Università degli Studi di Padova
Department of Mathematics "Tullio Levi-Civita"

Ph.D. COURSE IN: MATHEMATICS
CURRICULUM: COMPUTATIONAL MATHEMATICS
XXXVII CYCLE

Ph.D. Thesis

A mathematical construction of the digital twin of bread leavening

The doctoral scholarship was co-financed with resources from the PON REACT-EU programme.

Coordinator	Prof.	Giovanni Colombo
Supervisor	Prof.	Fabio Marcuzzi
Co-Supervisor	Prof.	Giulio Giuseppe Giusteri

Ph.D. student: LAURA RINALDI

A coloro che mi hanno resa quella che sono.

Abstract

Industry 4.0 is the term used to describe the fourth industrial revolution, which involves the integration of digital technologies and machine learning into the manufacturing sector. One of the main concepts of the fourth industrial revolution is the **digital twin** (DT), which helps build a bridge between the physical and the digital world and becomes a fundamental component to make decisions, check the status, modify the behavior and perform predictions. A DT is classically defined as the virtual replica of a real system that is continuously updated with data from its physical counterpart.

The main idea of our project, which leads to the development of this doctoral thesis is the mathematical building of a DT of bread leavening to avoid energy waste. The leavening machines are not energy-optimized, therefore it is useful to digitally reproduce the physical phenomena for energy saving. The focus of this thesis is the use, development and connection of mathematical theories, such as continuum mechanics, machine learning and source estimation via inverse problems, to formulate and solve mathematical problems with an industrial application.

In this doctoral thesis, we develop a mathematical model of bread leavening in a warm chamber by coupling heat transfer, yeast growth, and the presence of carbon dioxide with the deformation of bread dough. Specifically, the above-mentioned continuum model involves a heat equation to describe the evolution of temperature in the bread, an ordinary differential equation to mirror the life cycle of yeasts and their breeding, then a diffusive equation for the carbon dioxide production and propagation and finally, the volumetric expansion represented by considering the elastic energy related to the bread, seen as a hyperelastic material. We analyze the corresponding system of partial differential equations and we discretize the problem, by using a semi-implicit Euler method and the finite-element method (FEM) for the time and space variables, respectively. The resulting discretized scheme is then implemented by using the Python library FEniCSx. Numerical simulations allow a sensitivity analysis to identify the energy consumption necessary to achieve a target volume under different settings of the leavening chamber and different concentrations of yeast in the bread dough, thereby providing a tool for the identification of cost-effective energy usage protocols [1].

At this point, we study operator networks that emerged as promising deep learning tools for approximating the solution of PDEs. These networks map input functions that describe material properties, forcing functions and boundary data to the solution of a PDE. Therefore, we build a surrogate model, based on the numerical one, which simulates the physical behavior, is computationally cheaper and needs few data as input. We exploit the physical-mathematical model, especially its space-time weak formulation, to define an ad-hoc variationally mimetic operator network (VarMiON) by following and adapting the procedure presented in the article [2] to the PDEs that characterize our problem. In the application considered here, the presence of a surrogate model is motivated by the fact that it has to run online with the real process to monitor it and to estimate the energy consumption thus avoiding waste [3].

Finally, we present a general observation on how to replace changes of material proper-

ties in limited regions within a domain with fictitious forcing terms in initial- and boundary-value problems associated with linear wave propagation and diffusion [4]. Then, by considering a paradigmatic heat conduction problem on a domain with a cavity, we prove that the presence of the void can be replaced by a fictitious heat source with support that coincides with the cavity. We illustrate this fact in a situation where the source term can be analytically recovered from the values of the temperature and heat flux at the boundary of the cavity. Our result provides a strategy to map the nonlinear geometric inverse problem of void identification into a more manageable one that involves the estimate of forcing terms given the knowledge of external boundary data. To set the stage for a systematic study of the inverse problem, we present algebraic reconstructions, based on FEM, that give an approximation of the fictitious source from different sets of temperature measurements. We show how the accuracy of such reconstruction is reflected on the void identification.

In future work, we want to use the latter strategy to rebuild the porous texture of bread during leavening by solving the resulting inverse problem of source recognition through thermography data.

Sommario

Industria 4.0 è il termine utilizzato per descrivere la quarta rivoluzione industriale, la quale prevede l'integrazione delle tecnologie digitali e dell'apprendimento automatico nel settore manifatturiero. Uno dei concetti principali della quarta rivoluzione industriale è quello di **gemello digitale** ("Digital Twin" - DT) che permette di costruire un nesso tra il mondo fisico e digitale, diventando una componente fondamentale per prendere decisioni, controllare lo stato di un sistema, modificarne o prevederne il suo comportamento. Un DT è classicamente definito come la replica virtuale di un sistema reale in continuo aggiornamento grazie ai dati provenienti dalla sua controparte fisica.

L'idea principale del nostro progetto, nonché base di questa tesi di dottorato, consiste nella costruzione matematica di un DT per la lievitazione del pane al fine di evitare sprechi energertici, essendo le macchine di lievitazione non ottimizzate da tale punto di vista. L'obiettivo di questa tesi è quello di utilizzare, sviluppare e connettere teorie matematiche, come la meccanica dei continui, tecniche di machine learning e la stima di sorgenti tramite problemi inversi per tradurre problemi matematici in applicazioni industriali.

In questa tesi di dottorato abbiamo sviluppato un modello fisico-matematico per la lievitazione del pane in cui lo scambio di calore, la crescita dei lieviti e la presenza di anidride carbonica sono accoppiati con la deformazione dell'impasto. Nello specifico, tale modello continuo prevede un'equazione del calore, per descrivere l'evoluzione della temperatura nel pane, un'equazione differenziale ordinaria per il ciclo di vita dei lieviti e la loro riproduzione, quindi un'equazione diffusiva per la produzione e la propagazione dell'anidride carbonica. In questo modello l'espansione volumetrica è risolta minimizzando l'energia elastica dell'impasto, trattato al pari di un materiale iperelastico. La discretizzazione del sistema di equazioni differenziali rislutante, è stata realizzata con il metodo di Eulero semi-implicito per la variabile temporale ed il metodo degli elementi finiti (FEM) per le variabili spaziali. Lo schema discretizzato è stato quindi implementato numericamente utilizzando la libreria Python FEniCSx. Le simulazioni ottenute hanno consentito di eseguire un'analisi di sensibilità, identificando il consumo energetico necessario per raggiungere un volume target in diverse impostazioni della camera di lievitazione e diverse concentrazioni di lievito nell'impasto, definendo così uno strumento utile per l'dentificazione di protocolli di utilizzo al fine di ottenere un risparmio energetico [1].

A questo punto, abbiamo approfondito lo studio sulle reti neurali, strumenti promettenti di deep machine learning per approssimare le soluzioni di equazioni differenziali alle derivate parziali. Queste reti mappano funzioni di input che descrivono proprietà dei materiali, forze e dati al bordo, alla soluzione di una PDE. Pertanto, abbiamo costruito un modello surrogato, basato su quello numerico, che ne simula il comportamento fisico, risulta computazionalmente più economico e necessita di pochi dati come input. Per far questo, abbiamo quindi sfruttato il modello fisico-matematico, in particolare la sua formulazione debole spazio-tempo, per definire una rete di operatori neurali ad hoc ("Variational Mimetic Operator Network" - VarMiON) seguendo e adattando la procedura, presentata nell'articolo [2], alle PDEs che caratterizzano il sistema di lievitazione. Nella nostra applicazione industriale, la presenza di un modello surrogato è motivata dala necessità di

eseguirlo simultaneamente con il processo reale per poterlo monitorare e stimarne il consumo energetico evitando così sprechi [3].

Infine, abbiamo derivato una teoria generale, utile per il monitoraggio della lievitazione, che consiste nel sostituire variazioni delle proprietà fisiche dei materiali, in regioni limitate all'interno di un dominio, con termini di forzanti fittizie nei problemi associati alla propagazione e diffusione date le condizioni al bordo ed iniziali [4]. Quindi, abbiamo considerato come problema paradigmatico la conduzione di calore su un dominio con una cavità e dimostrato che la presenza di un vuoto può essere sostituita da una sorgente fittizia, il cui supporto, coincide con la cavità stessa. Abbiamo applicato questa strategia in una situazione in cui il termine sorgente potesse essere ricostruito analiticamente, dai valori della temperatura e del flusso di calore sul bordo della cavità. Questo risultato ha fornito quindi una strategia per mappare il problema inverso geometrico, non lineare dell'identificazione del vuoto, in uno più gestibile, che coinvolge la stima dei termini forzanti, data la conoscenza dei dati al bordo. Per preparare il terreno ad uno studio sistematico del problema inverso, abbiamo ricostruito la forzante algebricamente, basandoci sulla discretizzazione FEM, che ha fornito un'approssimazione della sorgente fittizia rispetto a diversi set di misurazioni della temperatura. Abbiamo poi mostrato, come l'accuratezza di tale ricostruzione si rifletta sull'identificazione della cavità.

Utilizzeremo quest'ultima strategia, in un prossimo lavoro, per ricostruire la consistenza porosa del pane durante la lievitazione risolvendo il problema inverso di riconoscimento di forzanti tramite termografie.

Contents

Introduction

Industry 4.0 is the term used to describe the fourth industrial revolution, which involves the integration of digital technologies such as IoT, cloud computing, analytics, AI, and machine learning into the manufacturing sector. The main target of Industry 4.0 is to increase productivity and efficiency across the value chain and enable the customization and optimization of products and services. One of the main concepts of the fourth industrial revolution is the **digital twin** (DT), which helps to build a bridge between the physical and the digital world and becomes a fundamental component to make decisions, check the status, modify the behavior and perform predictions.

A DT is classically defined as the virtual replica of a real-world product, system, being, communities, even cities, that is continuously updated with data from its physical counterpart [5]. The first known definition, that follows, was presented by NASA for reflecting the life of an air vehicle. The actual term "digital twin" was coined by Grieves [6] and Tuegel [7] in 2011 and further developed in Grieves' consecutive works [8].

Definition 0.0.1. *The **digital twin** is an integrated multi-physics, multi-scale, probabilistic simulation of a complex product and uses the best available physical models, sensor updates, etc., to mirror the life of its corresponding twin.*

Since DT is an inter-disciplinary subject there is no unique definition, because it depends on the scientific area and on the purpose. In fact, a digital twin may integrate all data models and other information of the physical object generated along its life cycle for a dedicated goal, leading to the following definition.

Definition 0.0.2. *The **digital twin** is a digital representation of a physical item or assembly using integrated simulations and service data, holding information from multiple sources across the product life cycle.*

Nevertheless, all the definitions are representative and useful for developing a general idea. DT can be applied in different situations and the aim is to describe all the product life cycle phases, from its design and prototyping to its disposal. The previous definitions highlight that the main objective reflects the life cycle of any element, product, or system that works as its physical twin. Grieves and Vickers [8] described the difference between the following concepts:

- Digital Twin Prototype: describes the prototypical physical artifact. It contains the informational sets necessary to describe and produce a physical version that duplicates or twins the virtual version.
- Digital Twin Instance: describes a specific corresponding physical product that an individual digital twin remains linked to throughout the life of that physical product.

- Digital Twin Environment: a multi-domain physics application space for operating on digital twins.

Moreover, a digital twin is made-up of two existing systems:

1. the tangible system of real physics;
2. its virtual replica is enabled by real data and underlying models through digital technologies.

The physical object in the physical world and the digital object in the digital world have to be connected between each other. The connection is given by data from the real system to the virtual one and information that you can retrieve, with indirect measurements, from the digital replica. The physical world is composed of two main elements:

- devices: the physical twins from which DTs are intended to be created;
- sensors: elements physically connected to devices from which one could get data and information.

The digital world is instead composed of two main elements:

- the virtual environmental platform to construct a 3-dimensional digital model;
- DT which mirrors the reality and allows multiple operations.

Finally the connections between the two depend on the choice of each author and on the scope to which the DT is created for.

Many times in real-world applications we often deal with limited knowledge of a system, due to the practical inaccessibility of the physical domain, of some variables or parameters. Such data inaccessibility could be due to costly or impractical physical measurement instruments, or because the measurement procedure is destructive with respect to the real system, therefore the need of having virtual alternatives. Thus the presence of DTs is motivated by the necessity of obtaining some information about the real system by querying the virtual one in a non-intrusive manner.

A mathematical model gives us a whole and detailed perspective of the system to monitor reality through soft sensors and indirect measures by estimating the quantity of our interest. Such technology helps us to control the real system, to carry out maintenance tasks or to optimize some processes but also, to avoid some failures or even to perform predictions. The objective is to have a digital representation suited to the purpose in terms of level of detail, completeness, accuracy and execution speed [9].

It is possible to divide the application of DTs into three main types [10]:

1. plain gadget models which include two sets of data: data-information obtained by sensors and the set of expectation values that want to be obtained by the gadget;
2. embedded DTs, where the interaction between the real and digital world happens in a bidirectional way;
3. networked twins, where different embedded DTs are connected between each other and communicate.

In this thesis, we will work with **embedded digital twin** [9] i.e. the virtual representation of physical systems that runs in embedded systems which have sensors and actuators to interact with the real world.

In general, sensors are technical devices that monitor their environment and continuously produce signals at a regular frequency. A physical sensor is a sensor that reacts to a physical stimulus and transmits a resulting impulse – typically through electrical signals that can be captured and stored in digital form. In contrast to physical sensors, a so-called virtual sensor is a pure software sensor which autonomously produces signals by combining and aggregating signals that it receives (synchronously or asynchronously) from physical or other virtual sensors [11]. Virtual sensors are useful to enrich available information about physical variables and parameters that cannot be provided by direct physical measurements.

Virtual sensing proves to be highly relevant when on-site measurement of the variable of interest is not feasible due to non-accessible locations, cost or the fact that introducing sensors would distort the system under test.

Definition 0.0.3 (Soft-sensor)**.** *Soft sensors are algorithms that process the available data to estimate these lacking measurements.*

In the perspective of using these algorithms within the DTs one could think to take advantage of the mathematical description of the model that is the foundation of a DT.

Definition 0.0.4 (Physics-aware soft-sensor [12])**.** *Physics-aware soft-sensors are algorithms that perform an indirect measurement by exploiting a* ***physical-mathematical model*** *plus a possible* ***data-driven extension****, used within an estimation algorithm.*

Following the analysis in [10] nowadays DTs are involved in many fields, in particular, the aeronautics and space area (69%) used to monitor the operation of a system and the reliability of the created model, in the manufacturing area (19%) used to monitor the life cycle of a system and to optimize its design, and in the informatics area (4%) for IoT life cycle management, thus to create control architectures based on this technology. Today, using and improving DTs could have a great impact since this technology helps us to save resources, money and processing time.

Even if there is not a generic way to build an embedded digital twin, we select and identify these main tasks to perform:

1 construct a **numerical model** which simulates the physical process and governed by physical laws;

2 define **inverse problems**, based on the previous numerical model, to set parameters;

3 define an **algorithm to capture the information** which you are interested in.

It is worth remarking that the simulations, obtained by the numerical model, could be computationally expensive or take a lot of time to give feedback to the user. To overcome such a possible issue it is necessary a surrogate model of reduced order which is based on the numerical one and that can bring the simulations to run online with the real process in an embedded system. Producing surrogate models reduces the complexity and improves the execution performance.

In addition, the inverse problem allows us to continuously calibrate the parameters which ensures that the digital twin is always updated with respect to its real counterpart.

Contribution

This thesis focuses on the use, development and connection of mathematical theories, such as continuum mechanics, machine learning, and source estimation via inverse problems, to formulate and solve mathematical problems with an industrial application.

Nowadays, bread production is responsible for significant energy usage and CO_2 emission. In the UK e.g., more than three-quarters of bread is manufactured on an industrial scale in large commercial bakeries and bread manufacturing consumes 2 TWh of energy and produces $570,000$ tonnes of CO_2 emissions per year [13]. To optimize the process and the energy consumption a lot of effort focuses on the cooking process in the oven [13–15]. Nevertheless, leavening is a much longer process and energy is spent both to accelerate leavening with heat and to slow it down by cooling. Issues of formulation, natural leavening time, production rhythm, and final product quality influence with competing interests in the use of energy in the bread making process. Many models focus their attention on the baking process, by developing partial differential equations to describe heat and mass transfer of different components and the overall energy balance [14, 16–20]. Some works do not consider the volumetric expansion but only address the effects of cooking on the dough [13, 16, 17]. For these reasons, we focus on leavening by coupling heat transfer phenomena with elastic response.

Firstly, we want to provide a continuum mathematical model for bread leavening backbone of DTs for leavening chambers to achieve a more cost-effective and sustainable bread

making process as presented in [1]. The *material domain* $\Omega \subset \mathbb{R}^3$ provides a fixed set of labels for the material points of the dough. The actual space occupied by the material is given by $\boldsymbol{u}(\Omega)$, where $\boldsymbol{u} : \Omega \to \mathbb{R}^3$ is the deformation map. We partition the boundary of Ω as the disjoint union of three parts to identify the points in contact with the bottom side and the lateral sides of the baking tray and the top points in contact with air. The different quantities involved in the process description are represented by fields on the fixed material domain Ω. To determine the actual shape of the dough under given conditions, we model it as a hyperelastic material and we postulate the elastic energy functional $\mathcal{E}$ to reach a target volume configuration. We then follow the deformation of the material with a quasi-static approach, by solving the Euler–Lagrange equation $\delta\mathcal{E}(\boldsymbol{u}) = 0$, subject to prescribed boundary conditions. The equation for the deformation map is coupled with evolution equations for other quantities via the dependence of the dilation field of the target volume configuration and of the elastic stiffness on the concentration of carbon dioxide. The second equation considered in the model is the heat equation, that describes the temperature evolution of the dough during leavening. For this equation, we prescribe initial conditions and add boundary conditions (of Robin type) describing the energy exchanged between the bread and the air in the leavening chamber. Then we take into account the concentration of yeast in the dough with respect to the flour quantity, which we suppose to be uniform at the initial time and that evolves according to a Cauchy problem depending on the growth rate function modeled as a cubic polynomial. Moreover, yeast metabolism involves the production of carbon dioxide, the concentration of which has a diffusive evolution with a source term depending on the rate of CO_2 production, on the temperature reached by the bread, and on the yeast concentration. Indeed, the carbon dioxide production due to fermentation implies a positive volume expansion thus, we introduce an estimate of the added CO_2 volume and compute the volume of the CO_2 gas. With this value, we can then update the dilation coefficient of the target volume configuration and of the other elastic stiffness to reflect the weakening of the elastic response due to the presence of CO_2.

At this point, we study the uniqueness of solutions for the evolution equations concerning the temperature, CO_2 and yeast concentrations, under the assumption that a static deformation is given; to this end, we work with the weak forms of our PDEs.

To deal with a discretized model, we first present the time discretization of the problem, which is at the basis of our numerical scheme. We use a semi-implicit Euler method to deal with the coupling among the different equations and we prove that there exists a unique weak solution associated with the n-th time step of the problem discretized in time. Moreover, we build a sequence of piece-wise constant functions on each time interval solutions of the single-time-step and prove the convergence of this procedure.

For the spatial discretization, we use the finite-element method (FEM) on an unstructured mesh, made of tetrahedra. We define a vector function space in which we approximate the vectorial placement with Lagrangian elements P1, and a function space, where the other functions are approximated with P1 elements as well. Notice that, working with material coordinates allows us to have a fixed domain independent of the volume defor-

mation. The simulations are performed in Python using the library FEniCSx [21, 22] and the results are analyzed with the software ParaView. Concerning the final goal of this DT: avoiding energy waste, we consider that the energy used by the leavening is the sum of the energy absorbed by the bread dough, and the power dissipated by the leavening chamber simply to reach and keep the temperature of the leavening chamber. At this stage, we use the previous model to perform some experiments, to compare the behavior of bread leavening in the different conditions of the temperature of the leavening chamber and of the yeast initial concentration, thereby providing a tool for the identification of cost-effective protocols to the end of avoiding energy waste [1].

At this point, we have the necessity of a surrogate model, because a DT has to run simultaneously with the real process, in embedded systems [3], thus we reduce the order of our numerical model by exploiting the Variational Mimetic Operator (VarMiON).

A neural network $\mathcal{F}(\boldsymbol{x}, \boldsymbol{\theta}, \boldsymbol{\Theta})$ represents a family of parameterized functions, where $\boldsymbol{\theta}$ needs to suitably chosen such that the network approximates the target function $f(\boldsymbol{x})$ at the input $\boldsymbol{x}$, while $\boldsymbol{\Theta}$ corresponds to the hyper-parameters of the network such as the depth, width, type of activation function [23]. The strategy to design a robust network involves the training phase to find the optimal values of $\boldsymbol{\theta}$ (for a fixed $\boldsymbol{\Theta}$) by minimizing the loss function, the validation phase to find the optimal values of $\boldsymbol{\Theta}$ and the testing phase to estimate the performance on data not used during the first two phases. An evolution of neural networks are operator networks useful to approximate operators $\mathcal{N}$ that map functions to functions, which a popular version is the Deep Operator Network (DeepONet), that comprises two neural networks (branch and trunk). Especially, it takes a dot product between the outputs of the branch and trunk nets, to get the final prediction $\tilde{\mathcal{N}}(\cdot, \cdot, \boldsymbol{\theta})$, where the trainable parameters of the DeepONet are the combined parameters of the branch and trunk nets, i.e. $\boldsymbol{\theta} = [\boldsymbol{\theta}_B, \boldsymbol{\theta}_T]$. In paper [2] the authors describe a new architecture for operator networks VarMiON, that mimics the form of the numerical solution for an elliptic problem obtained from its variational form. In this thesis we exploit this technique, to predict the solution of a time-dependent heat equation written in weak space-time formulation. We define the associated solution operator and we approximate the PDE data e.g., with the FEM, which leads to the discrete solution operator moreover, a data sensing operator allows to obtain the input data to be supplied to the neural network. The resulting inputs are transformed into learnable matrices by branches. Finally, we consider the space spanned by the trained trunk functions and define the final VarMiON operator which is given by the product between branches and trunk net. The training of VarMiON consists in minimizing the loss function, computed as the L_2 norm of the error between the numerical solutions and the model prediction.

In conclusion, to build the surrogate model of bread leavening we combine all the equations involved in the model in a bigger network, by adapting the VarMiON technology to the elasticity equation, to our heat equation for temperature evolution and to carbon dioxide diffusion equation.

As last steps, we start to investigate inverse problems to rebuild the porous structure of bread during leavening. Especially, we present a scheme that allows to replace changes of material properties in limited regions within a domain with fictitious forcing terms [4]. We consider two open and simply connected bounded domains Ω_b and Ω_c in $\mathbb{R}^n$ such that $\overline{\Omega_\mathrm{c}} \subset \Omega_\mathrm{b}$, where Ω_c may represent a region where mechanical parameters differ from the reference value or even a cavity within the body Ω_b. With these provisions, the domain $\Omega_\mathrm{d} := \Omega_\mathrm{b} \setminus \overline{\Omega_\mathrm{c}}$ presents a single hole. Therefore, we prove that a heat transfer problem for the temperature field $\theta : \Omega_\mathrm{d} \times [0, \tau] \to \mathbb{R}$ on a domain with an internal unknown cavity has an equivalent formulation, stated on the filled domain, in which the void presence is simulated by a fictitious source term added to the equation. This requires defining an extension of the physical solution able to ensure that proper boundary conditions at the hidden cavity surface are met. Clearly, there is no unique way to obtain an extension with the sought properties. In [4] we propose a strategy that allows to select an extension when the cavity has a Lipschitz boundary. Especially, we choose the solution of a fourth-order differential operator that permits to impose boundary conditions on both the solution and its normal derivative on all of the boundary. A general and quite important result is that the fictitious forces are supported within the region of material changes or void thus, providing a way to identify the void geometry by reconstructing that source. Our result provides a strategy to map the nonlinear geometric inverse problem of void identification into a more manageable one, that involves the identification of forcing terms given the knowledge of external boundary data. To set the stage for a systematic study of the inverse problem, we present algebraic reconstructions to approximate the fictitious source from different sets of temperature measurements, based on the discretized problem. It is important to note that, solving such a system allows us to identify and localize the void, if its solution approximates the fictitious heat source with sufficient accuracy. Here, the main observation that a pure algebraic reconstruction gives the projection of void on the domain region where temperatures are known is key to realize that the inverse problem solution method should, estimate the full field of temperatures, to be able to locate accurately the support of the fictitious heat source, and thus the void geometry.

In our application, we want to solve a source inverse problem to reconstruct the porous texture of bread during leavening. To this end, as future work, we will start with some thermographies, that give us the value of the temperature field on a part of the boundary and we will reconstruct the full field of temperatures and the fictitious forcing term, with an augmented Kalman Filter.

Outline of the manuscript

Since we cover many different topics in this thesis, we have chosen to recall some basic notions on each subject at the beginning of each chapter. In this way, before our contribution, the reader can take advantage of this short mathematical introduction for some references on the topic, or skip it directly. We summarize the content of each chapter below.

- The model presented in Chapter 2 and in Chapter 3, and the energetic study of Chapter 4 are taken from our article [1].
- Chapter 5 is developed on the idea presented in the article [2].
- Chapter 6 is taken form our article [4].
- Chapter 7 follows the idea shown in [24].

Chapter 1

Basic notions of continuum mechanics

In this chapter, we want to recall some basic notions of continuum mechanics that will be useful for the development of a continuum model of bread leavening. To this goal, we focus on elasticity and PDEs which describe diffusive/heat phenomena.

1.1 Tensor algebra and tensor analysis

In this section, we set notations of tensor algebra and tensor analysis. For some references on this topic see [25, 26]. The space under consideration is the three-dimensional Euclidean space $\mathcal{E}$. The term point is reserved for elements of $\mathcal{E}$ and the term vector for elements of the associated vector space V (isomorphic to $\mathbb{R}^3$).

The inner product (a scalar) and cross product (a vector) between two vectors are denoted by

$$\boldsymbol{u} \cdot \boldsymbol{v} \quad \text{and} \quad \boldsymbol{u} \times \boldsymbol{v}. \tag{1.1}$$

The inner product determines the magnitude (or length) of a vector $\boldsymbol{u} \in V$ via the relation

$$|\boldsymbol{u}| = \sqrt{\boldsymbol{u} \cdot \boldsymbol{u}}. \tag{1.2}$$

Theorem 1.1.1 (Representation theorem for linear forms)**.** *Let $\phi : V \to \mathbb{R}$ be linear. Then there exists a unique vector $\boldsymbol{a} \in V$ such that:*

$$\phi(\boldsymbol{v}) = \boldsymbol{a} \cdot \boldsymbol{v} \tag{1.3}$$

for every $\boldsymbol{v} \in V$.

We define a tensor as a linear transformation from V to V. Therefore a tensor $\mathbf{S}$ is a linear map that assigns to each vector $\boldsymbol{u} \in V$ a vector

$$\boldsymbol{v} = \mathbf{S}\,\boldsymbol{u}. \tag{1.4}$$

We write $\mathbf{S}^\top$ the transpose of $\mathbf{S}$, for which holds the property

$$\mathbf{S}\boldsymbol{u} \cdot \boldsymbol{v} = \boldsymbol{u} \cdot \mathbf{S}^\top \boldsymbol{v}, \tag{1.5}$$

for all vectors $\boldsymbol{u}, \boldsymbol{v} \in V$.

A tensor is symmetric if $\mathbf{S}^\top = \mathbf{S}$ and skew-symmetric if $\mathbf{S}^\top = -\mathbf{S}$.

Proposition 1.1.2. *Every tensor $\mathbf{S}$ can be written as the sum of a symmetric tensor $\mathbf{E}$ and a skew-symmetric tensor $\mathbf{W}$:*

$$\mathbf{S} = \mathbf{E} + \mathbf{W} \tag{1.6}$$

where $\mathbf{E} = \frac{1}{2}(\mathbf{S} + \mathbf{S}^\top)$ and $\mathbf{W} = \frac{1}{2}(\mathbf{S} - \mathbf{S}^\top)$.

The tensor product between two vectors $\boldsymbol{u} \in V$ and $\boldsymbol{v} \in V$ is the tensor such that for all $\boldsymbol{s} \in V$:

$$(\boldsymbol{u} \otimes \boldsymbol{v})\, \boldsymbol{s} := (\boldsymbol{v} \cdot \boldsymbol{s})\, \boldsymbol{u}. \tag{1.7}$$

The trace is the linear operation that associates to each tensor $\mathbf{S}$ a scalar tr$\mathbf{S}$ such that:

$$\mathrm{tr}(\boldsymbol{u} \otimes \boldsymbol{v}) = \boldsymbol{u} \cdot \boldsymbol{v} \tag{1.8}$$

for all vectors $\boldsymbol{u} \in V$ and $\boldsymbol{v} \in V$.

If we take an orthonormal basis $\{\boldsymbol{e}_i\}_i$ for the vector space V then, by linearity, the trace is well defined

$$\mathrm{tr}\mathbf{S} = \mathrm{tr}(\sum_{i,j} S_{ij} \boldsymbol{e}_i \otimes \boldsymbol{e}_j) = \sum_{i,j} S_{ij} \mathrm{tr}(\boldsymbol{e}_i \otimes \boldsymbol{e}_j) = \sum_{i,j} S_{ij} \mathrm{tr}(\boldsymbol{e}_i \cdot \boldsymbol{e}_j) = \sum_i S_{ii}. \tag{1.9}$$

A tensor $\mathbf{S}$ is invertible if there exists a tensor $\mathbf{S}^{-1}$ such that $\mathbf{S}\mathbf{S}^{-1} = \mathbf{S}^{-1}\mathbf{S} = \mathbb{1}$, where $\mathbb{1}$ is the identity tensor such that, for every vector $\boldsymbol{v}$, holds $\mathbb{1}\boldsymbol{v} = \boldsymbol{v}$.

A tensor $\mathbf{Q}$ is orthogonal if it preserves inner products, that means:

$$\mathbf{Q}\boldsymbol{u} \cdot \mathbf{Q}\boldsymbol{v} = \boldsymbol{u} \cdot \boldsymbol{v} \tag{1.10}$$

for all vectors $\boldsymbol{u} \in V$ and $\boldsymbol{v} \in V$.

Notice that $\mathbf{Q}$ is orthogonal if and only if $\mathbf{Q}\mathbf{Q}^\top = \mathbf{Q}^\top\mathbf{Q} = \mathbb{1}$.

We recall that the determinant of a tensor is the determinant of its matrix. We call rotation an orthogonal tensor with a positive determinant.

Theorem 1.1.3 (Polar decomposition theorem)**.** *Let* $\mathbf{F}$ *a tensor with* $\det \mathbf{F} > 0$*. Then there exist positive definite, symmetric tensors* $\mathbf{U}$ *and* $\mathbf{V}$ *and a rotation* $\mathbf{R}$ *such that:*

$$\mathbf{F} = \mathbf{R}\mathbf{U} = \mathbf{V}\mathbf{R} \tag{1.11}$$

which are called the right and left polar decomposition of $\mathbf{F}$*.*

A function on $\mathcal{E}$ is called a scalar, vector, tensor field according to its values are scalars, vectors, tensors. Thus let ϕ a scalar field and $D\phi(\boldsymbol{X})$ the linear map from V to $\mathbb{R}$ for each $\boldsymbol{X} \in \mathcal{E}$ then, for the representation theorem there exists the gradient vector $\nabla\phi(\boldsymbol{X})$ such that:

$$D\phi(\boldsymbol{X})(\boldsymbol{u}) = \nabla\phi(\boldsymbol{X}) \cdot \boldsymbol{u}. \tag{1.12}$$

Similarly, if $\boldsymbol{v} \in V$ is a smooth vector, then

$$D\boldsymbol{v}(\boldsymbol{X})\boldsymbol{u} = \nabla\boldsymbol{v}(\boldsymbol{X})\boldsymbol{u}. \tag{1.13}$$

Therefore follow the definitions of divergence of a vector $\boldsymbol{v}$

$$\operatorname{div} \boldsymbol{v} = \operatorname{tr} \nabla \boldsymbol{v}, \tag{1.14}$$

and of a tensor $\mathbf{S}$

$$(\operatorname{div} \mathbf{S})\boldsymbol{v} = \operatorname{div}(\mathbf{S}^\top \boldsymbol{v}) \tag{1.15}$$

for every vector $\boldsymbol{v}$.

Let ϕ be a scalar field, the Laplacian is defined by $\Delta\phi = \operatorname{div} \nabla\phi$.

In the following, we define with Lin the set of all tensors, with Lin^+ the set of all tensors with a positive determinant, and with Sym and Skw the sets of all symmetric and skew-symmetric tensors respectively.

Theorem 1.1.4 (Divergence theorem)**.** *Let B a bounded regular region and $\phi : B \to \mathbb{R}$, $\boldsymbol{v} : B \to V$ and $\mathbf{S} : B \to Lin$ smooth fields. Then:*

$$\int_{\partial B} \phi\, \boldsymbol{m}\, dS = \int_B \nabla\phi\, d\boldsymbol{X}, \tag{1.16}$$

$$\int_{\partial B} \boldsymbol{v} \cdot \boldsymbol{m}\, dS = \int_B \operatorname{div} \boldsymbol{v}\, d\boldsymbol{X}, \tag{1.17}$$

$$\int_{\partial B} \mathbf{S}\boldsymbol{m}\, dS = \int_B \operatorname{div} \mathbf{S}\, d\boldsymbol{X}, \tag{1.18}$$

where $\boldsymbol{m}$ is the outward unit normal to ∂B.

1.2 Kinematics

In continuum mechanics, the basic property of a body is that it may occupy regions of Euclidean point space $\mathcal{E}$. We identify a body with the region B occupied in some fixed configuration, called a reference configuration. We refer to a point $\boldsymbol{X} \in B$ as a material point.

Let us consider a time interval $[0, \tau]$ where a motion of B takes place by a smooth function χ that assigns at each time, to each material point $\boldsymbol{X}$ a new point $\boldsymbol{x}$ as the spatial point occupied by $\boldsymbol{X}$ at time t

$$\boldsymbol{x} = \chi_t(\boldsymbol{X}). \tag{1.19}$$

Then $\chi_t(\boldsymbol{X})$ is often considered as a function of $\boldsymbol{X}$ is called the deformation at time t.

We call the region of space occupied by the body at time t: $B_t = \chi_t(B)$ as the deformed body at time t. In accordance with this:

(i) vectors associated with the ambient space, through which B_t evolves, are referred to as spatial vectors;

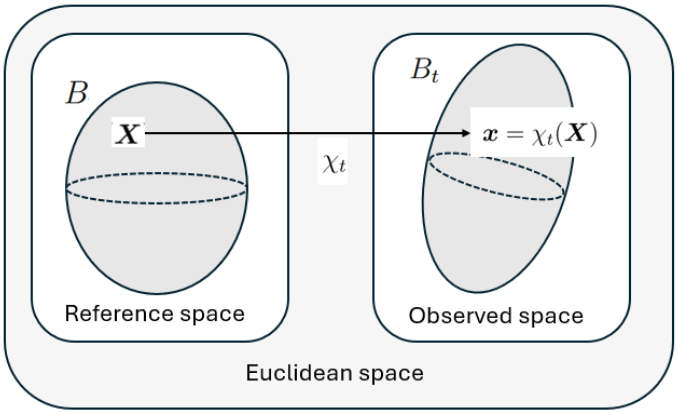

Figure 1.1: The reference body B and the deformed body B_t.

(ii) vectors associated with the ambient space for the reference body B are referred to as material vectors.

We define the tensor field $\mathbf{F}$ as the deformation gradient

$$\mathbf{F} := \nabla\chi \tag{1.20}$$

where $J = \det \mathbf{F} > 0$. Notice that $\mathbf{F}$ maps material vectors to spatial vectors.

Let us consider the reference map $\boldsymbol{x} = \chi_t(\boldsymbol{X})$. Hence $\boldsymbol{t}$ and the tangent $\boldsymbol{t}_R$ to a curve C on a surface S passing at $\boldsymbol{X}$ are related as

$$\boldsymbol{t} = \mathbf{F}(\boldsymbol{X})\boldsymbol{t}_R. \tag{1.21}$$

Further, since C lies on S, its tangent $\boldsymbol{t}_R$ at $\boldsymbol{X}$ must be tangent to S at $\boldsymbol{X}$, so that $\boldsymbol{t}_R \cdot \boldsymbol{m} = 0$ thus

$$\boldsymbol{n} = \mathbf{F}^{-\top}(\boldsymbol{X})\boldsymbol{m}. \tag{1.22}$$

With these provisions, it follows that the volume spanned by the basis $\{\boldsymbol{e}_i\}_i$ reads as

$$(\mathbf{F}\boldsymbol{e}_1 \times \mathbf{F}\boldsymbol{e}_2) \cdot \mathbf{F}\boldsymbol{e}_3 = \det \mathbf{F} = J \tag{1.23}$$

see Figure 1.2.

Instead by taking $\boldsymbol{e}_3 = \boldsymbol{m}$, the area element becomes

$$(\mathbf{F}\boldsymbol{e}_1 \times \mathbf{F}\boldsymbol{e}_2) = \det \mathbf{F}\, \mathbf{F}^{-\top}(\boldsymbol{e}_1 \times \boldsymbol{e}_2) = J\mathbf{F}^{-\top}\boldsymbol{m}, \tag{1.24}$$

$$|(\mathbf{F}\boldsymbol{e}_1 \times \mathbf{F}\boldsymbol{e}_2)| = J|\mathbf{F}^{-\top}\boldsymbol{m}| \tag{1.25}$$

see Figure 1.3.

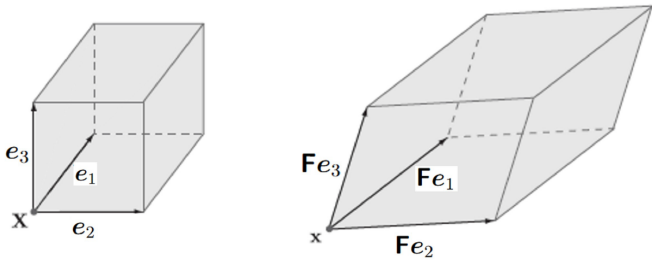

Figure 1.2: The volume element.

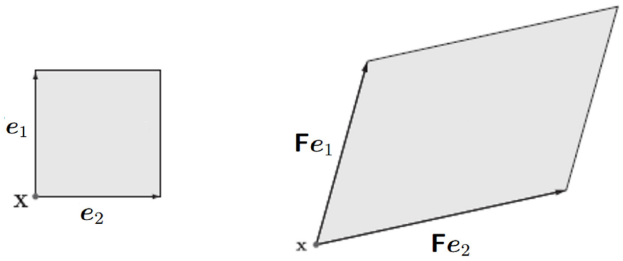

Figure 1.3: The area element.

1.3 Balance laws

Let us recall two important properties of bodies: mass conservation and balance of momentum.

1.3.1 Mass conservation

Every body has a mass which is distributed continuously and it can be expressed as the integral of a density field. A mass distribution for a body is a family of smooth density fields $\rho_f : f(B) \to \mathbb{R}^+$ one for each deformation f such that:

$$M(P) = \int_{f(P)} \rho_f \, d\boldsymbol{x} = \int_{g(P)} \rho_g \, d\boldsymbol{x}, \tag{1.26}$$

for any part P of the body B and all deformations f, g. This equation expresses the conservation of mass since the mass of a part cannot be altered by deforming the part itself.

1.3.2 Balance of momentum

We now recall the definitions of linear and angular momentum:

$$\boldsymbol{l}(P,t) = \int_{P_t} \boldsymbol{v}\rho \, d\boldsymbol{x}, \tag{1.27}$$

$$\boldsymbol{a}(P,t) = \int_{P_t} \boldsymbol{r} \times \boldsymbol{v}\rho \, d\boldsymbol{x}, \tag{1.28}$$

where $\boldsymbol{r}$ is a position vector with respect to the origin.

During a motion, mechanical interactions between parts of a body or between the body and other external elements are represented by forces of the following types:

- contact forces between different parts of the body;
- contact forces acting on the boundary of the body by the external environment;
- contact forces acting on the internal points of the body by the external environment.

Now we define $\boldsymbol{s}(\boldsymbol{n}, \boldsymbol{x}, t)$ the smooth function on a body that represents the surface force and $\boldsymbol{b}(\boldsymbol{x}, t)$ the continuous function that represents the body force, and define $\boldsymbol{f}$ and $\boldsymbol{m}_{\mathrm{o}}$ the force and momentum respectively on a part P at time t by

$$\boldsymbol{f}(P,t) = \int_{\partial P_t} \boldsymbol{s}(\boldsymbol{n}) ds + \int_{P_t} \boldsymbol{b} \, d\boldsymbol{x}, \tag{1.29}$$

$$\boldsymbol{m}_{\mathrm{o}}(P,t) = \int_{\partial P_t} \boldsymbol{r} \times \boldsymbol{s}(\boldsymbol{n}) ds + \int_{P_t} \boldsymbol{r} \times \boldsymbol{b} \, d\boldsymbol{x}. \tag{1.30}$$

Thus, the momentum balance laws assert

$$\boldsymbol{f}(P,t) = \dot{\boldsymbol{l}}(P,t) \tag{1.31}$$

$$\boldsymbol{m}_{\mathrm{o}}(P,t) = \dot{\boldsymbol{a}}(P,t) \tag{1.32}$$

that can be written as

$$\int_{\partial P_t} \boldsymbol{s}(\boldsymbol{n}) ds + \int_{P_t} \boldsymbol{b} \, d\boldsymbol{x} = \int_{P_t} \dot{\boldsymbol{v}}\rho \, d\boldsymbol{x}, \tag{1.33}$$

$$\int_{\partial P_t} \boldsymbol{r} \times \boldsymbol{s}(\boldsymbol{n}) ds + \int_{P_t} \boldsymbol{r} \times \boldsymbol{b} \, d\boldsymbol{x} = \int_{P_t} \boldsymbol{r} \times \dot{\boldsymbol{v}}\rho \, d\boldsymbol{x}. \tag{1.34}$$

Theorem 1.3.1 (Cauchy's theorem: existence of stress)**.** *Let $(\boldsymbol{s}, \boldsymbol{b})$ be a system of forces for B during a motion. Then a necessary and sufficient condition that the momentum balance laws be satisfied is that there exists a spatial tensor field $\mathbf{T}$ called the Cauchy stress such that:*

- *for each unit vector $\boldsymbol{m}$, $\boldsymbol{s}(\boldsymbol{m}) = \mathbf{T}\boldsymbol{m}$;*

- $\mathbf{T}$ *is symmetric;*
- $\mathbf{T}$ *satisfies the equation of motion:*

$$\operatorname{div}\mathbf{T} + \boldsymbol{b} = \rho\dot{\boldsymbol{v}}. \tag{1.35}$$

By Cauchy's theorem, to each force system consistent with the momentum balance laws there corresponds exactly one symmetric tensor field $\mathbf{T}$. Conversely the force system $(\boldsymbol{s}, \boldsymbol{b})$ is completely determined by $\mathbf{T}$ and by the motion $\chi_t(\boldsymbol{X})$. Indeed, the surface force $\boldsymbol{s}$ and the body force $\boldsymbol{b}$ are given by

$$\boldsymbol{s}(\boldsymbol{m}) = \mathbf{T}\boldsymbol{m}, \qquad \boldsymbol{b} = \rho\dot{\boldsymbol{v}} - \operatorname{div}\mathbf{T}. \tag{1.36}$$

Theorem 1.3.2 (Balance of momentum for a control volume)**.** *Let P a control volume at time t then, at that time*

$$\int_{\partial P} \boldsymbol{s}(\boldsymbol{m})\, dS + \int_P \boldsymbol{b}\, d\boldsymbol{X} = \frac{d}{dt}\int_P \boldsymbol{v}\rho\, d\boldsymbol{X} + \int_{\partial P} (\rho\boldsymbol{v})\boldsymbol{v}\cdot\boldsymbol{m}\, dS, \tag{1.37}$$

$$\int_{\partial P} \boldsymbol{r}\times\boldsymbol{s}(\boldsymbol{m})\, dS + \int_P \boldsymbol{r}\times\boldsymbol{b}\, d\boldsymbol{X} = \frac{d}{dt}\int_P \boldsymbol{r}\times\boldsymbol{v}\rho\, d\boldsymbol{X} + \int_{\partial P} \boldsymbol{r}\times(\rho\boldsymbol{v})\boldsymbol{v}\cdot\boldsymbol{m}\, dS. \tag{1.38}$$

Thanks to this theorem, we get that the total force on the control volume P is equal to the rate, at which the linear momentum of the material in P is increasing, plus the rate of outflow of momentum across ∂P.

1.4 Elasticity

In classical mechanics, the force on an elastic spring depends only on the change in length of the spring. For a body, the deformation gradient $\mathbf{F}$ measures the local distance changes.

1.4.1 Elastic bodies

An elastic body is a material body whose constitutive class $\mathcal{C}$ is defined by a smooth response function $\hat{\mathbf{T}}$ such that the stress $\mathbf{T}$ at $\boldsymbol{x} = \chi(\boldsymbol{X}, t)$ is:

$$\mathbf{T}(\boldsymbol{x}, t) = \hat{\mathbf{T}}(\mathbf{F}(\boldsymbol{X}, t), \boldsymbol{X}), \tag{1.39}$$

where the deformation gradient $\mathbf{F}$ is known.

Notice that, thanks to the polar-decomposition of $\mathbf{F} = \mathbf{RU}$ we get $\mathbf{C}$, $\mathbf{U}$, $\mathbf{B}$ and $\mathbf{V}$ useful for measuring the deformation of a material, where $\mathbf{C} = \mathbf{U}^2 = \mathbf{F}^\top\mathbf{F}$ and $\mathbf{B} = \mathbf{V}^2 = \mathbf{F}\mathbf{F}^\top$ are called the right and left Cauchy-Green strain tensor corresponding to $\mathbf{F}$. In addition, since the elastic stress cannot depend on rigid rotations, $\hat{\mathbf{T}}$ ($\bar{\mathbf{T}}, \tilde{\mathbf{T}}$ and $\mathbf{T}^*$) it is a function only of the strain part of $\mathbf{F}$, $\mathbf{U}$ ($\mathbf{V}$ or $\mathbf{C}$ or $\mathbf{B}$ depending on convenience):

- $\hat{\mathbf{T}}(\mathbf{F}) = \mathbf{F}\mathbf{T}^*(\mathbf{U})\mathbf{F}^\top$,
- $\hat{\mathbf{T}}(\mathbf{F}) = \mathbf{R}\tilde{\mathbf{T}}(\mathbf{C})\mathbf{R}^\top$,
- $\hat{\mathbf{T}}(\mathbf{F}) = \mathbf{F}\bar{\mathbf{T}}(\mathbf{C})\mathbf{F}^\top$.

Proposition 1.4.1. *The response functions $\hat{\mathbf{T}}, \bar{\mathbf{T}}, \tilde{\mathbf{T}}$ and $\mathbf{T}^*$ are invariant under symmetry transformation at $\boldsymbol{X}$.*

Moreover, if the material is invariant under orthogonal transformation at $\boldsymbol{X}$ then we say that it is isotropic, otherwise an-isotropic. We call the residual stress $\mathbf{T}_R = \hat{\mathbf{T}}(\mathbb{1})$ at $\boldsymbol{X}$ the stress when the body is non-deformed. If the material is isotropic at $\boldsymbol{X}$ then $\mathbf{T}_R$ is a pressure.

1.4.2 The Piola-Kirchoff stress

The Cauchy stress $\mathbf{T}$ measures the contact force per unit area in the deformed configuration. In many applications, it is useful to work with a stress tensor which gives the force measured per unit area in the reference configuration.

$$\int_{\partial P_t} \mathbf{T}\boldsymbol{n}\, ds = \int_{\partial P} J\mathbf{T}_m\mathbf{F}^{-\top}\boldsymbol{m}\, dS \tag{1.40}$$

where $\mathbf{T}_m$ is the material description of $\mathbf{T}$, thus the Piola-Kirchhoff stress is defined as:

$$\mathbf{S} := J\mathbf{T}_m\mathbf{F}^{-\top}. \tag{1.41}$$

Similarly, for the body force, that in the reference configuration is $\boldsymbol{b}_0 = J\boldsymbol{b}_m$.

Proposition 1.4.2. *The Piola-Kirchhoff stress $\mathbf{S}$ satisfies the balance equations:*

$$\int_{\partial P} \mathbf{S}\boldsymbol{m}\, dS + \int_P \boldsymbol{b}_0\, d\boldsymbol{X} = \int_P \ddot{\chi}_t \rho\, d\boldsymbol{X}, \tag{1.42}$$

$$\int_{\partial P} (\chi_t - \boldsymbol{o}) \times \mathbf{S}\boldsymbol{m}\, dS + \int_P (\chi_t - \boldsymbol{o}) \times \boldsymbol{b}_0\, d\boldsymbol{X} = \int_P (\chi_t - \boldsymbol{o}) \times \ddot{\chi}_t \rho\, d\boldsymbol{X}. \tag{1.43}$$

We can get that for elastic bodies $\mathbf{S}$ satisfies the field equations:

$$\operatorname{div}\mathbf{S} + \boldsymbol{b}_0 = \rho\ddot{\chi}_t \tag{1.44}$$

where $\mathbf{S} = J\mathbf{F}\bar{\mathbf{T}}(\mathbf{C})$.

Theorem 1.4.3 (Power Expended)**.** *Given any part P*

$$\int_{\partial P} \mathbf{S}\boldsymbol{m} \cdot \dot{\chi}_t\, dS + \int_P \boldsymbol{b}_0 \cdot \dot{\chi}_t\, d\boldsymbol{X} = \int_P \mathbf{S} \cdot \dot{\mathbf{F}}\, d\boldsymbol{X} + \frac{d}{dt}\int_P \frac{\dot{\chi}_t^{\,2}}{2}\rho_0\, d\boldsymbol{X}. \tag{1.45}$$

1.4.3 Hyperelasticity

At this point, we recall the implications for elastic bodies of the thermodynamic axiom which asserts that work must be non-negative in closed processes.

The work on P during a time interval $[0,\tau]$ for a process $(\chi,\mathbf{T})$ is given by

$$\int_0^\tau \left\{ \int_{\partial P_t} \mathbf{T}\boldsymbol{n}\cdot\boldsymbol{v}\,ds + \int_{P_t} \boldsymbol{b}\cdot\boldsymbol{v}\,d\boldsymbol{x} \right\} dt, \tag{1.46}$$

or equivalently in the reference configuration

$$\int_0^\tau \left\{ \int_{\partial P} \mathbf{S}\boldsymbol{m}\cdot\dot{\chi}_t\,dS + \int_P \boldsymbol{b}_0\cdot\dot{\chi}_t\,d\boldsymbol{X} \right\} dt. \tag{1.47}$$

Let us consider $(\chi,\mathbf{T})$ closed in $[0,\tau]$

$$\chi(\boldsymbol{X},0)=\chi(\boldsymbol{X},\tau),\quad \dot{\chi}(\boldsymbol{X},0)=\dot{\chi}(\boldsymbol{X},\tau) \tag{1.48}$$

for all $\boldsymbol{X}$, then the previous reduces to

$$\int_0^\tau \left\{ \int_{\partial P} \mathbf{S}\cdot\dot{\mathbf{F}}\,d\boldsymbol{X} \right\} dt. \tag{1.49}$$

In the following, we consider elastic bodies.

Theorem 1.4.4. *The work is non-negative in closed processes if and only if given any $\boldsymbol{X}$ and any interval $[0,\tau]$*

$$\int_0^\tau \left\{ \int_{\partial P} \mathbf{S}\cdot\dot{\mathbf{F}}\,d\boldsymbol{X} \right\} dt \geq 0, \tag{1.50}$$

in any process which is closed in $[0,\tau]$.

Definition 1.4.5. *An elastic body is hyperelastic if the Piola-Kirchhoff stress $\mathbf{S}$ is the derivative of a scalar function σ that is*

$$\mathbf{S}(\mathbf{F},\boldsymbol{X}) = D\sigma(\mathbf{F},\boldsymbol{X}), \tag{1.51}$$

where σ is called the strain-energy density. We can interpret $D\sigma(\mathbf{F},\boldsymbol{X})$ as a tensor which components are

$$D\sigma(\mathbf{F},\boldsymbol{X})_{ij} = \frac{\partial}{\partial \mathbf{F}_{ij}}\sigma(\mathbf{F},\boldsymbol{X}). \tag{1.52}$$

Theorem 1.4.6. *An elastic body is hyperelastic if and only if the work is non-negative in closed processes.*

Corollary 1.4.7 (Balance energy for hyperelastic materials)**.** *Each dynamical process for a hyperelastic body satisfies the energy equation*

$$\int_{\partial P} \mathbf{S}\boldsymbol{m} \cdot \dot{\chi}_t \, dS + \int_P \boldsymbol{b}_0 \cdot \dot{\chi}_t \, d\boldsymbol{X} = \frac{d}{dt} \int_P \left(\sigma + \frac{\dot{\chi}_t^2}{2} \rho_0 \right) d\boldsymbol{X}. \tag{1.53}$$

The term $\int_P \sigma \, d\boldsymbol{X}$ represents the strain energy of P.

We call the elasticity tensor $\mathbf{C} = D\mathbf{S}(\mathbb{1})$ for the material point $\boldsymbol{X}$, the derivative of the Piola-Kirchhoff stress with respect to $\mathbf{F}$ computed at $\mathbf{F} = \mathbb{1}$. Notice that if $\mathbf{E} = \frac{1}{2}(\mathbf{H} + \mathbf{H}^\top)$ is the symmetric part of $\mathbf{H} \in Lin$, then $\mathbf{C}$ is completely determined by its restriction to Sym. Thus follows the theorem:

Theorem 1.4.8. *If the material at $\boldsymbol{X}$ is isotropic. Then there exist scalars μ and λ such that:*

$$\mathbf{S} = 2\mu\mathbf{E} + \lambda(\mathrm{tr}\mathbf{E})\mathbb{1} \tag{1.54}$$

for every symmetric tensor $\mathbf{E}$. μ and λ are called Lamé moduli at $\boldsymbol{X}$.

For a displacement $\boldsymbol{u}$ hold the following

$$\mathbf{F} = \nabla \boldsymbol{u}, \tag{1.55}$$
$$\mathbf{S} = \mathbf{C}\mathbf{E}, \tag{1.56}$$
$$\mathbf{E} = \frac{1}{2}(\nabla \boldsymbol{u} + \nabla \boldsymbol{u}^\top), \tag{1.57}$$
$$\operatorname{div} \mathbf{S} + \boldsymbol{b}_0 = \rho_0 \ddot{\boldsymbol{u}}. \tag{1.58}$$

If the body is isotropic then (1.58) reads

$$\mu \Delta \boldsymbol{u} + (\lambda + \mu) \nabla \operatorname{div} \boldsymbol{u} + \boldsymbol{b}_0 = \rho_0 \ddot{\boldsymbol{u}}. \tag{1.59}$$

1.5 Heat equation

In this section, we give a survey about partial differential equations, and their solutions by focusing on heat and diffusive equations. For some references see [27].

A partial differential equation (PDE) is an equation involving an unknown function of two or more variables and certain of its partial derivatives.

Let us denote $Du = D_{\boldsymbol{x}} u = (\dots, u_{x_i}, \dots)$ the gradient of u with respect to the spatial variable $\boldsymbol{x}$.

Definition 1.5.1. *An expression of the form*

$$F(D^k u(\boldsymbol{x}), D^{k-1} u(\boldsymbol{x}), \dots, Du(\boldsymbol{x}), u(\boldsymbol{x}), \boldsymbol{x}) = 0 \tag{1.60}$$

is called a k^{th}-order partial differential equation, where $\boldsymbol{x} \in \Omega \subset \mathbb{R}^n$, $k \geq 1$ is an integer, $F : \mathbb{R}^{n^k} \times \mathbb{R}^{n^{k-1}} \times \cdots \times \mathbb{R}^n \times \mathbb{R} \times \Omega \to \mathbb{R}$ and $u : \Omega \to \mathbb{R}$ is the unknown.

Definition 1.5.2. *Definitions for PDEs.*

- *A PDE is called linear if it has the form*

$$\sum_{|\alpha| \leq k} a_\alpha(\boldsymbol{x}) D^\alpha u = f(\boldsymbol{x}) \tag{1.61}$$

  for given $a_\alpha, |\alpha| \leq k$ and f. It is homogeneous if $f = 0$.

- *A PDE is called semi-linear if it has the form*

$$\sum_{|\alpha| = k} a_\alpha(\boldsymbol{x}) D^\alpha u + a_0(D^{k-1}u(\boldsymbol{x}), \ldots, Du(\boldsymbol{x}), u(\boldsymbol{x}), \boldsymbol{x}) = 0. \tag{1.62}$$

- *A PDE is called quasi-linear if it has the form*

$$\sum_{|\alpha| = k} a_\alpha(D^{k-1}u(\boldsymbol{x}), \ldots, Du(\boldsymbol{x}), u(\boldsymbol{x}), \boldsymbol{x}) D^\alpha u + a_0(D^{k-1}u(\boldsymbol{x}), \ldots, Du(\boldsymbol{x}), u(\boldsymbol{x}), \boldsymbol{x}) = 0. \tag{1.63}$$

- *A PDE is fully nonlinear if it depends non-linearly upon the highest order derivatives.*

Definition 1.5.3. *An expression of the form*

$$\mathbf{F}(D^k \boldsymbol{u}(\boldsymbol{x}), D^{k-1}\boldsymbol{u}(\boldsymbol{x}), \ldots, D\boldsymbol{u}(\boldsymbol{x}), \boldsymbol{u}(\boldsymbol{x}), \boldsymbol{x}) = 0 \tag{1.64}$$

is called a k^{th}-order system of partial differential equation, where $\boldsymbol{x} \in \Omega \subset \mathbb{R}^n$, $k \geq 1$ is an integer, $\mathbf{F} : \mathbb{R}^{mn^k} \times \mathbb{R}^{mn^{k-1}} \times \cdots \times \mathbb{R}^{mn} \times \mathbb{R} \times \Omega \to \mathbb{R}^m$ and $\boldsymbol{u} : \Omega \to \mathbb{R}^m$, $\boldsymbol{u} = (u_1, \ldots, u_m)$ is the unknown.

From now on let us focus on second-order PDEs. Solving a PDE means finding all u that verify the previous equation, possibly among those functions satisfying certain auxiliary boundary conditions on some part Γ of the boundary $\partial\Omega$. If we assign the value of u on all the boundary

$$u = g \quad \text{ on } \partial\Omega, \tag{1.65}$$

where g is a given function, then we have the Dirichlet problem (if $g = 0$ homogeneous Dirichlet problem). Otherwise, we can assign the value of the normal derivative of u as

$$\nabla u \cdot \boldsymbol{n} = \frac{\partial u}{\partial \boldsymbol{n}} = h \quad \text{ on } \partial\Omega, \tag{1.66}$$

where $\boldsymbol{n}$ is the normal vector and h a given function. The problem associated with this boundary condition is said Neumann problem (if $h = 0$ homogeneous Neumann problem).

Finally, there are the mixed—or Robin—boundary conditions of type

$$u = g \quad \text{on } \Gamma_1, \tag{1.67}$$

$$\frac{\partial u}{\partial \boldsymbol{n}} = h \quad \text{on } \Gamma_2 \tag{1.68}$$

where $\partial\Omega = \Gamma_1 \cup \Gamma_2$ and $\Gamma_1 \cap \Gamma_2 = 0$.

Now we focus our attention on the non-homogeneous heat equation which involves a further derivative with respect to the time variable: ∂_t

$$\partial_t u - \Delta u = f(u), \tag{1.69}$$

subjected to appropriate initial and boundary conditions. Here $t > 0$ and $\boldsymbol{x} \in \Omega$ where $\Omega \subset \mathbb{R}^n$ is open. The unknown is $u(\boldsymbol{x}, t)$ and the Laplacian Δ is taken with respect to the spatial variables $\boldsymbol{x}$. In the non-homogeneous case, f is given.

1.5.1 Physical interpretation

The heat equation describes the evolution in time of the density u of some quantity such as heat:

$$u_t = \operatorname{div}(k\nabla u) = k\Delta u, \tag{1.70}$$

where k is a diffusivity of unit m^2/s.

We can also get the previous result by applying the Fourier law. In this context β represents the thermal conductivity of unit W/mK, $\boldsymbol{q}$ the heat flux and u a temperature,

$$\boldsymbol{q} = \frac{dQ}{dSdt} = -\beta\nabla u \tag{1.71}$$

where $\frac{dQ}{dSdt}$ is the heat flux that passes through the unit area in the time unit, and $k = \frac{\beta}{\rho c}$ is the thermal diffusivity with ρ a density and c a specific heat.

If Ω is any smooth region, the rate of change of total quantity within Ω equals the negative of the net flux $\boldsymbol{q}$ through $\partial\Omega$. Thus, the following holds

$$\frac{d}{dt}\int_\Omega \rho\, c\, u\, d\boldsymbol{X} = -\int_{\partial\Omega} \boldsymbol{q}\cdot\boldsymbol{m}\, dS \tag{1.72}$$

$$\frac{d}{dt}\int_\Omega \rho\, c\, u\, d\boldsymbol{X} = \int_{\partial\Omega} \beta\nabla u\cdot\boldsymbol{m}\, dS \tag{1.73}$$

$$\int_\Omega \rho\, c\frac{\partial}{\partial t}u\, d\boldsymbol{X} = \int_\Omega \operatorname{div}(\beta\nabla u)\, d\boldsymbol{X} \tag{1.74}$$

$$\frac{\partial u}{\partial t} = \operatorname{div}(k\nabla u). \tag{1.75}$$

Chapter 2

A continuum model for bread leavening

Bread is closely related to people's daily life and its making is an important operation in the food industry. This bread making is a complex process that involves physical, chemical and biochemical changes. Bread making can be divided into two sub-processes: leavening and baking. The leavening happens in a warm chamber and its result is a dough increased in volume due to the metabolic effect of yeast. The baking takes place in a hot oven that cooks the dough properly, making it a soft crumb inside and crunchy outside.

The leavening process is mostly treated by looking at chemical reactions [28] or by analyzing the effect of different amounts of wheat bran present in the dough [29], or from a microscale point of view through an analysis of CO_2 bubbles [28, 30–32].

In this chapter, we want to provide a continuum mathematical model for bread leavening that can form the basis for the development of digital twins for leavening chambers [1]. For the sake of definiteness, we chose paradigmatic conditions as regards the geometry of the bread, corresponding to common industrial bread production. Nevertheless, our model is sufficiently flexible to be applied to more general conditions.

We address the coupling of thermal conduction with the metabolism of yeast and with the consequent growth of the dough, which in turn affects the elastic properties of the material. All these effects contribute to the volume change observed during the leavening process. We develop a model that features a coupled system of ordinary and partial differential equations.

The *material domain* $\Omega \subset \mathbb{R}^3$ provides a fixed set of labels for the material points of the dough. The actual space occupied by the material is given by $\boldsymbol{u}(\Omega)$, where $\boldsymbol{u} : \Omega \to \mathbb{R}^3$ is the deformation map. We partition the boundary of Ω as the disjoint union of three parts: Γ_0 identifies the points that are in contact with the bottom side of the baking tray, Γ_1 the top points in contact with air, and Γ_2 the material points in contact with the lateral sides of the tray. The different quantities involved in the description of the process are represented by fields on the fixed material domain Ω. A list of symbols is given in Table 2.1.

2.1 Elasticity of the bread

To determine the actual shape of the dough under given conditions, we model it as a hyperelastic material of neo-Hookean type [25]. We denote by $\mathbf{F}$ the deformation gradient and set $J := \det \mathbf{F}$. Over time, the dough will expand and, to model this, we need to introduce the dilation field $J_{\mathrm{ref}} : \Omega \to \mathbb{R}^+$ that describes, locally, the target volume that the material would like to achieve to relax compressive stresses. By introducing the right Cauchy–Green tensor $\mathbf{C}$ and a body force per unit volume $\boldsymbol{b}$, that will represent the weight, the elastic energy functional is postulated to be

$$\mathcal{E}(\boldsymbol{u}) = \frac{1}{2}\int_\Omega \left[\mu\left(\operatorname{tr}\mathbf{C} - 3J_{\mathrm{ref}}^{2/3}\right) + \lambda|\log(J/J_{\mathrm{ref}})|^2\right] d\boldsymbol{X} - \int_\Omega \boldsymbol{b}\cdot\boldsymbol{u}\, d\boldsymbol{X}, \tag{2.1}$$

Functions	
$\boldsymbol{u}$	placement
θ	temperature
D	concentration of CO_2
Y	yeast concentration
K_{y}	yeast growth rate function
f	CO_2 production rate function
Constants	
λ and μ	Lame's moduli
E	Young's modulus
ν	Poisson's ratio
$\boldsymbol{b}$	body force
k	thermal diffusivity
h	heat transfer coefficient
c	specific heat
ρ	density
β	thermal conductivity
n_{mol}	the number of moles contained in a gram of CO_2
Subscripts	
0	initial condition
b	bread dough
f	flour
o	outside, leavening machine
w	water
t	steel tray
y	yeast
$co2$	carbon dioxide
bo	bread-to-air
bt	bread-to-tray

Table 2.1: List of symbols.

where λ and μ are the usual Lame's moduli

$$\mu = \frac{E}{2(1+\nu)}, \qquad \lambda = \frac{E\nu}{(1+\nu)(1-2\nu)}, \tag{2.2}$$

with E the Young modulus and ν the Poisson ratio.

In the analysis of the leavening process, it is reasonable to neglect vibrations of the dough, that is, inertial phenomena linked to the elastic response which take place on a much faster timescale than heat transfer, diffusion processes, and volume growth. We then follow the deformation of the material with a quasi-static approach. At each time, the deformation map $\boldsymbol{u}$ is obtained by solving the Euler–Lagrange equation $\delta\mathcal{E}(\boldsymbol{u}) = 0$, subject to the following boundary conditions:

$$\boldsymbol{u} = (0,0,0) \qquad \text{on } \Gamma_0, \tag{2.3}$$

$$\boldsymbol{u} = (0,0,u_z) \qquad \text{on } \Gamma_2. \tag{2.4}$$

These imply that the material points are fixed at the bottom, while they can slide vertically along the lateral sides of the tray. Lastly, on the free surface Γ_1 we have a traction-free condition, also known as natural boundary condition emerging from the variational problem of energy minimization. The equation for the deformation map is coupled with evolution equations for other quantities, introduced below, via the dependence of the target volume J_{ref} and the Young modulus E on the concentration of carbon dioxide, which, in turn, depends on the yeast activity and the temperature field.

2.2 Heat transfer

The second equation considered in the model is the heat equation, that describes the temperature evolution of the dough during leavening. Whereas heat fluxes are more easily described in reference to spatial domains in the actual configuration of the material, for the sake of simplicity on the numerical solution of the equations, we prefer to work with the fixed domain Ω for all of the fields involved in the model. We then need to rewrite the classical heat equation for a homogeneous domain in material coordinates. The details of this procedure follow.

Let us denote with $\boldsymbol{X}$ the material coordinates on Ω and with $\boldsymbol{x}$ the spatial one. In the following we write in material coordinates a general field $\theta : \Omega \times \mathbb{R} \to \mathbb{R}$ with its spatial counterpart $\hat{\theta}$ linked by the following relations:

$$\theta(\boldsymbol{X},t) = \hat{\theta}(\boldsymbol{u}(\boldsymbol{X},t),t), \qquad \hat{\theta}(\boldsymbol{x},t) = \theta(\boldsymbol{u}^{-1}(\boldsymbol{x},t),t). \tag{2.5}$$

Concerning the gradients, we have

$$\nabla_{\boldsymbol{X}}\boldsymbol{u} = \mathbf{F}, \qquad \nabla_{\boldsymbol{x}} = \mathbf{F}^{-\top}\nabla_{\boldsymbol{X}}, \tag{2.6}$$

$$\nabla_{\boldsymbol{x}}\hat{\theta} = \nabla_{\boldsymbol{X}}\theta\Big(\frac{\partial \boldsymbol{u}(\boldsymbol{x},t)}{\partial \boldsymbol{X}}\Big)^{-1} = \mathbf{F}^{-\top}\nabla_{\boldsymbol{X}}\theta, \tag{2.7}$$

and for the time derivatives we obtain

$$\begin{aligned}\frac{\partial\hat{\theta}}{\partial t}(\boldsymbol{x},t) &= \frac{\partial\theta(\boldsymbol{u}^{-1}(\boldsymbol{x},t),t)}{\partial t} = \frac{\partial\theta(\boldsymbol{u}^{-1}(\boldsymbol{x},t),t)}{\partial \boldsymbol{X}}\frac{\partial\boldsymbol{u}^{-1}(\boldsymbol{x},t)}{\partial t} + \frac{\partial\theta(\boldsymbol{u}^{-1}(\boldsymbol{x},t),t)}{\partial t} \\ &= \nabla_{\boldsymbol{X}}\theta\frac{\partial\boldsymbol{u}^{-1}(\boldsymbol{x},t)}{\partial t} + \frac{\partial\theta(\boldsymbol{u}^{-1}(\boldsymbol{x},t),t)}{\partial t} = \nabla_{\boldsymbol{X}}\theta\frac{\partial\boldsymbol{u}^{-1}(\boldsymbol{x},t)}{\partial t} + \frac{\partial\theta(\boldsymbol{X},t)}{\partial t} = \frac{\partial\theta(\boldsymbol{X},t)}{\partial t}\end{aligned} \tag{2.8}$$

where the last equality holds because we consider a static deformation of the bread, hence the absence of convective terms. The volume of the spatial domain $\boldsymbol{u}(\Omega)$ and surface of $\partial\boldsymbol{u}(\Omega)$ in material can be computed as

$$\int_{\boldsymbol{u}(\Omega)} d\boldsymbol{x} = \int_{\Omega} J\, d\boldsymbol{X}, \tag{2.9}$$

$$\int_{\partial\boldsymbol{u}(\Omega)} ds = \int_{\partial\Omega} |J\mathbf{F}^{-\top}\boldsymbol{m}|\, dS. \tag{2.10}$$

With the foregoing identities, we can easily obtain the following equations (2.11), (2.17) in material coordinates.

Let us now consider the evolution over a fixed time interval $[0,\tau]$ of the temperature field $\theta_{\mathrm{b}} : \Omega\times[0,\tau]\to\mathbb{R}$ as given by the equation

$$\rho_{\mathrm{b}}c_{\mathrm{b}}J\frac{\partial}{\partial t}\theta_{\mathrm{b}} = \mathrm{div}(\beta_{\mathrm{b}}J\mathbf{C}^{-1}\nabla\theta_{\mathrm{b}}) \qquad \text{on } \Omega\times[0,\tau], \tag{2.11}$$

where ρ_{b} is the density of the bread dough, c_{b} the specific heat and β_{b} is the thermal conductivity. It is worth noticing that the isotropic and homogeneous diffusion that takes place in spatial coordinates becomes anisotropic and non-homogeneous in material coordinates, as is evident from the presence of the term $J\mathbf{C}^{-1}$ in the heat flux. For this equation, we need to prescribe initial conditions and we add boundary conditions (of Robin type) describing the energy exchanged between the bread and the air in the leavening chamber. We set $\Gamma_3 := \Gamma_0\cup\Gamma_2$ to represent the portion of the boundary that is in contact with the baking tray. We do not resolve the heat flow within the tray, but simply summarize its effect in the transfer coefficient h_{bt} which is different from the one describing the direct transfer of heat between air and bread, namely h_{bo}. Denoting by $\boldsymbol{m}$ the unit outer normal to $\partial\Omega$, the boundary and initial conditions for the temperature field read

$$\theta_{\mathrm{b}}(\cdot,0) = \theta_{\mathrm{b}_0} \qquad \text{on } \Omega, \tag{2.12}$$

$$-\beta_{\mathrm{b}}J(\mathbf{C}^{-1}\nabla\theta_{\mathrm{b}})\cdot\boldsymbol{m} = |J\mathbf{F}^{-\top}\boldsymbol{m}|h_{\mathrm{bt}}(\theta_{\mathrm{b}}-\theta_{\mathrm{o}}) \qquad \text{on } \Gamma_3\times[0,\tau], \tag{2.13}$$

$$-\beta_{\mathrm{b}}J(\mathbf{C}^{-1}\nabla\theta_{\mathrm{b}})\cdot\boldsymbol{m} = |J\mathbf{F}^{-\top}\boldsymbol{m}|h_{\mathrm{bo}}(\theta_{\mathrm{b}}-\theta_{o}) \qquad \text{on } \Gamma_1\times[0,\tau]. \tag{2.14}$$

2.3 Yeast growth

A key role in the leavening is played by the presence and metabolism of yeast. The function $Y : \Omega\times[0,\tau]\to\mathbb{R}^+$ represents the concentration of yeast in the dough with respect to the

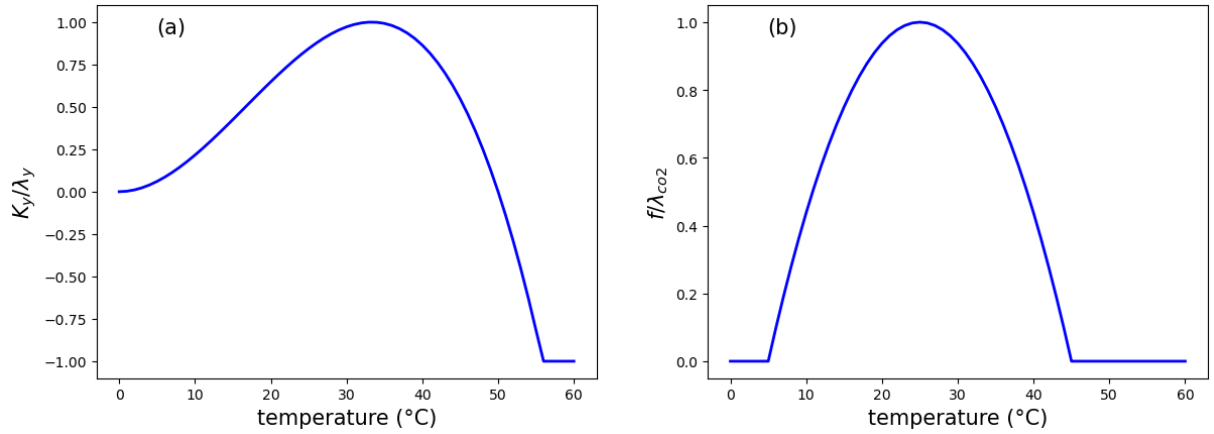

Figure 2.1: (a) The normalized growth rate function $K_{\mathrm{y}}/\lambda_{\mathrm{y}}$ as a function of the temperature. (b) The normalized production rate CO_2 f/λ_{co2} as a function of temperature.

flour quantity, that we suppose to be uniform at the initial time. Given an initial condition Y_0, the yeast concentration evolves according to the Cauchy problem [28, 31, 32]

$$\begin{cases} \dfrac{d}{dt}Y = K_{\mathrm{y}}(\theta_{\mathrm{b}})Y & \text{on } \Omega \times [0,\tau], \\ Y(\cdot,0) = Y_0 & \text{on } \Omega, \end{cases} \tag{2.15}$$

where K_{y} is the growth rate function modeled as a cubic polynomial (Figure 2.1(a)) that vanishes at 0°C and at the yeast death temperature θ_{y}:

$$K_{\mathrm{y}}(\theta) := \max\left\{\lambda_{\mathrm{y}}\frac{27\,\theta^2(\theta_{\mathrm{y}}-\theta)}{4\,\theta_{\mathrm{y}}^3}, K_{\min}\right\}, \tag{2.16}$$

where $K_{\min} := \lambda_{\mathrm{y}}\frac{27\,60^2(\theta_{\mathrm{y}}-60)}{4\,\theta_{\mathrm{y}}^3}$. A typical value for $\theta_{\mathrm{y}} = 50$°C instead the constant λ_{y} of unit s^{-1} is calibrated following Ref. [29]. Notice that the growth model for yeasts follows the idea presented in [31] where the Malthusian parameter is substituted by the function $K_y(\theta_{\mathrm{b}})$ however, other common choices for Y could be the logistic or Gompertz's function as mentioned in [28]. Since in our system, due to the comparatively small quantity of yeast, we do not expect saturation phenomena in the relevant time-frame, a Malthusian growth represents well the actual phenomenon.

2.4 Carbon dioxide production and diffusion

The metabolism of yeast involves the production of carbon dioxide. The CO_2 concentration with respect to flour, denoted by $D : \Omega \times [0,\tau] \to \mathbb{R}$, has a diffusive evolution with a source term depending on the rate of CO_2 production f, on the temperature reached by the bread

θ_{b} and on the yeast concentration Y. The equation for D is then

$$J\frac{\partial}{\partial t}D = \mathrm{div}(\beta_{co2}J\mathbf{C}^{-1}\nabla D) + J\,f(\theta_{\mathrm{b}})Y \qquad \text{on } \Omega\times[0,\tau], \tag{2.17}$$

$$D(\cdot,0) = D_0 \qquad \text{on } \Omega. \tag{2.18}$$

The boundary condition is of homogeneous Neumann type and the CO_2 production rate $f(\theta)$ is modeled by

$$f(\theta) := \max\left\{0, \lambda_{co2}\frac{(\theta-5)(45-\theta)}{400}\right\}, \tag{2.19}$$

with λ_{co2} a constant rate (Figure 2.1(b)).

2.5 Volume expansion

The carbon dioxide production due to fermentation implies a positive volume expansion associated with the leavening process. First of all, we introduce an estimate of the added CO_2 volume. For the sake of simplicity, we assume a linear relation between volume and temperature, by approximating the actual thermodynamic relation with the ideal gas law. At this stage, further corrections to that relation can be included in an empirical fitting coefficient. We thus compute the volume of the CO_2 gas considering the number of moles contained in a gram of CO_2 that is

$$PV^{1g}_{\mathrm{CO}_2} = n_{\mathrm{mol}}R\theta_{\mathrm{b}}, \qquad V^{1g}_{\mathrm{CO}_2} = \frac{n_{\mathrm{mol}}R\theta_{\mathrm{b}}}{P}. \tag{2.20}$$

With this value, we can then update the dilation coefficient of the target volume configuration according to

$$J_{\mathrm{ref}} = 1 + \frac{V^{1g}_{\mathrm{CO}_2}DF_{\mathrm{rate}}}{V^{1g}_0}, \tag{2.21}$$

where V^{1g}_0 is the total volume of one gram of dough and F_{rate} the flour fraction of the paste.

At the same time, the Young modulus changes depending on the concentration of CO_2. If we assume that the Young modulus E_{CO_2} of CO_2 is negligible compared to that of the dough, we can argue that its presence weakens the elastic response. Denoting by E_0 the Young modulus of the dough at the initial time, we postulate that, locally, $E = E_0/J_{\mathrm{ref}}$.

2.6 Continuum problem: system of equations

For the sake of clarity, we recap the system of equations presented above, to see more evidently the coupling among them.

$$
\begin{aligned}
&\delta\mathcal{E}(\boldsymbol{u}) = 0 && \text{on } \Omega,\\
&\rho_{\mathrm{b}} c_{\mathrm{b}} J \frac{\partial}{\partial t}\theta_{\mathrm{b}} = \operatorname{div}(\beta_{\mathrm{b}} J \mathbf{C}^{-1}\nabla\theta_{\mathrm{b}}) && \text{on } \Omega\times[0,\tau],\\
&\frac{d}{dt}Y = K_{\mathrm{y}}(\theta_{\mathrm{b}})Y && \text{on } \Omega\times[0,\tau],\\
&J\frac{\partial}{\partial t}D = \operatorname{div}(\beta_{co2} J \mathbf{C}^{-1}\nabla D) + J\, f(\theta_{\mathrm{b}})Y && \text{on } \Omega\times[0,\tau],
\end{aligned}
$$

where $\mathcal{E}(\boldsymbol{u}) = \frac{1}{2}\int_\Omega \left[\mu\left(\operatorname{tr}\mathbf{C} - 3J_{\mathrm{ref}}^{2/3}\right) + \lambda|\log(J/J_{\mathrm{ref}})|^2\right] d\boldsymbol{X} - \int_\Omega \boldsymbol{b}\cdot\boldsymbol{u}\, d\boldsymbol{X}$ and, the initial and boundary conditions are

$$
\begin{aligned}
&\theta_{\mathrm{b}}(\cdot,0) = \theta_{\mathrm{b}_0} && \text{on } \Omega,\\
&Y(\cdot,0) = Y_0 && \text{on } \Omega,\\
&D(\cdot,0) = D_0 && \text{on } \Omega,\\
&\boldsymbol{u} = (0,0,0) && \text{on } \Gamma_0,\\
&\boldsymbol{u} = (0,0,u_z) && \text{on } \Gamma_2,\\
&-\beta_{\mathrm{b}} J(\mathbf{C}^{-1}\nabla\theta_{\mathrm{b}})\cdot\boldsymbol{m} = |J\mathbf{F}^{-\top}\boldsymbol{m}| h_{\mathrm{bt}}(\theta_{\mathrm{b}} - \theta_{\mathrm{o}}) && \text{on } \Gamma_3\times[0,\tau],\\
&-\beta_{\mathrm{b}} J(\mathbf{C}^{-1}\nabla\theta_{\mathrm{b}})\cdot\boldsymbol{m} = |J\mathbf{F}^{-\top}\boldsymbol{m}| h_{\mathrm{bo}}(\theta_{\mathrm{b}} - \theta_{o}) && \text{on } \Gamma_1\times[0,\tau].
\end{aligned}
$$

Chapter 3

Functional setting and discretization of the leavening evolution

In this chapter, we recall some aspects of functional analysis useful in the perspective of defining the weak form and proving uniqueness of solution of our continuum model. Then, we recall the Euler method for the time discretization, which we apply subsequently to the leavening model. We conclude the chapter by proving the convergence of the time discretization.

3.1 Elements of functional analysis

In this sections, we recall some concepts of functional analysis such as functionals and bilinear forms, distributions, Sobolev spaces and L^p spaces. For complete references with respect to this recap see [33, 34].

3.1.1 Functionals and bilinear forms

Definition 3.1.1. *Given a function space V, we call functional on V an operator associating a real number to each element of V*

$$f : V \to \mathbb{R}. \tag{3.1}$$

A functional is linear if

$$f(\alpha \boldsymbol{v} + \beta \boldsymbol{u}) = \alpha f(\boldsymbol{v}) + \beta f(\boldsymbol{u}) \qquad \forall \alpha, \beta \in \mathbb{R}\,, \qquad \forall \boldsymbol{v}, \boldsymbol{u} \in V. \tag{3.2}$$

A linear functional is bounded if there is a constant $C > 0$ such that

$$f(\boldsymbol{v}) \leq C \|\boldsymbol{v}\|_V \qquad \forall \boldsymbol{v} \in V. \tag{3.3}$$

Recall that, a linear and bounded functional on a Banach space (i.e. a normed and complete space) is also continuous. We then define the space V', dual of V, as the set of linear and bounded functionals on V

$$V' = \{f : V \to \mathbb{R} \text{ such that } f \text{ is linear and bounded}\} \tag{3.4}$$

and we equip it with the norm $\|\cdot\|_{V'}$ defined as

$$\|f\|_{V'} := \sup_{\boldsymbol{v} \in V/\{0\}} \frac{|f(\boldsymbol{v})|}{\|\boldsymbol{v}\|_V}. \tag{3.5}$$

Theorem 3.1.2 (Riesz representation theorem)**.** *Let H be a Hilbert space, that is a Banach space whose norm is induced by a scalar product $(\cdot, \cdot)_H$. For each linear and bounded functional f on H there exists a unique element $\boldsymbol{x}_f \in H$ such that*

$$f(\boldsymbol{y}) = (\boldsymbol{y}, \boldsymbol{x}_f)_H \quad \forall \boldsymbol{y} \in H, \quad \|f\|_{H'} = \|\boldsymbol{x}_f\|_H. \tag{3.6}$$

Conversely, each element $\boldsymbol{x} \in H$ identifies a linear and bounded functional $f_{\boldsymbol{x}}$ on H such that

$$f_{\boldsymbol{x}}(\boldsymbol{y}) = (\boldsymbol{y}, \boldsymbol{x})_H \quad \forall \boldsymbol{y} \in H, \quad \|f_{\boldsymbol{x}}\|_{H'} = \|\boldsymbol{x}\|_H. \tag{3.7}$$

Thanks to the Riesz representation theorem there exists a bijective and norm-preserving transformation between H' and H thanks to which H' and H can be identified.

Definition 3.1.3. *Given a normed function space V we call form an application that associates to each pair of elements of V a real number*

$$a : V \times V \to \mathbb{R}. \tag{3.8}$$

A form is

- bilinear if it is linear with respect to both its two arguments;
- continuous if there exists a constant $M > 0$ such that

$$|a(\boldsymbol{v}, \boldsymbol{u})| \leq M\|\boldsymbol{v}\|_V \|\boldsymbol{u}\|_V \quad \forall \boldsymbol{v}, \boldsymbol{u} \in V; \tag{3.9}$$

- symmetric if

$$a(\boldsymbol{v}, \boldsymbol{u}) = a(\boldsymbol{u}, \boldsymbol{v}) \quad \forall \boldsymbol{v}, \boldsymbol{u} \in V; \tag{3.10}$$

- positive definite if

$$a(\boldsymbol{v}, \boldsymbol{v}) > 0 \quad \forall \boldsymbol{v} \in V \setminus \{\boldsymbol{0}\}; \tag{3.11}$$

- coercive if there exists $\alpha > 0$ such that

$$a(\boldsymbol{v}, \boldsymbol{v}) \geq \alpha \|\boldsymbol{v}\|_V^2 \quad \forall \boldsymbol{v} \in V. \tag{3.12}$$

3.1.2 Elements of distributions

In this section, we recall the main notions of the theory of distribution useful in the next sections to define Sobolev spaces. Let Ω be an open set of $\mathbb{R}^n$ and $f : \Omega \to \mathbb{R}$.

Definition 3.1.4. *By support of a function f we mean the closure of the set where the function itself takes values different from zero, that is*

$$\text{supp} f = \{\boldsymbol{x} : f(\boldsymbol{x}) \neq 0\}. \tag{3.13}$$

Notice that a function f is said to have a compact support Ω if there exists a compact set $K \subset \Omega$ such that $\text{supp} f \subset K$.

Definition 3.1.5. *$\mathcal{D}(\Omega)$ is the space of infinitely differentiable functions with compact support in Ω, that is*

$$\mathcal{D}(\Omega) = \{f \in \mathcal{C}^\infty(\Omega) : \exists K \subset \Omega, K \text{ compact}, \text{supp} f \subset K\}. \tag{3.14}$$

The space of distributions on Ω is therefore given by the dual space $\mathcal{D}'(\Omega)$ of $\mathcal{D}(\Omega)$.

The action of a distribution $T \in \mathcal{D}'(\Omega)$ on a function $\phi \in D(\Omega)$ is always denoted via the identity pairing $\langle T, \phi \rangle$.

Example 3.1.6. *Let $\boldsymbol{a}$ be a point of the set Ω. The Dirac delta relative to point $\boldsymbol{a}$ is the distribution $\delta_{\boldsymbol{a}}$ defined by the following relation*

$$\langle \delta_{\boldsymbol{a}}, \phi \rangle = \phi(\boldsymbol{a}) \qquad \forall \phi \in \mathcal{D}(\Omega). \tag{3.15}$$

Definition 3.1.7. *A sequence of distributions $\{T_n\}$ converges to a distribution T in $\mathcal{D}(\Omega)$ if holds*

$$\lim_{n\to\infty} \langle T_n, \phi \rangle = \langle T, \phi \rangle \qquad \forall \phi \in \mathcal{D}(\Omega). \tag{3.16}$$

Now we define the derivatives of $T \in \mathcal{D}'(\Omega)$ in the sense of distributions as

$$\langle \frac{\partial T}{\partial x_i}, \phi \rangle = -\langle T, \frac{\partial \phi}{\partial x_i} \rangle \qquad \forall \phi \in \mathcal{D}(\Omega),\ i = 1, \ldots, n. \tag{3.17}$$

Similarly, we define derivatives of arbitrary order. Precisely, for each multiindex $\alpha = (\alpha_1, \alpha_2, \ldots, \alpha_n)$, we have that $D^\alpha T$

$$\langle D^\alpha T, \phi \rangle = (-1)^{|\alpha|} \langle T, D^\alpha \phi \rangle \qquad \forall \phi \in \mathcal{D}(\Omega). \tag{3.18}$$

3.1.3 $L^p(\Omega)$ spaces

Square-integrable functions

We consider the space of square-integrable functions on $\Omega \subset \mathbb{R}^n$

$$L^2(\Omega) := \left\{ f : \Omega \to \mathbb{R} \text{ such that } \int_\Omega |f(\boldsymbol{x})|^2 d\boldsymbol{x} < +\infty \right\}. \tag{3.19}$$

Technically, $L^2(\Omega)$ is a space of equivalence classes of measurable functions, thus f is equivalent to g if and only if they are equal almost everywhere, i.e. they differ at most on a subset of Ω with zero measure. The space $L^2(\Omega)$ is a Hilbert space whose scalar product is

$$(f, g)_{L^2(\Omega)} = \int_\Omega f(\boldsymbol{x}) g(\boldsymbol{x}) d\boldsymbol{x}, \tag{3.20}$$

where the norm is induced by the scalar product as

$$\|f\|_{L^2(\Omega)} = \sqrt{(f,f)_{L^2(\Omega)}}. \tag{3.21}$$

Notice that, the space $\mathcal{D}(\Omega)$ is dense in $L^2(\Omega)$. Moreover, we can associate to each function $f \in L^2(\Omega)$ a distribution $T_f \in \mathcal{D}'(\Omega)$ as

$$\langle T_f, \phi \rangle = \int_\Omega f(\boldsymbol{x})\phi(\boldsymbol{x})d\boldsymbol{x} \qquad \forall \phi \in \mathcal{D}(\Omega). \tag{3.22}$$

The spaces $L^\infty(\Omega)$ and $L^p(\Omega)$, with $1 \leq p < \infty$

The space $L^2(\Omega)$ can be generalized in the following way: for each real number p with $1 \leq p < \infty$

$$L^p(\Omega) := \left\{ f : \Omega \to \mathbb{R} \text{ such that } \int_\Omega |f(\boldsymbol{x})|^p d\boldsymbol{x} < +\infty \right\}. \tag{3.23}$$

with the norm defined as

$$\|f\|_{L^p(\Omega)} = \left(\int_\Omega |f(\boldsymbol{x})|^p d\boldsymbol{x} \right)^{\frac{1}{p}}. \tag{3.24}$$

Notice that if p is $1 \leq p < \infty$ then $\mathcal{D}(\Omega)$ is dense in $L^p(\Omega)$. In the case where $p = \infty$, we define $L^\infty(\Omega)$ to be the space of functions that are bounded in Ω. Its norm is defined as

$$\|f\|_{L^\infty(\Omega)} = \inf\{C \in \mathbb{R} : |f(\boldsymbol{x})| \leq C, a.e. \text{ in } \Omega\} = \sup\{|f(\boldsymbol{x})|, a.e. \text{ in } \Omega\}. \tag{3.25}$$

For $1 \leq p \leq \infty$, the spaces $L^p(\Omega)$, provided with the norm $\|\cdot\|_{L^p(\Omega)}$, are Banach spaces.

We now provide two useful inequalities

- Holder inequality: given $f \in L^p(\Omega)$ and $g \in L^{p'}(\Omega)$ with $1 \leq p \leq \infty$ and $\frac{1}{p} + \frac{1}{p'} = 1$, then $fg \in L^1(\Omega)$ and

$$\int_\Omega |f(\boldsymbol{x})g(\boldsymbol{x})|d\boldsymbol{x} \leq \|f\|_{L^p(\Omega)}\|g\|_{L^{p'}(\Omega)}. \tag{3.26}$$

  The index p' is called the conjugate of p.

- For $p = 2$ the previous is known as Cauchy-Schwarz:

$$|(f,g)_{L^2(\Omega)}| \leq \|f\|_{L^2(\Omega)}\|g\|_{L^2(\Omega)} \quad \forall f, g \in L^2(\Omega). \tag{3.27}$$

Notice that if $1 < p < \infty$, then $L^p(\Omega)$ is a reflexive space i.e., any linear and continuous form $f : L^p(\Omega) \to \mathbb{R}$ can be identified with an element of $L^{p'}(\Omega)$.

3.1.4 Sobolev spaces

Finally, we introduce the Sobolev space $W^{k,p}(\Omega)$, with k a non-negative integer and $1 \leq p \leq \infty$, as the space of functions $f \in L^p(\Omega)$ such that all the distributional derivatives of f of order up to k are in $L^p(\Omega)$.

$$W^{k,p}(\Omega) = \{f \in L^p(\Omega) : D^\alpha f \in L^p(\Omega) \quad \text{for each non-negative multi-index } \alpha \text{ such that } |\alpha| \leq k\}. \tag{3.28}$$

For $1 \leq p < \infty$ this is a Banach space with norm

$$\|f\|_{W^{k,p}(\Omega)} = \left(\sum_{|\alpha| \leq k} \|D^\alpha f\|^p_{L^p(\Omega)} \right)^{1/p}. \tag{3.29}$$

Its seminorm $|f|_{W^{k,p}(\Omega)}$ is defined similarly, provided we sum over multi-integers α such that $|\alpha| = k$. Note that, for $k = 0$, $W^{0,p}(\Omega) = L^p(\Omega)$ instead, when $p = \infty$, the space $W^{k,\infty}(\Omega)$ is equipped with the norm

$$\|f\|_{W^{k,\infty}(\Omega)} = \max_{\alpha=1,\ldots,k} \|D^\alpha f\|_\infty.$$

Definition 3.1.8. *Let Ω be an open set of $\mathbb{R}^n$ and k be a positive integer. We call Sobolev space of order k on Ω the space formed by the totality of functions of $L^2(\Omega)$ whose (distributional) derivatives up to order k belong to $L^2(\Omega)$:*

$$H^k(\Omega) = W^{k,2}(\Omega) = \{f \in L^2(\Omega) | D^\alpha f \in L^2(\Omega) \quad \forall \alpha : |\alpha| \leq k\}. \tag{3.30}$$

For such spaces hold the inclusion of type $H^{k+1}(\Omega) \subset H^k(\Omega)$. In addition, $H^k(\Omega)$ are Hilbert spaces with respect to the scalar product

$$(f, g)_k = \sum_{|\alpha| \leq k} \int_\Omega (D^\alpha f)(D^\alpha g) d\boldsymbol{x}. \tag{3.31}$$

The space $H^1_0(\Omega)$

If Ω is bounded, the space $\mathcal{D}(\Omega)$ is not dense in $H^1(\Omega)$. We can then give the following definition:

Definition 3.1.9. *We denote by $H^1_0(\Omega)$ the closure of $\mathcal{D}(\Omega)$ in $H^1(\Omega)$.*

At this stage, we recall the Poincarè inequality: for a bounded set $\Omega \subset \mathbb{R}^n$, there exists a constant C such that:

$$\|f\|_{L^2(\Omega)} \leq C|f|_{H^1(\Omega)} \qquad \forall f \in H^1_0(\Omega). \tag{3.32}$$

Trace operators

To the end of defining the "value" of f on the boundary of Ω, we introduce a value that we will call the trace of f on $\partial\Omega$. Let f be an element of $H^1(\Omega)$.

Theorem 3.1.10. *Let Ω be a domain of $\mathbb{R}^n$ provided with a "sufficiently regular" boundary $\partial\Omega$, and let $k \geq 1$. There exists one and only one, linear and continuous application*

$$\gamma_0 : H^k(\Omega) \to L^2(\partial\Omega) \tag{3.33}$$

such that $\gamma_0 f = f|_{\partial\Omega}$, $\forall f \in H^k \cap C^0(\Omega)$; $\gamma_0 f$ is called trace of f on $\partial\Omega$. The continuity of γ_0 implies that there exists a constant $C > 0$ such that

$$\|\gamma_0 f\|_{L^2(\partial\Omega)} \leq C\|f\|_{H^k(\Omega)}. \tag{3.34}$$

The result still holds if we consider the trace operator $\gamma_\Gamma : H^k(\Omega) \to L^2(\Gamma)$ where Γ is a sufficiently regular portion of the boundary of Ω with positive measure.

The trace operator γ_Γ is not surjective on $L^2(\Gamma)$. In particular, the set of functions of $L^2(\Gamma)$ which are traces of functions of $H^1(\Omega)$ constitutes a subspace of $L^2(\Gamma)$ denoted by $H^{1/2}(\Gamma)$ and characterized by intermediate regularity properties between those of $L^2(\Gamma)$ and those of $H^1(\Gamma)$. In general, for every $k \geq 1$ there exists a unique linear and continuous application $\gamma_0 : H^k(\Omega) \to H^{k-1/2}(\Gamma)$ such that $\gamma_0 f = f|_\Gamma$ for each $f \in H^k(\Omega) \cap C^0(\Omega)$.

Note that $H^1_0(\Omega)$ is formed by the functions of $H^1(\Omega)$ having null trace on the boundary.

3.1.5 Spaces of time-dependent functions

We now consider space-time functions $f(\boldsymbol{x}, t)$, for $\boldsymbol{x} \in \Omega \subset \mathbb{R}^n$ with $n \geq 1$ and $t \in (0, \tau)$, $\tau > 0$. It is natural to introduce the following function space

$$L^q(0,\tau; W^{k,p}(\Omega)) = \Big\{ f : (0,\tau) \to W^{k,p}(\Omega) \text{ such that } f \text{ is measurable and } \int_0^\tau \|f(t)\|^q_{W^{k,p}(\Omega)} dt < \infty \Big\}, \tag{3.35}$$

where $k \geq 0$ is a non-negative integer, $1 \leq q < \infty$, $1 \leq p \leq \infty$, endowed with the norm

$$\|f(t)\|_{L^q(0,\tau,W^{k,p}(\Omega))} = \left(\int_0^\tau \|f(t)\|^q_{W^{k,p}(\Omega)} dt \right)^{1/q}. \tag{3.36}$$

For every $t \in (0, \tau)$ we have used the shorthand notation $f(t)$ to indicate the function:

$$f(t) : \Omega \to \mathbb{R}, \quad f(t)(\boldsymbol{x}) = f(\boldsymbol{x}, t) \quad \forall \boldsymbol{x} \in \Omega. \tag{3.37}$$

When dealing with time-dependent initial-boundary value problems, the following results (continuous and discrete) can be useful.

Lemma 3.1.11 (Gronwall)**.** *Let $A \in L^1(0,\tau)$ be a non-negative function, ϕ a continuous function on $[0,\tau]$.*

- *If g is non-decreasing and ϕ is such that*

$$\phi(t) \leq g(t) + \int_{t_0}^{t} A(s)\phi(s)ds \quad \forall t \in [0,\tau], \tag{3.38}$$

then

$$\phi(t) \leq g(t) \exp\left(\int_{t_0}^{t} A(s)ds\right) \quad \forall t \in [0,\tau]. \tag{3.39}$$

- *If g is a non-negative constant and ϕ a non-negative function such that*

$$\phi^2(t) \leq g + \int_{t_0}^{t} A(s)\phi(s)ds \quad \forall t \in [0,\tau], \tag{3.40}$$

then

$$\phi(t) \leq \sqrt{g} + \frac{1}{2}\int_{t_0}^{t} A(s)ds \quad \forall t \in [0,\tau]. \tag{3.41}$$

Lemma 3.1.12 (discrete Gronwall)**.** *Assume that k_n is a non-negative sequence, and that the sequence ϕ_n satisfies*

$$\phi_0 \leq g_0, \quad \phi_n \leq g_0 + \sum_{m=0}^{n-1} p_m + \sum_{m=0}^{n-1} k_m\phi_m, \quad n \leq 1. \tag{3.42}$$

If $g_0 \geq 0$ and $p_m \geq 0$ for $m \geq 0$, then

$$\phi_n \leq (g_0 + \sum_{m=0}^{n-1} p_m) \exp\,(\sum_{m=0}^{n-1} k_m), \quad n \leq 1. \tag{3.43}$$

3.2 Weak form of parabolic equations

We consider a parabolic equation of the form:

$$\frac{\partial}{\partial t}u + Lu = f, \quad \boldsymbol{x} \in \Omega,\, t > 0, \tag{3.44}$$

where Ω is a domain of $\mathbb{R}^n$, $f = f(\boldsymbol{x},t)$ is a given function, $L = L(\boldsymbol{x})$ is a generic elliptic operator acting on the unknown $u = u(\boldsymbol{x},t)$. We call the previous the strong formulation of the parabolic equation. This equation must be completed by assigning an initial condition

$$u(\boldsymbol{x},0) = u_0(\boldsymbol{x}) \qquad \boldsymbol{x} \in \Omega, \tag{3.45}$$

together with boundary conditions, such as

$$u(\boldsymbol{x},t) = g(\boldsymbol{x},t) \qquad \text{on } \Gamma_D \times [0,\tau], \tag{3.46}$$

$$\frac{\partial u(\boldsymbol{x},t)}{\partial \boldsymbol{n}} = q(\boldsymbol{x},t) \qquad \text{on } \Gamma_N \times [0,\tau], \tag{3.47}$$

where u_0, g and q are given functions instead, Γ_D and Γ_N represent the parts of Dirichlet and Neumann boundary respectively where $\partial\Omega = \Gamma_D \cup \Gamma_N$ and $\Gamma_D \cap \Gamma_N = \emptyset$.

A formulation of the problem, alternative to the strong one, is therefore necessary to reduce the order of the derivation required for the unknown solution u. We move from a second-order differential problem to a first-order one in integral form, which is called the weak formulation of the differential problem. To this end, we operate a sequence of formal transformations without worrying at this stage whether all the operations appearing in it are allowed. We start by multiplying the parabolic equation by a (so far arbitrary) test function $\tilde{u}$ and integrating on the domain Ω for each $t \in [0,\tau]$

$$\int_\Omega \frac{\partial u}{\partial t}\tilde{u}\, d\boldsymbol{X} + \int_\Omega Lu\tilde{u}\, d\boldsymbol{X} = \int_\Omega f\tilde{u}\, d\boldsymbol{X}. \tag{3.48}$$

In the particular case where $L = -\operatorname{div}(\nabla)$, we get a diffusive equation

$$\int_\Omega \frac{\partial u}{\partial t}\tilde{u}\, d\boldsymbol{X} - \int_\Omega \operatorname{div}(\nabla u)\tilde{u}\, d\boldsymbol{X} = \int_\Omega f\tilde{u}\, d\boldsymbol{X}. \tag{3.49}$$

By applying Green's formula we obtain

$$\int_\Omega \frac{\partial u}{\partial t}\tilde{u}\, d\boldsymbol{X} + \int_\Omega (\nabla u)\cdot(\nabla\tilde{u})\, d\boldsymbol{X} - \int_{\partial\Omega} \frac{\partial u}{\partial \boldsymbol{n}}\tilde{u}\, dS = \int_\Omega f\tilde{u}\, d\boldsymbol{X}, \tag{3.50}$$

that considering the boundary condition can be rewritten as

$$\int_\Omega \frac{\partial u}{\partial t}\tilde{u}\, d\boldsymbol{X} + \int_\Omega (\nabla u)\cdot(\nabla\tilde{u})\, d\boldsymbol{X} - \int_{\Gamma_N} q\tilde{u}\, dS = \int_\Omega f\tilde{u}\, d\boldsymbol{X}. \tag{3.51}$$

We define the bilinear form $a : V \times V \to \mathbb{R}$

$$a(u,\tilde{u}) = \int_\Omega (\nabla u)\cdot(\nabla\tilde{u})\, d\boldsymbol{X}, \tag{3.52}$$

and the linear and continuous functional $b : V \to \mathbb{R}$

$$b(\tilde{u}) = \int_{\Gamma_N} q\tilde{u}\, dS + \int_\Omega f\tilde{u}\, d\boldsymbol{X}. \tag{3.53}$$

The previous problem becomes:

$$\text{find } u \in V \text{ such that } \int_\Omega \frac{\partial u}{\partial t}\tilde{u}\, d\boldsymbol{X} + a(u,\tilde{u}) = b(\tilde{u}) \quad \forall \tilde{u} \in V. \tag{3.54}$$

When dealing with the time discretization of the problem we will make use of the following fundamental result for the existence and uniqueness of steady solutions.

Lemma 3.2.1 (Lax-Milgram)**.** *Let V be a Hilbert space, $a(\cdot,\cdot) : V \times V \to \mathbb{R}$ a continuous and coercive bilinear form, $b(\cdot) : V \to \mathbb{R}$ a linear and continuous functional. Then, there exists a unique solution to the problem*

$$\textit{find } u \in V : a(u,\tilde{u}) = b(\tilde{u}) \qquad \forall \tilde{u} \in V. \tag{3.55}$$

3.3 Uniqueness of the leavening evolution

In this section, we establish the uniqueness of solutions for the evolution equations concerning the temperature and CO_2 and yeast concentrations, under the assumption that a static deformation is given. To this end, we need to introduce the weak form of the parabolic PDEs.

We introduce test functions $\tilde{\theta}_\mathrm{b}$ for the heat equation and $\tilde{D}$ for the diffusion equation of carbon dioxide. The weak formulation of (2.11)–(2.14) and (2.17)–(2.18) is then

$$\begin{aligned}\int_0^\tau \int_\Omega \rho_\mathrm{b} c_\mathrm{b}\, J \frac{\partial \theta_\mathrm{b}}{\partial t} \tilde{\theta}_\mathrm{b}\, d\boldsymbol{X}\, dt + \int_0^\tau \int_\Omega (\beta_\mathrm{b}\, J\, \mathbf{C}^{-1} \nabla \theta_\mathrm{b}) \cdot \nabla \tilde{\theta}_\mathrm{b}\, d\boldsymbol{X}\, dt \\ = \int_0^\tau \int_{\Gamma_1} h_\mathrm{bo}(\theta_o - \theta_\mathrm{b}) \tilde{\theta}_\mathrm{b} |J\, \mathbf{F}^{-\top} \boldsymbol{m}|\, dS\, dt + \int_0^\tau \int_{\Gamma_3} h_\mathrm{bt}(\theta_o - \theta_\mathrm{b}) \tilde{\theta}_\mathrm{b} |J\, \mathbf{F}^{-\top} \boldsymbol{m}|\, dS\, dt\end{aligned} \tag{3.56}$$

and

$$\int_0^\tau \int_\Omega J\, \frac{\partial D}{\partial t} \tilde{D}\, d\boldsymbol{X}\, dt + \int_0^\tau \int_\Omega (\beta_{co2}\, J\, \mathbf{C}^{-1} \nabla D) \cdot (\nabla \tilde{D})\, d\boldsymbol{X}\, dt = \int_0^\tau \int_\Omega f(\theta_\mathrm{b})\, Y\, \tilde{D} J\, d\boldsymbol{X}\, dt, \tag{3.57}$$

where θ_b, $\tilde{\theta}_\mathrm{b}$ and D, $\tilde{D}$ in $\in L^2([0,\tau]; H^1(\Omega))$.

Given that the deformation is fixed over the time interval $[0,\tau]$, we can assume the boundedness condition $J\mathbf{C}^{-1} \in L^\infty([0,\tau]; L^\infty(\Omega))$. We also assume $\mathbf{A} := J\,\mathbf{C}^{-1}$ symmetric and positive-definite, so that there exists the smallest eigenvalue $c_A(\boldsymbol{X}) > 0$ and the largest one $C_A(\boldsymbol{X}) > 0$. We then obtain the following estimates for the generic functions θ and $\tilde{\theta}$

$$\begin{aligned}\int_\Omega (\beta\, J\, \mathbf{C}^{-1} \nabla\theta) \cdot \nabla\tilde{\theta}\, d\boldsymbol{X} &= \int_\Omega \beta \nabla\tilde{\theta} \cdot \mathbf{A}(\nabla\theta)\, d\boldsymbol{X} \geq \int_\Omega c_A(\boldsymbol{X}) \beta \nabla\tilde{\theta} \cdot (\nabla\theta)\, d\boldsymbol{X} \\ &= \int_\Omega \beta\, c_A(\boldsymbol{X}) (\nabla\theta \cdot \nabla\tilde{\theta})\, d\boldsymbol{X} \geq \beta \min_{\boldsymbol{X}\in\Omega} c_A(\boldsymbol{X}) \langle \nabla\theta, \nabla\tilde{\theta} \rangle_{L^2(\Omega)} \\ &\geq c \langle \nabla\theta, \nabla\tilde{\theta} \rangle_{L^2(\Omega)},\end{aligned} \tag{3.58}$$

$$\begin{aligned}\int_\Omega (\beta\, J\, \mathbf{C}^{-1} \nabla\theta) \cdot \nabla\tilde{\theta}\, d\boldsymbol{X} &\leq \int_\Omega \beta \nabla\tilde{\theta} \cdot \mathbf{A}(\nabla\theta)\, d\boldsymbol{X} \leq \int_\Omega C_A(\boldsymbol{X}) \beta \nabla\tilde{\theta} \cdot (\nabla\theta)\, d\boldsymbol{X} \\ &\leq \int_\Omega \beta\, C_A(\boldsymbol{X}) (\nabla\theta \cdot \nabla\tilde{\theta})\, d\boldsymbol{X} \leq \beta \max_{\boldsymbol{X}\in\Omega} C_A(\boldsymbol{X}) \langle \nabla\theta, \nabla\tilde{\theta} \rangle_{L^2(\Omega)} \\ &\leq C \langle \nabla\theta, \nabla\tilde{\theta} \rangle_{L^2(\Omega)}.\end{aligned} \tag{3.59}$$

In a similar way, we get the estimates

$$\int_\Omega J\,\theta\,\tilde{\theta}\,d\boldsymbol{X} \geq \min_{\boldsymbol{X}\in\Omega} J\langle\theta,\tilde{\theta}\rangle_{L^2(\Omega)} \geq J_{\min}\langle\theta,\tilde{\theta}\rangle_{L^2(\Omega)} \geq 0, \tag{3.60}$$

$$\int_\Omega J\,\theta\,\tilde{\theta}\,d\boldsymbol{X} \leq \max_{\boldsymbol{X}\in\Omega} J\langle\theta,\tilde{\theta}\rangle_{L^2(\Omega)} \leq J_{\max}\langle\theta,\tilde{\theta}\rangle_{L^2(\Omega)} \tag{3.61}$$

and, given that the boundedness of $J\mathbf{C}^{-1}$ implies the boundedness of $J\mathbf{F}^{-\top}$, we also have

$$\int_{\partial\Omega} |J\mathbf{F}^{-\top}\boldsymbol{m}|\tilde{\theta}\,dS \geq C\int_{\partial\Omega}\tilde{\theta}\,dS, \tag{3.62}$$

$$\int_{\partial\Omega} |J\mathbf{F}^{-\top}\boldsymbol{m}|\tilde{\theta}\,dS \leq c\int_{\partial\Omega}\tilde{\theta}\,dS. \tag{3.63}$$

Theorem 3.3.1. *The continuum problem given by the equations* (3.56)-(3.57) *coupled with* (2.15) *has at most one solution with* θ_b *and* D *in* $\in L^2([0,\tau];H^1(\Omega))$ *and* $Y \in L^2([0,\tau];L^2(\Omega))$*, for any choice of the finite time interval* $[0,\tau]$ *and of initial conditions in* $L^2(\Omega)$ *for each field.*

Proof. Let us consider two weak solutions of our problem $f^{(1)}(\boldsymbol{X},t)$ and $f^{(2)}(\boldsymbol{X},t)$, with equal initial conditions, represented as vectors given by

$$f^{(1)}(\boldsymbol{X},t) = \begin{pmatrix}\theta_\mathrm{b}^{(1)}(\boldsymbol{X},t)\\ Y^{(1)}(\boldsymbol{X},t)\\ D^{(1)}(\boldsymbol{X},t)\end{pmatrix}, \qquad f^{(2)}(\boldsymbol{X},t) = \begin{pmatrix}\theta_\mathrm{b}^{(2)}(\boldsymbol{X},t)\\ Y^{(2)}(\boldsymbol{X},t)\\ D^{(2)}(\boldsymbol{X},t)\end{pmatrix}. \tag{3.64}$$

We wish to estimate the L^2-norm of the difference between the two solutions computed at time t, given by

$$\begin{aligned}\|f^{(1)}(\boldsymbol{X},t) - f^{(2)}(\boldsymbol{X},t)\|^2_{L^2(\Omega)} &= \|\theta_\mathrm{b}^{(1)}(\boldsymbol{X},t) - \theta_\mathrm{b}^{(2)}(\boldsymbol{X},t)\|^2_{L^2(\Omega)}\\ &+ \|Y^{(1)}(\boldsymbol{X},t) - Y^{(2)}(\boldsymbol{X},t)\|^2_{L^2(\Omega)} + \|D^{(1)}(\boldsymbol{X},t) - D^{(2)}(\boldsymbol{X},t)\|^2_{L^2(\Omega)}.\end{aligned} \tag{3.65}$$

Let us start with the weak solutions $\theta_\mathrm{b}^{(1)}, \theta_\mathrm{b}^{(2)}$ of equation (3.56). By taking the difference of the equations satisfied by the two solutions at a given time t we easily get

$$\begin{aligned}&\int_\Omega \rho_\mathrm{b}\,c_\mathrm{b}\,J\frac{\partial}{\partial t}(\theta_\mathrm{b}^{(1)} - \theta_\mathrm{b}^{(2)})\tilde{\theta}_\mathrm{b}\,d\boldsymbol{X} + \int_\Omega (\beta_\mathrm{b}\,J\,\mathbf{C}^{-1}\nabla(\theta_\mathrm{b}^{(1)} - \theta_\mathrm{b}^{(2)}))\cdot(\nabla\tilde{\theta}_\mathrm{b})\,d\boldsymbol{X}\\ &= \int_{\Gamma_1} h_\mathrm{bo}(\theta_\mathrm{b}^{(2)} - \theta_\mathrm{b}^{(1)})\tilde{\theta}_\mathrm{b}|J\,\mathbf{F}^{-\top}\boldsymbol{m}|\,dS + \int_{\Gamma_3} h_\mathrm{bt}(\theta_\mathrm{b}^{(2)} - \theta_\mathrm{b}^{(1)})\tilde{\theta}_\mathrm{b}|J\,\mathbf{F}^{-\top}\boldsymbol{m}|\,dS.\end{aligned} \tag{3.66}$$

By choosing $\tilde{\theta}_\mathrm{b} = (\theta_\mathrm{b}^{(1)} - \theta_\mathrm{b}^{(2)})$, we obtain

$$\begin{aligned}&\int_\Omega \rho_\mathrm{b}\,c_\mathrm{b}\,J\frac{1}{2}\frac{\partial}{\partial t}(\theta_\mathrm{b}^{(1)} - \theta_\mathrm{b}^{(2)})^2\,d\boldsymbol{X} + \int_\Omega \beta_\mathrm{b}\,J\,\mathbf{C}^{-1}\nabla(\theta_\mathrm{b}^{(1)} - \theta_\mathrm{b}^{(2)})\cdot\nabla(\theta_\mathrm{b}^{(1)} - \theta_\mathrm{b}^{(2)})\,d\boldsymbol{X}\\ &= -\int_{\Gamma_1} h_\mathrm{bo}(\theta_\mathrm{b}^{(1)} - \theta_\mathrm{b}^{(2)})^2|J\,\mathbf{F}^{-\top}\boldsymbol{m}|\,dS - \int_{\Gamma_3} h_\mathrm{bt}(\theta_\mathrm{b}^{(1)} - \theta_\mathrm{b}^{(2)})^2|J\,\mathbf{F}^{-\top}\boldsymbol{m}|\,dS.\end{aligned} \tag{3.67}$$

From the previous computation, we conclude that

$$\frac{1}{2}\frac{d}{dt}\int_\Omega \rho_\mathrm{b}\, c_\mathrm{b}\, J|\theta_\mathrm{b}^{(1)} - \theta_\mathrm{b}^{(2)}|^2\, d\boldsymbol{X} \leq 0. \tag{3.68}$$

Finally, by integrating in time we obtain

$$\int_\Omega \rho_\mathrm{b}\, c_\mathrm{b}\, J|\theta_\mathrm{b}^{(1)}(\boldsymbol{X},t) - \theta_\mathrm{b}^{(2)}(\boldsymbol{X},t)|^2\, d\boldsymbol{X} \leq \int_\Omega \rho_\mathrm{b}\, c_\mathrm{b}\, J|\theta_\mathrm{b}^{(1)}(\boldsymbol{X},0) - \theta_\mathrm{b}^{(2)}(\boldsymbol{X},0)|^2\, d\boldsymbol{X}. \tag{3.69}$$

We work under the assumption that $J(\boldsymbol{X}) \geq J_{\min} > 0$. Combining this with the identity of initial conditions, we get

$$\rho_\mathrm{b} c_\mathrm{b}\, J_{\min}\|\theta_\mathrm{b}^{(1)} - \theta_\mathrm{b}^{(2)}\|^2_{L^2(\Omega)} \leq 0. \tag{3.70}$$

The previous formula tells us that $\|\theta_\mathrm{b}^{(1)} - \theta_\mathrm{b}^{(2)}\|_{L^2(\Omega)} = 0$ for each $t \in [0,\tau]$.

Given that the solution for the temperature field θ_b is unique and the function $K_\mathrm{y}(\theta_\mathrm{b})Y$ is continuous and Lipschitz in Y uniformly in t, there exists a unique solution Y for the Cauchy problem (2.15) for each fixed material point $\boldsymbol{X}$.

Finally by considering (3.57) and taking the difference between two solutions $D^{(1)}, D^{(2)}$ we get

$$\begin{aligned}\int_\Omega J\frac{\partial}{\partial t}(D^{(1)} - D^{(2)})\tilde{D}\, d\boldsymbol{X} + \int_\Omega \beta_{co2} J\mathbf{C}^{-1}\nabla(D^{(1)} - D^{(2)})\cdot(\nabla\tilde{D})\, d\boldsymbol{X} \\ = \int_\Omega J\left(f(\theta_\mathrm{b})Y - f(\theta_\mathrm{b})Y\right)\tilde{D}\, d\boldsymbol{X}.\end{aligned} \tag{3.71}$$

By choosing $\tilde{D} = D^{(1)} - D^{(2)}$ we easily obtain

$$\int_\Omega J\frac{1}{2}\frac{\partial}{\partial t}(D^{(1)} - D^{(2)})^2\, d\boldsymbol{X} + \int_\Omega (\beta_{co2}\, J\, \mathbf{C}^{-1}\nabla(D^{(1)} - D^{(2)}))\cdot(\nabla(D^{(1)} - D^{(2)})\, d\boldsymbol{X} = 0 \tag{3.72}$$

and, since $J\mathbf{C}^{-1}(\boldsymbol{X})$ is positive definite for any $\boldsymbol{X}$, we get

$$\frac{J_{\min}}{2}\frac{d}{dt}\|D^{(1)} - D^{(2)}\|^2_{L^2(\Omega)} \leq 0, \tag{3.73}$$

that, due to the initial conditions, implies $\|D^{(1)} - D^{(2)}\|^2_{L^2(\Omega)} = 0$ for any $t \in [0,\tau]$. □

3.4 Euler methods

A classical method to solve a ODE such as

$$\mathbf{M}\dot{\boldsymbol{u}}(t) + \mathbf{K}\boldsymbol{u}(t) = \boldsymbol{f}(t) \tag{3.74}$$

where $\mathbf{M}$, $\mathbf{K}$ and $\boldsymbol{f}$ are suitable matrices and a vector, is the $\theta-$method. The latter discretizes the temporal derivative by a simple difference quotient and replaces the other terms with a linear combination of the value at time t_k and of the value at time t_{k+1}, depending on the real parameter $0 \leq \theta \leq 1$

$$\mathbf{M}\frac{\boldsymbol{u}_{k+1} - \boldsymbol{u}_k}{\Delta t} + \mathbf{K}[\theta \boldsymbol{u}_{k+1} + (1-\theta)\boldsymbol{u}_k] = \theta \boldsymbol{f}_{k+1} + (1-\theta)\boldsymbol{f}_k. \tag{3.75}$$

In the previous discretization, we divided the time interval $[0, \tau]$, in which the dynamic happens, into sub-intervals of width Δt and we denote with the subscript k the quantity evaluated at time t_k.

Let us now see some cases for a particular choice of θ:

- for $\theta = 0$ we get the forward Euler (or explicit Euler) method

$$\mathbf{M}\frac{\boldsymbol{u}_{k+1} - \boldsymbol{u}_k}{\Delta t} + \mathbf{K}\boldsymbol{u}_k = \boldsymbol{f}_k; \tag{3.76}$$

- for $\theta = 1$ we get the backward Euler (or implicit Euler) method

$$\mathbf{M}\frac{\boldsymbol{u}_{k+1} - \boldsymbol{u}_k}{\Delta t} + \mathbf{K}\boldsymbol{u}_{k+1} = \boldsymbol{f}_{k+1}; \tag{3.77}$$

- for $\theta = \frac{1}{2}$ we get the Crank-Nicolson (or trapezoidal) method

$$\mathbf{M}\frac{\boldsymbol{u}_{k+1} - \boldsymbol{u}_k}{\Delta t} + \frac{1}{2}\mathbf{K}[\boldsymbol{u}_{k+1} + \boldsymbol{u}_k] = \frac{1}{2}(\boldsymbol{f}_{k+1} + \boldsymbol{f}_k). \tag{3.78}$$

Notice that $\theta = \frac{1}{2}$ is the only value for which we obtain a second-order method instead, for the other two cases we get a first-order method.

For both the Euler methods, we obtain a system of linear equations: if $\theta = 0$, the system to solve has matrix $\frac{\mathbf{M}}{\Delta t}$, if $\theta = 1$, it has matrix $\frac{\mathbf{M}}{\Delta t} + \mathbf{K}$.

In the $\theta = 0$ case, if we make $\mathbf{M}$ diagonal, we decouple the system. However, this scheme is not unconditionally stable and we have to consider a necessary condition for the stability of an explicit numerical scheme, given by the CFL condition (from Courant, Friedrichs and Lewy) which depends on the discretization in the space variable.

In the case $\theta > 0$, the system will have the form $\mathbf{A}\boldsymbol{u}_{k+1} = \boldsymbol{g}$, where $\boldsymbol{g}$ is a source term and $\mathbf{A} = \frac{\mathbf{M}}{\Delta t} + \mathbf{K}$. Under suitable conditions on the matrices $\mathbf{M}$ and $\mathbf{K}$, we can use, for instance, the Cholesky factorization, $\mathbf{A} = \mathbf{H}\mathbf{H}^\top$, with $\mathbf{H}$ being lower triangular, to solve two triangular systems associated to the original problem.

3.5 Time discretization of bread leavening model

In this section, we first present the time discretization of the leavening problem, which will be at the basis of our numerical scheme.

Let us consider the time interval $[0, \tau]$, we define a time step size Δt and a number of steps n_{steps} such that $\tau = \Delta t\, n_{\text{steps}}$. We use a semi-implicit Euler method to deal with the coupling among the different equations. We denote with the subscript n quantities related to the iteration at the current time.

Let us consider (3.56) that discretized in time reads on each sub-interval

$$\begin{aligned}&\int_\Omega \rho_{\mathrm{b}} c_{\mathrm{b}} J \frac{\theta_{\mathrm{b}_n} - \theta_{\mathrm{b}_{n-1}}}{\Delta t} \tilde{\theta}_{\mathrm{b}_n}\, d\boldsymbol{X} + \int_\Omega (\beta_{\mathrm{b}}\, J\, \mathbf{C}^{-1} \nabla \theta_{\mathrm{b}_n}) \cdot (\nabla \tilde{\theta}_{\mathrm{b}_n})\, d\boldsymbol{X} \\ &= \int_{\Gamma_1} h_{\mathrm{bo}} (\theta_o - \theta_{\mathrm{b}_n}) \tilde{\theta}_{\mathrm{b}_n} |J\, \mathbf{F}^{-\top} \boldsymbol{m}|\, dS + \int_{\Gamma_3} h_{\mathrm{bt}} (\theta_o - \theta_{\mathrm{b}_n}) \tilde{\theta}_{\mathrm{b}_n} |J\, \mathbf{F}^{-\top} \boldsymbol{m}|\, dS\end{aligned} \tag{3.79}$$

where $\theta_{\mathrm{b}_{n-1}}$ is known. Let us now define the bilinear form $a_{\mathrm{b}} : H^1(\Omega) \times H^1(\Omega) \to \mathbb{R}$ and the linear form $b_{\mathrm{b}} : H^1(\Omega) \to \mathbb{R}$ associated with the previous equation:

$$\begin{aligned}a_{\mathrm{b}}(\theta_{\mathrm{b}_n}, \tilde{\theta}_{\mathrm{b}_n}) = &\int_\Omega \rho_{\mathrm{b}} c_{\mathrm{b}}\, J \frac{\theta_{\mathrm{b}_n}}{\Delta t} \tilde{\theta}_{\mathrm{b}_n}\, d\boldsymbol{X} + \int_\Omega (\beta_{\mathrm{b}}\, J\, \mathbf{C}^{-1} \nabla \theta_{\mathrm{b}_n}) \cdot (\nabla \tilde{\theta}_{\mathrm{b}_n})\, d\boldsymbol{X} \\ &+ \int_{\Gamma_1} h_{\mathrm{bo}} \theta_{\mathrm{b}_n} \tilde{\theta}_{\mathrm{b}_n} |J\, \mathbf{F}^{-\top} \boldsymbol{m}|\, dS + \int_{\Gamma_3} h_{\mathrm{bt}} \theta_{\mathrm{b}_n} \tilde{\theta}_{\mathrm{b}_n} |J\, \mathbf{F}^{-\top} \boldsymbol{m}|\, dS,\end{aligned} \tag{3.80}$$

$$b_{\mathrm{b}}(\tilde{\theta}_{\mathrm{b}_n}) = \int_\Omega \rho_{\mathrm{b}} c_{\mathrm{b}}\, J \frac{\theta_{\mathrm{b}_{n-1}}}{\Delta t} \tilde{\theta}_{\mathrm{b}_n}\, d\boldsymbol{X} + \int_{\Gamma_1} h_{\mathrm{bo}} \theta_o \tilde{\theta}_{\mathrm{b}_n} |J\, \mathbf{F}^{-\top} \boldsymbol{m}|\, dS + \int_{\Gamma_3} h_{\mathrm{bt}} \theta_o \tilde{\theta}_{\mathrm{b}_n} |J\, \mathbf{F}^{-\top} \boldsymbol{m}|\, dS. \tag{3.81}$$

These forms translate the problem (3.79) into finding $\theta_{\mathrm{b}_n} \in H^1(\Omega)$ such that

$$a_{\mathrm{b}}(\theta_{\mathrm{b}_n}, \tilde{\theta}_{\mathrm{b}_n}) = b_{\mathrm{b}}(\tilde{\theta}_{\mathrm{b}_n}) \quad \forall\, \tilde{\theta}_{\mathrm{b}_n} \in H^1(\Omega). \tag{3.82}$$

Concerning the yeast production, by using the discretized solution of the temperature field θ_{b_n}, we can write the discrete version of (2.15) as

$$Y_n = e^{K_{\mathrm{y}}(\theta_{\mathrm{b}_{n-1}})\Delta t} Y_{n-1}. \tag{3.83}$$

Finally, the diffusion equation of carbon dioxide (3.57) becomes

$$\int_\Omega J\, \frac{D_n - D_{n-1}}{\Delta t} \tilde{D}_n\, d\boldsymbol{X} + \int_\Omega (\beta_{co2}\, J\, \mathbf{C}^{-1} \nabla D_n) \cdot (\nabla \tilde{D}_n)\, d\boldsymbol{X} = \int_\Omega J f(\theta_{\mathrm{b}_{n-1}})\, Y_{n-1}\, \tilde{D}_n\, d\boldsymbol{X}, \tag{3.84}$$

where $\theta_{\mathrm{b}_{n-1}}$ and Y_{n-1} are supposed to be known. The bilinear and linear forms, respectively $a_{\mathrm{D}} : H^1(\Omega) \times H^1(\Omega) \to \mathbb{R}$ and $b_{\mathrm{D}} : H^1(\Omega) \to \mathbb{R}$ read

$$a_{\mathrm{D}}(D_n, \tilde{D}_n) = \int_\Omega J\, \frac{D_n}{\Delta t} \tilde{D}_n\, d\boldsymbol{X} + \int_\Omega (\beta_{co2}\, J\, \mathbf{C}^{-1} \nabla D_n) \cdot (\nabla \tilde{D}_n)\, d\boldsymbol{X}, \tag{3.85}$$

$$b_{\mathrm{D}}(\tilde{D}_n) = \int_\Omega J\, \frac{D_{n-1}}{\Delta t} \tilde{D}_n\, d\boldsymbol{X} + \int_\Omega J f(\theta_{\mathrm{b}_{n-1}})\, Y_{n-1}\, \tilde{D}_n\, d\boldsymbol{X}. \tag{3.86}$$

These translate the problem (3.84) into finding $D_n \in H^1(\Omega)$ such that

$$a_{\mathrm{D}}(D_n, \tilde{D}_n) = b_{\mathrm{D}}(\tilde{D}_n) \quad \forall\, \tilde{D}_n \in H^1(\Omega). \tag{3.87}$$

Theorem 3.5.1. *There exists a unique weak solution f_n with components $\theta_{\mathrm{b}_n} \in H^1(\Omega)$, $Y_n \in L^2(\Omega)$, and $D_n \in H^1(\Omega)$ associated with the n-th time step of the problem corresponding to the equations* (3.82), (3.83), (3.87).

Proof. First of all, notice that the yeast production equation (3.83) is algebraic and its solution exists and is unique for every $\boldsymbol{X}$. Instead, to prove the existence and uniqueness of the solutions of (3.82) and (3.87) we have to check the conditions of continuity and coercivity of the bilinear forms to apply the Lax–Milgram Lemma 3.2.1. Let us start by verifying the conditions on (3.80). In what follows c_i with $i \in \mathbb{N}$ denotes a positive constant. The following estimates hold :

$$\begin{aligned}
|a_{\mathrm{b}}(\theta_{\mathrm{b}_n}, \tilde{\theta}_{\mathrm{b}_n})| &\leq \int_\Omega |\rho_{\mathrm{b}} c_{\mathrm{b}} J \frac{\theta_{\mathrm{b}_n}}{\Delta t} \tilde{\theta}_{\mathrm{b}_n}|\, d\boldsymbol{X} + \int_\Omega |(\beta_{\mathrm{b}} J \mathbf{C}^{-1} \nabla \theta_{\mathrm{b}_n}) \cdot \nabla \tilde{\theta}_{\mathrm{b}_n}|\, d\boldsymbol{X} \\
&\quad + \int_{\Gamma_1} |h_{\mathrm{bo}} \theta_{\mathrm{b}_n} \tilde{\theta}_{\mathrm{b}_n} | J\, \mathbf{F}^{-\top} \boldsymbol{m}|\, |\, dS + \int_{\Gamma_3} |h_{\mathrm{bt}} \theta_{\mathrm{b}_n} \tilde{\theta}_{\mathrm{b}_n} | J\, \mathbf{F}^{-\top} \boldsymbol{m}|\, |\, dS \\
&\leq c_1 |\langle \theta_{\mathrm{b}_n}, \tilde{\theta}_{\mathrm{b}_n} \rangle|_{H^1(\Omega)} + c_2 |\langle \theta_{\mathrm{b}_n}, \tilde{\theta}_{\mathrm{b}_n} \rangle|_{H^1(\Omega)} + c_3 |\langle \theta_{\mathrm{b}_n}, \tilde{\theta}_{\mathrm{b}_n} \rangle|_{H^1(\Omega)} \\
&\leq c_4 \|\theta_{\mathrm{b}_n}\|_{H^1(\Omega)}\, \|\tilde{\theta}_{\mathrm{b}_n}\|_{H^1(\Omega)}
\end{aligned} \tag{3.88}$$

and

$$\begin{aligned}
a_{\mathrm{b}}(\theta_{\mathrm{b}_n}, \theta_{\mathrm{b}_n}) &= \int_\Omega \rho_{\mathrm{b}} c_{\mathrm{b}}\, J \frac{\theta_{\mathrm{b}_n}^2}{\Delta t}\, d\boldsymbol{X} + \int_\Omega (\beta_{\mathrm{b}} J \mathbf{C}^{-1} \nabla \theta_{\mathrm{b}_n}) \cdot \nabla \theta_{\mathrm{b}_n}\, d\boldsymbol{X} \\
&\quad + \int_{\Gamma_1} h_{\mathrm{bo}} \theta_{\mathrm{b}_n}^2 |J\, \mathbf{F}^{-\top} \boldsymbol{m}|\, dS + \int_{\Gamma_3} h_{\mathrm{bt}} \theta_{\mathrm{b}_n}^2 |J\, \mathbf{F}^{-\top} \boldsymbol{m}|\, dS \\
&\geq c_5 \|\theta_{\mathrm{b}_n}\|^2_{L^2(\Omega)} + c_6 \|\nabla \theta_{\mathrm{b}_n}\|^2_{L^2(\Omega)} \\
&\geq c_7 \|\theta_{\mathrm{b}_n}\|^2_{H^1(\Omega)}.
\end{aligned} \tag{3.89}$$

Let us now consider the diffusion of carbon dioxide (3.84) and verify the continuity and coercivity of (3.85) by

$$\begin{aligned}
|a_{\mathrm{D}}(D_n, \tilde{D}_n)| &\leq \int_\Omega |J \frac{D_n}{\Delta t} \tilde{D}_n|\, d\boldsymbol{X} + \int_\Omega |(\beta_{co2}\, J\, \mathbf{C}^{-1} \nabla D_n) \cdot (\nabla \tilde{D}_n)|\, d\boldsymbol{X} \\
&\leq c_8 |\langle D_n, \tilde{D}_n \rangle|_{H^1(\Omega)} \leq c_9 \|D_n\|_{H^1(\Omega)}\, \|\tilde{D}_n\|_{H^1(\Omega)}
\end{aligned} \tag{3.90}$$

and

$$\begin{aligned}
a_{\mathrm{D}}(D_n, D_n) &= \int_\Omega J\, \frac{D_n^2}{\Delta t}\, d\boldsymbol{X} + \int_\Omega (\beta_{co2}\, J\, \mathbf{C}^{-1} \nabla D_n) \cdot (\nabla D_n)\, d\boldsymbol{X} \\
&\geq c_{10} \|D_n\|^2_{L^2(\Omega)} + c_{11} \|\nabla D_n\|^2_{L^2(\Omega)} \geq c_{12} \|D_n\|^2_{H^1(\Omega)}.
\end{aligned} \tag{3.91}$$

With these conditions, we can apply the Lax–Milgram Lemma 3.2.1 and obtain the existence and uniqueness of solutions for problems (3.82), (3.83), (3.87). □

3.6 Convergence of the time discretization

We can now build a sequence of functions $\{f^{(k)}(\boldsymbol{X},t)\}_k$ which are piece-wise constant and are on each time interval $[t_{n-1},t_n)$ solutions of the single-time-step equations (3.82), (3.83), (3.87). We set $\Delta t = t_n - t_{n-1} = \tau 2^{-k}$ and define, with

$$f_n^{(k)}(\boldsymbol{X}) = \begin{pmatrix} \theta_{\mathrm{b}_n}^{(k)}(\boldsymbol{X}) \\ Y_n^{(k)}(\boldsymbol{X}) \\ D_n^{(k)}(\boldsymbol{X}) \end{pmatrix}, \tag{3.92}$$

the global discrete-time solution

$$f^{(k)}(\boldsymbol{X},t) := \sum_{n=0}^{N(k)} f_n^{(k)}(\boldsymbol{X})\chi_{[t_n,t_n+\tau/2k[}(t), \tag{3.93}$$

where $N(k) = 2^k$ is the number of sub-intervals of $[0,\tau]$. We wish to study the convergence of $\{f^{(k)}(\boldsymbol{X},t)\}_k$ in the Hilbert space $L^2([0,\tau];L^2(\Omega))$.

Theorem 3.6.1. *The sequence $\{f^{(k)}(\boldsymbol{X},t)\}_k$ is bounded and hence admits a weakly convergent sub-sequence in $L^2([0,\tau];L^2(\Omega))$.*

Proof. We now compute the appropriate norm for each variable of the discrete problem.

Heat equation in the bread

We first consider estimates useful for equation (3.79). We have

$$\begin{aligned}
\frac{\rho_{\mathrm{b}}c_{\mathrm{b}}}{2\Delta t}\|\sqrt{J}\theta_{\mathrm{b}_n}\|^2_{L^2(\Omega)} + \tilde{\beta}\|\sqrt{J}\nabla\theta_{\mathrm{b}_n}\|^2_{L^2(\Omega)} + \bar{h}\|J\theta_{\mathrm{b}_n}\|^2_{L^2(\partial\Omega)} &\leq \frac{\rho_{\mathrm{b}}c_{\mathrm{b}}}{2\Delta t}\|\sqrt{J}\theta_{\mathrm{b}_{n-1}}\|^2_{L^2(\Omega)} \\
&\quad + \tilde{h}\|J\theta_{\mathrm{o}}\theta_{\mathrm{b}_n}\|_{L^1(\partial\Omega)}, \\
\frac{\rho_{\mathrm{b}}c_{\mathrm{b}}}{2\Delta t}\|\sqrt{J}\theta_{\mathrm{b}_n}\|^2_{L^2(\Omega)} + \tilde{\beta}\|\sqrt{J}\nabla\theta_{\mathrm{b}_n}\|^2_{L^2(\Omega)} + \bar{h}\|J\theta_{\mathrm{b}_n}\|^2_{L^2(\partial\Omega)} &\leq \frac{\rho_{\mathrm{b}}c_{\mathrm{b}}}{2\Delta t}\|\sqrt{J}\theta_{\mathrm{b}_{n-1}}\|^2_{L^2(\Omega)} \\
&\quad + \tilde{h}\|\theta_{\mathrm{o}}\|_{L^2(\partial\Omega)}\|J\theta_{\mathrm{b}_n}\|_{L^2(\partial\Omega)} \qquad (3.94) \\
\frac{\rho_{\mathrm{b}}c_{\mathrm{b}}}{2\Delta t}\|\sqrt{J}\theta_{\mathrm{b}_n}\|^2_{L^2(\Omega)} + \tilde{\beta}\|\sqrt{J}\nabla\theta_{\mathrm{b}_n}\|^2_{L^2(\Omega)} + \bar{h}\|J\theta_{\mathrm{b}_n}\|^2_{L^2(\partial\Omega)} &\leq \frac{\rho_{\mathrm{b}}c_{\mathrm{b}}}{2\Delta t}\|\sqrt{J}\theta_{\mathrm{b}_{n-1}}\|^2_{L^2(\Omega)} \\
&\quad + \tilde{h}\left(\frac{1}{2\varepsilon}\|\theta_{\mathrm{o}}\|^2_{L^2(\partial\Omega)} + \frac{\varepsilon}{2}\|J\theta_{\mathrm{b}_n}\|^2_{L^2(\partial\Omega)}\right) \qquad (3.95)
\end{aligned}$$

If we chose $\varepsilon = \frac{2\bar{h}}{\tilde{h}}$ we get

$$\frac{\rho_{\mathrm{b}}c_{\mathrm{b}}}{2\Delta t}\|\sqrt{J}\theta_{\mathrm{b}_n}\|^2_{L^2(\Omega)} + \tilde{\beta}\|\sqrt{J}\nabla\theta_{\mathrm{b}_n}\|^2_{L^2(\Omega)} \leq \frac{\rho_{\mathrm{b}}c_{\mathrm{b}}}{2\Delta t}\|\sqrt{J}\theta_{\mathrm{b}_{n-1}}\|^2_{L^2(\Omega)} + \tilde{h}\frac{1}{2\varepsilon}\|\theta_{\mathrm{o}}\|^2_{L^2(\partial\Omega)} \tag{3.96}$$

$$\frac{\rho_{\mathrm{b}}c_{\mathrm{b}}}{2\Delta t}\|\sqrt{J}\theta_{\mathrm{b}_n}\|^2_{L^2(\Omega)} \leq \frac{\rho_{\mathrm{b}}c_{\mathrm{b}}}{2\Delta t}\|\sqrt{J}\theta_{\mathrm{b}_{n-1}}\|^2_{L^2(\Omega)} + \frac{\tilde{h}}{2\varepsilon}\|\theta_{\mathrm{o}}\|^2_{L^2(\partial\Omega)}. \tag{3.97}$$

Thus, by summing over s we obtain:

$$\Delta t \sum_{s=1}^{n} \|\theta_{\mathrm{b}_s}^{(k)}\|_{L^2(\Omega)}^2 \leq M_{\mathrm{b}_1}\tau \|\theta_{\mathrm{b}_0}\|_{L^2(\Omega)}^2 + M_{\mathrm{b}_2}\tau^2 \|\theta_{\mathrm{o}}\|_{L^2(\partial\Omega)}^2 \tag{3.98}$$

where $\tau = N(k)\,\Delta t$.

Production of yeast

We then consider estimates useful for equation (3.83). Starting from

$$\begin{aligned} \|Y_n\|_{L^2(\Omega)}^2 &= \|e^{K_{\mathrm{y}}(\theta_{\mathrm{b}_{n-1}})\Delta t}\, Y_{n-1}\|_{L^2(\Omega)}^2 \\ &\leq \|e^{\max K_{\mathrm{y}}\Delta t} Y_{n-1}\|_{L^2(\Omega)}^2 \\ &= |e^{\max K_{\mathrm{y}}\Delta t}|^2 \|Y_{n-1}\|_{L^2(\Omega)}^2 \\ &\leq |e^{\max K_{\mathrm{y}}\Delta t}|^{2n} \|Y_0\|_{L^2(\Omega)}^2 \\ &\leq |e^{c\tau}|\, \|Y_0\|_{L^2(\Omega)}^2, \end{aligned} \tag{3.99}$$

by computing the sum over s we obtain

$$\Delta t \sum_{s=1}^{n} \|Y_s^{(k)}\|_{L^2(\Omega)}^2 \leq \Delta t \sum_{s=1}^{n} |e^{c\tau}|\, \|Y_0\|_{L^2(\Omega)}^2 \leq \tau\, c_Y\, \|Y_0\|_{L^2(\Omega)}^2. \tag{3.100}$$

Diffusion and production of carbon dioxide

We finally consider estimates useful for equation (3.84), which are

$$\begin{aligned} \frac{1}{\Delta t}\|\sqrt{J}D_n\|_{L^2(\Omega)}^2 + \tilde{\beta}\|\sqrt{J}\nabla D_n\|_{L^2(\Omega)}^2 &\leq \frac{1}{2\Delta t}\|\sqrt{J}D_n\|_{L^2(\Omega)}^2 + \frac{1}{2\Delta t}\|\sqrt{J}D_{n-1}\|_{L^2(\Omega)}^2 \\ &\quad + \|f(\theta_{\mathrm{b}_{n-1}})\, Y_{n-1}\, D_n J\|_{L^1(\Omega)}, \end{aligned} \tag{3.101}$$

$$\frac{1}{2\Delta t}\|\sqrt{J}D_n\|_{L^2(\Omega)}^2 + \tilde{\beta}\|\sqrt{J}\nabla D_n\|_{L^2(\Omega)}^2 \leq \frac{1}{2\Delta t}\|\sqrt{J}D_{n-1}\|_{L^2(\Omega)}^2 + \|f(\theta_{\mathrm{b}_{n-1}})\, Y_{n-1}\, D_n J\|_{L^1(\Omega)}, \tag{3.102}$$

$$\frac{1}{2\Delta t}\|\sqrt{J}D_n\|_{L^2(\Omega)}^2 \leq \frac{1}{2\Delta t}\|\sqrt{J}D_{n-1}\|_{L^2(\Omega)}^2 + \|f(\theta_{\mathrm{b}_{n-1}})\, Y_{n-1}\, D_n J\|_{L^1(\Omega)}, \tag{3.103}$$

$$\frac{1}{2\Delta t}\|\sqrt{J}D_n\|_{L^2(\Omega)}^2 \leq \frac{1}{2\Delta t}\|\sqrt{J}D_{n-1}\|_{L^2(\Omega)}^2 + \|\sqrt{J}f(\theta_{\mathrm{b}_{n-1}})\, Y_{n-1}\|_{L^2(\Omega)}\|\sqrt{J}D_n\|_{L^2(\Omega)}, \tag{3.104}$$

where the last inequality holds thanks to Holder's inequality. By summing over s we obtain

$$\sum_{s=1}^{n}\frac{1}{2\Delta t}\|\sqrt{J}D_s\|^2_{L^2(\Omega)} \leq \frac{1}{2\Delta t}\sum_{s=1}^{n}\|\sqrt{J}D_{s-1}\|^2_{L^2(\Omega)} + \sum_{s=1}^{n}\|\sqrt{J}f(\theta_{\mathrm{b}_{s-1}})\,Y_{s-1}\|_{L^2(\Omega)}\|\sqrt{J}D_s\|_{L^2(\Omega)}, \tag{3.105}$$

$$\frac{1}{2\Delta t}\|\sqrt{J}D_n\|^2_{L^2(\Omega)} \leq \frac{1}{2\Delta t}\|\sqrt{J}D_0\|^2_{L^2(\Omega)} + \sum_{s=1}^{n}\|\sqrt{J}f(\theta_{\mathrm{b}_{s-1}})\,Y_{s-1}\|_{L^2(\Omega)}\|\sqrt{J}D_s\|_{L^2(\Omega)} \tag{3.106}$$

$$\frac{1}{2\Delta t}\|\sqrt{J}D_n\|^2_{L^2(\Omega)} \leq \frac{1}{2\Delta t}\|\sqrt{J}D_0\|^2_{L^2(\Omega)} + \sum_{s=1}^{n}\frac{1}{2}\|\sqrt{J}f(\theta_{\mathrm{b}_{s-1}})\,Y_{s-1}\|^2_{L^2(\Omega)} + \sum_{s=1}^{n}\frac{1}{2}\|\sqrt{J}D_s\|^2_{L^2(\Omega)}. \tag{3.107}$$

Now, Gronwall's Lemma 3.1.12 assures us that

$$\|\sqrt{J}D_n\|^2_{L^2(\Omega)} \leq C(t_n)\Big(\|\sqrt{J}D_0\|^2_{L^2(\Omega)} + \Delta t\sum_{s=1}^{n}\|\sqrt{J}f(\theta_{\mathrm{b}_{s-1}})\,Y_{s-1}\|^2_{L^2(\Omega)}\Big), \tag{3.108}$$

$$\Delta t\sum_{s=1}^{n}\|D_s^{(k)}\|^2_{L^2(\Omega)} \leq M_{D_1}\tau\|D_0\|^2_{L^2(\Omega)} + M_{D_2}\tau^2\|Y_0\|^2_{L^2(\Omega)}. \tag{3.109}$$

We compute now the following estimate for $f_n^{(k)}$ when $\Delta t = \tau 2^{-k}$

$$\begin{aligned}\Delta t\sum_{n=0}^{N(k)}\|f_n^{(k)}\|^2_{L^2(\Omega)} &= \Delta t\sum_{n=0}^{N(k)}\Big[\|\theta_{\mathrm{b}_n}^{(k)}\|^2_{L^2(\Omega)} + \|D_n^{(k)}\|^2_{L^2(\Omega)} + \|Y_n^{(k)}\|^2_{L^2(\Omega)}\Big]\\ &\leq M_{\mathrm{b}_1}\tau\|\theta_{\mathrm{b}_0}\|^2_{L^2(\Omega)} + M_{\mathrm{b}_2}\tau^2\,\|\theta_{\mathrm{o}}\|^2_{L^2(\partial\Omega)} + M_{D_1}\tau\|D_0\|^2_{L^2(\Omega)}\\ &\quad + M_{D_2}\tau^2\|Y_0\|_{L^2(\Omega)} + \tau\,c_Y\,\|Y_0\|^2_{L^2(\Omega)} =: c_1.\end{aligned} \tag{3.110}$$

This estimate is uniform in k and leads to

$$\|f^{(k)}(\boldsymbol{X},t)\|^2_{L^2([0,\tau];L^2(\Omega))} = \int_0^\tau\sum_{n=0}^{N(k)}\|f_n^{(k)}(\boldsymbol{X})\|^2_{L^2(\Omega)}\;\chi_{[t_n,t_n+\tau/2^k[}(t)\,dt = \sum_{n=0}^{N(k)}\Delta t\|f_n^{(k)}(\boldsymbol{X})\|^2_{L^2(\Omega)} \leq c_1. \tag{3.111}$$

This proves the boundedness which implies the weak convergence, up to the extraction of a sub-sequence, thanks to the reflexivity of $L^2([0,\tau];L^2(\Omega))$. □

3.7 Variational problem: system of weak forms

For the sake of clarity, we now recap the variational formulation of the problem made of the weak forms, discretized in time of each equation which composes the leavening model.

$$\int_\Omega \rho_{\mathrm{b}} c_{\mathrm{b}} J \frac{\theta_{\mathrm{b}_n} - \theta_{\mathrm{b}_{n-1}}}{\Delta t} \tilde{\theta}_{\mathrm{b}_n}\, d\boldsymbol{X} + \int_\Omega (\beta_{\mathrm{b}}\, J\, \mathbf{C}^{-1} \nabla \theta_{\mathrm{b}_n}) \cdot (\nabla \tilde{\theta}_{\mathrm{b}_n})\, d\boldsymbol{X}$$
$$= \int_{\Gamma_1} h_{\mathrm{bo}} (\theta_o - \theta_{\mathrm{b}_n}) \tilde{\theta}_{\mathrm{b}_n} |J\, \mathbf{F}^{-\top} \boldsymbol{m}|\, dS + \int_{\Gamma_3} h_{\mathrm{bt}} (\theta_o - \theta_{\mathrm{b}_n}) \tilde{\theta}_{\mathrm{b}_n} |J\, \mathbf{F}^{-\top} \boldsymbol{m}|\, dS,$$

$$Y_n = e^{K_{\mathrm{y}}(\theta_{\mathrm{b}_{n-1}})\Delta t} Y_{n-1},$$

$$\int_\Omega J\, \frac{D_n - D_{n-1}}{\Delta t} \tilde{D}_n\, d\boldsymbol{X} + \int_\Omega (\beta_{co2}\, J\, \mathbf{C}^{-1} \nabla D_n) \cdot (\nabla \tilde{D}_n)\, d\boldsymbol{X} = \int_\Omega J f(\theta_{\mathrm{b}_{n-1}})\, Y_{n-1}\, \tilde{D}_n\, d\boldsymbol{X}.$$

Chapter 4

Simulation of the bread leavening process

In this chapter, we recall the Finite Element Method (FEM) and apply it to discretize the leavening model in the spatial variable. The resulting numerical model is then implemented and used to perform some experiments to underline energy consumption.

4.1 The Galerkin finite element method

The goal of this section is to present the Galerkin method for parabolic boundary-value problems. For further details see [34].

As seen in the previous chapter, the weak formulation of a generic parabolic problem set on a domain Ω, can be written in the following way

$$\text{find } u \in V \text{ such that } \int_\Omega \frac{\partial u}{\partial t}\tilde{u}\, d\boldsymbol{X} + a(u,\tilde{u}) = b(\tilde{u}) \quad \forall \tilde{u} \in V, \tag{4.1}$$

where V is an appropriate Hilbert space, subspace of $H^1(\Omega), a(\cdot,\cdot)$ is a continuous and coercive bilinear form from $V \times V$ in $\mathbb{R}$, $b(\cdot)$ is a continuous linear functional from V in $\mathbb{R}$.

Let V_h be a family of spaces that depends on a positive parameter h, such that $V_h \subset V, \dim V_h = N_h < \infty, \forall h > 0$. The approximate problem takes the form

$$\text{find } u_h \in V \text{ such that } \int_\Omega \frac{\partial u_h}{\partial t}\tilde{u}_h\, d\boldsymbol{X} + a(u_h,\tilde{u}_h) = b(\tilde{u}_h) \quad \forall \tilde{u}_h \in V_h. \tag{4.2}$$

and is called the Galerkin problem.

Denoting with $\{\phi_j\}_{j=1,\dots,N_h}$ a basis of V_h, it suffices that the previous equation be verified for each function of the basis, as all the functions in the space V_h are a linear combination of the ϕ_j. We will then require that

$$\int_\Omega \frac{\partial u_h}{\partial t}\phi_i\, d\boldsymbol{X} + a(u_h,\phi_i) = b(\phi_i) \quad i = 1,\dots,N_h. \tag{4.3}$$

Obviously, since $u_h \in V_h$,

$$u_h(\boldsymbol{x},t) = \sum_{j=1}^{N_h} u_j(t)\phi_j(\boldsymbol{x}), \tag{4.4}$$

where the u_j are unknown coefficients for $j = 1,\dots,N_h$. Thus we want to solve

$$\sum_{j=1}^{N_h} \dot{u}_j \underbrace{\int_\Omega \phi_j\phi_i\, d\boldsymbol{X}}_{m_{ij}} + \sum_{j=1}^{N_h} u_j \underbrace{a(\phi_j,\phi_i)}_{a_{ij}} = \underbrace{b(\phi_i)}_{f_i(t)} \quad i = 1,\dots,N_h, \tag{4.5}$$

where we have denoted with $\dot{u}_j(t)$ the derivatives of the function $u_j(t)$ with respect to time.

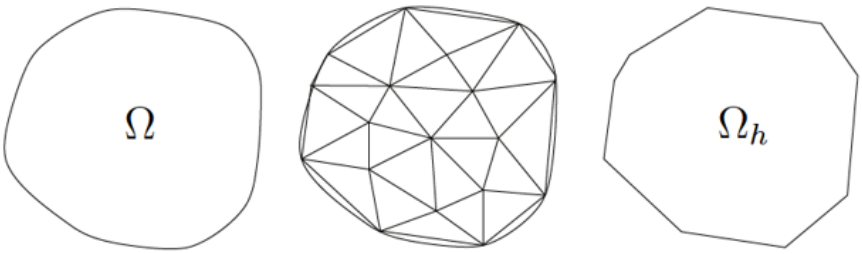

Figure 4.1: Triangulation of a non-polygonal domain.

If we define the vector of unknowns $\boldsymbol{u} = (u_1(t), u_2(t), \ldots, u_{N_h}(t))^\top$, the mass matrix $\mathbf{M} = (m_{ij})_{i,j}$, the stiffness matrix $\mathbf{A} = (a_{ij})_{i,j}$ and the right-hand side vector $\boldsymbol{f} = (f_1(t), f_2(t), \ldots, f_{N_h}(t))^\top$, the system can be rewritten in matrix form as

$$\dot{\mathbf{M}}\boldsymbol{u}(t) + \mathbf{A}\boldsymbol{u}(t) = \boldsymbol{f}(t). \tag{4.6}$$

Now, we apply the finite element method to the case of boundary-value problems in multi-dimensional regions. Let us consider a domain $\Omega \subset \mathbb{R}^2$ with polygonal shape and meshes T_h which represent their cover with non-overlapping triangles (triangulation). The domain discretized with these triangles can be written as the internal part of the union of the triangles (Figure. 4.1)

$$\Omega_h = \text{int}\left(\bigcup_{K \in T_h} K\right) \tag{4.7}$$

We set $h_K = diam(K)$, for each $K \in T_h$, that is the diameter of element K and we define $h = \max_{K \in T_h} h_K$.

Let us denote by $\mathbb{P}_r$ the space of polynomials of global degree less than or equal to r as

$$\mathbb{P}_r = \left\{ p(x_1, x_2) = \sum_{i,j \geq 0, i+j \leq r} a_{ij} x_1^i x_2^j, \quad \text{with } a_{ij} \in \mathbb{R} \right\}. \tag{4.8}$$

Notice that the spaces $\mathbb{P}_r$ have dimension $(r+1)(r+2)/2$. Thus the space of finite elements

$$X_h^r = \{\boldsymbol{v}_h \in \mathcal{C}^0(\bar{\Omega}) : \boldsymbol{v}_h | K \in \mathbb{P}_r, \, \forall K \in T_h\}, \qquad r = 1, 2, \ldots \tag{4.9}$$

is the space of globally continuous functions that are polynomials of degree r on the single triangles (elements) of the triangulation T_h. We must now choose a basis $\{\phi_i\}$ for the X_h^r space. It is convenient, that the support of the generic basis function ϕ_i have non-empty intersection only with the support of a negligible number of other functions of the basis. In such way, many elements of the stiffness matrix will be null. It is also convenient that the basis be Lagrangian: in that case, the coefficients of the expansion of a generic function $\boldsymbol{v}_h \in X_h^r$ in the basis itself will be the values taken by $\boldsymbol{v}_h$ at carefully chosen points, which we call nodes and which generally form a superset of the vertices of T_h.

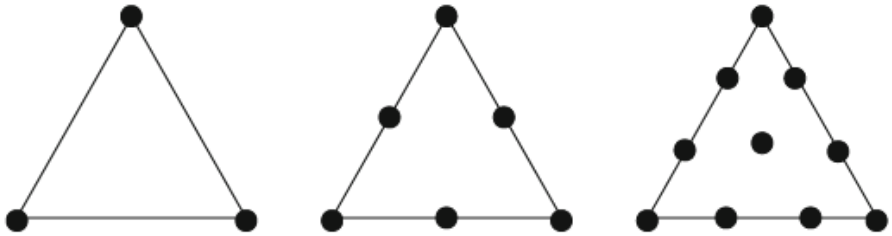

Figure 4.2: The nodes for linear ($r = 1$), quadratic ($r = 2$), and cubic ($r = 3$) polynomials on a triangle.

The nodes for linear ($r = 1$), quadratic ($r = 2$), and cubic ($r = 3$) polynomials on a triangle are shown in Figure 4.2.

4.2 Simulations and energetic analysis

For the spatial discretization we use the finite element method on an unstructured mesh, made of tetrahedra with maximum diameter h (see Figure 4.3). We define the spaces V_h, in which we approximate the vectorial placement $\boldsymbol{u}$ with Lagrangian elements P1, and W_h, where the temperature of the dough θ_b, the yeasts concentration Y and the CO_2 concentration D are approximated with P1 elements as well. Notice that, working with material coordinates allows us to have a fixed domain independent of the volume deformation.

The simulations are performed in Python using the library FEniCSx [21, 22] and the results are analyzed with the software ParaView. The coupled discretized equations are integrated with a semi-implicit Euler method. At each time step, we minimize the elastic energy with respect to the placement $\boldsymbol{u}$. From such computation we update the value of J and $\mathbf{C}$. Then we solve the equations for θ_b, Y, and D, and consequently update J_ref and the elastic moduli E, λ, and μ (see Table 4.1).

The material domain Ω_b is a box with sides of centimeters 40x16x16. An unstructured mesh of the domain with maximum element diameter $h = 0.1\,\mathrm{cm}$ was chosen after checking the mesh convergence of the solution to a test heat equation. A time step of 1 s, well within the bounds of the standard CFL condition for the heat equation, was used, since the CO_2 diffusion equation poses less restrictive bounds. In our simulations, the value of the Courant number associated with the employed space and time discretization is about 0.6. Given that we are interested in simulating leavening times of the order of two hours and the computational time for each iteration is of the order of a few seconds, the total time of each simulation is quite reasonable. We stress that, for the dynamic under consideration, temperature evolution is faster in comparison of yeast growth and carbon dioxide diffusion.

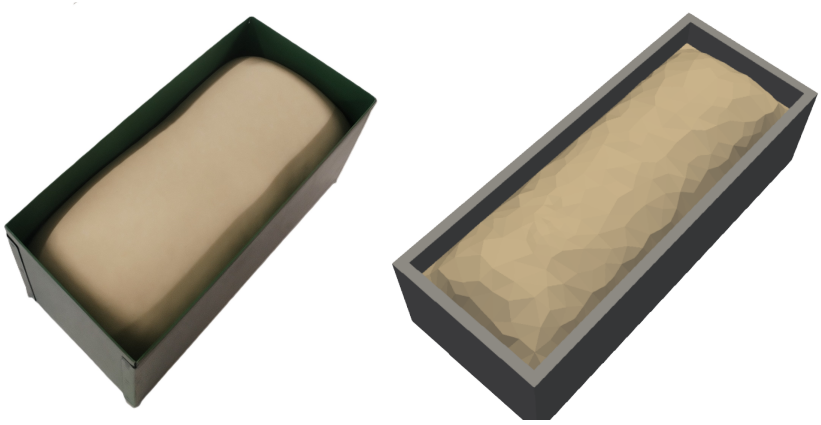

Figure 4.3: The real bread, produced industrially by "SMACT" on the right, on the left our 3D domain on which we perform numerical simulations.

Algorithm:
for $n = 0, \ldots, n_{\text{steps}}$:
$\rightarrow$ **solve** the **elasticity** eq. (iterative nonlinear solver)
$\rightarrow$ **update** value of J and $\mathbf{C}$
$\rightarrow$ **solve** the heat eq. in the bread dough
$\rightarrow$ **compute** the yeasts growth rate
$\rightarrow$ **solve** the diffusion eq. for the CO_2 concentration
$\rightarrow$ **compute** the volume occupied by CO_2
$\rightarrow$ **update** the value of J_{ref} and the elastic stiffness
end loop

Table 4.1: Semi-implicit iterative scheme used to solve the discretized model.

Value	Units	Description
$\boldsymbol{b} = (0, 0, -9.8)$	$\mathrm{m\,s^{-2}}$	body force- gravity
$\theta_0 = 18$	°C	initial temperature of the dough
$\beta_\mathrm{b} = 0.4125$	$\mathrm{J\,K^{-1}\,m^{-1}\,s^{-1}}$	thermal conductivity
$h_\mathrm{bo} = 50$	$\mathrm{J\,K^{-1}\,m^{-2}\,s^{-1}}$	bread-to-air heat transfer coefficient
$h_\mathrm{bt} = 60$	$\mathrm{J\,K^{-1}\,m^{-2}\,s^{-1}}$	bread-to-baking tray heat transfer coefficient
$c_\mathrm{w} = 4186$	$\mathrm{J\,K^{-1}\,kg^{-1}}$	specific heat water
$c_\mathrm{b} = 4186$	$\mathrm{J\,K^{-1}\,kg^{-1}}$	specific heat water
$\rho_\mathrm{b} = 170$	$\mathrm{kg\,m^{-3}}$	bread density
$\beta_{co2} = 1.e-3$	$\mathrm{m^2\,s^{-1}}$	diffusivity of concentration of CO_2: air value
$\lambda_\mathrm{y} = 0.00005$	$\mathrm{s^{-1}}$	
$\lambda_{co2} = 2.e-6$	$\mathrm{s^{-1}}$	
$R = 8.31$	$\mathrm{J\,mol^{-1}\,K^{-1}}$	
$n_\mathrm{mol} = 1/44$		number of moles contained in a gram of CO_2
$V_0^{1g} = 4,38e-6$	$\mathrm{m^{-3}}$	total volume of a gram of bread paste $= W_\mathrm{rate}\,V_\mathrm{w} + F_\mathrm{rate}\,V_\mathrm{f} = (1 - F_\mathrm{rate})\,V_\mathrm{w} + F_\mathrm{rate}\,V_\mathrm{f}$ $= 0.4 \cdot 1\mathrm{L} + 0.6 \cdot 1.64\mathrm{L} = 4.38 \cdot 10^{-6}\mathrm{cm}^3$
$P_\mathrm{diss} = 30$	$\mathrm{W°C^{-1}}$	dissipated power per °C

Table 4.2: Fixed input parameters used in the simulations.

4.2.1 Results

We perform some experiments to compare the behavior of bread leavening in the following different conditions:

- the temperature of the leavening chamber set to 18°C, 25°C, 30°C, or 45°C;
- the yeast initial concentration set to 2%, 3%, or 4%.

The other input parameters fixed throughout our tests are listed in Table 4.2. Notice that, when the baking paste is very humid the specific heat of the dough is similar to the one of water. In our model we assume that yeast produces CO_2 within a temperature range between 5°C and 45°C with a peak of production at 25°C (see Figure 2.1(b)).

From the simulation results, we can interrogate the model to understand how much time is needed to achieve a given increase in volume under different settings (see Figure 4.4). While the effect of yeast concentration is quite obvious (faster growth with more yeast), the temperature of the leavening chamber affects the volume expansion in a less trivial way. Indeed, setting the temperature too far beyond the peak of CO_2 production results in a rather fast expansion for short times followed by a substantial loss of the leavening capability, with a consequent saturation of the growth, as shown in Figure 4.4(d). This would bear little consequences if the target increase were 50%, but would be a crucial piece of information if one needed to reach a 200% expansion, which would be impossible with a temperature of 45°C. The most efficient growth over two hours is achieved with 30°C, a little above the peak CO_2 production temperature, see Figure 4.4(c).

The final goal of this study is to provide a tool for the prediction of energy consumption in industrial processes. We then consider that the energy used by the leavening process is the sum of the energy absorbed by the bread dough

$$\mathcal{E}_{\mathrm{b}}(t) = \int_{\Omega} \rho_{\mathrm{b}} c_{\mathrm{b}} \theta_{\mathrm{b}}(\boldsymbol{X}, t)\, d\boldsymbol{X} - \mathcal{E}_0, \tag{4.10}$$

where $\mathcal{E}_0 = \int_{\Omega} \rho_{\mathrm{b}} c_{\mathrm{b}} \theta_0(\boldsymbol{X})\, d\boldsymbol{X}$ is the initial thermal energy, and the energy $\mathcal{E}_{\mathrm{diss}}$ dissipated by the leavening chamber simply to reach and keep the temperature θ_{o} starting from θ_0 and for a time t, that is given by

$$\mathcal{E}_{\mathrm{diss}}(t) = P_{\mathrm{diss}}\, t\, (\theta_{\mathrm{o}} - \theta_0), \tag{4.11}$$

where the coefficient P_{diss} is the dissipated power per °C.

We then compute the energy expenditure as a function of time for all the performed experiments (Figure 4.5). First of all, we observe that the yeast concentration does not affect the energy required during the process in a measurable way, but does impact on the time necessary to reach any given target volume. By marking on the energy consumption

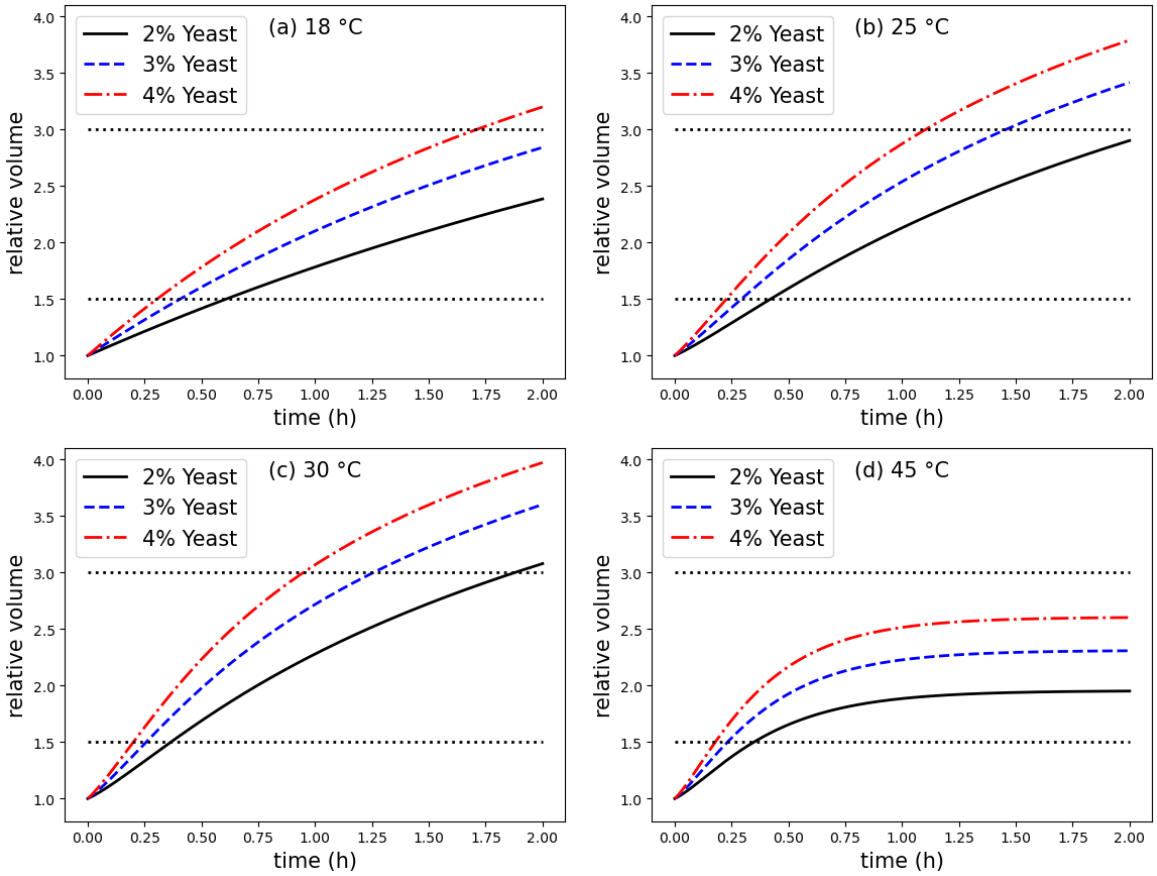

Figure 4.4: Yeast concentration and the temperature of the leavening chamber influence the volumetric expansion in different ways. By considering the relative volume expansion $v(t)/v_0$ as a function of time for different yeast concentrations and temperatures, as indicated on the graphs, we can predict the time necessary to reach a given expansion, if at all possible. Horizontal dotted lines represent two possible target volumes: $v_f = 1.5v_0$ or $v_f = 3v_0$.

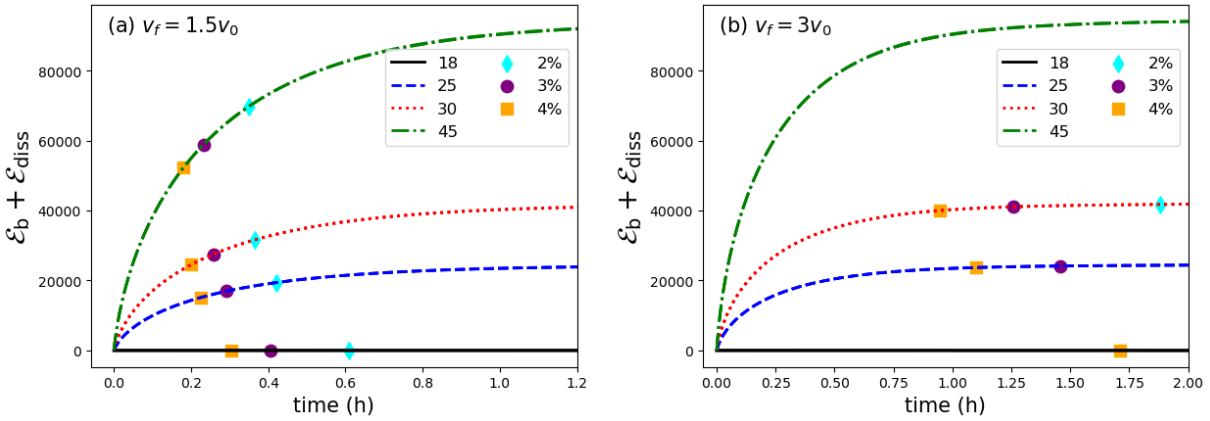

Figure 4.5: The time to reach a target volume increases upon decreasing the temperature and energy consumption, while the yeast concentration affects only the time to target. The energy consumption is plotted as a function of time for different system settings (see legend). The symbols on each curve represent the reaching of the target volume. In (a) we consider a target expansion of 50%, while in (b) a 200% expansion.

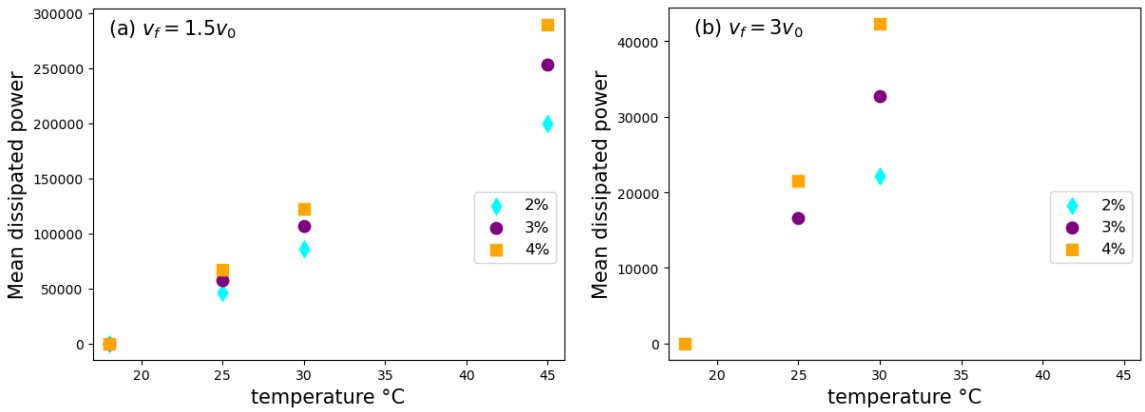

Figure 4.6: The evolution of the consumed energy divided by the time required to achieve the target volume $v_f = 1.5v_0$ or $v_f = 3v_0$ for different temperatures and yeast concentration.

$\mathcal{E}_b + \mathcal{E}_{diss}$ $v_f = 1.5\, v_0$	Temperature			
Yeast	18°C	25°C	30°C	45°C
0.2%	0 J	19405 J	31671 J	69790 J
0.3%	0 J	16932 J	27492 J	58803 J
0.4%	0 J	15157 J	24522 J	52215 J

$\mathcal{E}_b + \mathcal{E}_{diss}$ $v_f = 3\, v_0$	Temperature			
Yeast	18°C	25°C	30°C	45°C
0.2%	-	-	41733 J	-
0.3%	-	24135 J	41066 J	-
0.4%	0 J	23731 J	40085 J	-

Table 4.3: Energy consumption to reach the target volume v_f starting from v_0 for different system settings.

curves the point at which an expansion of either 50% or 200% is reached (Figure 4.5, panel (a) or (b), respectively), we see that there is a competition between the energetic cost and the time to target for any fixed yeast concentration. In Figures 4.6 we see the evolution of the consumed energy divided by the time required to achieve, when possible, the target volume v_f. This represents a measure of consumed energy. The resulting energy consumption to reach two different target volumes for all the experiments is summarized in Table 4.3. Practical considerations based on this kind of simulation can lead to more effective strategies for optimizing energy consumption given the time constraint due to production needs.

Chapter 5

Surrogate modelling for bread leavening

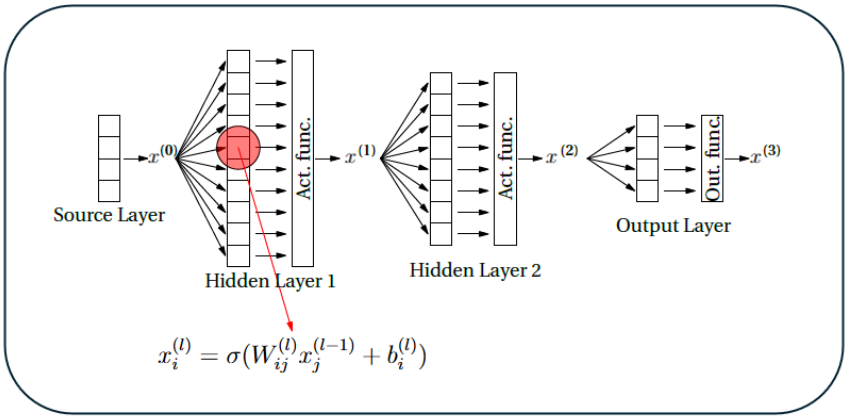

Figure 5.1: MLP with 2 hidden layers.

In this chapter, we want to introduce some machine learning techniques (see [23] for some references). Especially, we start with neural networks then, operator networks and finally, we describe the Variational Mimetic Operator Network (VarMiON) procedure [2] that allows us to reduce the order of our leavening model. The choice of using the latter to build a surrogate model, instead of the other existing methods in the literature, is due to the collaboration with Prof. Assad Oberai, who is a developer of VarMiON, and to the possibility of exploiting the variational formulation of our leavening model.

5.1 Deep neural networks

In this section, we describe the simplest network architecture that is available: the multi-layer perceptron (MLP).

Let us denote with $\mathcal{F}$ an MLP to the end of approximating the function $f : \boldsymbol{x} \in \mathbb{R}^d \to \boldsymbol{y} \in \mathbb{R}^D$. The units of an MLP are the artificial neurons stacked in several consecutive layers. The zeroth layer of $\mathcal{F}$ is called the source layer, which is only responsible for providing an input (of dimension d) to the network. The last layer of $\mathcal{F}$ is known as the output layer, which outputs the network's prediction (of dimension D). Every other layer in between is known as a hidden layer. The number of neurons in a layer defines the width of that layer (see Figure 5.1). We identify with H_l, $l = 0, 1, \dots, L+1$ the layer of width l, where $H_0 = d$, $H_{L+1} = D$ and $L+1$ represents the depth of the network. In each layer l the i-th neuron performs an affine transformation on that layers input $\boldsymbol{x}^{(l-1)}$ followed by a non-linear transformation as

$$x_i^{(l)} = \sigma(W_{ij}^{(l)} x_j^{(l-1)} + b_i^{(l)}), \qquad 1 \le i \le H_l, \quad 1 \le j \le H_{l-1} \tag{5.1}$$

where we set $\boldsymbol{x}(0) = \boldsymbol{x} \in \mathbb{R}^d$ the input provided by the layer and $W_{ij}^{(l)}$, $b_i^{(l)}$ are the weights and bias associated with i-th neuron of layer l, while the function σ is known as the activa-

tion function. We can re-write the action of the whole layer using the matrix formulation as

$$\boldsymbol{x}^{(l)} = \sigma(\mathbf{A}^{(l)}(\boldsymbol{x}^{(l-1)})) \qquad \mathbf{A}^{(l)}(\boldsymbol{x}^{(l-1)}) = \mathbf{W}^{(l)}\boldsymbol{x}^{(l-1)} + \boldsymbol{b}^{(l)} \tag{5.2}$$

with $\mathbf{W}^{(l)} \in \mathbb{R}^{H_{l-1}\times H_l}$ and $\boldsymbol{b}^{(l)} \in \mathbb{R}^{H_l}$. Thus, the action of the whole network $\mathcal{F} : \mathbb{R}^d \to \mathbb{R}^D$ can be mathematically seen as a composition of alternating affine transformations and component-wise activations

$$\mathcal{F}(\boldsymbol{x}) = \mathbf{A}^{(L+1)} \circ \sigma \circ \mathbf{A}^{(L)} \circ \sigma \circ \cdots \circ \sigma \circ \mathbf{A}^{(1)}(\boldsymbol{x}). \tag{5.3}$$

The parameters of the network are all the weights and biases, which we represent as $\boldsymbol{\theta} = \{\mathbf{W}^{(l)}, \boldsymbol{b}^{(l)}\}_{l=1}^{L+1}$. The network $\mathcal{F}(\boldsymbol{x}, \boldsymbol{\theta})$ represents a family of parameterized functions, where $\boldsymbol{\theta}$ needs to be suitably chosen such that the network approximates the target function $f(\boldsymbol{x})$ at the input $\boldsymbol{x}$.

The activation function is perhaps the most important component of an MLP, a list of possible choices follows.

- Linear activation: the simplest function $\sigma(\boldsymbol{\xi}) = \boldsymbol{\xi}$ which gives a linear approximation of the target function f.
- Rectified linear unit ReLU: is picewise linear $\sigma(\boldsymbol{\xi}) = \max\{0, \boldsymbol{\xi}\}$.
- Leaky ReLU: $\sigma(\boldsymbol{\xi}, \alpha) = \begin{cases} \boldsymbol{\xi} & \text{if } \boldsymbol{\xi} \geq 0 \\ \alpha\boldsymbol{\xi} & \text{if } \boldsymbol{\xi} < 0 \end{cases}$, where α is the hyper parameter.
- Logistic function: $\sigma(\boldsymbol{\xi}) = \frac{1}{1+e^{-\boldsymbol{\xi}}}$. This activation function can lead to slow convergence of the network while training.
- Tanh can be seen as the symmetric extension of the logistic function: $\sigma(\boldsymbol{\xi}) = \frac{e^{\boldsymbol{\xi}}-e^{-\boldsymbol{\xi}}}{e^{\boldsymbol{\xi}}+e^{-\boldsymbol{\xi}}}$.
- Sine $\sigma(\boldsymbol{\xi}) = \sin(\boldsymbol{\xi})$.

Now let us discuss how the parameters of these networks $\boldsymbol{\theta}$ are set to approximate some target function. We assume that we are given a dataset of pairwise samples $S = \{(\boldsymbol{x}_i, \boldsymbol{y}_i) : 1 \leq i \leq N\}$ corresponding to a target function f where $f(\boldsymbol{x}_i) = \boldsymbol{y}_i$. We wish to approximate this function using the neural network $\mathcal{F}(\boldsymbol{x}, \boldsymbol{\theta}, \boldsymbol{\Theta})$ where $\boldsymbol{\theta}$ are the network parameters defined before, while $\boldsymbol{\Theta}$ correspond to the hyper-parameters of the network such as the depth $L+1$, width H, type of activation function σ, etc. The strategy to design a robust network involves three steps:

1. find the optimal values of $\boldsymbol{\theta}$ (for a fixed $\boldsymbol{\Theta}$) in the **training phase**;
2. find the optimal values of $\boldsymbol{\Theta}$ in the **validation phase**;
3. test the network performance on unseen data on the **testing phase**.

Notice that it is useful to split the dataset S into three distinct parts: N_{train}, N_{val} and N_{test} samples that compose a training set S_{train}, a validation set S_{val} and a test set S_{test} respectively, such that $N = N_{\text{train}} + N_{\text{val}} + N_{\text{test}}$.

Training phase

Training the network makes use of the training set S_{train} to solve the following optimization problem: find $\boldsymbol{\theta}^*$ for fixed $\boldsymbol{\Theta}$ such that

$$\boldsymbol{\theta}^* = \underset{\boldsymbol{\theta}}{\operatorname{argmin}}\, \Pi_{\text{train}}(\boldsymbol{\theta}), \tag{5.4}$$

$$\Pi_{\text{train}}(\boldsymbol{\theta}) = \frac{1}{N_{\text{train}}} \sum_{\substack{i=1, \\ (\boldsymbol{x}_i, \boldsymbol{y}_i) \in S_{\text{train}}}}^{N_{\text{train}}} \|\boldsymbol{y}_i - \mathcal{F}(\boldsymbol{x}_i, \boldsymbol{\theta}, \boldsymbol{\Theta})\|^2. \tag{5.5}$$

The loss function is the function Π_{train}, above we have used the mean-squared loss function.

Validation phase

Validation of the network involves using the validation set S_{val} to solve the following optimization problem: find $\boldsymbol{\Theta}^*$ such that

$$\boldsymbol{\Theta}^* = \underset{\boldsymbol{\Theta}}{\operatorname{argmin}}\, \Pi_{\text{val}}(\boldsymbol{\Theta}), \tag{5.6}$$

$$\Pi_{\text{val}}(\boldsymbol{\Theta}) = \frac{1}{N_{\text{val}}} \sum_{\substack{i=1, \\ (\boldsymbol{x}_i, \boldsymbol{y}_i) \in S_{\text{val}}}}^{N_{\text{val}}} \|\boldsymbol{y}_i - \mathcal{F}(\boldsymbol{x}_i, \boldsymbol{\theta}^*, \boldsymbol{\Theta})\|^2. \tag{5.7}$$

where a standard technique to obtain $\boldsymbol{\Theta}^*$ is the (random or tensor) grid search.

Testing phase

The test set S_{test} is used to estimate the network performance on data not used during the first two phases

$$\Pi_{\text{test}} = \frac{1}{N_{\text{test}}} \sum_{\substack{i=1, \\ (\boldsymbol{x}_i, \boldsymbol{y}_i) \in S_{\text{test}}}}^{N_{\text{test}}} \|\boldsymbol{y}_i - \mathcal{F}(\boldsymbol{x}_i, \boldsymbol{\theta}^*, \boldsymbol{\Theta}^*)\|^2. \tag{5.8}$$

This testing error is also known as the (approximate) generalizing error of the network.

Neural networks, especially MLPs, are almost always over-parametrized, this would lead to a highly non-linear network model, for which the loss function Π_{train} can have a

landscape with many local minima. To overcome this type of issue it can be employed a regularization technique that involves augmenting a penalty term to the loss function $\Pi_{\text{train}} + \alpha\|\boldsymbol{\theta}\|$ where $\alpha \geq 0$ is a regularization hyper-parameter, and $\|\boldsymbol{\theta}\|$ is a suitable norm of the network parameters.

5.2 Physics-informed neural networks

A PDE can be solved numerically by exploiting some different methods such as finite difference methods, finite volume methods, or finite element methods (seen in Chapter 4), etc. Therefore, this section aims to describe how PDEs can be solved with MLPs (for some references on this topic see [23, 35]).

The term "Physics-Informed Neural Networks" (PINNs) reflects the idea of using neural networks to solve partial differential equations. Consider a general PDE: find the solution $\boldsymbol{u} : \Omega \subset \mathbb{R}^d \to \mathbb{R}^D$ such that

$$L(\boldsymbol{u}(\boldsymbol{x})) = f(\boldsymbol{x}) \qquad \boldsymbol{x} \in \Omega, \tag{5.9}$$

$$B(\boldsymbol{u}(\boldsymbol{x})) = g(\boldsymbol{x}) \qquad \boldsymbol{x} \in \partial\Omega, \tag{5.10}$$

where L is a differential operator, f is a known forcing term, B is a boundary operator, and g is a non-homogeneous boundary condition (also prescribed).

Let us see how to design a PINN, where the input to the network should be the independent variables $\boldsymbol{x}$ and the output should be the solution vector $\boldsymbol{u}$.

1. Construct the loss functions

   - we define the interior residual $R(\boldsymbol{u}) = L(\boldsymbol{u}) - f$;
   - we define the boundary residual $R_b(\boldsymbol{u}) = B(\boldsymbol{u}) - g$;
   - we select suitable N_v collocation points in the interior of the domain and N_b points on the domain boundary to evaluate the residuals.

   Then the loss function is

$$\Pi_{\text{train}}(\boldsymbol{\theta}) = \Pi_{\text{int}}(\boldsymbol{\theta}) + \lambda_b \Pi_b(\boldsymbol{\theta}), \tag{5.11}$$

$$\Pi_{\text{int}}(\boldsymbol{\theta}) = \frac{1}{N_v} \sum_{i=1}^{N_v} |R(\mathcal{F}(\boldsymbol{x}_i, \boldsymbol{\theta}))|^2, \tag{5.12}$$

$$\Pi_b(\boldsymbol{\theta}) = \frac{1}{N_b} \sum_{i=1}^{N_b} |R_b(\mathcal{F}(\boldsymbol{x}_i, \boldsymbol{\theta}))|^2. \tag{5.13}$$

2. Train the network: find $\boldsymbol{\theta}^* = \underset{\boldsymbol{\theta}}{\operatorname{argmin}}\, \Pi_{\text{train}}(\boldsymbol{\theta})$ and set the solution as $\boldsymbol{u}^* = \mathcal{F}(\boldsymbol{x}, \boldsymbol{\theta}^*)$.

Example 5.2.1. *If we consider the PDE*

$$\nabla \cdot (\beta \nabla \boldsymbol{u}) = f(\boldsymbol{x}) \qquad \boldsymbol{x} \in \Omega, \tag{5.14}$$

$$\boldsymbol{u}(\boldsymbol{x}) = g(\boldsymbol{x}) \qquad \boldsymbol{x} \in \partial\Omega, \tag{5.15}$$

and train a PINN $\mathcal{F}(\boldsymbol{x}_i, \boldsymbol{\theta})$ to minimize the loss function

$$\Pi_{\text{train}}(\boldsymbol{\theta}) = \frac{1}{N_v} \sum_{i=1}^{N_v} |\nabla \cdot (\beta \nabla \mathcal{F}(\boldsymbol{x}_i, \boldsymbol{\theta})) - f(\boldsymbol{x}_i)|^2 + \frac{\lambda_b}{N_b} \sum_{i=1}^{N_b} |\mathcal{F}(\boldsymbol{x}_i, \boldsymbol{\theta}) - g(\boldsymbol{x}_i)|^2. \tag{5.16}$$

If $\boldsymbol{\theta}^* = \underset{\boldsymbol{\theta}}{\operatorname{argmin}}\, \Pi_{\text{train}}(\boldsymbol{\theta})$ then the PINN approximating the PDE is $\boldsymbol{u}^*(\boldsymbol{x}) = \mathcal{F}(\boldsymbol{x}, \boldsymbol{\theta}^*)$.

Notice that if we change f and/or g then we need to retrain the network (with perhaps the same architecture) for the new f and g. In the next section, we will see how we can overcome this issue.

5.3 Operator networks

Now we, illustrate how to approximate operators that map functions to functions. This will be useful in the perspective of having PINN whose inputs are the source f and boundary term g of a PDE.

Consider a class of functions $a(\boldsymbol{y}) \in A$ such that $a : \Omega_A \to \mathbb{R}^D$ which has certain properties, such as $a \in \mathcal{C}(\Omega_A)$ or $a \in L^2(\Omega_A)$. Also consider the operator $\mathcal{N} : A \to \mathcal{C}(\Omega)$, with $u(\boldsymbol{x}) = \mathcal{N}(a)(\boldsymbol{x})$ for $\boldsymbol{x} \in \Omega$. We are now interested in networks that approximate the operator $\mathcal{N}$.

Example 5.3.1. *Let us see some examples of operators $\mathcal{N}$.*

1. *Consider the following PDE*

$$\nabla \cdot (\beta \nabla u) = f(\boldsymbol{x}) \qquad \boldsymbol{x} \in \Omega, \tag{5.17}$$

$$u(\boldsymbol{x}) = g(\boldsymbol{x}) \qquad \boldsymbol{x} \in \partial\Omega. \tag{5.18}$$

   - *The operator $\mathcal{N}$ maps the f to the solution u of the PDE. That is $u = \mathcal{N}(f)(\boldsymbol{x})$ for β and g given.*
   - *The operator $\mathcal{N}$ maps the diffusivity β to the solution u of the PDE. That is $u = \mathcal{N}(\beta)(\boldsymbol{x})$ for f and g given.*
   - *The operator $\mathcal{N}$ maps the boundary condition g and the diffusivity β to the solution u of the PDE. That is $u = \mathcal{N}(\beta, g)(\boldsymbol{x})$ for f given.*

2. Consider the following PDE

$$\frac{\partial}{\partial t}u + a \cdot \nabla u - \beta \nabla^2 u + u(1-u) = f \qquad (\boldsymbol{x}, t) \in \Omega \times [0, \tau], \tag{5.19}$$

$$u(\boldsymbol{x}, t) = g(\boldsymbol{x}) \qquad (\boldsymbol{x}, t) \in \partial\Omega \times [0, \tau], \tag{5.20}$$

$$u(\boldsymbol{x}, 0) = u_0(\boldsymbol{x}) \qquad \boldsymbol{x} \in \Omega. \tag{5.21}$$

We want to find the operator $\mathcal{N}$ that maps the initial condition u_0 to the solution u at the final time τ, i.e. $u(\boldsymbol{x}, \tau) = \mathcal{N}(u_0)(\boldsymbol{x})$ where a, β, f, g are fixed.

5.3.1 Deep Operator Network (DeepONet)

In this section, we describe a popular version of these operator networks: the Deep Operator Network (DeepONet).

A standard DeepONet comprises two neural networks to approximate an operator $\mathcal{N} : A \to U$ where A is the set of functions $a : \Omega_A \subset \mathbb{R}^d \to \mathbb{R}$ and U is the space of functions $u : \Omega \subset \mathbb{R}^D \to \mathbb{R}$. A schematic architecture for DeepONet follows (Figure 5.2):

1. Fix M distinct sensor points $\boldsymbol{y}(1), \ldots, \boldsymbol{y}(M)$ in Ω_A.

2. Sample a function $a \in A$ at these sensor points to get the vector $\boldsymbol{a} = [a(\boldsymbol{y}(1)), \ldots, a(\boldsymbol{y}(M))]^\top \in \mathbb{R}^M$.

3. Supply $\boldsymbol{a}$ as the input to a sub-network, called the branch net $\mathcal{B}(\cdot, \boldsymbol{\theta}_B) : \mathbb{R}^M \to \mathbb{R}^p$, whose output would be the vector $\boldsymbol{\beta} = [\beta_1(\boldsymbol{a}), \ldots, \beta_p(\boldsymbol{a})]^\top \in \mathbb{R}^p$. Here $\boldsymbol{\theta}_B$ are the trainable parameters of the branch net. The dimension of the output of the branch p is relatively small. From now on, we will refer to p as the latent dimension of the network.

4. Supply $\boldsymbol{x}$ as an input to a second sub-network, called the trunk net $\mathcal{T}(\cdot, \boldsymbol{\theta}_T) : \mathbb{R}^D \to \mathbb{R}^p$, whose output would be the vector $\boldsymbol{\tau} = [\tau_1(\boldsymbol{x}), \ldots, \tau_p(\boldsymbol{x})]^\top \in \mathbb{R}^p$. Here $\boldsymbol{\theta}_T$ are the trainable parameters of the trunk net.

5. Take a dot product of the outputs of the branch and trunk nets to get the final output of the DeepONet $\tilde{\mathcal{N}}(\cdot, \cdot, \boldsymbol{\theta}) : \mathbb{R}^D \times \mathbb{R}^M \to \mathbb{R}$ which will approximate the value of $u(\boldsymbol{x})$

$$u(\boldsymbol{x}) \approx \tilde{\mathcal{N}}(\boldsymbol{x}, \boldsymbol{a}, \boldsymbol{\theta}) = \sum_{k=1}^{p} \beta_k(\boldsymbol{a}) \tau_k(\boldsymbol{x}) \tag{5.22}$$

where the trainable parameters of the DeepONet will be the combined parameters of the branch and trunk nets, i.e. $\boldsymbol{\theta} = [\boldsymbol{\theta}_B, \boldsymbol{\theta}_T]$.

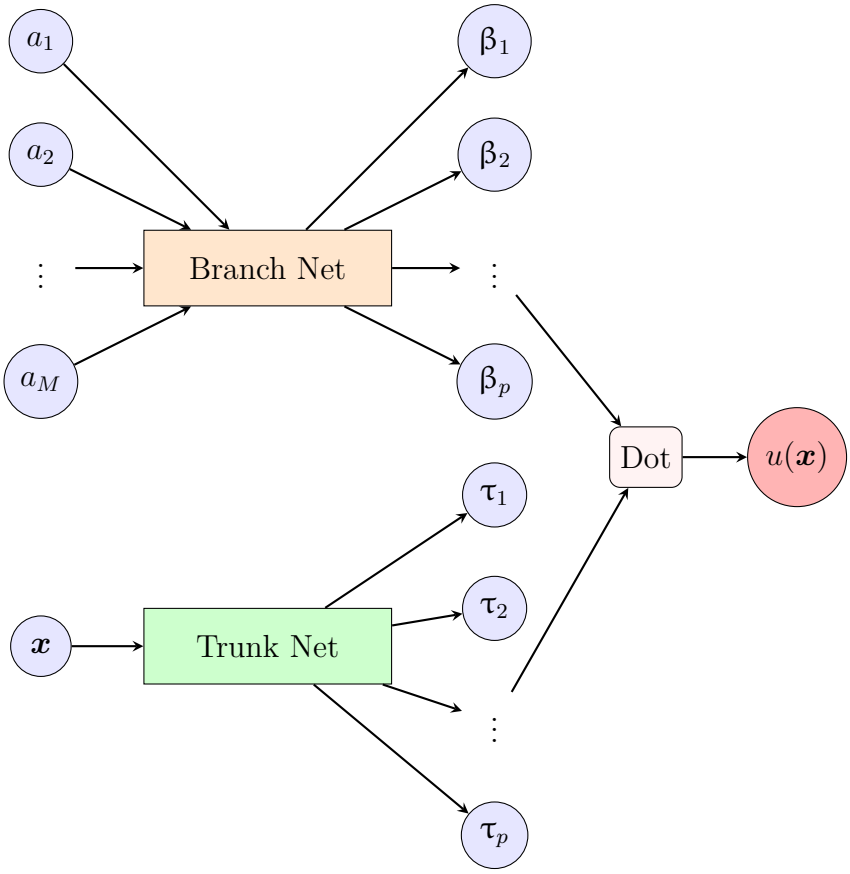

Figure 5.2: Schematic of a DeepONet.

The goal is to approximate the value of any $\mathcal{N}(a)(\boldsymbol{x})$ for any $a \in A$ and any $\boldsymbol{x} \in \Omega$. The solution is written as the sum of a series of coefficients and functions, where the coefficients are determined by the branch network, while the functions are determined by the trunk network. This procedure is similar to FEM but, in the DeepONet the basis functions are determined by a trunk network and their final form depends on the data used in the training. To train the network we have to perform the following steps.

1. Select J representative functions $a^j, 1 \leq j \leq J$ from the set A. Evaluate the values of these functions at the M sensor points, i.e., $a_i^j = a^j(\boldsymbol{y}(i))$ for $1 \leq i \leq M$. This gives us the vectors $\boldsymbol{a}^j = [a^j(\boldsymbol{y}(1)), \ldots, a^j(\boldsymbol{y}(M))]^\top \in \mathbb{R}^M$ for each $1 \leq j \leq J$.

2. For each $\boldsymbol{a}^j$, determine (numerically or analytically) the corresponding functions u^j given by the operator $\mathcal{N}$.

3. Sample the function u^j at L points in Ω, i.e. $u^j(\boldsymbol{x}_l)$ for $1 \leq l \leq L$.

4. Construct the training set

$$S = \left\{(\boldsymbol{a}^j, \boldsymbol{x}_l, u^j(\boldsymbol{x}_l)) : 1 \leq j \leq J, 1 \leq l \leq L\right\} \tag{5.23}$$

   which has $J \times L$ samples.

5. Define the loss function

$$\Pi(\boldsymbol{\theta}) = \frac{1}{J\,L} \sum_{j=1}^{J} \sum_{l=1}^{L} |\tilde{\mathcal{N}}(\boldsymbol{x}_l, \boldsymbol{a}^j, \boldsymbol{\theta}) - u^j(\boldsymbol{x}_l)|^2. \tag{5.24}$$

6. Training the DeepONet corresponds to finding $\boldsymbol{\theta}^* = \underset{\boldsymbol{\theta}}{\operatorname{argmin}}\, \Pi(\boldsymbol{\theta})$

7. Once trained, then given any new $a \in A$ samples at the M sensor points (which gives the vector $\boldsymbol{a} \in \mathbb{R}^M$) and a new point $\boldsymbol{x} \in \Omega$, we can evaluate the corresponding prediction $u^*(\boldsymbol{x}) = \tilde{\mathcal{N}}(\boldsymbol{x}, \boldsymbol{a}, \boldsymbol{\theta}^*)$.

Theorem 5.3.2. *Suppose Ω, A are compact sets in $\mathbb{R}^D$ (or more generally a Banach space) and $\mathbb{R}^d$, respectively. Let $\mathcal{N}$ be a nonlinear, continuous operator mapping. Then given $\varepsilon > 0$, there exists a DeepONet with M sensors and a single hidden layer of width p in the branch and trunk nets such that*

$$\max_{\boldsymbol{x} \in \Omega, a \in A} |\tilde{\mathcal{N}}(\boldsymbol{x}, \boldsymbol{a}, \boldsymbol{\theta}) - \mathcal{N}(a)(\boldsymbol{x})| < \varepsilon \tag{5.25}$$

for a large enough p and M.

Physics-informed DeepONet

If we want to exploit the PDE satisfied by the function u, for example,

$$\nabla \cdot (\beta \nabla u) = f(\boldsymbol{x}) \qquad \boldsymbol{x} \in \Omega, \tag{5.26}$$
$$u(\boldsymbol{x}) = g(\boldsymbol{x}) \qquad \boldsymbol{x} \in \partial\Omega, \tag{5.27}$$

we can modify the total loss function as a weighted sum of the physics-based loss Π_p and the standard data-driven loss function Π_d:

$$\Pi(\boldsymbol{\theta}) = \Pi_d(\boldsymbol{\theta}) + \lambda \Pi_p(\boldsymbol{\theta}) \tag{5.28}$$
$$\Pi_p(\boldsymbol{\theta}) = \frac{1}{\bar{J}\bar{L}} \sum_{j=1}^{\bar{J}} \sum_{l=1}^{\bar{L}} |\nabla \cdot (\beta \nabla \tilde{\mathcal{N}}(\boldsymbol{x}_l, \boldsymbol{f}^j, \boldsymbol{\theta})) - f^j(\boldsymbol{x}_l)|^2 \tag{5.29}$$
$$\Pi_d(\boldsymbol{\theta}) = \frac{1}{JL} \sum_{j=1}^{J} \sum_{l=1}^{L} |\tilde{\mathcal{N}}(\boldsymbol{x}_l, \boldsymbol{f}^j, \boldsymbol{\theta})) - u^j(\boldsymbol{x}_l)|^2 \tag{5.30}$$

where λ is a hyper-parameter.

Notice that the output sensor points used in the physics-based loss function $\bar{L}$ and the set of input functions $\bar{J}$ are usually distinct from the output sensor points L and the set of input functions J used in the data-driven loss term. The former represents the locations and functions at which we wish to minimize the residual of the PDE, while the latter represents the points and functions at which the solution is available to us through external means.

The advantages of adding the extra physics-based loss are that we don't have to generate as many solutions of the PDE for training the DeepONet and it makes the network more robust.

5.4 Variationally Mimetic Operator Networks (VarMiON)

In this section, we describe the Variationally Mimetic Operator Networks (VarMiON) presented in paper [2]. In that work, the authors describe a new architecture for operator networks that mimics the form of the numerical solution obtained from an approximation of the variational or weak formulation of the problem. Like the conventional DeepONet the VarMiON is also composed of a sub-network that constructs the basis functions for the output and another that constructs the coefficients for these basis functions, but where the architecture is precisely determined.

5.4.1 VarMiON for elliptic equations

Consider a general elliptic PDE:

$$L(u(\boldsymbol{x}), \beta(\boldsymbol{x})) = f(\boldsymbol{x}) \qquad \boldsymbol{x} \in \Omega, \tag{5.31}$$
$$B(u(\boldsymbol{x}), \beta(\boldsymbol{x})) = \eta(\boldsymbol{x}) \qquad \boldsymbol{x} \in \Gamma_N, \tag{5.32}$$
$$u(\boldsymbol{x}) = g(\boldsymbol{x}) \qquad \boldsymbol{x} \in \Gamma_D, \tag{5.33}$$

where $\Omega \subset \mathbb{R}^d$, Γ_D and Γ_N represent the parts of Dirichlet and Neumann boundary respectively with $\partial\Omega = \Gamma_D \cup \Gamma_N$ and $\Gamma_D \cap \Gamma_N = \emptyset$. L is a second order operator, $f \in L^2(\Omega)$ is the known forcing term, B is the boundary operator, $\eta \in L^2(\Gamma_N)$ is the flux data, $g \in H^{1/2}(\Gamma_D)$ is the Dirichlet condition and $\beta \in L^\infty(\Omega)$ is a spatially varying material parameter, such as thermal conductivity or diffusivity.

Let us now consider the steady-state heat equation with homogeneous Dirichlet boundary conditions i.e., $g = 0$:

$$-\operatorname{div}(\beta(\boldsymbol{x})\nabla u(\boldsymbol{x})) = f(\boldsymbol{x}) \qquad \boldsymbol{x} \in \Omega, \tag{5.34}$$
$$\beta(\boldsymbol{x})\nabla u(\boldsymbol{x}) \cdot \boldsymbol{n} = \eta(\boldsymbol{x}) \qquad \boldsymbol{x} \in \Gamma_N, \tag{5.35}$$
$$u(\boldsymbol{x}) = 0 \qquad \boldsymbol{x} \in \Gamma_D. \tag{5.36}$$

The previous problem can be rewritten as:

$$\text{find } u \in V = H^1_D(\Omega) \text{ such that } a(u, \tilde{u}; \beta) = (\tilde{u}, f) + (\tilde{u}, \eta)_{\Gamma_N} \quad \forall \tilde{u} \in V, \tag{5.37}$$

where $(\cdot, \cdot)$ is the inner product, $\tilde{u}$ a suitable test function and $a(\cdot, \cdot; \beta)$ the bilinear form associated with the operator and $H^1_D = \{u \in H^1 : u|_{\Gamma_D} = 0\}$.

The solution operator is defined as:

$$\mathcal{N} : X = L^2(\Omega) \times L^2(\Gamma_N) \times L^2(\Omega) \to H^1_D(\Omega) = V$$
$$\mathcal{N}(f, \eta, \beta) = u(\cdot, f, \eta, \beta) \tag{5.38}$$

which maps the data (f, η, β) to the unique solution $u(\cdot, f, \eta, \beta)$. Our goal is to approximate the operator $\mathcal{N}$ using a VarMiON.

Training the VarMiON requires solutions of the PDE for varying values (f, η, β). In the absence of analytical expressions, the solutions can be approximated by using numerical solvers, such as the FEM described in Chapter 4.

Let V_h be a family of spaces that depends on a positive parameter h, such that $V_h \subset V$, $\dim V_h = N_h < \infty$, thus any function can be expressed as a combination of basis functions $\{\phi_j(\boldsymbol{x})\}_{j=1}^{N_h}$ of V_h

$$u_h(\boldsymbol{x}) = \sum_{j=1}^{N_h} u_j \phi_j(\boldsymbol{x}) = \boldsymbol{U}^\top \boldsymbol{\Phi}(\boldsymbol{x}) \tag{5.39}$$

where

$$\boldsymbol{\Phi}(\boldsymbol{x}) = (\phi_1(\boldsymbol{x}), \dots, \phi_{N_h}(\boldsymbol{x}))^\top, \tag{5.40}$$

$$\boldsymbol{U} = (u_1, \dots, u_{N_h})^\top. \tag{5.41}$$

We also define the restricted space $V_h|_{\Gamma_N}$ for the boundary data. Consider the projector

$$\mathcal{P} : X \to X_h \subset V_h \times V_h|_{\Gamma_N} \times V_h \qquad \mathcal{P}(f, \eta, \beta) = (f_h, \eta_h, \beta_h), \tag{5.42}$$

where the given PDE data is approximated (projected) as

$$f_h(\boldsymbol{x}) = \boldsymbol{F}^\top \boldsymbol{\Phi}(\boldsymbol{x}), \tag{5.43}$$

$$\eta_h(\boldsymbol{x}) = \boldsymbol{N}^\top \boldsymbol{\Phi}(\boldsymbol{x})|_{\Gamma_N}, \tag{5.44}$$

$$\beta_h(\boldsymbol{x}) = \boldsymbol{\beta}^\top \boldsymbol{\Phi}(\boldsymbol{x}). \tag{5.45}$$

Notice that the coefficients $\boldsymbol{F}^\top, \boldsymbol{N}^\top, \boldsymbol{\beta}^\top$ depend on the choice of the basis functions. The discrete weak formulation can be rewritten in matrix form as

$$\mathbf{K}(\beta_h)\boldsymbol{U} = \mathbf{M}\boldsymbol{F} + \tilde{\mathbf{M}}\boldsymbol{N}, \tag{5.46}$$

$$\boldsymbol{U} = \mathbf{K}(\beta_h)^{-1}(\mathbf{M}\boldsymbol{F} + \tilde{\mathbf{M}}\boldsymbol{N}), \tag{5.47}$$

where

$$\mathbf{K}_{ji}(\beta_h) = a(\phi_j, \phi_i; \beta_h), \quad \mathbf{M}_{ji} = (\phi_j, \phi_i), \quad \tilde{\mathbf{M}}_{ji} = (\phi_j, \phi_i)|_{\Gamma_N}. \tag{5.48}$$

Notice that the basis $\{\phi_j(\boldsymbol{x})\}_{j=1}^{N_h}$ is chosen such that the matrix $\mathbf{K}(\beta_h)$ is invertible for all β_h. Indeed, depending on the value of β_h we change the elements of the basis so that $\mathbf{K}$ is invertible. Notice that, if $\beta_h > 0$ then the bilinear form, associated with the problem, is coercive and $\mathbf{K}$ is positive definite and hence invertible. The discrete solution operator is defined as:

$$\begin{aligned} &\mathcal{N}_h : V_h \times V_h|_{\Gamma_N} \times V_h \to V_h \\ &\quad \mathcal{N}_h(f_h, \eta_h, \beta_h) = u_h(\cdot, f_h, \eta_h, \beta_h) = (\mathbf{B}(f_h, \beta_h) + \tilde{\mathbf{B}}(\eta_h, \beta_h))^\top \boldsymbol{\Phi} \end{aligned} \tag{5.49}$$

where

$$\mathbf{B}(f_h, \beta_h) = \mathbf{K}^{-1}(\beta_h)(\mathbf{M}\boldsymbol{F}) \quad \tilde{\mathbf{B}}(\eta_h, \beta_h) = \mathbf{K}^{-1}(\beta_h)(\tilde{\mathbf{M}}\boldsymbol{N}). \tag{5.50}$$

The VarMiON architecture is motivated by the discrete weak variational form. The PDE data $(f, \eta, \beta) \in X$ is fed into the VarMiON, with f and β sampled at the sensor nodes $\{\hat{\boldsymbol{x}}_i\}_{i=1}^{k}$ while η is sampled at the boundary sensor nodes $\{\hat{\boldsymbol{x}}_i{}^b\}_{i=1}^{k'}$. We define the input vectors

$$\hat{\boldsymbol{F}} = (f(\hat{\boldsymbol{x}}_1), \dots, f(\hat{\boldsymbol{x}}_k))^\top \tag{5.51}$$

$$\hat{\boldsymbol{N}} = (\eta(\hat{\boldsymbol{x}}_1^b), \dots, \eta(\hat{\boldsymbol{x}}_{k'}^b))^\top \tag{5.52}$$

$$\hat{\boldsymbol{\beta}} = (\beta(\hat{\boldsymbol{x}}_1), \dots, \beta(\hat{\boldsymbol{x}}_k))^\top \tag{5.53}$$

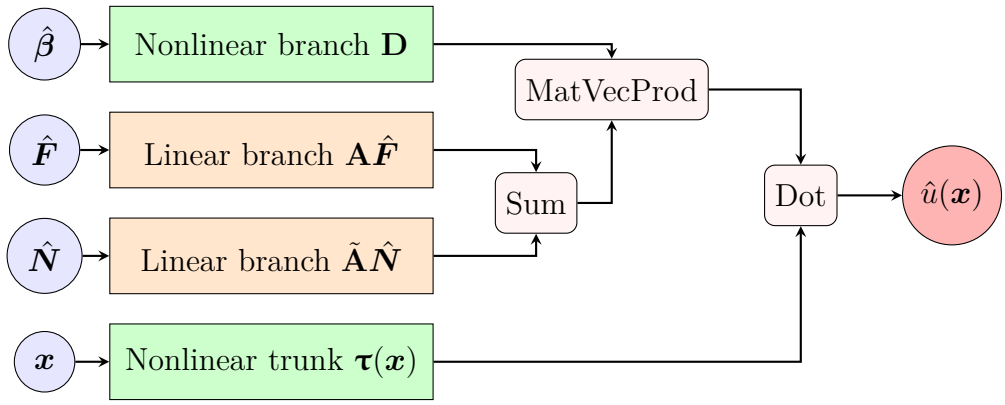

Figure 5.3: Schematic of a VarMiON for the steady-state heat equation.

given by the data sensing operator $\hat{\mathcal{P}}$ as

$$\hat{\mathcal{P}} : X \to \mathbb{R}^k \times \mathbb{R}^{k'} \times \mathbb{R}^k, \qquad \hat{\mathcal{P}}(f, \eta, \beta) = (\hat{\boldsymbol{F}}, \hat{\boldsymbol{N}}, \hat{\boldsymbol{\beta}}). \tag{5.54}$$

We consider the VarMiON shown in Figure 5.3, which comprises:

- A non-linear branch taking the input $\hat{\boldsymbol{\beta}}$, which is transformed into a matrix output $\mathbf{D}(\hat{\boldsymbol{\beta}}) \in \mathbb{R}^{p \times p}$. Here p is the latent dimension of the VarMiON.
- A linear branch taking the input $\hat{\boldsymbol{F}}$ and transforming it as $\mathbf{A}\hat{\boldsymbol{F}}$, where $\mathbf{A} \in \mathbb{R}^{p \times k}$ is a learnable matrix. The output of this branch is acted upon by the matrix $\mathbf{D}$ to give $\boldsymbol{\beta}(\hat{\boldsymbol{\beta}}, \hat{\boldsymbol{F}}) = \mathbf{D}(\hat{\boldsymbol{\beta}})\mathbf{A}\hat{\boldsymbol{F}} \in \mathbb{R}^p$.
- A linear branch taking the input $\hat{\boldsymbol{N}}$ and transforming it as $\tilde{\mathbf{A}}\hat{\boldsymbol{N}}$, where $\tilde{\mathbf{A}} \in \mathbb{R}^{p \times k'}$ is a learnable matrix. The output of this branch is acted upon by the matrix $\mathbf{D}$ to give $\tilde{\boldsymbol{\beta}}(\hat{\boldsymbol{\beta}}, \hat{\boldsymbol{N}}) = \mathbf{D}(\hat{\boldsymbol{\beta}})\tilde{\mathbf{A}}\hat{\boldsymbol{N}} \in \mathbb{R}^p$.
- A non-linear trunk taking the input $\boldsymbol{x} \in \Omega$, which gives the output $\boldsymbol{\tau}(\boldsymbol{x}) = (\tau_1(\boldsymbol{x}), \ldots, \tau_p(\boldsymbol{x}))^\top$, where each $\tau_i : \Omega \to \mathbb{R}$ is a trainable network.

Let V_τ be the space spanned by the trained trunk functions $\boldsymbol{\tau}$. Then the final VarMiON operator is given by (notice the analogy with (5.49))

$$\begin{aligned} \hat{\mathcal{N}} : \mathbb{R}^k \times \mathbb{R}^{k'} \times \mathbb{R}^k \to V_\tau \\ \hat{\mathcal{N}}(\hat{\boldsymbol{F}}, \hat{\boldsymbol{N}}, \hat{\boldsymbol{\beta}}) = \hat{u}(\cdot, \hat{\boldsymbol{F}}, \hat{\boldsymbol{N}}, \hat{\boldsymbol{\beta}}) = \mathbf{D}(\hat{\boldsymbol{\beta}})(\mathbf{A}\hat{\boldsymbol{F}} + \tilde{\mathbf{A}}\hat{\boldsymbol{N}}) \\ = (\boldsymbol{\beta}(\hat{\boldsymbol{\beta}}, \hat{\boldsymbol{F}}) + \tilde{\boldsymbol{\beta}}(\hat{\boldsymbol{\beta}}, \hat{\boldsymbol{N}}))^\top \boldsymbol{\tau}. \end{aligned} \tag{5.55}$$

The discrete operator $\mathcal{N}_h$ is used to generate samples to train the VarMiON in the following manner:

1. For $1 \leq j \leq J$, consider distinct samples $(f^j, \eta^j, \beta^j) \in X$.
2. Use the projector $\mathcal{P}$ to obtain the discrete approximations $(f_h^j, \eta_h^j, \beta_h^j) \in X_h$.
3. Find the discrete numerical solution $u_h^j = \mathcal{N}_h(f_h^j, \eta_h^j, \beta_h^j)$.
4. Use the $\hat{\mathcal{P}}$ to generate the VarMiON input vectors $(\hat{\boldsymbol{F}}^j, \hat{\boldsymbol{N}}^j, \hat{\boldsymbol{\beta}}^j)$.
5. Select a set of output nodes $\{\boldsymbol{x}_l\}_{l=1}^L$. For each $1 \leq j \leq J$, sample the numerical solution at these nodes as $u_h^j(\boldsymbol{x}_l)$.
6. Finally, collect all the inputs and output labels to form the training set containing $J \times L$ samples

$$S = \{(\hat{\boldsymbol{F}}^j, \hat{\boldsymbol{N}}^j, \hat{\boldsymbol{\beta}}^j, \boldsymbol{x}_l, u_h^j(\boldsymbol{x}_l)) : 1 \leq j \leq J, 1 \leq l \leq L\} \tag{5.56}$$

where $\hat{\boldsymbol{F}}^j, \hat{\boldsymbol{N}}^j, \hat{\boldsymbol{\beta}}^j$ are the inputs to the branch subnets, $\boldsymbol{x}_l$ is the trunk input, and $u_h^j(\boldsymbol{x}_l)$ is the target output.

We also denote the loss function which needs to be optimized to train the VarMiON: the L_2 norm of the difference between the solution and the prediction computed on the output nodes $\boldsymbol{x}_l$

$$\sum_{j=1}^{J} \int_\Omega (u_h^j(\boldsymbol{x}) - \hat{\mathcal{N}}_{\boldsymbol{\theta}}(\hat{\boldsymbol{F}}^j, \hat{\boldsymbol{N}}^j, \hat{\boldsymbol{\beta}}^j)(\boldsymbol{x}))^2 \, d\boldsymbol{x}. \tag{5.57}$$

First, let $\{\omega_l\}_{l=1}^L$ be positive weights corresponding to the output nodes $\{\boldsymbol{x}_l\}_{l=1}^L$ used to describe the following quadrature for the square of a function $\xi : \Omega \to \mathbb{R}$

$$\left| \sum_{l=1}^{L} \omega_l \xi^2(\boldsymbol{x}_l) - \int_\Omega \xi^2(\boldsymbol{x}_l) d\boldsymbol{x} \right| \leq \frac{C(\xi)}{L^\gamma} \tag{5.58}$$

where the constant C depends on ξ, and γ is the rate of convergence that depends on the quadrature rule used.

By using the notation presented above for the VarMiON where $\boldsymbol{\theta}$ is the vector of trainable parameters we have

$$\Pi(\boldsymbol{\theta}) = \frac{1}{J} \sum_{j=1}^{J} \Pi_j(\boldsymbol{\theta}), \qquad \Pi_j(\boldsymbol{\theta}) = \sum_{l=1}^{L} \omega_l (u_h^j(\boldsymbol{x}_l) - \hat{\mathcal{N}}_{\boldsymbol{\theta}}(\hat{\boldsymbol{F}}^j, \hat{\boldsymbol{N}}^j, \hat{\boldsymbol{\beta}}^j)(\boldsymbol{x}_l))^2. \tag{5.59}$$

Notice that $\Pi_j(\boldsymbol{\theta}) \approx \|\mathcal{N}_h \circ \mathcal{P}(f^j, \eta^j, \beta^j) - \hat{\mathcal{N}}_{\boldsymbol{\theta}} \circ \hat{\mathcal{P}}(f^j, \eta^j, \beta^j)\|$. Then training the VarMiON corresponds to solving the following optimization problem

$$\boldsymbol{\theta}^* = \operatorname*{argmin}_{\boldsymbol{\theta}} \Pi(\boldsymbol{\theta}), \tag{5.60}$$

which is typically solved using iterative algorithms such as stochastic gradient descent or Adam [36].

5.4.2 VarMiON for time-dependent heat equation

In this section, we want to extend the previous operator network, to a time-dependent heat equation defined as

$$C(\boldsymbol{x})\frac{\partial}{\partial t}u(\boldsymbol{x},t) - \operatorname{div}(\beta(\boldsymbol{x})\nabla u(\boldsymbol{x},t)) = f(\boldsymbol{x},t) \qquad (\boldsymbol{x},t) \in \Omega \times [0,\tau], \tag{5.61}$$

$$\beta(\boldsymbol{x})\nabla u(\boldsymbol{x},t)\cdot \boldsymbol{n} = \eta(\boldsymbol{x},t) \qquad (\boldsymbol{x},t) \in \Gamma_N \times [0,\tau], \tag{5.62}$$

$$u(\boldsymbol{x},t) = g(\boldsymbol{x},t) \qquad (\boldsymbol{x},t) \in \Gamma_D \times [0,\tau], \tag{5.63}$$

$$u(\boldsymbol{x},0) = u_0(\boldsymbol{x}) \qquad (\boldsymbol{x},t) \in \Omega \times \{0\}, \tag{5.64}$$

for the temperature field $u : \Omega \times [0,\tau] \to \mathbb{R}$ where $u \in L^2([0,\tau]; H^1_D(\Omega))$. For the sake of simplicity we consider, from now on, the homogeneous Dirichlet boundary conditions.

Space-time formulation

To take advantage of the structure of VarMiON presented for the static equations we work with the weak space-time formulation that treats the space and time variables in the same way by discretizing them simultaneously (see e.g. [37–39] for some references on space-time formulation).

Let us consider the domain $\Omega \times [0,\tau]$ and write the weak space-time form as

$$\int_\Omega \int_0^\tau C(\boldsymbol{x})\frac{\partial u(\boldsymbol{x},t)}{\partial t}\tilde{u}(\boldsymbol{x},t)dtd\boldsymbol{x} - \int_\Omega \int_0^\tau \operatorname{div}(\beta(\boldsymbol{x})\nabla u(\boldsymbol{x},t))\tilde{u}(\boldsymbol{x},t)dtd\boldsymbol{x} = \int_\Omega \int_0^\tau f(\boldsymbol{x},t)\tilde{u}(\boldsymbol{x},t)dtd\boldsymbol{x} \tag{5.65}$$

where $\tilde{u}(\boldsymbol{x},t) \in L^2((0,\tau]; H^1_D(\Omega))$ is a suitable test function. Therefore we want to find u such that the previous equation is satisfied for all $\tilde{u}$ and $u(\boldsymbol{x},0) = u_0(\boldsymbol{x})$.

Let us take V_h be a family of spaces that depends on a positive parameter h, such that $V_h \subset L^2([0,\tau]; H^1_D(\Omega))$, $\dim V_h = N_h < \infty$, thus any function can be expressed, using a space-time FEM, as

$$u_h(\boldsymbol{x},t) = \sum_{j=1}^{N_h} u_j\phi_j(\boldsymbol{x},t) = \boldsymbol{U}^\top \boldsymbol{\Phi}(\boldsymbol{x},t) \tag{5.66}$$

where now, each $\phi_j \in V_h$ depends on the space-time variables

$$\boldsymbol{\Phi}(\boldsymbol{x},t) = (\phi_1(\boldsymbol{x},t),\dots,\phi_{N_h}(\boldsymbol{x},t))^\top, \tag{5.67}$$

$$\boldsymbol{U} = (u_1,\dots,u_{N_h})^\top. \tag{5.68}$$

Therefore the weak formulation for any test function $\phi_i \in V_h$ with $t > 0$ reads

$$\sum_{j=0}^{N_h} u_j \int_\Omega \int_0^\tau C(\boldsymbol{x}) \frac{\partial \phi_j(\boldsymbol{x},t)}{\partial t} \phi_i(\boldsymbol{x},t) dt d\boldsymbol{x} + \sum_{j=0}^{N_h} u_j \int_\Omega \beta(\boldsymbol{x}) \int_0^\tau \nabla \phi_j(\boldsymbol{x},t) \nabla \phi_i(\boldsymbol{x},t) dt d\boldsymbol{x} = + \int_0^\tau \int_{\partial\Omega} \eta(\boldsymbol{x},t) \phi_i(\boldsymbol{x},t) dt ds + \int_\Omega \int_0^\tau f(\boldsymbol{x},t) \phi_i(\boldsymbol{x},t) dt d\boldsymbol{x} \tag{5.69}$$

By following the same notation as in the previous section, we define the associated solution operator on X as

$$\mathcal{N} : X \to V \qquad \mathcal{N}(f, \eta, C, \beta, u_0) = u(\cdot, \cdot, f, \eta, C, \beta, u_0). \tag{5.70}$$

and we consider the projector to X_h to approximate the PDE data

$$\mathcal{P} : X \to X_h \qquad \mathcal{P}(f, \eta, C, \beta, u_0) = (f_h, \eta_h, C_h, \beta_h, u_{0_h}). \tag{5.71}$$

As before, the discrete weak formulation in matrix form reads:

$$(\mathbf{W}(C_h) + \mathbf{K}(\beta_h))\boldsymbol{U} = \mathbf{M}\boldsymbol{F} + \tilde{\mathbf{M}}\boldsymbol{N}, \tag{5.72}$$

for which holds the initial condition $\boldsymbol{U}^\top \boldsymbol{\Phi}(\boldsymbol{x}, 0) = \boldsymbol{U}_0^\top \boldsymbol{\Phi}(\boldsymbol{x}, 0)$ thus, if we select the nodes n_0 (at $t = 0$ by taking the rows and columns of matrices with $[:, : n_0]$) where holds the initial condition, we get

$$(\mathbf{W}(C_h) + \mathbf{K}(\beta_h))[:, : n_0]\boldsymbol{U}[: n_0] = (\mathbf{W}(C_h) + \mathbf{K}(\beta_h))[:, : n_0]\boldsymbol{U}_0 \tag{5.73}$$

$$(\mathbf{W}(C_h) + \mathbf{K}(\beta_h))[:, n_0 :]\boldsymbol{U}[n_0 :] = \mathbf{M}\boldsymbol{F} + \tilde{\mathbf{M}}\boldsymbol{N} - (\mathbf{W}(C_h) + \mathbf{K}(\beta_h))[:, : n_0]\boldsymbol{U}_0 \tag{5.74}$$

$$\boldsymbol{U}[n_0 :] = (\mathbf{W}(C_h)\mathbf{K}(\beta_h))[:, n_0 :]^{-1}(\mathbf{M}\boldsymbol{F} + \tilde{\mathbf{M}}\boldsymbol{N} - (\mathbf{W}(C_h) + \mathbf{K}(\beta_h))[:, : n_0]\boldsymbol{U}_0) \tag{5.75}$$

where

$$\mathbf{W}_{ji}(C_h) = (C_h \frac{\partial}{\partial t}\phi_j, \phi_i), \quad \mathbf{K}_{ji}(\beta_h) = (\beta_h \nabla \phi_j, \nabla \phi_i), \tag{5.76}$$
$$\mathbf{M}_{ji} = (\phi_j, \phi_i), \quad \tilde{\mathbf{M}}_{ji} = (\phi_j, \phi_i)|_{\Gamma_N}.$$

Notice that, $(\mathbf{W}(C_h) + \mathbf{K}(\beta_h))[:, n_0 :]$ is a square matrix and the basis $\{\phi_j(\boldsymbol{x}, t)\}_{j=1}^{N_h}$ is chosen such that the $(\mathbf{W} + \mathbf{K})[:, n_0 :]$ is invertible for any C_h and β_h. The discrete solution operator for $t > 0$ is defined as:

$$\mathcal{N}_h : X_h \to V_h$$
$$\mathcal{N}_h(f_h, \eta_h, C_h, \beta_h, u_{0_h}) = u_h(\cdot, \cdot, f_h, \eta_h, C_h, \beta_h, u_{0_h}) = (\mathbf{B}(f_h, C_h, \beta_h) + \tilde{\mathbf{B}}(\eta_h, C_h, \beta_h) + \bar{\mathbf{B}}(u_{0_h}, C_h, \beta_h))^\top \boldsymbol{\Phi} \tag{5.77}$$

where

$$\mathbf{B}(f_h, C_h, \beta_h) = (\mathbf{W}(C_h) + \mathbf{K}(\beta_h))[:, n_0 :]^{-1}(\mathbf{M}\boldsymbol{F}), \tag{5.78}$$

$$\tilde{\mathbf{B}}(\eta_h, C_h, \beta_h) = (\mathbf{W}(C_h) + \mathbf{K}(\beta_h))[:, n_0 :]^{-1}(\tilde{\mathbf{M}}\boldsymbol{N}), \tag{5.79}$$

$$\bar{\mathbf{B}}(u_{0_h}, C_h, \beta_h) = -(\mathbf{W}(C_h) + \mathbf{K}(\beta_h))[:, n_0 :]^{-1}((\mathbf{W}(C_h) + \mathbf{K}(\beta_h))[:, : n_0]\boldsymbol{U}_0) \tag{5.80}$$

$$\approx -(\mathbf{W}(C_h) + \mathbf{K}(\beta_h))[:, n_0 :]^{-1}(\bar{\mathbf{M}}\boldsymbol{U}_0). \tag{5.81}$$

Notice that, for the sake of simplicity, we approximate $(\mathbf{W}(C_h) + \mathbf{K}(\beta_h))[:, : n_0]$ with a matrix $\bar{\mathbf{M}}$ which, will lead to a non-linear branch in the final VarMiON operator.

We define, as done before, a data sensing operator $\hat{\mathcal{P}}$ to describe how to obtain the input data to be supplied to the neural network. Firstly, we choose the time sensor nodes $\{\hat{t}_i\}_{i=1}^r$ and then the space sensor nodes $\{\hat{\boldsymbol{x}}_i\}_{i=1}^k$. The PDE data $(f, \eta, C, \beta, u_0) \in X$ is fed into the VarMiON, with f sampled at the sensor nodes $\{(\hat{\boldsymbol{x}}_1, \hat{t}_j), \dots (\hat{\boldsymbol{x}}_k, \hat{t}_j)\}_{j=1}^r$ while η is sampled at the boundary sensor nodes $\{(\hat{\boldsymbol{x}}_1^b, \hat{t}_j), \dots (\hat{\boldsymbol{x}}_{k'}^b, \hat{t}_j)\}_{j=1}^r$ instead, u_0, C and β are sampled at $\{\hat{\boldsymbol{x}}_i\}_{i=1}^k$. We define the input vectors

$$\hat{\boldsymbol{F}} = (f(\hat{\boldsymbol{x}}_1, \hat{t}_1), \dots, f(\hat{\boldsymbol{x}}_k, \hat{t}_r))^\top \tag{5.82}$$

$$\hat{\boldsymbol{C}} = (C(\hat{\boldsymbol{x}}_1), \dots, C(\hat{\boldsymbol{x}}_k))^\top \tag{5.83}$$

$$\hat{\boldsymbol{\beta}} = (\beta(\hat{\boldsymbol{x}}_1), \dots, \beta(\hat{\boldsymbol{x}}_k))^\top \tag{5.84}$$

$$\hat{\boldsymbol{N}} = (\eta(\hat{\boldsymbol{x}}_1^b, \hat{t}_1), \dots, \eta(\hat{\boldsymbol{x}}_{k'}^b, \hat{t}_r))^\top \tag{5.85}$$

$$\hat{\boldsymbol{U}}_0 = (u_0(\hat{\boldsymbol{x}}_1), \dots, u_0(\hat{\boldsymbol{x}}_k))^\top \tag{5.86}$$

and the sensing operator

$$\hat{\mathcal{P}} : X \to \mathbb{R}^{kr} \times \mathbb{R}^{k'r} \times \mathbb{R}^k \times \mathbb{R}^k \times \mathbb{R}^k, \qquad \hat{\mathcal{P}}(f, \eta, C, \beta, u_0) \mapsto (\hat{\boldsymbol{F}}, \hat{\boldsymbol{N}}, \hat{\boldsymbol{C}}, \hat{\boldsymbol{\beta}}, \hat{\boldsymbol{U}}_0). \tag{5.87}$$

We can visualize the high-level implementation of the VarMiON for this problem in Figure 5.4, which comprises:

- A non-linear branch taking the inputs $\hat{\boldsymbol{C}}$ and $\hat{\boldsymbol{\beta}}$, which are transformed into a matrix output $\mathbf{D}(\hat{\boldsymbol{C}}, \hat{\boldsymbol{\beta}}) \in \mathbb{R}^{p \times p}$. Here p is the latent dimension of the VarMiON.

- A linear branch taking the input $\hat{\boldsymbol{F}}$ and transforming it as $\mathbf{A}\hat{\boldsymbol{F}}$, where $\mathbf{A} \in \mathbb{R}^{p \times kr}$ is a learnable matrix. The output of this branch is acted upon by the matrix $\mathbf{D}$ to give $\boldsymbol{\beta}(\hat{\boldsymbol{C}}, \hat{\boldsymbol{\beta}}, \hat{\boldsymbol{F}}) = \mathbf{D}(\hat{\boldsymbol{C}}, \hat{\boldsymbol{\beta}})\mathbf{A}\hat{\boldsymbol{F}} \in \mathbb{R}^p$.

- A linear branch taking the input $\hat{\boldsymbol{N}}$ and transforming it as $\tilde{\mathbf{A}}\hat{\boldsymbol{N}}$, where $\tilde{\mathbf{A}} \in \mathbb{R}^{p \times k'r}$ is a learnable matrix. The output of this branch is acted upon by the matrix $\mathbf{D}$ to give $\tilde{\boldsymbol{\beta}}(\hat{\boldsymbol{C}}, \hat{\boldsymbol{\beta}}, \hat{\boldsymbol{N}}) = \mathbf{D}(\hat{\boldsymbol{C}}, \hat{\boldsymbol{\beta}})\tilde{\mathbf{A}}\hat{\boldsymbol{N}} \in \mathbb{R}^p$.

- A non-linear branch, justified by the approximation done in (5.81), taking the input $\hat{\boldsymbol{U}}_0$ and transforming it as $\bar{\mathbf{A}}\hat{\boldsymbol{U}}_0$, where $\bar{\mathbf{A}} \in \mathbb{R}^{p \times k}$ is a learnable matrix. The output of this branch is acted upon by the matrix $\mathbf{D}$ to give $\bar{\boldsymbol{\beta}}(\hat{\boldsymbol{C}}, \hat{\boldsymbol{\beta}}, \hat{\boldsymbol{U}}_0) = \mathbf{D}(\hat{\boldsymbol{C}}, \hat{\boldsymbol{\beta}})\bar{\mathbf{A}}\hat{\boldsymbol{U}}_0 \in \mathbb{R}^p$.

- A non-linear trunk taking the input $(\boldsymbol{x}, t) \in \Omega \times (0, \tau]$, which gives the output $\boldsymbol{\tau}(\boldsymbol{x}, t) = (\tau_1(\boldsymbol{x}, t), \dots, \tau_p(\boldsymbol{x}, t))^\top$, where each $\tau_i : \Omega \times (0, \tau] \to \mathbb{R}$ is a trainable network.

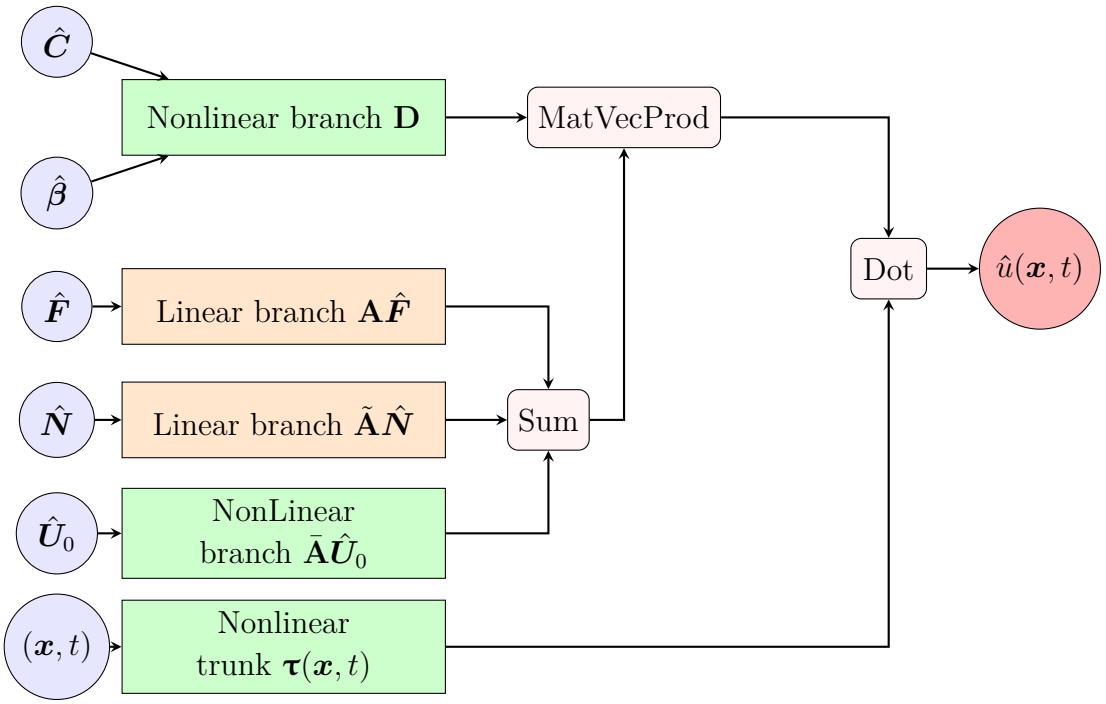

Figure 5.4: Schematic of a VarMiON for the time-dependent heat equation.

Let V_τ the space spanned by the trained trunk functions $\boldsymbol{\tau}$. The final VarMiON operator is given by

$$\begin{aligned}\hat{\mathcal{N}} &: \mathbb{R}^{kr} \times \mathbb{R}^{k'r} \times \mathbb{R}^k \times \mathbb{R}^k \times \mathbb{R}^k \to V_\tau \\ \hat{\mathcal{N}}(\hat{\boldsymbol{F}}, \hat{\boldsymbol{N}}, \hat{\boldsymbol{C}}, \hat{\boldsymbol{\beta}}, \hat{\boldsymbol{U}}_0) &= \hat{u}(\cdot,\cdot, \hat{\boldsymbol{F}}, \hat{\boldsymbol{N}}, \hat{\boldsymbol{C}}, \hat{\boldsymbol{\beta}}, \hat{\boldsymbol{U}}_0) = \mathbf{D}(\hat{\boldsymbol{C}}, \hat{\boldsymbol{\beta}})(\mathbf{A}\hat{\boldsymbol{F}} + \tilde{\mathbf{A}}\hat{\boldsymbol{N}} + \bar{\mathbf{A}}\hat{\boldsymbol{U}}_0) \\ &= (\boldsymbol{\beta}(\hat{\boldsymbol{C}}, \hat{\boldsymbol{\beta}}, \hat{\boldsymbol{F}}) + \tilde{\boldsymbol{\beta}}(\hat{\boldsymbol{C}}, \hat{\boldsymbol{\beta}}, \hat{\boldsymbol{N}}) + \bar{\boldsymbol{\beta}}(\hat{\boldsymbol{C}}, \hat{\boldsymbol{\beta}}, \hat{\boldsymbol{U}}_0))^\top \boldsymbol{\tau}. \quad (5.88)\end{aligned}$$

As in the previous section, we now wish to define a custom loss function that exploits the underlying physics; the main idea is unchanged, but we should point out some differences that occur since we are dealing with a time-dependent problem and different input functions. Samples now have the form

$$S = \{(\hat{\boldsymbol{F}}^j, \hat{\boldsymbol{N}}^j, \hat{\boldsymbol{C}}^j, \hat{\boldsymbol{\beta}}^j, \hat{\boldsymbol{U}}_0^j, t_i, \boldsymbol{x}_l, u_h^j(\boldsymbol{x}_l, t_i)) \ : 1 \le j \le J,\ 1 \le i \le M,\ 1 \le l \le L\}, \quad (5.89)$$

where $(\hat{\boldsymbol{F}}^j, \hat{\boldsymbol{N}}^j, \hat{\boldsymbol{C}}^j, \hat{\boldsymbol{\beta}}^j, \hat{\boldsymbol{U}}_0^j)$ are the inputs to the branch subnets, $(\boldsymbol{x}_l, t_i)$ is the trunk input, and $u_h^j(\boldsymbol{x}_l, t_i)$ the target output.

Now we look for parameters that minimize the sum of the L_2 norm of the error between the numerical solution u_h^j and the model predictions $\hat{u}^j = \hat{\mathcal{N}}(\hat{\boldsymbol{F}}^j, \hat{\boldsymbol{N}}^j, \hat{\boldsymbol{C}}^j, \hat{\boldsymbol{\beta}}^j, \hat{\boldsymbol{U}}_0^j)$

$$\sum_{j=1}^{J} \int_0^\tau \int_\Omega (u_h^j(\boldsymbol{x}, t) - \hat{\mathcal{N}}_{\boldsymbol{\theta}}(\hat{\boldsymbol{F}}^j, \hat{\boldsymbol{N}}^j, \hat{\boldsymbol{C}}^j, \hat{\boldsymbol{\beta}}^j, \hat{\boldsymbol{U}}_0^j)(\boldsymbol{x}, t))^2 \, d\boldsymbol{x} \, dt. \quad (5.90)$$

We choose $\{\omega_{li} : 1 \leq l \leq L, 1 \leq i \leq M\} \subset \mathbb{R}$ (positive) weights, corresponding to the nodes $\{(\boldsymbol{x}_l, t_i)\}_{l,i}$, for the numerical integration of the square of a function on $(0, \tau] \times \Omega$, i.e. such that for a function $\xi : (0, \tau] \times \Omega \to \mathbb{R}$:

$$\left| \sum_{l=1}^{L} \sum_{i=1}^{M} \omega_{li} \xi^2(\boldsymbol{x}_l, t_i) - \int_0^\tau \int_\Omega \xi^2(\boldsymbol{x}, t) dt d\boldsymbol{x} \right| \leq \frac{C(\xi)}{L^\gamma} \tag{5.91}$$

where the constant C depends on ξ and, γ is the rate of convergence that depends on the quadrature rule used.

By using the notation presented above for the VarMiON where $\boldsymbol{\theta}$ is the vector of trainable parameters we have

$$\Pi(\boldsymbol{\theta}) = \frac{1}{J} \sum_{j=1}^{J} \Pi_j(\boldsymbol{\theta}), \qquad \Pi_j(\boldsymbol{\theta}) = \sum_{l=1}^{L} \sum_{i=1}^{M} \omega_{li} (u_h^j(\boldsymbol{x}_l, t_i) - \hat{\mathcal{N}}_{\boldsymbol{\theta}}(\hat{\boldsymbol{F}}^j, \hat{\boldsymbol{N}}^j, \hat{\boldsymbol{C}}^j, \hat{\boldsymbol{\beta}}^j, \hat{\boldsymbol{U}}_0^j)(\boldsymbol{x}_l, t_i))^2. \tag{5.92}$$

Notice that $\Pi_j(\boldsymbol{\theta}) \approx \|\mathcal{N}_h \circ \mathcal{P}(f^j, \eta^j, C^j, \beta^j, u_0^j) - \hat{\mathcal{N}}_{\boldsymbol{\theta}} \circ \hat{\mathcal{P}}(f^j, \eta^j, C^j, \beta^j, u_0^j)\|$ and, training the VarMiON corresponds to solving the optimization problem $\boldsymbol{\theta}^* = \underset{\boldsymbol{\theta}}{\operatorname{argmin}}\, \Pi(\boldsymbol{\theta})$.

Learning process and numerical results

We train the network for the time-dependent heat conduction problem (5.61) described in the previous section with *zero* Dirichlet boundary conditions where the spatial domain Ω is a box of unitary length and the time interval is $[0, \tau] = [0, 1]$. The train is performed in Python, using the libraries NumPy, mpi4py and SymPy; in particular, for the neural network, we use PyTorch and FEniCSx for the solution of PDEs.

In our experiment, we partition the dataset into the following parts: training, validation and, testing set. We then split the former into k equally sized subsets on which we perform cross-validation to train the network, and finally, we use the testing set to check the performances on unknown data. In the cross-validation mechanism we successively choose one of the k subsets as validation set and the remaining $k-1$ as training set for a number of times, say fold, then for a number of epochs, we shuffle the training set. In our experiment, we fix 10 epochs and 15 folds.

Notice that the learning procedure consists in solving a sequence of nonlinear least squares, formulated with the loss function. To minimize the loss function we adopt a first-order gradient-based optimization algorithm, Adam algorithm [36], that uses adaptive learning rates for different parameters. To compute efficiently the gradients of the loss function with respect to the neural network parameters, we use back-propagation. This algorithm computes the gradients proceeding backward through the layers and applies the chain rule avoiding redundant computations.

- For the learning we, first generate the dataset

$$S = \{(\hat{\boldsymbol{F}}^j, \hat{\boldsymbol{N}}^j, \hat{\boldsymbol{C}}^j, \hat{\boldsymbol{\beta}}^j, \hat{\boldsymbol{U}}_0^j, t_i, \boldsymbol{x}_l, u_h^j(\boldsymbol{x}_l, t_i)) \, : 1 \leq j \leq J, 1 \leq l \leq L, 1 \leq i \leq M\} \tag{5.93}$$

  where $L = 10^3$, $M = 5$ and $J = 400$ or $J = 1600$.

- We choose f, η, C, β and u_0 to be Gaussian random fields and create J realizations: $(f_h^j, \eta_h^j, C_j^h, \beta_h^j, u_{0_h}^j)$ with $j = 1, \ldots, J$.

- Each realization $(f_h^j, \eta_h^j, C_j^h, \beta_h^j, u_{0_h}^j)$ is used to solve numerically with FEniCSx an instance of the heat equation.

- We evaluate each $(f_h^j, \eta_h^j, C_j^h, \beta_h^j, u_{0_h}^j)$ on the sensor nodes $\{(t_i, \boldsymbol{x}_l)\}_{i,l}$ and we finally get the samples: $(\hat{\boldsymbol{F}}^j, \hat{\boldsymbol{N}}^j, \hat{\boldsymbol{C}}^j, \hat{\boldsymbol{\beta}}^j, \hat{\boldsymbol{U}}_0^j, (t_i, \boldsymbol{x}_l), u_h^j(t_i, \boldsymbol{x}_l))$, with $i = 1, \ldots, M, j = 1, \ldots, J$ and $l = 1, \ldots, L$.

Notice that the performance results, plotted here, are not relative to the model obtained at the end of the training phase but, they refer to the 'best' model between those generated during training. Indeed, at the end of each epoch, we save the trained model, i.e. the model's parameters $\boldsymbol{\theta}$, so that once the training phase is completed we have the sequence $(\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_{\text{n\_epochs}})$. Now, to determine the best model we compute, for each set of parameters $\boldsymbol{\theta}_i$, the average relative L_2 error on the validation dataset and then we select the model relative to the epoch with the lowest error.

The average relative L_2 error to test the performance of the models between each testing field u_h^j and its prediction $\hat{u}^j = \hat{\mathcal{N}}_{\boldsymbol{\theta}}(\hat{\boldsymbol{F}}^j, \hat{\boldsymbol{N}}^j, \hat{\boldsymbol{C}}^j, \hat{\boldsymbol{\beta}}^j, \hat{\boldsymbol{U}}_0^j)$ is

$$\frac{\int_0^T \int_\Omega (\hat{u}^j(t, \boldsymbol{x}) - u_h^j(t, \boldsymbol{x}))^2}{\int_0^T \int_\Omega (u_h^j(t, \boldsymbol{x}))^2} \approx \frac{\sum_{i=1}^M \sum_{l=1}^L w_{il} (\hat{u}^j(t_i, \boldsymbol{x}_l) - u_h^j(t_i, \boldsymbol{x}_l))^2}{\sum_{i=1}^M \sum_{l=1}^L w_{il} (u_h^j(t_i, \boldsymbol{x}_l))^2}. \tag{5.94}$$

In Figure 5.5 we report the trend of the loss function for the time-dependent heat equation trained with VarMiON. From these graphs, we can notice a decrease in the error for both the training and validation phases, clearly a bigger dataset (b) reduces the loss in the initial phase. In Figure 5.6 and in Figure 5.7 we can see the average relative error probability and, the distribution of individual errors respectively, performed on the part of the dataset designated for the testing phase. By looking at these plots we can see that the relative error seems to be located more on the left portion of the plots, especially for the bigger dataset (b).

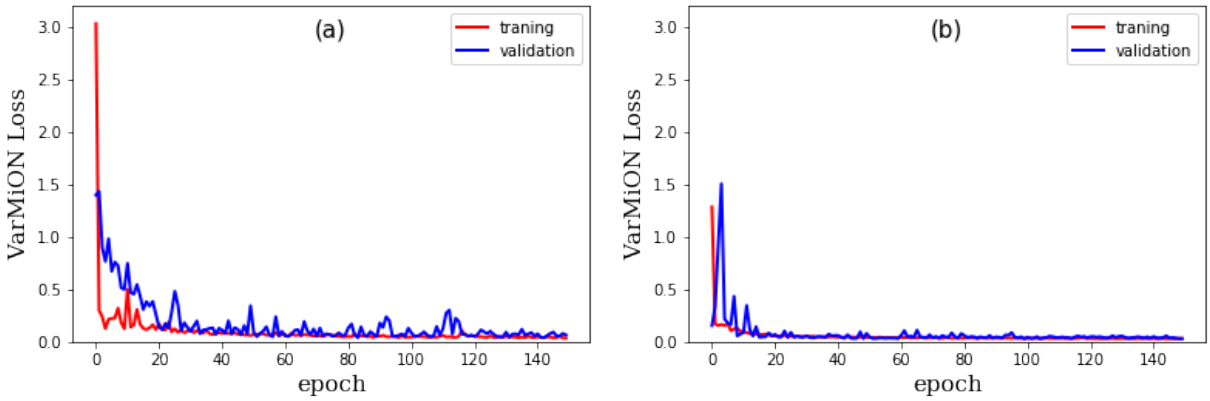

Figure 5.5: Loss error for the time-dependent heat equation with a dataset of 400 instances (a) and 1600 instances (b).

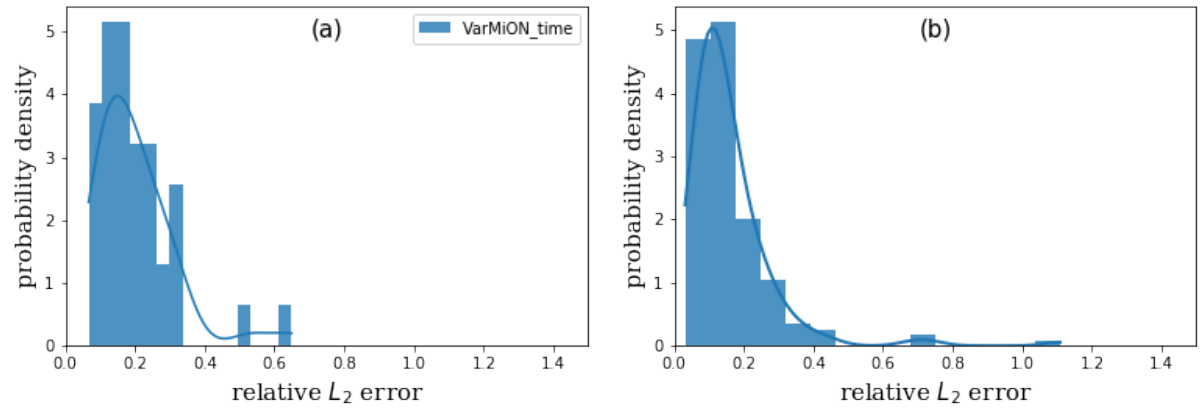

Figure 5.6: Error probability density for the time-dependent heat equation with a dataset of 400 instances (a) and 1600 instances (b).

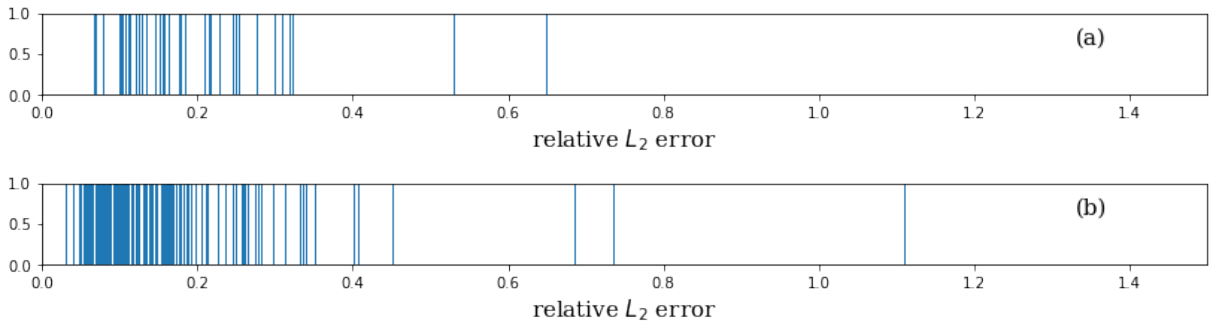

Figure 5.7: Relative error distribution for the time-dependent heat equation with a dataset of 400 instances (a) and 1600 instances (b).

5.4.3 VarMiON with Robin's boundary conditions

Let now consider a VarMiON for the time-dependent heat equation with Robin's boundary condition, that is

$$\frac{\partial}{\partial t}u(\boldsymbol{x},t) - \operatorname{div}(\beta(\boldsymbol{x})\nabla u(\boldsymbol{x},t)) = f(\boldsymbol{x},t) \qquad (\boldsymbol{x},t) \in \Omega \times [0,\tau], \tag{5.95}$$

$$\beta(\boldsymbol{x})\nabla u(\boldsymbol{x},t)\cdot \boldsymbol{n} = \alpha(u(\boldsymbol{x},t) - \eta(\boldsymbol{x},t)) \qquad (\boldsymbol{x},t) \in \partial\Omega \times [0,\tau], \tag{5.96}$$

$$u(\boldsymbol{x},0) = u_0(\boldsymbol{x}) \qquad (\boldsymbol{x},t) \in \Omega \times \{0\}. \tag{5.97}$$

Notice that, the structure of the network is the same as the previous section. Thus, for the sake of simplicity, in this part, we only itemize the operators useful to define a VarMiON

- The associated solution operator

$$\mathcal{N} : X \to V \qquad \mathcal{N}(f,\eta,\beta,u_0,\alpha) = u(\cdot,\cdot,f,\eta,\beta,u_0,\alpha). \tag{5.98}$$

- The projector to X_h to approximate the PDE data

$$\mathcal{P} : X \to X_h \qquad \mathcal{P}(f,\eta,\beta,u_0,\alpha) = (f_h,\eta_h,\beta_h,u_{0_h},\alpha) \tag{5.99}$$

  and, the discrete weak formulation

$$(\mathbf{W} + \mathbf{K}(\beta_h) - \alpha\tilde{\mathbf{M}})\boldsymbol{U} = (\mathbf{M}\boldsymbol{F} - \alpha\tilde{\mathbf{M}}\boldsymbol{N}), \tag{5.100}$$

  for which holds the initial condition $\boldsymbol{U}^\top\boldsymbol{\Phi}(\boldsymbol{x},0) = \boldsymbol{U}_0^\top\boldsymbol{\Phi}(\boldsymbol{x},0)$.

- The discrete solution operator

$$\begin{aligned}\mathcal{N}_h : X_h \to V_h, \quad \mathcal{N}_h(f_h,\eta_h,\beta_h,u_{0_h},\alpha) &= u_h(\cdot,\cdot,f_h,\eta_h,\beta_h,u_{0_h},\alpha)\\ &= (\mathbf{B}(f_h,\beta_h,\alpha) + \tilde{\mathbf{B}}(\eta_h,\beta_h,\alpha) + \bar{\mathbf{B}}(u_{0_h},\beta_h,\alpha))^\top\boldsymbol{\Phi}\end{aligned} \tag{5.101}$$

  where

$$\mathbf{B}(f_h,\beta_h,\alpha) = (\mathbf{W} + \mathbf{K}(\beta_h) - \alpha\tilde{\mathbf{M}})[:,n_0:]^{-1}(\mathbf{M}\boldsymbol{F}) \tag{5.102}$$

$$\tilde{\mathbf{B}}(\eta_h,\beta_h,\alpha) = (\mathbf{W} + \mathbf{K}(\beta_h) - \alpha\tilde{\mathbf{M}})[:,n_0:]^{-1}(-\alpha\tilde{\mathbf{M}}\boldsymbol{N}) \tag{5.103}$$

$$\begin{aligned}\bar{\mathbf{B}}(u_{0_h},\beta_h,\alpha) &= -(\mathbf{W} + \mathbf{K}(\beta_h) - \alpha\tilde{\mathbf{M}})[:,n_0:]^{-1}((\mathbf{W} + \mathbf{K}(\beta_h) - \alpha\tilde{\mathbf{M}})[:,:n_0]\boldsymbol{U}_0)\\ &\approx -(\mathbf{W} + \mathbf{K}(\beta_h) - \alpha\tilde{\mathbf{M}})[:,n_0:]^{-1}(\bar{\mathbf{M}}\boldsymbol{U}_0).\end{aligned} \tag{5.104}$$

- The data sensing operator

$$\hat{\mathcal{P}} : X \to \mathbb{R}^{kr} \times \mathbb{R}^{k'r} \times \mathbb{R}^k \times \mathbb{R}^k \times \mathbb{R}, \qquad \hat{\mathcal{P}}(f,\eta,\beta,u_0,\alpha) \mapsto (\hat{\boldsymbol{F}},\hat{\boldsymbol{N}},\hat{\boldsymbol{\beta}},\hat{\boldsymbol{U}}_0,\hat{\alpha}). \tag{5.105}$$

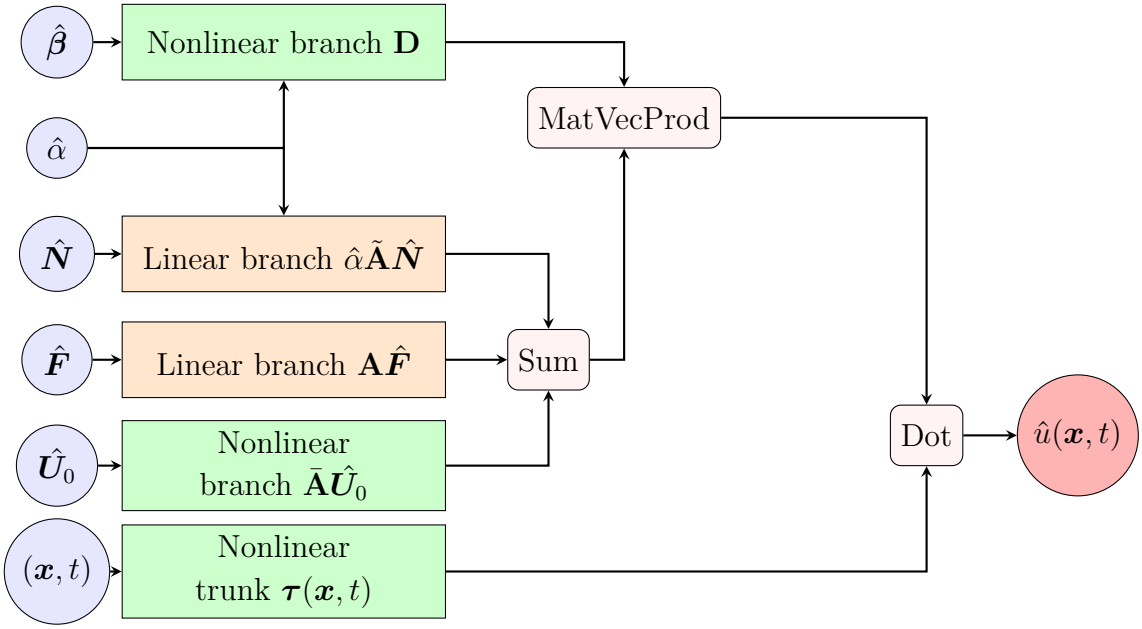

Figure 5.8: Schematic of a VarMiON for the time-dependent heat equation with Robin's boundary conditions.

- The final VarMiON operator

$$\begin{aligned}\hat{\mathcal{N}} &: \mathbb{R}^{kr} \times \mathbb{R}^{k'r} \times \mathbb{R}^k \times \mathbb{R}^k \times \mathbb{R} \to V_\tau \\ \hat{\mathcal{N}}(\hat{\boldsymbol{F}}, \hat{\boldsymbol{N}}, \hat{\boldsymbol{\beta}}, \hat{\boldsymbol{U}}_0, \hat{\alpha}) &= \hat{u}(\cdot,\cdot, \hat{\boldsymbol{F}}, \hat{\boldsymbol{N}}, \hat{\boldsymbol{\beta}}, \hat{\boldsymbol{U}}_0, \hat{\alpha}) = \mathbf{D}(\hat{\alpha}, \hat{\boldsymbol{\beta}})(\mathbf{A}\hat{\boldsymbol{F}} + \hat{\alpha}\tilde{\mathbf{A}}\hat{\boldsymbol{N}} + \bar{\mathbf{A}}\hat{\boldsymbol{U}}_0) \\ &= (\boldsymbol{\beta}(\hat{\boldsymbol{\beta}}, \hat{\boldsymbol{F}}, \hat{\alpha}) + \tilde{\boldsymbol{\beta}}(\hat{\boldsymbol{\beta}}, \hat{\boldsymbol{N}}, \hat{\alpha}) + \bar{\boldsymbol{\beta}}(\hat{\boldsymbol{\beta}}, \hat{\boldsymbol{U}}_0, \hat{\alpha}))^\top \boldsymbol{\tau} \quad (5.106)\end{aligned}$$

We can visualize the high-level implementation of the VarMiON for this problem in Figure 5.8.

Collecting this data we obtain a dataset of $J \times L \times M$ samples:

$$S = \{(\hat{\boldsymbol{F}}^j, \hat{\boldsymbol{N}}^j, \hat{\boldsymbol{\beta}}^j, \hat{\boldsymbol{U}}_0^j, \hat{\alpha}^j, t_i, \boldsymbol{x}_l, u_h^j(\boldsymbol{x}_l, t_i)) : 1 \le j \le J,\ 1 \le l \le L,\ 1 \le i \le M\}. \quad (5.107)$$

The definition of the loss function is analogous to the previous section.

5.4.4 VarMiON with temporal discontinuity

Let us now see how the formulation changes when we admit temporal jumps across the time interval.

Using a sequence of discrete time levels $0 = t_0 < t_1 < \cdots < t_N = \tau$, the space–time domain is partitioned in N slabs $\Omega \times I_n$, with $I_n = (t_n, t_{n+1})$. Each slab is then discretized

with elements that include approximations in space and time. The functions are allowed to be discontinuous across the slabs. This leads to temporal jump terms

$$[\![u(\boldsymbol{x},t_n)]\!] := u(\boldsymbol{x},t_n^+) - u(\boldsymbol{x},t_n^-), \tag{5.108}$$

where

$$u(\boldsymbol{x},t_n^\pm) = \lim_{\varepsilon\to 0^+} u(\boldsymbol{x},t_n \pm \varepsilon). \tag{5.109}$$

Therefore, by focusing on integrals on $\Omega \times I_n$ for problem (5.61) where $C(\boldsymbol{x})$ is the identity, we get

$$\begin{aligned}\sum_{j=0}^{N_h} u_j \int_\Omega \int_{I_n} \frac{\partial}{\partial t}\phi_j(\boldsymbol{x},t)\phi_i(x,t)dtd\boldsymbol{x} + \int_\Omega [\![u(\boldsymbol{x},t_n)]\!]\phi_i(\boldsymbol{x},t_n^+)d\boldsymbol{x} \\ + \sum_{j=0}^{N_h} u_j \int_\Omega \beta(\boldsymbol{x}) \int_{I_n} \nabla\phi_j(\boldsymbol{x},t)\nabla\phi_i(\boldsymbol{x},t)dtd\boldsymbol{x} = \\ + \int_{I_n}\int_{\Gamma_N} \eta(\boldsymbol{x},t)\phi_i(\boldsymbol{x},t)dtds + \int_\Omega\int_{I_n} f(\boldsymbol{x},t)\phi_i(\boldsymbol{x},t)dtd\boldsymbol{x},\end{aligned} \tag{5.110}$$

Notice that, the second term of this equation reflects a weak continuity condition that is imposed on $t = t_n$ [39, 40]

$$\begin{aligned}\sum_{j=0}^{N_h} u_j \int_\Omega \int_{I_n} \frac{\partial}{\partial t}\phi_j(\boldsymbol{x},t)\phi_i(x,t)dtd\boldsymbol{x} + \int_\Omega \left[\sum_{j=0}^{N_h} u_j\phi_j(\boldsymbol{x},t_n^+) - u(\boldsymbol{x},t_n^-)\right]\phi_i(\boldsymbol{x},t_n^+)d\boldsymbol{x} \\ + \sum_{j=0}^{N_h} u_j \int_\Omega \beta(\boldsymbol{x}) \int_{I_n} \nabla\phi_j(\boldsymbol{x},t)\nabla\phi_i(\boldsymbol{x},t)dtd\boldsymbol{x} = \\ + \int_{I_n}\int_{\Gamma_N} \eta(\boldsymbol{x},t)\phi_i(\boldsymbol{x},t)dtds + \int_\Omega\int_{I_n} f(\boldsymbol{x},t)\phi_i(\boldsymbol{x},t)dtd\boldsymbol{x},\end{aligned} \tag{5.111}$$

$$\begin{aligned}\sum_{j=0}^{N_h} u_j \int_\Omega \int_{I_n} \frac{\partial}{\partial t}\phi_j(\boldsymbol{x},t)\phi_i(x,t)dtd\boldsymbol{x} + \sum_{j=0}^{N_h} u_j \int_\Omega \phi_j(\boldsymbol{x},t_n^+)\phi_i(\boldsymbol{x},t_n^+)d\boldsymbol{x} - \int_\Omega u(\boldsymbol{x},t_n^-)\phi_i(\boldsymbol{x},t_n^+)d\boldsymbol{x} \\ + \sum_{j=0}^{N_h} u_j \int_\Omega \beta(\boldsymbol{x}) \int_{I_n} \nabla\phi_j(\boldsymbol{x},t)\nabla\phi_i(\boldsymbol{x},t)dtd\boldsymbol{x} = \\ + \int_{I_n}\int_{\Gamma_N} \eta(\boldsymbol{x},t)\phi_i(\boldsymbol{x},t)dtds + \int_\Omega\int_{I_n} f(\boldsymbol{x},t)\phi_i(\boldsymbol{x},t)dtd\boldsymbol{x},\end{aligned} \tag{5.112}$$

where in the integral on $\Omega \times I_0$ we use $u(\boldsymbol{x},t_0^-) = u_0(\boldsymbol{x})$. For the sake of simplicity from now on, we denote $u(\boldsymbol{x},t_n^-)$ with $u_n^-(\boldsymbol{x})$.

By following the same procedure of the previous section, we define the solution operator in each $\Omega \times I_n$ which maps the data (f,η,β,u_n^-) to the unique solution $u(\cdot,\cdot,f,\eta,\beta,u_n^-)$ where u_n^- represents a sort of initial condition for the equation on $\Omega \times I_n$.

- The associated solution operator

$$\mathcal{N}(f,\eta,\beta,u_n^-) = u(\cdot,\cdot,f,\eta,\beta,u_n^-) \in L^2(I_n;H^1(\Omega)) \tag{5.113}$$

- The projector to X_h to approximate the PDE data

$$\mathcal{P} : X \to X_h \qquad \mathcal{P}(f, \eta, \beta, u_n^-) = (f_h, \eta_h, \beta_h, u_{n_h}^-), \tag{5.114}$$

and, the discrete weak formulation

$$\mathbf{W}\boldsymbol{U} + \bar{\mathbf{M}}\boldsymbol{U} - \bar{\mathbf{M}}\boldsymbol{U}_n^- + \mathbf{K}(\beta_h)\boldsymbol{U} = \mathbf{M}\boldsymbol{F} + \tilde{\mathbf{M}}\boldsymbol{N}, \tag{5.115}$$
$$\boldsymbol{U} = (\mathbf{W} + \bar{\mathbf{M}} + \mathbf{K}(\beta_h))^{-1}(\mathbf{M}\boldsymbol{F} + \tilde{\mathbf{M}}\boldsymbol{N} + \bar{\mathbf{M}}\boldsymbol{U}_n^-) \tag{5.116}$$

with $\mathbf{W} + \bar{\mathbf{M}} + \mathbf{K}(\beta_h)$ invertible for all β_h and where

$$\mathbf{K}_{ji}(\beta_h) = (\beta_h \nabla\phi_j, \nabla\phi_i)_{\Omega\times I_n}, \quad \mathbf{W} = (\frac{\partial}{\partial t}\phi_j, \phi_i)_{\Omega\times I_n}, \tag{5.117}$$
$$\bar{\mathbf{M}}_{ij} = (\phi_j, \phi_i)_{\Omega\times t_n^+}, \quad \mathbf{M}_{ij} = (\phi_j, \phi_i)_{\Omega\times I_n}, \quad \tilde{\mathbf{M}}_{ij} = (\phi_j, \phi_i)|_{\Gamma_N\times I_n} \tag{5.118}$$

- The discrete solution operator

$$\begin{aligned}\mathcal{N}_h(f_h, \eta_h, \beta_h, u_{n_h}^-) &= u_h(\cdot, \cdot, f_h, \eta_h, \beta_h, u_{n_h}^-) \\ &= (\mathbf{B}(f_h, \beta_h) + \tilde{\mathbf{B}}(\eta_h, \beta_h) + \bar{\mathbf{B}}(u_{n_h}^-, \beta_h))^\top \boldsymbol{\Phi}\end{aligned} \tag{5.119}$$

where

$$\mathbf{B}(f_h, \beta_h) = (\mathbf{W} + \bar{\mathbf{M}} + \mathbf{K}(\beta_h))^{-1}(\mathbf{M}\boldsymbol{F}), \tag{5.120}$$
$$\tilde{\mathbf{B}}(\eta_h, \beta_h) = (\mathbf{W} + \bar{\mathbf{M}} + \mathbf{K}(\beta_h))^{-1}(\tilde{\mathbf{M}}\boldsymbol{N}), \tag{5.121}$$
$$\bar{\mathbf{B}}(u_{n_h}^-, \beta_h) = (\mathbf{W} + \bar{\mathbf{M}} + \mathbf{K}(\beta_h))^{-1}(\bar{\mathbf{M}}\boldsymbol{U}_n^-). \tag{5.122}$$

- The data sensing operator $\hat{\mathcal{P}}$ as

$$\hat{\mathcal{P}} : X \to \mathbb{R}^{kr} \times \mathbb{R}^{k'r} \times \mathbb{R}^k \times \mathbb{R}^k, \qquad \hat{\mathcal{P}}(f, \eta, \beta, u_n^-) = (\hat{\boldsymbol{F}}, \hat{\boldsymbol{N}}, \hat{\boldsymbol{\beta}}, \hat{\boldsymbol{U}}_n^-). \tag{5.123}$$

- Then the final VarMiON operator

$$\begin{aligned}&\hat{\mathcal{N}} : \mathbb{R}^{kr} \times \mathbb{R}^{k'r} \times \mathbb{R}^k \times \mathbb{R}^k \to V_\tau \\ &\quad \hat{\mathcal{N}}(\hat{\boldsymbol{F}}, \hat{\boldsymbol{N}}, \hat{\boldsymbol{\beta}}, \hat{\boldsymbol{U}}_n^-) = \hat{u}(\cdot, \hat{\boldsymbol{F}}, \hat{\boldsymbol{N}}, \hat{\boldsymbol{\beta}}, \hat{\boldsymbol{U}}_n^-) = \mathbf{D}(\hat{\boldsymbol{\beta}})(\mathbf{A}\hat{\boldsymbol{F}} + \tilde{\mathbf{A}}\hat{\boldsymbol{N}} + \bar{\mathbf{A}}\hat{\boldsymbol{U}}_n^-) \\ &\qquad\qquad = (\boldsymbol{\beta}(\hat{\boldsymbol{\beta}}, \hat{\boldsymbol{F}}) + \tilde{\boldsymbol{\beta}}(\hat{\boldsymbol{\beta}}, \hat{\boldsymbol{N}}) + \bar{\boldsymbol{\beta}}(\hat{\boldsymbol{\beta}}, \hat{\boldsymbol{U}}_n^-))^\top \boldsymbol{\tau}\end{aligned} \tag{5.124}$$

We can visualize a schematic representation of VarMiON for this problem in Figure 5.9. The dataset and the definition of the loss follow the previous section. The learning of this VarMiON is the subject of future study and follows the article [3].

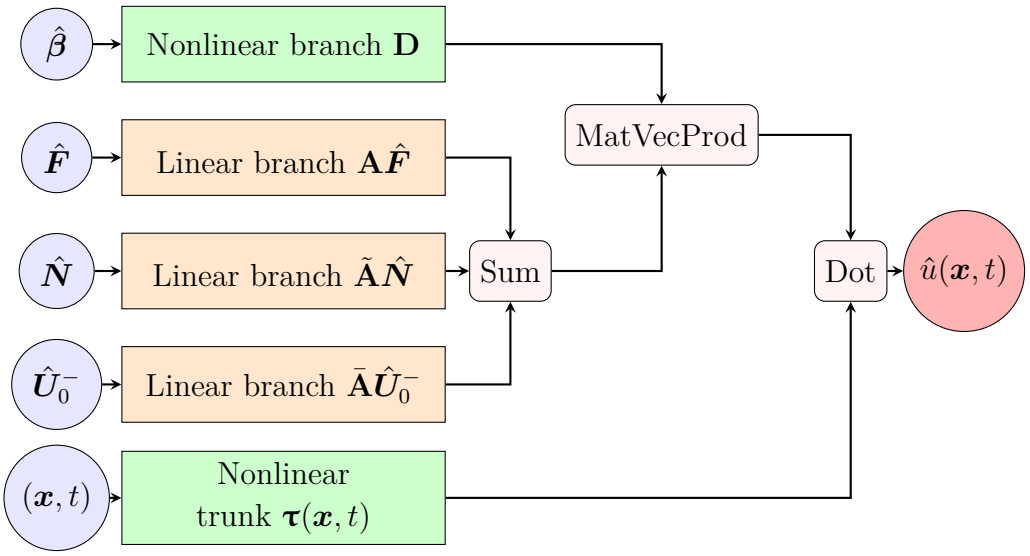

Figure 5.9: Schematic of a VarMiON for time-dependent heat equation in $\Omega \times I_0$.

5.5 Surrogate model of bread leavening

This section aims to build the surrogate model of bread leavening which mimics our numerical simulations and exploits the space-time VarMiON formulation for the equations involved. It is important to note that, the equations in the leavening problem are coupled with each other, as we have seen in Chapter 2 where the coupling is managed with a semi-implicit method - see Table 4.1, thus to maintain it, we divide the time domain $[0, \tau]$, where the dynamic takes place, into N intervals I_n. In this way, we define a surrogate model, by recursively focusing, on the sub-domain $\Omega \times I_n$ (sketch representation in Figure 5.10 with the main functions) where are defined three VarMiON networks, which predictions, computed at the sensor nodes $(\boldsymbol{x}, t_{n+1})$, will be used as input in $\Omega \times I_{n+1}$ allowing the connection between equations.

Notice that, we compute the discrete solution of (2.15) for yeast $Y_n = e^{K_{\mathrm{y}}(\theta_{\mathrm{b}_{n-1}})\Delta t} Y_{n-1}$ evaluated at suitable sensor nodes to feed the VarMiON CO_2 network.

The learning of this model consists of the training for the elasticity equation (linearized for simplicity) that predicts the placement $\boldsymbol{u}$ from which we compute J, $\mathbf{F}^{-\top}$ and $\mathbf{C}$ and the training of the time-varying equations. Especially, in each time sub-interval we proceed with the VarMiON net for the heat equation, then we go ahead with the VarMiON for the carbon dioxide equation by taking θ_{b} and Y from the dataset. Here, we want to train the three VarMiON networks independently of each other. Instead, in the testing and validation phases we want to use the prediction of each network as an input for the following one as in Figure 5.10.

Let us now see in detail, the structure of each network that composes the surrogate

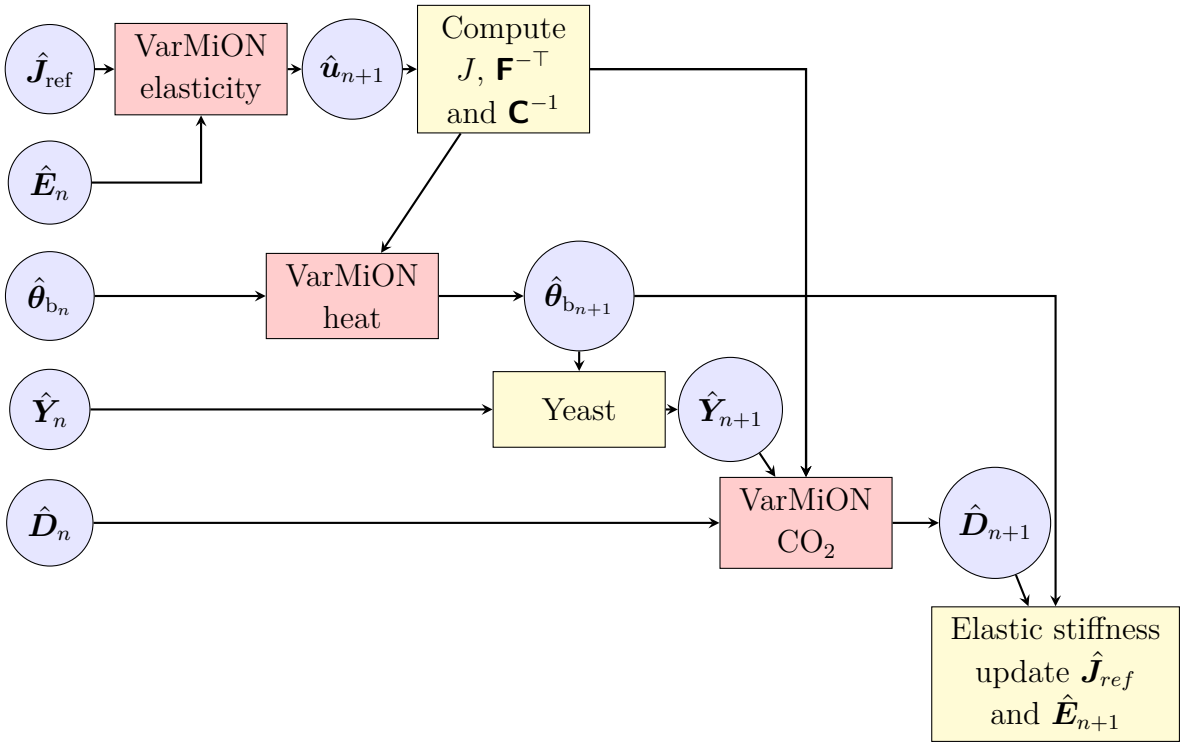

Figure 5.10: A sketch of surrogate bread leavening model on $\Omega \times I_n$. Only the main variables for bread leavening are represented here.

model represented with a red box in Figure 5.10.

5.5.1 VarMiON for elasticity equation

We start by recalling the elastic energy involved in our model of bread leavening (2.1)

$$\mathcal{E}(\boldsymbol{u}) = \frac{1}{2}\int_\Omega \left[\mu\left(\operatorname{tr}\mathbf{C} - 3J_{\text{ref}}^{2/3}\right) + \lambda|\log(J/J_{\text{ref}})|^2\right]\, d\boldsymbol{X} - \int_\Omega \boldsymbol{b}\cdot\boldsymbol{u}\, d\boldsymbol{X}$$

where, at each time, the deformation map $\boldsymbol{u}$ is obtained by solving the Euler–Lagrange equation $\delta\mathcal{E}(\boldsymbol{u}) = 0$, subject to boundary conditions:

$$\begin{aligned} &\boldsymbol{u} = (0,0,0) && \text{on } \Gamma_0,\\ &\boldsymbol{u} = (0,0,u_z) && \text{on } \Gamma_2. \end{aligned}$$

For the sake of simplicity, to reduce the order of the model, we consider the linearized elastic energy with respect to $\nabla(\boldsymbol{u}-\boldsymbol{u}_{\text{ref}})$ where $\boldsymbol{u}_{\text{ref}} = (0,0,(J_{\text{ref}}-1)z)$, thus

$$\sigma = [\mu\,(\nabla^s(\boldsymbol{u}-\boldsymbol{u}_{\text{ref}})) + \lambda\operatorname{tr}(\nabla(\boldsymbol{u}-\boldsymbol{u}_{\text{ref}}))\mathbb{1}] \tag{5.125}$$

where ∇^s is the symmetrized gradient. With this provision, we have

$$-\operatorname{div}(\sigma) = \boldsymbol{f} \qquad \text{on } \Omega \tag{5.126}$$

for $\boldsymbol{f} = \boldsymbol{b}$. Therefore the elasticity equation, written in the variational formulation, useful for the VarMiON formalism becomes

$$\int_\Omega [\mu\,(\nabla^s(\boldsymbol{u}-\boldsymbol{u}_{\text{ref}})) + \lambda\operatorname{tr}(\nabla(\boldsymbol{u}-\boldsymbol{u}_{\text{ref}}))\mathbb{1}] : \nabla\tilde{\boldsymbol{u}}\, d\boldsymbol{X} = \int_\Omega \boldsymbol{f}\cdot\tilde{\boldsymbol{u}}\, d\boldsymbol{X} \tag{5.127}$$

$$\int_\Omega [\mu\nabla^s(\boldsymbol{u}) - \mu\nabla^s(\boldsymbol{u}_{\text{ref}}) + \lambda\operatorname{tr}(\nabla(\boldsymbol{u}))\mathbb{1} - \lambda\operatorname{tr}(\nabla(\boldsymbol{u}_{\text{ref}}))\mathbb{1}] : \nabla\tilde{\boldsymbol{u}}\, d\boldsymbol{X} = \int_\Omega \boldsymbol{f}\cdot\tilde{\boldsymbol{u}}\, d\boldsymbol{X} \tag{5.128}$$

$$\begin{aligned} &\int_\Omega \mu\nabla^s(\boldsymbol{u}) : \nabla\tilde{\boldsymbol{u}}\, d\boldsymbol{X} + \int_\Omega \lambda\operatorname{tr}(\nabla(\boldsymbol{u}))\mathbb{1} : \nabla\tilde{\boldsymbol{u}}\, d\boldsymbol{X} =\\ &\int_\Omega \mu\nabla^s(\boldsymbol{u}_{\text{ref}}) : \nabla\tilde{\boldsymbol{u}}\, d\boldsymbol{X} + \int_\Omega \lambda\operatorname{tr}(\nabla(\boldsymbol{u}_{\text{ref}}))\mathbb{1} : \nabla\tilde{\boldsymbol{u}}\, d\boldsymbol{X} + \int_\Omega \boldsymbol{f}\cdot\tilde{\boldsymbol{u}}\, d\boldsymbol{X} \end{aligned} \tag{5.129}$$

for $\tilde{\boldsymbol{u}}$ suitable test function and where the previous boundary conditions are held.

Let us observe that, the reference tensor to which the material tries to tend, could not be the gradient of a tensor $\boldsymbol{u}_{\text{ref}}$, in that case, we postulate to have the gradient of the following form:

$$\nabla_{\text{ref}} = \begin{pmatrix} 0 & 0 & 0\\ 0 & 0 & 0\\ 0 & 0 & J_{\text{ref}}-1 \end{pmatrix}. \tag{5.130}$$

In addition, it is worth noticing that μ and λ are computed starting from the Young modulus E and Poisson's ratio ν instead, $\boldsymbol{u}_{\text{ref}}$ depends on J_{ref}. Let us remark that, this equation is static and the placement $\boldsymbol{u}$ is a vectorial function on Ω, thus the VarMiON needs

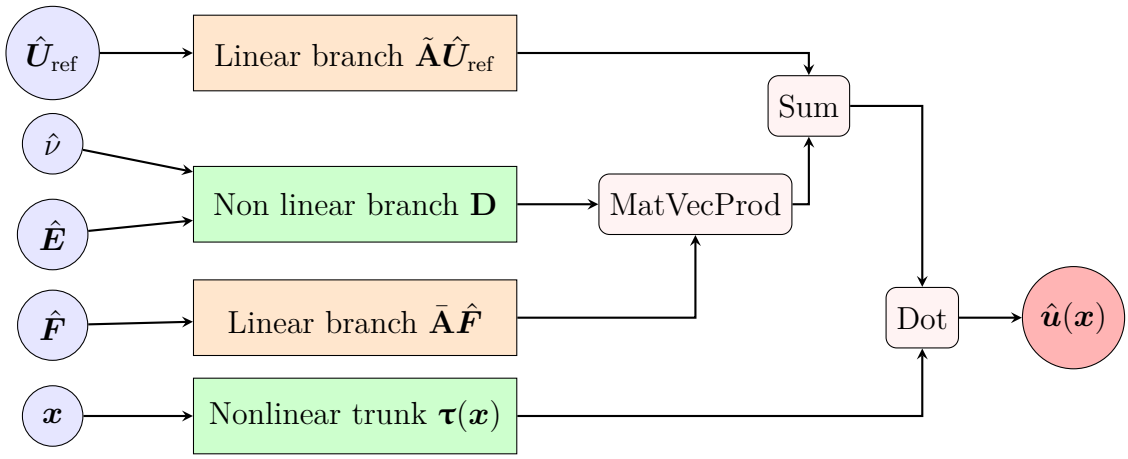

Figure 5.11: Schematic of VarMiON for the linearized elasticity equation in Ω.

- The discrete weak formulation

$$(\mathbf{K}(E,\nu)+\bar{\mathbf{K}}(E,\nu))\boldsymbol{U} = (\mathbf{K}(E,\nu)+\bar{\mathbf{K}}(E,\nu))\boldsymbol{U}_{\text{ref}} + \mathbf{M}\boldsymbol{F} \tag{5.131}$$

$$\boldsymbol{U} = (\mathbf{K}(E,\nu)+\bar{\mathbf{K}}(E,\nu))^{-1}(\mathbf{K}(E,\nu)+\bar{\mathbf{K}}(E,\nu))\boldsymbol{U}_{\text{ref}} + (\mathbf{K}(E,\nu)+\bar{\mathbf{K}}(E,\nu))^{-1}\mathbf{M}\boldsymbol{F} \tag{5.132}$$

$$\boldsymbol{U} = \mathbb{1}\boldsymbol{U}_{\text{ref}} + (\mathbf{K}(E,\nu)+\bar{\mathbf{K}}(E,\nu))^{-1}\mathbf{M}\boldsymbol{F} \tag{5.133}$$

- The discrete solution operator

$$\begin{aligned}&\mathcal{N}_h : X_h \to V_h,\\ &\mathcal{N}_h(E_h,\nu,\boldsymbol{u}_{\text{ref}_h},f_h) = \boldsymbol{u}_h(\cdot,E_h,\nu,\boldsymbol{u}_{\text{ref}_h},\boldsymbol{f}_h) = (\tilde{\mathbf{B}}(\boldsymbol{u}_{\text{ref}_h})+\bar{\mathbf{B}}(E_h,\nu,\boldsymbol{f}_h))^{\top}\boldsymbol{\Phi}.\end{aligned} \tag{5.134}$$

- The data sensing operator

$$\hat{\mathcal{P}} : X \to \mathbb{R}^k \times \mathbb{R} \times \mathbb{R}^{3k} \times \mathbb{R}^{3k}, \quad \hat{\mathcal{P}}(E,\nu,\boldsymbol{u}_{\text{ref}},\boldsymbol{f}) \mapsto (\hat{\boldsymbol{E}},\hat{\nu},\hat{\boldsymbol{U}}_{\text{ref}},\hat{\boldsymbol{F}}). \tag{5.135}$$

- The final VarMiON operator

$$\begin{aligned}&\hat{\mathcal{N}} : \mathbb{R}^k \times \mathbb{R} \times \mathbb{R}^{3k} \times \mathbb{R}^{3k} \to V_\tau\\ &\hat{\mathcal{N}}(\hat{\boldsymbol{E}},\hat{\nu},\hat{\boldsymbol{U}}_{\text{ref}},\hat{\boldsymbol{F}}) = \hat{\boldsymbol{u}}(\cdot,\hat{\boldsymbol{E}},\hat{\nu},\hat{\boldsymbol{U}}_{\text{ref}},\hat{\boldsymbol{F}}) = \tilde{\boldsymbol{\beta}}(\hat{\boldsymbol{U}}_{\text{ref}})+\bar{\boldsymbol{\beta}}(\hat{\boldsymbol{E}},\hat{\nu},\hat{\boldsymbol{F}}))^{\top}\boldsymbol{\tau}\end{aligned} \tag{5.136}$$

We can visualize the high-level implementation of the VarMiON for this equation on Ω in Figure 5.11.

In the testing and validation phases, we use the resulting prediction $\hat{\boldsymbol{u}}$ to compute J, $\mathbf{F}^{-\top}$ and $\mathbf{C}^{-1}$.

5.5.2 VarMiON for heat equation

Let us now write the weak space-time formulation for the heat equation involved in the bread leavening model in $\Omega \times I_n$, which reads

$$\sum_{j=0}^{N_h} \theta_{\mathrm{b}_j} \int_{I_n} \int_{\Omega} \rho_{\mathrm{b}} c_{\mathrm{b}}\, J \frac{\partial}{\partial t} \phi_j \phi_i \, d\boldsymbol{X}\, dt + \sum_{j=0}^{N_h} \theta_{\mathrm{b}_j} \int_{I_n} \int_{\Omega} (\beta_{\mathrm{b}}\, J\, \mathbf{C}^{-1} \nabla \phi_j) \cdot \nabla \phi_i \, d\boldsymbol{X}\, dt$$
$$= \int_{I_n} \int_{\Gamma_1} h_{\mathrm{bo}} (\theta_o - \sum_{j=0}^{N_h} \theta_{\mathrm{b}_j} \phi_j) \phi_i |J\, \mathbf{F}^{-\top} \boldsymbol{m}| \, dS\, dt + \int_{I_n} \int_{\Gamma_3} h_{\mathrm{bt}} (\theta_o - \sum_{j=0}^{N_h} \theta_{\mathrm{b}_j} \phi_j) \phi_i |J\, \mathbf{F}^{-\top} \boldsymbol{m}| \, dS\, dt. \tag{5.137}$$

Notice that, the previous equation depends on $J, \mathbf{C}^{-1}, \mathbf{F}^{-\top}$ due to the transformation in material coordinates. Thus, for the sake of simplicity, from now on, we will write only the dependence of $\boldsymbol{u}$, i.e., the vectorial placement of the elastic equation.

- The discrete weak formulation reads

$$[\mathbf{W}(\boldsymbol{u}, \rho_{\mathrm{b}} c_{\mathrm{b}}) + \mathbf{K}(\boldsymbol{u}, \beta_h) + h_{\mathrm{bo}} \tilde{\mathbf{M}}(\boldsymbol{u}) + h_{\mathrm{bt}} \bar{\mathbf{M}}(\boldsymbol{u})] \boldsymbol{\theta}_{\mathbf{b}} = h_{\mathrm{bo}} \tilde{\mathbf{M}}(\boldsymbol{u}) \boldsymbol{\theta}_{\mathbf{o}} + h_{\mathrm{bt}} \bar{\mathbf{M}}(\boldsymbol{u}) \boldsymbol{\theta}_{\mathbf{o}} \tag{5.138}$$

  for which holds the condition $\boldsymbol{\theta}_{\mathrm{b}} \boldsymbol{\Phi}(\boldsymbol{x}, t_n) = \boldsymbol{\theta}_{\mathrm{b}_n}^{\top} \boldsymbol{\Phi}(\boldsymbol{x}, t_n)$.

- The discrete solution operator

$$\begin{aligned} &\mathcal{N}_h : X_h \to V_h, \\ &\mathcal{N}_h(\boldsymbol{u}, \rho_{\mathrm{b}_h} c_{\mathrm{b}_h}, \beta_{\mathrm{b}_h}, \theta_{\mathrm{o}_h}, \theta_{\mathrm{b}_{nh}}, h_{\mathrm{bt}}, h_{\mathrm{bo}}) = \theta_{\mathrm{b}_h}(\cdot, \cdot, \boldsymbol{u}, \rho_{\mathrm{b}_h} c_{\mathrm{b}_h}, \beta_{\mathrm{b}_h}, \theta_{\mathrm{o}_h}, \theta_{\mathrm{b}_{nh}}, h_{\mathrm{bt}}, h_{\mathrm{bo}}) \\ &= (\tilde{\mathbf{B}}(\boldsymbol{u}, \rho_{\mathrm{b}_h} c_{\mathrm{b}_h}, \beta_{\mathrm{b}_h}, \theta_{\mathrm{o}_h}, h_{\mathrm{bt}}, h_{\mathrm{bo}}) + \bar{\mathbf{B}}(\boldsymbol{u}, \rho_{\mathrm{b}_h} c_{\mathrm{b}_h}, \beta_{\mathrm{b}_h}, \theta_{\mathrm{o}_h}, h_{\mathrm{bt}}, h_{\mathrm{bo}}) + \\ &\mathbf{B}(\boldsymbol{u}, \rho_{\mathrm{b}_h} c_{\mathrm{b}_h}, \beta_{\mathrm{b}_h}, h_{\mathrm{bt}}, h_{\mathrm{bo}}, \theta_{\mathrm{b}_{nh}}))^{\top} \boldsymbol{\Phi}. \end{aligned} \tag{5.139}$$

- The data sensing operator

$$\hat{\mathcal{P}} : X \to \mathbb{R}^{3k} \times \mathbb{R}^k \times \mathbb{R}^k \times \mathbb{R}^{k'r} \times \mathbb{R}^{kr} \times \mathbb{R} \times \mathbb{R},$$
$$\hat{\mathcal{P}}(\boldsymbol{u}, \rho_{\mathrm{b}} c_{\mathrm{b}}, \beta_{\mathrm{b}}, \theta_{\mathrm{o}}, \theta_{\mathrm{b}_n}, h_{\mathrm{bt}}, h_{\mathrm{bo}}) \mapsto (\hat{\boldsymbol{U}}, \hat{\boldsymbol{C}}, \hat{\boldsymbol{\beta}}, \hat{\boldsymbol{\theta}}_o, \hat{\boldsymbol{\theta}}_{\mathrm{b}_n}, \hat{h}_{\mathrm{bo}}, \hat{h}_{\mathrm{bt}}). \tag{5.140}$$

- The final VarMiON operator

$$\begin{aligned} &\hat{\mathcal{N}} : \mathbb{R}^{3k} \times \mathbb{R}^k \times \mathbb{R}^k \times \mathbb{R}^{k'r} \times \mathbb{R}^{kr} \times \mathbb{R} \times \mathbb{R} \to V_{\boldsymbol{\tau}} \\ &\hat{\mathcal{N}}(\hat{\boldsymbol{U}}, \hat{\boldsymbol{C}}, \hat{\boldsymbol{\beta}}, \hat{\boldsymbol{\theta}}_o, \hat{\boldsymbol{\theta}}_{\mathrm{b}_n}, \hat{h}_{\mathrm{bo}}, \hat{h}_{\mathrm{bt}}) = \hat{\theta}_{\mathrm{b}}(\cdot, \cdot, \hat{\boldsymbol{U}}, \hat{\boldsymbol{C}}, \hat{\boldsymbol{\beta}}, \hat{\boldsymbol{\theta}}_o, \hat{\boldsymbol{\theta}}_{\mathrm{b}_n}, \hat{h}_{\mathrm{bo}}, \hat{h}_{\mathrm{bt}}) \\ &= (\tilde{\boldsymbol{\beta}}(\hat{\boldsymbol{U}}, \hat{\boldsymbol{C}}, \hat{\boldsymbol{\beta}}, \hat{\boldsymbol{\theta}}_o, \hat{h}_{\mathrm{bo}}, \hat{h}_{\mathrm{bt}}) + \bar{\boldsymbol{\beta}}(\hat{\boldsymbol{U}}, \hat{\boldsymbol{C}}, \hat{\boldsymbol{\beta}}, \hat{\boldsymbol{\theta}}_o, \hat{h}_{\mathrm{bo}}, \hat{h}_{\mathrm{bt}}) \\ &+ \boldsymbol{\beta}(\hat{\boldsymbol{U}}, \hat{\boldsymbol{C}}, \hat{\boldsymbol{\beta}}, \hat{h}_{\mathrm{bo}}, \hat{h}_{\mathrm{bt}}, \hat{\boldsymbol{\theta}}_{\mathrm{b}_n}))^{\top} \boldsymbol{\tau}. \end{aligned} \tag{5.141}$$

We can visualize the high-level implementation of the VarMiON for this equation on $\Omega \times I_n$ in Figure 5.12.

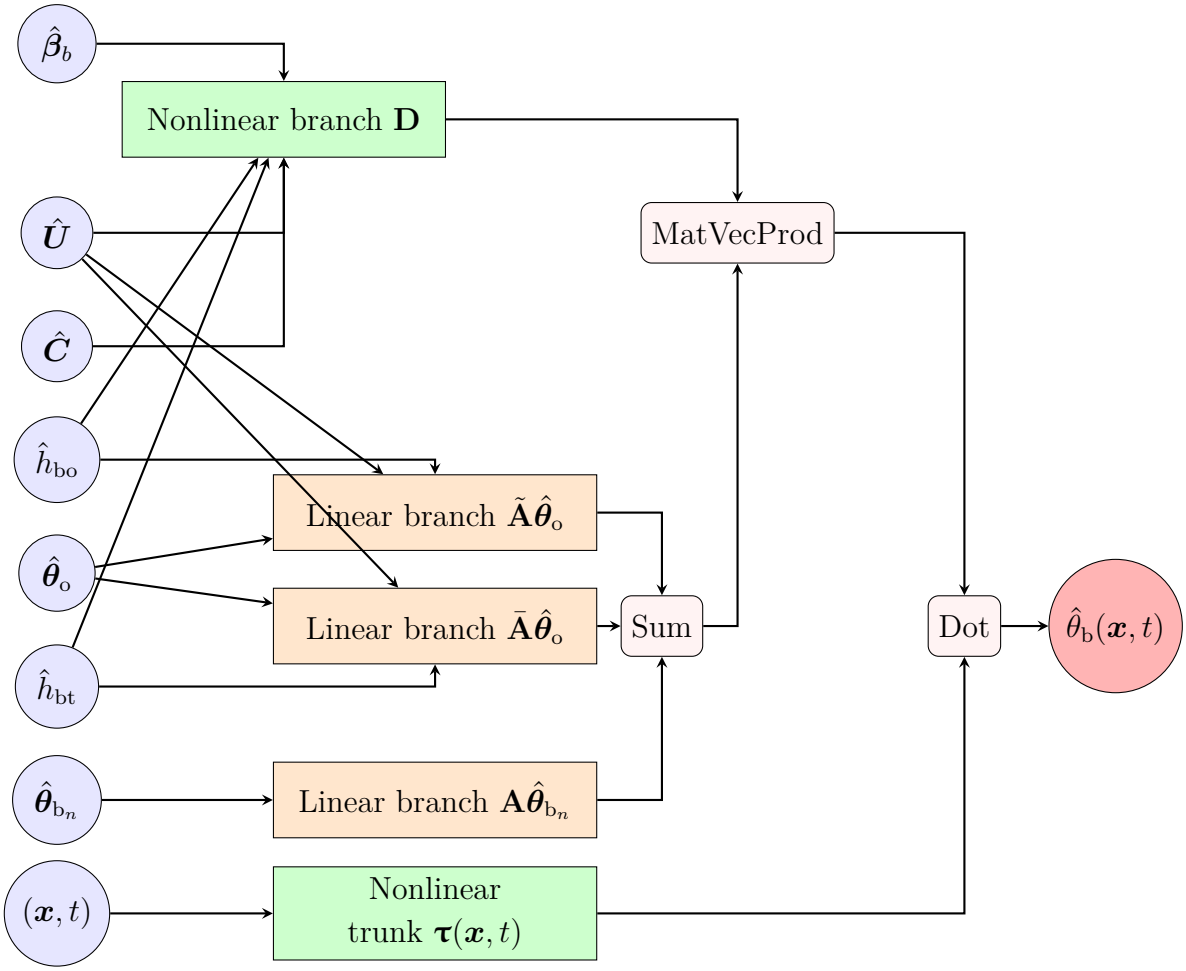

Figure 5.12: Schematic of a VarMiON for time-dependent heat equation for the temperature evolution of bread in $\Omega \times I_n$.

5.5.3 VarMiON for CO_2 diffusion equation

Let us now write the weak space-time formulation for the CO_2 diffusion involved in the bread leavening model in $\Omega \times I_n$, which reads

$$\sum_{j=0}^{N_h} D_j \int_{I_n}\int_{\Omega} J\,\frac{\partial}{\partial t}\phi_j\phi_i\,d\boldsymbol{X}\,dt + \sum_{j=0}^{N_h} D_j \int_{I_n}\int_{\Omega} (\beta_{co2}\,J\,\mathbf{C}^{-1}\nabla\phi_j)\cdot(\nabla\phi_i)\,d\boldsymbol{X}\,dt = \int_{I_n}\int_{\Omega} J\,f(\theta_{\mathrm{b}})\,Y\,\phi_i\,d\boldsymbol{X}\,dt. \quad (5.142)$$

- The discrete weak formulation with the initial condition reads

$$(\mathbf{W}(\boldsymbol{u}) + \mathbf{K}(\boldsymbol{u}, \beta_{co2}))\boldsymbol{D} = \tilde{\mathbf{M}}(\boldsymbol{u})\boldsymbol{F}(\boldsymbol{\theta}_{\mathrm{b}})\boldsymbol{Y} \quad (5.143)$$

  for which holds the condition $\boldsymbol{D}\boldsymbol{\Phi}(\boldsymbol{x}, t_n) = \boldsymbol{D}_n^{\top}\boldsymbol{\Phi}(\boldsymbol{x}, t_n)$.

- The discrete solution operator

$$\begin{aligned}\mathcal{N}_h : X_h &\to V_h,\\ \mathcal{N}_h(\cdot,\cdot,\boldsymbol{u},\beta_{co2_h},\theta_{\mathrm{b}_h},Y_h,D_{n_h}) &= D_h(\boldsymbol{u},\beta_{co2_h},\theta_{\mathrm{b}_h},Y_h,D_{n_h})\\ &= (\tilde{\mathbf{B}}(\boldsymbol{u},\beta_{co2_h},\theta_{\mathrm{b}_h},Y_h) + \mathbf{B}(\boldsymbol{u},\beta_{co2_h},D_{n_h}))^{\top}\boldsymbol{\Phi}. \quad (5.144)\end{aligned}$$

- The data sensing operator

$$\begin{aligned}\hat{\mathcal{P}} : X &\to \mathbb{R}^{3k}\times\mathbb{R}^{k}\times\mathbb{R}^{kr}\times\mathbb{R}^{kr}\times\mathbb{R}^{k}\\ \hat{\mathcal{P}}(\boldsymbol{u},\beta_{co2},\theta_{\mathrm{b}},Y,D_n) &\mapsto (\hat{\boldsymbol{U}},\hat{\boldsymbol{\beta}}_{co2},\hat{\boldsymbol{\theta}}_{\mathrm{b}},\hat{\boldsymbol{Y}},\hat{\boldsymbol{D}}_n). \quad (5.145)\end{aligned}$$

- The final VarMiON operator

$$\begin{aligned}\hat{\mathcal{N}} : \mathbb{R}^{3k}\times\mathbb{R}^{k}\times\mathbb{R}^{kr}\times\mathbb{R}^{kr}\times\mathbb{R}^{k} &\to V_{\boldsymbol{\tau}}\\ \hat{\mathcal{N}}(\hat{\boldsymbol{U}},\hat{\boldsymbol{\beta}}_{co2},\hat{\boldsymbol{\theta}}_{\mathrm{b}},\hat{\boldsymbol{Y}},\hat{\boldsymbol{D}}_n) &= \hat{D}(\cdot,\cdot,\hat{\boldsymbol{U}},\hat{\boldsymbol{\beta}}_{co2},\hat{\boldsymbol{\theta}}_{\mathrm{b}},\hat{\boldsymbol{Y}},\hat{\boldsymbol{D}}_n)\\ &= (\tilde{\boldsymbol{\beta}}(\hat{\boldsymbol{U}},\hat{\boldsymbol{\beta}}_{co2},\hat{\boldsymbol{\theta}}_{\mathrm{b}},\hat{\boldsymbol{Y}}) + \boldsymbol{\beta}(\hat{\boldsymbol{U}},\hat{\boldsymbol{\beta}}_{co2},\hat{\boldsymbol{D}}_n))^{\top}\boldsymbol{\tau}. \quad (5.146)\end{aligned}$$

We can visualize the high-level implementation of the VarMiON for this equation in Figure 5.13.

At this point, we recall that in our bread leavening model, the growth of the elastic paste results by updating the elastic stiffness values: especially the Young's modulus E_n and the value of J_{ref} (which represents a dilation coefficient w.r.t. a target volume configuration) that change according to (2.21).

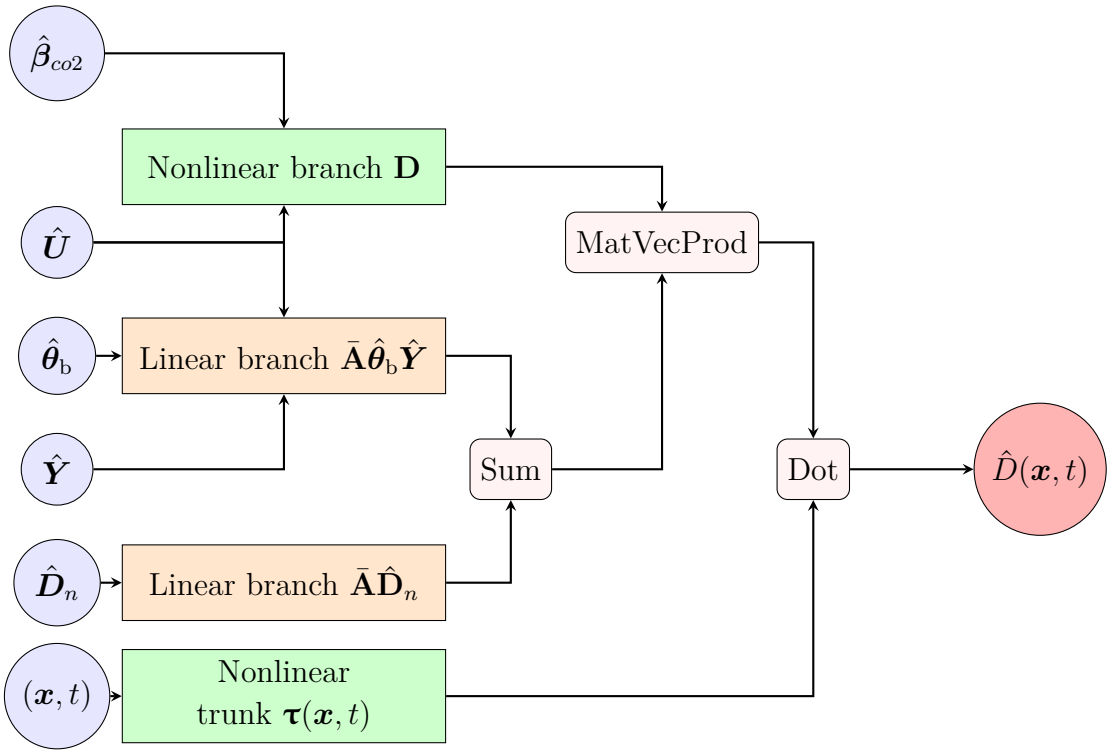

Figure 5.13: Schematic of a VarMiON for time-dependent carbon dioxide production and diffusion in $\Omega \times I_n$.

Chapter 6

Leavening monitoring: a theory to replace voids with fictitious forcing terms

In this chapter, we present a process on how to replace changes of material properties in limited regions within a domain with fictitious forcing terms in initial- and boundary-value problems associated with wave propagation and diffusion. Then, by considering a paradigmatic heat conduction problem on a domain with a cavity, we prove that the presence of the void can be replaced by a fictitious heat source with support contained within the cavity. We illustrate this fact in a situation where the source term can be analytically recovered from the values of the temperature and heat flux at the boundary of the cavity following the paper [4].

At this point, our theory of voids identification sets the stage for monitoring the leavening. In particular, the equivalence between voids and fictitious forces can be applied to our case, to reconstruct the porous texture of bread arising during leavening. Indeed, with the final goal of calibrating the parameters of our model (e.g., the elastic stiffness), which change their values during the process, we need to solve a source inverse problem (see Chapter 7) instead of the nonlinear inverse problem of geometry identification thanks to our strategy.

6.1 The strategy

In several applications, the solution of a partial differential equation on a domain with a complex geometry can be found starting from solutions defined on a larger, but topologically simpler, domain. In particular, problems in exterior domains or domains with holes can be mapped onto problems in the whole space or in filled domains. This is for instance the case in the method of image charges for electrostatics [41] or that of singularities in low-Reynolds-number flows [42]. Here, we present a way to map the estimation of regions with altered properties into a force reconstruction problem.

A specialization of such a scheme provides a way to tackle geometry estimation or void identification problems, in which the domain is unknown, as source or force identification problems stated on a fixed, known domain. To this end, we need to introduce suitable extensions of differential problems from complex domains to larger and simpler ones, in such a way that the restriction of the solution to the physical domain remains unchanged. From the point of view of computational efficiency, this is important because the nonlinear inverse problem of geometry identification is replaced by an easier-to-solve linear inverse problem of source identification, for which numerous theoretical results and numerical approaches are available, see e.g. [43] for thermal problems.

Many of the developed approaches are related to the formulation of an optimization problem with a suitable cost function to minimize by using some technique such as [44–47]. All the approaches formulate nonlinear inverse problems based upon temperature-prediction-errors and as already shown by mere intuition and algorithmic exploration [48] that estimating the fictitious heat source turns out to be much easier than estimating the

void by solving a geometric inverse problem, and much more accurate than processing the gradient of temperature-prediction errors.

For definiteness, we will employ a common geometrical setting throughout the whole section. We consider two open and simply connected bounded domains Ω_b and Ω_c in $\mathbb{R}^n$ such that $\overline{\Omega_\mathrm{c}} \subset \Omega_\mathrm{b}$, where Ω_c may represent a region where mechanical parameters differ from the reference value or even a cavity within the body Ω_b. With these provisions, the domain $\Omega_\mathrm{d} := \Omega_\mathrm{b} \setminus \overline{\Omega_\mathrm{c}}$ presents a single hole and we can identify the exterior and interior boundaries as $\partial\Omega_\mathrm{b}$ and $\partial\Omega_\mathrm{c}$, respectively, with the obvious property that $\partial\Omega_\mathrm{d} = \partial\Omega_\mathrm{b} \cup \partial\Omega_\mathrm{c}$ and $\partial\Omega_\mathrm{b} \cap \partial\Omega_\mathrm{c} = \emptyset$. Moreover, we introduce the final time $\tau > 0$ and set differential problems on the time interval $[0, \tau]$.

We first consider the propagation of waves in the domain Ω_b. In the simplest linear and isotropic approximation, the relevant mechanical parameter in the D'Alembert equation is the wave speed c, which may be non-homogeneous. Given a reference value $c_0 > 0$, we assume

$$c(\boldsymbol{x}) = \begin{cases} c_0 & \text{if } \boldsymbol{x} \in \Omega_\mathrm{d}, \\ c_1(\boldsymbol{x}) & \text{if } \boldsymbol{x} \in \Omega_\mathrm{c}, \end{cases} \tag{6.1}$$

in such a way that c is a smooth function in Ω_b.

Denoting by $u : \Omega_\mathrm{b} \times [0, \tau] \to \mathbb{R}$ any scalar field of interest, the corresponding wave equation, together with suitable initial and boundary conditions, becomes

$$\begin{cases} \partial^2_{tt} u - \operatorname{div}(c^2 \nabla u) = 0 & \text{on } \Omega_\mathrm{b} \times [0, \tau], \\ u = g & \text{on } \partial\Omega_\mathrm{b} \times [0, \tau], \\ u(0, \cdot) = u_0(\cdot) & \text{on } \Omega_\mathrm{b}, \\ \partial_t u(0, \cdot) = v_0(\cdot) & \text{on } \Omega_\mathrm{b}, \end{cases} \tag{6.2}$$

where g is a prescribed datum, depending on time and space, and u_0 and v_0 represent the initial conditions.

Proposition 6.1.1. *Given a solution $u : \Omega_\mathrm{b} \times [0, \tau] \to \mathbb{R}$ of the differential problem (6.2) with smooth boundary data $g : \partial\Omega_\mathrm{b} \times [0, \tau] \to \mathbb{R}$ and initial conditions u_0 and v_0, there exists a fictitious force field $f : \Omega_\mathrm{b} \times [0, \tau] \to \mathbb{R}$ such that:*

(i) the function u is a solution of the differential problem

$$\begin{cases} \partial^2_{tt} u - c_0^2 \Delta u = f & \text{on } \Omega_\mathrm{b} \times [0, \tau], \\ u = g & \text{on } \partial\Omega_\mathrm{b} \times [0, \tau], \\ u(0, \cdot) = u_0(\cdot) & \text{on } \Omega_\mathrm{b}, \\ \partial_t u(0, \cdot) = v_0(\cdot) & \text{on } \Omega_\mathrm{b}, \end{cases} \tag{6.3}$$

(ii) $f = 0$ in the region Ω_d, where $c = c_0$.

Proof. It is immediate to verify that defining $f := \operatorname{div}[(c^2 - c_0^2)\nabla u]$ leads to (i) and (ii). □

A completely analogous statement holds for the case of diffusion equations.

Proposition 6.1.2. *Let $\alpha_0 > 0$ be a reference diffusivity (constant and uniform) and let $\alpha : \Omega_\mathrm{b} \to \mathbb{R}$ be a positive and smooth function such that $\alpha = \alpha_0$ in Ω_d. Given a solution $u : \Omega_\mathrm{b} \times [0, \tau] \to \mathbb{R}$ of the differential problem*

$$\begin{cases} \partial_t u - \operatorname{div}(\alpha \nabla u) = 0 & \text{on } \Omega_\mathrm{b} \times [0, \tau], \\ u = g & \text{on } \partial\Omega_\mathrm{b} \times [0, \tau], \\ u(0, \cdot) = u_0(\cdot) & \text{on } \Omega_\mathrm{b}, \end{cases} \tag{6.4}$$

with smooth boundary data $g : \partial\Omega_\mathrm{b} \times [0, \tau] \to \mathbb{R}$ and initial condition u_0, there exists a fictitious source field $f : \Omega_\mathrm{b} \times [0, \tau] \to \mathbb{R}$ such that u is a solution of

$$\begin{cases} \partial_t u - \alpha_0 \Delta u = f & \text{on } \Omega_\mathrm{b} \times [0, \tau], \\ u = g & \text{on } \partial\Omega_\mathrm{b} \times [0, \tau], \\ u(0, \cdot) = u_0(\cdot) & \text{on } \Omega_\mathrm{b}, \end{cases} \tag{6.5}$$

and such that $f = 0$ in Ω_d.

Proof. It is enough to define $f := \operatorname{div}[(\alpha - \alpha_0)\nabla u]$. □

As simple as they may be, the foregoing facts have important practical implications in the context of inverse problems. We can think for instance of a scenario where we wish to identify the region Ω_c in which the parameters change significantly having available only measurements of u on a portion of the boundary and a knowledge of the reference parameters. Instead of attacking the inverse problem of estimating the region itself or the values of the unknown parameters, we can find an estimate of the fictitious force f and then approximate Ω_c with the support of f. Since the force appears linearly in the equation and does not multiply any unknown, we obtain important simplifications from a computational point of view.

6.1.1 Heat conduction in domains with a cavity

Considering Ω_c as an actual cavity requires a more careful treatment of the equations, but we can set up a scheme parallel to what we just presented. We will illustrate this working on a heat conduction problem that, not only provides an important application for the methods we introduced, but also represents a paradigmatic case in the context of void identification procedures. In this particular setting, we show the equivalence of two

differential problems in describing the evolution of the temperature field in the physical domain. Introducing the constant mass density $\rho > 0$, specific heat $c > 0$, and thermal conductivity $\beta > 0$, we can write the heat conduction problem for the temperature field $\theta : \Omega_{\mathrm{d}} \times [0, \tau] \to \mathbb{R}$ as

$$\begin{cases} \rho c \partial_t \theta = \beta \Delta \theta & \text{on } \Omega_{\mathrm{d}} \times [0, \tau], \\ \beta \nabla \theta \cdot \boldsymbol{n} = g & \text{on } \partial\Omega_{\mathrm{b}} \times [0, \tau], \\ \beta \nabla \theta \cdot \boldsymbol{n} = 0 & \text{on } \partial\Omega_{\mathrm{c}} \times [0, \tau], \\ \theta(0, \cdot) = \theta_0(\cdot) & \text{on } \Omega_{\mathrm{d}}, \end{cases} \tag{6.6}$$

where $\boldsymbol{n}$ is the unit outer normal to $\partial\Omega_{\mathrm{d}}$, g is a prescribed (space- and time-dependent) heat flux on the external boundary, and θ_0 is the initial temperature field. For the sake of simplicity, we assume a vanishing heat flux across the internal boundary $\partial\Omega_{\mathrm{c}}$.

Our goal is now to prove that there exists a second differential problem, stated on the filled domain Ω_{b}, the solution of which is a temperature field $\tilde{\theta} : \Omega_{\mathrm{b}} \times [0, \tau] \to \mathbb{R}$ such that $\tilde{\theta} = \theta$ on $\Omega_{\mathrm{d}} \times [0, \tau]$. This problem takes the form

$$\begin{cases} \rho c \partial_t \tilde{\theta} = \beta \Delta \tilde{\theta} + f & \text{on } \Omega_{\mathrm{b}} \times [0, \tau], \\ \beta \nabla \tilde{\theta} \cdot \boldsymbol{n} = g & \text{on } \partial\Omega_{\mathrm{b}} \times [0, \tau], \\ \tilde{\theta}(0, \cdot) = \tilde{\theta}_0(\cdot) & \text{on } \Omega_{\mathrm{b}}, \end{cases} \tag{6.7}$$

where the initial condition $\tilde{\theta}_0$ coincides with θ_0 on Ω_{d} and $f : \Omega_{\mathrm{b}} \times [0, \tau] \to \mathbb{R}$ is a fictitious heat source.

Theorem 6.1.3. *Given a solution θ of the differential problem* (6.6) *with smooth boundary data g and initial condition θ_0, there exists a fictitious source field f such that:*

(i) the solution $\tilde{\theta}$ of the differential problem (6.7) *coincides with θ on $\Omega_{\mathrm{d}} \times [0, \tau]$;*

(ii) $f = 0$ in the physical domain $\Omega_{\mathrm{d}} \times [0, \tau]$.

Proof. Let us seek conditions on f that guarantee that $\tilde{\theta} = \theta$ on $\Omega_{\mathrm{d}} \times [0, \tau]$. Denoting by ϕ a smooth test field on $\Omega_{\mathrm{b}} \times [0, \tau]$, the weak formulations of problems (6.6) and (6.7) read

$$\int_{\Omega_{\mathrm{d}}} \rho c \partial_t \theta \phi \, dx + \int_{\Omega_{\mathrm{d}}} (\beta \nabla \theta) \cdot (\nabla \phi) \, dx - \int_{\partial\Omega_{\mathrm{b}}} \beta g(x) \phi \, ds = 0, \tag{6.8}$$

$$\int_{\Omega_{\mathrm{b}}} \rho c \partial_t \tilde{\theta} \phi \, dx + \int_{\Omega_{\mathrm{b}}} (\beta \nabla \tilde{\theta}) \cdot (\nabla \phi) \, dx - \int_{\Omega_{\mathrm{b}}} f \phi \, dx - \int_{\partial\Omega_{\mathrm{b}}} \beta g(x) \phi \, ds = 0. \tag{6.9}$$

The condition $\tilde{\theta} = \theta$ on $\Omega_{\mathrm{d}} \times [0, \tau]$ holds if and only if taking the difference of the two equations, we obtain

$$\begin{aligned} &\Big(\int_{\Omega_{\mathrm{d}}} \rho c \partial_t \theta \phi \, dx + \int_{\Omega_{\mathrm{d}}} (\beta \nabla \theta) \cdot (\nabla \phi) \, dx - \int_{\partial\Omega_{\mathrm{b}}} \beta g(x) \phi \, ds \Big) \\ &- \Big(\int_{\Omega_{\mathrm{b}}} \rho c \partial_t \tilde{\theta} \phi \, dx + \int_{\Omega_{\mathrm{b}}} (\beta \nabla \tilde{\theta}) \cdot (\nabla \phi) \, dx - \int_{\Omega_{\mathrm{b}}} f \phi \, dx - \int_{\partial\Omega_{\mathrm{b}}} \beta g(x) \phi \, ds \Big) = 0 \end{aligned} \tag{6.10}$$

notice that the holds $\Omega_\mathrm{d} := \Omega_\mathrm{b} \setminus \overline{\Omega_\mathrm{c}}$

$$\int_{\Omega_\mathrm{c}} \rho c \partial_t \tilde{\theta} \phi \, dx + \int_{\Omega_\mathrm{c}} (\beta \nabla \tilde{\theta}) \cdot (\nabla \phi) \, dx - \int_{\Omega_\mathrm{b}} f \phi \, dx = 0. \tag{6.11}$$

By the arbitrariness of ϕ and under suitable regularity assumptions for the temperature fields, we can apply the fundamental lemma of the calculus of variations and obtain $f = 0$ on the physical domain Ω_d and the expression of the fictitious source inside the cavity Ω_c as

$$f = \rho c \partial_t \tilde{\theta} - \beta \Delta \tilde{\theta}, \tag{6.12}$$

with $\beta \nabla \tilde{\theta} \cdot \boldsymbol{n} = 0$ and $\tilde{\theta} = \theta$ on $\partial\Omega_\mathrm{c}$, obtained by integrating by parts (6.11).

We can conclude that, given a solution θ of the physical problem (6.6), a fictitious source f to be used in (6.7) to correctly reproduce the temperature field on the physical domain can be constructed as follows:

1. Set $f = 0$ on the physical domain Ω_d.
2. Find any extension $\tilde{\theta}$ to the cavity domain Ω_c of the physical temperature field θ, in such a way that the boundary conditions $\beta \nabla \tilde{\theta} \cdot \boldsymbol{n} = 0$ and $\tilde{\theta} = \theta$ on $\partial\Omega_\mathrm{c}$ are satisfied at all times.
3. Use equation (6.12) to compute the fictitious source inside the cavity Ω_c.

If boundary data require using weak solutions in Sobolev spaces rather than smooth ones, we shall interpret the foregoing identities as holding almost everywhere in the respective domains. □

6.2 An admissible extension

Clearly, there is no unique way to obtain an extension with the sought properties.

If the domain has a very simple shape we may explicitly write smooth extensions, but we propose here a strategy that allows us to select a specific extension whenever Ω_c has a Lipschitz boundary. We fix a characteristic length ε (say, the diameter of Ω_c) and, for each time instant, consider the unique solution of the differential problem

$$\begin{cases} -\Delta(\tilde{\theta} - \varepsilon^2 \Delta \tilde{\theta}) = 0 & \text{on } \Omega_\mathrm{c}, \\ \tilde{\theta} = \theta & \text{on } \partial\Omega_\mathrm{c}, \\ \nabla \tilde{\theta} \cdot \boldsymbol{n} = 0 & \text{on } \partial\Omega_\mathrm{c}. \end{cases} \tag{6.13}$$

Existence and uniqueness of solution to (6.13) can be easily proved with standard techniques in the calculus of variations. Indeed, it is enough to look for minimizers of a convex functional, equivalent to the squared H^2-norm, on the closed convex subset of $H^2(\Omega_\mathrm{c})$ identified by the boundary conditions [33]. We expect extensions found in this way to feature smaller gradients and be more amenable to numerical approximations. A further crucial aspect in choosing a fourth-order differential operator to determine the extension is that it allows to impose boundary conditions on both the solution and its normal derivative on all of the boundary. This would be impossible if we were to use, for example, a second-order operator such as the Laplacian.

Even though there are infinite possible choices for the temperature extension and, as a consequence, for the fictitious source f, we stress that all of them do vanish in the physical domain Ω_d, while possibly being discontinuous at the cavity boundary $\partial\Omega_\mathrm{c}$. This represents a good feature if one wishes to use fictitious sources to determine the shape of voids within the physical domain.

6.2.1 Numerical example

We shall now illustrate the theoretical result of the previous section by means of an explicit comparison taking the filled domain $\Omega_\mathrm{b} \subset \mathbb{R}^2$ as the rectangle of width 0.41 and height 2.2 (in arbitrary units of length) and the cavity Ω_c as the circle with radius $R = 0.1$ and centered in $O = (0.2, 0.2)$. We consider a time-dependent boundary condition that represents a thermal flash $g(\boldsymbol{x}, t) = 5 \cdot 10^7 t \exp(-10\sqrt{5t})$ on the bottom side of Ω_b and $g(\boldsymbol{x}, t) = 0$ on the rest of the boundary. Since the cavity is circular, we consider polar coordinates $(r, \omega) \in [0, R] \times [0, 2\pi)$ centered in O. We denote by $\theta^\mathrm{e}(t, \omega)$ the value of the temperature on the cavity boundary $\partial\Omega_\mathrm{c}$ as given by the solution of the physical problem (6.6). To find the extension $\tilde{\theta}$, we consider the Fourier series expansions

$$\theta^\mathrm{e}(t,\omega) = \sum_{j=0}^{\infty} \hat{\theta}_j^{(\mathrm{e})}(t) \exp(-j\omega\mathrm{i}) \qquad \text{and} \qquad \tilde{\theta}(t, r, \omega) = \sum_{j=0}^{\infty} \hat{\theta}_j(t, r) \exp(-j\omega\mathrm{i}). \tag{6.14}$$

If we now substitute $\varepsilon = R$ and the above expansion of $\tilde{\theta}$ in (6.13) and use the orthogonality of the Fourier basis, we obtain the equations

$$\hat{\theta}_j \frac{j^2}{r^2}\Big(\frac{3R^2}{r^2} - 1\Big) + \frac{\partial\hat{\theta}_j}{\partial r}\frac{1}{r}\Big(1 - \frac{R^2}{r^2}(1+2j^2)\Big) + \frac{\partial^2\hat{\theta}_j}{\partial r^2}\Big(1 + \frac{R^2}{r^2}(1+2j^2)\Big) - \frac{2R^2}{r}\frac{\partial^3\hat{\theta}_j}{\partial r^3} - R^2\frac{\partial^4\hat{\theta}_j}{\partial r^4} = 0 \tag{6.15}$$

for each $j \in \mathbb{N}$, with boundary conditions

$$\hat{\theta}_j(t, R) = \hat{\theta}_j^\mathrm{e}(t), \qquad \partial_r\hat{\theta}_j(t, R) = 0 = \partial_r\hat{\theta}_j(t, 0), \qquad \lim_{r\to 0} |\hat{\theta}_j(t, r)| < \infty. \tag{6.16}$$

By introducing the dimensionless variable $y = r/R$, we can express the general solutions of (6.15) as

$$\hat{\theta}_0(t, Ry) = c_1^0(t) + c_2^0(t)\log(y) + c_3^0(t)I_0(y) + c_4^0(t)K_0(y) \qquad \text{if } j = 0, \tag{6.17}$$

$$\hat{\theta}_j(t, Ry) = c_1^j(t)y^j + c_2^j(t)y^{-j} + c_3^j(t)I_j(y) + c_4^j(t)K_j(y) \qquad \text{if } j > 0, \tag{6.18}$$

where $I_n(y)$ and $K_n(y)$ are modified Bessel functions of order n [49]. The time-dependent coefficients $c_i^j(t)$ for $i = 1, \ldots, 4$ are determined by the boundary conditions (6.16) and lead to the solution

$$\hat{\theta}_0(t, r) = \hat{\theta}_0(t, Ry) = \hat{\theta}_0^{\mathrm{e}}(t) \qquad \text{if } j = 0, \tag{6.19}$$

$$\begin{aligned}\hat{\theta}_j(t, r) &= \hat{\theta}_j(t, Ry) \\ &= \left[\frac{(I_{j-1}(1) + I_{j+1}(1))}{2jI_{j+1}(1)}y^j - \frac{I_j(y)}{I_{j+1}(1)}\right]\hat{\theta}_j^{\mathrm{e}}(t) =: \alpha_j(y)\,\hat{\theta}_j^{\mathrm{e}}(t) \qquad \text{if } j > 0.\end{aligned} \tag{6.20}$$

From these, we can finally compute the fictitious source

$$f(t, r, \omega) = \rho c\frac{\partial}{\partial t}\tilde{\theta} - k\Delta\tilde{\theta} = \rho c\frac{\partial}{\partial t}\hat{\theta}_0^{\mathrm{e}}(t) + \sum_{j=0}^{\infty}\left[\rho c\alpha_j(y)\frac{\partial}{\partial t}\hat{\theta}_j^{\mathrm{e}}(t) + \frac{k}{R^2}\hat{\theta}_j^{\mathrm{e}}(t)\frac{I_j(y)}{I_{j+1}(1)}\right]\exp(-j\omega\mathrm{i}). \tag{6.21}$$

To verify the coherence of the solution of the heat conduction problem (6.6) on the domain Ω_{d} with the one of Problem (6.7) on Ω_{b} with the fictitious source given by (6.21), we implemented a finite-element discretization using the Python library FEniCS [21, 22]. Since $\theta^{\mathrm{e}}(t, \omega)$ is known only on a discrete set of mesh points, we need to perform, at each time step, a discrete Fourier transform to obtain the coefficients in (6.21). Within the approximation accuracy, we find an excellent agreement between the two solutions (Figure 6.1).

6.3 Experimental evidences

At this stage, we make some experiments to underline the different temperature evolutions obtained by heating two loaves of bread: one intact and the other with a cavity. We compare the thermographies of the experiments achieved by a thermal imaging camera.

First of all, we perform the experiment with the intact bread, then we repeat it with the previous bread in which we simulate a cavity. In particular, we use localized heating for about 20 s and we get regularly thermal pictures with the thermocamera. We process the images by saving the time-steps, at which the thermographies were acquired, and the matrices where we load the values of temperature corresponding to each pixel of the image.

On the thermographies of the two experiments, we localize the position of the bread by scanning the images and selecting the points where the color difference between two close

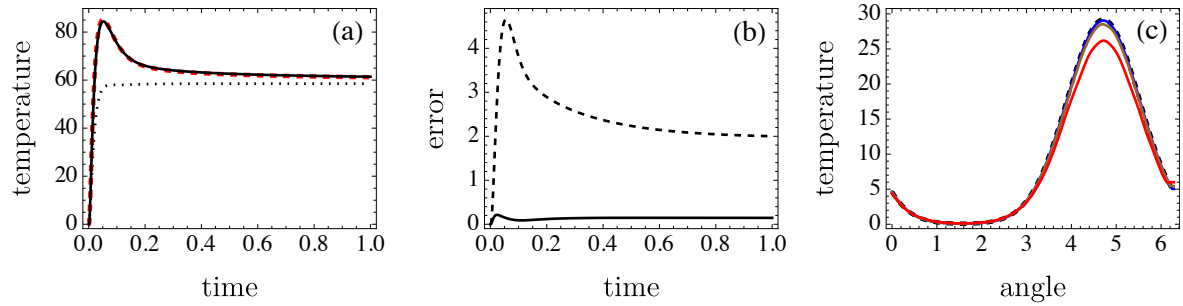

Figure 6.1: The source term added on the filled domain well simulates the void. (a) The physical temperature evolution at one point on the cavity boundary (red dashed line) is well captured by the solution of the problem with a fictitious force (black solid line), whereas the heat equation on the filled domain and without source term (dotted line) gives a very different prediction. (b) The global error given by the L^2-norm of the difference, on the physical domain, between the physical temperature and the one obtained with the fictitious force (solid line) is much lower than what would be predicted (dashed curve) using a filled domain and without the fictitious source. (c) The grid refinement is crucial to obtain a good approximation of the datum at the cavity boundary. Here we show the comparison between boundary data obtained by interpolation (black dashed line) and its reconstruction with Fourier series expansions as the number of mesh nodes on the cavity varies: $40, 80, 160$ nodes (red, brown, blue line respectively).

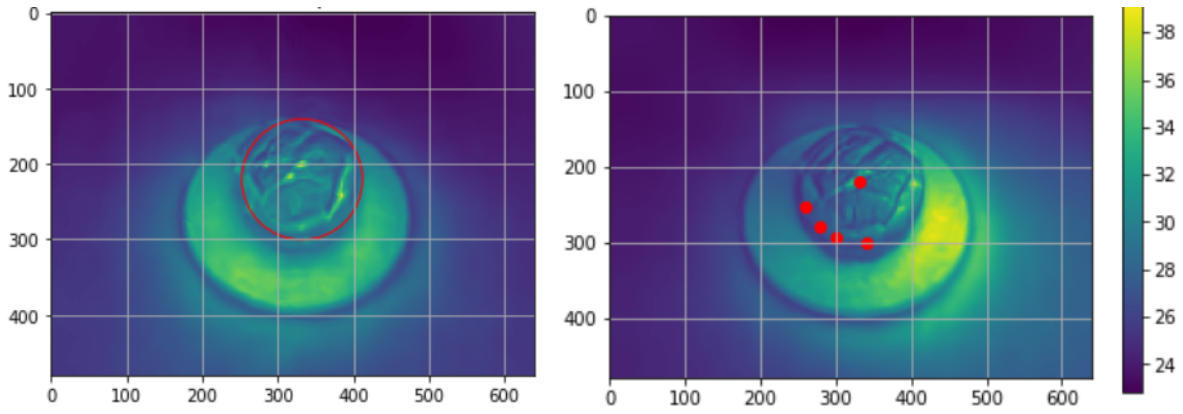

Figure 6.2: Thermographies of the experiments.

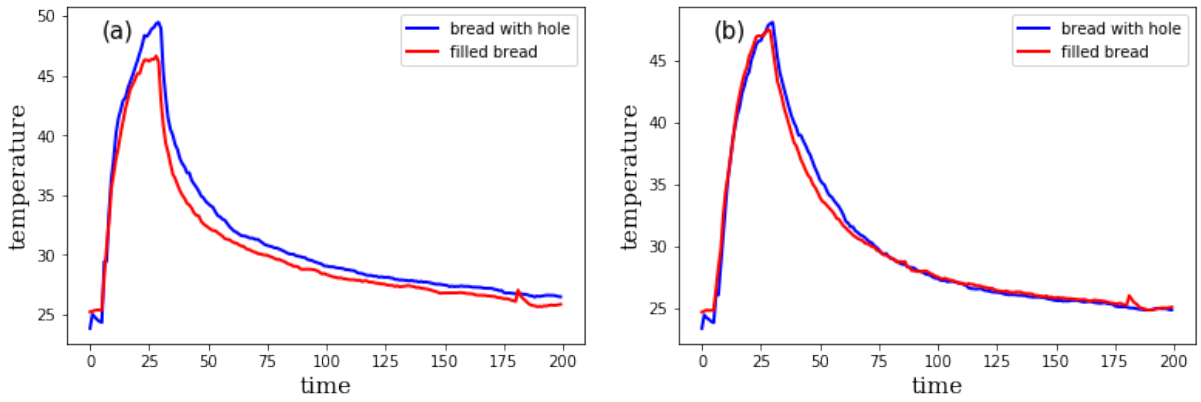

Figure 6.3: Experimental plots: (a) near the cavity, (b) far away from the cavity. The plots represent the temperature evolution when we heat up the bread for about 20 s. It is clear that the temperature trend for a point close to the cavity (a) is more affected by the presence of the hole and therefore heats up more than a distant point (b).

pixels is greater than a threshold. This procedure allows us to identify a circle passing through the aforementioned selected points, thus the bread position Figure 6.2. Moreover, we get the correspondence between centimeters and pixels and the cavity position on the thermographies. Indeed, we compare the temperature evolutions of points close to, or far away from the cavity on the bread with the hole and, the same points on the filled one, to underline the different thermal behaviors (see Figure 6.3).

By looking at these plots we note that points far away from the cavity have quite similar temperature evolution for the two experiments, see Figure 6.3 (b) instead, points close to the cavity on bread with the hole heat up more than the corresponding ones on the filled bread see Figure 6.3 (a).

Finally, we reproduce numerically the experiments, mentioned above, using the Python library FEniCS [21, 22] on a 3-dimensional geometry Figure 6.4. We solve a heat equation with a source term that acts on the boundary for about 20 time steps and, after calibration of physical constants, we obtain temperature evolutions that well-simulated the experimental ones, underlying that the bread with hole, near the cavity, heats up more than the whole one as we can expect, see Figure 6.5. Notice that, such thermal difference can be replaced by a fictitious forcing term following the theory explained in the previous Section 6.1.1.

In addition, the support of "SMACT" and "SimNumerica" allows us to another empirical phase where we perform the same thermal experiments as before, on an industrially produced bread, whose resulting thermographies, captured with the thermocamera, seem to confirm previous outcomes Figure 6.6.

By cutting the bread after baking, we recognize the position of some bubbles arising

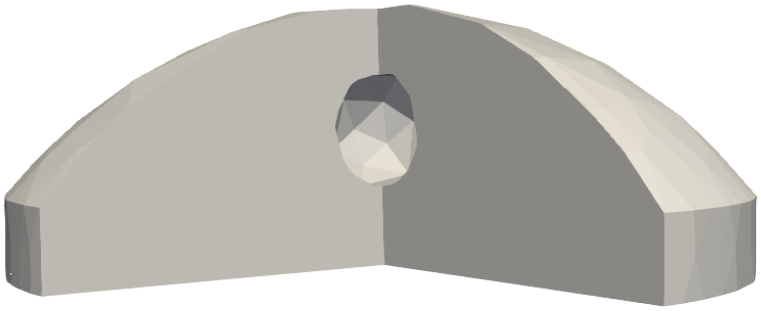

Figure 6.4: $3D$ numerical simulation-domain.

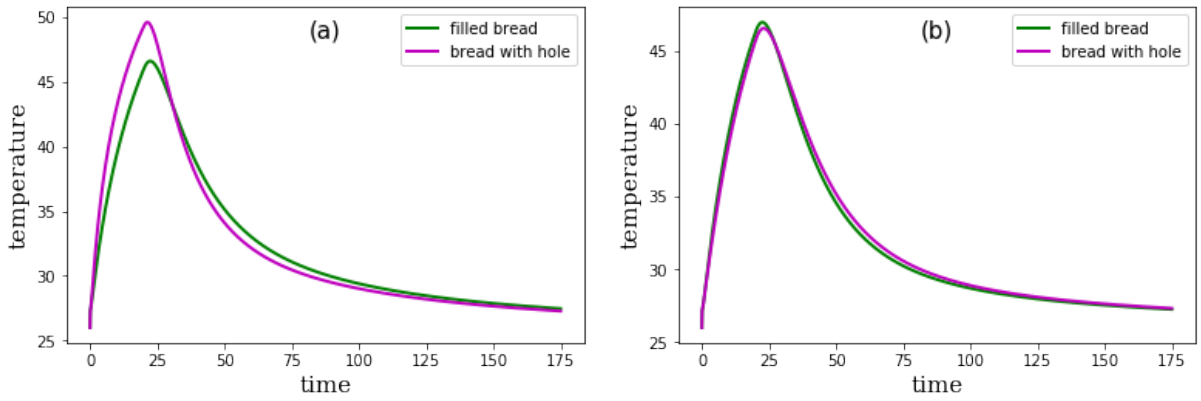

Figure 6.5: $3D$ numerical simulation: (a) near the cavity, (b) far away from the cavity. These plots well reflect the experimental trends of Figure 6.3

.

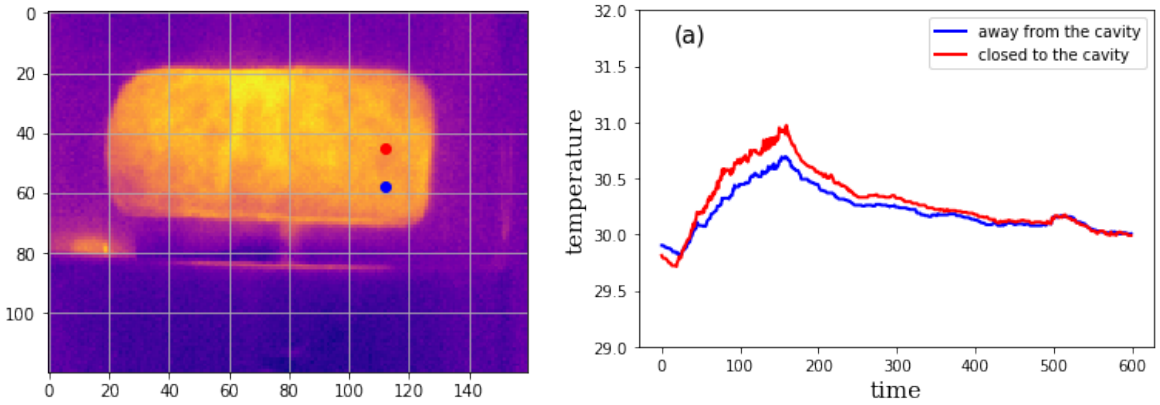

Figure 6.6: Experimental phase with bread industrially produced: on the left there is the thermography, on the right (a) there is the comparison between temperature evolution when we heat up the bread for about 20 s for a point close to the cavity (red dot on the thermography, red line on the temperature evolution graph) and far away from it (blue dot on the thermography, blue line on the temperature evolution graph).

during leavening, thus, we compare temperature evolution for points near and far away from such bubbles (an example in Figure 6.6 for two selected points).

6.4 Paving the way for the inverse problem

The opportunity of substituting a void with an equivalent fictitious heat source, proven in Section 6.1.1, allows to reformulate the inverse problem of identifying voids from temperature measurements as an equivalent source estimation problem. Before the present study, this fact has been confirmed only by mere intuition and algorithmic exploration [48] which, however, made it evident that estimating the fictitious heat source turns out to be much easier than estimating the void by solving a geometric inverse problem (see e.g. [46, 48] and the references therein). Indeed, the geometric inverse problem has a nonlinear formulation and requires a two-level iterative optimization algorithm, while heat source estimation is a linear problem.

Focusing on source estimation, mathematical results about inverse heat transfer problems ensure that under suitable hypotheses, e.g. if it is known that the source expression has separable variables, it is possible to uniquely reconstruct both the temperature evolution and the source term from time-dependent boundary data (see theorems 6.1, 6.2 and 6.3 in [43]). However, in our example, we have chosen the extension as a solution of (6.13) and we obtained the source (6.21) where variables are not separable in the way required by the cited theorems, as it may often occur. To deal with generic situations, we must therefore introduce a numerical estimator that should infer the fictitious heat sources from

a restricted set of available data. More precisely, we point toward a model-based estimator, where the model is given by the equations (6.7) through the algebraic inversion of their discretization in space and time.

Here we study the behavior of this algebraic inversion of the model with respect to the obtainable degree of knowledge about the temperature field values inside the domain. The aim is to show how the information about the void boundary is at least partially retained, as the number of temperature-measurement points decreases. Let us start by supposing to know all the temperatures $\tilde{\theta}$ that satisfies (6.7) on Ω_b, i.e. all the temperatures on the physical domain plus the fictitious temperature extension inside the void region. We consider the problem (6.7) discretized in space, using FEM with Lagrangian elements P1, and in time, with implicit Euler method, that reads, at iteration n:

$$\mathbf{M}\frac{\tilde{\theta}_n - \tilde{\theta}_{n-1}}{\Delta t} + \mathbf{K}\tilde{\theta}_n = f_n \qquad \Rightarrow \qquad \underbrace{(I + \Delta t\ \mathbf{M}^{-1}\mathbf{K})}_{\mathbf{A}}\,\tilde{\theta}_n = \tilde{\theta}_{n-1} + \underbrace{\Delta t\ \mathbf{M}^{-1}}_{\mathbf{B}}\,f_n \tag{6.22}$$

where $\mathbf{M}$ and $\mathbf{K}$ are the mass and stiffness matrices of the FEM discretization, Δt is the time step chosen in the time-discretization and f_n is the fictitious heat source at time t_n. Therefore, we get the estimate $\hat{f}_n$:

$$\mathbf{B}\ \hat{f}_n = \mathbf{A}\ \tilde{\theta}_n - \tilde{\theta}_{n-1}. \tag{6.23}$$

It is possible to compute f_n, at every time instant t_n, by inverting the previous algebraic relations:

$$\hat{f}_n = \mathbf{B}^{-1}\left(\mathbf{A}\ \tilde{\theta}_n - \tilde{\theta}_{n-1}\right).$$

Notice that, thanks to the correspondence between the support of the fictitious heat source and the void region, solving the previous system allows to identify and localize the void, if its solution approximates the fictitious heat source with sufficient accuracy (see Figure 6.7(a–c)).

As a further step, let us now assume that, as happens in a real application, we do not have complete knowledge of all measurements $\tilde{\theta}$ because, for example, not all temperatures are detectable by the physical instruments.

Precisely, in Figure 6.7(c) we suppose to know all the temperatures on the physical domain Ω_d, in Figure 6.7(d) we suppose to know all the temperatures on the extended domain Ω_b at mesh nodes with a y coordinate $y \leq 0.32$, in Figure 6.7(e) with $y \leq 0.24$ and in Figure 6.7(f) with $y \leq 0.08$. Note that this method gives the projection of the void/source support on the domain region where temperatures are known. This is not surprising, anyway: when we drop the unknown temperatures from $\tilde{\theta}$, the right-hand-side of (6.23) remains sparse and localized within the region of the retained components of $\tilde{\theta}$, i.e. the region where the temperatures are known, thanks to the typical sparsity pattern of finite element matrices and for the same reasons it holds also for the estimated heat source $\hat{f}_n$.

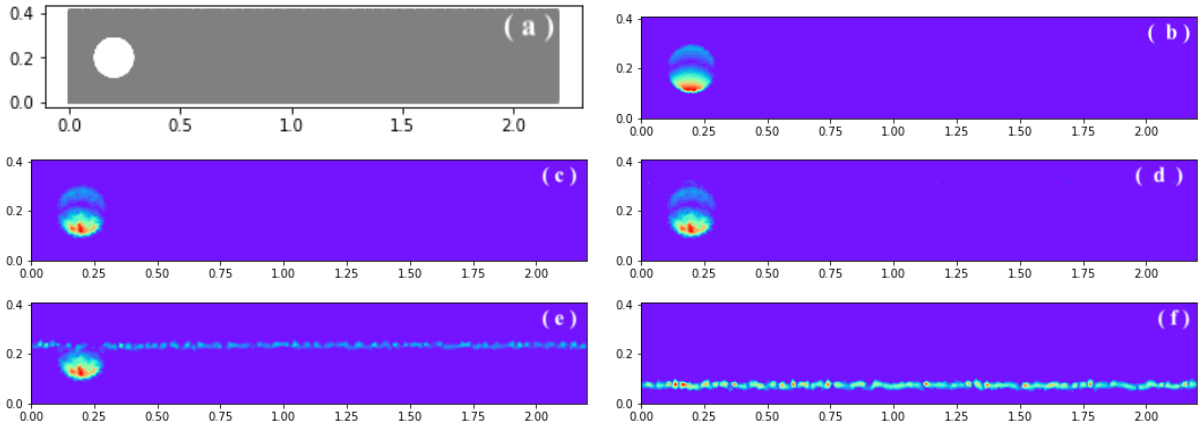

Figure 6.7: The rectangular domain $\Omega_b : [0.0, 2.2] \times [0.0, 0.41]$ (a), the absolute value of the fictitious source term $|f(t, r, \theta)|$ (b) and its algebraic reconstruction with the sparsity constraint in different conditions as a rainbow color map (max value = red, min value = blue): when all temperatures on Ω_d are known (c), when all temperatures on Ω_b with $y \leq 0.32$ are known (d), when all temperatures on Ω_b with $y \leq 0.24$ are known (e), when all temperatures on Ω_b with $y \leq 0.08$ are known (f).

Finally, to enforce $\hat{f}_n$ to have compact support, we have imposed the sparsity constraint [50] to solve the under-determined system resulting from pre-multiplying both terms of equation (6.23) by the inverse of the matrix $\mathbf{A}$ and dropping the equations corresponding to the unknown temperatures:

$$\begin{aligned}
\mathbf{B}\hat{f}_{v,n} &= \mathbf{A}\tilde{\theta}_n - \tilde{\theta}_{n-1} \\
\mathbf{A}^{-1}\mathbf{B}\hat{f}_{v,n} &= \mathbf{A}^{-1}\mathbf{A}\tilde{\theta}_n - \mathbf{A}^{-1}\tilde{\theta}_{n-1} \\
\mathbf{A}^{-1}\mathbf{B}\hat{f}_{v,n} &= \tilde{\theta}_n - \mathbf{A}^{-1}\tilde{\theta}_{n-1}
\end{aligned}$$

Notice that the subscript v on the force term stands for "void region" because of the support of f_n. Moreover, $\hat{\mathbf{A}}^{-1}$ and $\hat{\theta}_n$ indicate the matrix $\mathbf{A}^{-1}$ and the vector of temperatures $\tilde{\theta}_n$ both reduced with respect to the unknown measurements.

$$\hat{\mathbf{A}}^{-1} \cdot \mathbf{B}\hat{f}_{v,n} = \hat{\theta}_n - \hat{\mathbf{A}}^{-1}\tilde{\theta}_{n-1}. \tag{6.24}$$

The best results have been obtained with a variant [51] of the well-known Lawson–Hanson algorithm, for its better robustness against not-too-well conditioned matrices and not-too-sparse problems, compared to greedy algorithms for sparse recovery, like the Orthogonal Matching Pursuit [50]. However, the reformulation (6.24) with the sparsity constraint was not able to significantly reconstruct the void/source support outside the domain region where temperatures are known, thus again confirming the need to reconstruct the full field of temperatures, for instance with a Kalman Filter [52].

Chapter 7

Leavening monitoring: an algorithm for solving source inverse problems

In these sections, we would like to introduce the inverse problems, their structure, and their intrinsic issue. Moreover, we recall the Kalman filter and describe its augmented version. We are interested in source inverse problems to calibrate the parameters of our leavening model (e.g., the elastic stiffness), which change their values during the process, for some references we address to [43] and [53]. The chapter will conclude with the main idea of our future study.

7.1 Classification of inverse problems

Many important real-world applications give rise to an inverse problem, which entails recovering unknown causes (question) based on observation of their effects (answer). This is in contrast with the corresponding direct problem, whose solution involves predicting the effects (answer) based on a complete description of their causes (question). Inverse problems are more difficult to address than their direct problem counterpart because, in general, they are ill-posed i.e. the solution either does not exist, is not unique or does not depend continuously on the input data. Nevertheless, uniqueness is relevant in situations where we search for the cause that generates an observed effect. On the other hand, if we search for a cause giving rise to a desired effect then the non-uniqueness does not matter but the existence is meaningful, as it is important to be able to search for a desired effect.

The background needed to formulate and solve inverse problems consists of mainly mathematical analysis and algebra, and it uses techniques from functional analysis, ordinary/partial differential equations, etc. Moreover, to realize the solution to many applied problems in an impact-full way, tools of numerical analysis and scientific computing are also needed as summarized in [43] and reported here :

1. Mathematical Modelling (inverse problem formulation including governing PDEs, geometry, initial and boundary conditions, sources and coefficients, additional information that is available or needs to be supplied).
2. Mathematical Analysis (existence, uniqueness, stability).
3. Numerical Analysis (convergence, regularization, error estimates).
4. Computational Analysis (numerical implementation in Python, Matlab, etc).
5. Experimental Analysis (inversion of raw experimental data supplied by laboratory or industry).
6. Design (optimal placement of sensors and duration of measurements, suggestions for radical changes in equipment and installation).

In general, the model underlying the inverse problem rises from a physical-mathematical model that can be classified as:

- concentrated parameters model, where the system is described by components that do not have a spatial distribution but are assigned to a material point;
- distributed parameters model, where the physical proprieties are locally defined, thus the answers of the model also have a physical meaning/interpretation;
- a combination of the previous two models, where usually, a complex system has the more relevant part modeled as a distributed system and the latter part as a concentrated model.

A distributed parameters model is generally used by the continuum mechanics and is typically governed by PDEs, for which there exist the following types of inverse problems:

- inverse initial-value problems;
- inverse material-coefficient identification problems;
- inverse source estimate problems;
- inverse boundary-value problems;
- state-space estimate problems.

In our case, we will deal with source inverse problems, justified by the treatment of Chapter 6.

7.2 Need for regularization

Regularization is a necessary step to solve ill-posed problems, especially when the data are affected by noise. In this section, we recall and fix notations that are useful in the treatment of inverse problems, especially ill-posed systems, and we illustrate a possible regularization strategy.

Let us consider the discretization of a linear inverse and ill-posed problem that is reduced to solving an ill-conditioned system of $M \times N$ linear algebraic equations

$$\mathbf{A}\boldsymbol{x} = \boldsymbol{y}. \tag{7.1}$$

Then, a naive approach is based on the straightforward inversion in the case of determined systems when $M = N$

$$\boldsymbol{x} = \mathbf{A}^{-1}\boldsymbol{y}, \tag{7.2}$$

at least-squares solution in case of over-determined systems when $M > N$

$$\boldsymbol{x} = (\mathbf{A}^{\top}\mathbf{A})^{-1}\mathbf{A}^{\top}\boldsymbol{y}, \tag{7.3}$$

or the Moore-Penrose minimal-norm solution in case of under-determined systems when $M < N$

$$\boldsymbol{x} = \mathbf{A}^{+}\boldsymbol{y}, \tag{7.4}$$

where $\mathbf{A}^{+} = \lim_{\lambda\to 0}(\mathbf{A}^{\top}\mathbf{A} + \lambda\mathbb{1}_N)^{-1}\mathbf{A}^{\top}$ denotes the pseudo-inverse of the matrix $\mathbf{A}$ and $\mathbb{1}_N$ is the identity matrix of order N would lead to an unstable solution due to the ill-conditioning of the matrix $\mathbf{A}$ characterized by a large condition number $cond(\mathbf{A})$ (herein defined as the ratio between the largest and the smallest singular value of $\mathbf{A}$).

The instability appears with unbounded and highly oscillatory solutions when the input data $\mathbf{A}$ and/or $\boldsymbol{y}$ are contaminated with noise $\delta > 0$ and/or $\varepsilon > 0$ that is due to any measurement campaign,

$$\|\mathbf{A}^{\delta} - \mathbf{A}\| \leq \delta, \qquad \|\boldsymbol{y}^{\varepsilon} - \boldsymbol{y}\| \leq \varepsilon. \tag{7.5}$$

In this case, we have to solve the perturbed system

$$\mathbf{A}^{\delta}\boldsymbol{x}^{\delta,\varepsilon} = \boldsymbol{y}^{\varepsilon}. \tag{7.6}$$

Thus, solving (7.1) and (7.6) implies a relative error for which holds the estimate

$$\frac{\|\boldsymbol{x}^{\delta,\varepsilon} - \boldsymbol{x}\|}{\|\boldsymbol{x}\|} \leq \frac{1}{1 - cond(\mathbf{A})\|\delta\|/\|\mathbf{A}\|} cond(\mathbf{A}) \left(\frac{\delta}{\|\mathbf{A}\|} + \frac{\varepsilon}{\|\boldsymbol{y}\|}\right). \tag{7.7}$$

This estimate shows how the instability increases with the condition number of the matrix $\mathbf{A}$ in (7.1).

For the over-determined case, the least-squares solution of (7.1), based on minimizing $\|\mathbf{A}\boldsymbol{x} - \boldsymbol{y}\|^2$ that leads to $\boldsymbol{x} = (\mathbf{A}^{\top}\mathbf{A})^{-1}\mathbf{A}^{\top}\boldsymbol{y}$ will be unstable, manifesting highly unbounded oscillations, as the dimensions of the system of equations increase.

7.2.1 Tikhonov regularization.

A possibility is to penalize the least-square functional by adding a regularization term to it. This procedure imposes an extra smoothness on the solution e.g., smoothness of order $\mathcal{C}^k$ with $k \in \mathbb{N}$ means minimizing

$$T_{\lambda}(\boldsymbol{x}) := \|\mathbf{A}\boldsymbol{x} - \boldsymbol{y}\|^2 + \lambda\|\mathbf{R}_k\boldsymbol{x}\|^2, \tag{7.8}$$

where $\lambda > 0$ is a regularization parameter to be prescribed and $\mathbf{R}_k$ is a regularizing derivative matrix given by

$$\mathbf{R}_0 = \mathbb{1}_N \in \mathbb{R}^{N\times N} \tag{7.9}$$

$$\mathbf{R}_1 = \begin{pmatrix} 1 & -1 & & \\ & \ddots & \ddots & \\ & & 1 & -1 \end{pmatrix} \in \mathbb{R}^{(N-1)\times N} \tag{7.10}$$

$$\mathbf{R}_2 = \begin{pmatrix} 1 & -2 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & 1 & -2 & 1 \end{pmatrix} \in \mathbb{R}^{(N-2)\times N} \tag{7.11}$$

etc.

Therefore, the minimization of $T_\lambda(\boldsymbol{x})$ leads to the solution

$$\boldsymbol{x}_\lambda = (\mathbf{A}^\top\mathbf{A} + \lambda\mathbf{R}_k^\top\mathbf{R}_k)^{-1}\mathbf{A}^\top\boldsymbol{y}^\varepsilon. \tag{7.12}$$

Finally, the regularization parameter λ can be chosen according to one of the following criteria:

- The discrepancy principle criterion selects $\lambda = \lambda(\varepsilon) > 0$ for which

$$\|\mathbf{A}\,\boldsymbol{x}_{\lambda(\varepsilon)} - \boldsymbol{y}^\varepsilon\| \approx \varepsilon. \tag{7.13}$$

- The generalized cross-validation (GSV) criterion minimizes over $\lambda > 0$, the GCV functional

$$GCV(\lambda) = \frac{\|\mathbf{A}\,\boldsymbol{x}_\lambda - \boldsymbol{y}^\varepsilon\|^2}{\operatorname{tr}\mathbb{1} - \mathbf{A}(\mathbf{A}^\top\mathbf{A} + \lambda\mathbf{R}_k^\top\mathbf{R}_k)^{-1}\mathbf{A}^\top{}^2}. \tag{7.14}$$

- The L-curve criterion is based on plotting, on a log-log graph, the residual norm $\|\mathbf{A}\,\boldsymbol{x}_\lambda - \boldsymbol{y}^\varepsilon\|$ versus the solution norm $\|\mathbf{R}_k\boldsymbol{x}_\lambda\|$, for various values of $\lambda > 0$. If the resulting curve has an L-shape, then λ can be chosen at the corner of this L-curve. Since this choice does not depend on the amount of noise, an L-curve is not always obtained.

7.3 Kalman filter

In this section, we briefly present the Kalman filter (KF) [53–55], how it works and a possible extension such as the augmenting KF. To this goal, we fix the notation for linear system written in the state-space form [56].

7.3.1 State-space representation

Let us now consider the case when the system is governed by linear differential equations, and it is represented with a state-space model.

$$\dot{\boldsymbol{x}}(t) = \mathbf{A}_c\boldsymbol{x}(t) + \mathbf{B}_c\boldsymbol{u}(t) \tag{7.15}$$
$$\boldsymbol{y}(t) = \mathbf{C}_c\boldsymbol{x}(t) + \mathbf{D}_c\boldsymbol{u}(t), \tag{7.16}$$

where $\boldsymbol{x}$ is a time-dependent vector as $\boldsymbol{u}$ and $\boldsymbol{y}$ which are the state vector, the input vector and the output vector respectively. $\mathbf{A}_c, \mathbf{B}_c, \mathbf{C}_c$ and $\mathbf{D}_c$ are suitable matrices (where the subscripts c stands for continuum model), beside the order of the model is given by the number of components of the state vector.

Clearly, the previous continuum model can be discretized with respect to the time variable, leading to the following form

$$\boldsymbol{x}(k+1) = \mathbf{A}\boldsymbol{x}(k) + \mathbf{B}\boldsymbol{u}(k) \tag{7.17}$$
$$\boldsymbol{y}(k) = \mathbf{C}\boldsymbol{x}(k) + \mathbf{D}\boldsymbol{u}(k), \tag{7.18}$$

where $\boldsymbol{x}(k)$, $\boldsymbol{u}(k)$ and $\boldsymbol{y}(k)$ are the state, the input and the output vector respectively.

In this setting, the Markov's coefficients for (7.17) $\mathbf{G}_k$ are defined as

$$\mathbf{G}_0 = \mathbf{D} \tag{7.19}$$
$$\mathbf{G}_k = \mathbf{C}\,\mathbf{A}^{k-1}\mathbf{B} \qquad k = 1, 2, \dots \tag{7.20}$$

and, they correspond to the impulse response to the discrete system (i.e. if we apply the discrete impulse to the i-input we get $\boldsymbol{h}^{(i)}(0) = \mathbf{D}(:, i)$ and $\boldsymbol{h}^{(i)}(k) = \mathbf{C}\,\mathbf{A}^{k-1}\mathbf{B}(:, i)$).

Stochastic state-space representation

Many applications involve the stochastic state-space model

$$\boldsymbol{x}(k+1) = \mathbf{A}\boldsymbol{x}(k) + \mathbf{B}\boldsymbol{u}(k) + \boldsymbol{v}(k) \tag{7.21}$$
$$\boldsymbol{y}(k) = \mathbf{C}\boldsymbol{x}(k) + \mathbf{D}\boldsymbol{u}(k) + \boldsymbol{w}(k), \tag{7.22}$$

which differs from the previous one (7.17) for the presence of noises. $\boldsymbol{v}(k)$ and $\boldsymbol{w}(k)$ are the model and measure noise respectively of zero-mean and covariance matrices:

$$\mathbb{E}\left\{\begin{pmatrix}\boldsymbol{v}(k)\\ \boldsymbol{w}(k)\end{pmatrix}\cdot\begin{pmatrix}\boldsymbol{v}(k)^\top & \boldsymbol{w}(k)^\top\end{pmatrix}\right\} = \begin{pmatrix}\mathbf{Q} & \mathbf{S}\\ \mathbf{S}^\top & \mathbf{R}\end{pmatrix}, \tag{7.23}$$

where $\mathbf{Q}$ and $\mathbf{R}$ are positive-definite.

In this setting, we can re-formulates the inverse problem as a state-space estimate starting from some measurements of the outputs and of the inputs, instead in Section 7.3.2 we will include the dynamic of the unknown input, with the state variable.

Indeed, firstly we want to get the minimum variance estimate of the vector $\boldsymbol{x}(k)$ of stochastic variables in (7.21), given the measurements $\boldsymbol{y}(1), \ldots, \boldsymbol{y}(j)$:

$$\hat{\boldsymbol{x}}_{k|j} = \hat{\boldsymbol{x}}_{k|\boldsymbol{y}(1),\ldots,\boldsymbol{y}(j)} \,. \tag{7.24}$$

When $j = k$ the estimate is called filter, while for $j = k - p$ it is called predictor at p-steps. The Kalman Filter is a recursive algorithm that solves this problem. We will derive the equation of the Kalman Filter by applying Newton's method.

Let us recall that, for the Gauss-Markov theorem [53], the unbiased linear estimator at minimum variance for the standard linear model

$$\mathbf{A}\boldsymbol{x} = \boldsymbol{b} = \bar{\boldsymbol{b}} + \varepsilon, \qquad \varepsilon \sim N(0; \sigma^2), \tag{7.25}$$

is given by the solution of the normal equations. The covariance matrix of the parameter estimation error is:

$$\mathbf{C}_{\hat{x}} = \sigma^2 (\mathbf{A}^\top \mathbf{A})^{-1} \tag{7.26}$$

and that for the generalized Gauss-Markov model we can trace back to the standard case, if $\mathbf{W}$ is the covariance matrix, positive definite and

$$\mathbf{W} = \mathbf{B}\mathbf{B}^\top \tag{7.27}$$

is its Cholesky factorization, and the problem becomes:

$$\hat{\boldsymbol{x}} = \operatorname{argmin}_{\boldsymbol{x}} \|\mathbf{B}^{-1}(\boldsymbol{b} - \mathbf{A}\boldsymbol{x})\|_2 = \operatorname{argmin}_{\boldsymbol{x}} (\boldsymbol{b} - \mathbf{A}\boldsymbol{x})^\top \mathbf{W}^{-1} (\boldsymbol{b} - \mathbf{A}\boldsymbol{x}). \tag{7.28}$$

Now, let us consider that, applying the Newton method to a general least-square problem of the form:

$$f(\boldsymbol{x}) = \|\mathbf{A}\boldsymbol{x} - \boldsymbol{b}\|_2^2 = (\mathbf{A}\boldsymbol{x} - \boldsymbol{b})^\top (\mathbf{A}\boldsymbol{x} - \boldsymbol{b}) \tag{7.29}$$

that has gradient $\mathbf{A}^\top \mathbf{A}\boldsymbol{x} - \mathbf{A}^\top \boldsymbol{b}$ and Hessian $\mathbf{A}^\top \mathbf{A}$, we have, at the generic $(k+1)$-th iteration:

$$\boldsymbol{x}(k+1) = (\mathbf{A}^\top \mathbf{A})^{-1} \mathbf{A}^\top \boldsymbol{b} \tag{7.30}$$

which is independent by $\boldsymbol{x}(k)$.

Let us see the KF equations in the case of uncorrelated noise, i.e. $\mathbf{S} = 0$ in (7.23), and suppose that the initial state is $\boldsymbol{x}(0) = \boldsymbol{\mu}(0) + \boldsymbol{v}(0)$, and $\boldsymbol{\mu}(0)$ is known. Let us write a

linear system by writing the equations (7.21)

$$\begin{aligned} \boldsymbol{\mu}(0) &= \boldsymbol{x}(0) - \boldsymbol{v}(0) \\ \mathbf{B}\boldsymbol{u}(0) &= \boldsymbol{x}(1) - \mathbf{A}\boldsymbol{x}(0) - \boldsymbol{v}(1) \\ \boldsymbol{y}(1) &= \mathbf{C}\boldsymbol{x}(1) + \boldsymbol{w}(1) \\ &\vdots \\ \mathbf{B}\boldsymbol{u}(k) &= \boldsymbol{x}(k+1) - \mathbf{A}\boldsymbol{x}(k) - \boldsymbol{v}(k+1) \\ \boldsymbol{y}(k+1) &= \mathbf{C}\boldsymbol{x}(k+1) + \boldsymbol{w}(k+1) \end{aligned} \tag{7.31}$$

and indicate them in matrix form:

$$\boldsymbol{b}_{k+1|k+1} = \mathbf{A}_{k+1|k+1}\boldsymbol{z}_{k+1} + \boldsymbol{\varepsilon}_{k+1|k+1}, \tag{7.32}$$

that is a Gauss-Markov generalized model, where

- $\mathbf{A}_{k+1|k+1}$ has full column rank,
- $\boldsymbol{z}_k = (\boldsymbol{x}(0), \boldsymbol{x}(1), \ldots, \boldsymbol{x}(k)) \in \mathbb{R}^{(k+1)n}$
- $\boldsymbol{\varepsilon}_{k+1|k+1}$ is a vector of stochastic variables whose covariance is the following block matrix:
$$\mathbf{W}_{k+1|k+1} = diag(\mathbf{Q}, \mathbf{Q}, \mathbf{R}, \ldots, \mathbf{Q}, \mathbf{R}). \tag{7.33}$$

Let us define the following cost function:

$$\begin{aligned} J_{k+1|k+1}(\boldsymbol{z}_{k+1}) &= \frac{1}{2}\|\mathbf{A}_{k+1|k+1}\boldsymbol{z}_{k+1} - \boldsymbol{b}_{k+1|k+1}\|^2_{\mathbf{W}_{k+1|k+1}} \\ = \frac{1}{2}(\|\boldsymbol{x}(0) - \boldsymbol{\mu}(0)\|^2_{\mathbf{Q}} &+ \sum_{i=1}^{k+1}\|\boldsymbol{y}(i) - \mathbf{C}\boldsymbol{x}(i)\|^2_{\mathbf{R}} + \sum_{i=1}^{k+1}\|\boldsymbol{x}(i) - \mathbf{A}\boldsymbol{x}(i-1) - \mathbf{B}\boldsymbol{u}(i)\|^2_{\mathbf{Q}}) \end{aligned} \tag{7.34}$$

whose minimum is given by the weighted least squares:

$$\hat{\boldsymbol{z}}_{k+1|k+1} = (\mathbf{A}^\top_{k+1|k+1}\mathbf{W}^{-1}_{k+1|k+1}\mathbf{A}_{k+1|k+1})^{-1}\mathbf{A}^\top_{k+1|k+1}\mathbf{W}^{-1}_{k+1|k+1}\boldsymbol{b}_{k+1|k+1} \tag{7.35}$$

which is our estimator, whose error variance is

$$\mathbb{E}\left((\hat{\boldsymbol{z}}_{k+1|k+1} - \boldsymbol{z}_{k+1})(\hat{\boldsymbol{z}}_{k+1|k+1} - \boldsymbol{z}_{k+1})^\top\right) = (\mathbf{A}^\top_{k+1|k+1}\mathbf{W}^{-1}_{k+1|k+1}\mathbf{A}_{k+1|k+1})^{-1}. \tag{7.36}$$

Let us formulate the previous in a recursive way, by defining $\mathbf{F}_k = (0, \ldots, 0, \mathbf{A}) \in \mathbb{R}^{n \times kn}$ such that

$$\mathbf{F}_{k+1}\boldsymbol{z}_k = \mathbf{A}\boldsymbol{x}(k) \tag{7.37}$$

and the cost function $J_{k+1|k}(\boldsymbol{z}_{k+1})$ as

$$J_{k+1|k}(\boldsymbol{z}_{k+1}) = J_{k|k}(\boldsymbol{z}_k) + \frac{1}{2}\|\boldsymbol{x}(k+1) - \mathbf{F}_{k+1}\boldsymbol{z}_k - \mathbf{B}\boldsymbol{u}(k)\|^2_{\mathbf{Q}}. \tag{7.38}$$

Thus, one Newton iteration gives the minimum, for an arbitrary $\boldsymbol{z}_{k+1}$:

$$\hat{\boldsymbol{z}}_{k+1|k} = \boldsymbol{z}_{k+1} - \nabla^2 J_{k+1|k}(\boldsymbol{z}_{k+1})^{-1} \nabla J_{k+1|k}(\boldsymbol{z}_{k+1}). \tag{7.39}$$

Actually, since by definition of $\hat{\boldsymbol{z}}_{k|k}$ we have $\nabla J_{k|k}(\hat{\boldsymbol{z}}_{k|k}) = 0$ and $\mathbf{F}_{k+1}\hat{\boldsymbol{z}}_{k|k} = \mathbf{A}\hat{\boldsymbol{x}}_{k|k}$, a convenient choice for $\boldsymbol{z}_{k+1}$ is:

$$\boldsymbol{z}_{k+1} = \begin{pmatrix} \hat{\boldsymbol{z}}_{k|k} \\ \mathbf{A}\hat{\boldsymbol{x}}_{k|k} + \mathbf{B}\boldsymbol{u}(k) \end{pmatrix}, \tag{7.40}$$

that gives $\nabla J_{k+1|k}(\boldsymbol{z}_{k+1}) = 0$ and, therefore, the best estimate of $\boldsymbol{x}_{k+1}$ given the measurements $\boldsymbol{y}_i$ is the last row of $\boldsymbol{z}_{k+1}$. Proceeding in a similar way when we get the measurement $\boldsymbol{y}(k+1)$, we arrive to the Kalman Filter.

In short, the Kalman Filter is given by the following equations:

- time update:

$$\hat{\boldsymbol{x}}(k+1|k) = \mathbf{A}\,\hat{\boldsymbol{x}}(k|k) + \mathbf{B}\,\boldsymbol{u}_{known}(k) \tag{7.41}$$

$$\mathbf{P}(k+1|k) = \mathbf{A}\,\mathbf{P}(k|k)\mathbf{A}^\top + \mathbf{Q} \tag{7.42}$$

- measurements update:

$$\hat{\boldsymbol{x}}(k+1|k+1) = \hat{\boldsymbol{x}}(k+1|k) + \mathbf{L}(k+1)(\boldsymbol{y}(k+1) - \mathbf{C}\hat{\boldsymbol{x}}(k+1|k)) \tag{7.43}$$

$$\mathbf{P}(k+1|k+1) = \mathbf{P}(k+1|k) - \mathbf{P}(k+1|k)\mathbf{C}^\top \mathbf{\Lambda}(k+1)^{-1}\mathbf{C}\mathbf{P}(k+1|k) \tag{7.44}$$

  where

  - $\mathbf{\Lambda}(k) = \mathbf{C}\mathbf{P}(k+1|k)\mathbf{C}^\top + \mathbf{R}$,
  - $\mathbf{L}(k) = \mathbf{P}(k+1|k)\mathbf{C}^\top \mathbf{\Lambda}^{-1}(k)$,
  - $\mathbf{P}(k+1|k)$ and $\mathbf{P}(k|k)$ are, respectively, the covariance matrices of the prediction error and the filtering error.

The theory of KF states that the whiteness of the prediction error is a sufficient condition for the optimality of the KF. From this point of view, the Bartlett white-noise test may be useful.

7.3.2 The augmented Kalman filter and a feed-forward strategy

The Augmented Kalman Filter (AKF), which is a joint input-state estimation method relying on Bayesian recursion is used to estimate the state of the system using some measurements. For this procedure, we follow the results presented in [24]. In this section, we will deal with the heat source problem.

Starting from the concept that there cannot be any uniqueness result for source inverse problems in the class of general heat sources that depend on both space and time, a lot of effort has been made to prove uniqueness in some specific cases. For instance, a popular choice is the variable separable source, i.e.,

$$f(\boldsymbol{x}, t) := p(\boldsymbol{x})q(t). \tag{7.45}$$

Given the spatial component $p(\boldsymbol{x})$ or the temporal component $q(t)$, recovering the other unknown part is a classical field in the inverse problems as done in the paper [57]. However, [58] studied the uniqueness of determining both the two separable heat source terms which are time- and space-dependent respectively from the initial and boundary data along with two additional measurements. Instead, in the paper [59] the authors proved that the flux data from any nonempty open subset of the boundary can uniquely determine the semi-discrete source

$$f(\boldsymbol{x}, t) := \sum_{k=1}^{K} \mathbf{p}_k(\boldsymbol{x}) \chi_{\, t \in (t_{k-1}, t_k)}. \tag{7.46}$$

This means the observed area can be extremely small, and that is the reason they called it sparse boundary data. Even in the work [60] it is identified the source by using the flux boundary data, for point sources

$$f(\boldsymbol{x}, t) = \sum_{k=1}^{K} \delta_{S_k}(\boldsymbol{x}) \lambda_k(t), \tag{7.47}$$

where δ_{S_k} stands for the Dirac distribution at the point S_k and both S_k, λ_k are unknown. Moreover, in the paper [61] the source is assumed as the sum of two unknown functions of spatial variables multiplied for functions which decade exponentially in time

$$f(\boldsymbol{x}, t) = \mathbf{p}_1(\boldsymbol{x}) e^{-\mu_1 t} + \mathbf{p}_2(\boldsymbol{x}) e^{-\mu_2 t}. \tag{7.48}$$

The inverse problem is stated as determining two unknown functions of spatial variables from additional information on the solution of the initial-boundary value problem.

It is proved that the solution of the inverse problem is unique in the class of sufficiently smooth compactly supported functions such that the supports of the unknown functions do not intersect. This result is extended to the case of a source involving an arbitrary finite number of unknown functions of spatial variables multiplied by exponentially decaying functions of time.

Problem settings for Kalman filter estimation of the forcing term

As a model problem of a diffusive process, let us consider the heat equation

$$\begin{cases} \rho c \, \frac{\partial}{\partial t}\theta = \beta \, \Delta\theta + \mathcal{F}, & \text{on } \Omega \times (0, \tau) \\ \beta \, \nabla\theta \cdot \boldsymbol{n} = g, & \text{on } \Gamma \times (0, \tau) \\ \beta \, \nabla\theta \cdot \boldsymbol{n} = 0, & \text{on } \partial\Omega/\Gamma \times (0, \tau) \\ \theta(0, \cdot) = \theta_0(\cdot), & \text{on } \Omega. \end{cases} \tag{7.49}$$

and suppose that the heat source term $\mathcal{F}$ is an unknown function, in general, except that it is assumed different from zero only in a few disconnected regions of compact support.

We want to estimate $\mathcal{F}$ from a limited number of temperature measurements $\tilde{\theta}$, typically taken at the boundary. The estimate of $\mathcal{F}$ can be seen as an indirect measurement of $\mathcal{F}$ from physical temperature measurements and we do this by exploiting the combination of the physico-mathematical model (7.49) and an abstract, data-driven model to describe $\mathcal{F}$. In applications, the resulting method may be called a physics-aware soft-sensor[12].

Let us consider problem (7.49), discretized in space using FEM with Lagrangian elements P1

$$\theta(\boldsymbol{x},t)=\sum_{i=1}^{N}\theta_i(t)\,\phi_i(\boldsymbol{x}) \qquad \mathcal{F}(\boldsymbol{x},t)=\sum_{i=1}^{N}f_i(t)\,\phi_i(\boldsymbol{x}) \tag{7.50}$$

$$\sum_{i=1}^{N}\int_\Omega \rho c\frac{\partial}{\partial t}\theta_i(t)\phi_i\,\phi_j d\boldsymbol{x}+\sum_{i=1}^{N}\int_\Omega \beta\theta_i(t)\nabla\phi_i\nabla\phi_j d\boldsymbol{x}=\sum_{i=1}^{N}\int_\Omega f_i(t)\phi_i\phi_j d\boldsymbol{x}+\int_\Gamma g\,\phi_j ds \tag{7.51}$$

and, defining:

$$\mathbf{K}_{ij}:=\int_\Omega \nabla\phi_i\nabla\phi_j d\boldsymbol{x} \qquad \mathbf{M}_{ij}:=\int_\Omega \phi_i\phi_j d\boldsymbol{x} \qquad \mathbf{G}_{jk}:=\int_\Gamma g(\boldsymbol{x},t_k)\phi_j ds. \tag{7.52}$$

Time discretization with the implicit Euler method, at iteration k reads:

$$\rho c\mathbf{M}\frac{\theta(k)-\theta(k-1)}{\Delta t}+\beta\mathbf{K}\theta(k)=\mathbf{M}f(k)+\mathbf{G}_{:k} \tag{7.53}$$

$$(\rho c\mathbb{1}+\Delta t\,\beta\mathbf{M}^{-1}\mathbf{K})\theta(k)=\rho c\,\theta(k-1)+\Delta t\,\mathbb{1}f(k)+\Delta t\;\mathbf{M}^{-1}\mathbf{G}_{:k} \tag{7.54}$$

where $\mathbf{M}\in R^{n\times n}$ and $\mathbf{K}\in R^{n\times n}$ are the mass and stiffness matrices of the FEM discretization, Δt is the time step chosen in the time-discretization and $\mathbf{G}_{:k}$ is the boundary flux, at time t_k.

If we knew all the values of θ then the computation of the unknowns f would be a simple algebraic reconstruction as done in Section 6.4. Since we can measure only a few components of θ, we can reformulate this model as a state-space dynamical system to estimate its state from output measurements. Let us consider the following state-space discrete model in physical coordinates:

$$\begin{aligned}\boldsymbol{x}_m(k+1)&=\rho c\mathbf{A}_m\boldsymbol{x}_m(k)+\Delta t\mathbf{A}_m f(k)+\mathbf{B}_m\boldsymbol{u}_m(k)+\boldsymbol{v}_m(k)\\ \boldsymbol{y}_m(k)&=\mathbf{C}_m\boldsymbol{x}_m(k)+\boldsymbol{w}(k)\end{aligned} \tag{7.55}$$

where

- $\boldsymbol{x}_m(k)=\theta(k)$,

- $\boldsymbol{u}_m(k) = \mathbf{G}_{:k}$,
- $\mathbf{A}_m = (\rho c \mathbb{1} + \Delta t\, \beta \mathbf{M}^{-1}\mathbf{K})^{-1}$,
- $\mathbf{B}_m = \mathbf{A}_m \Delta t \mathbf{M}^{-1} = (\rho c \mathbb{1} + \Delta t\, \beta \mathbf{M}^{-1}\mathbf{K})^{-1} \Delta t \mathbf{M}^{-1}$,
- $\mathbf{C}_m$ is a matrix where rows of the identity matrix corresponding to measured nodes,
- $\boldsymbol{v}_m(k)$ is a stochastic term for modelling errors and $\boldsymbol{w}(k)$ for measurement errors, both supposed uncorrelated Gaussian noise [62].

Now, we augment the state of (7.55) by adding a model for the forcing term $f(k)$:

$$\boldsymbol{x}(t) = \begin{pmatrix} \boldsymbol{x}_f \\ \boldsymbol{x}_m \end{pmatrix}, \quad \boldsymbol{u}(t) = \begin{pmatrix} \mathbf{G}_{:k} \end{pmatrix} \tag{7.56}$$

where $\boldsymbol{x}(t)$ is the augmented state vector, $\boldsymbol{x}_f = f(k)$, $\boldsymbol{u}(t)$ the new input vector, which can be now omitted, $\boldsymbol{y}(t)$ the output vector and:

$$\mathbf{A} = \begin{pmatrix} \mathbf{A}_f & 0 \\ \mathbf{Z} & \rho c \mathbf{A}_m \end{pmatrix}, \quad \mathbf{B} = \begin{pmatrix} \mathbf{B}_m \end{pmatrix}, \quad \mathbf{C} = \begin{pmatrix} 0 & \mathbf{C}_m \end{pmatrix}, \tag{7.57}$$

where $\mathbf{A}_f = \mathbb{1}$, $\mathbf{Z} = \Delta t \mathbf{A}_m \mathbf{C}_u$, $\mathbf{C}_u = \mathbb{1}$, and we obtain the augmented state-space model:

$$\begin{aligned} \boldsymbol{x}(k+1) &= \mathbf{A}\boldsymbol{x}(k) + \mathbf{B}u(k) + \boldsymbol{v}(k) \\ \boldsymbol{y}(k) &= \mathbf{C}\boldsymbol{x}(k) + \boldsymbol{w}(k) \end{aligned} \tag{7.58}$$

Now, to estimate the state vector it is a common choice to adopt a KF (Section 7.3). In this case, the state vector $\boldsymbol{x}(k)$ is the concatenation of the temperatures θ computed at the mesh nodes and the source term f at the same nodes

$$\boldsymbol{x}(k) = \begin{pmatrix} f(k) \\ \theta(k) \end{pmatrix}, \tag{7.59}$$

while the vector $y(k)$ contains the measured temperatures. In the problem at hand, the reference model for the KF is the union of a constant model for the heating source (this is the usual choice made in the literature for this kind of problem) and the FEM discretization of (7.49), and we set the covariance matrix of the model error as

$$\mathbf{Q}(k) = \begin{pmatrix} \mathbf{Q}_f(k) & \mathbf{Q}^{\theta}_{mf}(k) \\ \mathbf{Q}_{mf}(k) & \mathbf{Q}_m(k) \end{pmatrix}, \tag{7.60}$$

where $\mathbf{Q}_f(k) = \sigma^2_{\mathbf{Q}_f}\ \mathbb{1}$, $\mathbf{Q}_m(k) = \sigma^2_{\mathbf{Q}_m}\ \mathbb{1}$ and $\mathbf{Q}_{mf}(k) = \sigma^2_{\mathbf{Q}_{mf}} \mathbb{1}$ in general, since we don't know the location of the forcing term. Then, set the initial covariance matrix of the state-estimation error as

$$\mathbf{P}(0) = \begin{pmatrix} \mathbf{P}_f(0) & \mathbf{P}_{mf}(0)^{\top} \\ \mathbf{P}_{mf}(0) & \mathbf{P}_m(0) \end{pmatrix} \tag{7.61}$$

where $\mathbf{P}_f(0) = \sigma^2_{\mathbf{P}_f}\mathbb{1}$ and $\mathbf{P}_m(0) = \sigma^2_{\mathbf{P}_m}\mathbb{1}$.

To use the AKF, described above, for highly diffusive problems it is convenient and necessary to exploit a feed-forward strategy as presented in [24].

Finally, we present the idea of our ongoing work: the leavening monitoring. In particular, with the aim of reconstructing the porous structure of bread arising with leavening, we will have to solve a source inverse problem due to the treatment of Chapter 6.

The starting point will be some thermographies, very similar to the ones seen in Section 6.3, obtained during an experimental phase. Therefore, to solve the source recognition problem numerically, we will need the complete temperature field. At this stage, we will want to exploit the AKF with the feed-forward strategy, which will allow an efficient rebuilding of both the temperature field and the fictitious forcing term starting from boundary data.

Conclusions

The challenges in the industrial practice today can be dealt with using DTs. With the support of such techniques, the modern industry brings a wide range of online tasks useful for making decisions, developing architectures, predicting behavior or detecting failure. It can be expected that, with the rising awareness regarding the economic benefits of adopting DT technology, it will act as the backbone of Industry 4.0.

In this doctoral thesis, we want to underline the fundamental role of mathematics in industrial applications. Especially, the final goal of this project is the building of a digital twin of bread leavening for energy saving. In this context, the reader can see that, under the technology of a DT and its use in the manufacturing field, there are several advanced mathematical topics that are developed and connected to each other to allow the building of such a real product.

The interesting thing is that this project involves very different mathematical themes which range from continuum mechanics, and machine learning theories to inverse problems. The tangible result of these theories will give rise to a real product, such as the DT, that should be able to run in industrial environments.

The first subject that must be developed is a continuum physical-mathematical model which goal is the mathematical description of the phenomena that you are looking at that again, represents the focal point of a DT.

In our case the continuum model emulates the leavening process and is governed by partial differential equations that couple the volume expansion with the temperature evolution, life cycle of yeasts and their metabolic effect therefore the production and diffusion of carbon dioxide. The theory behind it is given by the continuum mechanics, especially hyper-elasticity and diffusive phenomena.

In order to implement such a model, it is necessary to discretize it in the space and time variables. Here, the well-known Finite Element Method and Euler Method allow us to translate the continuum formulation to a discrete one that can be solved numerically. The resulting simulations represent a starting point for making energy considerations and for studying some strategies for its optimization.

A crucial step of DT is that it has to run online with a real process. This simultaneity allows, on the one hand, to take some data from the real process, captured by physics-aware soft sensors, and on the other hand, to give back some information about the system, that otherwise we would not be able to get. Indeed, DT has to run within an embedded system, such as on a microprocessor that is not able to solve the previous numerical simulations; in fact, it is necessary a surrogate model that mimics them, needs few data as input and is computationally cheaper. We chose to work with the variationally mimetic operator network (VarMiON) to answer to this need. Such a choice reflects our intent of exploiting

the physical-mathematical description, in all the components of a DT, as a matter of fact, this particular operator network predicts the result of the simulations starting from the variational formulation of the model, roughly speaking the weak forms of the PDEs that constitute the model.

Finally, another key role for DTs is played by inverse problems to calibrate the parameters of the model by using data, taken from the real system. This allows adapting the model to shadow the process that we are looking at. From the perspective of monitoring the leavening, it is useful the equivalence between voids and fictitious forces, therefore as the last step of this thesis, we try to define an algorithm of source identification. This will be the starting point for future works that will consist of estimating the porosity of bread arising during leavening.

Bibliography

1. Rinaldi, L. & Giusteri, G. G. *Modelling and simulation of bread leavening to monitor energy consumption in industrial processes* (Submitted preprint 2024).
2. Patel, D., Ray, D., Abdelmalik, M. R., Hughes, T. J. & Oberai, A. A. Variationally mimetic operator networks. *Computer Methods in Applied Mechanics and Engineering* **419,** 116536 (2024).
3. Rinaldi, L., Chinellato, E., Martin, P. & Marcuzzi, F. Exploiting scientific machine learning on embedded digital twins. *Springer series: Lectures Notes in Computational Science and Engineering - Math to Product* (submitted 2024).
4. Rinaldi, L., Giusteri, G. G. & Marcuzzi, F. Replacing voids and localized parameter changes with fictitious forcing terms in boundary-value problems. *Results in Applied Mathematics* **20,** 100402. ISSN: 2590-0374 (2023).
5. Jiang, Y., Yin, S., Li, K., Luo, H. & Kaynak, O. Industrial applications of digital twins. *Philosophical Transactions of the Royal Society A* **379,** 20200360 (2021).
6. Grieves, M. *Virtually perfect: driving innovative and lean products through product lifecycle management* (Space Coast Press Cocoa Beach, 2011).
7. Kobryn, P. A., Tuegel, E. J. & Branch, S. M. Condition-based maintenance plus structural integrity (CBM+ SI) & the airframe digital twin. *USAF Air Force Research Laboratory, 88ABW-201101428* (2011).
8. Grieves, M. & Vickers, J. Digital twin: Mitigating unpredictable, undesirable emergent behavior in complex systems. *Transdisciplinary perspectives on complex systems: New findings and approaches,* 85–113 (2017).
9. Van der Auweraer, H. & Hartmann, D. The Executable Digital Twin: merging the digital and the physics worlds. *arXiv preprint arXiv:2210.17402* (2022).
10. Juarez, M. G., Botti, V. J. & Giret, A. S. Digital twins: Review and challenges. *Journal of Computing and Information Science in Engineering* **21,** 030802 (2021).
11. Martin, D., Kühl, N. & Satzger, G. Virtual sensors. *Business & Information Systems Engineering* **63,** 315–323 (2021).
12. Chinellato, E., Pierobon, S. & Marcuzzi, F. *Physics-aware soft sensors for embedded digital twins* in *Proceedings of 9th International Congress on Information and Communication Technology (ICICT 2024)* (Springer LNNS, 2024, in press). ISBN: 978-981-99-3236-8.

13. Paton, J., Khatir, Z., Thompson, H., Kapur, N. & Toropov, V. Thermal energy management in the bread baking industry using a system modelling approach. *Applied Thermal Engineering* **53,** 340–347 (2013).

14. Al-Nasser, M, Fayssal, I & Moukalled, F. Numerical simulation of bread baking in a convection oven. *Applied Thermal Engineering* **184,** 116252 (2021).

15. Pask, F *et al.* Systematic approach to industrial oven optimisation for energy saving. *Applied thermal engineering* **71,** 72–77 (2014).

16. Purlis, E. Bread baking: Technological considerations based on process modelling and simulation. *Journal of Food Engineering* **103,** 92–102 (2011).

17. Purlis, E. Baking process design based on modelling and simulation: Towards optimization of bread baking. *Food control* **27,** 45–52 (2012).

18. Zanoni, B, Peri, C, Pierucci, S., *et al.* A study of the bread-baking process. I: A phenomenological model. *Journal of food engineering* **19,** 389–398 (1993).

19. Ravula, S. R., Arepally, D. & Datta, A. K. Estimation of the energy requirement of bread during baking by inverse heat transfer method. *Journal of Thermal Analysis and Calorimetry,* 1–15 (2023).

20. Zhang, J. & Datta, A. Mathematical modeling of bread baking process. *Journal of Food Engineering* **75,** 78–89 (2006).

21. Alnaes, M. S. *et al.* The FEniCS Project Version 1.5. *Archive of Numerical Software* **3** (2015).

22. Logg, A., Mardal, K.-A. & Wells, G. *Automated solution of differential equations by the finite element method: The FEniCS book* (Springer Science & Business Media, 2012).

23. Ray, D., Pinti, O. & Oberai, A. A. Deep Learning and Computational Physics (Lecture Notes). *arXiv preprint arXiv:2301.00942* (2023).

24. Marcuzzi, F. A Numerical Feed-Forward Scheme for the Augmented Kalman Filter. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* **14837 LNCS.** Cited by: 1, 131 – 145 (2024).

25. Gurtin, M. E. *An introduction to continuum mechanics* (Academic press, 1982).

26. Gurtin, M. E., Fried, E. & Anand, L. *The mechanics and thermodynamics of continua* (Cambridge university press, 2010).

27. Evans, L. C. *Partial differential equations* (American Mathematical Society, 2022).

28. Romano, A., Toraldo, G., Cavella, S. & Masi, P. Description of leavening of bread dough with mathematical modelling. *Journal of Food Engineering* **83,** 142–148 (2007).

29. Packkia-Doss, P. P., Chevallier, S., Pare, A. & Le-Bail, A. Effect of supplementation of wheat bran on dough aeration and final bread volume. *Journal of food engineering* **252,** 28–35 (2019).

30. Chiotellis, E & Campbell, G. M. Proving of bread dough II: measurement of gas production and retention. *Food and bioproducts processing* **81,** 207–216 (2003).

31. De Cindio, B & Correra, S. Mathematical modelling of leavened cereal goods. *Journal of food engineering* **24,** 379–403 (1995).

32. Shah, P, Campbell, G. M., McKee, S. & Rielly, C. Proving of bread dough: modelling the growth of individual bubbles. *Food and Bioproducts Processing* **76,** 73–79 (1998).

33. Brezis, H. *Functional analysis, Sobolev spaces and partial differential equations* **3** (Springer, 2011).

34. Quarteroni, A. *Numerical Models for Differential Problems* (Springer-Verlag Mailand, 2014).

35. Raissi, M., Perdikaris, P. & Karniadakis, G. E. 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).

36. Kingma, D. P. & Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

37. Frank, M. Implementation of the space time finite element method within the ad-hoc++ framework (2021).

38. Langer, U., Steinbach, O., Tröltzsch, F. & Yang, H. Space-time finite element methods for the initial temperature reconstruction. *arXiv preprint arXiv:2103.16699* (2021).

39. Von Danwitz, M., Voulis, I., Hosters, N. & Behr, M. Time-continuous and time-discontinuous space-time finite elements for advection-diffusion problems. *International Journal for Numerical Methods in Engineering* **124,** 3117–3144 (2023).

40. Antonietti, P. F., Mazzieri, I. & Migliorini, F. A space-time discontinuous Galerkin method for the elastic wave equation. *Journal of Computational Physics* **419,** 109685 (2020).

41. Jackson, J. D. *Classical electrodynamics* (John Wiley & Sons, 2021).

42. Kim, S. & Karrila, S. J. *Microhydrodynamics: principles and selected applications* (Courier Corporation, 2013).

43. Lesnic, D. *Inverse Problems with Applications in Science and Engineering* (Chapman and Hall/CRC, 2021).

44. Kirsch, A. *An Introduction to the Mathematical Theory of Inverse Problems* (Springer, 2021).

45. Bryan, K. & Caudill, L. Reconstruction of an unknown boundary portion from Cauchy data in n dimensions. *Inverse Problems* **21,** 239–255 (2005).

46. Marcuzzi, F & Marinetti, S. Efficient reconstruction of corrosion profiles by infrared thermography. *Journal of Physics: Conference Series* **124,** 012033 (2008).

47. Kazemzadeh-Parsi, M. J. & Daneshmand, F. Cavity-shape identification with convective boundary conditions using non-boundary-fitted meshes. *Numerical Heat Transfer, Part B: Fundamentals* **57,** 283–305 (2010).

48. Dessole, M. & Marcuzzi, F. Accurate detection of hidden material changes as fictitious heat sources from thermographic data. *Numerical Heat Transfer, Part B: Fundamentals* (2023).

49. Lebedev, N. N. *Special functions and their applications* (Prentice–Hall, 1965).

50. Elad, M. *Sparse and Redundant Representations: From Theory to Applications in Signal and Image Processing* (Springer Publishing Company, Incorporated, 2010).

51. Dessole, M., Dell'Orto, M. & Marcuzzi, F. The Lawson-Hanson algorithm with deviation maximization: Finite convergence and sparse recovery. *Numerical Linear Algebra with Applications,* e2490 (2023).

52. Lourens, E.-M., Reynders, E., De Roeck, G., Degrande, G. & Lombaert, G. An augmented Kalman filter for force identification in structural dynamics. *Mechanical Systems and Signal Processing* **27,** 446â–460 (Feb. 2012).

53. Marcuzzi, F. Analisi dei dati mediante modelli matematici. *Libreria Cortina* (2011).

54. Humpherys, J., Redd, P. & West, J. A Fresh Look at the Kalman Filter. *SIAM Review* **54,** 801–823 (2012).

55. Welch, G., Bishop, G., *et al.* An introduction to the Kalman filter (1995).

56. Willems, J. C. & Polderman, J. W. *Introduction to mathematical systems theory: a behavioral approach* (Springer Science & Business Media, 1997).

57. Slodička, M & Johansson, B. T. Uniqueness and counterexamples in some inverse source problems. *Applied Mathematics Letters* **58,** 56–61 (2016).

58. Shi, C., Wang, C. & Wei, T. Numerical solution for an inverse heat source problem by an iterative method. *Applied Mathematics and Computation* **244,** 577–597 (2014).

59. Lin, G., Zhang, Z. & Zhang, Z. Theoretical and numerical studies of inverse source problem for the linear parabolic equation with sparse boundary measurements. *Inverse Problems* **38,** 125007 (2022).

60. El Badia, A. & Ha-Duong, T. On an inverse source problem for the heat equation. Application to a pollution detection problem. *Journal of inverse and ill-posed problems* **10,** 585–599 (2002).

61. Denisov, A. M. Uniqueness and nonuniqueness of the solution to the problem of determining the source in the heat equation. *Computational mathematics and mathematical physics* **56,** 1737–1742 (2016).

62. Grewal, M. S. & Andrews, A. P. *Kalman Filtering: Theory and Practice with MATLAB®: Fourth Edition* 1–617 (John Wiley & Sons, Inc., 2014).