POLYTECHNIC OF TURIN

Master's Degree in Engineering Mathematics

Politecnico
di Torino

Master's Degree Thesis

Estimating the thermal parameters of a brake pad from experimental data

Supervisors

Prof. Enrico BIBBONA

Ing. Pietro MACCHI

Candidate

Edoardo ABATE

July 2024

Abstract

The aim of this thesis is to develop and test a reliable one-dimensional physical model of heat diffusion within the brake system of a car. The ITT company has been able to develop a brake pad, termed Smart Pad, that is endowed with sensors installed on the backplate, in particular a temperature sensor. The long-term goal of the research project developed with the ITT company is to use the data from this sensor to prevent the brake from overheating by estimating the disk temperature in real time. Since the backplate and the disk are the two opposite sides of the friction material it is necessary to model the heat propagation inside the pad. To this aim we use a one-dimensional model of heat conduction neglecting dissipation in other directions.
To validate the 1d-model, ITT conducts an experiment in controlled conditions collecting data that we use to estimate the thermal parameters of the pad. This approach is initially employed on synthetic data generated using the model, imposing initial and boundary conditions. Subsequently, experimental data are used. This study enabled the determination of the minimum conditions for identifiability and the investigation of datasets and acquisition methodologies (other than those employed) that are optimal for parameter estimation.

Acknowledgements

Al termine del mio percorso universitario, ci tengo a ringraziare tutte le persone che mi sono state vicine da sempre e le persone che ho avuto modo di conoscere e con cui ho avuto modo di legare in questi, faticosi quanto belli, anni al Politecnico.

Un grazie particolare al Prof. Enrico Bibbona e all'Ing. Pietro Macchi, che mi hanno dato la possibilità di svolgere la tesi sotto la loro illuminante guida. Grazie per tutti i consigli e le dritte che mi avete fornito, ne farò sicuramente tesoro. Colgo inoltre l'occasione per ringraziare ITT per avermi concesso l'opportunità di lavorare al loro progetto.

Vorrei inoltre ringraziare la mia famiglia. Un grazie speciale a nonno Giuseppe, che da sempre desiderava un ingegnere in famiglia; posso finalmente dire di esserlo, mi dispiace solo non aver fatto in tempo. Ringrazio inoltre mia zia Anna, nonna Rosa e nonna Carla per il supporto e l'interesse che mi avete dato. Ringrazio i miei genitori, che, per quanto possano ripetere di essere felici di questo traguardo e lamentino il fatto di non essersene nemmeno resi conto, di aver creduto in me: non sarebbe stato possibile senza il vostro inesauribile sostegno. Ringrazio anche mio fratello e mia sorella per il supporto, per i preziosi consigli e per la loro essenziale presenza.

Ci tengo a ringraziare anche i miei amici e i miei compagni di corso. Un grazie dunque a Fabio, Andrea, Ilaria e Michele per aver reso le ore di lezione piacevoli e meno faticose con la vostra presenza. E un grazie speciale a Giulia e Domenico per l'appoggio, i bei momenti e le ore di studio passate in vostra compagnia che hanno alleggerito il peso delle sessioni.

Last but not least, un grazie di cuore a Martina: la tua presenza è stata fondamentale, il tuo appoggio immancabile, la tua pazienza incrollabile, la tua intelligenza e determinazione motivo di ammirazione.

Table of Contents

List of Tables

List of Figures

Acronyms

RBC
Right Boundary condition

IC
Initial Condition

FMT
Friction Material Thickness

1s
1 sensor

2s
2 sensors

ODE
Ordinary Differential Equation

PDE
Partial Differential Equation

EE
Euler Explicit

CN
Crank Nicolson

UL
Underlayer

Chapter 1

Introduction

1.1 Motivation

The temperatures reached during braking inside the brake pad of a car can be extremely high. Overheating of the disc is a very dangerous phenomenon when driving a vehicle: the disc temperature can reach such critical values that the pad slips on the disc, compromising the efficiency of braking.
The causes of this phenomenon can be many:

- Long braking at high speed
- Brake caliper jamming
- A bad braking system
- Incorrect installation of brake pads
- Brake disc deformation.

It can be argued that knowledge of the temperature of the disc and of the pad while driving would be a vital piece of information, with potential applications in road safety and motor racing contests. For instance, a truck driver who was aware of the temperature of their braking system might choose to take a break and allow the system to cool down, rather than risk using the emergency ramp for deceleration or even worse causing an incident. To achieve a predictive analysis of behavior, it is necessary to develop models. These models can vary in complexity. We have developed a very simple one, focusing on the identifiability of thermal parameters, considering it a crucial strategy for model validation.
The main problem in trying to predict the temperature of the brake is that we have records of temperatures only at the backplate, where the sensor is placed, before we start to see a change in temperature due to braking, some time must pass for

the heat to cross for the heat to cross the friction material. This delay prevents real-time determination of the brake temperature but allows for it with a short lag.

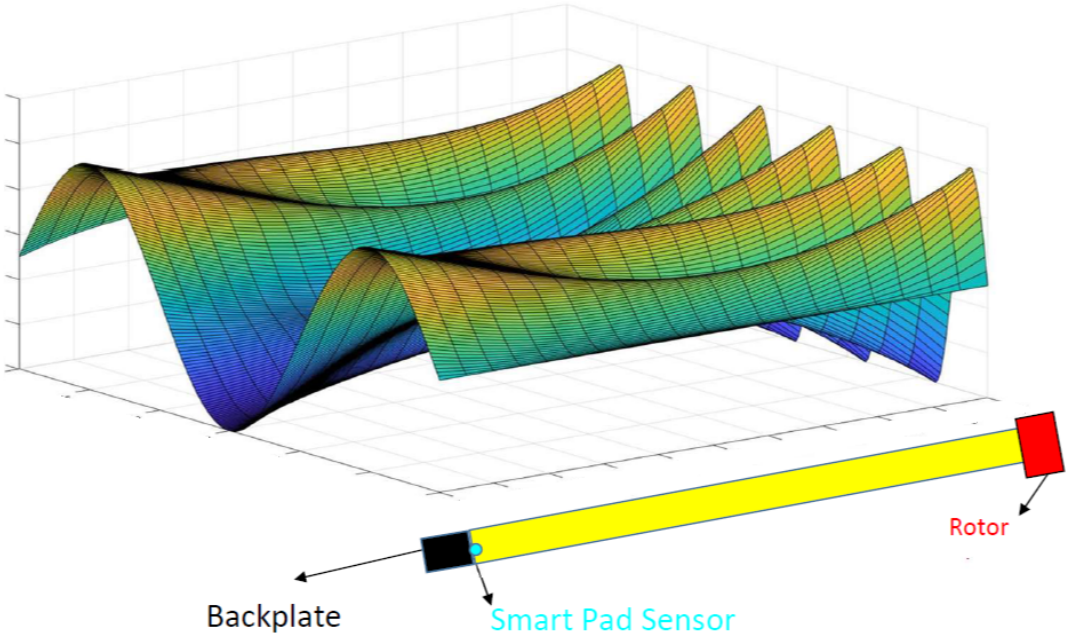


Figure 1.1: Example of temperature profile

However, while this is a problem that can be solved by knowing the energy spent during the breaking, which is predictable, a more complex challenge is modeling the cooling phenomenon of the brake during periods when we are not braking. Indeed, the moving vehicle dissipates heat through mechanisms such as ventilation, convection, and conduction where valid models are much more complicated. In the model, this translates to applying the correct boundary conditions for braking and moving. Thus, we distinguish between two problems: one simple, because we know how to impose the boundary conditions, and the other complex, because we do not know how to model the cooling factors of the brake.

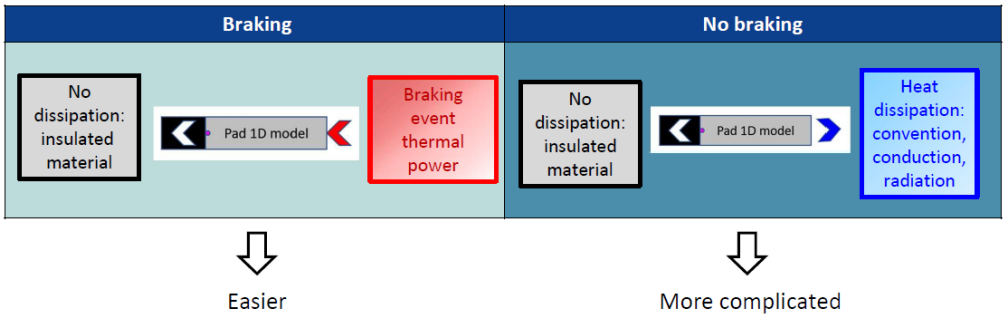


Figure 1.2: Representation of the problems

In this thesis, we will focus on validating the one-dimensional model by considering the simplest of the two problems. This is because if the model can provide accurate predictions of the brake temperature during braking, then it will also be a valid model for cooling when the right boundary conditions are imposed. As mentioned earlier, this involves sufficiently modeling the rotor-side heat dissipation and providing the thermal diffusion equation solver with the correct boundary conditions.

1.2 Basic Concepts and Definitions

A differential equation that contains, in addition to the dependent variable and the independent variables, one or more partial derivatives of the dependent variable is referred to as a partial differential equation. In general, it may be written in the following form:

$$f(x, y, ..., u, u_x, u_y, ..., u_{xx}, u_{xy}, ...) = 0) \tag{1.1}$$

The concept of a function of several independent variables, denoted by $x, y, ...$ is introduced, along with the partial derivatives of this function, $u_x, u_y, ..., u_{xx}, u_{xy}, ...$ which represent the rate of change of the function with respect to each variable. Subscripts on dependent variables denote differentiations, e.g.,

$$u_x = \frac{\partial u}{\partial x} \qquad\qquad u_{xy} = \frac{\partial^2 u}{\partial y \partial x}$$

The order of a partial differential equation is defined as the order of the highest-ordered partial derivative appearing in the equation. For example

$$u_{xx} + 2xu_{xy} + u_{yy} = e^y$$

is a second-order partial differential equation. and

$$u_{xxy} + xu_{yy} + 8u = 7y$$

is a third-order partial differential equation. A third-order partial differential equation. A partial differential equation is said to be linear if it is linear in the unknown function and all its derivatives, with coefficients depending only on the independent variables. It is said to be quasi-linear if it is linear in the highest-ordered derivative of the unknown function. For example, the equation

$$yuxx + 2xyuyy + u = 1$$

The equation which is not linear is called a nonlinear equation.

1.3 Model and equations

1.3.1 1D Heat equation

The heat equation 1D is a linear partial differential equation which governs the temperature distribution in an object. Let us consider a thin metal rod located at the interval $[0, L]$ on the x-axis. By identifying the bar with its axis, we can describe all the physical quantities of interest as a function of the x-axis and time. In particular, we shall see that the temperature $u = u(x, t)$ obeys the equation

$$c\rho\frac{\partial u}{\partial t} - \frac{\partial}{\partial x}\left(\kappa\frac{\partial u}{\partial x}\right) = \rho q \tag{1.2}$$

In the stationary case, where all variables are not time-dependent, the equation reduces to

$$-\frac{d}{dx}\left(\kappa\frac{du}{dx}\right) = \rho q \tag{1.3}$$

where:

- ρ is the mass density per unit length
- c is the specific heat of the bar
- κ is the thermal conductivity coefficient of the bar
- q is the heat input per unit mass and length

It is advisable to make a few comments about these quantities before proceeding. The specific heat, $c(x) > 0$, of a material is the amount of heat energy that it takes to raise one unit of mass of the material by one unit of temperature. As indicated, we are going to assume, at least initially, that the specific heat may not be uniform throughout the bar. It should also be noted that in practice the specific heat depends upon the temperature.

The mass density, $\rho(x)$, is the mass per unit volume of the material. As with the specific heat, we will initially assume that the mass density may not be uniform throughout the bar.
The thermal conductivity of a material, denoted by $\kappa(x)$, is a measure of its ability to conduct heat. The greater the ability of a material to conduct heat, the larger the value of $\kappa(x)$ will be. The thermal conductivity of a material can vary depending on its location within the bar. Additionally, as with the specific heat, the thermal conductivity may vary with temperature. However we will assume that the total temperature change is not significant, and thus that the thermal conductivity will not vary with temperature.
The final quantity defined above is $q(x,t)$, which represents any external sources or sinks of heat energy (i.e. heat energy taken out of the system). If $q(x,t) > 0$, heat energy is being added to the system at that location and time, while if $q(x,t) < 0$, heat energy is being removed from the system at that location and time.
Having successfully addressed the one-dimensional heat equation, we must now turn our attention to the initial and boundary conditions. As it is known the number of condition required in order to solve the problem always match the highest order of derivative in the equation. So for the heat equation we've got a first order time derivative and so we'll need one initial condition and a second order spatial derivative and so we'll need two boundary conditions. The initial condition that we'll use here is,

$$u(x,0) = f(x)$$

this equation tell us what the inital temperature distribution in the bar is.
The boundary conditions will inform us as to the temperature and/or heat flow occurring at the boundaries of the bar. There are four common boundary conditions that may be employed [1]. The first type of boundary conditions that can be considered are the prescribed temperature boundary conditions, also known as Dirichlet conditions. These are defined as follows:

$$u(0,t) = g_1(t) \qquad\qquad u(L,t) = g_2(t) \tag{1.4}$$

The subsequent type of boundary conditions is prescribed heat flux, also referred to as Neumann conditions. These can be expressed in accordance with Fourier's law as follows:

$$k\frac{du}{dx}(0) = \psi_0 \qquad\qquad k\frac{du}{dx}(L) = \psi_L \tag{1.5}$$

The third type of boundary conditions employs Newton's law of cooling and are sometimes referred to as Robins conditions. These are typically employed when the bar is situated within a moving fluid, with air being considered a fluid for the purposes of this discussion. The following equations represent the equations for

this type of boundary condition.

$$k\frac{du}{dx}(0,t) = -H[u(0,t) - g_1(t)] \qquad k\frac{du}{dx}(L,t) = H[u(L,t) - g_1(t)] \tag{1.6}$$

The quantity H, which is a positive experimental value, is used to determine the temperature of the surrounding fluid at the respective boundaries, as indicated by the functions $g_1(t)$ and $g_2(t)$. It should be noted that the two conditions do vary slightly depending on which boundary is being considered. At $x = 0$, a minus sign is present on the right side, while this is not the case at $x = L$. The final type of boundary conditions are periodic boundary conditions. These are defined as follows:

$$u(0,t) = u(L,t) \qquad \frac{du}{dx}(0,t) = \frac{du}{dx}(L,t) \tag{1.7}$$

1.4 Mathematical Problem

A first problem consist in finding solution of a particular differential heat equation satisfying appropriate supplementary conditions. Specifically, the model we will analyze is a one-dimensional thermal diffusion model without boundary dispersion, involving two materials and assuming mixed boundary conditions. In particular, for $x = 0$, we will have a no-flux condition (Neumann condition), and for $x = L$, we will have a constant temperature condition.

$$\frac{\partial u(0)}{\partial x} = \psi_0 \qquad u(L,t) = g_2(t)$$

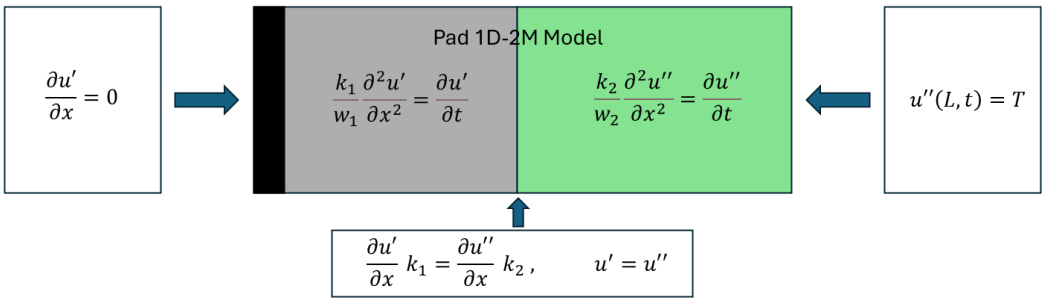


Figure 1.3: diffusion model

where:

- κ_1 is the thermal conductivity coefficient of the Underlayer (UL)
- w_1 is the product of UL mass density for the UL specif heat
- κ_2 is the thermal conductivity coefficient of the Friction material (FM)
- w_1 is product of FM mass density for the FM specif heat

with two additional condition in between the two material which state the flux and the temperature in the middle have to be the same in the two material.

The second problem involves determining the thermal parameters of the brake pad based on experimental data, using the differential equation solver described earlier. The approach to estimate these parameters relies on minimizing an objective function defined as the sum of squared distances between the experimental data recorded by sensors located as shown in Figure 3.1, and the approximation of the dynamics at grid points corresponding to the actual sensor locations. Therefore, the objective function is a function of the thermal parameters passed to the differential equation solver, also we can define the minimization problem as:

$$\min_{\boldsymbol{\theta}} \sum_{i=1}^{n} \sum_{j=1}^{m} \left(y_{ij} - f(x_{ij}; \boldsymbol{\theta})\right)^2 \tag{1.8}$$

where $\boldsymbol{\theta} = [k_1, k_2, w_1, w_2]$ represents the thermal parameters of the pad, y_{ij} denotes the actual temperature readings at the i-th point of the pad at time j, and $f(x_{ij}; \boldsymbol{\theta})$ denotes the approximations at the ij-th grid point of the space-time grid by the solver using the parameters $\boldsymbol{\theta}$. For convenience, we refer to this function as the sum squared error (SSE). In our specific case, each dataset includes recordings from 5 different sensors, of which at most 2 sensors will be utilized per dataset, hence $n \leq 2$. The value of m, on the other hand, depends on the sampling frequency and duration of the test under consideration.

Since it is defined as the sum of squared deviations between real and approximated temperature profiles, we have limited insight into the properties or minima of this function with respect to thermal parameters. To solve the optimization problem 1.8, an iterative optimization algorithm based on a heuristic approach called Nelder-Mead will be employed.

However, it is evident that this process is computationally expensive because each optimization iteration requires solving one or more differential equations (depending on the number of datasets involved). Therefore, using a pre-existing solver is impractical due to time constraints, necessitating the development of a custom solver to expedite the optimization process.

Chapter 2

Background

2.1 Numerical method

We will begin our exposition on numerical methods for solving differential equations by first addressing ordinary differential equations (ODEs). This is crucial for our purpose as these methods play an essential role in the analysis and construction of methods for partial differential equations (PDEs). In fact, the discretization methods for the latter reduce the problem to solving ODEs.

Our current aim is thus the approximation of solutions to problems of the type:

$$y' = f(t, y), \quad t \geq t_0, \quad y(t_0) = y_0 \tag{2.1}$$

Where f is a sufficiently smooth function, in this case we will require it to be Lipschitz, mapping $[t_0, \infty) \times \mathbb{R}^d$ given an initial condition $y_0 \in \mathbb{R}^d$, which is a state in the state space $\mathbb{R}^d$. Moreover, being a Lipschitz function means satisfying the condition

$$\|f(t, x) - f(t, y)\| \leq \lambda \|x - y\| \quad \forall x, y \in \mathbb{R}^d, \quad t \geq t_0 \tag{2.2}$$

2.1.1 Euler's method

Reflecting on the meaning of ODEs, we have two pieces of information available: the value of y at time $t = t_0$, and given any value of the function $y \in \mathbb{R}^d$ at time $t \geq t_0$, we can determine the slope from the differential equation. With this information, our goal is to predict the value of y at a new point. The most immediate approach to this problem is to use linear interpolation. That is, we want to estimate $y(t)$ by assuming that by moving to a sufficiently small time interval, the slope remains approximately the same. Translated into mathematical terms, the approximation

we are making is the following:

$$f(t, y(t)) \approx f(t_0, y(t_0)) \quad \text{for} \quad t \in [t_0, t_0 + h]$$

where $h > 0$ is sufficiently small. Integrating 2.1,

$$y(t) = y(t_0) + \int_{t_0}^{t} f(\tau, y(\tau))d\tau \approx y_0 + (t - t_0)f(t, y_0) \tag{2.3}$$

Given a sequence $t_0, t_1 = t_0 + h, t_2 = t_0 + 2h, \ldots$, where $h > 0$ is the time step, we denote by y_n the numerical estimate of the exact solution $y(t_n)$. Motivated by the formula 2.3, we will say that

$$y_1 = y_0 + hf(t_0, y_0)$$

we can iterate this process to obtain the general recursive formula.

$$y_{n+1} = y_n + hf(t_n, y_n), \quad n = 0,1, ..., \tag{2.4}$$

The entire procedure just described is known by the famous name of Euler's method. As simple as this method is, it forms the basis, in a certain sense, for all multistep methods and Runge-Kutta methods, which are nothing more than generalizations of Euler's method [2]. Before assessing how well Euler's method approximates the real solution, we need to examine the convergence of the method.

Theorem 1 *Euler's method is convergent*

Proof: We will prove this theorem with an additional hypothesis about the function f; specifically, we will assume that this function is also analytic. Given $h > 0$ and $y_n = y_{n,h}, n = 0, 1, \ldots, \left\lfloor \frac{t^*}{h} \right\rfloor$, we define $e_{n,h} = y_{n,h} - y(t_n)$ to denote the numerical error. We wish to prove that $\lim_{h\to 0} \max_n \|e_{n,h}\| = 0$. Expanding the differential equation using Taylor series, 2.1

$$y(t_{n+1}) = y(t_n) + hy'(t_n) + \mathcal{O}(h^2) = y(t_n) + hf(t_n, y(t_n)) + \mathcal{O}(h^2) \tag{2.5}$$

because of y is continously differentiable, the term $\mathcal{O}(h^2)$ can be bounded by a generic term of the form ch^2.Subtructing from 2.4 the 2.5, we obtain

$$e_{n+1,h} = e_{n,h} + h[f(t_n, y(t_n) + e_{n,h}) - f(t_n, y(t_n))] + \mathcal{O}(h^2) \tag{2.6}$$

thus it follow by the triangle inequality from Lipschitz condition that

$$\|e_{n+1,h}\| \leq \|e_{n,h}\| + h\|f(t_n, y(t_n) + e_{n,h}) - f(t_n, y(t_n))\| + ch^2 \tag{2.7}$$

$$\leq (1 + h\lambda)\|e_{n,h}\| + ch^2, \quad n = 0,1, ..., \lfloor t^*/h \rfloor \tag{2.8}$$

We now claim that

$$\|e_{n,h}\| \leq \frac{c}{\lambda} h[(1+h\lambda)^n - 1], \quad n = 0,1,... \tag{2.9}$$

the proof of this statement is by induction on n. When $n = 0$ we need to prove that $\|e_{0,h}\| \leq 0$ and hence that $e_{0,h} = 0$ this is true because t_0 the numerical solution matches the initial condition and the error is zero. For $n \geq 0$ we assume that is true up n and try to demonstrate for $n + 1$

$$\|e_{n+1,h}\| \leq (1+h\lambda)\frac{c}{\lambda} h[(1+h\lambda)^n - 1] + ch^2 = \frac{c}{\lambda} h[(1+h\lambda)^{n+1} - 1] \tag{2.10}$$

this advances the inductive argument and state that what we have claimed is true. the constant $h\lambda$ is positive, therefore $1 + h\lambda < e^{h\lambda}$ hence $(1+h\lambda)^n < e^{nh\lambda}$. The index n is allowed to range in 0,1, ..., $\lfloor t^*/h \rfloor$, so $(1+h\lambda)^n < e^{\lfloor t^*/h \rfloor h\lambda} \leq e^{t^*\lambda}$. Substituting into 2.9 we obtain the inequality

$$\|e_{n,h} \leq \frac{c}{\lambda}(e^{t^*\lambda} - 1)h, \quad n = 0,1, ...\lfloor t^*/h \rfloor. \tag{2.11}$$

Since $c(e^{t^*\lambda} - 1)/\lambda$ is independent of h, it follows that

$$\lim_{\substack{h \to 0 \\ 0 \leq nh \leq t^*}} \|e_{n,h}\| = 0 \tag{2.12}$$

In other words, Euler's method is convergent. □
Moreover, Euler's method can be rewritten in the form $y_{n+1} - [y_n + hf(t_n, y(t_n))] = 0$. Replacing y_k by the exact solution $y(t_k), k = n, n+1$, and expanding the first few terms of the Taylor series about $t = t_0 + nh$, we obtain

$$\begin{aligned} y(t_{n+1}) - [y(t_n) + hf(t_n, y(t_n))] \\ = [y(t_n) + hy'(t_n) + \prime(h^2)] - [y(t_n) + hy'(t_n)] = \prime(h^2). \end{aligned}$$

For this reason we say that Euler's method is of order 1, the order of a numerical method provides us information about its local behaviour advancing in time from t_n to t_{n+1}, where $h > 0$ is sufficiently small we incur in an error of $\mathcal{O}(h^2)$

2.1.2 Trapezoidal rule

Euler's method approximates derivatives with a constant over the interval $[t_n, t_{n+1}]$ using the constant at t_n. This approximation is not very accurate. An alternative is to approximate the derivative over the interval with the average of the derivatives at the endpoints. This leads us to an expression similar to 2.3:

$$y(t) = y(t_n) + \int_{t_n}^{t} f(\tau, y(\tau))d\tau \approx y(t_n) + \frac{1}{2}(t - t_n)[f(t_n, y(t_n)) + f(t, y(t))] \tag{2.13}$$

This is the motivation behind the trapezium rule

$$y_{n+1} = y_n + \frac{1}{2}h[f(t_n, y_n) + f(t_{n+1}, y_{n+1})]. \tag{2.14}$$

also to obtain the order of 2.14 we substitute the exact solution,

$$\begin{aligned} y(t_{n+1}) - y(t_n) + \frac{1}{2}h[f(t_n, y(t_n)) + f(t_{n+1}, y(t_{n+1}))] \\ = [y(t_n) + hy'(t_n) + \frac{1}{2}h^2y''(t_n) + \mathcal{O}(h^3)] \\ - (y(t_n) + \frac{1}{2}hy'(t_n) + [y(t_n) + hy''(t_n) + \mathcal{O}(h^2)]) = \mathcal{O}(h^3) \end{aligned}$$

Hence the trapezoid methods have order 2, but before we infer that the error decays globally as $\mathcal{O}(h^2)$, we will first prove that the method is convergent, this proof can be done generalizing the results obtained for Euler's method

Theorem 2 *The trapezoidal rule 2.14 is convergent.*

Proof: Subtracting

$$y(t_{n+1}) = y(t_n) + \frac{1}{2}h[f(t_n, y(t_n)) + f(t_{n+1}, y(t_{n+1}))] + \mathcal{O}(h^3)$$

from 2.14, we obtain

$$\begin{aligned} e_{n+1,h} = e_{n,h} + \frac{1}{2}h[f(t_n, y_n) - f(t_n, y(t_n))] \\ + f(t_{n+1}, y_{n+1}) - f(t_{n+1}, y(t_{n+1})) + \mathcal{O}(h^3) \end{aligned}$$

for any analytic f we may bound the $\mathcal{O}(h^3)$ term by ch^3 for some $c > 0$, and this upper bound is valid uniformly throughout $[t_0, t_0 + t^*]$. Therefore, it follows from the Lipschitz condition and triangle inequality that

$$\|e_{n+1,h}\| \leq \|e_{n,h}\| + \frac{1}{2}h\lambda\|e_{n,h} + e_{n+1,h}\| + ch^3.$$

Since we are ultimately interested in letting $h \to 0$ we can assume that $h\lambda < 2$, and we thus deduce that

$$\|e_{n+1,h}\| \leq \left(\frac{1 + \frac{1}{2}h\lambda}{1 - \frac{1}{2}h\lambda}\right)\|e_{n,h}\| + \left(\frac{c}{1 - \frac{1}{2}h\lambda}\right)h^3$$

Our next step closely parallels the derivation of inequality 2.9. We thus argue that

$$\|e_{n,h}\| \leq \frac{c}{\lambda}\left[\left(\frac{1 + \frac{1}{2}h\lambda}{1 - \frac{1}{2}h\lambda}\right)^n - 1\right]h^2 \tag{2.15}$$

this is follows as before on n as before, and after some calculations this yields to

$$\|e_{n,h}\| \leq \frac{ch^2}{\lambda}\left(\frac{1+\frac{1}{2}h\lambda}{1-\frac{1}{2}h\lambda}\right)^n \leq \frac{ch^2}{\lambda}\exp\left(\frac{nh\lambda}{1-\frac{1}{2}h\lambda}\right).$$

This bound is true for every n such that $nh \leq t^*$. Therefore

$$\|e_{n,h}\| \leq \frac{ch^2}{\lambda}\exp\left(\frac{t^*\lambda}{1-\frac{1}{2}h\lambda}\right)$$

hence

$$\lim_{\substack{h\to 0 \\ 0\leq nh\leq t^*}} \|e_{n,h}\| = 0 \tag{2.16}$$

In other words, the trapezoidal rule converges. □

The error of trapezoidal rule decays globally as $\mathcal{O}(h^2)$. This is to be expected from a second-order method if its convergence has been establish.

2.1.3 Euler scheme and Crank-Nicolson

We will now discuss the first numerical method for solving partial differential equations (PDEs) used to address our two-material heat diffusion problem. Before delving into the Euler scheme, it is useful to classify PDEs into two categories: steady-state equations, where all variables are spatial, and evolutionary equations, which involve differentiation with respect to both space and time. PDEs are also classified as elliptic, parabolic, or hyperbolic. Elliptic equations are of the steady-state type, while both parabolic and hyperbolic PDEs are evolutionary. The distinction among these types lies in the different kinds of characteristic curves they admit.

Ordinary differential equations (ODEs) can be viewed as evolutionary equations without spatial variables. This similarity allows us to solve evolutionary PDEs by approximating them as ODEs. The Euler scheme is essentially a transposition of the Euler method for ODEs to PDEs. The numerical solution of evolutionary PDEs requires discretization in both time and space, and these two procedures cannot be done independently. Considering the simplest version of the diffusion equation:

$$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2} \tag{2.17}$$

the function $u = u(x,t)$ is accompained by two kinds of side conditions, an initial condition and a boundary condition. There are a lot of ways to state this conditions 1.4 1.5 1.6 1.7.

The first step is to create a grid to apply finite difference, chosen a positive integer d and inscribe into the strip

$$\{(x,t): \quad x \in [0,1], t > 0\}$$

a rectangular grid

$$\{(\ell\Delta x, l\Delta t), \quad \ell = 0,1,...,d+1, n \geq 0\}$$

where $\Delta x = 1/(d+1)$. The approximation of $u(\ell\Delta x, l\Delta t)$ is denoted by u_ℓ^n. Replacing the second spatial derivative and the first temporal derivative respectively by the central difference

$$\frac{\partial^2 u(x,t)}{\partial x^2} \approx \frac{1}{(\Delta x)^2}[u(x-\Delta x,t) - 2u(x,t) + u(x+\Delta x,t)] + \mathcal{O}((\Delta x)^2), \quad \Delta x \to 0,$$

and the forward difference

$$\frac{\partial u(x,t)}{\partial t} \approx \frac{1}{\Delta t}[u(x,t+\Delta t) - u(x,t)] + \mathcal{O}(\Delta t), \quad \Delta t \to 0,$$

Replacing into 2.17 and multiply by Δt results in the Euler method

$$u_\ell^{n+1} = u_\ell^n + \mu(u_{\ell-1}^n - 2u_\ell^n + u_{\ell+1}^n), \quad \ell = 1,2,...,d, \quad n = 0,1,..., \tag{2.18}$$

where μ is the ratio

$$\mu = \frac{\Delta t}{(\Delta x)^2}$$

To start the recursive procedure we begin calculation from the initial condition of the problem imposing

$$u_\ell^0 = g(\ell\Delta x), \quad l = 1,2,...,d.$$

Important note the initial condition enters in the method when we need to computer first and last step, indeed $u_0^n = \varphi_0(n\Delta t)$ and $u_{d+1}^n = \varphi_1(n\Delta t)$.
As always the concept of the order is important in studying how well a finite difference scheme models a continuous differential equation but the major concern is convergence, not order. A method is convergent if, given any $t^* \geq 0$, it is true that

$$\lim_{\Delta x \to 0}\left[\lim_{\ell \to x/\Delta x}\left(\lim_{n \to t/\Delta t} u_\ell^n\right)\right] = u(x,t) \quad \forall x \in [0,1], \quad t \in [0,t^*].$$

Theorem 3 *If $\mu \leq \frac{1}{2}$ then the method is convergent*

Proof: Let $t > 0$ be an arbitrary constant and define

$$e_\ell^n := u_\ell^n - u(\ell\Delta x, n\Delta t), \quad \ell = 0,1,...,d+1 \quad n = 0,1,...,n\Delta t,$$

where $n_{\Delta n} = \lfloor t^*/\Delta t \rfloor = \lfloor t^*/(\mu(\Delta x)^2) \rfloor$ is the right-hand endpoint of the range of n. The definition of convergence can be expressed in the terminology of the variables e_ℓ^n as

$$\lim_{\Delta x \to 0} \left[\max_{\ell=0,1,...,d+1} \left(\max_{n=0,1,...,n_{\Delta t}} |e_\ell^n| \right) \right] = 0$$

Letting

$$\eta_n := \max_{\ell=0,1,...,d+1} |e_\ell^n|, \quad n = 0,1,...,n_{\Delta t},$$

rewriting this as

$$\lim_{\Delta x \to 0} \left(\max_{n=0,1,...,n_{\Delta t}} |\eta_n| \right) = 0 \tag{2.19}$$

Since

$$\begin{aligned} u_\ell^{n+1} &= u_\ell^n + \mu(u_{\ell-1}^n - 2u_\ell^n + u_{\ell+1}^n), \\ \tilde{u}_\ell^{n+1} &= \tilde{u}_\ell^n + \mu(\tilde{u}_{\ell-1}^n - 2\tilde{u}_\ell^n + \tilde{u}_{\ell+1}^n) + \mathcal{O}((\Delta x)^4), \\ &\qquad \ell = 0,1,...,d+1, \quad n = 0,1,...,n_{\Delta t} - 1 \end{aligned}$$

Where $\tilde{u}_\ell^n = u(\ell\Delta x, n\Delta t)$ subtraction results in

$$\begin{aligned} e_\ell^{n+1} &= e_\ell^n + \mu(e_{\ell-1}^n - 2e_\ell^n + e_{\ell+1}^n) + \mathcal{O}((\Delta x)^4), , \\ &\qquad \ell = 0,1,...,d+1, \quad n = 0,1,...,n_{\Delta t} - 1 \end{aligned}$$

provided u sufficiently smooth, there exists a constant $c > 0$, independent of Δx, such that, for every $\ell = 0,1,...,d+1$,

$$\begin{aligned} |e_\ell^{n+1} - e_\ell^n - \mu(e_{\ell-1}^n - 2e_\ell^n + e_{\ell+1}^n)| &\le c((\Delta x)^4), \\ &\qquad \ell = 0,1,...,d+1, \quad n = 0,1,...,n_{\Delta t} - 1 \end{aligned}$$

Therefore, by the triangle inequality and the definition of η^n,

$$\begin{aligned} |e_\ell^{n+1}| &\le |e_\ell^n + \mu(e_{\ell-1}^n - 2e_\ell^n + e_{\ell+1}^n)| + c((\Delta x)^4), \\ &\le \mu|e_{\ell-1}| + |1 - 2\mu||e_\ell^n| + \mu|e_{\ell+1}^n| + c((\Delta x)^4), \\ &\le (2\mu + |1 - 2\mu|)\eta^n + c((\Delta x)^4), \quad n = 0,1,...,n_{\Delta t} - 1 \end{aligned}$$

Because of $\mu \le \frac{1}{2}$ is deduct that

$$\eta^{n+1} = \max_{\ell=0,1,...,d+1} |e_\ell^{n+1}| \le \eta^n + c(\Delta x)^4, \quad n = 0,1,...,n_{\Delta t} - 1$$

by induction

$$\eta^{n+1} \leq \eta^n + c(\Delta x)^4 \leq \eta^{n-1} + 2c(\Delta x)^4 \leq \eta^{n-2} + 3c(\Delta x)^4 \leq ...$$

is concluded that

$$\eta^n \leq \eta^0 + nc(\Delta x)^4, \quad n = 0,1, ..., n_{\Delta t}.$$

Since $\eta^0 = 0$ because for initial condition the discretization matches exactly the solution in that point of the grid and $n(\Delta x)^2 = n\Delta t/\mu \leq t^*/\mu$ is deducted that

$$\eta^n \leq \frac{ct^*}{\mu}(\Delta x)^2, \quad n = 0,1, ..., \eta_{\Delta t}.$$

Therefore $\lim_{\Delta x \to 0} \eta^n = 0$ for all n, and this conclude the proof because 2.19 $\square$

The Euler scheme is a method of the first order in time, we can do better in this term with a second order method in time by applying the trapezoidal rule 2.14 to the discratization in time resulting in to Crank-Nicolson method.

$$y_{n+1} = y_n + \frac{1}{2}\Delta t[f(n\Delta t, y_n) + f((n+1)\Delta, y_{n+1})]$$

Another difference between explicit euler and crank nicolson is that while the former is asymptotically stable only if the time step fulfils the condition

$$\Delta t < \frac{2}{\max_p |\lambda_p|}.$$

Proof: the explicit Euler method applied to the equation $z' = \lambda z$ gives:

$$z^{k+1} = z^k + \Delta\lambda z^k = (1 + \Delta\lambda)z^k, \quad k \geq 0,$$

then for the recursion formula taking into account the initial condition, gives:

$$z^k = (1 + \Delta\lambda)^k z_0,$$

The condition $|z_k| \to 0$ for $k \to +\infty$ which is equivalent to

$$|1 + \Delta\lambda| < 1.$$

setting $\alpha = \Delta\lambda$ so the inequality $|1 + \alpha| < 1$ define a into the complex plain the closed circle of center -1 and radius 1, this region is called of asymptotically stable, so if every component of $z_k = (z_p^k)$ have to decay to 0, Δt has to be chosen such that the condition

$$\Delta\lambda_p \in int(R_{EE}) \quad \forall p = 1, ..., n$$

this is surely true if and only if

$$-2 < \Delta\lambda_p < 0, \quad \Delta < \frac{2}{|\lambda_p|}, \quad \Delta < \frac{2}{\max_p |\lambda_p|}$$

With an analogous argument it's possible demonstrate that Crank-Nicolson is asymptotically stable indeed

$$\begin{aligned} z^{k+1} &= z^k + \Delta\lambda \left(\frac{1}{2}z^k + \frac{1}{2}z^{k+1}\right), \quad k \geq 0, \\ \left(1 - \frac{\Delta t}{2}\lambda\right) z^{k+1} &= \left(1 - \frac{\Delta t}{2}\lambda\right) z^k, \quad k \geq 0, \\ z^k &= \left(\frac{1 + \frac{\Delta t}{2}\lambda}{1 - \frac{\Delta t}{2}\lambda}\right)^k z_0, \quad k \geq 0. \end{aligned}$$

setting $\alpha = \Delta\lambda$, the last equation tends to 0 as k approach to ∞ if and only if

$$|1 + \frac{\alpha}{2}| < |1 - \frac{\alpha}{2}|.$$

it's easy verify that if α is any complex number, the last inequality holds every time $\mathbf{R}e(\alpha) \geq 0$ and in this case

$$\Delta\lambda_p \in (R_{CN}) \quad \forall p = 1, ..., n \quad \textit{and any} \quad \Delta t > 0$$

□

2.2 Nelder-Mead

The Nelder-Mead method, also known as the simplex method, is a popular algorithm for multidimensional unconstrained optimization problems. This method is particularly useful when the objective function is not differentiable, discontinuous, noisy, or expensive to evaluate[3]. The Nelder-Mead method operates on a simplex, which is a geometric figure consisting of $n + 1$ vertices in n-dimensional space. For example, in two dimensions, the simplex is a triangle, and in three dimensions, it is a tetrahedron. The algorithm iteratively modifies the simplex to converge towards an optimal solution. Following are described the steps for Nelder Mead algorithm:

1. **Initialization**
    - Start with an initial simplex consisting of $n + 1$ vertices.
    - Evaluate the objective function at each vertex of the simplex.

2. **Iteration**

   (a) **Order**

   - Sort the vertices based on their objective function values.

   (b) **Reflection**

   - Compute the reflection point x_r. This point is calculated by reflecting the worst point through the centroid of the remaining n points.
   - If the reflection point has a better objective function value than the second-worst point but not better than the best, replace the worst point with the reflection point.

   (c) **Expansion**

   - If the reflection point is the best point found so far, compute the expansion point x_e.
   - If the expansion point is better than the reflection point, replace the worst point with the expansion point. Otherwise, replace the worst point with the reflection point.

   (d) **Contraction**

   - If the reflection point is worse than the second-worst point, perform a contraction. There are two types of contraction:
     - **Outside Contraction**: If the reflection point is better than the worst point, compute the outside contraction point x_{oc}. If x_{oc} is better than the reflection point, replace the worst point with x_{oc}.
     - **Inside Contraction**: If the reflection point is worse than the worst point, compute the inside contraction point x_{ic}. If x_{ic} is better than the worst point, replace the worst point with x_{ic}.

   (e) **Shrink**

   - If neither the reflection nor contraction improves the simplex, perform a shrink operation. Reduce the size of the simplex towards the best point found so far.

3. **Termination**

   - The algorithm terminates when the simplex size becomes sufficiently small, or the objective function values at the vertices converge within a specified tolerance.

2.2.1 Mathematical Formulation of Nelder-Mead Method

Let $f(x)$ be the objective function to minimize. The steps of the Nelder-Mead method can be mathematically described as follows:

- **Reflection:**
$$x_r = x_c + \alpha(x_c - x_h) \tag{2.20}$$
where x_c is the centroid of the simplex excluding the worst point x_h, and α is the reflection coefficient (typically $\alpha = 1$).

- **Expansion:**
$$x_e = x_c + \gamma(x_r - x_c) \tag{2.21}$$
where γ is the expansion coefficient (typically $\gamma = 2$).

- **Outside Contraction:**
$$x_{oc} = x_c + \beta(x_r - x_c) \tag{2.22}$$
where β is the contraction coefficient (typically $\beta = 0.5$).

- **Inside Contraction:**
$$x_{ic} = x_c - \beta(x_c - x_h) \tag{2.23}$$

- **Shrink:**
$$x_i = x_l + \delta(x_i - x_l) \quad \text{for all } i \neq l \tag{2.24}$$
where x_l is the best point, and δ is the shrink coefficient (typically $\delta = 0.5$).

Chapter 3

Results

3.1 Data collection

ITT coducts 24 experiment, recording temperature over time from various sensors embedded in the pad under multiple different conditions. For these tests, the same pad was consistently used, a crucial aspect when aiming to identify the thermal parameters of the pad. These parameters can vary based on the materials used and the production process. While the materials for the pad remain consistent, the production process can differ. Since both the material and the production process are industrial secrets we will not disclose any numerical value that can be reveal thus, even indirectly

Within the pad, a minimum of four to a maximum of five thermocouples were inserted, and their distribution was maintained throughout all twenty-four tests, acquiring temperature recordings at a frequency of 1Hz. The number of thermocouples depends on the length of the pad used for recording. The length of the pad in the various tests 10.7 mm at 100% of its total length or 6.42 mm at 60%. The following figure illustrates the distribution of the sensors. From the figure 3.1, it is evident that when the pad is reduced to 60% of its length, the sensor labeled as Aux3, which is the furthest from the backplate, is removed along with the pad. As previously mentioned, the data were collected under different conditions:

1. Pad thickness with values $(100\%, 60\%)$
2. Plate temperature with values $(100[C], 250[C])$
3. Insulated Back Plate (1,0)

All experiments were repeated three times. This is not coincidental but a standard practice, as having multiple recordings of the same physical phenomenon

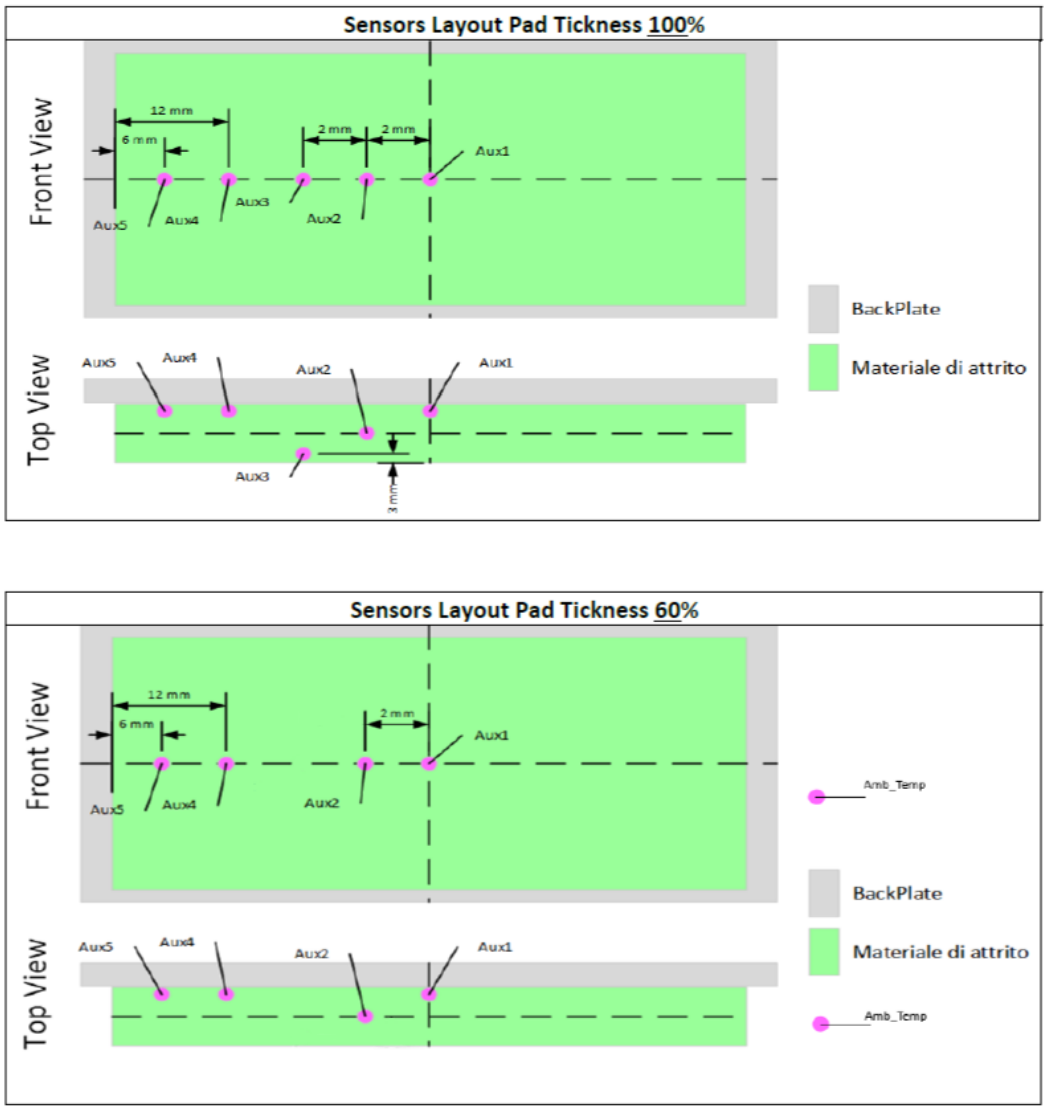


Figure 3.1: Thermocouples distribution

is crucial for reducing the impact of random errors inherent in any measurement process.

3.1.1 Data Analysis

In this subsection, a brief analysis of the data will be conducted. Not all datasets are equally suitable for optimization and parameter estimation. First, all data

Test_name	Pad tickness	PlateTemp	InsulatedBackPlate
	[%]	[°C]	[-]
TestPlate_F01	100	100	YES
TestPlate_F02	100	100	YES
TestPlate_F03	100	100	YES
TestPlate_F04	100	250	YES
TestPlate_F05	100	250	YES
TestPlate_F06	100	250	YES
TestPlate_F07	100	100	NO
TestPlate_F08	100	100	NO
TestPlate_F09	100	100	NO
TestPlate_F10	100	250	NO
TestPlate_F11	100	250	NO
TestPlate_F12	100	250	NO
TestPlate_F13	60	100	YES
TestPlate_F14	60	100	YES
TestPlate_F15	60	100	YES
TestPlate_F16	60	100	NO
TestPlate_F17	60	100	NO
TestPlate_F18	60	100	NO
TestPlate_F19	60	250	YES
TestPlate_F20	60	250	YES
TestPlate_F21	60	250	YES
TestPlate_F22	60	250	NO
TestPlate_F23	60	250	NO
TestPlate_F24	60	250	NO

Figure 3.2: Table of data

collected where the pad was not thermally insulated from the backplate were excluded. This exclusion was necessary because our data would not be consistent in the model built, as we assumed zero heat flux at the backplate.

Secondly, all acquisition graphs were plotted to identify any physical inconsistencies. Specifically, we focused on determining whether all acquisitions reached the nominal temperature of the hot plate. It was observed that datasets F_13, F_14, and F_15 did not reach the nominal temperature of 100$[C]$, and therefore, the recorded steady-state temperature was adjusted to 90$[C]$. This is plausible because all the other observations in the insulated setting reach their nominal temperatures. Therefore, we can infer that these three experiments fail to reach their target not due to dissipation but rather due to an incorrect imposition of the

hot plate temperature.

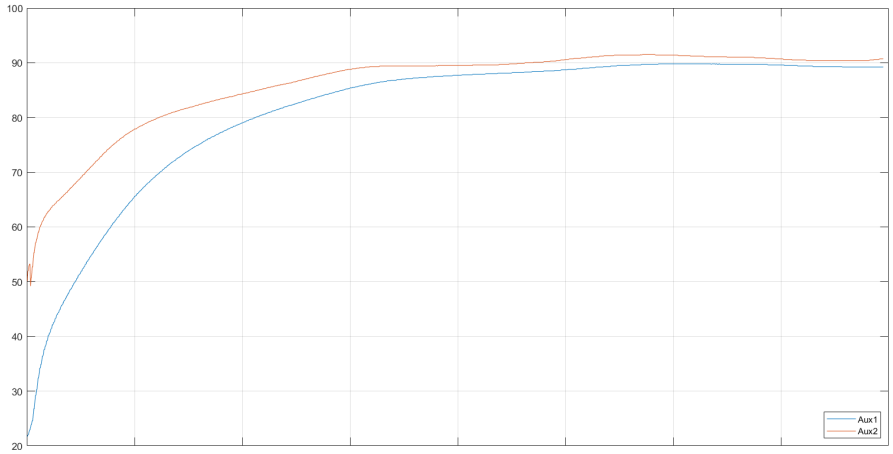


Figure 3.3: TestF_13

Following this, the time taken for acquisitions of the same type to reach steady-state was analyzed. This analysis served two purposes:

1. To invalidate tests that significantly differed from others of the same type.

2. To align the temperature knees, ensuring that the time t_0 coincided with the moment when the hot plate comes into contact with the pad.

Following this analysis, TestF_01 was discarded because, in addition to exhibiting obvious oscillations inconsistent with the type of experiment, it had a transient time much shorter than the other tests. Such a discrepancy is not compatible with a delay by the operator in starting sensor acquisition.
Conversely, Tests F_04 and F_19 were shifted forward by 16 seconds and 6 seconds, respectively, to let them overlap.

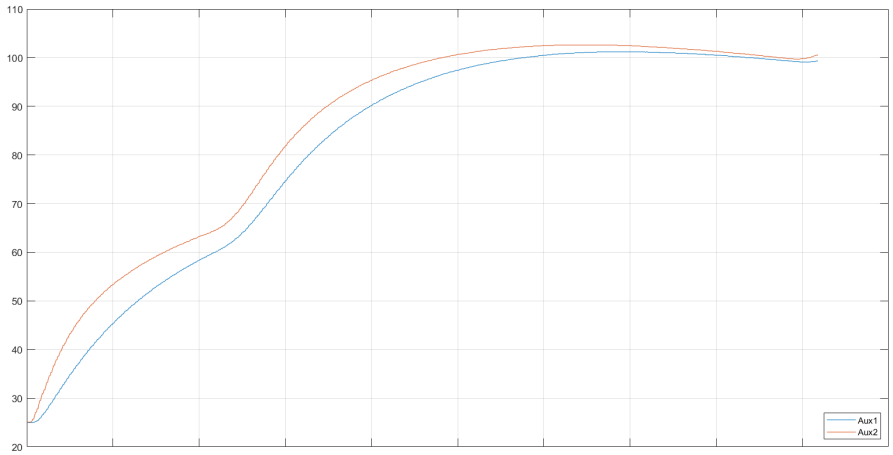


Figure 3.4: Odd oscillation TestF_01

3.1.2 Design of experiment

With the data that we ultimately deemed suitable for producing an estimate of the thermal parameters of the pad, a table of experiments was created with the aim of verifying that, starting from any subset of datasets under appropriate conditions of identifiability (which will be discussed in the next section), the same parameter estimates can be obtained.
The following table summarizes the suitable data with a brief description of the type of dataset.

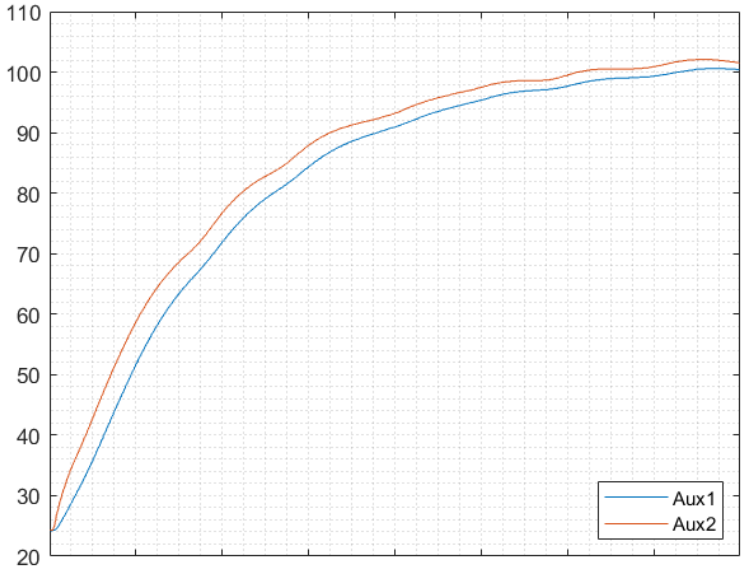
Test_name	Tickness[%]	PlateTemp[°C]	InitTemp[°C]	Delay[s]	Transitory[s]
Test_F02	100	100	25,10	0	T_1
Test_F03	100	100	24,20	0	T_1
Test_F04	100	250	24,20	16	T_2
Test_F05	100	250	21,30	0	T_2
Test_F06	100	250	20,10	0	T_2
Test_F13	60	90	21,70	0	T_3
Test_F14	60	90	22,80	0	T_3
Test_F15	60	90	22,80	0	T_3
Test_F19	60	250	21,00	6	T_4
Test_F20	60	250	23,80	0	T_4
Test_F21	60	250	24,30	0	T_4

Table 3.1: Description of suitable data

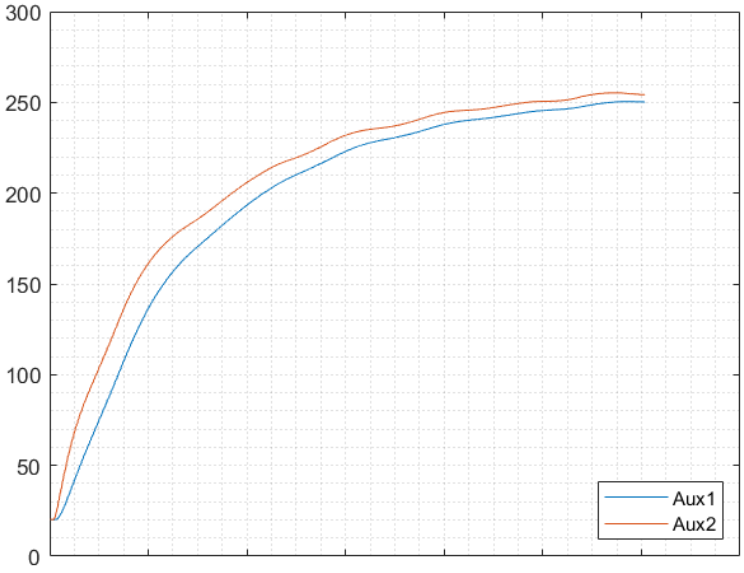
Design of experiment		
name	n.batch	Dataset F_
exp1	5	02-04-05-19-20
exp2	3	02-04-19
exp3	3	03-06-21
exp4	2	03-05
exp5	2	02-21
exp6	2	06-20
exp7	3	03-06-20

Table 3.2: DoE for experimental Data

Next, we will add columns to table 3.2 containing estimated parameter values that will help us say something about the identifiability of these. Meantime, it can be seen that the datasets F_13, F_14, and F_15 are never used in the optimization of the parameters; only later will they be used to validate the goodness of the estimated parameters. Although the datasets contain good data, we are unsure of the nominal temperature of the hot plate. In fact, the PlateTemp [°C] values for these datasets have been modified retrospectively, and the Transitory [s] values refer to the modified temperatures. The last column represents the transient time, which is the time it takes for the entire pad to reach at least 90% of the hot plate temperature at each point. Beyond this time, the steady state begins.

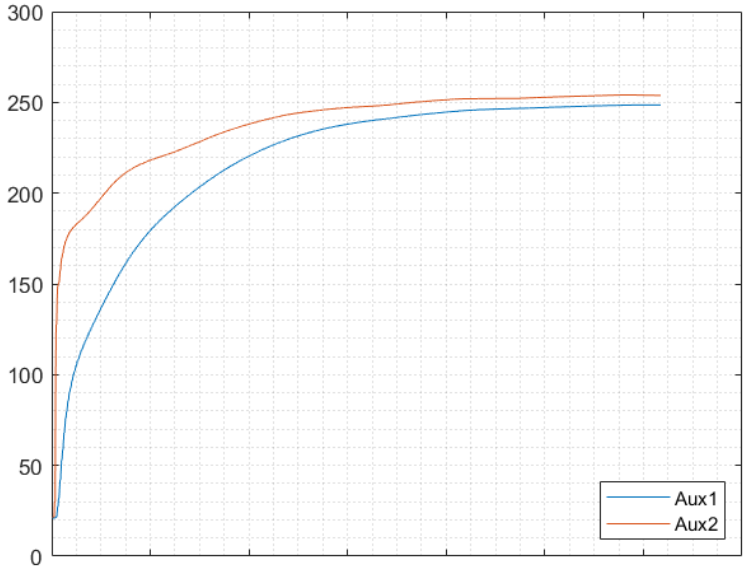


(a) TestF_03

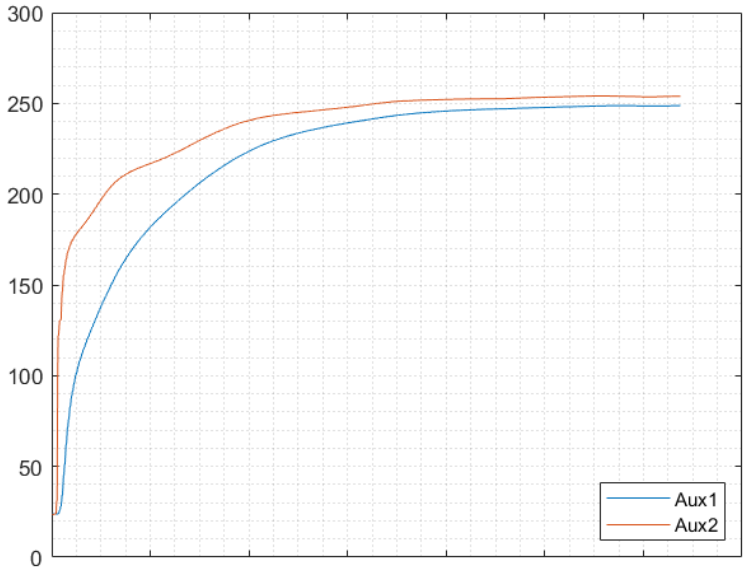


(b) TestF_06

Figure 3.5: Suitable data example 1: minimal oscillations, recordings from sensors consistent with the dynamics



(a) TestF_19



(b) TestF_20

Figure 3.6: Suitable data example 2: minimal oscillations, recordings from sensors consistent with the dynamics

3.2 Synthetic dataset and identificability

To ensure there were no errors in the code and to find the minimal conditions for parameter identifiability, we tested the code in a more "hospitable" environment compared to the more "hostile" experimental one. We created synthetic datasets that mimic experimental one. fig: 3.5, 3.6

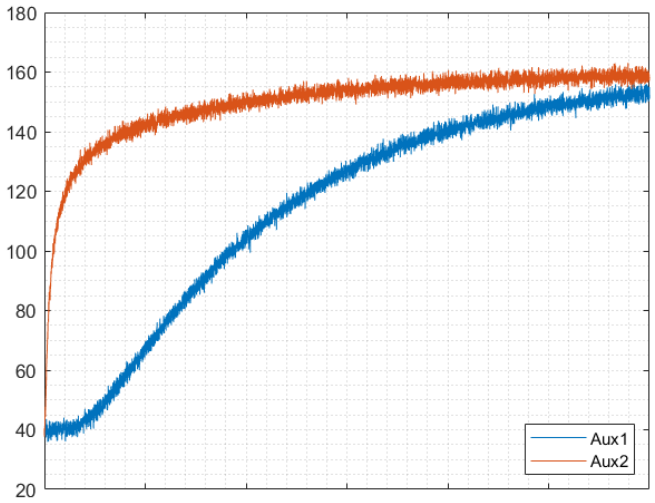
For the construction of our datasets, we assumed a pad with two materials having the following characteristics:

- Variable length:
  - UL of 2 mm
  - FM varying with the following values: $[9, 8, 7, 6, 5, 4, 3, 2]$ mm
- Sensors positioned at:
  - 0 mm
  - 4 mm
  - 8 mm
- Thermal parameters:
  - UL thermal conductivity = k_1
  - product of the UL mass density for the UL specific heat = w_1
  - FM thermal conductivity = k_2
  - product of the FM mass density for the FM specific heat = w_2

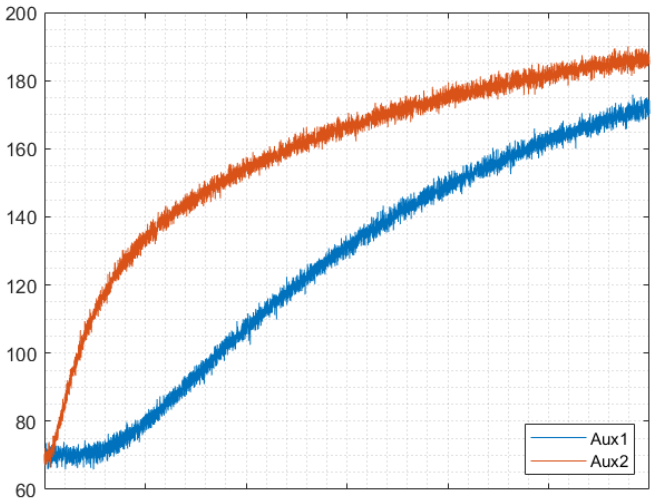
Obviously, in this case, the sensors are fictitious; it's just a way of indicating the physical points on the pad where we took the measurements and where we will make comparisons. When solving the heat equation with the solver, the solution will be outlined on a much finer grid. To match the experimental data conditions, we will only take two of the many spatial points at the positions indicated above. Furthermore, the thermal parameter values are not arbitrary but come from some ITT estimates for the pad used in the experiments.
At this point, the phenomenon of thermal diffusion under these conditions was simulated on this pad using the heat equation solver:

Name	Set of values	Number of values
Right_Boundary_Condition (RBC)	{80, 120, 160, 200}	4
Initial_Condition (IC)	{25, 40, 55, 70}	4
Friction_Material_Thickness (FMT)	{2, 3, 4, 5, 6, 7, 8, 9}	8

A total of 128 datasets were created in this way. However, since the experimental datasets contain measurement errors, a small amount of white noise from a random variable was added to account for this $X \sim \mathcal{N}(0, 1.5)$.



(a) Synthetic data ex 1



(b) Synthetic data ex 2

Figure 3.7: Synthetic data 10^2 Hz

This process of creating and optimizing synthetic data was repeated several times, and the results presented now specifically apply to synthetic data with an acquisition frequency of 10^2 Hz and an acquisition time of T seconds.

3.2.1 Synthetic Dataset 10^2Hz

Experiments were conducted on the following datasets, aiming to change not only the characteristics among the datasets but also varying the number of sensors used:

experimental condition					
name	n. batch	RBC [°C]	IC [°C]	FMT [mm]	n. sensors
exp1	1	160	40	6	BP
exp2	1	160	40	6	BP+1
exp3	2	160	40	9-6	BP
exp4	2	160	40	9-6	BP+1
exp5	2	160	25 - 70	9	BP
exp6	2	160	25 - 70	9	BP+1
exp7	2	80 -160	40	9	BP
exp8	2	80 -160	40	9	BP+1
exp9	2	160	40	3-7	BP
exp10	2	160	40	3-7	BP – BP+1

Table 3.3: Synthetic data DoE

From this experiment, it is found that with this model, we can estimate at most 3 parameters given the fourth parameter is known. In other words, in an experimental environment where the parameters generating the dataset are unknown, we can accurately determine at most the ratios of one parameter to the other three. For this reason, we will assume to know a particular parameter. From this point onward, we will assume to know the parameter k_1, thus obtaining the following results:

percentage error			
name	k2[%]	w1[%]	w2[%]
exp1	44,9912	61,581	6,4524
exp2	0,2816	0,3605	0,0406
exp3	-0,569	-0,4109	-0,5956
exp4	0,3084	0,3847	0,0932
exp5	51,6664	85,1743	14,0633
exp6	-0,0747	0,334	-0,3642
exp7	63,2225	103,3626	16,3023
exp8	-0,0827	0,3929	-0,4207
exp9	-0,5749	-0,4385	-0,581
exp10	-0,488	-0,3627	-0,5415

Table 3.4: Results of the experiments

It was found that the percentage error varies greatly across experiments: some experiments show a very satisfactory percentage error, while others exhibit a significantly large percentage error. Specifically, correct identification was only observed in cases where multiple sensors were used for each dataset, or in cases where the single sensor used is located at the Backplate (BP) position for each dataset, with data containing different pad length. This allows us to formulate the minimal conditions for identifiability.:

- 1 sensor and 2 dataset with different FMT
- 2 sensor and every number of dataset and any conditions

To support this assertion, we report on the method's consistency. Indeed, all experiments were repeated starting from different initial parameters. In cases where identifiability was achieved, this consistently led to a more or less precise vicinity of the parameters used in creating the dataset.

Initial guess			percentage error exp1			percentage error exp2		
w1	k2	w2	w1[%]	k2[%]	w2[%]	w1[%]	k2[%]	w2[%]
w_1	k_2	w_2	61.6%	44.99%	6.4%	0.47%	0.28%	0.0%
$2.33w_1$	$0.25k_2$	$4.0w_2$	-94.3%	127.90%	17.5%	0.47%	0.28%	0.0%
$2.33w_1$	$0.1k_2$	$2.8w_2$	-75.6%	-42.45%	-2.0%	0.47%	0.28%	0.0%
$0.23w_1$	$4.0k_2$	$0.4w_2$	181.4%	160.33%	13.2%	0.47%	0.29%	0.0%
$0.12w_1$	$0.5k_2$	$0.4w_2$	-77.3%	-50.01%	-15.6%	0.47%	0.28%	0.0%
$0.023w_1$	$10k_2$	$0.12w_2$	167.4%	143.74%	17.2%	0.47%	0.28%	0.0%
$11.63w_1$	$5.0k_2$	$0.2w_2$	123.5%	98.44%	10.4%	0.47%	0.29%	0.0%

Table 3.5: Identifiability of thermal parameters

3.2.2 Syntetich Dataset 1Hz

Subsequently, we created datasets even more similar to the experimental ones. The main difference between the previously described datasets and the experimental ones lies in the sensor sampling frequency and the acquisition duration. Specifically, the previously created datasets had many more data points within a very short time interval, which even prevented the thermal dynamics from reaching steady state for some datasets. Therefore, we reduced the sampling frequency to match that of the experimental data (1 Hz) and extended the acquisition time to T_{exp} seconds to align it with the experimental data.

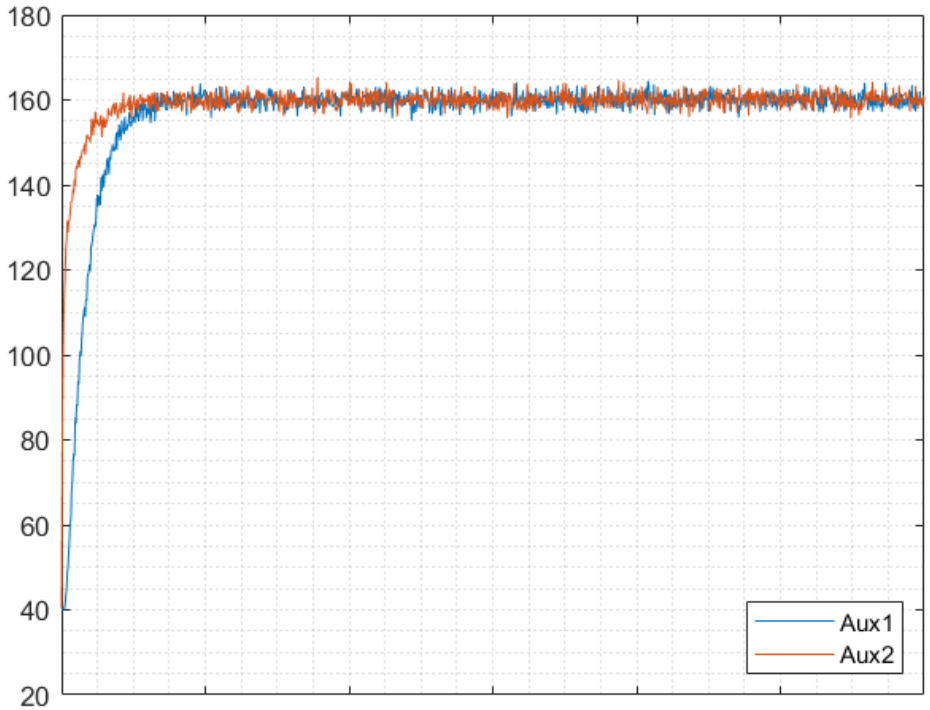


Figure 3.8: Synthetic data ex 3

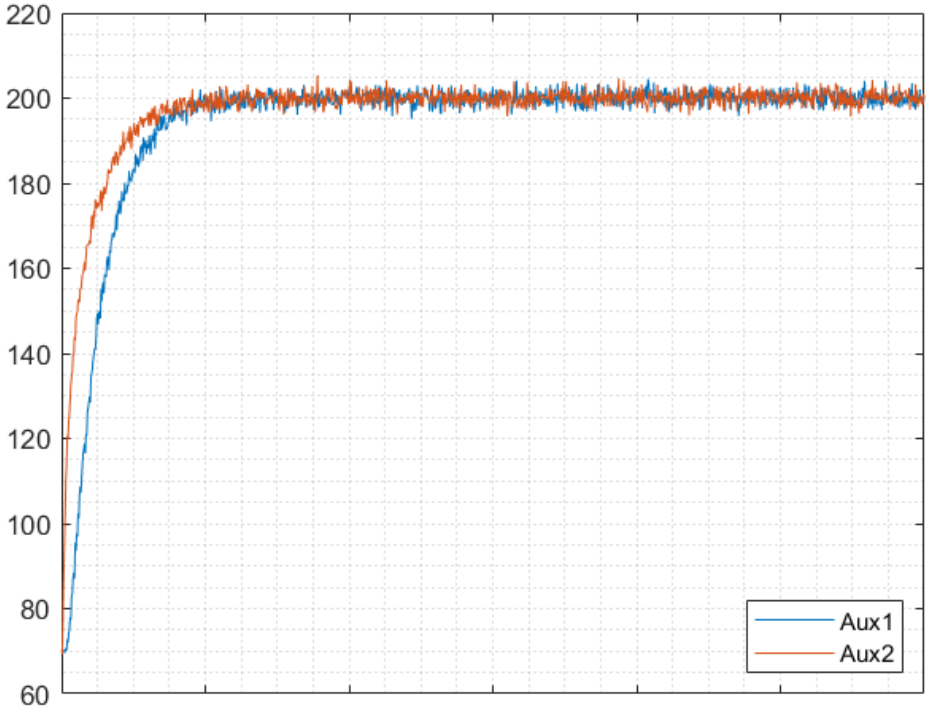


Figure 3.9: Synthetic data ex 4

Figure 3.10: Synthetic data 1Hz

By optimizing for the following experiments the same in table 3.3

esperimental condition					
name	n. batch	RBC [°C]	IC [°C]	FMT [mm]	n. sensors
exp1	1	160	40	6	BP
exp2	1	160	40	6	BP+1
exp3	2	160	40	9-6	BP
exp4	2	160	40	9-6	BP+1
exp5	2	160	25 - 70	9	BP
exp6	2	160	25 - 70	9	BP+1
exp7	2	80 -160	40	9	BP
exp8	2	80 -160	40	9	BP+1
exp9	2	160	40	3-7	BP
exp10	2	200-80	25	2-9	BP+1

The following results are obtained:

percentage error			
name	k2[%]	w1[%]	w2[%]
exp1	2,6935	-82,6019	147,1959
exp2	3,4926	2,1266	4,9475
exp3	8,4806	5,6726	9,2833
exp4	3,19	1,998	4,187
exp5	108,2471	-79,1629	302,9865
exp6	2,7741	1,4114	3,8403
exp7	743,2642	-79,3822	1,58E+03
exp8	3,2417	1,644	4,4929
exp9	3,167	1,737	4,4105
exp10	-0,4569	-0,0581	0,128

These results, though unexpected, are consistent with previous statements. Despite the lack of identifiability where expected, significant changes occur in situations where identifiability should be present. In particular, percentage errors increase significantly compared to previous conditions. This suggests that reducing the sampling frequency increases the error in parameter estimation. However, the results of the last experiment indicate that a careful selection of two datasets for optimization may be ideal to maintain error levels comparable to those previously observed. One possible explanation for this phenomenon may lie in the increased importance of the transient regime compared to the steady state, suggesting that the transient behavior of the system can significantly influence the accuracy of

parameter estimates compared to the steady state. To verify this, the data were segmented to use the transient regime as a baseline, and subsequently adjustments were made by adding or removing a certain percentage of that regime.

	tr = transitory		tr + tr*x						
	x =	30%	15%	10%	5%	0%	-5%	-10%	-20%
	err_perc_k2	*5,50*	7,3671	6,7105	5,17	5,63	5,85	5,97	7,54
	err_perc_w1	*2,91*	4,1303	3,8698	3,25	3,36	3,32	3,91	4,14
RBC_80_IC_25_FMT_9	*err_perc_w2*	*7,62*	9,6672	8,8522	6,96	7,56	7,90	7,75	9,86
	err_perc_k2	*-12,78*	-11,58	-11,86	-11,75	-13,70	-14,24	-14,87	-14,47
	err_perc_w1	*-7,83*	-7,18	-7,32	-7,24	-7,78	-8,20	-8,44	-8,60
RBC_200_IC_70_FMT_2	*err_perc_w2*	*-2,30*	-1,26	-1,52	-1,52	-4,72	-4,81	-5,66	-4,31
	err_perc_k2	*-3,67*	-2,90	-3,30	-3,68	-5,38	-5,66	-5,66	-5,93
RBC_80_IC_25_FMT_9 &	*err_perc_w1*	*-1,57*	-1,26	-1,43	-1,60	-2,02	-2,21	-2,21	-2,53
RBC_200_IC_70_FMT_2	*err_perc_w2*	*-3,32*	-2,29	-2,72	-3,06	-5,55	-5,80	-5,80	-5,76
	err_perc_k2	-1,3034	-1,0753	-1,4797	-2,2131	-2,1478	-2,1574	-2,5874	-2,3852
RBC_80_IC_25_FMT_9 &	*err_perc_w1*	-0,4167	-0,2984	-0,4754	-0,8048	-0,7759	-0,7826	-0,96	-0,8685
RBC_200_IC_70_FMT_2	*err_perc_w2*	-0,8445	-0,5928	-1,0398	-1,8358	-1,7642	-1,7794	-2,2262	-2,0372

Figure 3.11: The transient is the time it takes for the BP (backplate) to reach 95% of the nominal temperature of RBC (right boundary condition).
err_perc: percentage error

It is noted, therefore, that the percentage error on the parameters tends to increase when part of the transient regime is removed from the data, while exceeding in the steady state regime does not improve them much. This observation also allows us to reduce computational time in experimental environments, as steady-state recordings are numerous and we have determined that they are not as useful. This allows us to trim the dataset, reducing computational time without sacrificing accuracy.

3.2.3 Incorrect positioning of sensors

At this point, we questioned the effect of incorrect placement of thermal sensors. Here, we felt the need to change the solver used so far, transitioning from an explicit Euler solver to a Crank-Nicolson solver, to reduce the error in sensor placement due to approximating the sensor's position to its nearest point on the grid. To achieve this, it was necessary to increase the spatial integration step while keeping the temporal integration step unchanged to avoid significantly increasing computational time.

Apart from the possible need to densify the spatial grid, what happens if incorrect information about the positions of the sensors recording temperatures is passed? The result is a loss of parameter identifiability. To investigate this, experiments were conducted on the same synthetic datasets. However, during optimization, incorrect

information about sensor positions was passed. For instance, if the recordings from points [0, 4] were supposed to be the temperatures recorded, in optimization they would be assigned to points [0, 6.4] mm, where the second sensor is positioned 2.4 mm further ahead. Initially, experiments were conducted with a batch number of 1.

esperimental condition						percentage error		
name	n.batch	RBC[°C]	IC[°C]	FMT[mm]	n.sensors	k2[%]	w1[%]	w2[%]
exp1	1	160	40	6	BP+1	-9,38E+01	-1,00E+02	-8,23E+01
exp2	1	160	40	9	BP+1	5,78E+12	-9,16E+01	1,19E+13
exp3	1	160	25	9	BP+1	3,28E+12	-9,14E+01	6,77E+12
exp4	1	200	70	6	BP+1	-9,77E+01	-1,00E+02	-9,35E+01
exp5	1	80	40	9	BP+1	1,23E+13	-8,81E+01	2,55E+13

Table 3.6: RBC: right boundary condition; IC: initial condition; FMT: friction material tickhness

On its own, this table is not sufficient to prove non-identifiability but merely demonstrates that the parameters cannot be accurately estimated. To demonstrate non-identifiability, it is necessary to examine what happens to the objective function and the endpoints for each experiment. In fact, each optimization is repeated for different initial points. This is done to exclude or verify the existence of local minima or entire flat regions, which would demonstrate the identifiability or non-identifiability of the parameters. For convenience and stylistic choice, only one will be reported here, but similar results apply to every other experiment conducted in this manner.

starting points			exp1 percentage error			
start_w1	start_k2	start_w2	w1[%]	k2[%]	w2[%]	Fvalues
w1	k2	w2	-1,00E+02	1,85E+02	7,17E+02	7,78E+03
w1 *2	k2/10	w2/3	-1,00E+02	-9,38E+01	-8,23E+01	7,78E+03
w1 *10	k2 *3	w2/2	-1,00E+02	6,56E+03	1,90E+04	7,78E+03
w1 *8	k2/10	w2/3	-1,00E+02	-9,78E+01	-9,36E+01	7,78E+03

Table 3.7: w1, k2 & w2 real parameters of the datasets created.
Every row is a starting point and an end point of the optimization with it's corresponding value function

It is observed how the value of the objective function reaches the same value despite the initial points being widely spaced, varying by orders of magnitude.
At this point, we wonder if considering a combination of datasets still shows this phenomenon. To investigate this, we will consider a batch number equal to 2 and vary the plate length, edge temperature, and initial temperature.

esperimental condition						percentage error		
name	n.batch	RBC[°C]	IC[°C]	FMT[mm]	n.sensors	k2[%]	w1[%]	w2[%]
exp1	2	160	40	9-6	BP+1	6,47E+01	-7,55E+01	2,48E+02
exp2	2	160	25-70	9	BP+1	3,95E+12	-9,13E+01	8,16E+12
exp3	2	80-160	40	9	BP+1	3,78E+12	-9,12E+01	7,82E+12

Table 3.8: RBC: right boundary condition; IC: initial condition; FMT: friction material tickhness

In this case as well, parameter identifiability, with the exception of the first experiment which, despite yielding incorrect parameters, converges to a different endpoint for each point. In this instance as well, only two examples will be reported.

starting points			exp3 percentage error			
start_w1	start_k2	start_w2	w1[%]	k2[%]	w2[%]	Fvalues
w1	k2	w2	-9,12E+01	2,35E+12	4,85E+12	2,03E+03
w1*2	k2/10	w2/3	-1,00E+02	-9,77E+01	-9,52E+01	2,08E+03
w1*10	k2*3	w2/2	-9,12E+01	3,78E+12	7,82E+12	2,03E+03
w1*8	k2/10	w2/3	-1,00E+02	-9,83E+01	-9,65E+01	2,08E+03

Table 3.9: Exp3 same value function very different set of points.
Every row is a starting point and an end point of the optimization with it's corresponding value function

starting points			exp1 percentage error			
start_w1	start_k2	start_w2	w1[%]	k2[%]	w2[%]	Fvalues
w1	k2	w2	-7,55E+01	6,47E+01	2,48E+02	3,23E+04
w1*2	k2/10	w2/3	-7,55E+01	6,47E+01	2,48E+02	3,23E+04
w1*10	k2*3	w2/2	-7,55E+01	6,47E+01	2,48E+02	3,23E+04
w1*8	k2/10	w2/3	-7,55E+01	6,47E+01	2,48E+02	3,23E+04

Table 3.10: Exp1 converges exceptionally to three parameters.
Every row is a starting point and an end point of the optimization with it's corresponding value function

The fact that a particular combination of datasets converges indicates that incorrect sensor placement can have a more or less significant effect depending on the datasets chosen for optimization.

3.3 Esperimental data results

In this section, the method's performance on real data will be analyzed using the same optimization methodology as for synthetic data. The only difference is that in this case, the actual values of the pad will not be available for comparison with the estimated parameters. Therefore, the only measure of parameter estimation quality will be the value of the objective function, which will depend strictly on the datasets selected for optimization. Additionally, instead of reporting the optimized parameters, a percentage error relative to an estimate of the thermal parameters of the pad from ITT will be reported, although these estimates should not be considered the real parameters.

An initial attempt on the set of experiments from Table 3.2 was conducted assuming all nominal conditions of the experiments were correct. These experiments, conducted both using a single sensor (1s) and two sensors (2s), yielded the following results:

name	1s percentage error				2s percentage error			
	w1[%]	k2[%]	w2[%]	Fvalues	w1[%]	k2[%]	w2[%]	Fvalues
exp1	-1,42E+01	-92,79	-9,75E+01	1,78E+05	-1,00E+02	-100,00	-1,00E+02	7,64E+05
exp2	-1,49E+01	-93,04	-9,76E+01	1,08E+05	-1,00E+02	-100,00	-1,00E+02	4,29E+05
exp3	-4,88E+01	-95,72	-9,85E+01	1,17E+05	-1,00E+02	-100,00	-1,00E+02	3,92E+05
exp4	-5,62E+01	-95,30	-9,37E+01	4,50E+04	-1,00E+02	-100,00	-1,00E+02	1,16E+05
exp5					-1,00E+02	-100,00	-1,00E+02	1,46E+05
exp6	1,93E+01	-89,70	-9,77E+01	6,35E+04	-1,00E+02	-100,00	-1,00E+02	3,30E+05
exp7	-9,89E+01	-99,91	-1,00E+02	1,07E+05	-1,00E+02	-100,00	-1,00E+02	4,11E+05

The first thing that stands out when looking at the table above is that exp 5, when attempting to optimize with a single sensor, does not provide any parameter estimates. This effect is not yet well understood; it is known only that this particular choice of dataset with the grid used so far leads, during optimization, to the construction of poorly conditioned matrices that prevent the solver from correctly solving the differential equations. Furthermore, from experiments on synthetic datasets, we know that exp4, performed with only one sensor, should not be identifiable. This is confirmed by examining what happens at each initial point, in fact as before, each optimization is repeated for different initial points.

starting points			exp4 percentage error			
start_w1	start_k2	start_w2	w1[%]	k2[%]	w2[%]	Fvalues
w1	k2	w2	-1,00E+02	8,48E-01	3,85E+02	8,41E+04
w1*2	k2/10	w2/3	-5,62E+01	-9,53E+01	-9,37E+01	4,50E+04
w1*10	k2*3	w2/2	1,10E+03	2,88E+01	-8,63E+01	4,50E+04
w1*8	k2/10	w2/3	-5,62E+01	-9,53E+01	-9,37E+01	4,50E+04

It is noticeable how, for each initial point, there exists a vastly different minimum

output, even with the same objective function value.
For optimization with 2 sensors, however, it is observed that not only do significantly different values consistently result, but they are also orders of magnitude smaller compared to those with 1 sensor. Upon examining each experiment, it is discovered that there is no identifiability. This is quite surprising, as what holds true for synthetic datasets should remain valid for real ones, barring measurement errors. However, the nature of the error suggests a serious issue in the positioning of the second thermal sensor, as observed in synthetic data as well.

starting points			exp1 percentage error			
start_w1	start_k2	start_w2	w1[%]	k2[%]	w2[%]	Fvalues
w1	k2	w2	-1,00E+02	-1,00E+02	-1,00E+02	7,64E+05
w1*2	k2/10	w2/3	-1,00E+02	-1,00E+02	-1,00E+02	7,64E+05
w1*10	k2*3	w2/2	-1,00E+02	-1,00E+02	-1,00E+02	7,64E+05
w1*8	k2/10	w2/3	-1,00E+02	-1,00E+02	-1,00E+02	7,64E+05

Table 3.11: Optimization over exp1 with 2s is reported in table as example of non identificability of parameters, we enter in a region of parameters increasingly smaller due to the non-identifiable nature of 3 parameters

3.3.1 Corrected optimization

By hypothesizing a different position for the second sensor from its nominal position, it is discovered that the data aligns better when comparing the sensor's readings to a point on the grid that is at least 1.4 mm behind the nominal position. This does not mean that the sensor's actual physical position changes; instead, we assume a different sensor position for comparison purposes. In this sense, we "shift the position" by comparing the data to a point on the grid that is at least 1.4 mm behind the nominal position. Using this correction the parameters of the pad can be identified. This results in a range of valid values for each experiment similar to those obtained in the 1s optimization.

name	2s corrected percentage error			
	w1[%]	k2[%]	w2[%]	Fvalues
exp1	6,94E+01	-81,19	-7,37E+01	5,25E+05
exp2	8,73E+01	-79,64	-7,14E+01	3,19E+05
exp3	8,42E+01	-79,94	-7,50E+01	3,22E+05
exp4	-3,61E+01	-92,25	-8,74E+01	9,21E+04
exp5	9,34E+01	-81,30	-7,71E+01	2,34E+05
exp6	8,74E+01	-78,69	-7,31E+01	1,96E+05
exp7	4,98E+01	-83,39	-7,76E+01	2,89E+05

Obtaining in this case, as well, physically admissible values according to ITT engineers' judgment suggesting that the model is reliable. Indeed, the only cases where convergence cannot be achieved are due to the absence of minimal identifiability conditions or due to poor matrix conditioning. The reason for the divergence in values between optimization with 1s and 2s can be explained by the poor positioning of the second sensor, a sensor that we disregard in the 1s setting. This poor positioning can be understood by examining the images of the tests conducted. We know, in fact, from Figure 3.1, that it should be located halfway through the friction material, however

Figure 3.12: Real distribuition of the sensors in the Pad used to collect data. The 2nd sensor position could not be the one expected

The image shows how this thermocouple penetrates into the pad as it erodes up to 60% of its friction material (FM). Without knowing the correct position of the thermocouple, assuming an incorrect positioning of the sensor by at least 1.4 mm backward could be the real reason for non-identifiability in the 2s setting without correction.

3.4 Methods and pipeline

3.4.1 PDE solvers

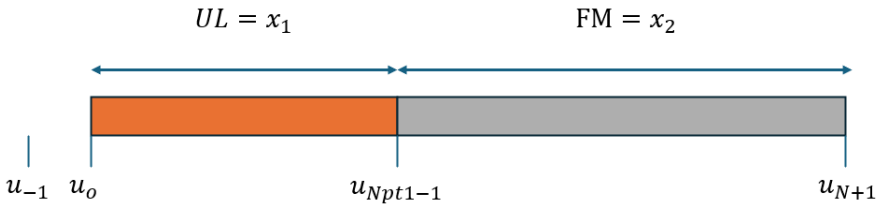
In this section, we refer to the dissemination model described in section 1.4, which will be presented in detail to illustrate how the numerical method solving it was

constructed. We begin with the generic thermal diffusion equation 1.2, adopting an initial simplification where the thermal parameters of the two materials remain invariant over time and space. By considering the steady-state case of the dynamics, we obtain

$$\mu \frac{\partial^2 u}{\partial x} = f \tag{3.1}$$

where $\mu = k$ e $f = \rho q$, at this point we are going to discretize in space, paying close attention to the notation by distinguishing between points of the for which we must calculate the solution, i.e. points of discretization located on the pad and for which the solution must be calculated, and internal points for which we do not have to calculate a solution, i.e. points on the pad for which we do not have to calculate the solution and fictitious points, points for which we do not have to calculate the solution but which are useful for calculating it. it is crucial to obtain

Figure 3.13: Space Pad discretization



a good solution that there is a point where the two materials meet in the image called u_{Npt1-1}. For this reason, the integration step is decided and maintained based on the number of points we want in the underlayer $Npt1$. Therefore, if we denote h as the integration step, we have that

$$h = \frac{x_1}{Npt1}$$

At this point, replacing the second spatial derivative with its approximation by central difference, we obtain

$$\frac{\mu}{h^2}(-u_{j-1} + 2u_j - uj + 1) = f \quad \forall j \quad s.t. \quad 1 \leq j \leq N$$

This equation is valid only for the interior points. For the boundary points, we need to consider the boundary conditions. Specifically, we impose a Dirichlet condition on the right side, hence $u_{N+1} = g(t)$. As for the left side, with a Neumann condition, we use a fictitious point to calculate the temperature at u_0. We know that

$$\mu \frac{\partial u}{\partial x}(0) = \psi(0)$$

where $\phi(0)$ represents the flux at point 0. Therefore, it is possible to derive u_0 as

$$\frac{\mu}{h^2}(-u_{-1} + 2u_0 - u1) = f$$

We know also that

$$\mu \frac{u_{-1} - u_1}{2h} = \psi(0)$$
$$u_{-1} = u_1 + \frac{2h}{\mu}\psi(0)$$

Substituting u_{-1} into the previous equation, we find

$$\frac{\mu}{h^2}(u_0 - u_1) = \frac{1}{2}f(0) + \frac{1}{h}\psi(0) =: f_{N+1}.$$

We can then summarise this in matrix form as

$$Au = f \tag{3.2}$$

where f is the vector of incoming external heat at each discretization point of the Pad. In our case, it will be $f = [0, ...,0, \frac{\mu_2}{\rho_2} g(T)]$, a vector with all entries zero except the last one, which corresponds to the heat value at the boundary temperature. A is a matrix that would be $A = h^{-2}\mu \cdot \text{tridiag}[1, -1; -1,2, -1]$ if our model consisted of a single material. Since we have two materials, this matrix is the product of a matrix $\mu = \text{diag}[\mu_1, ..., \mu_1, \mu_2, ..., \mu_2]h^{-2}$ and tridiag$[1, -1; -1,2, -1]$. It will later be adjusted to account for additional continuity conditions of heat flux and temperature at the boundary between the two materials.

$$A = \frac{1}{h^2}\begin{bmatrix} \mu_1 & 0 & \dots & \dots & \dots & 0 \\ 0 & \ddots & \ddots & & & \vdots \\ \vdots & \ddots & \mu_1 & \ddots & & \vdots \\ \vdots & & \ddots & \mu_2 & \ddots & \vdots \\ \vdots & & & \ddots & \ddots & 0 \\ 0 & \dots & \dots & \dots & 0 & \mu_2 \end{bmatrix} \cdot \begin{bmatrix} 1 & -1 & 0 & \dots & \dots & 0 \\ -1 & 2 & -1 & \ddots & & \vdots \\ 0 & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & 0 \\ \vdots & & \ddots & \ddots & \ddots & \ddots \\ 0 & \dots & \dots & 0 & -1 & 2 \end{bmatrix} \tag{3.3}$$

Overall, the equation $Au = f$ thus becomes

$$\begin{bmatrix} \frac{\mu_1}{h^2} & 0 & \dots & \dots & \dots & 0 \\ 0 & \ddots & \ddots & & & \vdots \\ \vdots & \ddots & \frac{\mu_1}{h^2} & \ddots & & \vdots \\ \vdots & & \ddots & \frac{\mu_2}{h^2} & \ddots & \vdots \\ \vdots & & & \ddots & \ddots & 0 \\ 0 & \dots & \dots & \dots & 0 & \frac{\mu_2}{h^2} \end{bmatrix} \cdot \begin{bmatrix} 1 & -1 & 0 & \dots & \dots & 0 \\ -1 & 2 & -1 & \ddots & & \vdots \\ 0 & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & 0 \\ \vdots & & \ddots & \ddots & \ddots & \ddots \\ 0 & \dots & \dots & 0 & -1 & 2 \end{bmatrix} \cdot \begin{bmatrix} u_0 \\ \vdots \\ u_{Npt1-1} \\ u_{Npt1} \\ \vdots \\ u_N \end{bmatrix} = \begin{bmatrix} 0 \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \frac{\mu_2}{\rho_2} g_2 \end{bmatrix}$$

It is now possible to rewrite the system as

$$\begin{cases} Bu' + Au = f, \\ u(0) = u_0, \end{cases}$$

$$\begin{cases} u' = B^{-1}Au + B^{-1}f \\ u(0) = u_0, \end{cases}$$

Where B is called the mass matrix, which in our case is a diagonal matrix containing at each entry the product $c\rho = w$ corresponding to the material at that point, thus $B = \text{diag}[w_1, \dots, w_1, w_2, \dots, w_2]$.

The choice of the temporal discretization method also defines the final method. For example, opting for the Euler scheme results in the Euler method, while advancing with the trapezoidal rule yields the Crank-Nicolson method.

$$\begin{cases} Bu' + Au = f, \quad \forall 0 < t < T \\ u(0) = u_0, \end{cases}$$

The final method is also defined by the choice of temporal discretization. For example, opting to proceed with the Euler scheme results in the Euler method, while advancing with the trapezoidal rule yields the Crank-Nicolson method.

Thus, using Euler, we obtain the following final scheme:

$$u^{k+1} = u^k - \Delta t B^{-1} A u^k + \Delta t f(t_k)$$

With the trapezoid method instead

$$\left(B + \tfrac{\Delta t}{2} A\right) u^{k+1} = \left(B - \tfrac{\Delta t}{2} A\right) u^k + \tfrac{\Delta t}{2}(f(t_k) + f(t_{k+1}))$$

Finally, once we have decided which method of temporal discretisation to adopt, we go on to modify the A matrix so that it contains the conditions of matching

conductivity and diffusivity at the interface of the two materials [4]:

$$
\begin{aligned}
&A(Npt1+1, Npt1-1) = \tau((2k_1+3k_2)a_1 - a_2k_1),\\
&A(Npt1+1, Npt1) = \tau(-2(k_1+3k_2)a_1 + 4a_2k_1),\\
&A(Npt1+1, Npt1+1) = 1,\\
&A(Npt1+1, Npt1+2) = \tau(4a_1k_2 - 2(3k_1+k_2) * a_2),\\
&A(Npt1+1, Npt1+3) = \tau(-a_1k_2 + (3k_1+2k_2) * a_2),
\end{aligned}
$$

Below is the pseudocode for explicit Euler (EE) and Crank-Nicolson (CN) respectively

Algorithm 1 Explicit Euler solver (EE).

procedure EE($k1, k2, w1, w2, ic, Npt1, time\_step\_num, dx, T\_OA\_Int, N, M$)
    ▷ $k1$ Thermal conductivity UL
    ▷ $k2$ Thermal conductivity FM
    ▷ $w1$ UL mass density * UL specif heat
    ▷ $w2$ FM mass density * FM specif heat
    ▷ ic vector of initial temperatures
    ▷ $time\_step\_num$ Integration step over time
    ▷ dx Integration step over space
    ▷ T_OA_Int vector of boundary temperatures
    ▷ N Number of points in the numeric grid over space
    ▷ M Number of points in the numeric grid over time
    ▷ Initialization and construction of matrix A
    ▷ Execution
    for $j = 1 : M$ **do**
        $f \leftarrow [zeros(1, N), \lambda 2 * T\_OA\_Int(j)]'$
        $u(:, J+1) \leftarrow Au(:, j) + f$
    end for
    return u ▷ Solution u returned
end procedure

Algorithm 2 Crank-Nicolson solver (CN).

procedure CN($k1, k2, w1, w2, ic, Npt1, time\_step\_num, dx, T\_OA\_Int, N, M$)
    ▷ $k1$ Thermal conductivity UL
    ▷ $k2$ Thermal conductivity FM
    ▷ $w1$ UL mass density * UL specif heat
    ▷ $w2$ FM mass density * FM specif heat
    ▷ ic vector of initial temperatures
    ▷ $time\_step\_num$ Integration step over time
    ▷ dx Integration step over space
    ▷ T_OA_Int vector of boundary temperatures
    ▷ N Number of points in the numeric grid over space
    ▷ M Number of points in the numeric grid over time
    ▷ Initialization and construction of matrix A
    $B \leftarrow (2 * I) - A$
    $C \leftarrow inv(B)$
    ▷ Execution
    for $j = 1 : M$ **do**
        $f \leftarrow [zeros(1, N), 2 * \lambda 2 * T\_OA\_Int(j)]'$
        $u(:, J + 1) \leftarrow C * Au(:, j) + C * f$
    end for
    return u ▷ Solution u returned
end procedure

3.4.2 Optimizer and objective function

The optimizer function used in this work is fminsearch from Matlab, which implements the Nelder-Mead method, a well-known optimization algorithm based on the simplex method. This method does not require the calculation of derivatives, making it particularly useful for problems where objective functions are nonlinear and non-differentiable. The role of fminsearch is to optimize the model parameters by iteratively invoking the objective function, which in turn calls the heat equation solver to evaluate the proposed solutions.

The objective function is crucial as it determines the quality of the solutions proposed by the optimizer: a lower value of the objective function indicates a better solution. In our case, the objective function is defined as the Sum of Squared Errors (SSE), which measures the discrepancy between observed and predicted values. Formally, the SSE is expressed as:

$$SSE = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$

where y_i are the observed values and $\hat{y}_i$ are the predicted values.
Developing an appropriate objective function required numerous attempts and adjustments, given its central importance in any optimization algorithm. Initially, the objective function was designed to be a Mean Squared Error (MSE), a normalized version of SSE:

$$MSE = \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$

This version was intended not only to optimize the thermal parameters of the system but also to address potential data delays and sensor mispositioning. However, these additional functionalities proved to be computationally too demanding and did not lead to significant improvements in the final solutions.

Ultimately, it was decided to simplify the objective function, retaining only the sum of squared errors. This choice was driven by the need for a balance between computational complexity and optimization effectiveness. The current objective function can take any number of datasets as input and build solutions that are comparable to the input data, providing a robust and practical method for optimizing the thermal parameters of the system.

In summary, the implemented objective function was designed to effectively evaluate the quality of the solutions proposed by the optimization algorithm while ensuring a sustainable computational process. This balance between accuracy and computational efficiency is crucial for the success of the optimization in the context of heat differential equations.

Algorithm 3 f_obj.

procedure F_OBJ($x, ic, Npt1, dt, dx, TempSensors, Nv, Mv, TBpCut, TSacq$)
    ▷ x Parameters proposed from fminsearch
    ▷ $TempSensors$ Data structure which contain every selected Dataset
    ▷ ic vector of initial temperatures
    ▷ dt Integration step over time
    ▷ $TSacq$ Frequency of data acquisition
    ▷ dx Integration step over space
    ▷ $TBpCut$ vector of boundary temperatures
    ▷ Nv Number of points in the numeric grid over space
    ▷ MV Number of points in the numeric grid over time
    ▷ Execution
    for $i = 1 : length(TempSensors)$ **do**
        $temp\_sim \leftarrow PDE\_solver$
        $temp\_sim\_sel \leftarrow downsample(temp\_sim)$
        $sse\_v(i) \leftarrow sum(sum(temp\_sim\_sel - Temp\_sensor)^2)$
    end for
    $sse \leftarrow sum(sse\_v)$
    return sse ▷ Return SSE
end procedure

3.4.3 Optimization pipeline

The optimisation pipeline can be summarised by the following flow charts showing how the functions are nested starting:

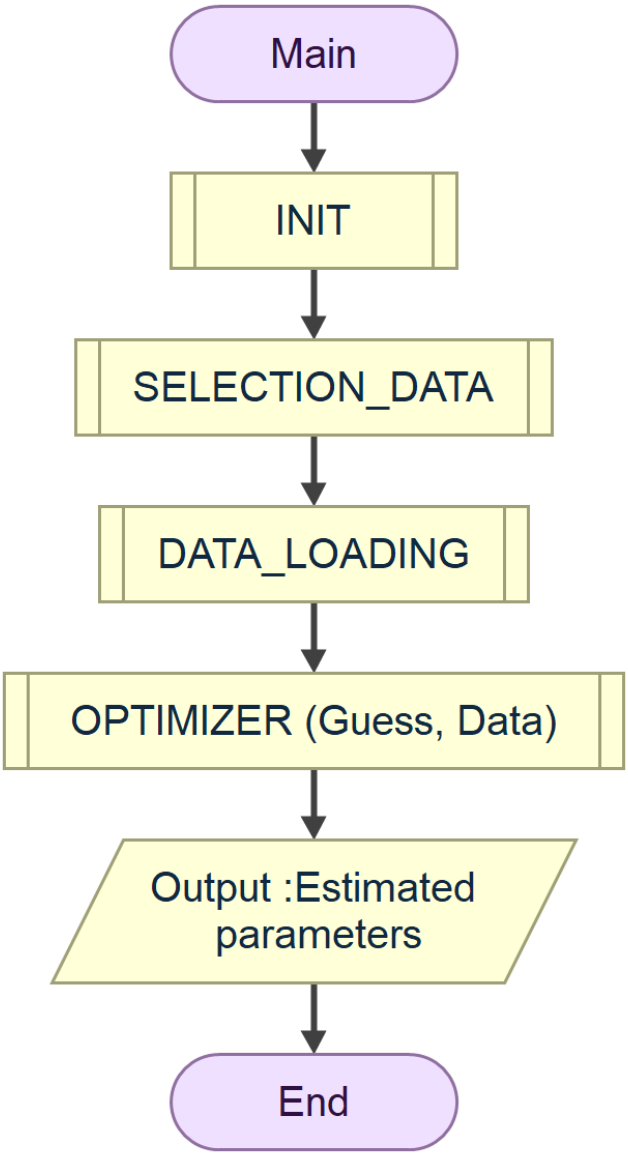


Figure 3.14: Main flowchart

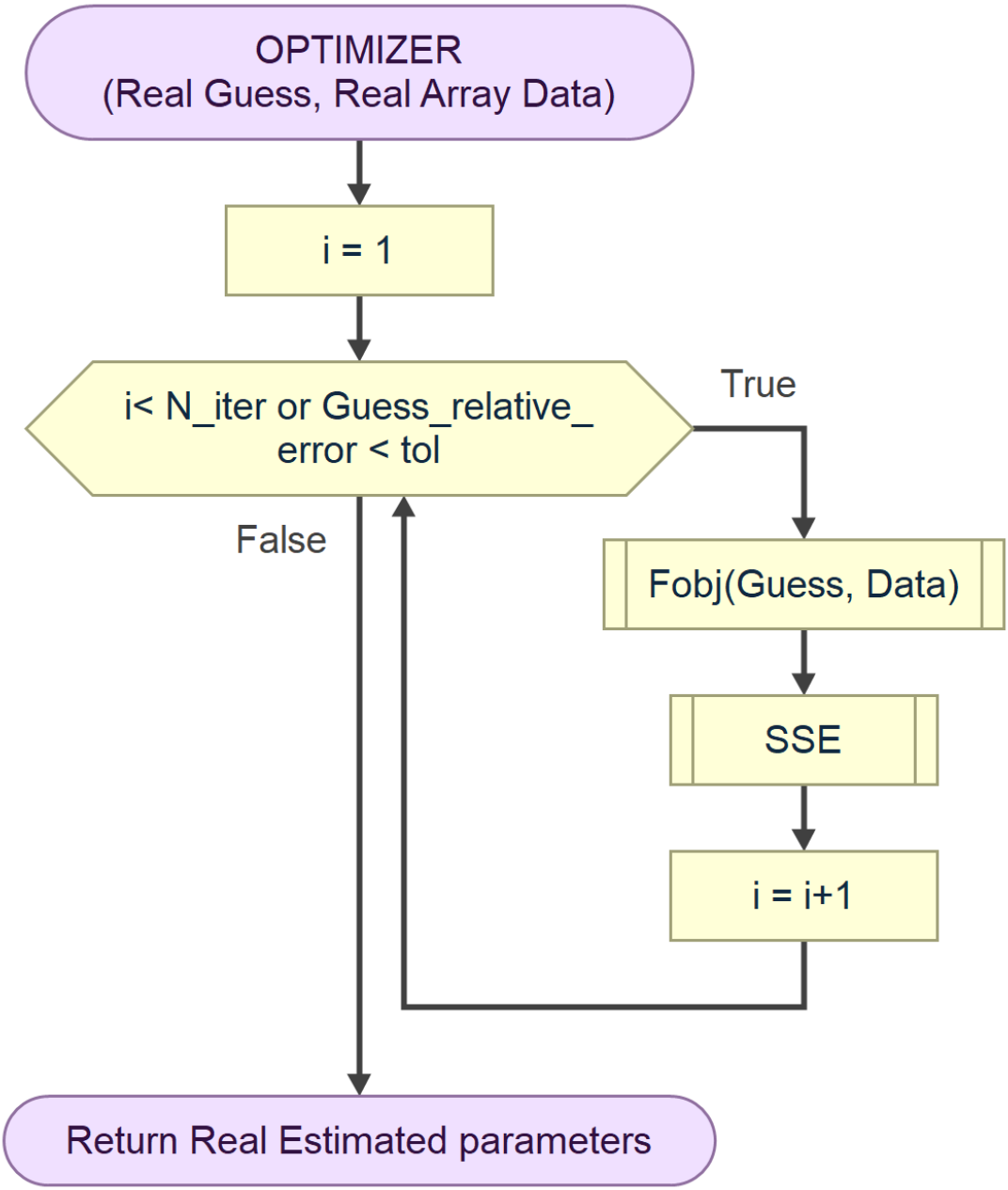


Figure 3.15: Optimizer flowchart

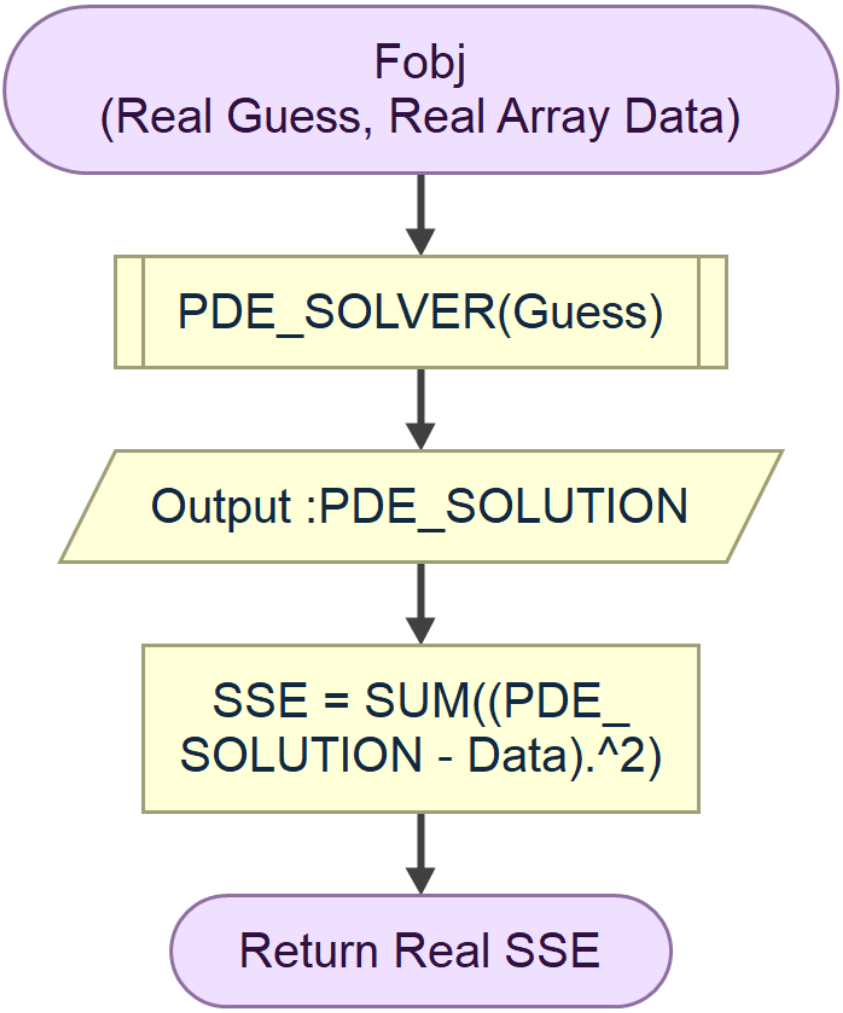


Figure 3.16: Fobj flowchart

The main is nothing more than the script that starts the optimisation, which calls the optimiser function, which in turn calls the objective function that calls the differential equation solver chosen for the optimisation

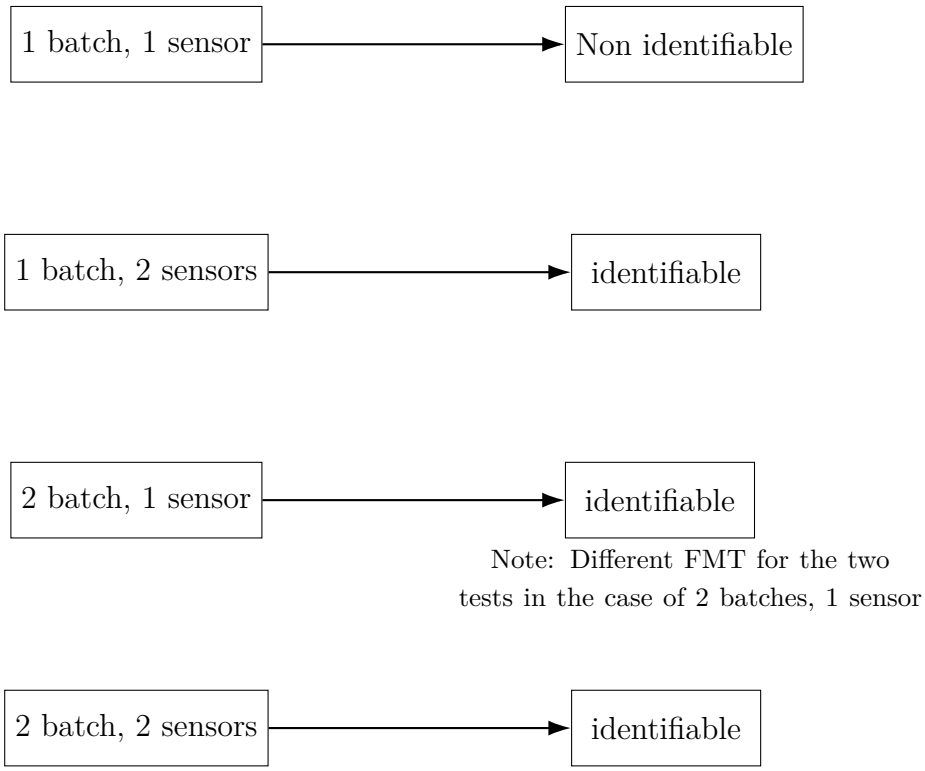
```
Function Main
    Call INIT
    Call SELECTION_DATA
    Call DATA_LOADING
    Call OPTIMIZER (Guess, Data)
    Output :Estimated parameters
End

Function OPTIMIZER (Real Guess, Real Array Data)
    Assign i = 1
    While i< N_iter or Guess_relative_error < tol
        Call Fobj(Guess, Data)
        Call SSE
        Assign i = i+1
    End
    Return Real Estimated parameters
End

Function Fobj (Real Guess, Real Array Data)
    Call PDE_SOLVER(Guess)
    Output :PDE_SOLUTION
    Assign SSE = SUM((PDE_SOLUTION - Data).^2)
    Return Real SSE
End
```

Chapter 4

Conclusion

At the end of this thesis, we can assert that we are able to estimate with relative accuracy three out of the four thermal parameters, provided that minimal identifiability conditions are met, which we have derived based on a thorough analysis of the data.

Identifiable Conditions



For the acquisition of experimental data, it is essential to consider the following factors to minimize percentage errors:

- Higher sampling frequency: A higher sampling frequency can provide more accurate data in the transient regime.
- Lower acquisition time: Similarly, reducing the acquisition time can speed up the optimization process by providing fewer data points in the steady-state regime.

Clearly, to achieve a smaller percentage error, it is necessary and sufficient to increase the sampling frequency, but decreasing the acquisition time is neither necessary nor sufficient. This consideration implies that if these two quantities could be adjusted freely to maintain their product constant

$$T_{acq} * f_{acq} = c \quad c \in \mathcal{N}$$

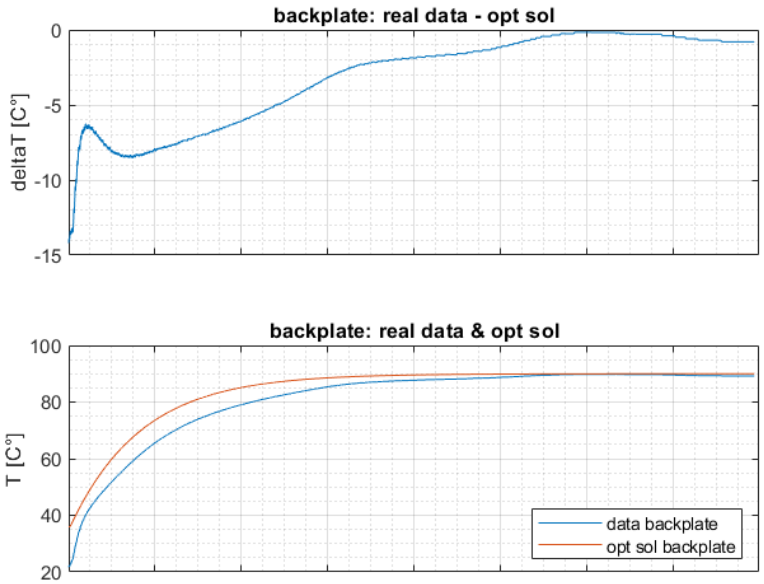
I should prefer to increase the sampling frequency at the expense of the acquisition time.

Furthermore, since we have been able to find a range of thermal parameter values from real data in a reasonable physical region, we can assert that the proposed two-material one-dimensional model for predicting the backplate temperature of the brake pad is a valid model and can have practical applications within a vehicle. Comparing the backplate temperatures from real data with those found by solving the equation further supports its validity. Additionally, being aware of an error in the positioning of the second sensor and still being able to identify the parameters with a single sensor, if I have to rely on parameter estimates, I would lean towards those obtained from the 1s optimization.

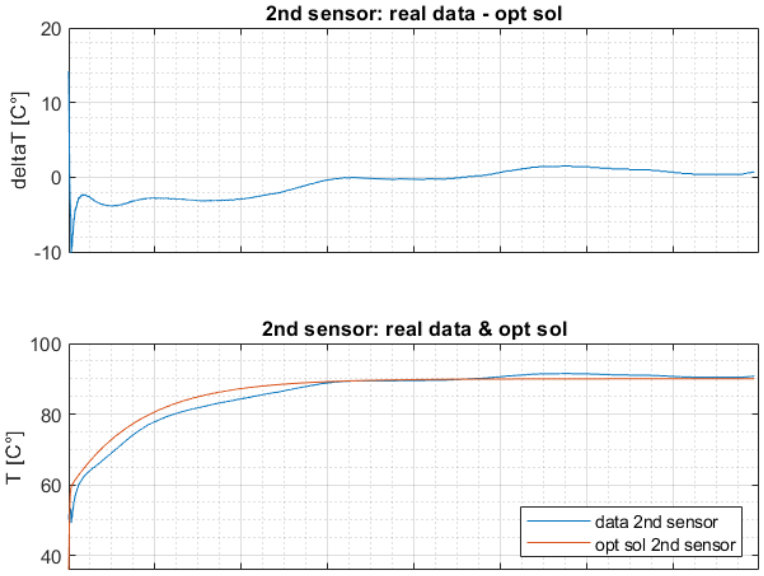
Although the work presented demonstrates the effectiveness of the one-dimensional model, there is still room for improvement. One promising direction involves exploring Physically Informed Neural Networks (PINNs) [5] [6] [7]. These neural networks incorporate not only data but also the governing equations of the system during the training process. PINNs can be particularly beneficial for estimating thermal parameters of the brake, as they allow for the training of both network parameters and the parameters of the differential equations themselves.

In the specific case of brake temperature modeling, PINNs could be utilized to estimate the thermal parameters more accurately. Moreover, their implementation on a vehicle would eliminate the need to solve differential equations with a traditional solver. Instead, we would input the relevant information into the neural network, which would then output the temperature distribution across the brake pad.

By employing a sufficiently complex network, it would also be possible to capture the cooling phenomena of the brake within the wheel well of a moving vehicle, which is depicted as the "complex problem" in Figure 1.2. This approach would bypass the need to explicitly model the cooling process, as the neural network would automatically learn to account for it based on the input variables believed to influence cooling.

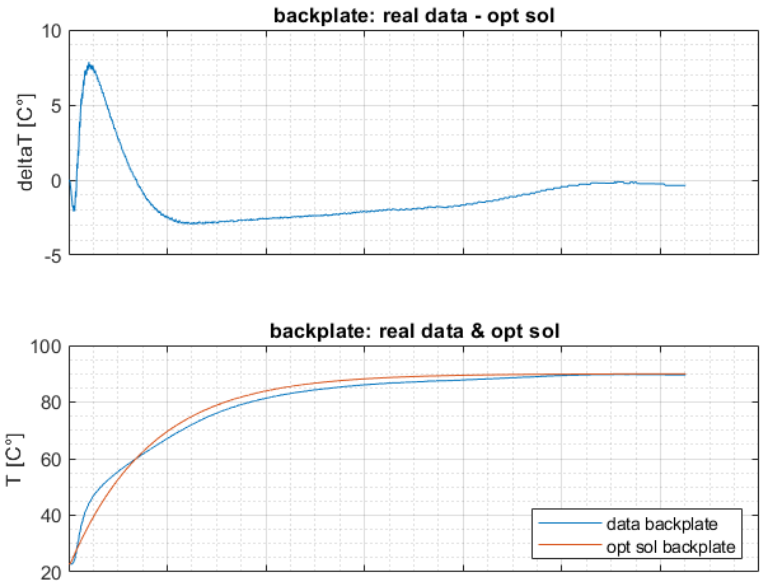


(a) TestF_13 BP

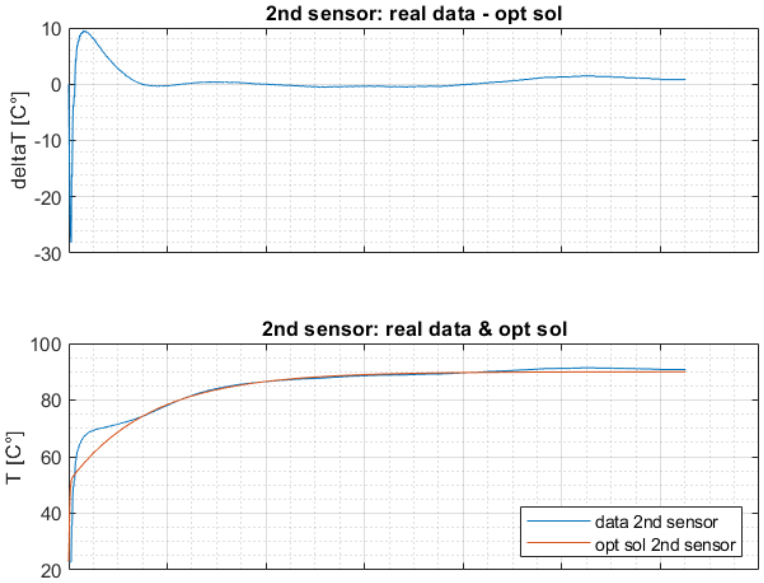


(b) TestF_13 2s

Figure 4.1: Error over dataset F_13

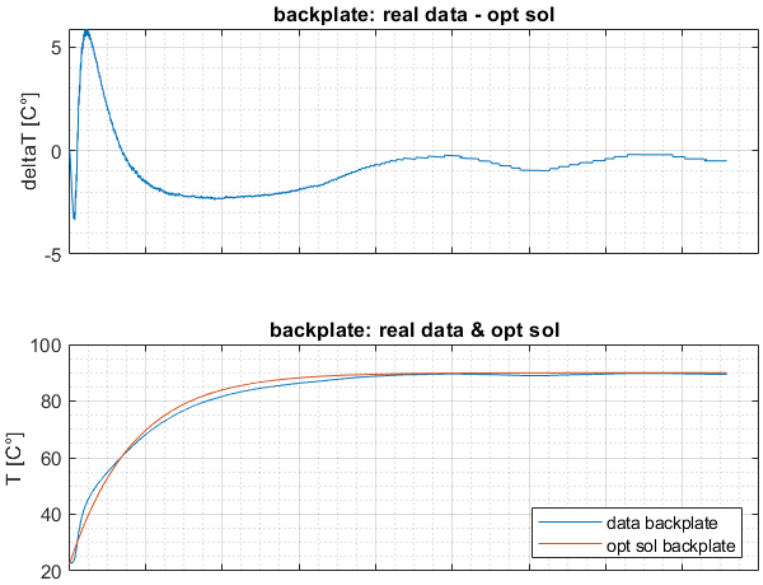


(a) TestF_14 BP

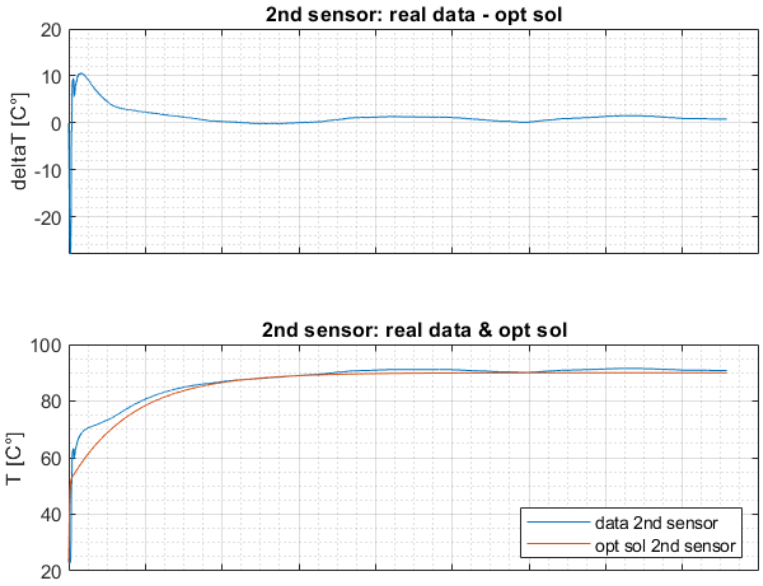


(b) TestF_14 2s

Figure 4.2: Error over dataset F_14



(a) TestF_15 BP



(b) TestF_15 2s

Figure 4.3: Error over dataset F_15

Bibliography

[1] WALTER A. STRAUSS. *PARTIAL DIFFERENTIAL EQUATIONS AN INTRODUCTION*. Laurie Rosatone, 2007 (cit. on p. 5).

[2] Arieh Iserles. *A First Course in the Numerical Analysis of Differential Equations*. 2009 (cit. on p. 9).

[3] C.T. Kelley. *Iterative Methods for Optimization*. 1999 (cit. on p. 16).

[4] R.I. Hickson, S.I. Barry, G.N. Mercer, and H.S. Sidhu. «Multilayer Diffusion». In: *Journal of Applied Mathematics* 44.2 (2009), pp. 567–589 (cit. on p. 43).

[5] Maziar Raissi, Paris Perdikaris, and George Em Karniadakis. «Physics Informed Deep Learning (Part I): Data-driven Solutions of Nonlinear Partial Differential Equations». In: *arXiv preprint arXiv:1711.10561* (2017) (cit. on p. 53).

[6] Maziar Raissi, Paris Perdikaris, and George Em Karniadakis. «Physics Informed Deep Learning (Part II): Data-driven Discovery of Nonlinear Partial Differential Equations». In: *arXiv preprint arXiv:1711.10566* (2017) (cit. on p. 53).

[7] Maziar Raissi, Paris Perdikaris, and George E Karniadakis. «Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations». In: *Journal of Computational Physics* 378 (2019), pp. 686–707 (cit. on p. 53).