang Yang, Di He, Kai Zheng, Shuxin Zheng, Chen Xing, Huishuai Zhang, Yanyan Lan, Liwei Wang, and Tieyan Liu. On layer normalization in the transformer architecture. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 10524–10533. PMLR, 13–18 Jul 2020.

[122] David Zwicker. py-pde: A python package for solving partial differential equations. *Journal of Open Source Software*, 5:2158, 04 2020. doi: 10.21105/joss.02158.

[123] Cynthia Rudin. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence*, 1: 206–215, 05 2019. doi: 10.1038/s42256-019-0048-x.

[124] Alexander Kolesnikov, Alexey Dosovitskiy, Dirk Weissenborn, Georg Heigold, Jakob Uszkoreit, Lucas Beyer, Matthias Minderer, Mostafa Dehghani, Neil Houlsby, Sylvain Gelly, Thomas Unterthiner, and Xiaohua Zhai. An image is worth 16x16 words: Transformers for image recognition at scale. 2021.