UNIVERSIDADE DO ALGARVE

INTELLIGENT MODELLING OF TEMPERATURE PROPAGATION INDUCED BY THERAPEUTIC ULTRASOUND

Raul Ferreira

Dissertação

Mestrado Integrado em Engenharia Electrónica e Telecomunicações

Trabalho efectuado sob a orientação de:

Prof. Dra. Maria Graça Ruano

2015

Declaration of Authorship

I, Raul Manuel Ribeiro Ferreira, declare that this thesis titled, 'Intelligent modeling of temperature propagation induced by therapeutic ultrasound' and the work presented in it are my own. Authors and works consulted during the realization of this work are properly cited and included in the reference listing.

Signed: _____

Date: _____

Abstract

This thesis intended to study the feasibility in applying an innovative approach to estimate the temperature propagation during thermal therapies, in a non invasive way. The standard reference in this field is imposed by the temperature resolution obtained with MRI techniques, 0.5 $^{o}C/cm^{3}$. It was proposed to estimate the temperature evolution using predictive models using b-splines neural networks evolved by the ASMOD algorithm.

Initially the data used to construct the models was characterized to provide the reader the possibility to assess if the data is trustworthy and representative of the physical phenomena intended to model. The modelling environment complexity was gradually increased which resulted in three different models typologies: SPSI, MPSI, MPMI. For each one of the different typologies the relevant features to be taken as input variables were defined along with the network structures associated with the typology.

Ensembles of neural networks were also studied in an attempt to enhance the prediction accuracy of the system. Three methods were assessed: Simple average (SA), where the average of the individual predictions is taken as the final prediction. An evolutionary strategy (ES) was also applied. Again the average of the individual predictions is taken as the final input however each individual network N_i is affected by a weight ω_i. The weight vector ω was evolved by using a evolutionary strategy. A different combination mechanism was proposed in this thesis, neural dynamic ensemble optimization (NDEO), which introduces a second layer formed by a b-spline network takes all the individual predictions as inputs, $o_1 \dots o_N$ where N is the ensemble size and generates an output o_f, which is taken as the final prediction.

A clear division was made between the heating and cooling dynamics involved in a typical thermal therapy. This division resulted in the creation of two distinct models that model the two different dynamics observed. Two experiments were always considered regarding the data used for training, validation and testing: a) Uncorrupted data. This data set is composed of the original data collected in the conditions exposed in this work; b) Corrupted data. After a contamination process, where Gaussian noise was added to the original set, the corrupted data was used to train and validate the models.

Using corrupted data to train and validate the models provides two different analysis perspectives. For one side the robustness of the system was assessed and it helps the designer to ascertain if the structure modelling power is in adequate level for the task. This last assessment is possible by observing the model behaviour in the test set. Ideally the model should only learn the dynamics of the phenomena intended to model and filter all external dynamics derived from the various possible noise sources. On the other side it alleviates the need for acquiring high quality data, which can only be captured using an invasive technique. A reliable temperature estimation method can be used to collect all the data needed to create models of complex environments.

Several models were developed for estimating the temperature curves in a non invasive way. We found that the modeling approach applied was capable of providing highly accurate predictive models. This observation holds in the experiments using Gaussian contaminated data, which evidences the robustness of the approach. A second crucial observation is that the performance figures obtained remain comparable when the modeling environment complexity is increased, suggesting a modelling approach with the desirable scalability.

The performance figures were obtained using relatively simple models, which might be crucial for applications with scarce resources or that require real time responses. It was observed the average model complexity evolved at a slow pace with the modelling environment complexity, which means the system complexity can be managed as the environment approaches ideal conditions.

Combining BSNNs by forming neural network ensembles creates a potential performance enhancement mechanism, if the designing is appropriated. However we noted that a great deal of effort by the designer is needed to create the favorable conditions on which combining individual forecasting entities might pay off.

When compared to the state of art, the BSNN structures over-perform the maximum absolute error obtained using MRI, which is a very impressive result. Obviously the environments on which MRI operates are far more complex than the ones studied in this work. However we observed a modelling approach with very good indicators concerning scalability in response to increases in the complexity of the modeling environment. Together with neural network ensemble methods the systems can be forced to be more accurate and robust. We conclude that the approach followed in this thesis is feasible, and future research is highly recommended.

Resumo

Este tese pretende estudar a possibilidade de aplicar um uma abordagem inovativa para estimar a propagação de temperatura em tecidos durante termoterapias, num paradigma não invasivo. A referência do estado da arte é imposta pela uso de técnicas de ressonância magnética (MRI), onde são obtidas resoluções de temperatura com erros absolutos inferiores a 0.5 $^{o}C/cm^{3}$. Propõe-se estimar a evolução da temperatura através do uso de modelos preditivos, baseados em redes neuronais b-spline, evoluídas pelo algoritmo ASMOD.

Inicialmente os dados utilizados foram caracterizados de forma a que o leitor possa avaliar se os dados em questão são representativos e adequados do fenómeno físico que se pretende modelar. Gradualmente a complexidade do ambiente visado na modelação foi aumentada, resultando em três diferentes tipologias de modelo: SPSI, MPSI e MPMI. Para cada uma das tipologias as variáveis de interesse foram indentificadas bem como as estruturas de rede mais adequadas para o tipologia em questão.

Conjuntos combinados de redes neuronais foram também alvo de estudo numa tentativa de melhorar a eficácia nas predições dos modelos. Três métodos foram alvo de estudo: Média simples (SA), onde trivialmente a média do conjunto é tida como a predição final. Uma estratégia evolutiva (ES) foi também considerada, resultando assim numa média ponderada onde cada predição individual de cada rede neuronal N_i vem afectadad de um peso ω_i. Um terceiro mecanismo foi proposto nesta tese, NEURAL DYNAMICS ENSEMBLE OPTIMIZATION (NDEO), que introduz uma segunda camada activa na arquitectura do sistema. Esta é constituída por uma rede neuronal que recebe como entrada todas as predições individuais y_i, combinando-as de uma forma activa para uma solução final.

A metodologia de modelação preveu uma separação clara entre a fase de aquecimento e arrefecimento, devido ao distanciamento existente entre a correspondente dinâmica de subida e descida. Esta divisão resultou na criação de pares de modelos, referentes às duas distintas fases da terapia. Duas experiências foram sempre consideradas: a) Dados não contaminados. Este conjunto de dados corresponde ao original, não modificado e cujas condições de captura estão expostas neste trabalho. b) Dados contaminados. O conjunto original é contaminado por um processo aditivo Gaussiano. Este conjunto corrupto é usado para treino, validação e teste. O uso de conjuntos de dados contaminado vem afectado de duas motivações. por um lado fornece um claro teste à robustez do

sistema e ajuda o designer a averiguar se a estrutura possui o potencial adequado ao problema. Por outro lado alivia a necessidade de recolha de dados de alta qualidade, que apenas poderão ser recolhidos utilizando procedimentos invasivo.

Vários modelos foram desenvolvidos para estimar as curvas de temperatura de uma forma não invasiva. Observou-se que a metodolgia aplicada foi capaz de construir modelos predictivos de alta exactidão.

Contents

List of Figures

List of Tables

Abbreviations

BS	**B**asis **S**lines
FIR	**F**inite **I**mpulse **R**esponse
IIR	**I**nfinite **I**mpulse **R**esponse
ANN	**A**rtificial **N**eural **N**etowrks
NN	**N**eural **N**etworks
BSNN	**B**asis **S**lines **N**eural **N**etworks
AMN	**A**ssociative **M**emory **N**etowrks
LMS	**L**east **M**ean **S**quare
ASMOD	**A**daptive **S**pline **M**odelling of **O**bservation **D**ata
NN	**N**eural **N**etworks
MSE	**M**ean **S**quare **E**rror
MA	**M**oving **A**verage
TUS	**T**herapeutic **U**ltrasound
ERA	**E**ffective **R**adiation **A**rea
NFL	**N**ear **F**ield **L**length
SPSI	**S**ingle **P**oint **S**ingle **I**ntensity
SPMI	**S**ingle **P**oint **M**ulti **I**ntensity
MPSI	**M**ulti **P**oint **S**ingle **I**ntensity
MPMI	**M**ulti **P**oint **M**ulti **I**ntensity
MRI	**M**agnetic **R**esonance **I**maging
EIT	**E**lectrical **I**mpedance **T**omography
BSU	**B**ack **S**cattered **U**ltrasound
LWN	**L**inear **W**eight **N**orm
TSF	**T**ime **S**eries **F**orecasting
KTB	**K**eep **T**he **B**est

GA	**G**enetic **A**lgorithm
ES	**E**volutionary **S**trategy
NCL	**N**egative **C**orrelation **L**erning

1

Introduction

1.1 Motivation

The field of thermal therapy has been growing tenaciously in the last few decades. *Thermal medicine* can be defined as the manipulation of the body (or tissue) temperature for the treatment of disease. The application of heat to living tissues is being researched intensely for medical applications, particularly for treatment of solid cancerous tumors using image guidance. Nevertheless thermal therapies are extensively applied in physiotherapy, for the treatment of muscle-skeletal problems. Recently the application in oncology has received an increasing attention by the scientists. Oncologic hyperthermia is a technique where the temperature of tumours is raised to values between 43 and 45 ^{o}C. Hyperthermia can be applied alone or in conjunction with traditional methods, such as the radiotherapy and chemotherapy [10].

A main aspect concerning the application of thermal therapies is the monitoring of temperature in the time and space sense, during treatment. A quantitative assessment of temperature is of extreme relevance for both patient security, and for the efficacy of the therapy. Initial approaches make use of invasive measurements, [11] and [12], where the temperature was measured directly at the temperature site. Although it is a precise temperature measurement (the quality of the measurement is highly dependent on the sensor accuracy), this modality suffers from serious problems. Only a limited number of sensors can be placed because there is tissue damage at each sensor placement and in even certain regions it is impossible to place sensors. An insufficient number of sensors can result in a poor spatial resolution.

It has also been pointed that only the methods based on magnetic resonance imaging (MRI) reached the desired resolution for hyperthermia untill now. The referred resolution is a maximum absolute error of 0.5 $^{o}C/cm^{3}$ [13]. Naturally, the major drawback of this approach is the cost associated with MRI instrumentation.

This work proposes a non-invasive approach to monitor the evolution of the temperature during a thermal therapy. Biomedical instrumentation to *estimate* the temperature in a non-invasive way can benefit from the use of intelligent models to estimate the temperature, by dramatically reducing the costs associated with the practice. This works studies the feasibility of creating *predictive models* that can estimate, and hence monitor, the tissue's temperature evolution during a thermal therapy.

1.2 Proposed goals

The main goal of this thesis resides on the creation of predictive models based on B-splines neural networks, which allows to predict the necessary therapy time and the intensity of the ultrasound (most common heating source), according with the characteristics of the region intended to heat. Therefore it is needed to:

- Identifying temperature-dependent signal and ultrasonic system features and determine the most relevant ones to the system, i.e. which input information should the system have that allows it to accurately estimate the temperature. Increasing

the number of input variables would result in more complex models, therefore a variable should just be introduced as an input if the increase in the prediction performance is justifiable.

- Create predictive b-splines neural network models based on the identified features. These models should be able to effectively estimate the temperature curve evolution on the region of interest, i.e. time and space. The benchmark reference is always the MRI standard of 0.5 $^{o}C/cm^{3}$, which provides a comparison of the results obtained with the current *state of art*.

- Enhance the system prediction accuracy by applying *neural network ensembles* methods. Biomedical applications are naturally subject to the most strict constrains regarding the accuracy of the systems used in the practice of medicine. Therefore any performance enhancement mechanism that can be applied should be studied.

1.3 Thesis outline

This chapter describes the motivation of the work based on main background readings, the proposed goals and summary of the main contributions of this thesis.

Chapter 2 covers all background theory that supports the work developed in this thesis. It starts by defining and representing the problem of predicting temperature propagation. Following this section b-splines are introduced and the structuring of these function to be used as neural networks is also given. We justify why this problem can be solved using such structures. The chapter ends exposing the theory behind neural network ensembles.

Chapter 3 exposes the experimental set-up used in the data acquisition process, which is also characterized. Furthermore this chapter provide a measure of the validity of the data used to construct the models, and it can be used to discuss about how close the environments considered resembles the real, ideal human tissue characteristics.

The modelling methodology and the detailed structure of the models is presented in Chapter (4). This chapter exposes the different temperature predictive models applied in this work. An approach of gradually increasing model complexity was taken while modeling the dynamics of the process. We start by considering models for single-point and single-intensity estimation. Then gradually the complexity of the models is increased towards multi-point and multi-intensity estimation. The motivation behind this methodology is due to the interest we have in study and develop insight concerning the feasibility of using BSNN to predict the temperature propagation, in the environments characterized in Chapter(3).

Chapter (5) exposes all of the results obtained Chapter (6) concludes this thesis by providing a global assessment of the predictive models performance. Neural networks ensembles methods are also evaluated in this chapter. Personal thoughts from the author about artificial intelligent field are also presented and the chapter is concluded by pointing out some future research guidance.

1.4 Publications

Currently two publications based on this thesis are being developed, which we leave here for future reference:

- Ferreira, R, Ruano, M.G, Ruano, A.E, Intelligent non-invasive modelling of ultrasound-induced temperature in tissue phantoms. *Annals of Biomedical Engineering*, Springer, http://www.springer.com/biomed/journal/10439
- Ferreira, R, Ruano, M.G, Ruano, A.E, b-splines neural networks non-invasive modelling of ultrasound-induced temperature in tissues. *4th IFAC International Conference on Intelligent Control and Automation Sciences (ICONS 2016)*. http://icons2016.univ-reims.fr/

2

Background theory

2.1 Introduction

The application intended to develop under the light of this work can be abstracted and crafted in its general form. Going up one abstraction layer, we see from an engineering perspective, a well known problem, namely a *time series predicting* problem, characterized in Section(2.2). Section (2.3) deals with the representation of curves, whose study is fundamental to understand basis splines, introduced in Section (2.3.1). In Section (**??**) the problem is observed from a wider context, where the predictive control architecture is introduced. Neural networks (NN) are introduced in Section (2.4.2), with special attention to associative memory networks (AMN), Section (2.4.2.1). These structures have a specific organization, on which b-spline neural networks (BSNN) are included, Section (2.4.2.2). The ASMOD algorithm is studied in Section (2.4.2.3). The performance criteria applied in this work to assess and compare the predictive models are exposed in

Section (2.5). The optimization of the global predictive system was forced by the use of neural network ensembles, whose theory is covered in Section (2.7) and (2.8).

2.2 Prediction

Generally, the process of prediction can be seen as the forecasting side of information processing, which can be seen as a *filtering* process. The term *filter* refers to a device, or an algorithm, that can be used to extract information about a prescribed quantity of interest, from a set of noisy data. Concerning prediction, the aim is to derive information about how the quantity of interest will be like at some time $n + N$, with $N > 0$, using data (information) measured up until time n. Observe that the process of prediction is based on experience or knowledge of the process that is required to predict. Essentially a forecast can be obtained by:

1. purely judgmental approaches.
2. causal or explanatory (regression) methods.
3. extrapolative (time series) methods.
4. any combination of the above

We are interested in *time series* methods, since the available source of information are samples from a process behavior acquired over time.

Predictive modeling is a wide area of study, therefore it is of our interest to develop insight about process to predict, without loss of generality regarding time series. Notice that the propagation of the temperature on a tissue inherently depends on time, hence it can be thought as a *time series*, thus we are dealing with a *time series forecasting (TSF)* problem, whose goal consists in learning patterns from historical data in order to predict the behavior of the system, and *not* how it works.

A time series is defined as a sequence of vectors (or scalars) which depend on time t.

$$[x(t_0), x(t_1), \ldots, x(t_{i-1}), x(t_i), x(t_{i+1}), \ldots] \tag{2.1}$$

It is pretended to predict the output $x(t)$, of a given process P. The phenomena behind the process may be of *discrete* or *continuous* nature. If t is real valued, we are in the presence of a continuous time series, therefore the process needs to be *sampled* accordingly.

With the time series available, we intend to *estimate* the output of the process P, at some point in the future:

$$\hat{x}[t+h] = f(x[t], x[t-1], \ldots) \tag{2.2}$$

We define h as the *prediction horizon*. For $h = 1$, the prediction is called *one step ahead prediction*. If $h = n$, with $n \geq 2$, the prediction is called n *step ahead prediction*.

An important observation is that the problem of prediction can be thought as a function approximation problem [14]. A general predictive system is illustrated in Figure (2.1).

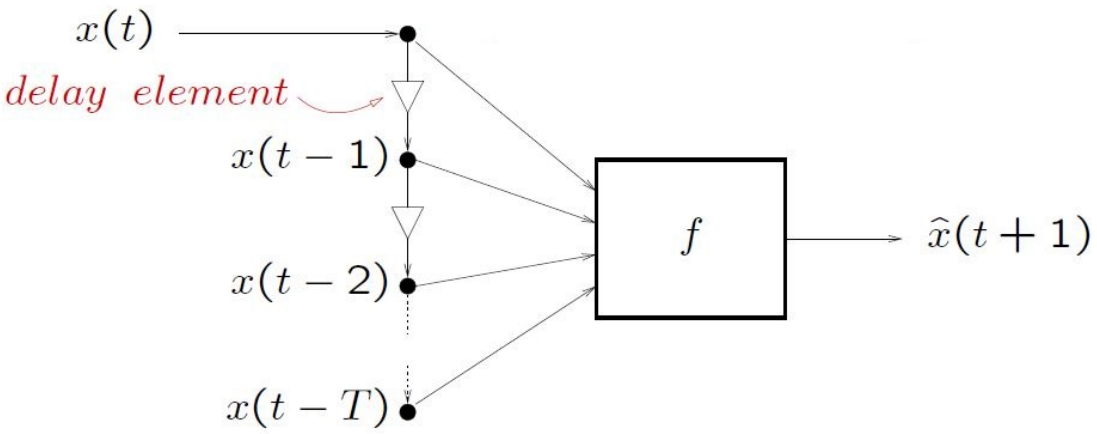

FIGURE 2.1: One step ahead estimation, based on T past values of the process. Adapted from [1].

With respect to the above figure, observe that T delay operations (z^{-1}) are needed, which translates in the need of T storage locations.

Furthermore, the nature of the process must be defined. The process P may be governed by linear dynamics. If so, the problem is in the domain of *Digital Signal Processing*. The operations carried out on the time series are then be implemented by filters, where two basic architectures are possible: the well known **IIR** and **FIR**. However, if the process is governed by non linear dynamics, the problem is a study case of others fields. One of those fields, and the support of this work, consists on Artificial Neural Networks (ANN), structures that provide powerful tools to predict the output of a non linear process. ANNs are introduced in Section(2.4.2).

As noticed on [14], a forecasting problem can be formulated as a function approximation problem. First of all, the function has to be represented. This can be done in wide variety of formulations. Thus, the mathematical tools for representation of the function must be chosen in accordance with the problem specifications. The definition of this functions is done in the following section.

2.3 Representations of curves

The representation of a curve is a subject of the numerical analysis field, with an extensive use in various applications. B-Splines (BS) allow the representation of polynomial parametric curves and have been used in: curve (surface) fitting [15], geometric modeling [16], identification of non linear [17] systems, control applications [18] and naturally, extensively used in computer graphics applications and CAD systems. Another possible representation of curves can be achieved using Bezier curves [19]. However, B-splines are equipped with more attractive properties as we will see. A *parametric representation* consists in representing a curve as a function of one (or more) parameter(s):

$$y(t) : \mathbb{R} \to \mathbb{R}^n, \quad n = 1, 2, 3, \ldots$$

It would be convenient that the function $y(t)$ is as simple as possible, otherwise the evaluation of such function can be computationally expensive. Ideally we are interested in a class of functions which are simple as possible, yet diverse enough to represent a wide variety of curves. To a large extent, polynomial functions satisfy this requirements. A general polynomial function is represented by:

$$y(t) = \sum_{i=0}^{n} a_i t^i \tag{2.3}$$

where n is the degree of the polynomial and a_i the coefficients.

The diversity of curves that one can obtain using polynomials functions is highly dependent on the maximum allowed degree. Naturally, the higher the degree, the greater is the flexibility regarding the shape of the curve.

An inflection point is defined as the point on a curve, in which its curvature (second order derivative), changes its signal. A polynomial $P(x)$ of degree n exhibits, at most, $n-1$ inflection points. However this high flexibility comes with a cost. The first, and most obvious, relies on the computational complexity, which scales as the degree increases. However, it is also important to observe that the higher degree of a curve, the less controllable it is, in the sense that small changes in coefficients are likely to result in large changes in the shape of the curve, which is a non desirable effect. A small change in a coefficient should, ideally, have its consequences locally constrained, since a local control of the curve is desired in order to have a robust system. To summarize, some commonly desirable properties of curves are:

- C^2 **continuity**: The curve should be C^2 continuous at all points. Notice that a function $f(x)$ is said to be of class C^k if the first k derivatives of $f(x)$ exist and are continuous. This can be seen as a smoothing condition of the curve.
- **Interpolation**: Should interpolate all of the control points.
- **Local control**: The modification of a particular control point should modify the curve only locally.

2.3.1 B-Splines

A possible solution to represent curves that meet the previous requirements is to use b(asis)-splines (BSs). BSs were first introduced by Schoenberg in [20]. Originally B-spline basis functions were calculated using a divided difference formula, introduced by

DeBoor in 1978. However the numeric stability of this process sufferd from some restrictions. Later an important contribution was given by Cox [21], where the author derived an efficient recurrent relationship to evaluate basis functions, which is numerically stable.

To appreciate the usefulness of BSs, notice that any spline function of order k, defined by a set of control points, can be expressed by a linear combination of BSs of the form:

$$S_{k,t}(x) = \sum_i \alpha_i B_{i,k}(x) \tag{2.4}$$

Where k represents the order of the spline function, and α_i is the associated set of control points. B_{i_k} defines the polynomial pieces, that can be derived by the recursion algorithm presented on [21]. Note that a linear combination of BSs allows for a flexible construction of curves with a high number of inflection points, showing a smooth and robust behavior to changes of control points. This is done by piecing together several polynomials, as illustrated in Figure (2.2).

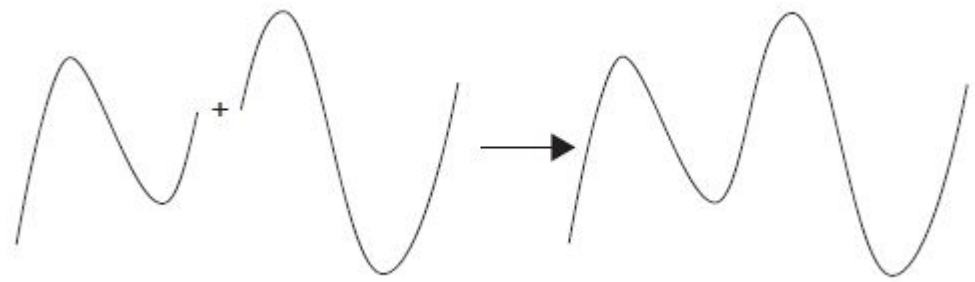

FIGURE 2.2: Piecing polynomials. [2]

However note that the pieces should join continuously at the break point in compliance with the desirable conditions of a curve.

In order to define a BS we need to start with a **knot sequence**. This sequence should be a **non decreasing** sequence $t := (t_i)$:

$$t_i \leq t_{i+1}, \quad all \quad i \tag{2.5}$$

Then the BSs of order 1 for this knot sequence, are the characteristic functions of this sequence:

$$B_{i1}(t) = \begin{cases} 1 & t_i \leq t < t_{i+1} \\ 0 & otherwise \end{cases} \tag{2.6}$$

The only constraint is that these B-splines should form a **partition of unity**:

$$\sum_i B_{i1}(t) = 1, \quad all \quad t \tag{2.7}$$

Basically, this property assures the invariance of the BS shape under translation and rotational operations. This property is very attractive for geometric applications, which in mathematics is known as *affine invariance*.

From these first-order BSs, one can obtain high order B-splines by recurrence [21]:

$$B_{i,k} = \omega_{i,k} B_{i,k-1} + (1 - \omega_{i+1,k}) B_{i+1,k-1} \tag{2.8}$$

with:

$$\omega_{ik}(t) = \begin{cases} \dfrac{t - t_i}{t_{i+k-1} - t_i} & if \quad t_i \neq t_{i+k-1} \\ 0 & otherwise \end{cases} \tag{2.9}$$

Thus, a second order BS is given by:

$$B_{i,2} = \omega_{i,2} B_{i,1} + (1 - \omega_{i+1,2}) B_{i+1,1} \tag{2.10}$$

Observing the nature of this recurrent algorithm, one can conclude that a BS of order k consists of polynomial pieces $b_{j,k}$ of exact order $k - 1$. Note that the second order BS previously defined, consists of two linear pieces that join continuously to form a piecewise linear function that vanishes outside the interval $[t_i, \ldots, t_{i+2})$. Thus, a BS of order k has **support** along the interval $[t_i, \ldots, t_{i+k})$. **Support** refers to the region of the input space where the function assumes non zero values. The number of *internal knots* must be greater or equal to $k - 1$. For any interval $[t_i, \ldots, t_{i+k})$ at most k of $B_{i,k}$

are non zero. This property is illustrated in Figure (2.3), where we can observe three non zero $B_{i,3}$ over the interval $[t_j, \ldots, t_{j+k})$.

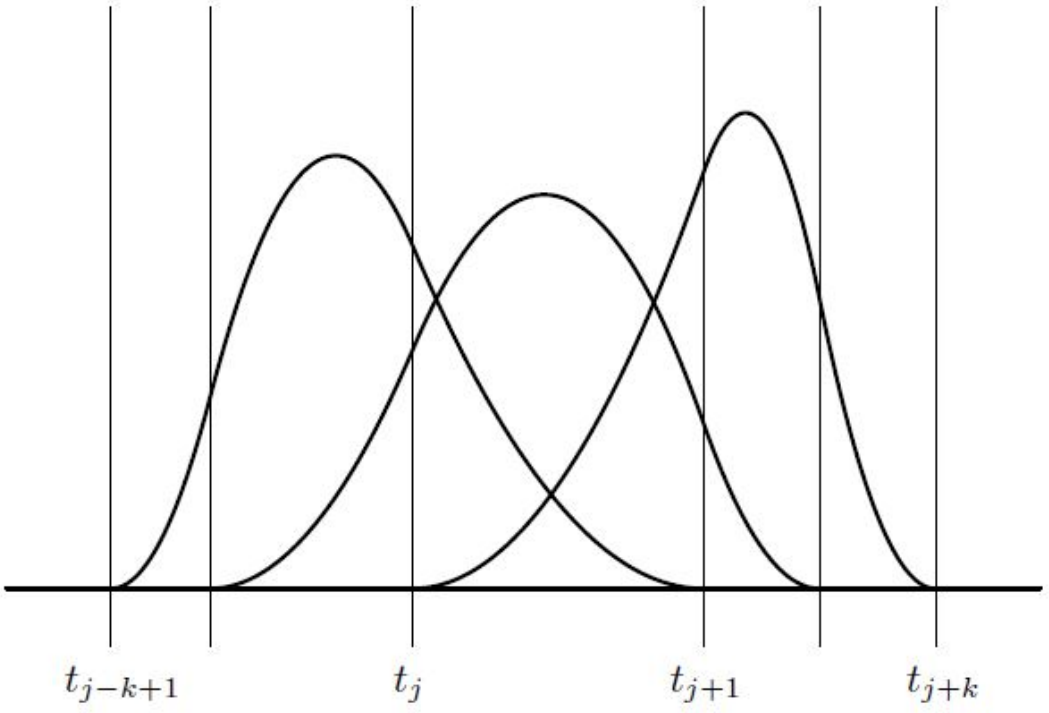

FIGURE 2.3: Three BS or order 3. Notice the three non zero $B_{i,3}$ over the interval $[t_j, \ldots, t_{j+k})$. Adapted from [2]

If in the **knot sequence**, exists a knot with **multiplicity** higher than one, the continuous condition might be violated in that knot. As an example, consider a BS of order 2. If in the knot sequence exists a knot with double multiplicity, e.g., $t_i = t_{i+1}$, but still $t_{i+1} < ti + 2$, then B_{i2} consists of just one piece and fails to be continuous at the double knot. This is illustrated in Figure (2.4)b), in contrast with a BS of order 2 in which the knot sequence is **monotonically increasing**, i.e., a sequence just with simple knots, Figure (2.4)a). It is also important to notice that we desire the influence of a control point to be maximum at regions of the curve close to that point, and it should ideally decrease as we move away along the curve, eventually disappearing. Increasing the multiplicity of a knot reduces the continuity of the curve at that knot, i.e., the curve looses smoothness. Generally, a curve its $(k - p - 1)$ times continuously differentiable at a knot with multiplicity p, given that $p < k$, being k the order of the polynomial. Thus

at that knot, the curve belongs to the class of C^{k-p-1} continuity. We can then conclude that a knot with multiplicity $p = k$, indicates C^{-1} continuity, i.e. a discontinuous curve, situation that is not acceptable. The choice of the **knot sequence** is of major importance in the design phase.

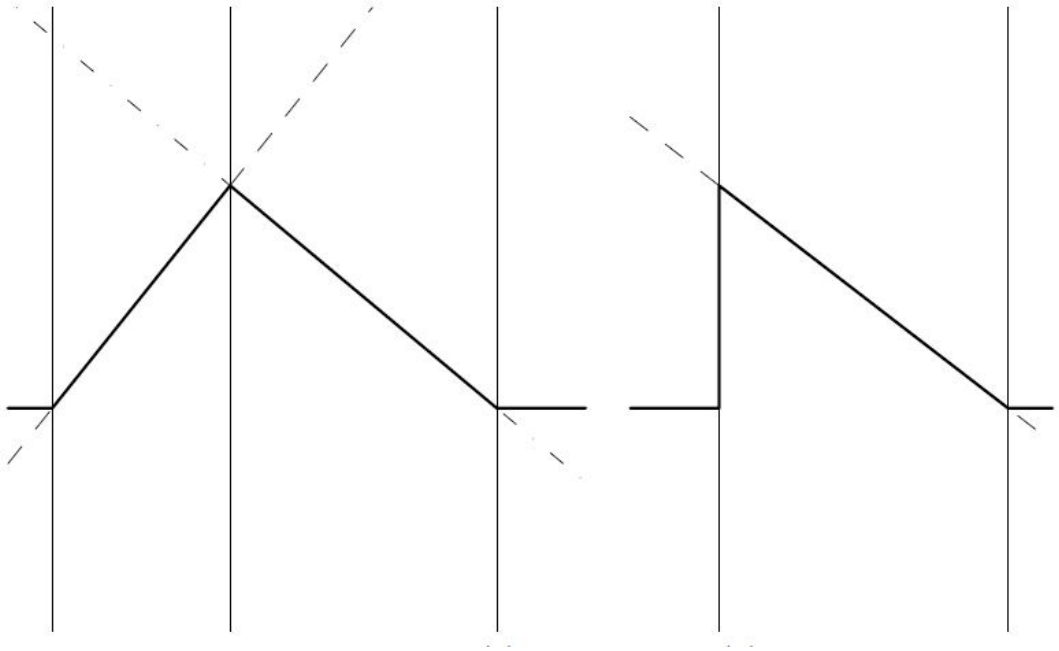

FIGURE 2.4: BS of order 2 with (a) simple knots, (b) a double knot. Adapted from [2]

For instance consider a B-spline of order 3. A simple knot would mean two smoothness conditions, i.e., continuity of function and first derivative, while a double knot would only leave one smoothness condition, i.e. just function continuity, and a triple knot would leave no smoothness condition, i.e. even the function would be discontinuous. This leads us to the following rule for calculating the order k of the basis function:

$$k = knot\ multiplicity + condition\ multiplicity \tag{2.11}$$

The following notation can be used to empathize the function dependency on its knot sequence t:

$$B(.|t_i, \ldots, t_{i+k}) := B_{i,k} \tag{2.12}$$

Any smooth piecewise polynomial function is called a *spline*. If the *spline* is represented on its *B-form*, then the *spline* is described as a linear combination of B-splines:

$$\sum_{j=1}^{n} B_{j,k} a_j \tag{2.13}$$

Therefore a *univariate* (function of just one variable), spline f, is specified by its non decreasing sequence knot sequence t and by its coefficient sequence a_k, which are called the *control points* for the curve. The length of the knot sequence t should respect the following rule:

$$length(t) = n + k \tag{2.14}$$

Where k is the order and n is the number of B-splines that form the *spline* function. Furthermore, the order of a BS can be given by:

$$k = length(t) - n_a \tag{2.15}$$

Where n_a represents the number of control points (coefficients).

Thus the free parameters are the **control points** and the **order** of the BS, which provides a B-splines based system a wide flexibility in the design. In contrast with Bezier curves, BS offers a much more localized control, a higher degree of freedom. In Bezier curves a change applied in one point creates a chain of global changes in the whole curve. Also note that the latter offers a less number of free parameters [22], which translates in less flexibility in the system design. Furthermore, the degree of the BS is logically independent of the number of control points. This means the we can use lower degree curves and still maintain a large number of control points.

A more intuitive example can be given in order to better visualize the process behind the construction of a B-spline. Consider a single B-spline of order $k = 4$ (cubic function), with knot sequence $t = [0\ 1.5\ 2.3\ 4\ 5]$, a sequence with a length of 5, since we are considering a single B-spline $n = 1$ of order $k = 4$ ($length(t) = n + k$). The graphical representation of the building blocks that constitute the B-spline is shown in Figure (2.5). This figure was generated for illustrative purposes.

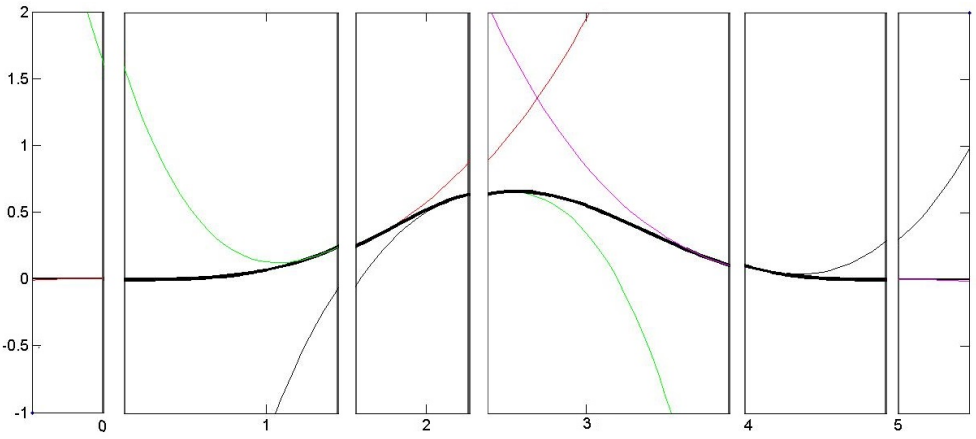

FIGURE 2.5: B-spline of order 4 with knot sequence $t = [0\ 1.5\ 2.3\ 4\ 5]$. Each interval, with the respective color, represents one piece of the basis function. Figure generated with MATLAB, using the *Curve Fitting Toolbox* [3]

.

The knots are represented by the gray vertical lines. Separation between blocks has been made to empathize the piecing process. Four polynomials of order $k-1$ (3) are used in the construction of the basis function, represented by the green, red, violet and black curves. For each interval, formed by two adjacent elements in the knot sequence, a piece (portion) of one of the four $k-1$ polynomials is used to form the basis function.

In Figure (2.6) a 6th order BS with the six polynomials of order 5 is presented, which selected pieces (intervals) make up the B-spline.

This figure clearly illustrates the nature of the process behind the constructing of a BS. For a B-spline of order k, specific intervals of order $k-1$ polynomials are selected. All of these pieces are joined continuously to form the BS.

Once the basis functions are defined, linearly combining them, *weighted* by a control point vector a_k, gives rise to a *spline* in its *B-form*. The short-hand:

$$f \in S_{k,t} \tag{2.16}$$

indicates that f is a spline or order k with knot sequence t, i.e, a linear combination of B-splines of order k for the knot sequence t.

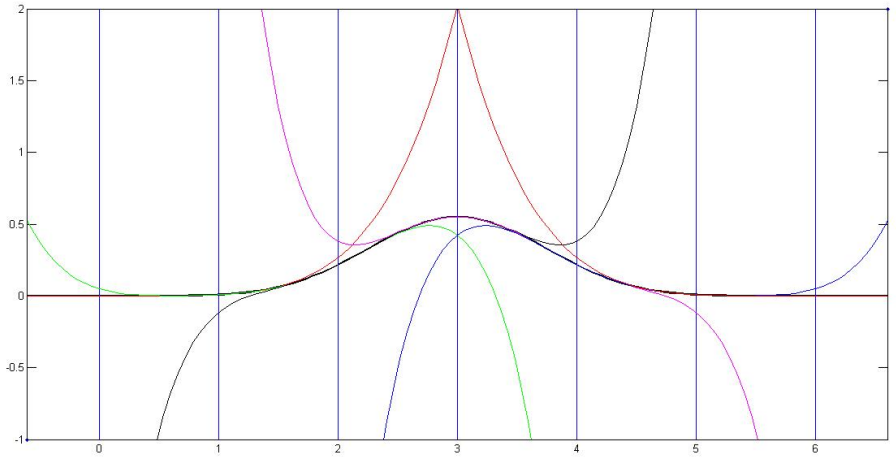

FIGURE 2.6: A 6th order B-spline and the six 5th order polynomials whose selected pieces make up the B-spline. Knot sequence $t = [0\ 1\ 2\ 3\ 4\ 5\ 6]$. Each piece of the basis function is represented in a different color. Figure generated with MATLAB, using the *Curve Fitting Toolbox* [3]

.

The process of constructing a spline by combining basis functions is illustrated in Figure (2.7). This example was constructed to develop insight concerning the local nature of these B-splines.

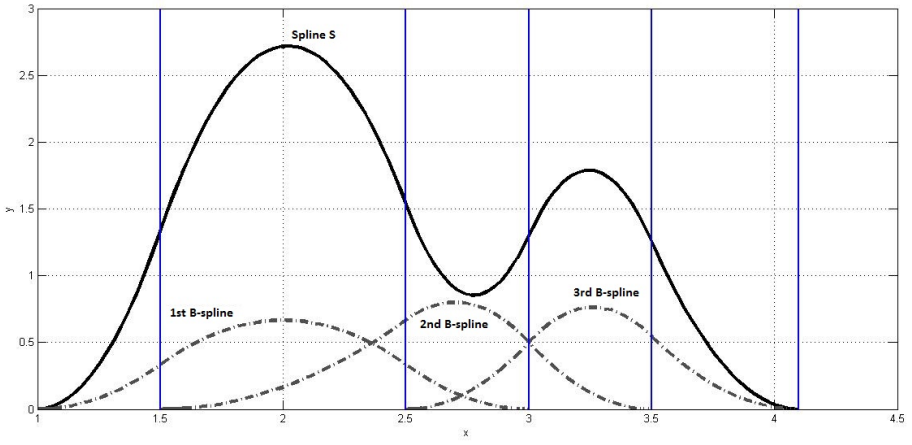

FIGURE 2.7: Spline function of order 3, constructed by linearly combining three B-splines of order 3. The blue lines indicate the position of knot, the gray dashed lines represent the B-splines, and the solid black line shows the resulting spline function. Control points employed $a = [4\ 0.3\ 2.3]$. Figure generated with MATLAB, using the *Curve Fitting Toolbox* [3]

.

Constructing of the spline function S was done by defining a knot sequence $t = [1\ 1.5\ 2.5\ 3\ 3.5\ 4.1]$, with control points $a = [4\ 0.3\ 2.3]$. With three control points and $k = length(t) - n_a = 3$, three B-splines of order 3 form the building blocks of the spline function. The output of each basis function is affected by a weighting term a_j. Thus, by linear combining the basis functions, the output of S is directly given by:

$$S = \sum_{j=1}^{n} B_{j_k} a_j = B_{1,3}(x|t_1, t_2, t_3, t_4)a_1 + B_{2,3}(x|t_2, t_3, t_4, t_5)a_2 + B_{3,3}(x|t_3, t_4, t_5, t_6)a_3 \tag{2.17}$$

To prove that B-splines are robust to changes in control points, lets make a change in the control vector $a : a_3 = 2.3 \rightarrow 1$, resulting in the new control vector $a = [4\ 0.3\ 1]$. The resulting spline function from this change is illustrated in Figure (2.8).

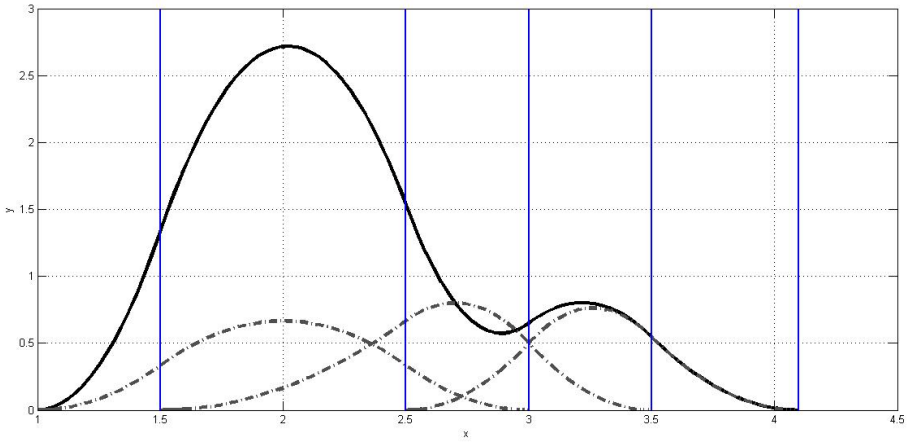

FIGURE 2.8: Spline function of order 3 resulting from changing the control vector. The blue lines indicate the position of knot, the gray dashed lines represent the B-splines, and the solid black line shows the resulting spline function. Figure generated with MATLAB, using the *Curve Fitting Toolbox* [3]

.

Contrasting Figure (2.7) with Figure (2.8), it is obvious that the alteration performed on the control vector, had just a local consequence, concerning the third basis function. The changes were not propagated throughout all the function. This provides the designer local control over the entire function range.

Lets evaluate computational efficiency of the recursion method for evaluating B-splines. The number of operations to evaluate a set of k non-zero BSs or order k can be calculated.

We can denote this functions as N_k^j, N_k^{j-1}, ..., N_k^{j+k-1}. This process is illustrated in Figure (2.9).

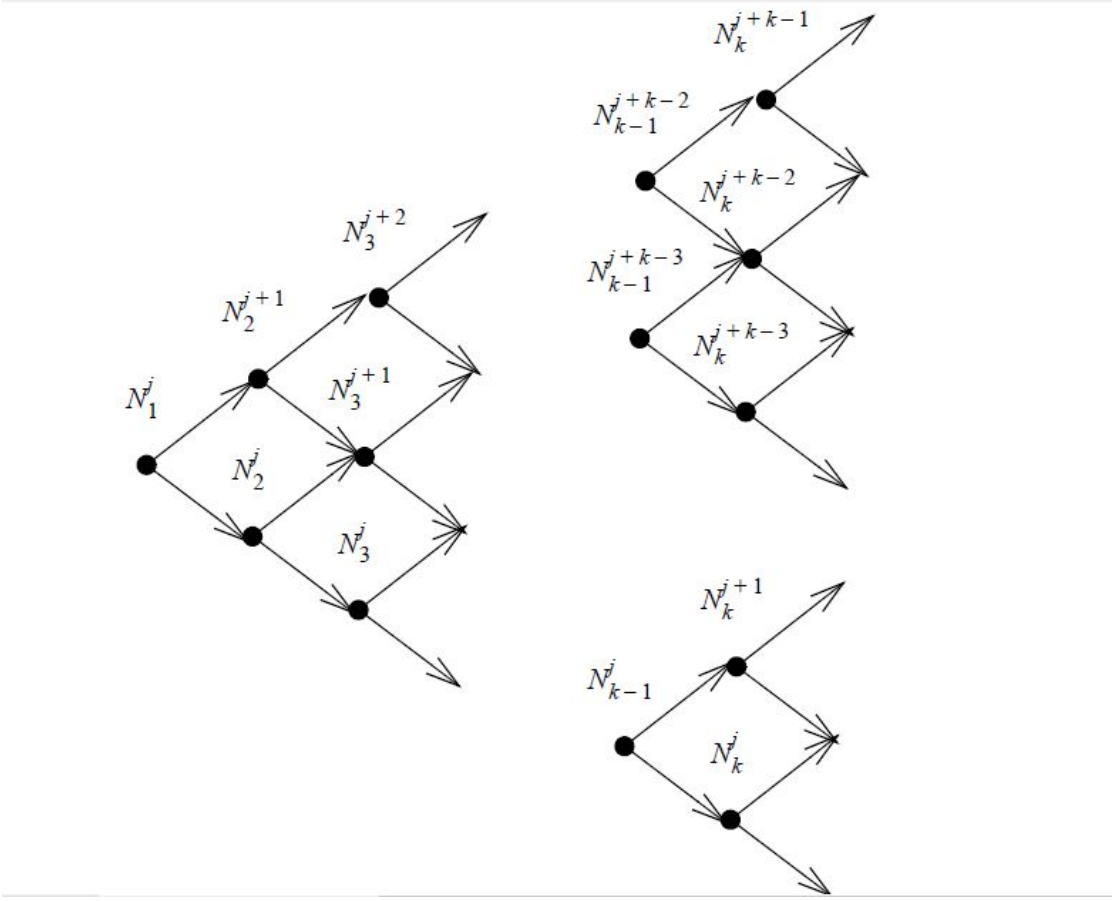

FIGURE 2.9: Triangular array used for evaluating k BSs of order k. [4]

In [6] the number of arithmetic operations for evaluating a set of k B-splines of order k was proven to be:

- $k(k-1)$ floating point multiplications.
- $\frac{k(k-1)}{2}$ floating point divisions.

Which can be done with a reasonable computational cost, since the order of the BSs is usually kept low.

Another possible way of defining B-splines is using **convolution operation**, an alternative approach to characterize B-splines functions that is derived in Appendix(A).

All the previous analysis was derived assuming univariate B-splines basis functions. A wide variety of applications of interest requires a bigger dimensional space. A generalization of the B-spline curve design method to surfaces requires an extension of the B-splines basis to higher dimensions. Note that the generalization of the univariate BSs should admit arbitrary knot configurations while preserving desirable features.

Multivariate basis functions are constructed by taking the *tensor product* of the univariate basis functions. This construction is made with the following relation:

$$\mathbb{R}^n \to \mathbb{R} : (x_1, x_2, ..., x_n) \to f(x_1)g(x_2)...h(x_n) \tag{2.18}$$

The tensor product idea is very simple. If f is a function of x ($f(x)$), and g a function of y ($g(y)$), then their tensor product:

$$p(x, y) := f(x)g(y) \tag{2.19}$$

is a function of x and y, a bivariate function.

Therefore, a multivariate basis function of dimension n is formed from the product of n univariate basis functions, one for each input axis. An example of a multivariate basis function is shown in Figure (2.10).

For example, a trivariate spline in its *B-form*, is given by:

$$\sum_{u=1}^{U}\sum_{v=1}^{V}\sum_{w=1}^{W} = B_{u,k}(x|s_i, \ldots, s_{i+h})B_{v,p}(y|j_i, \ldots, j_{i+n})B_{w,m}(Z|g_i, \ldots, g_{i+r})a_{u,v,w} \tag{2.20}$$

This spline is of order k in x, of order p in y, and of order m in z. $a_{u,v,w}$ defines the control points for the spline, and $g = [g, \ldots, g_{i+r}]$, $j = [j, \ldots, j_{i+r}]$, and $s = [s, \ldots, s_{i+r}]$, defines the knot sequence of each univariate basis function.

However notice that this increase in dimensionality comes with a cost. The complexity of the system scales exponentially with the input dimension n. From Figure (2.10), observe that at each point 3^2 basis functions are active, in contrast to 3^1 active functions

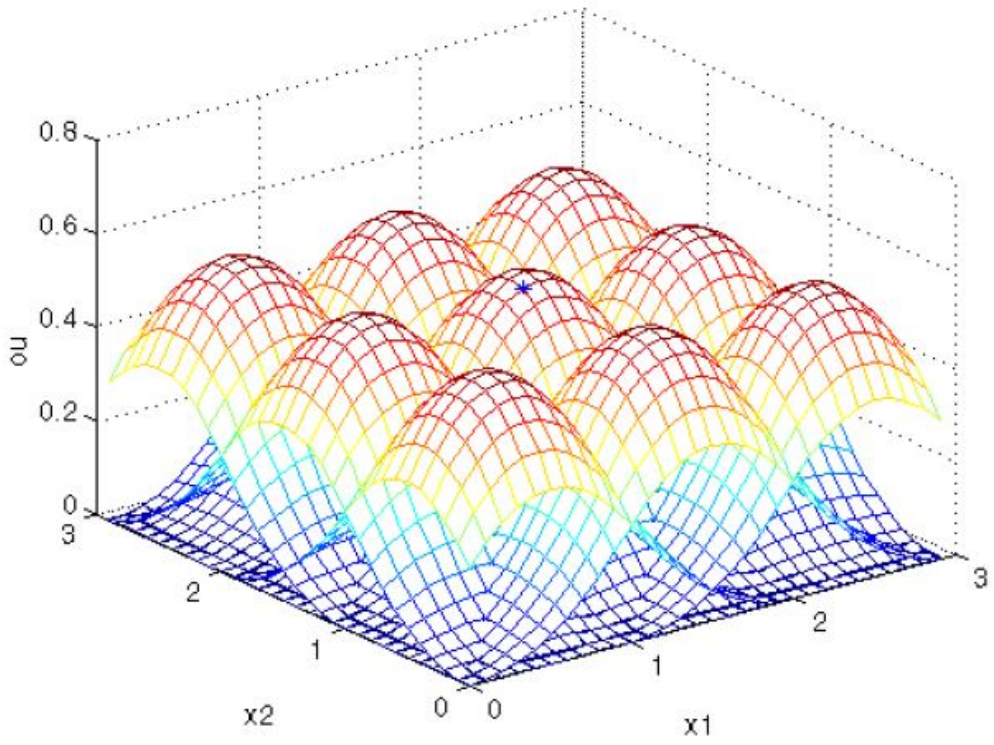

FIGURE 2.10: Two-dimensional multivariable basis functions formed with order 3 univariate basis functions. Adapted from [4].

that would be active for an unitary input dimension. This is known as the *curse of dimensionality*. Some works have been developed to break or at least to minimize this curse [23].

The *tensor product construct* just defines one possible way to generalize basis functions to higher input spaces. This approach requires that the data comes in tensor product form, i.e., on a (hyper)rectangular grid. Naturally this construction rises some concerns. Is the proper multivariate version of an interval (for the univariate case), a (hyper) rectangle? Another constructions have been made concerning multivariate basis functions, which are not supported by the tensor product. C. de Boor proposed an alternative in [24].

2.3.2 Approximating functions with B-splines

As mentioned on the previous section, a forecasting problem can be formulated as a function approximation problem. Therefore, it is of our interest to study the B-spline capabilities to approximate a function. We start by providing an important theorem, the *universal approximation theorem* [25]. This can be used to support the prove that B-splines can approximate any continuous functions with an arbitrary precision on a compact set. The universal approximation theorem, for a nonlinear input-output mapping, may be stated as:

Let $\vartheta(.)$ be a nonconstant, bounded, and monotone-increasing continuous function. Let I_{m_0} denote the m_0 dimensional unit hypercube $[0,1]^{m_0}$. The space of continuous functions on I_{m_0} is denoted by $C(I_{m_0})$. Then, given any function $f \in C(I_{m_0})$ and $\epsilon > 0$, there exist an integer M and sets of real constants α_i, b_i and ω_{ij}, where $i = 1, \ldots, m_1$ and $j = 1, \ldots, m_0$, such that we define:

$$F(x_1, \ldots, x_{m_0}) = \sum_{i=1}^{m_1} \alpha_i \vartheta(\sum_{=1}^{m_0} \omega_{ij} x_j + b_i) \tag{2.21}$$

as an approximate realization of the function $f(.)$, that is:

$$|F(x_1, \ldots, x_{m_0}) - f(x_1, \ldots, x_{m_0})| < \epsilon \tag{2.22}$$

for all $x_1, x_2, \ldots, x_{m_0}$ that lie in the input space.

As previously referred, spline functions can be represented as piece wise polynomial functions. The classic *Weierstrass approximation theorem* [6], states that any continuous function over a closed interval on the real axis can be expressed in that interval as an absolutely and uniformly convergent series of polynomials. This assures us that polynomial approximation can get arbitrarily close to any continuous function as the polynomial order is increased. The universal approximation theorem may be viewed as a natural extension of the Weierstrass theorem. As so, spline functions are universal approximators.

Polynomials are then the approximating functions of choice when a smooth function is to be approximated locally. A *smooth* function refers to a function that changes gradually, without discontinuities. Temperature propagation in tissues when applying relatively low beam intensities, governed by the laws of thermodynamics, can be assumed to be a *smooth* process, with no abrupt changes. However if the function is to be approximated on a large interval, the degree of the approximating polynomial may have to be chosen unacceptably large. B-splines provide an alternative to subdivide the whole interval $[a..b]$ of approximation into sufficiently small intervals $[x_i \dots x_{i+1}]$ so that, on each such interval, a polynomial p_i of relatively low degree can provide a good approximation of the function. Therefore we conclude that B-splines provide a suitable tool for the prediction of the time series that this work focus on.

2.4 Process control

As stated in the previous chapter, the ultimate goal of the wider project, in which this work is included, is to design and implement an intelligent instrumentation control system which controls the therapy time as well as the ultrasound intensity, in a efficient and secure way. The *process* we intend to control is the temperature propagation in the patient tissue. We can identify four key elements that enable a control application to be considered *intelligent* [5]:

- **Performance function** for evaluating the state of the process.
- **Learning** to construct a predictive model.
- **Exploration** of different control strategies.
- **Remember** previous control actions.

These elements can be translated to a control architecture shown in Figure (2.11).

The learning control elements must generate the best possible control strategy which can track the desired response of the system. The aim of this work is to derive the *plant model*. This model should be a predictive model that estimates the output of the plant. Optimization routines search the space of possible control actions, evaluating the performance of the control of the plant model. Once a satisfactory control action has been

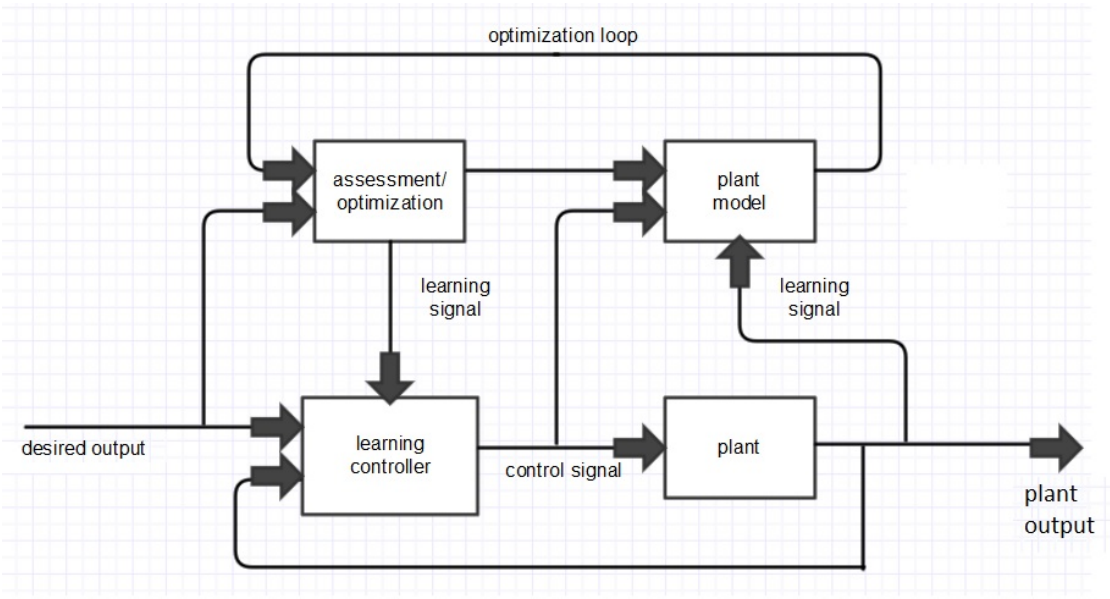

FIGURE 2.11: Predictive control architecture. Adapted from [5]

calculated, it is applied to the plant and used to train the learning control module. In the environment of this project, the *plant* consists in the therapy ultrasound instrument that is being applied to the patient. The *plant output* its given by the real temperature of the tissue, on the spatial points considered.

2.4.1 Modeling the temperature propagation

The process we are dealing with (temperature propagation), can be thought as a time series. In order to predict behavior of this process, its dynamics need to modeled. One possible solution is to derive an *analytic model* of the process. These mathematical models have a closed form solution. A set of equations describing the changes in a system must be found. The solutions to these equations can then be expressed as mathematical analytic functions. However, analytics solutions describing complex processes can often become very complicated. These analytics models can give information about the system's behavior in a very direct way. An attempt to model the temperature propagation in some *phantoms*, simulating human tissue, was made in an early stage of this project [7]. Using the *least square principle*, a squared polynomial was fitted to the data measured in the two distinct phases of the process, i.e, the heating phase and the

cooling phase. The measured temperature and the estimation, given by the square analytic model, are almost overlapped. However, this analytics models are constructed concerning a very specific spatial location, therapeutic intensity and phantom considered. This models don't have the ability of *generalizing*, i.e. interpolate and extrapolate to unseen data. Ideally, we want a model that is capable of *learning* from the data, though a *learning process*, provided that the data represents well the dynamics of the process. Making the model intelligent allows it to learn the process dynamics. If data is available, construction of a *data driven* model is possible. Regarding our particular application suppose that a set of data was collected, considering a set of spatial points (sensors) S as well as a set of therapeutic intensities I ($I \times S$ operating points). This data is used to train a model M. After the model learns from the data, we expect from it to have learned the process dynamics, so it can *generalize* to input conditions that were not directly leaned, i.e, different spatial points, different therapeutic intensities and possibly different phantom conditions. This intelligence, embedded in the model's structure through the learning process, is not present in the analytic models.

Neural Networks (NN) provide the means to build these models, and are used in this project to model the dynamics of the process considered. In terms of prediction, NNs models have a significant advantage over analytic models, though, because they require only history as input, no assumptions are considered. Using data history, the neural-network model automatically develops its own internal model of the process and predicts future behavior.

Neural-network approaches develop models that can be relatively complex. However this models can be refined to obtain the most simple model with the required performance. The model complexity can be automatically adjusted to the complexity of the process, an important advantage of using these networks. In contrast, analytic models can be simple and their complexity is fixed. A work that compares the two models paradigms can be found in [26].

2.4.2 Neural Networks

Neural networks (NN) are a computational metaphor inspired by studies of the brain and nervous system in biological organisms. They are an (very naive) attempt to model the biological leaning mechanism, which consist of highly idealized mathematical models of how we understand the essence of these nervous systems. To achieve good performance, neural networks employ a massive interconnection of simple computing cells referred to as *neurons* or *processing units*. NN are data-driven, self-adaptive, non-parametric, nonlinear methods that do not require specific assumptions about the underlying model. This modeling approach has the ability to learn from experience, which can be very useful for many practical problems since it is often easier to have data than to have good theoretical guesses about the underlying laws governing the systems from which data are generated.

The procedure used to perform the learning process is called a *learning algorithm*, whose function is to modify the synaptic weights of the network in an orderly fashion to attain a desired design objective. A *supervised learning paradigm* is used in this work . This paradigm considers that the learning is achieved by the help of a *teacher*, which owns *knowledge* about the behaviour of the process. This paradigm is illustrated in Figure (2.12).

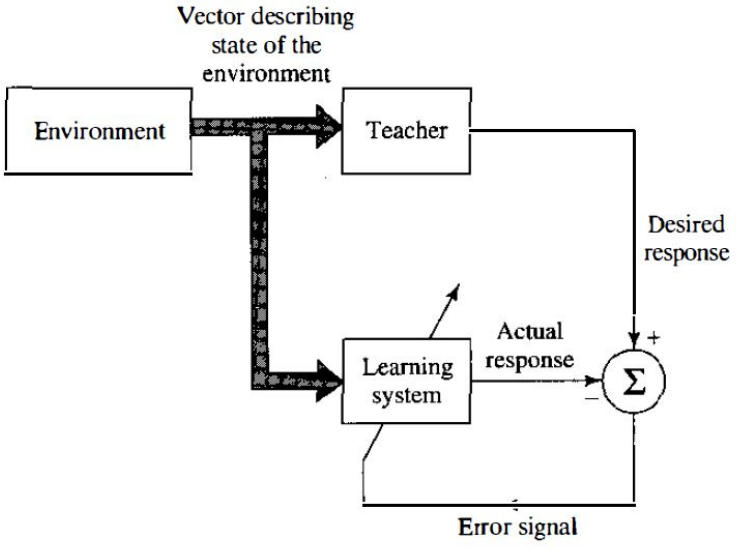

FIGURE 2.12: Block diagram of a learning process with a teacher. Adapted from [6]

The teacher is masked through the available measured data of the process (*environment*), and is used to train the model. The *learning process* allows the network to learn from the environment. Various learning algorithms exist based on [6]. In this work we are interested in *error correction learning* methods. Once having available the desired output of the system the error can be calculated.

The previous discussion over B-splines revealed the approximation capabilities of this functions. Therefore, in this work, a decision was made to consider neural networks based on BSs. An overview about these networks follows.

2.4.2.1 Associative Memory Networks

B-splines neural networks (BSNNs) are members of a class of NNs, called Associative Memory Networks (AMNs). This class of networks have desirable properties which can be used for adaptive non linear modeling [22]. The function approximation capabilities of AMNs are of great interest to modeling and control purposes. Their architecture is fixed and consists of three logical layers, as shown in Figure (2.13).

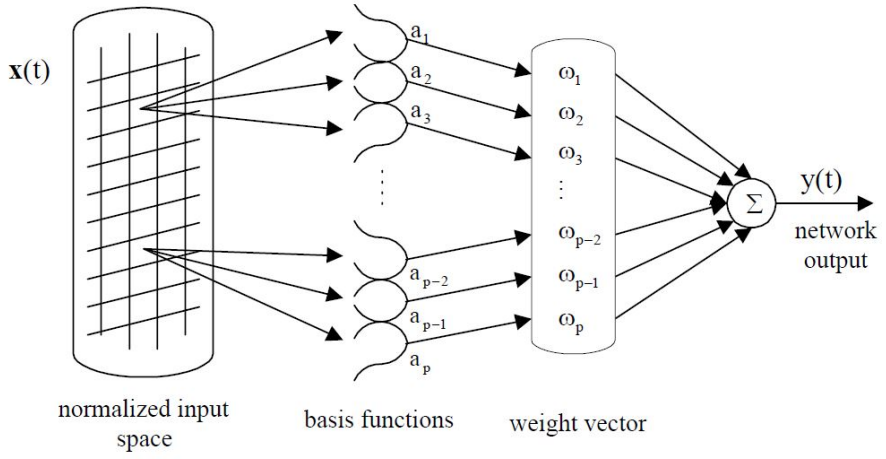

FIGURE 2.13: Associative memory network structure. Adapted from [4].

The first layer *normalizes* the original input space. The normalization process consists on defining a n- dimensional lattice on the input space. The basis functions are defined over this lattice, which consist of a very simple strategy with some desirable characteristics. The construction of the lattice is simple: an axis for every input dimension.

Prior knowledge of the process, with respect to a certain input, can be introduced when structuring the lattice, hence biasing the model. Finally, the procedure to find the region (*cell*) where the input lies in the lattice, can be done in a computational efficient way.

The main potential disadvantage of lattice-based AMN resides on the *curse of dimensionality*, which results in the network's memory requirements being exponentially dependent on the input space dimensions. The *knot sequence* discussed before in Section (2.3.1), defines the partition of the lattice. Thus, one knot sequence is needed for each input dimension.

Going back to Figure (2.13), we observe that the second layer is constituted by the basis functions, which represents the *associative cells*. These functions are defined on the normalized input space. We define the *receptive field* (support) of a basis function to be the domain in the input space for which the basis function's output is non-zero. A basis function has a *bounded support* when its support is smaller than the network's domain, which is generally true. The size, shape and overlap of the basis function determine how the network *generalizes* and also its complexity.

Lastly, the output layer of an AMN is formed from a linear combination of the outputs of the basis functions. It is required for this output to be *continuous*, so the network can predict *smooth* processes. The linear coefficients, w_i, are the adjustable weights of the networks and, because the output is linearly dependent on the weight vector, learning consists in a linear *optimization* problem. The update of the weight vector can be done by a error *gradient descent* method.

Important is to retain that AMNs perform two *mappings*. From the input x_i to the output of the basis function a_i, $x_i \rightarrow a_i$ the network performs a **fixed**, *non linear mapping*, assuming that the network structure is maintained fixed. Posteriorly the network processes an **adaptive** *linear mapping*, from a_i to the network's output y, $a_i \rightarrow y$. Thus, the adaptation (learning) is performed only in the last layer, which has the advantage that the output is linear with respect to the weights w_i. An obviously drawback of NNs is that the interpretation of the knowledge stored in the weights cannot be accomplished

in a direct way, which is something that analytic models provide.

For any input to a B-spline AMN, the number of active basis functions (non-zero), is always a constant, called the *generalization parameter*, which we denote by ρ. In response to an input, ρ basis functions contribute to the output. The ρ basis functions that contribute to the output can be organized into ρ sets, where one and only one basis function in each set is active. The union of the supports in each set forms a complete and non-overlapping n-dimensional *overlay*. For a two dimensional case, with $\rho = 3$, the formed overlays are illustrated in Figure (2.14).

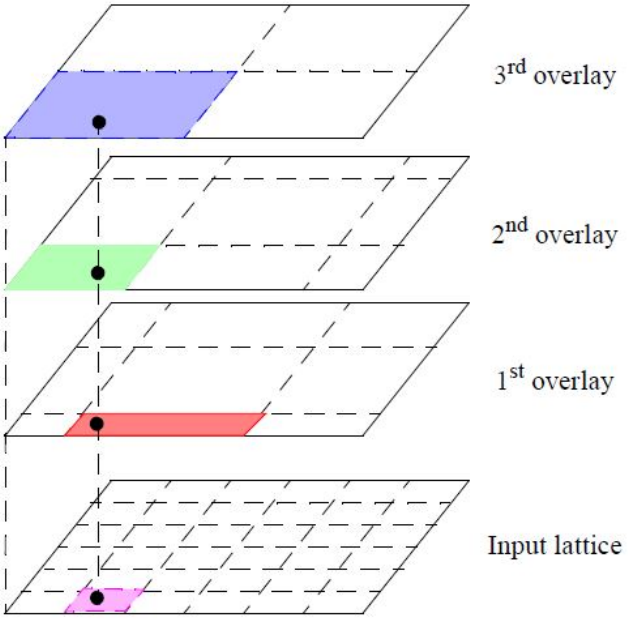

FIGURE 2.14: A two dimensional AMN with $\rho = 5$ and $r_i = 5$ interiors knots for each input. Figure taken from [4].

With r_i being the number of interior knots for each input dimension, for each axis there are $r_i + 1$ intervals, which might be empty if coincident knots exist, and $p = \prod_{i=1}^{n}(r_i + 1)$ n-dimensional cells in the input lattice. The total number of basis functions p is then *exponentially* dependent on the input space dimension n, hence the *curse of dimensionality*.

For a given input, a small number of weights is expected to contribute to the network's output. In this case, ρ weights. Hence only these parameters should be updated when the network is trained. With this in mind, note that the non linear transformation from

the first layer to the second layer, is *topology conserving*, meaning that similar inputs map to similar sets of active basis functions. This provides the network with *local leaning* capabilities, where the knowledge is stored *locally*, which is a very desirable feature. In this way, learning in one area of the input lattice space does not interfere with learning in another distinct region. A discussion about the learning interference for B-splines networks can be found on [27], where the results suggest that the interference slightly increases with the order of the B-splines. Note that this *data driven* models require an uniformly distributed set of data through all the input domain, in order to construct an useful model, i.e., a complete data set in the sense that it contains a sufficient amount of information from all interesting operating conditions and system variables.

Concerning the robustness of an AMN network, when summing all of the basis functions outputs, a constant result is a desirable characteristic for an AMN:

$$\sum_{i=1}^{\rho} a_i(x(t)) = k \tag{2.23}$$

Where $x(t)$ represents the input vector, of which a_i depends on. If a network possesses this property, it is said to form a *partition of unity* or a *constant field strength*. As previously discussed, B-splines endue this property. With this sum being constant, any variation in the network's surface is solely due to the weights in the network. Otherwise the network response dynamics may vary according to the position of the input or the size of the magnitude of the field strength. The variance of the field strength increases with the input dimension, unless the network forms a partition of unity, which enhances the robustness of the network. Another important aspect of this property is observable in the smoothness of the network, which generally is much higher in networks that form a partition of unity. However, if the network does not hold this property, it can be artificially forced, as derived on [28].

2.4.2.2 B-splines neural networks

As demonstrated in Section (2.3.1), B-splines functions are simply piecewise polynomials mappings, formed from linear combinations of basis functions, and the multivariate

basis functions are defined in a lattice. Thus, a B-spline can be considered as an element belonging to the class of lattice AMNs. The second mapping performed by these networks is *adaptive*, hence the weights in the last layer must incur into an adaptation process, the training phase, which can be done in an *online* or *offline* way. If the weight adjustment is only performed after a complete set of patterns has been presented to the network, the process is denoted as *offline learning*. On the other hand, if upon the presentation of each pattern there is always a weight update, we can say that we are in the presence of an *online learning* or *instantaneous learning*. The latter is more common for adjusting the weight vector, generally using instantaneous LMS[1] algorithms. As the knowledge is stored *locally*, the LMS rule at each iteration modifies just a small set of weights, the ones for which the basis functions are active and contribute to the output. If there is plenty of data available, sampled from the process to be modeled, an offline learning algorithm is more adequate, adapting the weights either by pseudo-inverting the auto-correlation matrix or using a gradient-type algorithm.

Following the previous discussion on AMNs, by using B-splines of order k we expect to have k^n active (non-zero) functions contributing to the output of the network. Therefore for this class of AMNs we have a *generalization parameter* of $\rho = k^n$. The structure of this networks mimics the general AMNs structure, shown in Figure (2.13). Hence the output of the B-spline network is given by:

$$y(t) = \sum_{i=1}^{\rho} a_i(t)\omega(t-1) \tag{2.24}$$

Where ω_i is the weight corresponding to the i^{th} basis function, and a_i is the output of the i^{th} basis function, which depends on the input vector $x(t)$.

With respect to these networks, observe that the designer has the liberty of hard wiring discontinuities into the network. Recall that for r coincident knots in the knot sequence, the basis functions (and hence the network output) have $(k-(r+1))$ discontinuous derivatives at this point. For instance, an *a priori* information, about a discontinuous behavior by the process at a single point p, can be introduced in the model defining basis functions of order k, in an interval which contains k coincident knots. Regarding

[1] Least Mean Squared (LMS).

the network output, note that the univariate basis functions defined for each input axis are piecewise polynomials of order k, therefore the output of a BSNN is also a piecewise polynomial of order k. *A priori* knowledge about the process can then have a great deal of importance, once the designer can bias the order of the output of the BSNN to interpolate the dynamics of the process.

With respect to the network, we expect it to have the capability of **generalize** correctly outside itsr training domain. Weight convergence is essential if the BSNN is expected to generalize. This convergence is highly dependant on the conditionality of the model, i.e. if the model is *well conditioned*, which in turn is given by the *condition number* of the basis functions. Considering a univariate BSNN, where the output $y = a(x(t))w$ is defined on the interval $[o_{min}, o_{max}]$, the *condition number* of the basis function is defined as [29]:

$$C(a) = \frac{M}{m} \tag{2.25}$$

Where M and n are two positive numbers, given by:

$$m = min_{\omega} \frac{||y(x)||}{||w||} \tag{2.26}$$

$$M = max_{\omega} \frac{||y(x)||}{||w||} \tag{2.27}$$

and the following condition holds:

$$m||w|| \leq ||y(x)|| \leq M||w|| \tag{2.28}$$

The condition number of the basis functions can then be used to infer information about the convergence of the free parameters of the model. Thus, information about the bounds of the condition number are useful to ensure a proper generalization of the model. An important work by C. de Boor [30], proved that the condition number of a set of B-splines is bounded, independently of the underlying knot sequence. In [31], the same author estimated the worst possible condition number of a B-spline of order

k, with respect to the p norm, is given by:

$$C_{k,p} < k9^k \tag{2.29}$$

This condition number grows with 2^k:

$$C_{k,p} \sim 2^k \tag{2.30}$$

In the work, the author derived that the extreme case occurs for a knot sequence without interior knots.

Therefore we can conclude that the order of the basis functions should be sufficiently high that the desired function can be modelled adequately, but it should be as small as possible to keep the basis well conditioned.

2.4.2.3 BSNN internal structure

Besides the adaptation of the weights in the network, it is also possible to evolve and optimize the internal structure of the model, the network's input (number and type) and also the number, position and shape (order) of the B-spline. If the model's internal structure is not adequate, the performance of the network can be compromised and the output may not converge to the desirable region. As a consequence, the model might not successfully learn the dynamics of the process.

One heuristic taken when constructing the internal structure of the network postulates that the *simplest* acceptable network performs the best. For a multivariate B-spline with l univariates basis functions of order k, defined on each axis of the input dimension n, it is required l^n storage locations, and each input presented to the network activates k^n basis functions, producing a model of the form:

$$y = f(x_0, x_1, \ldots, x_n) \tag{2.31}$$

Increasing the model complexity results in higher *conditions numbers*, which hampers the learning process. We can see that expanding the model to higher dimensions can result in a growth of the number of parameters to impractical levels, thus methods must be found to reduce the complexity of the models. One extensively used method is the B-spline Adaptive Spline Modelling Of Observation Data (ASMOD) [32]. The ASMOD algorithm uses B-splines for representing general nonlinear models of several variables. It attempts to solve the *curse of dimensionality* by adapting the model structure to the dependencies (coupled or decoupled) that are observed in the data.

However this algorithm assumes that the desired function can be *additively decomposed*, such that it can be modelled from a linear combination of, more simple subnetworks. This decomposition is shown in Figure (2.15).

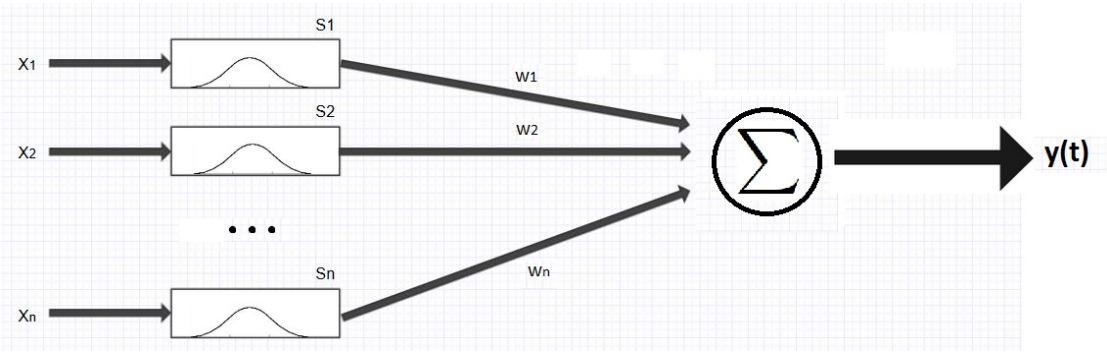

FIGURE 2.15: Additive decomposition of the network.

This figure illustrates the extreme case, where each submodel s_i is associated with an input dimension x_i. However, this addictive decomposition also agrees with the linear combination of multivarite subnetowrks. The memory requirements are thus dramatically reduced. In contrast to l^n storage locations, for the extreme decomposition we have:

$$Mem_{loc} = \sum_{i=1}^{n} l = n \times l \tag{2.32}$$

For any input the number of active functions is:

$$N_{active} = \sum_{i=1}^{n} k = n \times k \tag{2.33}$$

From this requirements it is obvious that the complexity of the model lowers by a considerable amount.

Thus in the ASMOD algorithm the output variable is modelled as a sum of several low dimensional submodels where each submodel only depends on a small subset of the input variables. The decomposition of the high dimensional input space into low dimensional additive subspaces makes the model more transparent to the user and at the same time the complexity number of parameters of the model is dramatically reduced. This process can be classified as an *empirical modelling*.

A complete overview over the algorithm is out of the scope of this work, we leave [33] as a reference. However, we highlight just the main mechanisms by which the ASMOD algorithm enhances a B-spline network. These operations are:

- **Introducing a new input variable**.
- **Modelling input dependencies**, which are found by combining the univariate and multivariate submodels to form new tensor product multivariate submodels.
- **Introducing new basis functions**. This step consists in the refinement of a representation of a certain input variable. This occurs when a new knot is introduced in the axis, to enhance the behavior of the model with respect to an input dimension.

However this refinement process increases continuously the network complexity. Therefore it is often necessary to *prune* the model by removing knots or splitting a sub network into sub models, thus simplifying the structure.

The major drawback of this approach relies on the high dependency that this algorithm exhibits in respect to the *initial model* considered. Furthermore, if the search space of the internal structure of the network is wide, i.e. extensive set of input candidates variables, knot sequences, shape of the basis functions, and biasing the model is not possible due to the lack of *a priori* knowledge about the process, more intelligent strategies need to be considered, concerning model structure optimization. *Evolutionary Computing*

(EC) provide solutions for optimizing the structure of a model, following an *intelligent trajectory* through the search space. Regarding EC, we make a special reference to *Genetic Programming*, an emerging study field that allows the evolution of more complex structures than the ones evolved by the traditional *Genetic Algorithms*. An example of such a work can be found on [34].

2.5 Model performance evaluation

Once a model is constructed, it has to be *evaluated* over a set of data. For this purpose we need a *performance criterion*, that provides a measure of the behavior of the model when presented with some set of data. A wide used criterion consists in using the *Mean Square Error* (MSE), given by:

$$MSE = \frac{1}{n}\sum_{i=1}^{n} e_i^2(t) \tag{2.34}$$

Which basically is the sum of squared errors over the set of data. Notice that this assumes a *supervised* paradigm, whose desired output should be avaiable for each input pattern. Another frequently used measure of the quality of the model (or estimator), is the *Mean Square Relative Error* (MSRE), expressed by:

$$MSRE = \frac{1}{n}\sum_{i=1}^{n} |\frac{e_i^2(t)}{y(i)}| \tag{2.35}$$

Where $y(i)$ is the observed value.

Despite the simplicity of this criteria, they do not provide information about the complexity of the model, which may be critical for real time applications, where the resources can be limited. This rises the need for more detailed indicators, that balance accuracy and complexity. Three criteria are usually employed:

- **Bayesian information criterion** (BIC)

$$K = Lln(J) + pln(L) \tag{2.36}$$

- **Akaike's information criterion** (AIC)

$$K = Lln(J) + p \tag{2.37}$$

- **Final prediction error**

$$K = Lln(J) + L\frac{L+p}{L-p} \tag{2.38}$$

Where K is the performance measure, p the size of the current model, J the MSE and L the number of patterns pairs used to train the network. BIC is a more conservative criterion when compared to AIC, insisting on a greater improvement in fit before accepting a more complex model, thus we decided to adopt the first criterion (BIC) to assess the models constructed in this work, since simple models are demanded for an embedded system application.

Concerning only the complexity of the model, the *linear weight norm* (LWN) was calculated for each model, providing a complexity descriptor, given by:

$$LWN = \sqrt{\sum_{i=1}^{n} \alpha_i + b^2} \tag{2.39}$$

Where $\{\alpha_i\}_{i=1}^{n}$ represents the number of neurons and b is the bias value. Models with a high LWN are bad conditioned models. They are usually specialized in the training data, and when considering other data sets they tend to exhibit large errors. BSNN do not admit a bias as input to the network and hence the last equation is reduced to:

$$LWN = \sqrt{\sum_{i=1}^{n} \alpha_i} \tag{2.40}$$

2.6 Model validation and stopping the training

Ultimately, the essence of learning consists in encoding an input-output mapping into the synaptic weights and structure of the model, with the hope that the network becomes so well trained that it learns enough about the past so it can estimate future behaviour.

A model validation methodology tries to assess how the network will generalize to independent data, i.e. data that was not used for training. Thus this technique provides an estimation about the performance of a predictive model with respect to unseen data.

Firstly, the available data set is randomly partitioned into a *training set* and a *test set*. The former is used to train the network, i.e. to compute the error gradient and update the network weights, while the test set is used only for testing the final solution in order to confirm the actual predictive power of the network. The training set is further partitioned into two disjoint subsets:

- *Training set*, used to train the model, though a learning mechanism.
- *Validation set*, used to validate the model.

The motivation here is to validate the model on a data set different from the one used for parameter adaptation.

The importance of these techniques comes from the possibility that a model *overfits* the training set. A model may present the best performance indicators in a set of models considered, but it might not be able to generalize to new data, due to *overtraining*, which means that the model became specialized on the training set and has lost its capability of generalization. Instead of learning the true dynamics of the process, the model also learns dynamics from external sources which do not consist in the *core* of the process. Overtraining a model can result in a network that has also learned the *noise* dynamics, inherent in data acquired by practical sensors. This is a highly non-desirable effect once we want to abstract the presence of noise in the data, so that the model can just learn the dynamics of the process.

To overcome this problem a method for stopping the training is needed, to force the model to learn what is intended. An extensively used procedure is referred to as the *early stopping method of training*. Using this method, the *estimation subset* is used to train the network in the usual way. However, the training session is stopped periodically, and the network is tested on the *validation subset* after each period of training. This works as follows [6]:

- After a period of estimation (training), the synaptic weights of the network are all fixed. The validation error is thus measured for each example in the *validation subset*.
- When the validation phase is completed, the estimation (training) is resumed for another period, and the process is repeated.

This procedure is conceptualized in Figure (2.16). By observing it we conclude that the error over the *training subset* exhibits a monotonic decreasing behavior. However, periodically testing the model in the *validation set* shows that the error in the validation phase has a quadratic behavior and so it has one global minimum. Theoretically stopping the training at this point provide the most capable model of generalizing properly. Training beyond this point, translates into a *overtrained* model.

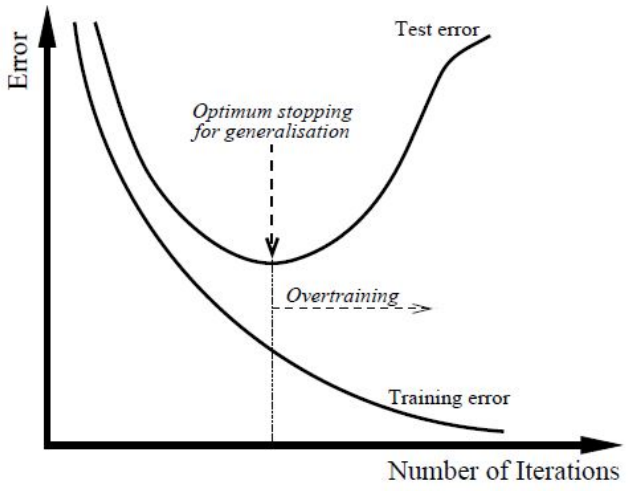

FIGURE 2.16: Illustration of the early-stopping rule based on cross-validation. [6].

2.7 Enhancing forecasting

Although it has been shown theoretically that a B-spline model has an universal functional approximating capability, and can approximate any nonlinear function with arbitrary accuracy, no universal guideline exists when choosing the appropriate model structure for practical applications. We already had seen that BSNNs follows a *fixed* three logical layers organization. However, the number of inputs of the model (*lags*),

lacks of theoretical result suggesting the best number of lags for a nonlinear forecasting problem. Thus, a trial-and-error approach or cross validation experiment is often adopted to help finding the *best* model.

Another problem arises after the training phase is complete. By this time, typically a large number of neural network models are available, which were evolved during the training phase. Thus, is is necessary to *select* the final model, which will be used in the application. Again, this model selection phase lacks of theoretical background to support the decision of the winner. A commonly followed heuristic designates the best model as being the one with the best performance in the validation set, the others are discarded.

Naturally this *keep-the-best* (KTB) approach suffers from limitations. The most obvious one resides in the fact that this network ultimately may not be the optimum model, due to the number of factors that affect network training and model selection, which can include network architecture and structure, training algorithm and data normalization. Regarding BSNNs, the number of inputs (lags) used can condition the performance of the model. Furthermore, the *data-driven* nature of neural networks might have a great impact in the model being selected. Different data sampling of a stationary process can have a significant effect in individual model selection and prediction. As a result, KTB approach can limit the generalization ability of the model. Time series forecast relies completely in this one KTB forecasting entity and hence is susceptible to abnormalities present in the model. One possible solution to this problems is to combine multiple neural networks for the time series forecasting problem.

2.8 Combining forecasts

> "In combining the results of these two methods,one can obtain a result whose probability law of error will be more rapidly decreasing."
>
> — Laplace, 1818

Combining several forecasting entities to enhance forecasting accuracy of the predictive system has been widely studied over the years. A survey of the work and bibliography

in this area can be found at [35], where the author points to the agreement observed in this line of research that forecast accuracy can be substantially improved through the combination of multiple individual forecasts. Another important conclusion states that *simple* combination methods often work reasonably well relative to more complex combinations. Besides alleviating the model selection phase, combining networks outputs can in fact contribute to increase the robustness of the predictive system, producing more stables forecasts and reducing the probability of incurring into catastrophic predictions. This paradigm is illustrated in Figure (2.17).

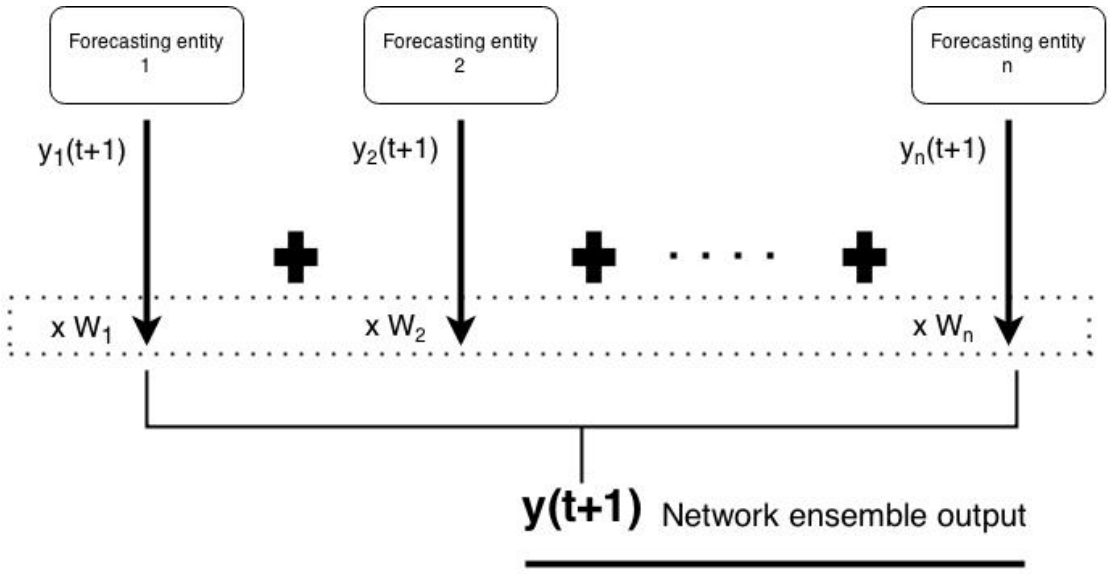

FIGURE 2.17: Weighted combination of forecasts.

As so, the output of the ensemble network is given by:

$$y(t+1) = y_1(t+1)\omega_1 + y_2(t+1)\omega_2 + \ldots + y_n(t+1)\omega_n = \sum_{i=1}^{n} y_i(t+1)\omega_i \tag{2.41}$$

Although with some constraints on the weights, to insure numerically stability:

$$\sum_i \omega_i = 1 \tag{2.42}$$

$$\omega_i \geq 0, \quad alli \tag{2.43}$$

2.8.1 Theory behind neural networks ensembles

The implications of making the right choice are extremely important both from a theoretical standpoint and in practical terms. Small improvements in forecasting accuracy can result in considerable savings. The foundations of the theoretical background in this area can be found at [35]. Contributions from psychology are also worthy of mentioning, the theory of group processing information (GPI) can establish a motivational background to the employment of this techniques. Often real world decisions rely on information from a panel of experts (council). Understanding how a panel of experts processes information and formulates a consensus might greatly improve our use of expert information. The motivation behind these techniques is justified largely by empirical results in opposition to theory. The authors in [36] term this empirical methodologies as *romantic* as opposed to *classic*. The *classic* style is theory driven that contrasts with empiric nature of the *romantic* style.

Fortunately, the construction of good forecasting ensembles is often possible. One of the strongest fundamentals is *statistical*. Assuming that an *accurate* group of forecasting entities is available, different choices among the group may lead to similar accurate results in the predictions. Constructing an ensemble out of all of these accurate entities, by averaging their outputs, enhances the prediction performance. Another strong reason is *representational*. In most neural network applications, the true function f might not be satisfactorily approximated by the KTB approach. By forming weighted sums of predictions, it may be possible to expand the space of representable functions. This contradicts the observation made over the universal approximator nature of B-spline functions. However, we are dealing with finite training sets, thus the learning algorithms will explore only a finite set of functions.

Works have shown that in practice there is no interest in combining models that have their outputs correlated. The performance enhancement obtained by combining forecast entities is naturally inversely proportional to the correlation between the forecasting entities. Although network ensembles can effectively improve model variance and improve the network generalization ability, the effectiveness is largely limited if the errors generated by the different models are correlated [37]. The idea of combining forecast

implicitly assume that different models are able to capture different aspects of the information available for prediction. In the well-known M-competition [38], the results of combining the models led to robust predictions, since the group of entities performed well for most of the various type of series. An important work in this area can be found at [39], where the authors introduce the concept of *ambiguity*. For an individual network i, its ambiguity is defined by:

$$a_i(x) = (V_i(x) - \overline{V}(x))^2 \tag{2.44}$$

Where $a_i(x)$ is the ambiguity of the network i on input x, $V_i(x)$ the output of network i, and $\overline{V}(x)$ is the ensemble output to the same input.

Thus the *ensemble ambiguity* may be expressed as:

$$\overline{a}(x) = \sum_N \omega_i a_i(x) = \sum_N \omega_i (V_i(x) - \overline{V}(x))^2 \tag{2.45}$$

Assuming an ensemble of N networks. The last expression consists of the variance presented in the weighted ensemble, around the weighted mean and hence it gives a measure over the *disagreement* among the networks to the input x. Assuming that $f(x)$ is a real function, the quadratic *individual* and *ensemble error* are, respectively, given by:

$$\epsilon_i(x) = (f(x) - V_i(x))^2 \tag{2.46}$$

$$e(x) = (f(x) - \overline{V}(x))^2 \tag{2.47}$$

Substituting (2.46) and (2.47) in (2.45) , yields:

$$e(x) = \sum_N \overline{e}(x) - \overline{a}(x) \tag{2.48}$$

Where $\overline{e}(x)$ is the weighted average of the individual errors. Averaging all the terms in (2.48) over the input distribution, i.e, $Y = \int p(x)y(x)dx$, we reach to the *ensemble generalization error*:

$$E = \overline{E} - \overline{A} \tag{2.49}$$

Where $\overline{E}$ is the weighted average of the individual generalization errors and $\overline{A}$ the weighted average of ambiguities. This last expression dictates that the generalization error of the ensemble network has two independent terms. One that only depends on the generalizations errors of the individual networks and a second term that isolates the correlations between networks. Equation (2.49) shows that the ensemble generalization error is always smaller then the weighted average of the ensemble errors. It is important to highlight the need to increase the ambiguity among the ensemble, as the authors clearly demonstrated. The networks *should* disagree.

Further theoretical works still needs to be done in this area. However the empirical work has proven the success of this approach and thus it is employed in this work. If several different models can be combined to obtain a better forecast, it should theoretically be possible to construct a single model that makes optimal use of the different kinds of information used by the forecasts pieces in the combination. Nowadays we can point two main directions of forecast combination that can be found in literature [40]: combining for adaptation and combining for improvement. The first one targets the best individual performance among the pool of forecast candidates. The second one aims at significantly outperforming each individual forecast candidate. In this work we intend to combine to achieve improvement.

Neural ensembles have been well studied and applied for pattern classification problems, by using *boosting* and *bagging* voting classification algorithms [41], few applications have been reported in forecasting applications. At [42], the improvement of time series forecasting performance, using neural networks assembles, was assessed in comparison with the traditional KTB approach. The methods were applied to the problem of exchange rate forecasting, and the results show a consistent increase in performance in the test set, suggesting an improvement of the generalization capability of the overall system.

2.8.2 Designing the ensemble

A parallel system whose individual decisions are combined by some system of weighted or unweighted voting is an *ensemble classifier*. From last section it is important to retain two ideas. Firstly, the correlation among predictions is the bigger limitation factor to the

success of the network ensemble and hence it should be minimized. Secondly empirical results show that simple combining schemes generally work better when compared to complex forecast combining methods.

In this work we shall begin to consider the traditional KTB approach. The performance of this methodology serves as comparison to assess the performance of the forecast ensemble. In a first phase a trivial combination of the estimates is employed, where a simple average with equal weights is evaluated. This method adds little effort to the system and is backed-up by strong empirical results in this line of research. After this, at the cost of extra complexity, more advanced techniques can be applied to enhance the predictions of the system.

We shall consider a group of 4 forecasting entities (models). When a large number of candidates to the network ensemble is verified, [43] suggests an *genetic algorithm* (GA) to choose a suitable subset among the whole space of candidates.

Training and validating all the models with the same data sets generally leads to a high correlation levels among the predictions. Due to the lack of abundant data of the process intended to model, the reduction of the harmful correlation is forced by randomizing the data that constitutes each set (training, validation and test). This approach is justified by the *unstable* nature of neural networks, as noted in [44]. which means that it is not necessary that, for a trained neural network, small changes in the input translate to small changes in the output. Therefore one should expect major changes in the output function, encapsulated in the network, in response to small changes in the training set. Randomizing training sets can indeed act as a decorrelation agent between the models and hence increase the ensemble ambiguity.

Furthermore, results have shown the expected observation that the number of lags in the neural network model largely determines the autocorrelation structure of a time series. Therefore if the neural networks models are built using the same number of inputs, the predictions will be highly correlated and, consequently, also the errors. Thus, the effectiveness of the ensemble method is reduced. In such a situation the group does not

benefit from a wide variety of opinions and hence the errors can not be compensated by a distinct individual. To diminish the autocorrealtion of the time series, each one of the four models is built with a distinct lag input number, from 2 to 5. By doing this, the ensemble as a whole should have the benefit of reduced correlation and increased modelling power. If the data was abundant, completely distinct, non-overlapping sets could be employed for the construction of the models, forcing decorrelation among forecasts even further. Adding more input variables (lags) to the network does not rise serious concerns due to the employment of the ASMOD algorithm, which breaks complex multivariable models into additive, more simple, submodels, thus allowing us to relax the curse of dimensionality present in BSNNs.

After the trivial solution, the forecast combining method can be made more complex. In [45] the authors tried a variety of methods for combining time series forecasts, and the results have demonstrated that better results are achieved by calculating combing weights on the basis of relative precision, ignoring any known correlation between the models in estimating combining weights. This suggestion is followed in this work. The natural extension to the trivial solution consists on revising the weighted combination.

There is no analytical solution for the optimum weights, thus they should incur into an optimization process. A least square solution, that minimizes the error using the validation and the training set, by finding the optimum weight vector $\overline{\omega}$ that minimizes an error criterion can be found, although in a non-trivial way. However, ordinary least squares methods often do not provide satisfactory results in real applications due to the variability of weight estimations.

2.8.2.1 Evolving the ensemble

Network ensemble theory suggests a weighted combination of forecasting networks in such a way to minimize the ensemble generalization error:

$$E = \hat{E} - \hat{A} \tag{2.50}$$

Some insight can be developed by trying to minimize the last equation in order to the weights,

$$\frac{dE}{d\omega_i} = 0 \tag{2.51}$$

Which admits the following solutions:

$$(\epsilon_i - A_i) = E \vee \omega_i = 0 \tag{2.52}$$

Ideally the difference between the generalization error and its ambiguity should be identical among the networks. In [43] this work was extended, and the ensemble generalization error was represented using correlations between the individual networks:

$$E = \sum_{i=1}^{N} \sum_{j=1}^{N} \omega_i \omega_j C_{ij} \tag{2.53}$$

Where C_{ij} is the correlation between $y_i(x)$ and $y_j(x)$, expressed by:

$$C_{ij} = E[(f(x) - y_i(x))(f(x) - y_j(x))] \tag{2.54}$$

Where $f(x)$ represents the desired output to the input x, $y_i(x)$ and $y_j(x)$ represent the output of network i and j, respectively, when the input x is applied. Equation (2.53) emphasizes the need to avoid correlations among the networks to minimize the final ensemble generalization error.

Then a method for an optimum weight solution, in theory, was derived. This vector minimizes the expected prediction error of the ensemble by making use of an estimation of the error correlation matrix. However the calculation of this correlation matrix might not be straightforward, due to ill-conditioned or even an irreversible correlation matrix. Instead an evolutionary strategy (ES) was chosen as the process to optimize the weight vector, which ideally would be:

$$\hat{\omega} = arg \quad min(\sum_{i=1}^{N} \sum_{j=1}^{N} \omega_i \omega_j C_{ij}) \tag{2.55}$$

The ensemble weights are evolved by minimizing the expected error in the joint validation sets of the four models. In section (4.4.3) details about the ES implementation are

briefed.

2.8.2.2 Increasing the ambiguity

In a second experiment, the simulations are repeated but this time we propose to deliberately degrade the available data set, by adding Gaussian noise. This methodology is motivated by three reasons. Firstly, by doing so the robustness of the models is tested, and the adequacy of BSNN to the problem is assessed. Secondly it can also be used to assess the model ability to be integrated in a biomedical instrumentation system, together with a temperature estimation method, which will inevitably produce some errors. A more detailed explanation for this second point is given in Section (4.4). Lastly by randomly adding noise to an original, noise free data set, it is possible to train multiple networks with completely different data sets, thus broadly increasing the ambiguity among the networks.

Nevertheless, it is crucial that the model's structure has just the right amount of function approximating power, when noise is added. If the network is too much powerful it may incur in learning the noise dynamics, which constitutes a scenario that we are not interested. Therefore, the network structure should be biased to learn just the real process dynamics and ignore the noise.

Negative correlation learning (NCL) [46] approaches complete the state-of-art concerning neural networks ensembles. This methods add a penalty term to the cost function, enforcing a weak relationship among the entities of the ensemble, work that was extended in [47]. The goal of this penalty term is to measure the error correlation between the ith network output and the rest of the ensemble. The error correlation $P_i(n)$ is obtained by doing:

$$P_i(n) = (F_i(n) - F(n)) \sum_{j \neq i} (F_j(n) - F(n)) \tag{2.56}$$

Assuredly the purpose of this methods focus on decreasing the correlation between the individual errors, thus increasing the ambiguity. Under the NCL paradigm, all the networks are trained *simultaneously* and *interactively* using the same data set. Therefore

the errors of the individual networks are being forced to be uncorrelated at the training phase.

2.8.2.3 An ensemble of degraded networks

The addition of noise to the data set does not, however, guarantee a network ensemble with uncorrelated errors. Increasing the noise strength can compromise the learning process, thus other alternatives must be available to enforce uncorrelated errors among the ensemble. In [48] the authors propose the creation of an *ensemble of degraded neural networks*. In a first phase a single accurate network is trained, the *base network*, then the neural network ensemble is formed by *degrading* the base network, adding controlled noise to its parameters. The results shown suggests that such an ensemble can improve the performance of the base network. Another major advantage is the time required to construct the ensemble, since it is only needed to train a single (base) network.

By degrading the base network, we expect the degraded networks that compose the ensemble to have their errors weakly correlated. Let $N(W)$ be a neural network trained using the training set $\mathcal{L}$, the *base network*. W is a vector containing all learnable parameters of the network, $w = (\omega_1, \omega_2, \ldots, \omega_p)$. Causing a small perturbation in this vector will generate a different network, whose performance would still be comparable with the base network. An ensemble of this degraded networks is created, which then is combined to form the final prediction.

A degraded version of $N(W)$ can be obtained by adding a zero mean Gaussian noise to each of its learnable components. Thus, if $w = (\omega_1, \omega_2, \ldots, \omega_p)$ is the parameter vector of the base network $N(W)$ and $W^d = (\omega_1^d, \omega_2^d, \ldots, \omega_p^d)$ is the parameter of a degraded version of $N(W)$, then:

$$\omega_i^d = \omega_i + e_i, \quad \forall i = 1, 2, \ldots, p \tag{2.57}$$

Where $e_i \sim \mathcal{N}(0, \sigma)$, i.e. e_i is a random number drawn from a normal distribution with zero mean and variance σ. Furthermore, to generate each component of the parameter vector of a degraded network, e_i is drawn independently from its previous values. Thus the amount of degradation is controlled by σ, the *degradation parameter*. Let ϵ be the

training error of the base network $N(W)$ on the training set $\mathcal{L}$ and let ϵ_d be the error committed by the degraded network $N(W)^d$ on $\mathcal{L}$. $N(W)^d$ is considered a *valid candidate* if $\epsilon_d \leq t \times \epsilon$, where t is an user-defined threshold, the *selection threshold*. Therefore by setting $t = 1.05$ we are assuming a degraded network to be a valid candidate if the error committed by it on the training set is within 5% of the base network.

This work was originally derived for multilayered perceptrons (MLPs), hence the parameters vector contains all of the weights and bias, which are the learnable components of a MLP. The general network structure of an associative memory network, which BSNNs falls into, was presented in Figure (2.13). We will extend the degraded neural networks ensemble work to BSNNs. The *learnable* layer of AMNs is the third one, which contains the weight vector that linearly combines the output of the basis functions. However we propose to also degrade the *fixed* basis function layer of AMNs, in order to try to decorrelate even more the degraded versions. The middle fixed, *non adaptive* AMN layer performs a *non linear mapping*, from the input x_i to the output of the basis function a_i, $x_i \rightarrow a_i$. In Section (2.3.1) we defined a *spline* f in its B-form as:

$$f = \sum_{j=1}^{n} B_{j,k} a_j \tag{2.58}$$

Also it was shown that a B-spline is dependant on its *knot sequence* t and the order k.

$$B(.|t_i, \ldots, t_{i+k}) := B_{i,k} \tag{2.59}$$

Thus f is a spline of order k with knot sequence t, i.e, a linear combination of B-splines of order k for the knot sequence t.

From the study of B-splines it was noticed that one of the major advantages of BS is the robustness of this function to changes in the *control points*, illustrated in Figures (2.7) and (2.8). In general the structure of BSNNs, Figure (2.13), the control points are represented by the last layer, in the form of the adaptive weight vector. BSs offers a *localized control*, which means that a perturbation in a control point, i.e. a weight in the last layer, causes just a local consequence, i.e the changes are not propagated

throughout all the functions. Therefore degrading just the adaptive weight vector of the BSNN base network might not be sufficient to create an uncorrelated ensemble of degraded networks, due to the localized control of BSs, hence instead of just degrading the *control points* we propose to also degrade the non linear mapping performed in the second layer. It seems intuitive that deforming the non-linear process performed by the basis functions will lead to a more distinct network. Let $f(k,t)$ be a spline of order k with knot sequence t. Notice that by applying the ASMOD algorithm it is possible that the BSNN is composed of multiple additive splines. However we assume one spline, without loss of generality. A degraded version of the base spline $f(k,t)$ can be obtained by perturbing each one of the knot sequences t_i that form the B-splines $B_{i,k}$ that, when linearly combined, form the spline $f(k,t)$. Thus we define the *base knot vector* as:

$$t_b = (t_{1,1}, t_{1,2}, \ldots, t_{1,p_1}, t_{2,1}, t_{2,2}, \ldots, t_{2,p_2}, \ldots, t_{n,1}, t_{n,2}, \ldots, t_{n,p_n}) \tag{2.60}$$

Where n is the number of basis function that form the spline f and p_i is the length of the knot sequence of basis function i, $B_{i,k}$.

A degraded spline f^d is formed by degrading the base knot vector, where the degradation of each knot is made by adding noise in a controlling way:

$$t_i^d = t_i + e_i, \quad \forall i = 1, 2, \ldots, p_i \tag{2.61}$$

Where $e_i \sim \mathcal{N}(0, \sigma)$.

Therefore we have two degradations parameters:

1. weight degradation parameter σ_w.
2. knot degradation parameter σ_t.

Both can be chosen independently. In the original work [48] is suggested to select σ in the range of [0.001,0.002]. Degrading the base knot sequence must be done in a constrained way, since this sequence should be a *non decreasing* sequence:

$$t_i \leq t_{i+1}, \quad all \quad i \tag{2.62}$$

Despite this expansion of the *degraded network ensemble* concept to deal with B-splines, we decided to not apply it in this work, due to the extensive contours that this work is moving to. Nevertheless we leave the proposed adaptation of the methodology to B-splines for future research.

2.8.3 Neural dynamic ensemble optimization (NDEO)

Ensemble learning involves two stages: training the networks and combining their outputs. Most of the effort is being done to find an optimal weight vector $\hat{\omega}$ that minimizes some criterion. Naturally training methodologies to increase the ambiguity, as well as optimization methods to select the best candidates to form an ensemble, are crucial. Nevertheless the way these individuals networks are combined is also vital to the system performance. In neural forecasting applications this last combing step is traditionally made using an (optimized) weight vector.

Adding *intelligence* to the combination of the individual networks seems a reasonable idea. A vector of weights lacks of flexibility, it is not adaptive and does not take into consideration the current region of the process domain. This linear combination scheme may not explore the individuality of each model, thus we suggest a non-linear combination. This mechanism should combine the networks in a *dynamic* way, taking into account current information about the process. More importantly, this mechanism should *generalize* the best possible way of combining information from different sources, having current information about the process dynamics. This paradigm of ensemble output combination is done in an *active* way, in contrast to the *passive* traditional weighted sum. We believe this paradigm can provide means to enhance the knowledge present in each network, as well as mask the individual flaws and attenuate them.

The proposal consists in arranging a combination of the individual networks by means of a *second-layer* neural network, that acts as an optimization agent, combining the

information from all the different sources. A process we call *neural dynamic ensemble optimization (NDEO)*, illustrated in Figure (2.18).

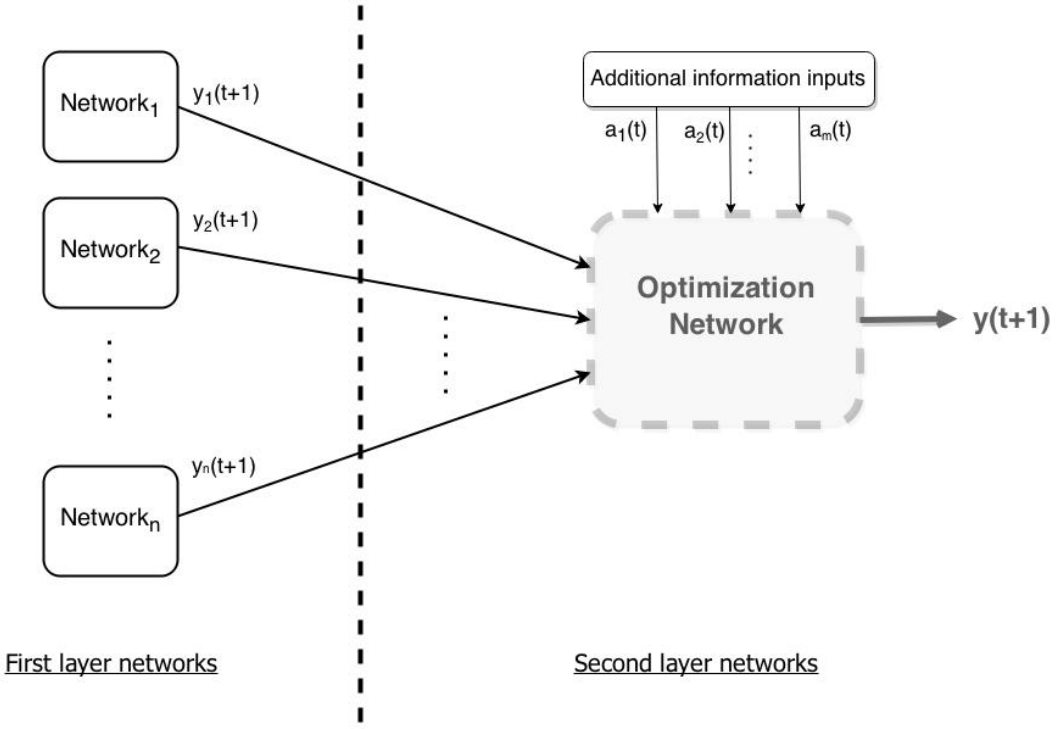

FIGURE 2.18: Purposed architecture for a neural ensemble system, employing NDEO.

The purposed 2-layered architecture comes in opposition to the traditional, weight combining, architecture shown in Figure (2.17). The goal of the second layer is to expose the outputs of the individual networks to a dynamic, adaptive optimization process. Note that last network admits additional information (a_i) to the optimization process, which might be crucial to the process. A BSNN is to be placed in the second layer, thus caution is needed concerning the number of inputs to this last network. If a high number of individuals is needed to compose the ensemble, a different typology of NNs should be employed in the second layer.

Furthermore it should be noticed that an intelligent ensemble combination inherently increases the system complexity and demands a higher effort in the modeling project. However, if the ambiguity can be forced to satisfactory levels, NDEO can be used to explore and enhance the particularities of each individual network. As so, the system should be able to dynamically decide the best arrangement of the outputs to construct the prediction. The second layer output is given by:

$$y(t+1) = f(y_1(t+1), y_2(t+1), \ldots, y_n(t+1), a_1(t), a_2(t), \ldots, a_m(t)) \tag{2.63}$$

Where $y_i(t)$ represents the individual next-step prediction of network i, and $a_i(t)$ consists of additional information up to moment t, that may guide the network towards an optimal combining of $y_i(t)$. $f(\ldots)$ is the underlying function to the NN, performing a non linear mapping $\mathbb{R}^{n+m} \to \mathbb{R}$. We expect this mapping to minimize the generalization error. However it should be clear that this minimization is just useful if the individual networks are highly uncorrelated, i.e. high ambiguity, providing that the individual networks are admitted to be *accurate*.

Albeit more intelligent, this combining mechanism introduces more load into the system, which can be problematic in some real time applications and embedded system with scarce resources. However, after the training of the second layer network, this is *added* to the network and should not add a significant delay if properly designed. Nevertheless NDEO can only be justified by large gains in performance, which we intend to assess in this work.

3

Experimental set-up and data acquisition

3.1 Introduction

Three steps are involved in developing a neural network to achieve a reliable prediction: specifying a suitable network architecture, choosing the training data, and training the network. In the last chapter the architecture of the network was discussed. Associative Memory Networks, a class on which BSNN are embedded, provides a suitable architecture for the prediction of a time series. The next step is to collect the data needed for training the model, which we discuss in this chapter. In Chapter(4) the specification of the methodologies used to train the network (the last step) are presented.

The performance of *data-driven* models is highly dependent on the quality of the captured data. This data should represent all the domain of the process aimed to model. It is generally difficult to incorporate prior knowledge into a neural network, therefore the network can only be as accurate as the data used to train the network. Therefore

we reserve the following chapter to fully characterize the experimental environments constructed to extract the data used in this work. More details can be found on the original work [7] where that data was acquired. Furthermore this chapter provide a measure of the validity of the data used to construct the models, and it can be used to discuss about how close the environments considered resembles the real, ideal human tissue characteristics.

An overview about the materials used is present in Section (3.2), and the hardware and software configurations are explained in Section (3.3). The development of the sequential of experimental setups assumed an increased complexity along the work. Homogeneous phantoms were considered. The experimental setup developed is presented in Section (**??**), where the temperature propagation in a homogeneous *phantom* was measured. The experimental procedure approach is explained in Section (3.6).

3.2 Materials

In order to simulate **human tissue**, a matrix solution studied in [49] was used. Mimicking solutions are named *phantoms*, a material exhibits similar characteristics found in human tissue. The basic composition of the solution is shown is Table (3.1).

Material	% Mass
Water	86.5
Glycerol	11
Agar	2.5

TABLE 3.1: Composition of the constructed solution used to resemble human tissue.

In order to adjust the attenuation coefficient graphite powder was added. Ultrasonic *phantoms* are created as to respect biologic tissue properties such as sound speed, acoustic impedance and attenuation coefficient.

Justifications regarding the solution composition are present in the original work.

3.3 Experiment configurations

In order to force the results obtained to be reliable, efforts have been made to assure a fault-tolerant hardware configuration, with respect to noise, coupling of the transducers, connection, and other possible flaws.

For the localized heating of the *phantoms*, a therapeutic ultrasound device (TUS), *Sonopulse Generation 2000, Ibramed*, was used. It contains a two-face transducer, i.e. two nominal effective radiation areas (ERAs), one with $1cm^2$ and another with $3.5cm^2$. The biggest face allows for the use of frequencies ($1MHz$ and $3MHz$). The ERA level of $3.5cm^2$ was employed, otherwise the focused heating area would be too small. Furthermore, the therapy goals considered in this work focus in deep areas of the tissue, as is the case of cancer treatments. Thus the use of $1MHz$ frequency is recommended, as longer waves are less attenuated then short waves, enabling deeper penetration in the tissue. The T device has two operation modes, continuous and pulsed. The later was applied in this work. Intensities from 0 to $2W/cm^2$ with increments of $0.1W/cm^2$ can be transmitted to the media, using any of the transducer faces. Further characterization of the device is shown in Figure (3.1), where the acoustic pressure distribution of the TUS is presented .

This profile was taken assuming a room temperature of 24 ^oC, employing a $1MHz$ frequency. Analyzing this pressure profile, it is possible to conclude that the transducer has it natural focus at $42mm$, i.e. its near field length (NFL) is 42 mm. Furthermore, the acoustic pressure applied by the TUS is approximately uniformly distributed in space, as shown in Figure (3.2), where the spatial pressure field is illustrated on a plan parallel to the face of the transducer at $48mm$ distance.

Temperature at the spatial points (inside tissue) under study was measured using type-K thermocouples, connected to a compensation module (*80TK, Fluke, Everett, WA, USA*). This module is then connected to a digital multimeter (*2700/7700, Keithey*), which digitalises the temperature and makes it available to a general purpose PC. These temperature values were transferred to the PC via a GPIB bus (*GPIB-USB-B, National Instruments*). The acquisition of the data was handled by an open-source application, *Echotherm* [50], which was specially developed for this type of experiments.

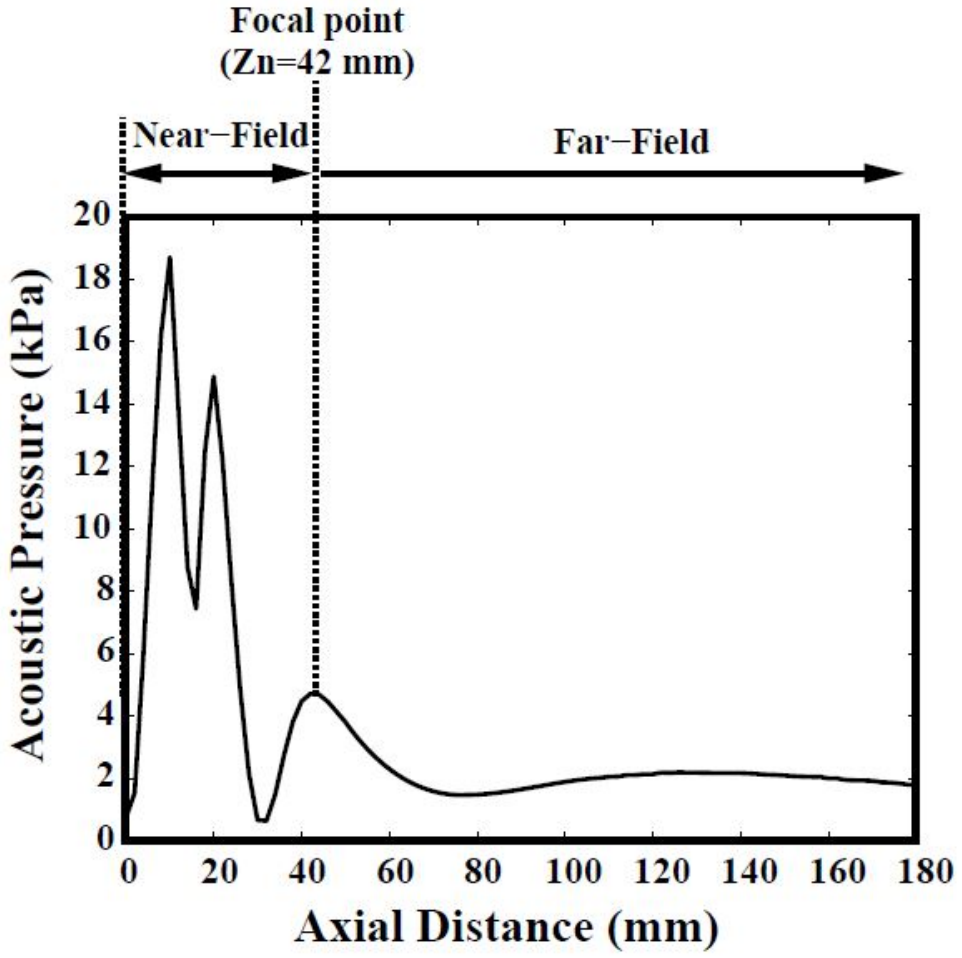

FIGURE 3.1: Pressure profile across the axis of the therapeutic transducer. Figure taken from [7].

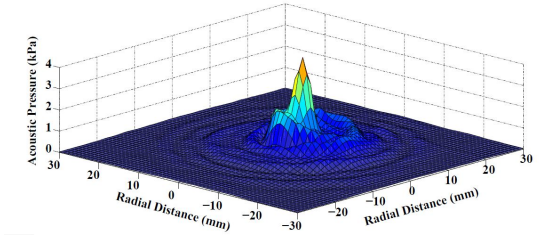

FIGURE 3.2: Pressure field of the therapeutic transducer measured in a plan parallel to the face and at 48mm distance. Figure taken from [8].

3.4 Setting-up

The temperature propagation was measured in an invasive way, inside the prepared solutions. The assembly of the experiment to collect the data is detailled in the following

section.

For the experiment a simple homogeneous *phantom* was considered. The setup is illustrated in Figure (3.3).

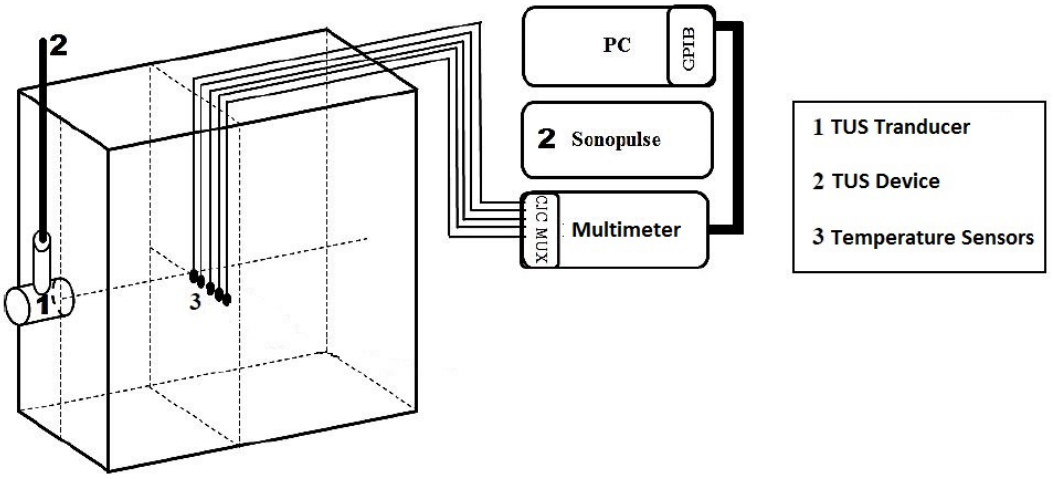

FIGURE 3.3: Schematic diagram of the experiment setup used in the first environment, homogeneous *phantom*. Figure adapted [7].

A parallelepiped phantom holds five thermocouples (device 3, in Figure (3.3)), connected to the multimeter, that invasively measures the temperature inside the phantom. In one exterior side of the phantom, the therapeutic ultrasound device was positioned (device 1, in Figure (3.3)). Samples were taken from the sensors and processed by the *Ecotherm* software.

3.5 Sensor positioning

This section is reserved to briefly explain the positioning of the sensors inside the *phantom*. As previously mentioned, the natural focus of the TUS device is at $42mm$. Therefore, the sensors were positioned already outside the near-field, in the far-field Figure (3.1), $50mm$ away from the transducer face, forming a parallel line to it. In the experiment, five sensors were considered, Figure (3.4). The thermocouples are spaced with $5mm$ intervals.

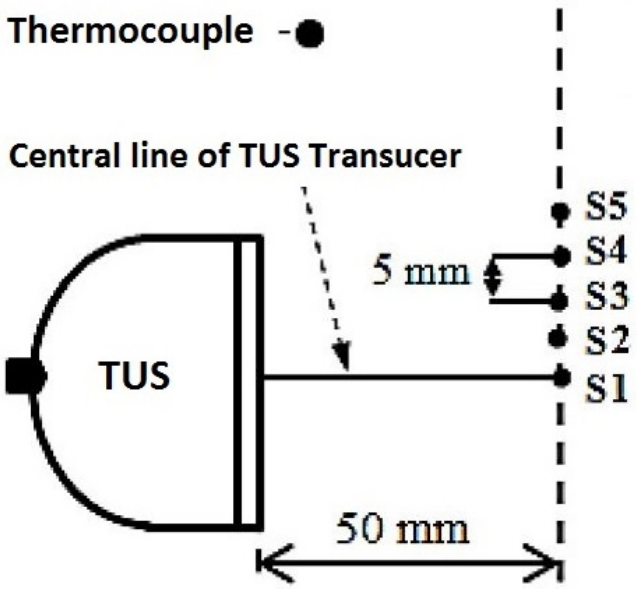

FIGURE 3.4: Thermocouple positioning in relation to the TUS device. Figure adapted [7].

The thermocouples were placed after the natural focus because after this point the beam geometry is more well-behaved, being spread as the axial distance increases (in the far-field) the energy is more and more spread. From the spacial distribution of the acoustic pressure, Figure (3.2), it is expected a more energetic TUS transducer central line, therefore the heating, experienced by the sensor placed over this line, is also expected to be the most significant one. The acoustic pressure should then decay as we move through the rest sensors, away from the TUS transducer central beam.

3.6 Experimental procedures

Each experiment trial has a 45 minute duration, divided in three phases. The first 5 minutes serve as a reference for future measurements, in following 20 minutes a heating process occurs, provoked by switching on the TUS transducer. The last 20 minutes are reserved for the *phantom* to experience a natural cooling process, where the the TUS device was turned off. Temperature samples, measured by the sensors, were taken each 10 seconds. Therefore, for each trial $N = 6 \times 45 = 270$ temperature data points are

available.

Four beam intensities were considered: 0.5, 1.0, 1.5 and 1.8 W/cm^2. Each trial assumed a different beam intensity which, in turn, led to a data file. In all of the experiments, a $1MHz$ frequency was employed.

3.7 Experimental results

In this section the data obtained in the experiments[7] are presented. Results are shown in a graphical form. Namely, the signals measured by the thermocouples, connected to the multimeter, are shown for each trial considered. It is intended the reader to get familiarized with the process we want to model. The shown results constitute the basis for discussion in respect to the quality of the data used to build the posterior BSNNs models.

Following the experimental schematic as shown in Figure (3.3), four beam intensities were applied to the simple homogeneous *phantom*. Figures (3.5) and (3.6) present the temperature registered by each sensor, considering the four cases of beam intensities.

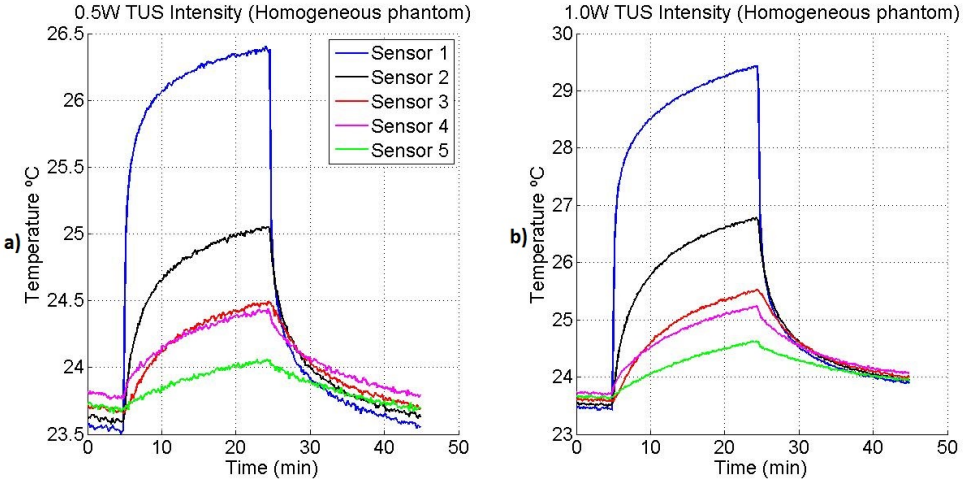

FIGURE 3.5: Temperature recorded by the temperature sensors in the experiment described in Section (**??**), considering a beam intensity of: a) $0.5W/cm^2$ and b) $1.0Wcm^2$.

The highest temperature, in all trials, is naturally registered by *sensor 1*, once it is positioned on the TUS transducer central beam axial line. The behavior of the temperature for each spatial position, as shown by the temperature progress recorded by the sensors,

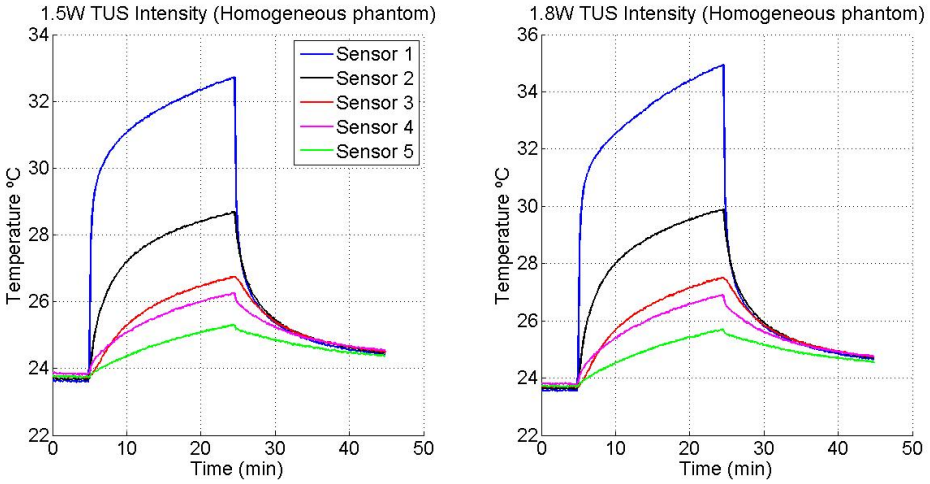

FIGURE 3.6: Temperature recorded by the temperature sensors in the experiment described in Section (**??**), considering a beam intensity of: a) $1.5W/cm^2$ and b) $1.8Wcm^2$.

is justified by its position relatively to the center of the TUS face.

As mentioned before, each 45 minute trial starts with a 5 minute long interval, where the *phantom* does not suffer any intervention, its temperature remains almost constant. After this interval, the TUS transducer is turned on and the heating process begins, which provokes a raise in temperature at the focused and surrounding regions. After 20 minutes, the external energy source is turned off, the *phantom* cools in a natural way, and the temperature is recorded by another 20 minutes interval. We observe that, despite the beam intensity considered, temperature curves behave is similar. Naturally a more intense beam is translated into higher temperature values experienced in the *phantom*. Nevertheless, process dynamics remains the same.

3.7.1 Final remarks

Observing the plots illustrated in Figures (3.5) and (3.6), we can state some particularities of the data. The most sensitive area of the data domain resides at temperatures experienced at sensors close to the TUS transducer central beam axial line, which is highly aggravated if a strong intensity is being applied. Abrupt temperature variations are observed under this circumstances, which assuredly will constitute challenging areas to be modeled. As we move away from the TUS transducer central beam axial line,

temperature variations tend to be smooth, with smaller thermal amplitude during the course of the therapy session.

4

Applied estimation models

4.1 Introduction

This chapter exposes the different temperature predictive models applied in this work. An approach of gradually increasing model complexity was taken while modeling the dynamics of the process. We start by considering models for single-point and single-intensity estimation. Then gradually the complexity of the models is increased towards multi-point and multi-intensity estimation. The motivation behind this methodology is due to the interest we have in study and develop insight concerning the feasibility of using BSNN to predict the temperature propagation, in the environments characterized in Chapter(3). Separating between the complexity of the predictive models is made in order to clearly observe the ability of BSNNs to *generalize* . Firstly specific environments are considered, on which the temperature propagation is considered regarding a single-intensity at a specific point, and then we gradually move towards a more *generic* therapeutic environment, where the process is modelled concerning a discrete region

(multi-point), and all the intensities are considered.

The chapter starts by presenting the pre-processing methods applied to the data collected from the experimental setups presented in Chapter(3). The overall pre-processing ends with data selection for the different model construction phases, i.e *cross validation*. Then follows some network design considerations concerning the B-spline design cycle employed. The algorithms involved in structure selection and network training are presented. Some *a-priori* knowledge about the process is also forced into to the network. Section (4.3) exposes some considerations about the modelling scheme to be applied in this work.

4.2 Modelling methodology

4.2.1 Data preparation

For modelling purposes, we make the distinction between to phases of the temperature propagation.

- Heating phase.
- Cooling phase.

After a heating period, the TUS device is turned off and the *phantom* is let to cool in a natural way. These to distinct phases can be observed in the example shown in Figure (4.1), where the two cycles are marked.

Following the two different cycles of the complete process, two models are always considered, one for each phase. The reasoning behind this approach is due to the fact that the dynamics of the two phases are governed by different rules. The heating experienced in the *phantom* is forced by the TU, applying sound waves of different intensities. On the other hand, once the device is turned off, the *phantom* cools down in a natural way, following the laws of thermodynamics, without an external source doing work on the system.

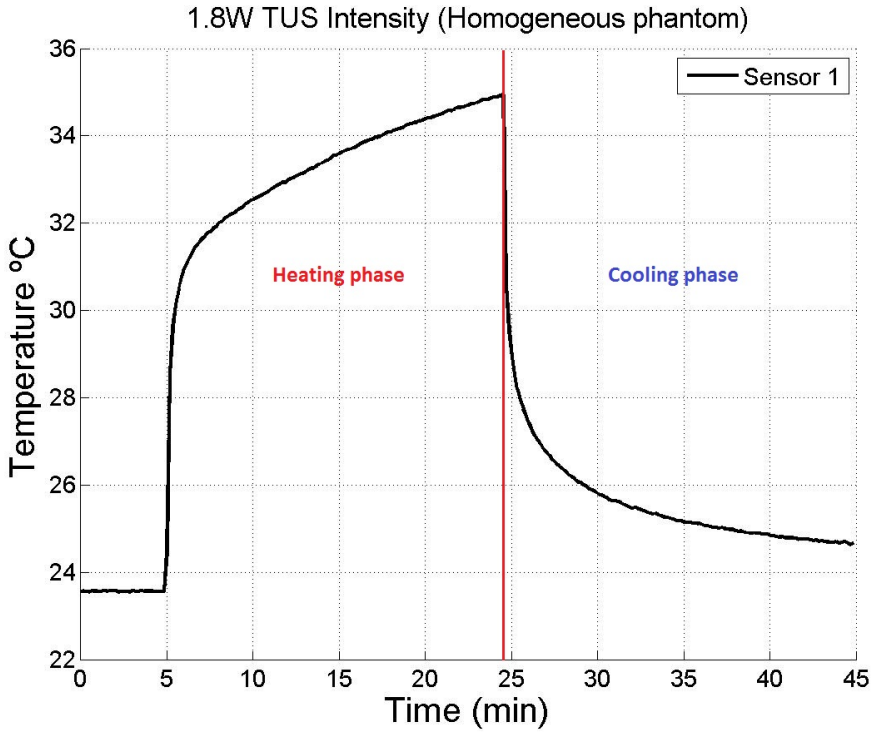

FIGURE 4.1: Two distinct phases can be observed in the temperature propagation. Firstly the temperature rises due to the ultrasound being applied to the *phantom*. After some time, the device is turned off and the *phantom* cools down naturally.

The data is divided into two subsets, one for each cycle of the process. Each one of the subsets are used to construct a model. Furthermore, the results and conclusions concerning a scenario of modelling are presented considering always the two models.

Due to the inherent noise present on the experimental set-up and consequently on the sensor's measurements, the peak value of temperature in each experiment usually does not coincide with the time instant where the TUS device was turned off, thus indicating the start of the cooling process. Furthermore, all the experiments assumed an initial five minutes interval on which the TU was still not active. This stationary interval can be observed in Figure (4.1). Therefore the data must be processed before being used to train the model. The starting point of the heating phase, as well as the one that marks the start of the cooling phase, must be found. A *Moving Average* (MA) filter was employed to obtain a more noiseless version of the data. An example is shown in Figure (4.2).

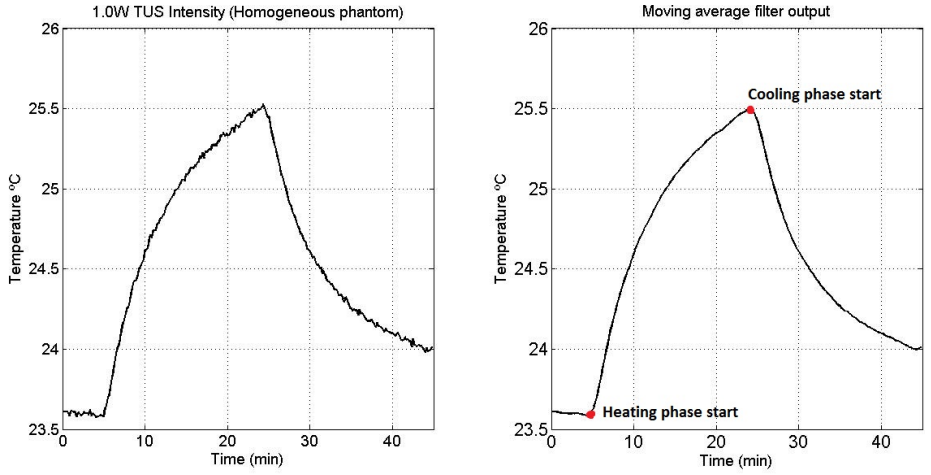

FIGURE 4.2: Noise reduction using an 8-point moving average filter. In the figure, data taken from the homogeneous phantom experiment, with a TUS intensity of $1.0W/cm^2$, exhibits noisy variations in the temperature.

From this last figure, it is obvious that the moving average filter employs a smoothing effect over the data, thus reducing the noise. As the name implies, the moving average filter operates by averaging a number of points from the input signal to produce each point in the output signal. It was employed a 8-point MA filter which, in equation form, is given by:

$$y_{MA}[i] = \frac{1}{8}\sum_{j=0}^{7} x[i+j] \tag{4.1}$$

The number of points used by the filter was obtained empirically, following a trial and error scheme.

Once the data as been smoothed, the starting points for each phase are found, using gradient methods. A high positive value of the derivative of the signal marks the beginning of the heating process, whereas a negative slope informs about the start of the cooling phase.

Temperature evolves with a fast gradient in the first initial moments, after the TUS device is applied. This effect is more noteworthy when the TUS intensity considered is $1.8W/cm^2$. However, despite this fast temperature evolution, the data acquisition

frequency was always constant through all the experiment. This fact is highlighted in Figure (4.3).

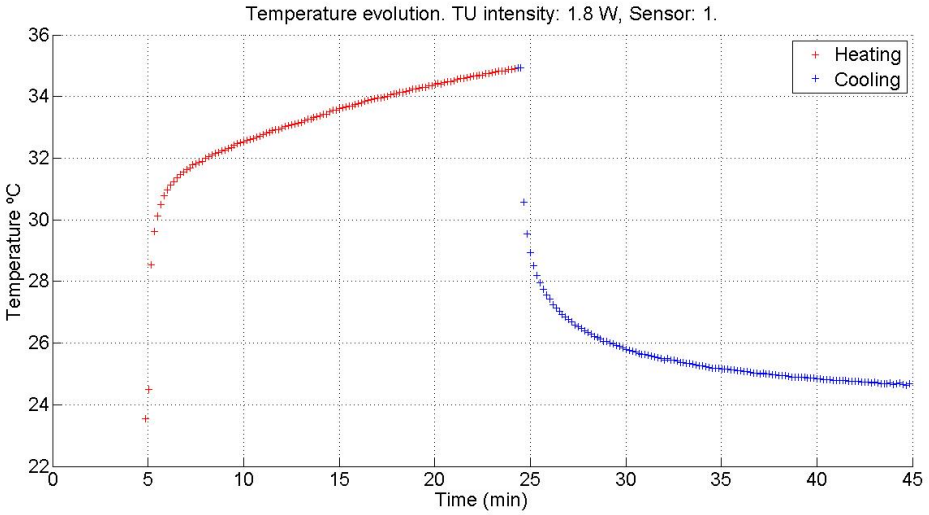

FIGURE 4.3: Data set collected from the homogeneous *phantom* experimental setup. TUS Intensity:$1.8W/cm^2$. Sensor: 1. The initial moments clearly exhibit a deficit of data, which can compromise the model learning potential over this region.

Data observation reveals a serious lack of information in two distinct moments: the initial moments right after the switch of the TUS device, present in the heating phase; and a second moment coincident with device shutdown, at the start of the cooling phase. This lack of knowledge at this areas, despite of being short in time, can compromise the model learning process and lead to erroneous predictions. This data deficit derived from the fixed data acquisition frequency, which should had been dynamic, providing a higher sampling rate in rapidly changing regions. Nevertheless, models should be capable of performing well through all the rest of the data set, since the data in those regions is capable of providing a higher quantity of knowledge about the process dynamics to the model. Furthermore, this two swift regions are translated to just about one or two data points, which corresponds to a fast transient state in the therapy of about 10 or 20 seconds, respectively, with a data acquisition sampling period of 10 seconds. Figure (4.4) provides an example of the effects of this lack of data. A disproportionated error occurs in the first moments of both phases (heating and cooling), in comparison with the predictions that follow, consequence of the lack of information in that area.

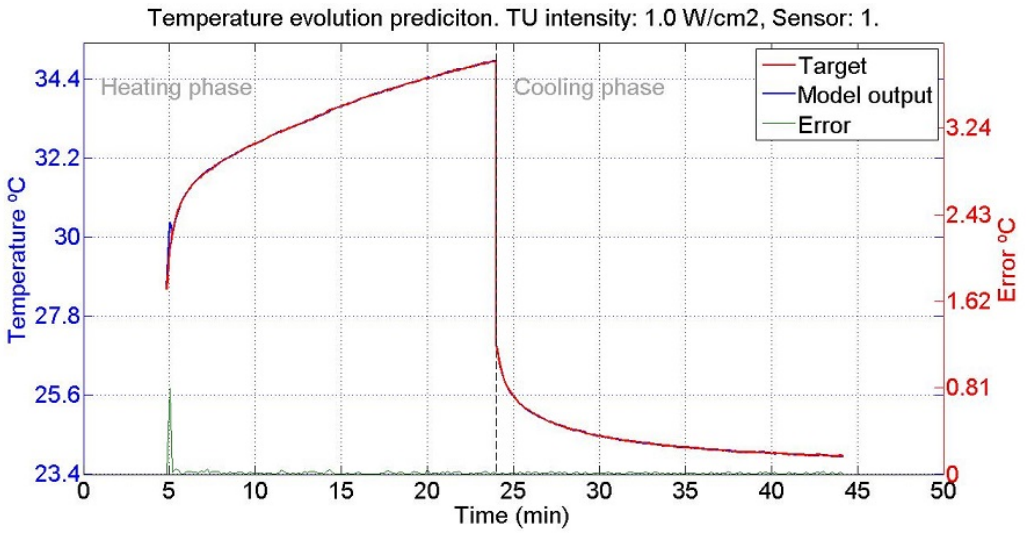

FIGURE 4.4: Results comparison between a model prediction and desired observed values. TUS Intensity:$1.0 W/cm^2$. Sensor: 1. The green line represents the absolute error evaluated through all the data set. The effects of the lack of data points are visible, in the initial moments when the temperature is rising rapidly.

This issue has to be addressed if satisfactory models are to be achieved. It is crucial to mask this lack of information in the data set, otherwise the models will not know how to behave in those two regions. Interpolation of the data points seems to be the most reasonable approach, since we assume to know the real behavior of the system in those areas. Switching the TUS device on causes a fast transient temperature rising. In contrast, switching off the device provokes a fast transient temperature downward.

Concerning the interpolation methods employed, it should be noticed that some parameters should be submitted to an optimization phase: number of data points to interpolate; sampling frequency of the interpolation frequency; and order of the interpolation. By adapting this parameters to each data set, the approximation error can be minimized. The scope of the interpolation method naturally covers the whole data set, otherwise the time relation between the data points is incoherent. The interpolation was made using cubic spline functions, belonging to C^2 continuous class. Using the data from Figure (4.3), which exhibits a high temperature gradient as function of time, the set was interpolated using a cubic spline, by sampling the function with a $1s$ period. The result is shown in Figure (4.5).

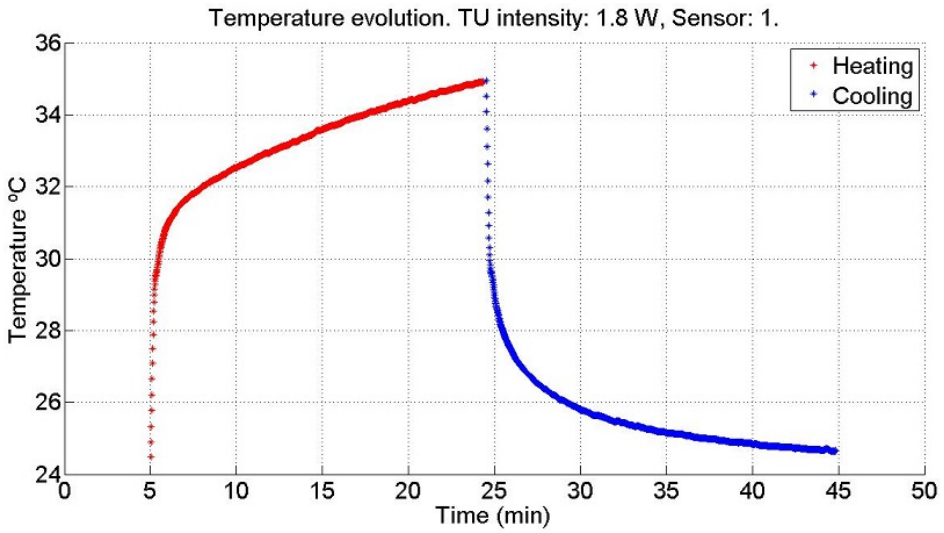

FIGURE 4.5: Interpolation result, applied to the data illustrated in Figure (4.4). The results exhibit a more robust and compact data set, assigning more knowledge about the process to the deficient areas.

Observing the interpolated data we can assume that the lack of information issue was addressed with realistic and approximate assumptions. By doing so, we expect models to be able to extract information in the transient temperature propagation phase.

Following this interpolation scheme, each two consecutive points, originally spaced by 10 seconds, are interpolated by a cubic spline, which then is sampled each 1 second. As a result the whole data set consists on temperature values separated by 1 second. Therefore each trial now consists of $N = 10 \times 270 = 2700$ temperature data points. Using this set to construct a model, reduces the prediction horizon from 10 seconds to 1 second, since we are considering one step ahead prediction. On the other hand, we are also increasing the volume of available data, which may induce an over-training effect in the models or cause them to learn the dynamics of the noise. Employing the early stopping method we expect to retract both of this undesirable effects. This prediction horizon is increased later to test the robustness of the forecasting networks.

4.2.2 Model validation

Concerning the model validation, discussed in Chapter (2), the selection of the data was not done in contiguous blocks. Instead, the subsets were constructed by randomly

choosing input/output pairs from the data set. With this, we expect to improve the generalization power of the models because the model is *tested* in the *test set*, using unseen data. This forces the generalization performance of the model to be evaluated, not just on a restricted area. The stopping criteria employed was the *early stopping method*, discussed on Chapter(2). Partitioning of the data set is usually performed using the following ratios:

- 70 % for **training (estimation) subset**.
- 20 % for **validation subset**.
- 10 % for **test set**.

If nothing is said about the division ratios, the partitioning ratios just mentioned are used. Accordingly the first subset is the training (estimation) set, which is used for computing the gradient and updating the network weights. The second subset is the validation set, which is not directly used to train the network. The error on the validation set is then monitored during the training process. The validation error normally decreases during the initial phase of training, as does the training set error. However, when the network begins to overfit the data, the error in the validation set typically begins to rise and hence the performance of the network begins to deteriorate. The network weights are saved when the validation set error is minimized.

In practice, the test set error is not used during training, but it is used to compare different models, in their ability to generalize. A heuristic for optimal neural network training says that, if the error on the test set reaches a minimum at a significantly different iteration number than the validation set error, this might indicate a poor division of the data set. Such a situation suggest that a revision on the split ratio should be considered.

The order by which the observations (input patterns), comprising the training set, are presented to the network, also requires discussion. Observations can be applied to the network following a *randomized arrangement* or by a natural *ordered arrangement*. The former is strongly preferable when an *on line learning* method is employed. However, as discussed in Chapter(2), when the whole data set is available, a batch (*offline*) training

is preferable. For this reason the data is presented to the network in a *ordered arrangement*, since shuffling the data has no effect in batch learning.

The number of past lagged observations lacks of theoretical result suggesting the best number of lags for a general nonlinear forecasting problem. However, this number should be minimized, which is justified by the need to construct simple, yet accurate models. Section (4.3) explores the most suitable number of past lagged observations concerning each model typology. If the performance is not satisfactory, this number will be revised. However a preference is naturally given to simpler models, targeting real time applications support. Models are constructed with this parameter being varied between $2 - 6$ lags.

4.2.3 Network designs, structure selection and algorithms

Most modelling schemes consider a given model structure or a fixed set of given model structures and estimate parameters in these structures. However due to the curse of dimensionality problem, exposed in Chapter (2), this is not feasible when the input space has a high dimension. In this work the ASMOD algorithm, Section (2.4.2.3), was intensively used to select the most proper model structure according to a performance-complexity balance. Therefore, an interactive model construction algorithm was employed, where B-spline networks are grown by iteratively refining of a very simple model. This process is illustrated in Figure (4.6).

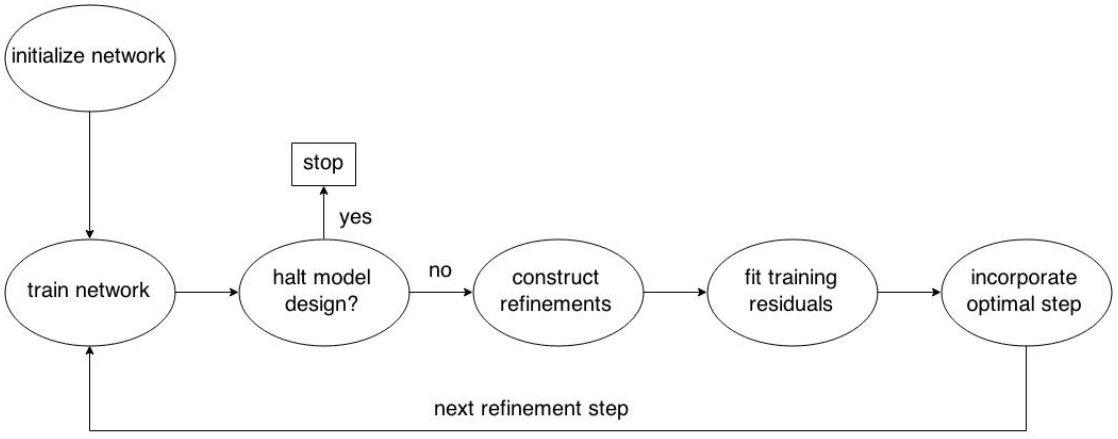

FIGURE 4.6: B-spline network design cycle.

Concerning this work, the design started from an initially empty network. However, the initial model might be biased to include a small number of relevant subnetworks. As the networks considered in this work are relatively simple, there is no need to bias the initial model. During the design cycle, this base model is gradually enhanced, by including new inputs, identifying cross-product terms and by representing each input in a better way, i.e, changing the knot sequence associated with a certain input axis. At each iteration, a number of possible ways by which the network can be made more flexible is assessed, i.e. the performance is calculated. The algorithm then chooses the optimum refinement step and applies it to the current model. B-spline networks evolved by the ASMOD algorithm hold a great synergy, because any enhancement to the current model, triggered by any refinement step, generates a more complex model, which is capable of *exactly* reproducing the previous model [28]. This is due to B-splines modelling capabilities robustness in respect to changes in the knot sequence. Nevertheless, the ASMOD algorithm does not take into account refinements concerning the number of the basis functions, otherwise the refined model may not be capable of *exactly* reproduce the previous one. Hence, the order of the splines which represent each univariate input must be determined before the learning begins.

After an initial model structure has been specified, model structures are identified according to the following algorithm:

Algorithm 1 General ASMOD algorithm.

1: Let the initial model structure be the current model structure.
2: Let $i = 0$, and let the *stop refinement* criterion be FALSE.
3: **while** the *stop refinement* criterion is FALSE, do: **do**
4: Let $i = i + 1$
5: From the current model structure generate a set $M = \{M_1, M_2, \ldots, M_n\}$ of candidate model structures grown and/or pruned.
6: Estimate the parameters in each model structure in M_i. Denote the estimated parameter vectors by $c_{i,j} \quad j = 1, \ldots, N_i$.
7: Compute a criterion function $g(M)$ for all candidate model structures.
8: Select the model structure with the smallest value of the criterion function $g(M)$ as the new current model structure. Denote this model structure by $\hat{M}_i$ and denote the corresponding parameter vector from c by $\hat{c}_i$.
9: Compute the stop refinement criterion.
10: **end while**
11: The identified model structure $\hat{M}$ is the one which gives the minimum value of $g(\hat{M}_j) \quad j = 1, \ldots, i$, and the identified model within this structure is given by the corresponding parameter vector $\hat{c}$.

Here i represents the iteration number in the refinement procedure. The last step performed by the algorithm selects from among the model structures the one that gives the minimum value of the performance criterion.

Concerning the input space limits, the inferior and superior knot values, for each input dimension, are always set to be the maximum and minimum value present in the training set, respectively.

Following the discussion on B-splines, in the context of neural networks, the predictive models are constructed using basis functions of relatively low order $1-4$. The motivation behind this choice comes from the quadratic nature observed in the processed data, when the two distinct phases are separated, as shown in Figure (4.7). Note that by observing the data we are able to introduce *a-priori* knowledge in the networking by biasing the order of the splines considered.

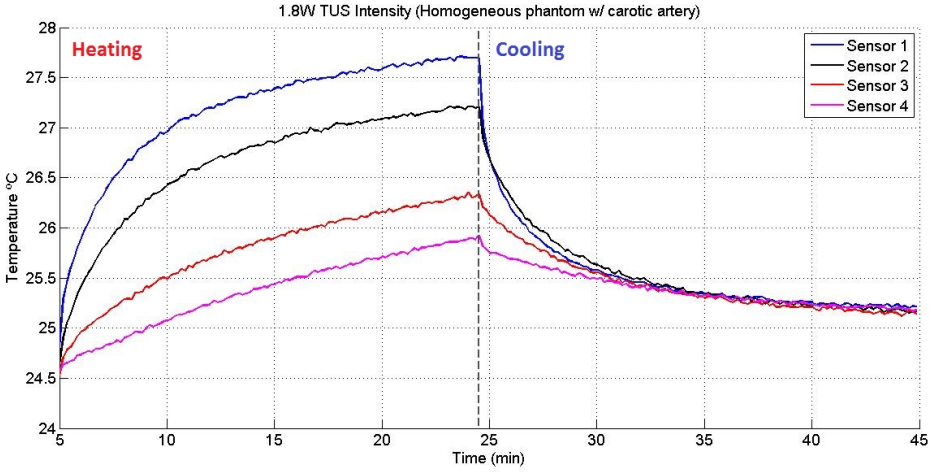

FIGURE 4.7: Data measured by all sensors in the homogeneous phantom with carotid artery experience ($1.5W/cm^2$).

Despite of the spatial location, the temperature propagation process seems to be governed by quadratic dynamics, both in the heating and cooling phase. This figure also suggests that the order of the dynamic is affected by the presence of an artificial artery present in the experiment environment. Being the network generalization one of the main objectives of the predictive model, it can be forced by using a network that is complex enough to provide an adequate fit. The larger network complexity, the more

complex the functions the network can create, therefore giving rise to the possibility of learning the noise present in the data. Using a simple network, it will not have enough power to overfit the data. A very complex model tends to have a small bias towards the process being modelled, thus not having a powerful learning ability to approximate the underlying data generating process. However the complexity provides the model a large variance, which measures the generalization capability. In contrast, a simple model may have a large bias but suffer from a small variance. The balance is inclined towards model bias in order to avoid noise interference in the core of the network.

4.2.4 Adapting the free parameters

The BSNN architecture admits a **fixed** middle layer, on which the input suffers a non-linear transformation performed by the basis functions. Then follows the **adaptive** layer, where the weights of the network are adapted by means of a linear optimization method. Concerning the adaptation, the *least square solution* (optimum weight values), was found using a *pseudo-inverse* solution, where the optimum weight vector is given by:

$$\hat{w} = (A^T A)^{-1} A^T t \tag{4.2}$$

Where A is a matrix of size (m x n), whose m^{th} row is composed of the transformed input vector for the m^{th} input, assuming the network is built using n basis functions in the second layer. t is the vector of desired outputs of length m and $\hat{w}$ is the optimal weight vector.

This method directly provides an analytic solution for the optimum weight values of the network, given a training set and a set of defined basis functions that transform the input vectors. Thus is used on this work.

4.3 Estimation models

Once fully characterized all of the network designing steps, the predictive networks are in conditions to be built. We shall consider one-step ahead predictions. As mentioned

before, a gradually increasing complexity approach was followed. We start by considering models for single-point and single-intensity estimation. Then gradually, the complexity of the models is increased towards multi-point and multi-intensity estimation. Three typologies of models were admitted:

- single-point, single-intensity (SPSI)
- single-point, multi-intensity (SPMI)
- multi-point, multi-intensity (MPMI)

4.3.1 Network design structures

This section explores the network structures applied to each model typology. The structure should be suitable for the network purpose and, as we are dealing with BSNN, the number of inputs must be forced to its minimum, admitting only crucial non-redundant inputs.

By considering more complex scenarios, more input variables must enter the network structure. This extra variables provide the indispensable information needed to guide the network predictions, thus the variables and their numerical representation must be properly chosen.

4.3.1.1 Single-point, single-intensity (SPSI)

A simple model that just admits a single point and a single TUS intensity, just needs to have as input m past temperature values, with m being the number of considered lags. This is the only information that the network needs to estimate the temperature one step ahead. We shall consider $2-5$ lags as previously discussed. The network structure used in this typology is shown in Figure (4.8).

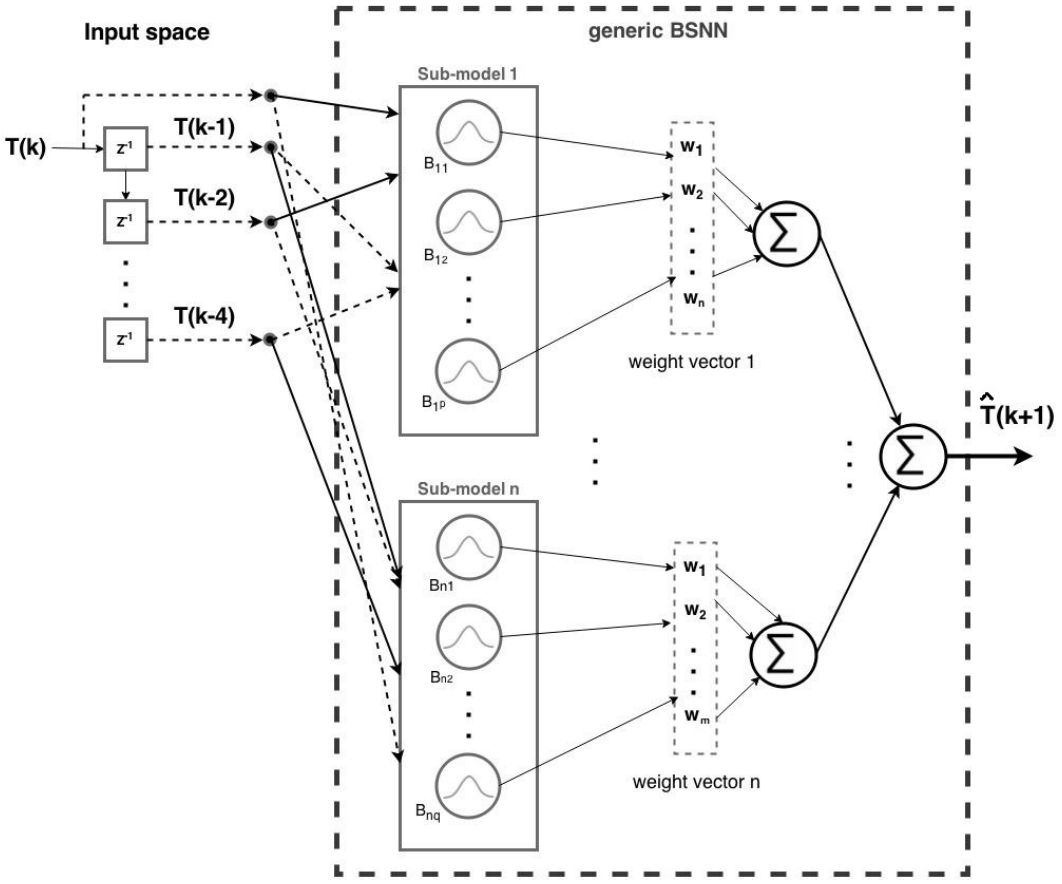

FIGURE 4.8: Network structure used in SPSI typology models.

The input space consists only of past temperature values $T(k)$, which are the only information needed by the network. z^{-1} is the unit delay operator. Figure (4.8) illustrates a generic BSNN structure, composed by additive sub-models. The connection arrangement between the input space and the next layer is merely demonstrative. Since this arrangement, as well as the decomposition in sub-models, is done by the ASMOD algorithm. Each input connects only to one sub-model. $\hat{T}(k+1)$ denotes the one step ahead

temperature value estimated by the network. As discussed before four models are to be created, each one with a different number of inputs lags, ranging from $2 - 5$.

4.3.1.2 Single-point, multi-intensity (SPMI)

The next step is to allow the network to accept different TUS intensities, albeit with all the action still happening at a single point. Since the spatial location is fixed, i.e. only one point is considered, the network does not need information about the spatial location because it simply does not vary. The model is just forecasting the temperature curve in a single point, thus spatial input to the network is unnecessary. However the same cannot be said about the TUS intensity, which is varied. As so, the network needs to have information about the intensity at each pattern. So the structure presented in Figure (4.8), is now extended to the one shown in Figure (4.9).

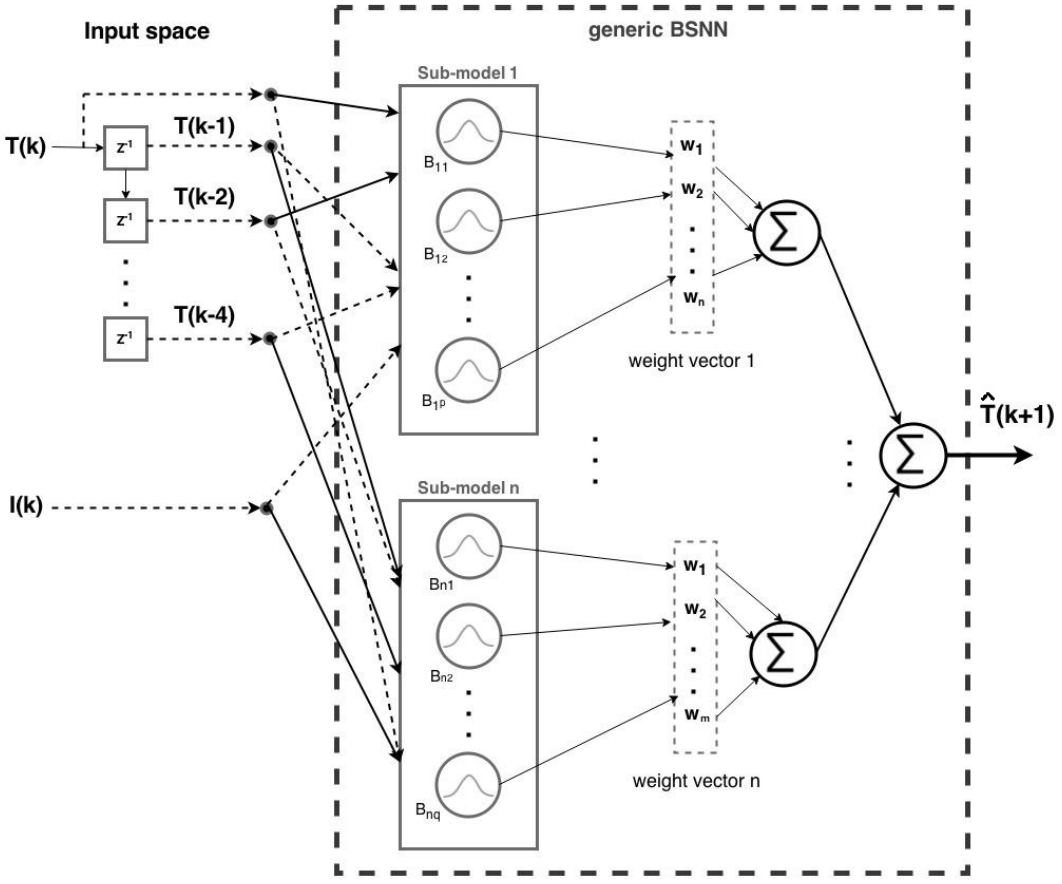

FIGURE 4.9: Network structure used in SPMI typology models.

This structure admits a SPMI model typology, provided by the additional input $I(k)$,

denoting the TUS beam intensity at instant k. The numerical representation is straightforward, a real number with one decimal point, admitting the following possible values:

$$I(k) = \{0.5; 1.0; 1.5; 1.8\} \tag{4.3}$$

Which correspond to TUS beam intensity values for each data is available. However this input is only constrained to be positive $I(k) \geq 0$ (since a beam intensity cannot be negative), any positive real number is admitted to this input.

4.3.1.3 Multi-point, multi-intensity (MPMI) ($1D$)

Towards a gradual more complex scenario, the model should now admit a dynamic $1-D$ spatial behaviour. Thus MPMI $1-D$ typology models assuredly require an additional input that provides information about the current spatial location of the input pattern. As so, the network structure is naturally extended to the one illustrated in Figure (4.10).

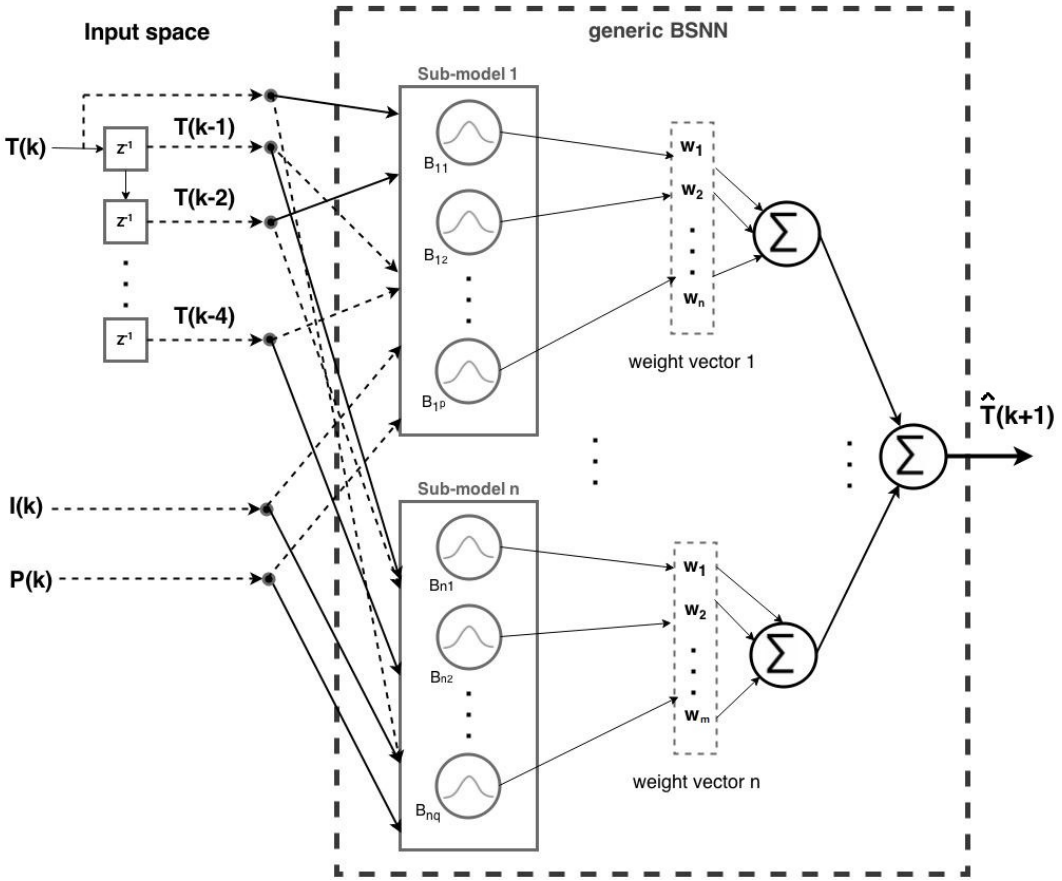

FIGURE 4.10: Network structure used in MPMI $1-D$ typology models.

$P(K)$ represents the spatial location of the input pattern, in the virtual line formed by the temperature sensors, Figure (3.4). In order to numerically represent this input, the location of the sensor closest to the TUS device (sensor 1 in Figure (3.4)) was taken as the the origin of the referential. A vertical line, centered in compliance with the center of the TUS face, is drawn orthogonally to the horizontal line formed by the sensors The angle θ illustrated in Figure (4.11) was applied as an input, in degrees.

Since the sensors are separated by $5mm$, the angle θ is trivially given by:

$$\theta_i = \arctan\left(\frac{D}{N_i * 5mm - 5mm}\right) \tag{4.4}$$

Where N_i is the number of the operating sensor and D is the distance from the line formed by the array of sensors which is parallel to the face of the transducer. D was set to $D = 50$mm, since this typology is trying to model just the $1-D$ space formed by the

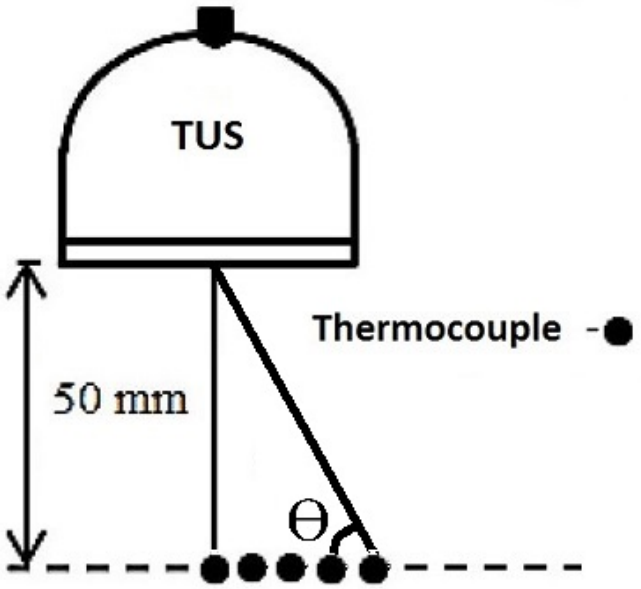

FIGURE 4.11: The angle θ formed between the operating sensor and the TUS central line was chosen to numerically represent the spatial location of the sensor in the $1-D$ line.

array of sensors. The $5mm$ is an offset to take the position of sensor 1 as the reference:

$$\arctan\left(\frac{D}{0}\right) = \arctan(\infty) = \pi \tag{4.5}$$

Therefore $P(k)$ admits the following set for the data available:

$$P(k) = \{90; 84.29; 78.69; 73.3; 68.2\} \tag{4.6}$$

Again it should be notice that the former set correspond to the numeric values that will be used during training, validation and test. However, this input is just constrained to be positive $P(k) \geq 0$.

The motivation for the angular numerical representation of this input to the network is justified by the resulting distinguished dynamics concerning $I(k)$ and $P(K)$. Note that $I(k)$, the TUS device beam intensity is a linear input, whereas $P(K)$ is governed by the dynamics of the hyperbolic tangent function. This two distinct dynamics allows the

network to distinguish more clearly the current operation point.

Gradually hardening the model forecasting task, provides an overview about how feasible it is to the predict temperature propagation using BSNNs, by measuring the scalability that the biomedical instrumentation system model shows, in terms of performance, when the environment is made more complex. If the systems responds well when the environment complexity is scaled, then we might assume that such a system, using a BSNN temperature predictive model, is achievable and suitable.

4.3.2 Adding noise

Motivated by the reasons briefed in Section (2.8.2.2), another experiment took place, consisting of deliberately adding Gaussian noise to the original data, in order to assess the model robustness and adequacy. We also expect the ambiguity between the ensemble to increase and hence minimize the ensemble generalization error (or test error).

It should also be emphasized that the data used in this work was collected in an invasive way, using thermocouples placed inside a *phantom*. Our biomedical instrumentation system should operate under the watch of a *data-driven* model. Thus, it is crucial to have available the largest amount of datapossible, in order to have represented a large set of temperature evolution dynamics, highly dependent on the unique characteristics present on the tissue region focused for hyperthermia/diathermia purposes. A high spatial resolution is desirable. Unfortunately invasive methods for temperature measurement are highly unpractical in real living tissues, plus a quantitative assessment of temperature is of extreme relevance for both patient security, and for the efficacy of the therapy, which requires a large number of sensors to be place into the tissue, increasing the impracticalities of this approach. Nevertheless, instead of directly measuring, it is possible to derive a temperature estimation method, by which the temperature is estimated indirectly, in a non-invasive way.

Ideally a thermal therapy system should have besides the heating source, a precise and effective time-spatial non-invasive temperature estimator. The estimator reliability must be maximized, so it can be used to provide an efficient therapy control, which would then result in the correct application of pre-defined heating patterns, preventing undesired effects and improving effectiveness. For hyperthermia/diathermia applications, the accepted maximum absolute error is of 0.5 $^oC/cm^3$ [51], that constitutes the gold standard resolution, only admittedly achieved by using magnetic resonance imaging (MRI) methods, a very expensive technology, when compared to other instrumentation. A lot of research has been made on non-invasive temperature estimation. Published works are based on electrical impedance tomography (EIT) [16], microwave thermometry [52], magnetic resonance imaging [53], and backscattered ultrasound (BSU) [54]. We leave a special reference to ultrasound based techniques, which consist on a very cheap technology when compared to MRI. Several methods have been reported, based on the extraction of temporal-echo shifts [54], frequency shifts [55], changes on the attenuation coefficient [56], and changes on the backscattered energy [57].

Lets assume a reliable (according to the MRI standard), practical, and non-invasive temperature estimation method is employed to obtain a large set of data, i.e. a big database of temperature curves, taken from a set of patients with diversified characteristics. If the patients are chosen in a way that the data represents knowledge over diverse types of tissues with diverse characteristics, then the only question to be answered is: can a model learn the dynamics of the process, masked under a noisy set of data (because the temperature estimator would certainly introduce error), and still have the power to generalize accordingly? We are admitting a reliable temperature estimation technique, i.e. noise magnitude is bounded by $0.5/cm^3$ (MRI gold standard). This would mean that we have available a large data base of *reliable* data. Thus one could argue that a sufficient amount of knowledge, about the temperature evolution on human tissues, is available. If so, equally reliable models can be constructed that make use of this abundant data. The assessment of this question can be partially provided by introducing additive noise to the set of data collected by the experimental setup exposed at Chapter(3). This noise represents the inherent error associated with the temperature estimation method. The noisy data can then be used to train and validate the network using the same methodologies described early in this chapter. By doing so we are challenging our system to

learn the dynamics of temperature space-time propagation in a non-invasive way. Again, when the environment complexity is scaled, the system should respond accordingly and perform well. If so, it might become practical to obtain a large data set to be used to train a reliable temperature predictive model.

Concerning the additive noise, we assume the error over a large set of temperature estimations can be considered as normally distributed, with zero mean $\mu = 0$ and a standard deviation $\sigma = 0.15$. Such a distribution is shown in Figure (4.12).

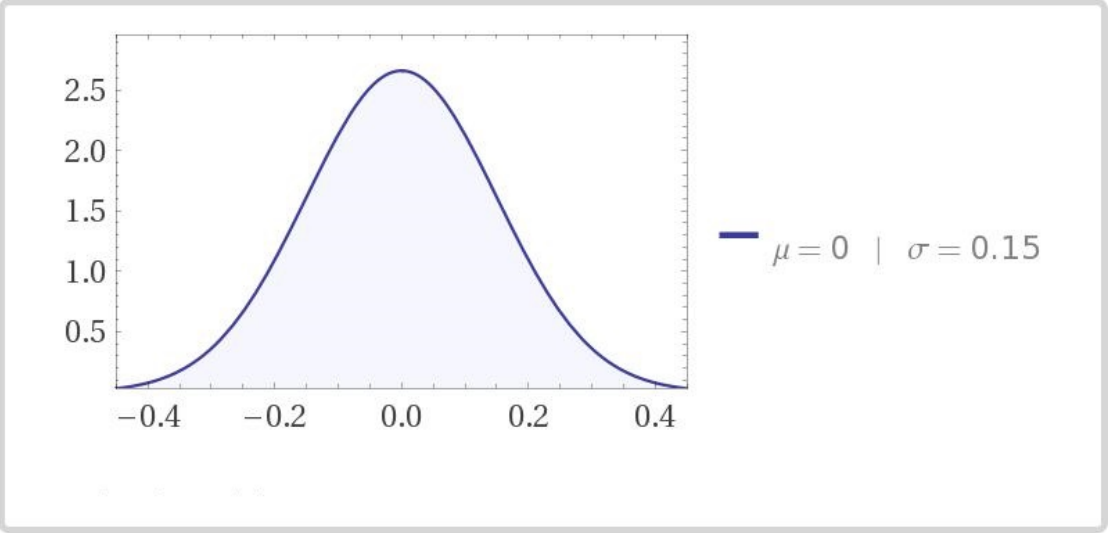

FIGURE 4.12: Gaussian distribution with $\mu = 0$ and $\sigma = 0.15$.

With a probability density function given by:

$$f(x, \mu, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} e^{-(x-\mu)^2/2\sigma^2} = 2.65962 e^{-22.2222x^2} \tag{4.7}$$

Therefore, the probability of the error being lower than 0.45 is $P(error < 0.45) = 99.7\%$.

Regarding the addition of the noise, the same methodology is *always* applied. The addition of Gaussian noise is made on a *point-to-point* basis. This means each temperature value in a curve is independently contaminated with a real number drawn from a Gaussian distribution, $e_i \sim \mathcal{N}(0, \sigma)$. Also it should be mentioned that every a noisy contaminated temperature curve is used to create and test a model, the noise addition

is repeated. It should be emphasized that all the additions are uncorrelated.

For exemplification purposes, Figures (4.13) and (4.14) show the temperature evolution of the homogeneous *phantom* experiment, with and without the addition of Gaussian noise, correspondingly.

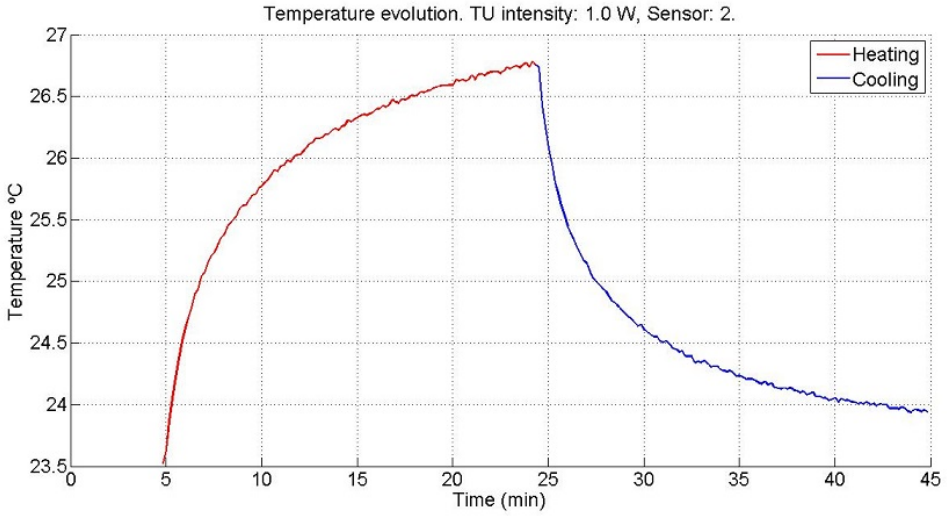

FIGURE 4.13: Temperature evolution measured by *sensor 2*, on the simple homogeneous *phantom* experiment ($1.0W/cm^2$). The plot illustrates the noise free version of the signal, in contrast with Figure (4.14), where Gaussian noise was added to the signal.

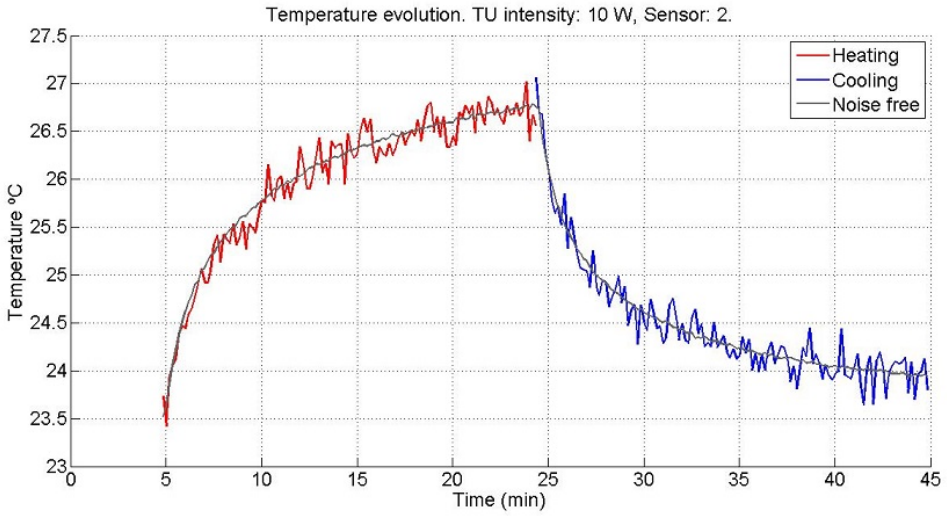

FIGURE 4.14: Gaussian noise was added to the previous signal, taken from a normal distribution with $\mu = 0$ and $\sigma = 0.15$.

The noisy version of signal naturally constitutes a more challenging task to the model learning process. By adequately designing the network we can minimize the trade off between ease of data acquisition and model correctness, since increasing the noise concurrently increases the probability of the noise dynamics being learned by the network. Design considerations must be aware of this fact, in order to create networks immune to noise. This addition gives rise to construction of models based on *simulated non-invasive estimations*. Furthermore, adding noise will force a disagreement among the individual networks that constitute that ensemble, thus one can expect uncorrelated individual errors, which means that high ambiguity levels are present in the ensemble.

In order to prevent the network of learning the noise dynamics, we need to limit their power. Ideally, the network should be designed with just enough function approximation power to learn the process dynamics. Thus, hindering the network from learning the noise dynamics can be achieved by imposing a limit in the orders of the splines that form the BSNN. Initially the maximum allowed order was set to four. However, as previously noted the high TUS intensity may induce fast abrupt temperature changes, which possibly need a higher order to be approximated. The maximum allowed oreder should be revised if such situation is encountered.

4.4 Modelling approaches

This section is reserved to expose the approaches taken when modelling the temperature evolution. We intend to characterize in detail the methods employed during the different experiments done in this work. All the approaches taken are independent of the model typology being considered. In a compact form, we propose to apply the following methods:

1. keep-the-best (KTB)
2. simple average ensemble
3. ensemble optimized with an evolutionary strategy
4. ensemble optimization with NDEO

It is intended to apply the four methods always two times: one using the original data; and a second time with the random Gaussian noise corrupted data set. In experiments using the original data, it's expected that the ensemble methods don't return satisfactory performance gains, when compared with KTB, due to the high correlation present among the networks. Randomization of the patterns that constitute the data sets (validation, training and test), should not be sufficient to achieve the desired levels of ambiguity. Nevertheless the results are shown to confirm or refute the expectations.

However using noisy data to train and validate the models one can expect a high increase in the ambiguity levels of the ensemble, since the addition of random noise to the data acts as an decorrelation agent in the ensemble. Once the desired ambiguity levels are achieved, the system can incur into network output combination that should decrease the generalization error as desired. By adding Gaussian noise to the data, the whole complete set of original noise-free data can be used in the test set, which is useful to assess the performance and robustness of the system. The four listed approaches are described next.

4.4.1 Keep-the-best (KTB)

Firstly the traditional KTB approach is considered. The architecture of the predictive system employing this simple approach is illustrated in Figure (4.15).

Four models are constructed for each experiment. Each network has a specific number of lags, ranging from $2 - 5$, for comparison purposes. The selection of *each* one of the four models follows the traditional KTB scheme, i.e. the model that best performs in the validation set is chosen. Creating networks with different number of input lags provides means of comparison between the best number of inputs to use and also acts as a decorrelation agent amid the ensemble, albeit most likely not sufficient to justify ensemble approaches when using the original data.

All the following approaches consider a network ensemble, hence more overhead has to be introduced. The simple architecture presented in Figure (4.15) has to be modified.

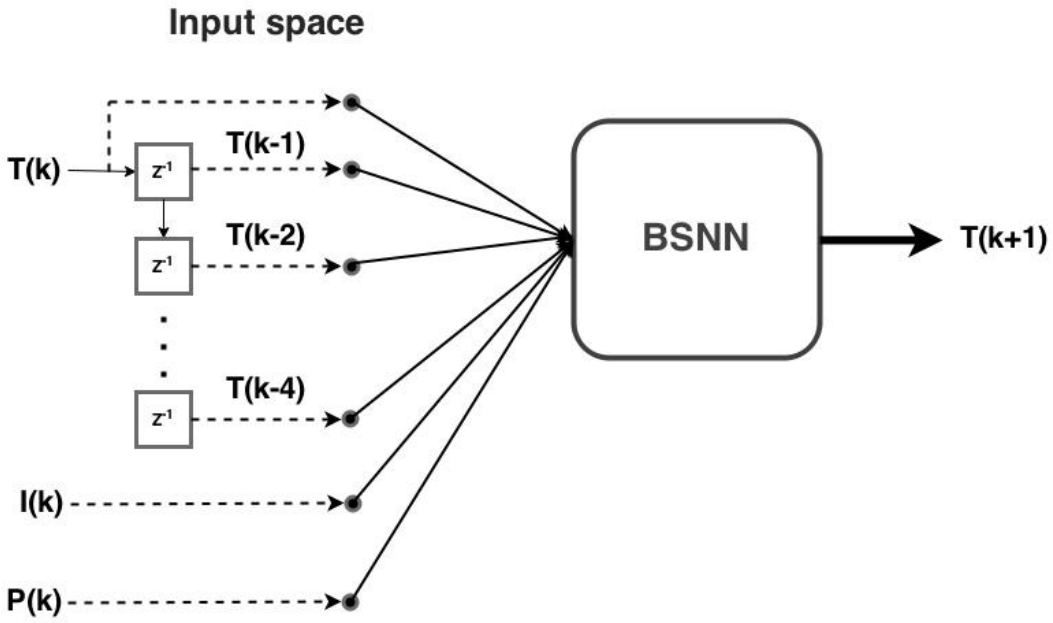

FIGURE 4.15: Predictive system architecture employing the traditional KTB method.

4.4.2 Simple average ensemble

In this approach the four models with distinct number of input lags are combined to forge a prediction. The architecture employed is illustrated in Figure ((4.16).

The final output consists in a weighted sum all the individual predictions. The output is given by:

$$T(k+1) = \frac{1}{N}\sum_{i=1}^{N} \omega_i T_i(k+1) \tag{4.8}$$

By considering a simple average scheme, each individual output is equally weighted:

$$\omega_i = \frac{1}{N} \quad for\, all\, i \tag{4.9}$$

This represents the most basic network ensemble scheme. Nevertheless if the models are sufficiently uncorrelated, this approach is expected to outperform the KTB approach. Hereafter the weight vector can be optimized for better results. A evolutionary strategy was chosen to optimize the weight vector.

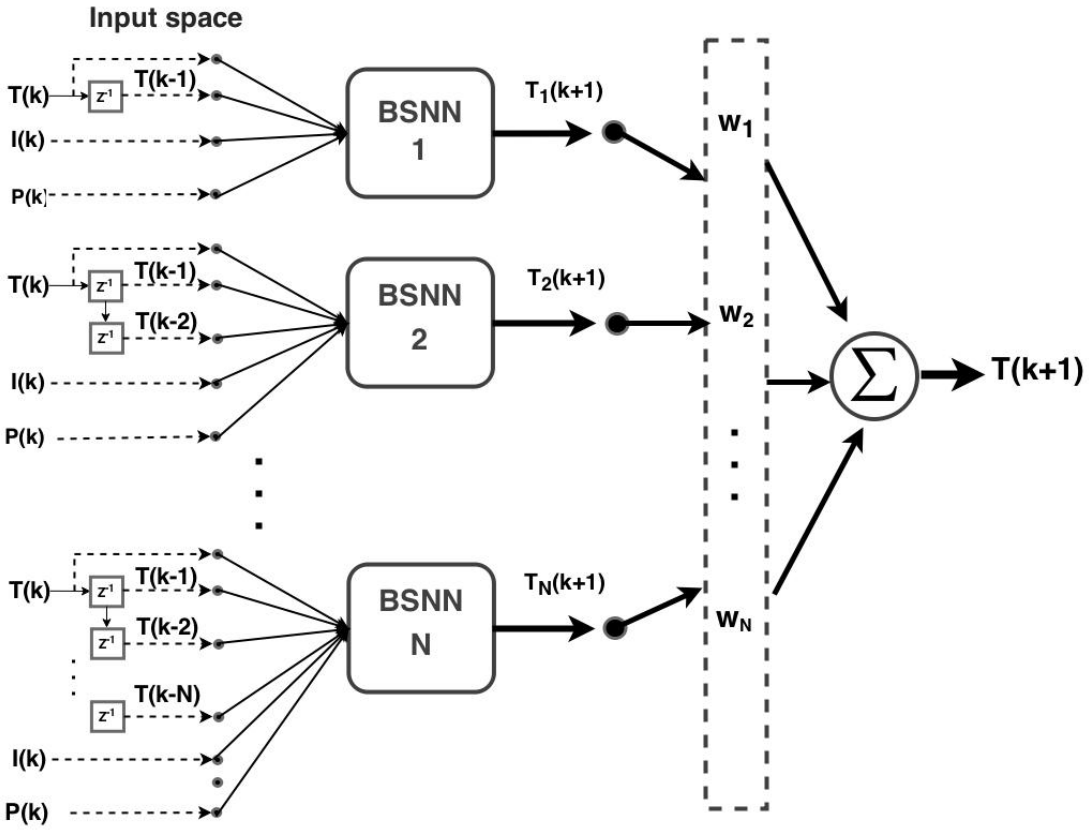

FIGURE 4.16: Predictive system architecture employing a neural network ensemble.

4.4.3 Ensemble optimized (ES)

A standard ES with uncorrelated mutations and n step sizes [58] was employed to optimize the weight vector, present in Figure (4.16). The mutation operator in ES is based in a Gaussian distribution, characterized by two parameters: the mean μ and the standard deviation σ. Then the basic mutation is done applying the following change:

$$x_i^{t+1} = x_i^t + N(\mu, \sigma) \tag{4.10}$$

With $N(\mu, \sigma)$ given by:

$$N(\mu, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} e^{-(x-\mu)^2/2\sigma^2} \tag{4.11}$$

The mean was set to $\mu = 0$ and the standard deviation to $\sigma = 1$. x is a n dimensional vector to be optimized ($n = 4$ for optimization process). An uncorrelated mutation

optimization with n step sizes, admits the following mutation mechanism:

$$\sigma_i^{t+1} = \sigma_i^t e^{\tau_1 N(0,1)+\tau_2 N_i(0,1)} \tag{4.12}$$

$$x_i^{t+1} = x_i^t + \sigma_i N_i(0,1) \tag{4.13}$$

τ_1 is the *global learning rate*, given by:

$$\tau_1 = \frac{1}{2\sqrt{n}} \tag{4.14}$$

And τ_2 the individual learning rate:

$$\tau_2 = \frac{1}{\sqrt{2\sqrt{n}}} \tag{4.15}$$

τ_1, τ_2 and σ_i form the strategy parameters. σ_i is mutated as in equation (4.12).

Initially the step sizes σ_i are initialized to 0.001 and the weights are randomly initialized between 0.1 and 0.4. The justification for the initial step size is empirical following the tests that were done. The weight initialization intended to not assign *a priori* a wide preference to a network, since the weight vector has to respect the following constrains:

$$x_i \leq 1 \quad for\, for\, all\, i \tag{4.16}$$

and

$$\sum_{i=1}^{n} x_i = 1 \tag{4.17}$$

The population size was set to $\mu = 15$ and the offspring size to $\lambda = 6\mu = 90$. The optimization process runs for 100 generations and the best individual (weight vector) is chosen.

The minimization is done just in the validation set used to train the model. Hence the cost function forces the minimization of the MSE in the validation set.

$$\hat{\omega} = arg \quad min(MSE_v) \tag{4.18}$$

Lastly, the NDEO approach was employed, which we discuss next.

4.4.4 Neural dynamic ensemble optimization (NDEO)

As discussed in Section (2.8.3) we propose a new paradigm for combining the ensemble outputs. This solution employs a neural network as the optimization mechanism. The proposed two layered architecture is shown in Figure (2.8.3).

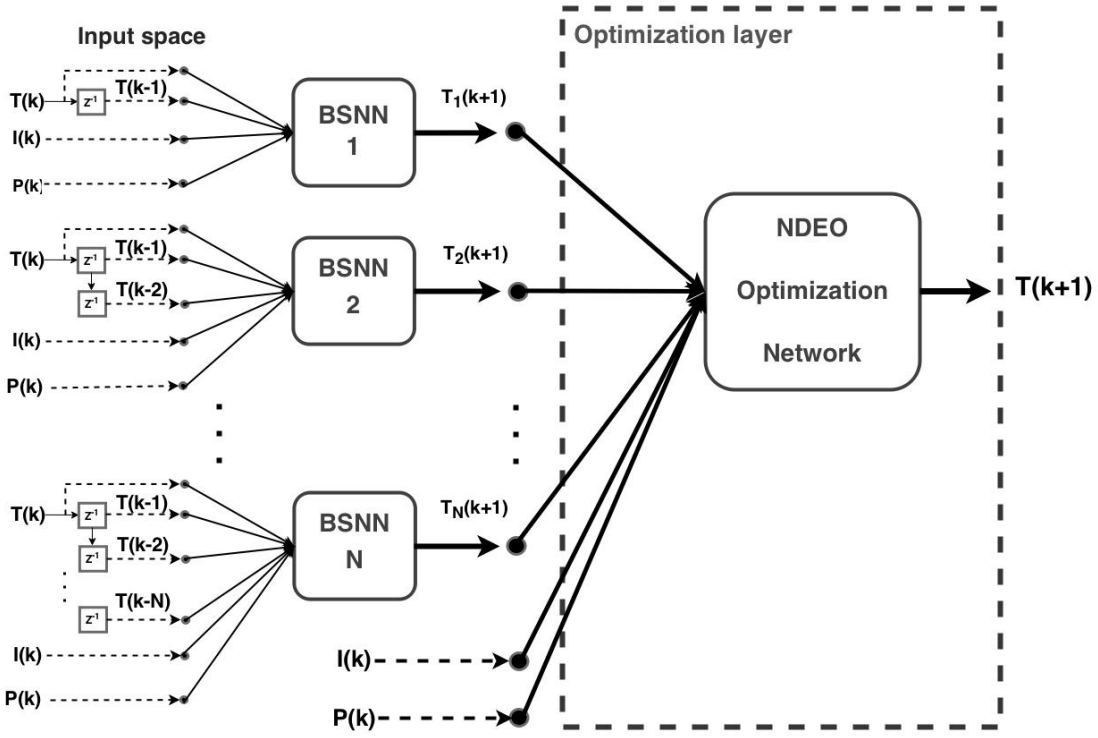

FIGURE 4.17: Predictive system architecture employing NDEO.

The second layer adds intelligence to the combination mechanism, thus we expect a considerable return in performance gains. The additional inputs to the second layer network ($I(k)$ and $P(K)$) are model typology dependant. This extra knowledge about the current input pattern should allow the network to dynamically optimize the arrangement of outputs, enhancing the best particularities learned in each one of the models, having in consideration the current intensity-spatial information.

The large additional overhead introduced by this optimization mechanism is just justified in ensembles with small correlations levels. Otherwise, this overhead does not return

benefits in performance, i.e not an advantageous trade-off. Nevertheless, by adding random Gaussian noise the original data set before training the networks, we expect to create suitable opportunities to explore and assess the NDEO approach.

All the results and discussions are exposed in Chapter (5).

5

Results and discussion

5.1 Introduction

The data gathered from the experimental setups, presented in Chapter (3), was subsequently used to derive predictive models, whose performance we expose at this chapter. The methodologies followed to design and construct the models are stated in Chapter (4). As referred, four model typologies were considered (SISP, SIMP, MIMP), and the complexity of the environments being modelled was increased in a gradual fashion. Following the methodology detailed in Section (4.2.1), two models are considered for each *operation environment* and model typology, one for each thermal phase, heating and cooling. A division is made between the results of each phase, but are presented together. An operation environment is characterized by its model typology as well as the operating points (data) used to construct the models.

Concerning model performance descriptors, the MSE and MSRE, Section (2.5), were employed as error performance criteria. We present the error evaluated through all the data sets. The maximum absolute error in all the the subsets is also indicated. Model complexity was assessed calculating the linear weight norm (LWN), which is an important criterion, since it provides a descriptor with information about the model specialization to the training data, which we highly want to avoid. Furthermore, the balance between these two indicators was done by using the bayesian information criterion (BIC), that takes in consideration both error and complexity indicators. This criteria are detailed in Section (2.5). The stopping reasoning is also pointed: n for normal stopping; and e for a stop due to early stopping method. Normal stopping is triggered after the achievement of satisfactory performance conditions. Listing all the model descriptors in a more compact form:

- bayesian information criterion BIC.
- mean square error in the training set MSE.
- mean square relative error in the training set $MSRE$.
- mean square error in the validation set MSE_v.
- mean square relative error in the validation set $MSRE_v$.
- mean square error in the test set MSE_t.
- mean square relative error in the test set $MSRE_t$.
- maximum absolute error though all the data set M_{ae}.
- linear weight norm LWN.
- training stopping reason SR.

Graphical illustrations contrasting the original data with the one predicted by the models are thoroughly shown, together with tables constructed from the performance figures. We shall consider predictive models based on **invasive measurements**, as well as predictive models based on **simulated non-invasive estimations**, by considering additive Gaussian noise over the original data, and repeating the modeling activity and compare the results, i.e. assess the robustness of the models to corrupted data.

5.2 Single-point single-intensity (SPSI)

We begin the presentation of the obtained results by first considering the simplest model typology, SPSI. By being the simplest environment, good performance indicators are expected in this section. The experimental arrangements, detailed in Chapter (3), were subject to the applied models, taking into account the various spatial points and intensities applied. A reference is made to Section (4.3.1.1), where the general network designs employed for this typology are briefed.

This section makes use of the data acquired using the experimental setup presented in Section (**??**). Just a few SPSI predictive models results are presented with graphical illustration support, due to the extensive number of models applied.e

Model environment: TUS Intensity ($1.0W/cm^2$), Sensor (1)

We begin by considering a model to predict the temperature evolution experienced at the closest sensor to the TUS device, Figure (3.4), with a TUS beam intensity of $1.0W/cm^2$. Using the original data shown in Figure (5.1), 70% was used for training, 20% for validation and 10% for testing. The pattern splitting was random as opposed to contiguous.

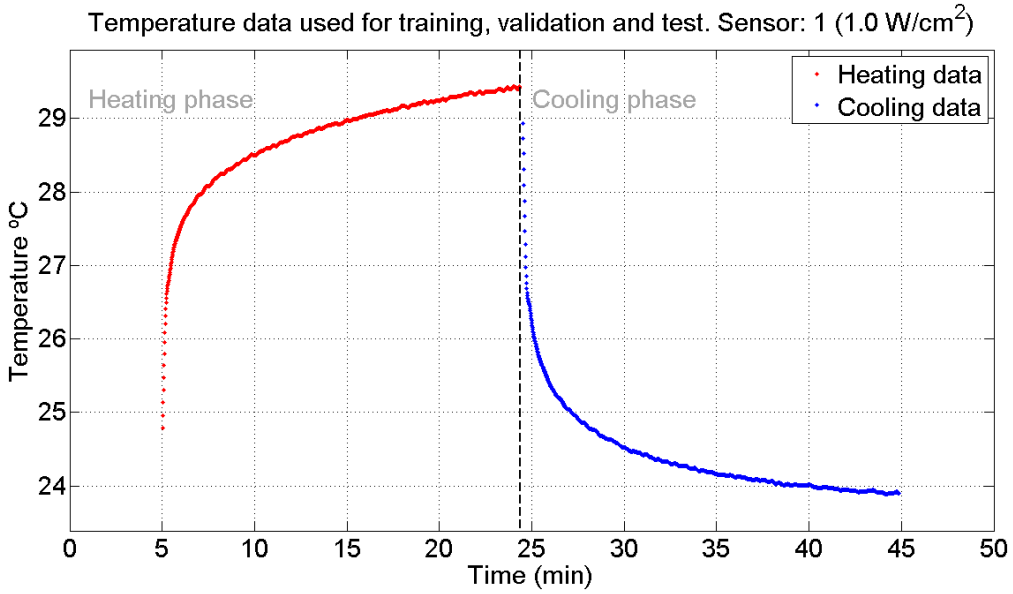

FIGURE 5.1: Unaltered data set used for SPSI model training, validation and test. Collected from the *homogeneous phantom* experimental setup. TUS intensity: $1.0 W/cm^2$. Sensor 1.

As discussed in the previous chapters, four different models were constructed, each one with a different number of input lags, ranging from $2 - 5$. Table (5.1) presents the performance descriptors calculated for each one of the four models considered.

TABLE 5.1: Performance descriptors obtained concerning models with different number of input lags. The models presented were selected using the KTB approach. SPSI, Sensor 1 (1.0 W/cm^2). Used data: uncorrupted

Model 1(2 lags)		
Criterion	Heating	Cooling
BIC	-9248	-8826
MSE	2.4055e-05	2.4349e-05
$MSRE$	8.8628e-07	9.5225e-07
MSE_v	7.1255e-04	1.4595e-03
$MSRE_v$	2.5453e-05	5.6024e-05
MSE_t	9.1783e-04	3.4404e-04
$MSRE_t$	3.2229e-05	1.3986e-05
M_{ae}	0.2890	0.3603
LWN	7	16
SR	e	e

Model 2(3 lags)		
Criterion	Heating	Cooling
BIC	-8859	-7861
MSE	2.1325e-05	9.7163e-05
$MSRE$	7.7766e-07	3.6108e-06
MSE_v	2.1609e-04	6.9510e-04
$MSRE_v$	7.5944e-06	2.7347e-05
MSE_t	5.8617e-04	3.2529e-04
$MSRE_t$	2.0520e-05	1.3312e-05
M_{ae}	0.0878	0.3194
LWN	7	7
SR	e	e

Model 3(4 lags)		
Criterion	Heating	Cooling
BIC	-8449	-8398
MSE	3.6447e-05	5.4516e-05
$MSRE$	1.3700e-06	2.0085e-06
MSE_v	4.0105e-04	4.8289e-04
$MSRE_v$	1.4156e-05	1.9033e-05
MSE_t	3.8191e-04	5.1135e-04
$MSRE_t$	1.3326e-05	2.0758e-05
M_{ae}	0.1650	0.1836
LWN	11	11
SR	e	e

Model 4(5 lags)		
Criterion	Heating	Cooling
BIC	-9595	-10000
MSE	8.6044e-06	1.1740e-05
$MSRE$	3.0504e-07	4.6166e-07
MSE_v	1.3824e-04	5.2748e-04
$MSRE_v$	4.8266e-06	2.0850e-05
MSE_t	8.7054e-04	4.1142e-04
$MSRE_t$	3.0425e-05	1.6794e-05
M_{ae}	0.1207	0.2195
LWN	11	12
SR	e	e

Despite the distinct number of input lags all the models learned the process dynamics and performed well in the generalization (test) set. The training phase was always

stopped due to early stopping method. The models exhibit maximum absolute errors M_{ae} always below 0.35 ^{o}C, which constitutes a small error in a process that shows a 3 ^{o}C variation in less then 20 seconds. Decision about which model to plot took into consideration the best average (heating and cooling) validation error . Therefore, Figure (5.1) shows the behaviour of model 2 in training, validation and data set.

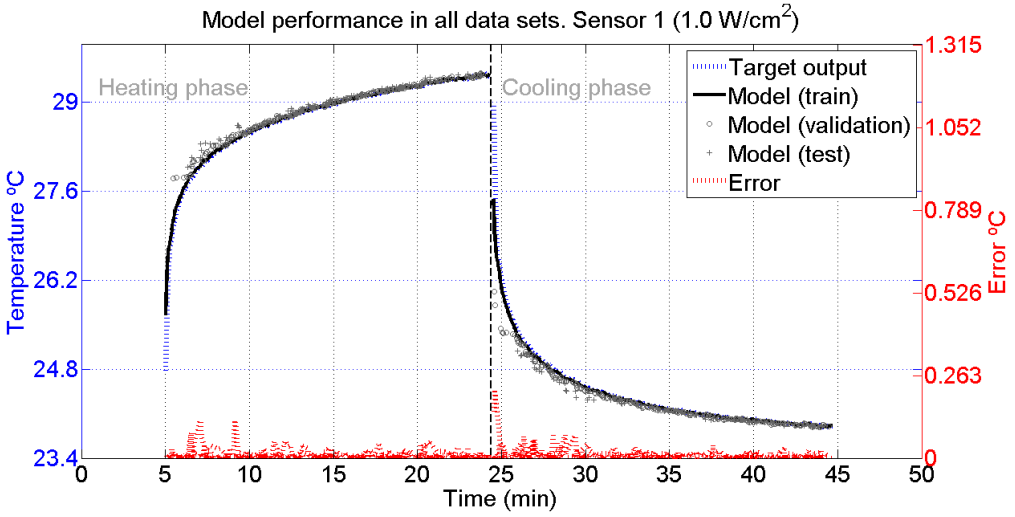

FIGURE 5.2: Behaviour of model 2 through the whole data set, selected using the KTB approach. The blue line represents the desired behaviour and the model's training output is given by the black line. The error line is red, circle and cross markers represents the model's validation and test output respectively. SISP,*homogeneous phantom* experimental setup. TUS intensity: $1.0 W/cm^2$. Sensor 1. Used data: uncorrupted

Observing the model output, it is clear that the network learned the process dynamics and follows the output.

Following the construction of the four models, the ensemble approaches were employed and assessed. Table (5.2) shows the results obtained for all the methods applied. On average the generalization error was improved, in both heating and cooling phases, albeit not substantially. These results are better appreciated observing Table (5.3), which compares the error obtained in the test set from all approaches applied. When the generalization error comparison between the paradigms is performed, three figures are assessed: in the heating phase; in the cooling phase; and a last which takes the simple average between the two phases (*Average comparison*).

TABLE 5.2: Performance comparison between all methodologies employed (KTB and ensemble methods). SPSI, Sensor 1 (1.0 W/cm^2). Used data: uncorrupted

Model 1	Heating			Cooling		
Criterion	MSE_v	MSE_t	M_{ae}	MSE_v	MSE_t	M_{ae}
KTB	7.1255e-04	9.1783e-04	0.2890	1.4595e-03	3.4404e-04	0.3603
Ensemble (SA)	6.8421e-04	8.3069e-04	0.2900	1.2982e-03	3.0665e-04	0.3413
Ensemble optimized (ES)	7.1179e-04	8.3775e-04	0.2900	1.3059e-03	2.9457e-04	0.3413
NDEO	6.9808e-04	8.5217e-04	0.2900	1.1479e-03	3.0232e-04	0.3407

Model 2	Heating			Cooling		
Criterion	MSE_v	MSE_t	M_{ae}	MSE_v	MSE_t	M_{ae}
KTB	2.1609e-04	5.8617e-04	0.0878	6.9510e-04	3.2529e-04	0.3194
Ensemble (SA)	2.0145e-04	5.3277e-04	0.0847	6.6896e-04	2.9166e-04	0.3189
Ensemble optimized (ES)	2.0779e-04	5.2247e-04	0.0864	6.7441e-04	2.9574e-04	0.3142
NDEO	2.2283e-04	5.9415e-04	0.0810	6.6396e-04	3.0266e-04	0.3131

Model 3	Heating			Cooling		
Criterion	MSE_v	MSE_t	M_{ae}	MSE_v	MSE_t	M_{ae}
KTB	4.0105e-04	3.8191e-04	0.1650	4.8289e-04	5.1135e-04	0.1836
Ensemble (SA)	5.1782e-04	4.0567e-04	0.2278	5.2533e-04	5.0317e-04	0.1868
Ensemble optimized (ES)	4.1018e-04	3.9166e-04	0.1670	4.7881e-04	5.1068e-04	0.1834
NDEO	4.0102e-04	3.8182e-04	0.1650	4.6776e-04	5.0624e-04	0.1805

Model 4	Heating			Cooling		
Criterion	MSE_v	MSE_t	M_{ae}	MSE_v	MSE_t	M_{ae}
KTB	1.3824e-04	8.7054e-04	0.1207	5.2748e-04	4.1142e-04	0.2195
Ensemble (SA)	1.4380e-04	7.0261e-04	0.1275	5.9516e-04	4.2069e-04	0.2481
Ensemble optimized (ES)	1.4987e-04	6.6091e-04	0.1191	5.3924e-04	4.0676e-04	0.2326
NDEO	1.4163e-04	7.7495e-04	0.1198	5.2096e-04	4.0461e-04	0.2234

Observing the generalization error comparison table one can conclude the generalization performance of the ensemble was enhanced by a small amount, with the *Ensemble optimized (ES)* exhibiting consistent improvement results. The NDEO approach suffers from deficient ambiguity levels to justify the overhead introduced in the system. The results obtained applying this method are worst then one obtained with the simple average (SA) ensemble.

TABLE 5.3: Generalization error obtained with the ensemble approaches in comparison with the KTB model selection. SPSI, Sensor 1 (1.0 W/cm^2). Used data: uncorrupted. Notice that a negative value means mitigation of the performance, i.e. the performance got worst.

Model 1 (2 Lags)	**Heating**	**Cooling**	**Average comparison**
Ensemble (SA)	9.49 %	10.87 %	10.18 %
Ensemble optimized (ES)	8.72 %	14.38 %	11.55 %
NDEO	7.15 %	12.13 %	9.64 %

Model 2 (3 Lags)	**Heating**	**Cooling**	**Average comparison**
Ensemble (SA)	9.11 %	10.34 %	9.72 %
Ensemble optimized (ES)	10.87 %	9.09 %	9.98 %
NDEO	-1.36 %	6.96 %	2.80 %

Model 3 (4 Lags)	**Heating**	**Cooling**	**Average comparison**
Ensemble (SA)	-6.22 %	1.60 %	-2.31 %
Ensemble optimized (ES)	-2.55 %	0.13 %	-1.21 %
NDEO	0.02 %	1.00 %	0.51 %

Model 4 (5 Lags)	**Heating**	**Cooling**	**Average comparison**
Ensemble (SA)	19.29 %	-2.25 %	8.52 %
Ensemble optimized (ES)	24.08 %	1.13 %	12.61 %
NDEO	10.98 %	1.66 %	6.32 %

The results just presented made use of the uncorrupted data to construct and validate the model. Then Gaussian noise is added to the data, where the addition is repeated four independent times, one for each of the four models. More specifically this means

that the uncorrupted data set was taken as the base set. Then this set was corrupted four independent times. Each time admits a independent *point-to-point* contamination, as detailed in Section(4.3.2). So each point x_i, belonging to the original data set S, is corrupted by doing:

$$x_i = x_i + e_i \sim \mathcal{N}(0, \sigma)$$

This process is repeated four times to create four independent corrupted data sets C_j, with $j = 1, 2, 3, 4$, that are used to construct four models. By using this scheme, both the training and validation sets employed in the construction of each network are completely differently, which hopefully will translate in highly uncorrelated models as pretended. An illustrative data set used to train, validate and test the next four models is plotted in Figure (5.3). This time the models are to be constructed using corrupted data.

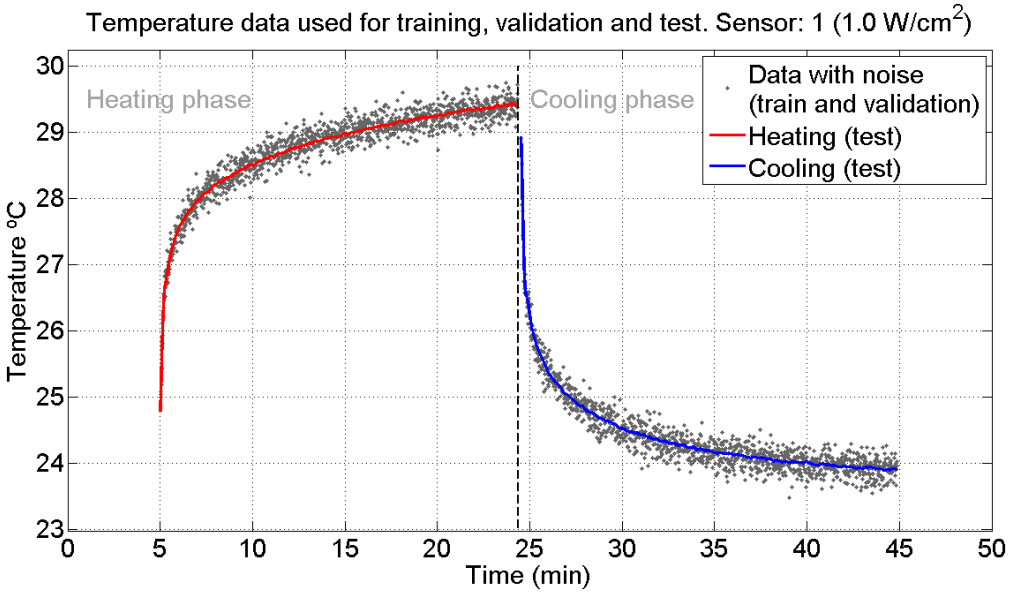

FIGURE 5.3: Data after the addition of random Gaussian noise. The noisy data is used for training and validation. The unaltered, noise free data is used to test the model. Collected from the *homogeneous phantom* experimental setup. TUS intensity: $1.0 W/cm^2$. Sensor 1.

A 70/30 data splitting scheme was applied for the training and validation set respectively. Concerning the data test, the whole unaltered, uncorrupted data set was used to assess the generalization ability model. This provides a complete and robust assessment of the network, which ideally should not learn the noise dynamics and hence perform well in the test set. Table (5.4) exposes the performance figures calculated.

TABLE 5.4: Performance descriptors obtained concerning models with different number of input lags. The models presented were selected using the KTB approach. SPSI, Sensor 1 (1.0 W/cm^2). SPSI, Sensor 1 (1.0 W/cm^2). Used data: corrupted

Model 1(2 lags)		
Criterion	Heating	Cooling
BIC	-2748	-2935
MSE	4.1291e-02	3.7153e-02
$MSRE$	1.4387e-03	1.5170e-03
MSE_v	4.7449e-02	4.0248e-02
$MSRE_v$	1.6614e-03	1.6436e-03
MSE_t	2.5462e-03	2.2165e-03
$MSRE_t$	8.9262e-05	9.0777e-05
M_{ae}	0.2501	0.1254
LWN	7	9
SR	n	n

Model 2(3 lags)		
Criterion	Heating	Cooling
BIC	-2913	-3062
MSE	3.2146e-02	3.3540e-02
$MSRE$	1.1191e-03	1.3708e-03
MSE_v	3.2683e-02	2.7664e-02
$MSRE_v$	1.1342e-03	1.1335e-03
MSE_t	9.8486e-04	8.8478e-04
$MSRE_t$	3.5378e-05	3.5631e-05
M_{ae}	0.3924	0.1903
LWN	10	12
SR	n	n

Model 3(4 lags)		
Criterion	Heating	Cooling
BIC	-2937	-3048
MSE	2.8745e-02	2.9837e-02
$MSRE$	9.9828e-04	1.2236e-03
MSE_v	3.0809e-02	3.1125e-02
$MSRE_v$	1.0739e-03	1.2737e-03
MSE_t	7.0740e-04	1.1927e-03
$MSRE_t$	2.5086e-05	4.6561e-05
M_{ae}	0.2227	0.3632
LWN	16	16
SR	n	n

Model 4(5 lags)		
Criterion	Heating	Cooling
BIC	-3005	-3188
MSE	2.7276e-02	2.8771e-02
$MSRE$	9.4835e-04	1.1812e-03
MSE_v	2.8545e-02	2.7145e-02
$MSRE_v$	9.9020e-04	1.1123e-03
MSE_t	5.7017e-04	7.7317e-04
$MSRE_t$	2.0246e-05	3.1034e-05
M_{ae}	0.1481	0.1489
LWN	20	20
SR	n	n

The maximum absolute error M_{ae} was calculated admitting only the errors obtained in the test set. Naturally the errors obtained concerning a noisy pattern do not provide information about the network's fulfillment of learning the process dynamics, hence we decided to just consider the test set to derive the M_{ae}. By doing so we enhance the value of this performance criterion. The model's behaviour over the test set is plotted in Figure (5.4). The chosen model this time was the one who performed better in the test set (model 4).

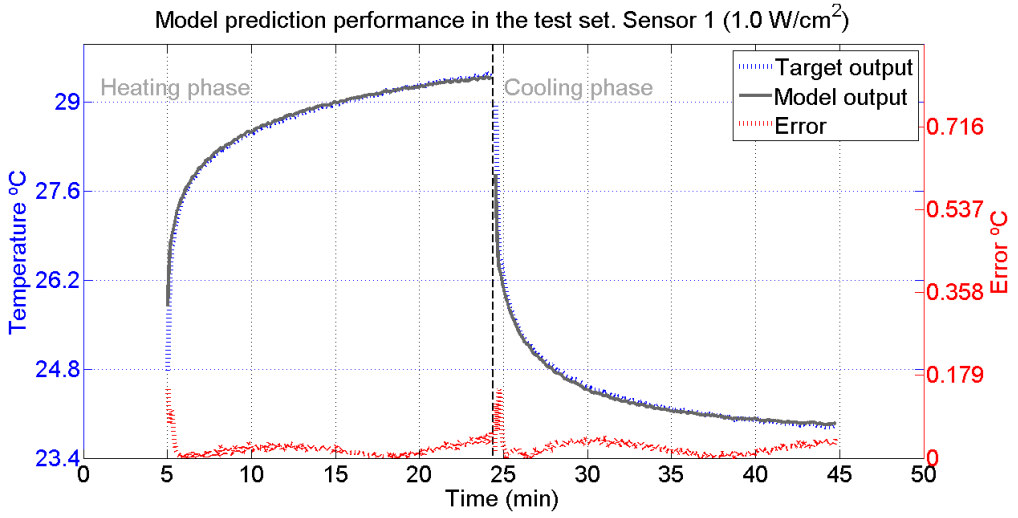

FIGURE 5.4: Behaviour of model 4 in the test set, selected using the KTB approach. The blue line represents the desired behaviour and the model's test output is given by the black line. The error line is red. SISP, *homogeneous phantom* experimental setup. TUS intensity: $1.0 W/cm^2$. Sensor 1. Used data: corrupted

The generalization error was always kept under 0.2 ^oC, a result that suggests a consistency. The network has proven to have learned the process dynamics in a robust way, i.e. immune to the additive noise.

We now present the results obtained regarding the network ensembles approaches. Table (5.5) assesses the performance of the network ensemble methods, and Table (5.6) provides a comparison between the later methods with the traditional KTB model selection scheme.

The SA and ES ensemble schemes exhibit a very unstable behaviour, as can be observed by the comparison table. This two methods just seem to justify if the models are trained with just two input lags, and their performance deteriorates as the number of lags is increased. However, as expected, the NDEO method outperforms all of the other approaches in a consistent way. With highly uncorrelated models, due to the presence of noise, the NDEO approach combines the individual outputs in an *active*, proficient way.

Concerning SPSI typology models, five additional environments were considered, corresponding to five different operating points:

TABLE 5.5: Performance comparison between all methodologies employed. SPSI, Sensor 1 (1.0 W/cm^2). Used data: corrupted

Model 1	Heating			Cooling		
Criterion	MSE_v	MSE_t	M_{ae}	MSE_v	MSE_t	M_{ae}
KTB	4.7449e-02	2.5462e-03	0.2501	4.0248e-02	2.2165e-03	0.1254
Ensemble (SA)	2.9488e-02	1.1367e-03	0.3183	2.6414e-02	1.0905e-03	0.1739
Ensemble optimized (ES)	2.9119e-02	1.1797e-03	0.3183	2.5781e-02	7.4995e-04	0.1739
NDEO	3.0180e-02	2.2287e-04	0.3130	2.6448e-02	8.7589e-04	0.4370

Model 2	Heating			Cooling		
Criterion	MSE_v	MSE_t	M_{ae}	MSE_v	MSE_t	M_{ae}
KTB	3.2683e-02	9.8486e-04	0.3924	2.7664e-02	8.8478e-04	0.1903
Ensemble (SA)	2.4170e-02	1.0837e-03	0.3183	2.2605e-02	1.0888e-03	0.1739
Ensemble optimized (ES)	2.4177e-02	1.1267e-03	0.3183	2.1939e-02	7.4800e-04	0.1739
NDEO	2.3749e-02	2.4885e-04	0.3375	2.3391e-02	2.0987e-04	0.2134

Model 3	Heating			Cooling		
Criterion	MSE_v	MSE_t	M_{ae}	MSE_v	MSE_t	M_{ae}
KTB	3.0809e-02	7.0740e-04	0.2227	3.1125e-02	1.1927e-03	0.3632
Ensemble (SA)	2.2382e-02	9.9712e-04	0.2476	2.5498e-02	1.0837e-03	0.1739
Ensemble optimized (ES)	2.2276e-02	1.0402e-03	0.2476	2.5450e-02	7.4259e-04	0.1739
NDEO	2.1932e-02	4.1210e-04	0.2488	2.5755e-02	3.2995e-04	0.1457

Model 4	Heating			Cooling		
Criterion	MSE_v	MSE_t	M_{ae}	MSE_v	MSE_t	M_{ae}
KTB	2.8545e-02	5.7017e-04	0.1481	2.7145e-02	7.7317e-04	0.1489
Ensemble (SA)	2.2637e-02	9.4498e-04	0.1870	2.1699e-02	1.0597e-03	0.1317
Ensemble optimized (ES)	2.2537e-02	9.8807e-04	0.2056	2.0871e-02	7.1835e-04	0.1073
NDEO	2.2970e-02	5.3478e-04	0.1063	2.1659e-02	2.8762e-04	0.1057

- $1.8W/cm^2$, Sensor 1
- $0.5W/cm^2$, Sensor 2
- $1.5W/cm^2$, Sensor 3
- $1.8W/cm^2$, Sensor 4
- $1.0W/cm^2$, Sensor 5

The results obtained admitting the listed operating points are exposed at Appendix(C) so we can move forward to more complex scenarios, modeling different operating environments.

TABLE 5.6: Generalization error obtained with the ensemble approaches in comparison with the KTB model selection. SPSI, Sensor 1 (1.0 W/cm^2). Used data: corrupted

Model 1 (2 Lags)	**Heating**	**Cooling**	**Average comparison**
Ensemble (SA)	55.36 %	50.80 %	53.08 %
Ensemble optimized (ES)	53.67 %	66.16 %	59.92 %
NDEO	91.25 %	60.48 %	75.87 %

Model 2 (3 Lags)	**Heating**	**Cooling**	**Average comparison**
Ensemble (SA)	-10.04 %	-23.06 %	-16.55 %
Ensemble optimized (ES)	-14.40 %	15.46 %	0.53 %
NDEO	74.73 %	76.28 %	75.51 %

Model 3 (4 Lags)	**Heating**	**Cooling**	**Average comparison**
Ensemble (SA)	-40.96 %	9.14 %	-15.91 %
Ensemble optimized (ES)	-47.04 %	37.74 %	-4.65 %
NDEO	41.74 %	72.34 %	57.04 %

Model 4 (5 Lags)	**Heating**	**Cooling**	**Average comparison**
Ensemble (SA)	-65.74 %	-37.06 %	-51.40 %
Ensemble optimized (ES)	-73.29 %	7.09 %	-33.10 %
NDEO	6.21 %	62.80 %	34.50 %

5.2.0.1 Results discussion

This section dealt with the simplest model typology considered, a single intensity applied on a single spatial point. Nevertheless considering KTB approach, the models revealed to be consistently accurate, with maximum absolutes errors far below the MRI standard of 0.5 ^oC. These results were independent of the addition of Gaussian noise to the training and validation set which exposes the robustness of the models constructed.

Fast, abrupt temperature evolutions consist in the biggest challenges that the predictive networks need to deal with. The sensors closer to the TUS transducer face experienced this sharp variations of temperature, while the further sensors presented smooth, slow temperature variations, dynamics that the models can predict without a significant effort.

Concerning the ensembles approaches, the SA method proved itself to be an unreliable method due to an unstable behaviour, exhibiting oscillatory performances figures. Regarding the ES approach, considerable performance gains were achieved applying this

method. It demonstrated the particularity of enhancing the predictions even in conditions with low ambiguity, i.e. models trained with the original data. Constructing the models with noisy data, i.e. forcing uncorrelated models, the NDEO approach outperformed all the others combination schemes, due to the addition of a second intelligent layer in the ensemble system architecture.

The predictive models complexity is now increased to admit multiple intensities, at a single spatial point. The results are presented in the following section.

5.3 Single-point multi-intensity (SPMI)

This section presents the results obtained with respect to SPMI models which, in addition to the previous model typology, now admit multiple intensities at a single spatial point, thus allowing for the creation of more complex models. The models were structured following the strategies explained in Section(4.3.1.2).

Due to the extensive amount of data needed to be present in this work, we chose to just deal in this section with models trained with noisy corrupted data. We assume that if satisfactory models can be built with this corrupted data, the same condition is true if they are trained with the original data, since the former consists of a more complex task.

Concerning the ensembles approaches we are interested to ascertain if their predictive performance enhancement is scaled as the models complexity is increased, i.e. if it is feasible to assume that performance gains obtained with the ensemble approaches in simple environments would exhibit a comparable counter part when the environment complexity is scalable.

Model environment: Sensor (1), all TUS intensities

Analogous to the previous model typology, sensor 1 is firstly considered, the more closest one to the TUS device face. This time the models were built using all of the data available

for this point: $0.5W/cm^2$, $1.0W/cm^2$, $1.5W/cm^2$ and $1.8W/cm^2$. Figure (5.5) depicts this data after the addition of Gaussian noise.

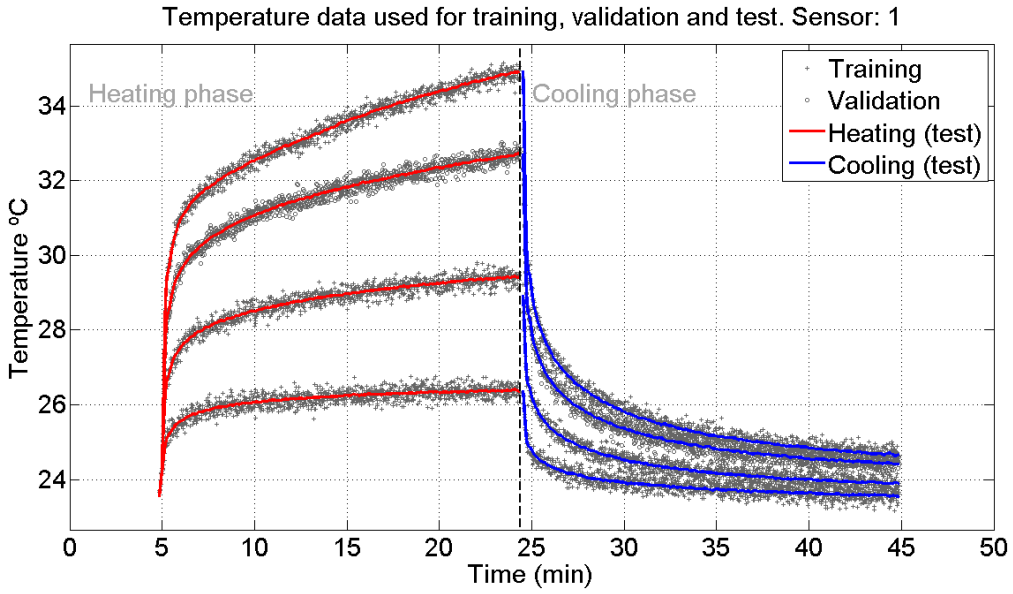

FIGURE 5.5: Data after the addition of random Gaussian noise. The noisy data is used for training and validation. The unaltered, noise free data is used to test the model. Collected from the *homogeneous phantom* experimental setup. The top curve corresponds to the strongest intensity. TUS intensity (from the shortest curve to the tallest curve): $0.5W/cm^2$, $1.0W/cm^2$, $1.5W/cm^2$ and $1.8W/cm^2$. Sensor 1.

Data collected at $0.5W/cm^2$, $1.0W/cm^2$ and $1.8W/cm^2$ was used for training and data collected at $1.5W/cm^2$ was used for validation. The whole complete noise free data set was used for testing. By doing so, the test set is capable of providing a complete assessment of the model. Four models with different number of input lags were built and the performance descriptions are calculated in Table (5.7).

Regarding the SPMI model typology, the data collected at sensor 1 undoubtedly represents the most difficult learning task, due to the more accentuated abrupt temperature changes. At $1.8W/cm^2$ the temperature rises from below 24 ^oC to 31 ^oC in a matter of seconds. Nevertheless, the models proved to have learned the process dynamics, even by just having available noisy data. Model 1 and 2 just needed information from two and three past observations (input lags), respectively, to predict the temperature evolution through all the test set (complete unaltered data set) within a maximum absolute error threshold of 0.5 ^oC. The unaltered data of $1.5W/cm^2$ was also part of the test set, providing an evaluation about the *interpolation* ability available in the networks. This is true since their parameters have not been adapted in compliance this data ($1.5W/cm^2$).

TABLE 5.7: Performance descriptors obtained concerning models with different number of input lags. The models presented were selected using the KTB approach. SPMI (Sensor 1). Used data: corrupted.

Model 1(2 lags)		
Criterion	Heating	Cooling
BIC	-11670	-12347
MSE	3.4370e-02	3.3361e-02
$MSRE$	1.1796e-03	1.3571e-03
MSE_v	3.2934e-02	3.0774e-02
$MSRE_v$	1.0475e-03	1.2247e-03
MSE_t	8.3782e-04	1.0303e-03
$MSRE_t$	2.9493e-05	4.0939e-05
M_{ae}	0.2997	0.2227
LWN	11	12
SR	e	n

Model 2(3 lags)		
Criterion	Heating	Cooling
BIC	-12068	-12794
MSE	3.0288e-02	2.9174e-02
$MSRE$	1.0440e-03	1.1880e-03
MSE_v	3.4142e-02	3.1575e-02
$MSRE_v$	1.0877e-03	1.2512e-03
MSE_t	1.1708e-03	7.2136e-04
$MSRE_t$	4.1357e-05	2.8036e-05
M_{ae}	0.4559	0.2891
LWN	15	16
SR	e	e

Model 3(4 lags)		
Criterion	Heating	Cooling
BIC	-12200	-13104
MSE	2.8805e-02	2.6549e-02
$MSRE$	9.9212e-04	1.0790e-03
MSE_v	3.0790e-02	2.9344e-02
$MSRE_v$	9.8403e-04	1.1640e-03
MSE_t	1.5028e-03	6.8125e-04
$MSRE_t$	5.2962e-05	2.5494e-05
M_{ae}	0.5575	0.3049
LWN	19	19
SR	e	e

Model 4(5 lags)		
Criterion	Heating	Cooling
BIC	-12164	-12882
MSE	2.9016e-02	2.7875e-02
$MSRE$	1.0019e-03	1.1316e-03
MSE_v	2.7253e-02	2.8519e-02
$MSRE_v$	8.6871e-04	1.1302e-03
MSE_t	1.4677e-03	6.8226e-04
$MSRE_t$	5.1484e-05	2.5676e-05
M_{ae}	0.5740	0.3552
LWN	19	23
SR	e	e

Model 1 with just two input lags had the best performance once it has predicted the temperature evolution, concerning all intensities, within an error threshold of 0.3 ^oC.Figures (5.6), (5.7), (5.8) and (5.9) depict the actual output of model 3 through all the test set.

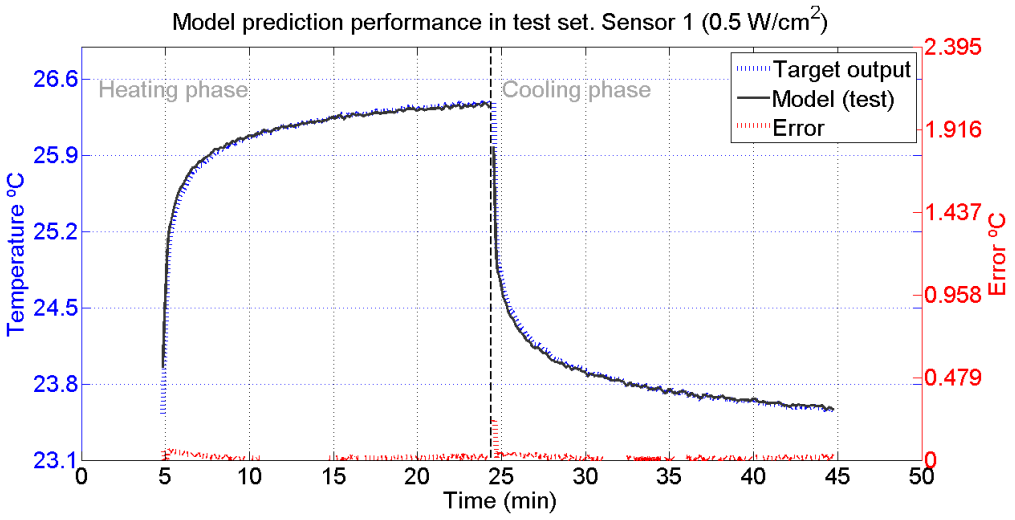

FIGURE 5.6: Behaviour of model 3 in the test set (Sensor 1 $0.5W/cm^2$), selected using the KTB approach. The blue line represents the desired behaviour and the model's test output is given by the black line. The error line is red. SIMP, *homogeneous phantom* experimental setup. Used data: corrupted.

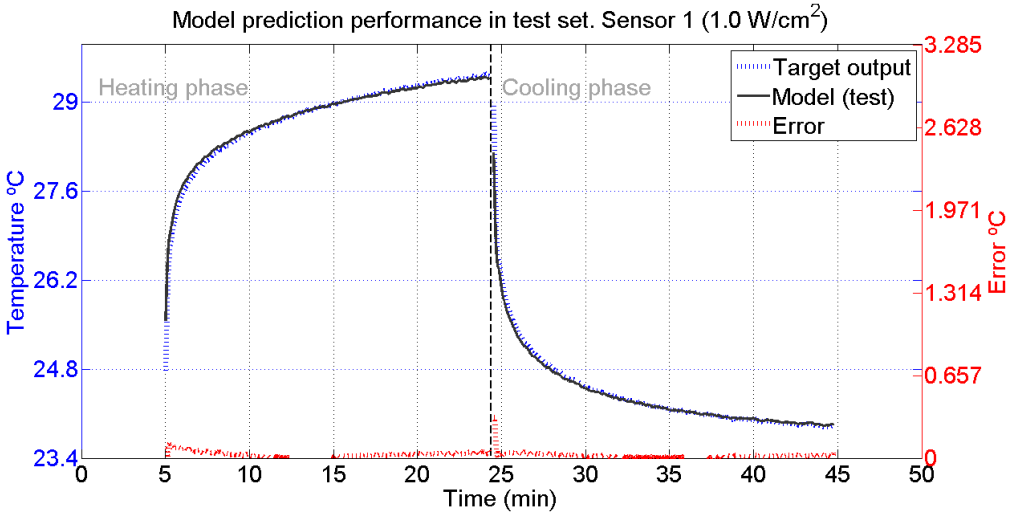

FIGURE 5.7: Behaviour of model 3 in the test set (Sensor 1 $1.0W/cm^2$), selected using the KTB approach. The blue line represents the desired behaviour and the model's test output is given by the black line. The error line is red. SIMP, *homogeneous phantom* experimental setup. Used data: corrupted.

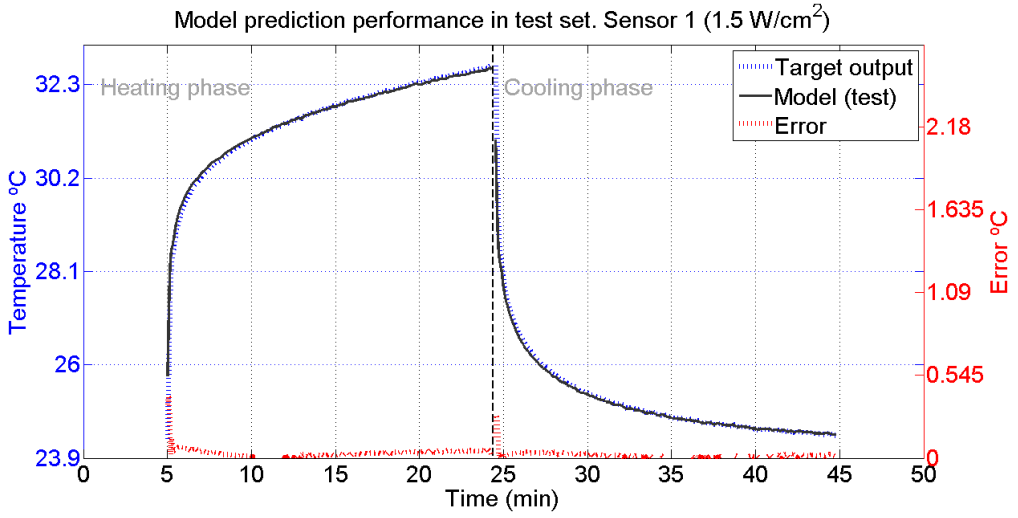

FIGURE 5.8: Behaviour of model 3 in the test set (Sensor 1 $1.5W/cm^2$), selected using the KTB approach. The blue line represents the desired behaviour and the model's test output is given by the black line. The error line is red. SIMP, *homogeneous phantom* experimental setup. Used data: corrupted.

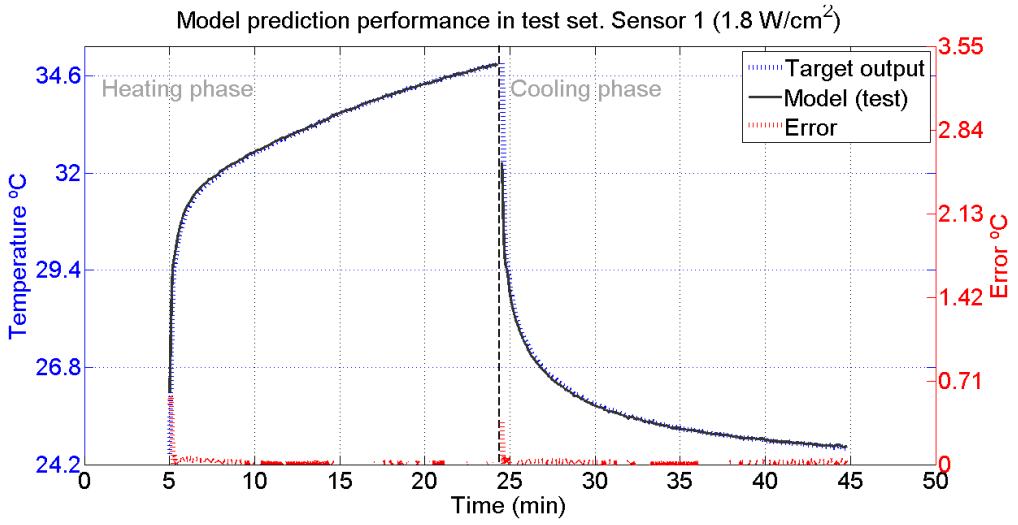

FIGURE 5.9: Behaviour of model 3 in the test set (Sensor 1 $1.8W/cm^2$), selected using the KTB approach. The blue line represents the desired behaviour and the model's test output is given by the black line. The error line is red. SIMP, *homogeneous phantom* experimental setup. Used data: corrupted.

Assuredly the process dynamics were learned by the model since it performed well through all the test set, covering all TUS intensities experienced at sensor 1. Table (5.8) depicts the ensemble approaches performance while Table (5.9) compares the generalization ability with the KTB scheme.

TABLE 5.8: Performance comparison between all methodologies employed. SPMI (Sensor 1). Used data: corrupted.

Model 1	Heating			Cooling		
Criterion	MSE_v	MSE_t	M_{ae}	MSE_v	MSE_t	M_{ae}
KTB	3.2934e-02	8.3782e-04	0.2997	3.0774e-02	1.0303e-03	0.2227
Ensemble (SA)	2.3309e-02	1.2209e-03	0.4718	2.2110e-02	7.1536e-04	0.2851
Ensemble optimized (ES)	2.2391e-02	6.3881e-04	0.3462	2.1911e-02	8.3465e-04	0.2963
NDEO	2.2202e-02	1.2414e-04	0.4842	2.1852e-02	2.0014e-04	0.3915

Model 2	Heating			Cooling		
Criterion	MSE_v	MSE_t	M_{ae}	MSE_v	MSE_t	M_{ae}
KTB	3.4142e-02	1.1708e-03	0.4559	3.1575e-02	7.2136e-04	0.2891
Ensemble (SA)	2.6831e-02	1.2145e-03	0.4718	2.4366e-02	6.9943e-04	0.2851
Ensemble optimized (ES)	2.5813e-02	6.3188e-04	0.3462	2.4207e-02	8.1881e-04	0.2963
NDEO	2.5244e-02	3.5702e-04	0.1141	2.4025e-02	1.5968e-04	0.1473

Model 3	Heating			Cooling		
Criterion	MSE_v	MSE_t	M_{ae}	MSE_v	MSE_t	M_{ae}
KTB	3.0790e-02	1.5028e-03	0.5575	2.9344e-02	6.8125e-04	0.3049
Ensemble (SA)	2.4268e-02	1.1959e-03	0.4718	2.2951e-02	6.8680e-04	0.2851
Ensemble optimized (ES)	2.3148e-02	6.1281e-04	0.3462	2.2835e-02	8.0628e-04	0.2963
NDEO	2.2749e-02	1.0355e-04	0.1121	2.2444e-02	3.9783e-05	0.1054

Model 4	Heating			Cooling		
Criterion	MSE_v	MSE_t	M_{ae}	MSE_v	MSE_t	M_{ae}
KTB	2.7253e-02	1.4677e-03	0.5740	2.8519e-02	6.8226e-04	0.3552
Ensemble (SA)	2.2825e-02	1.1586e-03	0.4718	2.3741e-02	6.6876e-04	0.2851
Ensemble optimized (ES)	2.2104e-02	5.7500e-04	0.1681	2.3516e-02	7.8834e-04	0.2963
NDEO	2.1364e-02	1.1089e-04	0.0827	2.3381e-02	1.5531e-04	0.1666

The ES method managed to achieve a stable performance improvement around 20% in all Models. However it couldn't keep up with the enhancement levels achieved when NDEO is applied. This performance is comparable with the performance improvements obtained using NDEO in the previously model typology (SPSI). This suggests that NDEO performance can scale along side with the models complexity, a highly desirable feature.

The study of SPMI typolgy models proceeds further in Appendix(D), where the following operating points are covered. We move forward MPMI model typolgy.

Table 5.9: Generalization error obtained with the ensemble approaches in comparison with the KTB model selection. Used data: corrupted.

Model 1 (2 Lags)	**Heating**	**Cooling**	**Average comparison**
Ensemble (SA)	-45.73 %	30.57 %	-7.58 %
Ensemble optimized (ES)	23.75 %	18.99 %	21.37 %
NDEO	85.18 %	80.57 %	82.88 %

Model 2 (3 Lags)	**Heating**	**Cooling**	**Average comparison**
Ensemble (SA)	-3.73 %	3.04 %	-0.35 %
Ensemble optimized (ES)	46.03 %	-13.51 %	16.26 %
NDEO	69.51 %	77.86 %	73.69 %

Model 3 (4 Lags)	**Heating**	**Cooling**	**Average comparison**
Ensemble (SA)	20.42 %	-0.82 %	9.80 %
Ensemble optimized (ES)	59.22 %	-18.35 %	20.43 %
NDEO	93.11 %	94.16 %	93.64 %

Model 2 (4 Lags)	**Heating**	**Cooling**	**Average comparison**
Ensemble (SA)	21.06 %	1.98 %	11.52 %
Ensemble optimized (ES)	60.82 %	-15.55 %	22.64 %
NDEO	92.44 %	77.24 %	84.84 %

- Sensor 2, all TUS beam intensities ($0.5W/cm^2$, $1.0W/cm^2$, $1.5W/cm^2$, $1.8W/cm^2$)
- Sensor 3, all TUS beam intensities ($0.5W/cm^2$, $1.0W/cm^2$, $1.5W/cm^2$, $1.8W/cm^2$)
- Sensor 4, all TUS beam intensities ($0.5W/cm^2$, $1.0W/cm^2$, $1.5W/cm^2$, $1.8W/cm^2$)
- Sensor 5, all TUS beam intensities ($0.5W/cm^2$, $1.0W/cm^2$, $1.5W/cm^2$, $1.8W/cm^2$)

5.3.0.2 Results discussion

The model complexity was increased to deal with single point environments admitting all TUS intensities considered. The models constructed have shown satisfactory results, again with errors thresholds below the MRI standard of 0.5 ^oC. All the networks were constructed using data contaminated with noise which explicits the robustness of the modelling approaches applied. Albeit the increased environment complexity, it was not necessary to increase the BSNNs modelling power, i.e. the maximum allowed spline order was kept at 4. Yet the highest errors occur using data collected at Sensor 1. Due to abrupt temperature variation the network exhibits some difficulties to follow the fast temperature rise.

Regarding the ensembles approaches, the SA method couldn't perform well and continues to present an oscillatory behaviour in compliance with the results from last typology. The ES approach achieved consistent generalization performance improvements though the environments, although there was a reduction in the enhancement margins. Concerning the NDEO, the results were similar to the ones obtained for the last typology, making this approach more robust than the ES. Until now the results are indicating that a non-linear combination of the individual outputs might compensate the overhead introduced in the system.

Furthermore the results suggest that scaling the environment complexity, and hence the forecasting task, do not compromise the performance figures obtained for the models, which remain, to a large extent, comparable with the results obtained regarding the previous model typology (SPSI). This is a good indicator concerning the robustness of the modelling approach employed and creates confidence in the results as we move to our last model typology.

5.4 Multi-point multi-intensity (MPMI) $1 - D$

Having obtained satisfactory results in previous models typologies we expect to extend the same line of results in this section. Being this the last typology considered, we shall enquire and test the limits of the approach regarding the prediction horizon h. Given the importance of this parameter in any predictive model, we increased it to 7 seconds ($h = 7s$), which provides a more comfortable margin. This increased prediction horizon scenario is covered in Appendix(E), where the results prove themselves, in a large extent, highly comparable to the results obtained so far, concerning the performance criteria applied, which is a positive indicator about the scalability of the solution being derived in this work, providing great insight regarding the robustness of the model approach.

Once having reached to this point with satisfactory results, it would be interesting to further increase the prediction horizon, hardening the forecasting task. We propose to extend it to 1 minute, $h = 60$ seconds. Therefore the data set should be highly shortened. The trials briefed in Chapter (3) consists in 45 minutes each trial, hence approximately

20 data points for each phase (heating and cooling) will be used. The challenge is to assess if such a reduced amount of information about the process is sufficient for the network to learn the dynamics of the process. Figure (5.10) provides a glimmer about the amount of data to be used in each trial.

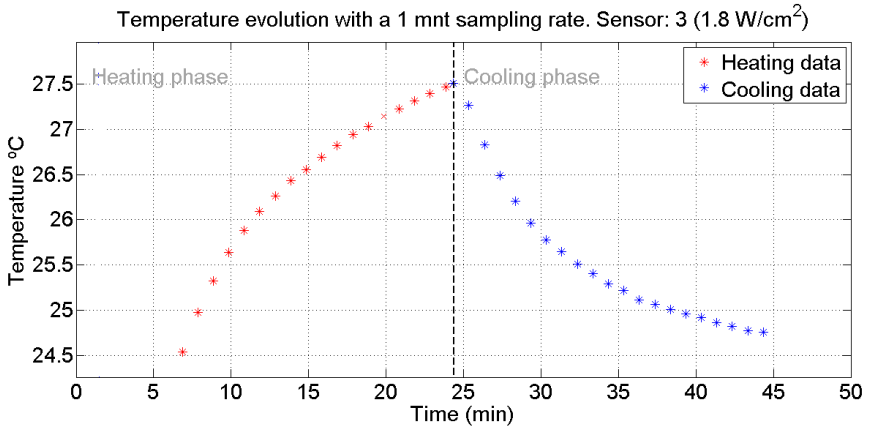

FIGURE 5.10: An example of a trial consisting of 40 temperature data points, divided between heating and cooling phase. Sampling period $T = 60$ seconds. Collected from the *homogeneous phantom* experimental setup.

From the above plot one can note that the amount of data has been dramatically reduced. Again we constructed four distinct models using the uncorrupted data exposed in Figure (5.11). However, for this Model we consider numbers of input lags ranging from $3 - 6$, due to the hardening of the forecasting task, an additional input lag was introduced. The maximum allowed spline order was kept at 5, and the data was splitted in compliance with the division exposed in Appendix (B), **Model 2**.

Table (5.10) reflects the performance criteria obtained for all models.

Even dramatically reducing the amount of data, i.e. the information about the process, the performing figures calculated are to a great extent, comparable with the line of results that we have been presenting. Regarding all models, the maximum absolute error M_{ae} was consistently kept below 0.1 ^oC through all the data sets (test included). This results emphasize and highlight BSNNs approximation power and ability to interpolate and extrapolate data with precision.

We chose to provide a graphical representation of the output in a distinct way. Instead of providing one plot for each trial, we decided to present a global overview over the entire data universe. We divided this representation between the heating and cooling

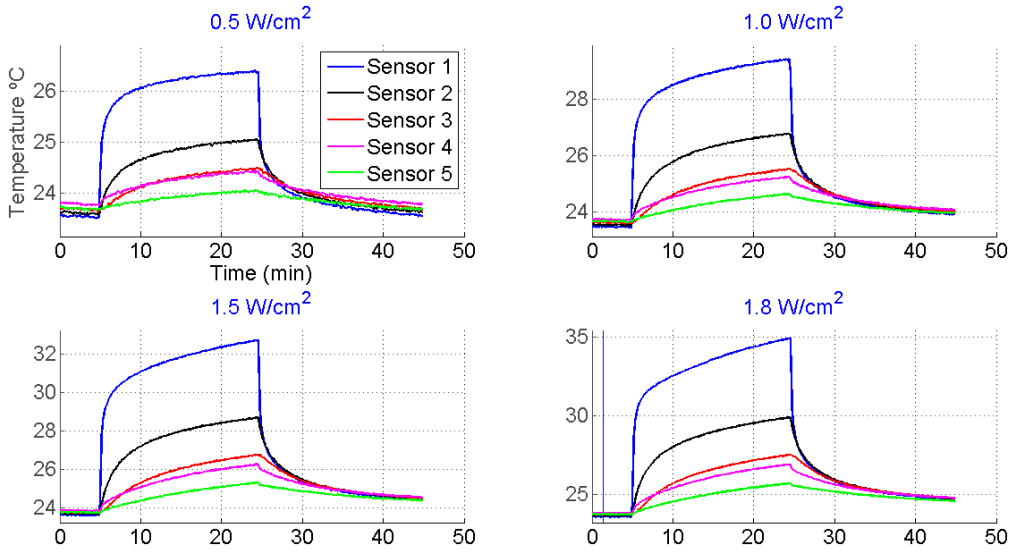

FIGURE 5.11: Unaltered data set used for MPMI model training, validation and test. Collected from the *homogeneous phantom* experimental setup. Data used: uncorrupted.

phase. Thus we present in a single plot the complete model's behaviour in the heating (or cooling) phase, regarding all sensors and intensities. Figures (5.12) and (5.13) provide this overview in respect to the heating and cooling phase respectively. The results reflect the behaviour of model 3 in the training set.

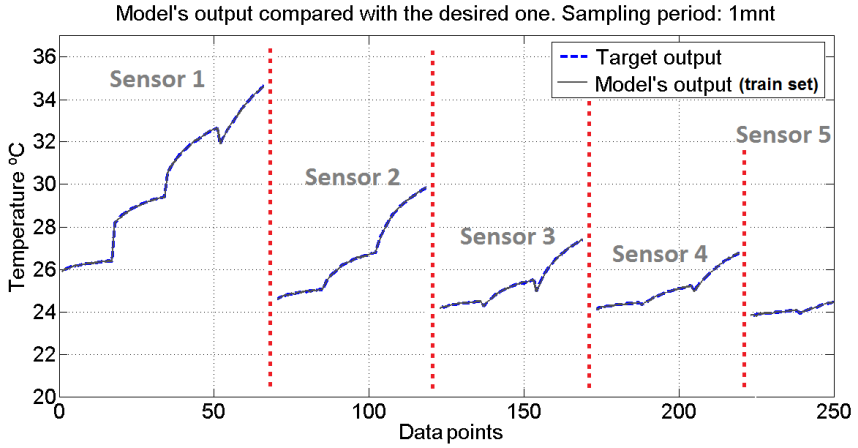

FIGURE 5.12: Top view from the output of model 3 concerning training set (heating phase), selected using the KTB approach. The blue line represents the desired behaviour and the model's test output is given by the black line. *homogeneous phantom* experimental setup. The sensors labels just serve as a guidance. The model was constructed using the data illustrated in Figure (E.1). Data division during the model constructed followed thes scheme exposed in Appendix(B), Model 2.

TABLE 5.10: Performance descriptors obtained concerning models with different number of input lags. The models presented were selected using the KTB approach. MPMI. Data used: uncorrupted

Model 1(3 lags)		
Criterion	Heating	Cooling
BIC	-1785	-2056
MSE	5.4081e-04	3.0614e-04
$MSRE$	2.0504e-05	1.2510e-05
MSE_v	4.9297e-04	5.1184e-04
$MSRE_v$	1.9065e-05	2.0483e-05
MSE_t	9.5985e-04	4.8647e-04
$MSRE_t$	3.7932e-05	1.9477e-05
M_{ae}	0.0741	0.0636
LWN	24	23
SR	n	n

Model 2(4 lags)		
Criterion	Heating	Cooling
BIC	-1759	-1952
MSE	3.8854e-04	3.2060e-04
$MSRE$	1.4625e-05	1.3089e-05
MSE_v	3.2829e-04	1.9759e-04
$MSRE_v$	1.2879e-05	8.0907e-06
MSE_t	5.7796e-04	4.5521e-04
$MSRE_t$	2.2625e-05	1.8260e-05
M_{ae}	0.0712	0.0615
LWN	23	18
SR	n	n

Model 3(5 lags)		
Criterion	Heating	Cooling
BIC	-1684	-1868
MSE	3.7283e-04	2.4673e-04
$MSRE$	1.3648e-05	1.0005e-05
MSE_v	2.1692e-04	2.3333e-04
$MSRE_v$	8.6492e-06	9.5787e-06
MSE_t	5.9423e-04	2.9731e-04
$MSRE_t$	2.4316e-05	1.2482e-05
M_{ae}	0.0518	0.0474
LWN	17	23
SR	n	n

Model 4(6 lags)		
Criterion	Heating	Cooling
BIC	-1581	-1775
MSE	3.3990e-04	2.4266e-04
$MSRE$	1.2508e-05	9.9022e-06
MSE_v	2.5806e-04	3.7787e-04
$MSRE_v$	9.9980e-06	1.5366e-05
MSE_t	2.0359e-04	1.2971e-04
$MSRE_t$	8.3812e-06	5.4384e-06
M_{ae}	0.0451	0.0501
LWN	18	18
SR	n	n

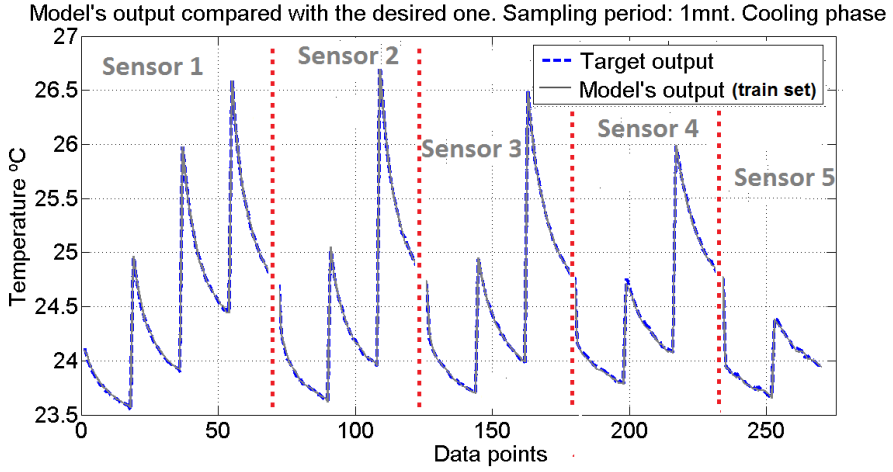

FIGURE 5.13: Top view from the output of model 3 concerning training set (cooling phase), selected using the KTB approach. The blue line represents the desired behaviour and the model's test output is given by the black line. *homogeneous phantom* experimental setup. The sensors labels just serve as a guidance. The model was constructed using the data illustrated in Figure (5.11). Data division during the model constructed followed thes scheme exposed in Appendix(B), Model 2.

The above figures, despite confusing and hard to read, are capable of providing a top view over the network performance, which is important to globally assess the model's learning process. The sensor's labels merely give some information about the region and were not strictly positioned. The results clearly show that, for both heating and cooling phases, the networks completely learned how the temperature evolves with respect to the TUS beam intensity and spatial location of the operating point. Note that in the two previous plots the areas reserved to Sensor 1 one reveal four bumps (curves), whereas areas regarding Sensors 2, 3 and 4 exhibit 3 bumps, and finally Sensor 5 with 2 noticeable curves. This fact is due to the data division exposed in Appendix(B), Model 2. The curves depict only the results obtained for the training set.

Figures (5.14) and (5.15) illustrates the output of the same model (3), but now considering two operating point embedded in the test set, Sensor 3 $0.5W/cm^2$ and Sensor 4 $1.8W/cm^2$, respectively.

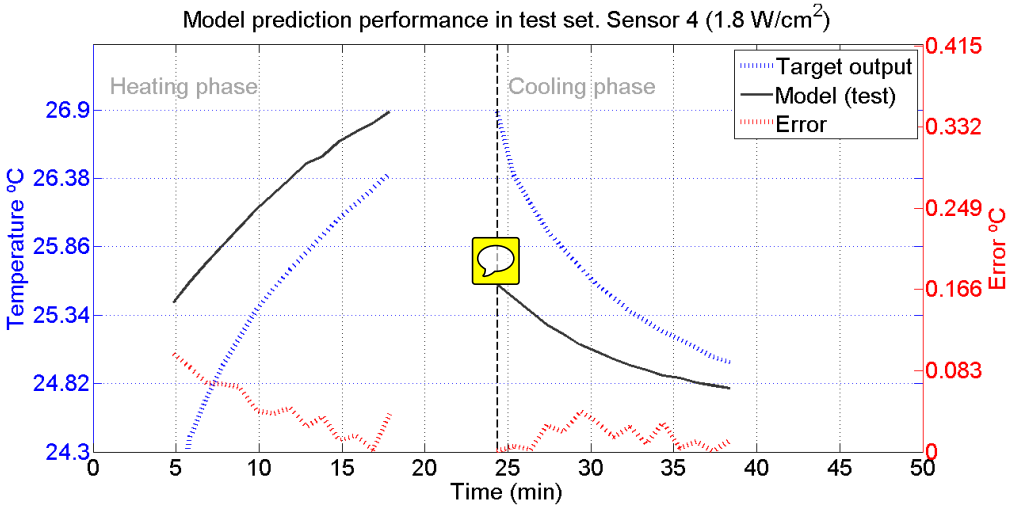

FIGURE 5.14: Output curve of model 3, selected using the KTB approach, in two operating points embedded in test set (sensor 3, $0.5W/cm^2$). The blue line represents the desired behaviour and the model's test output is given by the black line. *homogeneous phantom* experimental setup. The sensors labels just serve as a guidance. The model was constructed using the data illustrated in Figure (5.11). Used data: uncorrupted.

Assessing this hard tests one can visualise a noticeable trend from the model output dynamic to follow the real dynamic, and it does it with errors below 0.1 ^oC.

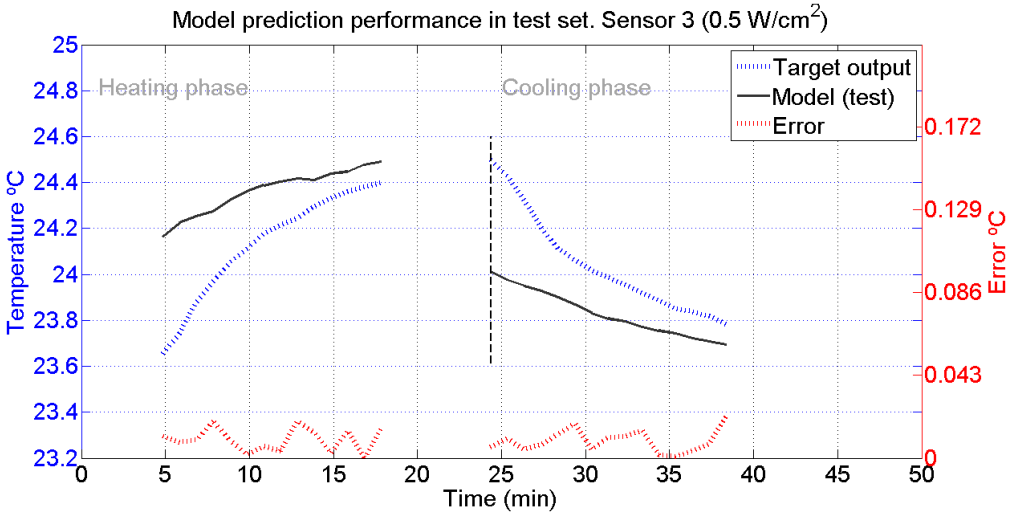

FIGURE 5.15: Output curve of model 3, selected using the KTB approach, in two operating points embedded in test set (sensor 4, $1.8W/cm^2$). The blue line represents the desired behaviour and the model's test output is given by the black line. *homogeneous phantom* experimental setup. The sensors labels just serve as a guidance. The model was constructed using the data illustrated in Figure (5.11). Used data: uncorrupted.

The natural extension to the line of work being done consists of adding noise to the original, scarce data. We now propose to contaminate the temperature evolution information and see how the models react while keeping the 1 minute prediction horizon. Figure (5.16) illustrates a contaminated data set (single point, single intensity), admitting a 60 seconds sampling period, thus around 40 temperature data points are available for each trial.

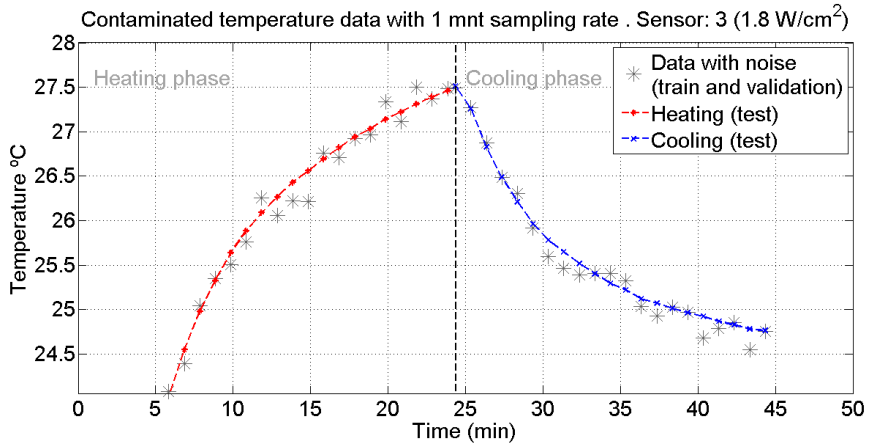

FIGURE 5.16: An example of a trial consisting of 40 temperature data points contaminated with Gaussian noise, divided between heating and cooling phase. Sampling period $T = 60$ seconds. Collected from the *homogeneous phantom* experimental setup.

The above plot illustrates the destruction of information about the process dynamic

caused by the addition of Gaussian noise $e_i \sim \mathcal{N}(0, \sigma)$. This disruption of information is now dramatically amplified since the data is scarce. Therefore one can allege that such an scenario provides an ultimate robustness assessment test to the modelling approach, since we are considering scarce data with low quality. Using the noisy data shown in Figure (5.17) four models were again built. Data division was the same as in the previous Model, with the difference that the test set was merged into the validation set. However note that this merge only occurs in this experiment and not the previous one, the test set will now be the uncorrupted data, to asses the learning process robustness. In compliance with this last remark, the uncorrupted data universe was used to assess the generalization ability of the network (test set). The performance descriptors concerning all models are shown in Table (5.11).

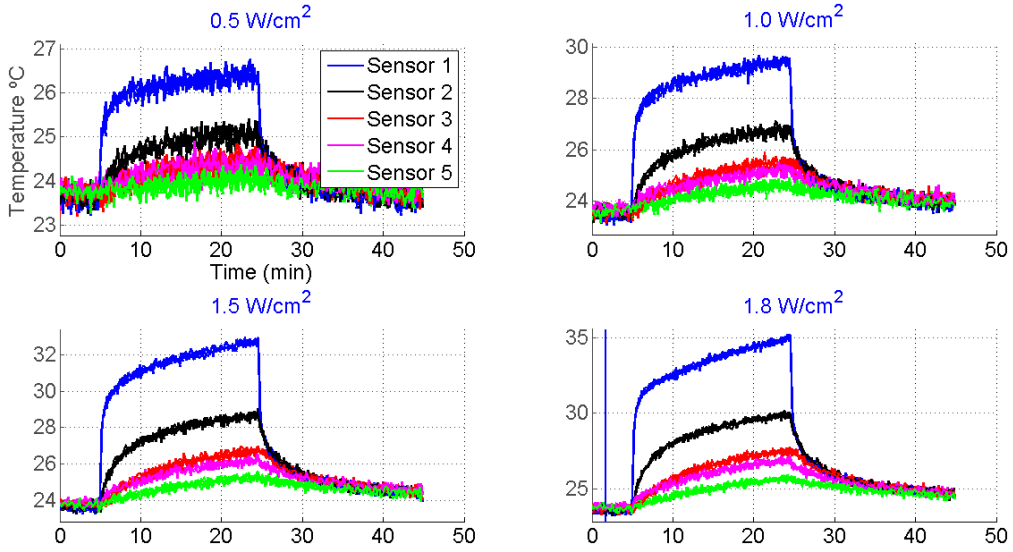

FIGURE 5.17: Data after the addition of random Gaussian noise. The noisy data is used for training and validation. The unaltered, noise free data is used to test the model. Collected from the *homogeneous phantom* experimental setup.

All the models achieved interesting and appreciable performance figures. Despite the harsh conditions of the learning process, whose available data was scarce and of low quality, the performance descriptors are still comparable with the results obtained with the first considered prediction horizon of $1s$. The amount of data used was dramatically reduced by a factor of $K = 1/60$ and still the results are similar, with maximum absolute errors kept below 0.3 ^oC, which evince the robustness of BSNNs when the forecasting task difficulty is scaled. Figures (5.18) and (5.19) provide a top view over the complete behaviour concerning the output curve of Model 4 over the heating and cooling phase,

TABLE 5.11: Performance descriptors obtained concerning models with different number of input lags. The models presented were selected using the KTB approach. MPMI. Data used: corrupted.

Model 1(3 lags)		
Criterion	Heating	Cooling
BIC	-860	-1024
MSE	3.2440e-02	2.2087e-02
$MSRE$	1.2343e-03	9.0177e-04
MSE_v	3.3205e-02	3.5442e-02
$MSRE_v$	1.2755e-03	1.4263e-03
MSE_t	6.0667e-03	2.5653e-03
$MSRE_t$	2.2874e-04	1.0514e-04
M_{ae}	0.4539	0.1505
LWN	13	13
SR	n	n

Model 2(4 lags)		
Criterion	Heating	Cooling
BIC	-789	-963
MSE	3.4553e-02	1.9626e-02
$MSRE$	1.3007e-03	8.0298e-04
MSE_v	4.2219e-02	2.1593e-02
$MSRE_v$	1.6236e-03	8.6601e-04
MSE_t	8.5161e-03	2.0860e-03
$MSRE_t$	3.1937e-04	8.4590e-05
M_{ae}	0.5220	0.1949
LWN	13	19
SR	n	n

Model 3(5 lags)		
Criterion	Heating	Cooling
BIC	-740	-880
MSE	3.1031e-02	2.1269e-02
$MSRE$	1.1766e-03	8.7002e-04
MSE_v	2.5197e-02	2.1947e-02
$MSRE_v$	9.7181e-04	8.8626e-04
MSE_t	4.8442e-03	2.6515e-03
$MSRE_t$	1.8186e-04	1.0760e-04
M_{ae}	0.2928	0.3047
LWN	17	19
SR	n	n

Model 4(6 lags)		
Criterion	Heating	Cooling
BIC	-710	-853
MSE	2.9993e-02	1.9412e-02
$MSRE$	1.1254e-03	7.9900e-04
MSE_v	2.7696e-02	2.9739e-02
$MSRE_v$	1.0726e-03	1.2069e-03
MSE_t	4.1604e-03	2.1082e-03
$MSRE_t$	1.5499e-04	8.6064e-05
M_{ae}	0.2629	0.2151
LWN	14	17
SR	n	n

respectively. Note that all of the data present in these figures is part of the test set, that consists of the original, unaltered data, concerning all intensities and spatial locations. This figures, albeit not very specific, provide a quick assessment of the learning process the networks were incurred. Note also that now all the different areas concerning the various sensors all have the same amount of data. This is because the test set comprises all the available uncorrupted test, whereas the model training and validation was done using the corrupted data.

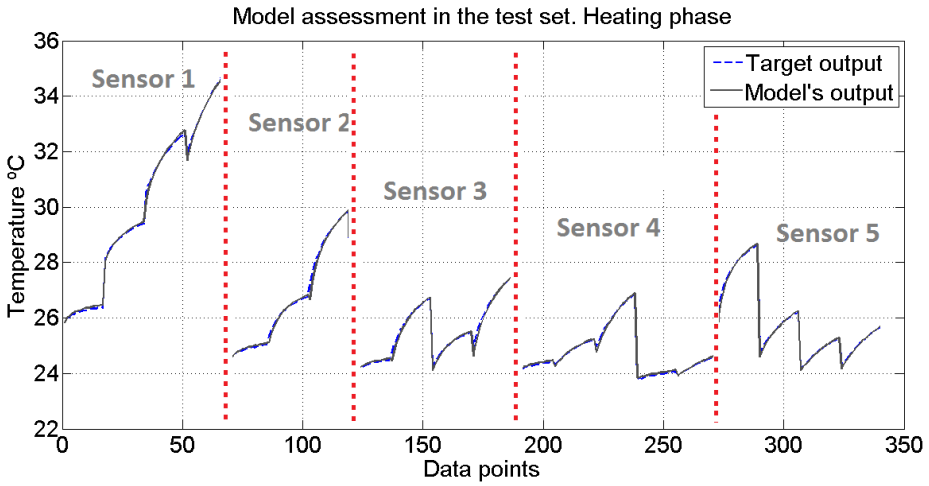

FIGURE 5.18: Top view from the output of model 3 through all the test set (heating phase), selected using the KTB approach. The blue line represents the desired behaviour and the model's test output is given by the black line. *homogeneous phantom* experimental setup. The sensors labels just serve as a guidance. The model was constructed using the data illustrated in Figure (E.4). Data division during the model constructed followed thes scheme exposed in Appendix(B), Model 2. Data used: corrupted

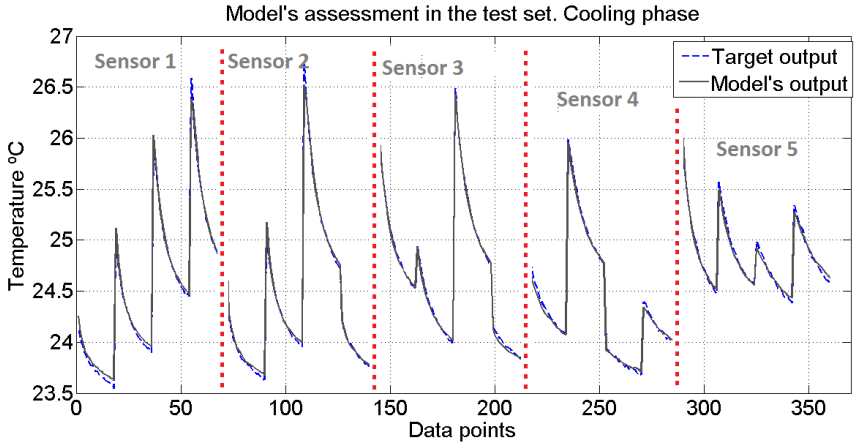

FIGURE 5.19: Top view from the output of model 3 through all the test set (cooling phase), selected using the KTB approach. The blue line represents the desired behaviour and the model's test output is given by the black line. *homogeneous phantom* experimental setup. The sensors labels just serve as a guidance. The model was constructed using the data illustrated in Figure (5.17). Data division during the model constructed followed thes scheme exposed in Appendix(B), Model 2. Data used: corrupted

Observing the figures we get a quick indicator that the learning process was successful and the process dynamics were understood by the BSNNs. Using scarce corrupted data, with a different number of input lags in each model, one can infer an ensemble with

uncorrelated models, or at least not strongly correlated. Thus the ensemble approaches were applied and the assessment can be made in Table (5.12). Table (5.13) reflects the comparison between the ensemble and the KTB paradigm, in the test set.

TABLE 5.12: Performance comparison between all methodologies employed. MPMI. Data used: corrupted

Model 1	Heating			Cooling		
Criterion	MSE_v	MSE_t	M_{ae}	MSE_v	MSE_t	M_{ae}
KTB	3.3205e-02	6.0667e-03	0.4539	3.5442e-02	2.5653e-03	0.1505
Ensemble (SA)	2.6175e-02	6.5921e-03	0.4539	2.6675e-02	1.9205e-03	0.1383
Ensemble optimized (ES)	2.6205e-02	6.8850e-03	0.4539	2.6015e-02	1.9975e-03	0.1383
NDEO	3.1347e-02	1.0638e-02	0.5186	2.6607e-02	5.0165e-03	0.6551

Model 2	Heating			Cooling		
Criterion	MSE_v	MSE_t	M_{ae}	MSE_v	MSE_t	M_{ae}
KTB	4.2219e-02	8.5161e-03	0.5220	2.1593e-02	2.0860e-03	0.1949
Ensemble (SA)	4.0581e-02	4.7687e-03	0.3898	2.0823e-02	1.6091e-03	0.1217
Ensemble optimized (ES)	3.9585e-02	5.0152e-03	0.3898	1.9507e-02	1.6623e-03	0.1217
NDEO	4.7837e-02	8.8094e-03	0.5255	1.9385e-02	6.7776e-01	15.1535

Model 3	Heating			Cooling		
Criterion	MSE_v	MSE_t	M_{ae}	MSE_v	MSE_t	M_{ae}
KTB	2.5197e-02	4.8442e-03	0.2928	2.1947e-02	2.6515e-03	0.3047
Ensemble (SA)	1.8996e-02	3.2280e-03	0.2595	2.1598e-02	1.5086e-03	0.1208
Ensemble optimized (ES)	1.8225e-02	3.5600e-03	0.2595	2.0720e-02	1.5953e-03	0.1115
NDEO	2.5223e-02	4.8420e-03	0.2925	2.2003e-02	2.7599e-03	0.3414

Model 4	Heating			Cooling		
Criterion	MSE_v	MSE_t	M_{ae}	MSE_v	MSE_t	M_{ae}
KTB	2.7696e-02	4.1604e-03	0.2629	2.9739e-02	2.1082e-03	0.2151
Ensemble (SA)	2.5603e-02	2.2882e-03	0.1912	2.7698e-02	1.3309e-03	0.1208
Ensemble optimized (ES)	2.4527e-02	2.6439e-03	0.2270	2.8442e-02	1.4233e-03	0.1115
NDEO	2.7241e-02	3.3818e-03	0.1858	2.9719e-02	2.0975e-03	0.2149

The results demonstrate a poor performance using the NDEO approach, while the SA approach, whose performance usually was the worst, appears as the top performance booster. We deduce these results comes from the fact that due to the scarcity of data, NDEO fails to develop insight about how to combine to enhance the final output. Note that in NDEO combining learning phase, just the training and validation phases are used, which in this Model are contaminated and moreover and most importantly, scarce. On the contrary, the simple SA method spreads the combination more widely, and

TABLE 5.13: Generalization error obtained with the ensemble approaches in comparison with the KTB model selection. Data used: corrupted

Model 1 (2 Lags)	**Heating**	**Cooling**	**Average comparison**
Ensemble (SA)	-8.66 %	25.13 %	8.24 %
Ensemble optimized (ES)	-13.49 %	22.13 %	4.32 %
NDEO	-75.35 %	-95.55 %	-85.45 %

Model 2 (3 Lags)	**Heating**	**Cooling**	**Average comparison**
Ensemble (SA)	44.00 %	22.86 %	33.43 %
Ensemble optimized (ES)	41.11 %	20.31 %	30.71 %
NDEO	-3.44 %	-323.03 %	-161.74 %

Model 3 (4 Lags)	**Heating**	**Cooling**	**Average comparison**
Ensemble (SA)	33.36 %	43.10 %	38.23 %
Ensemble optimized (ES)	26.51 %	39.84 %	33.17 %
NDEO	0.05 %	-4.09 %	-2.02 %

Model 2 (4 Lags)	**Heating**	**Cooling**	**Average comparison**
Ensemble (SA)	45.00 %	36.87 %	40.94 %
Ensemble optimized (ES)	36.45 %	32.49 %	34.47 %
NDEO	18.71 %	0.51 %	9.61 %

its therefore less prone to overfitting. The ES method also provided very satisfactory results, albeit it has also been evolved using noisy data. The difference might reside on the fact that ES approach evolves its weights considering only the validation set, while the NDEO uses both, which may be the cause poor performance. Nevertheless these methods provided a further boost in the accuracy of the predictions, which can be used to generally decrease the errors.

5.4.0.3 Results discussion

Concerning MPMI $1-D$ model typology we started by considering a prediction horizon of 1 second which was further increased to 7 seconds and at a second experiment to 60 seconds. It was observed comparable results among all modelling prediction horizons. This observation is crucial to assess the robustness of the models, which already had given satisfactory responses to increases in the environment complexity. Furthermore the models demonstrate to be robust to several increases in complexity of various natures, which is essential to the forecasting activity. The observations are analogous when the information present in the data was disrupted by the addition of noise, with errors kept below a threshold, within a satisfactory margin, 0.5 ^{o}C . Also important its the fact

that this performance figures are possible with really simple networks, whose average linear weight norms (LWN) were about 20. Therefore, nearly 20 neurons are sufficient to model a $1 - D$ line of 5 sensors admitting four models intensities (20 operating points), which suggests this approach is highly scalable to more complex environments.

The maximum allowed spline order was set to 5. However, this increased is just needed for regions with abrupt temperature changes. Just the data collected by sensor 1 at high TUS intensities fall into this category, therefore one might argue that this increased order is not necessary, and order 3 or 4 splines could get the job done with the same performance figures in all the others areas, with more robustness to noisy data. Another solution to address the problem would be to reserve a specific model that just deals with data from sensor 1, and another model for the rest of the non-sensitive data regions.

Concerning the ensemble approaches, the NDEO method demonstrated that it doesn't has available enough data to combine the outputs in a proficient way, it does not provide a robust combing scheme in the presence of scarce data, fact that is aggravated when the data is contaminated. Combining neural forecasts is not an easy task and some methods are just applicable in certain situations. Furthermore we saw that simple methods like the SA resulted in high performance boosts of the predictive activity, around 30%, which are excellent results since ensembles with just four individuals were considered, reinforcing the idea that simple combining methods are efficient. One can conclude that the problem with complex combining methods comes from the lack of robustness that this methods may exhibit, when they are subject to different and unexpected conditions. The whole combination mechanism (when complex) needs to be ascertained and revised to comply with the new encountered scenario, whereas a simple approach like the simple average (SA) combination scheme, has the robustness needed to perform reasonable well under changes in the modelling approach. Thus, complex methods demand more work and caution in the designing phase in order to maximize the performance boost that this methods can introduce in the forecasting activity, a trade-off between overhead in the designing phase and robustness

Chapter (6) concludes this work with an analysis and discussion of the results obtained in the present Chapter and some guidelines to future research.

6

Final discussion and future work

6.1 Introduction

Chapter (5) revealed good indicators about the feasibility of the modelling approach assessed in this work. Hopefully after more research we can bring such a system to reality to assist the real biomedical instrumentation practice. This chapter finishes this work giving an overview of the work derived as well as pointing guidelines for future research. Section (6.2) provides a detailed assessment of the most relevant performance criteria for each one of the models typologies considered. Section (6.2.4) focus on the results obtained when the ensemble methods were applied and tries to develop insight regarding the use of these techniques. In Section (6.4) some personal thoughts and observations are shared regarding artificial intelligence, more concretely in the field of neural networks. Section (6.3) sums up the main conclusions of this work and Section (6.5) suggest some directions of possible future researches.

6.2 Global assessment of the performance criteria

In a general way the results regarding all models typologies considered were satisfactory. However a deeper inspection can develop more insight for current improvement and guide future researches.

6.2.1 Mean Square Error

The *Mean Square Error*, Section (2.5), exposes the general prediction performance of the network. Such a performance figure gives the designer a initial clue over the success of the constructed model. Table (6.1) reveals the average MSE obtained for each model typology, with the models built using uncorrupted and contaminated data. Please note that these values only concern the MSE calculated in the test set, where the network generalization ability is tested.

	Average MSE_t	
Typology	**uncorrupted**	**corrupted**
SPSI	6.1011e-04	1.7883e-04
SPMI	1.5329e-03	5.6790e-05
MPMI (7)	1.3007e-04	1.3007e-04
MPMI (60)	2.0359e-04	2.0359e-04

TABLE 6.1: Average MSE (regarding heating and cooling phases) in test set, calculated for all models typologies, using uncorrupted and Gaussian contaminated data. Section (2.5) briefly explains this performance figure.

A quick glance at the these calculated figures suggests a successful modelling at all stages of the work. Despite this initial observation we should study how the performance of the system scales when the modelling environment complexity is also scaled. Note that a distinction has been made regarding the MPMI model typology, where we differentiate two different cases, with respect to the two different prediction horizon h considered, 7 and 60 seconds, linking to MPMI (7) and MPMI (60) respectively. Figure (6.1) provides a second view of these results.

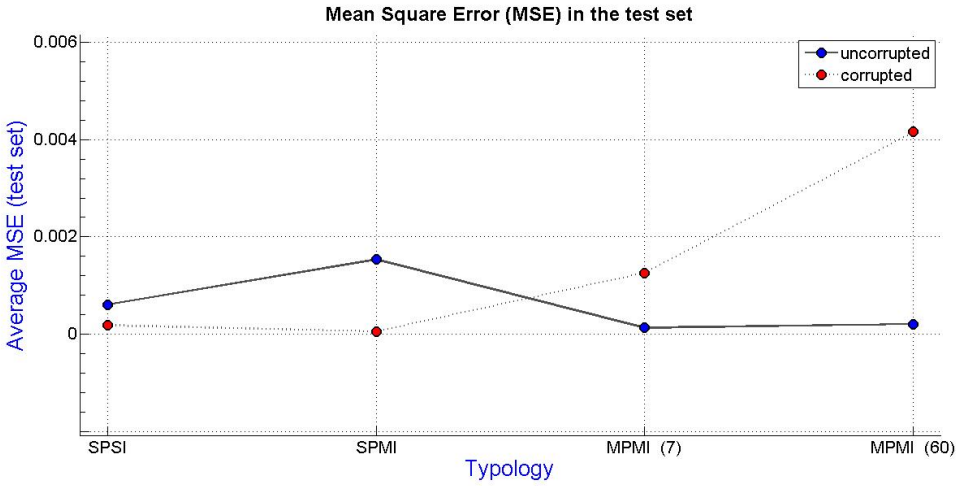

FIGURE 6.1: Graphical illustration of the average MSE (test set) calculated for all models typologies, using uncorrupted and Gaussian contaminated data. Solid line: values obtained with uncorrupted data; Dashed line: vales obtained with corrupted data.

Observe that for the first two model typologies (SPSI and SPMI) the generalization test exhibits better performance figures when the models were built using a Gaussian contaminated (corrupted) data set. Obviously a higher performance was expect from a model whose training set consists if high quality data (uncorrupted), however we are observing the opposite in this phase. These values are explained when we add to the equation the performance boost that appears when we successful combine four networks to form an ensemble. By doing so, we registered great significant improvements by linearly combining the networks in the ensemble (SA and ES) or by using more complex combination mechanisms (NDEO). Assuredly this performance enhancements justify the overhead introduced both at the designing phase and in the network's generalization ability. Naturally this enhancements can only come attached with a cost, as the ensemble total LWN[1] will also scale, although in a *linear* way, since the ensemble is formed by *adding* networks to the ensemble. A more detailed study is done more ahead at Section (6.2.2).

In the two following typologies (MPMI: 7 and 60) is observable that the models built with corrupted data can't follow with the performance achieved when training a network with the original data. Before digging into this point we make an important observation that notes the similar figures obtained concerning the MPMI model typology with

[1]Linear Weight Norm, Section (2.5).

two different predicting horizons using uncorrupted data. The fact that this value is piratically constant suggests a highly robust approach to variations in the **prediction horizon**. We justify the robustness classification since moving from a prediction horizon of 7 seconds to 60 seconds, is translated in a data volume decrease by a factor of approximately $k = \frac{1}{10}$ and the performance figures obtained were similar.

With respect to the figures obtained for these typologies, now using corrupted data, is also possible to derive some illations. Regarding the scenario where we considered a prediction horizon of 7 seconds it's noticeable a performance mitigation in comparison with the equivalent case using uncorrupted data. We justify this decrease by the lack of sufficient ambiguity levels, a necessary condition for the ensembles approach make an impact. We can then infer from the results that just contaminating the data set with random noise is not sufficient to create proper conditions for ensembles approaches to bloom its enhancements, i.e. the networks composing the ensembles were positively correlated. To solve this, more sophisticated mechanisms should be employed to negatively correlate the ensemble. Using different network structures can be a starting point. This effect is even more aggravated when the prediction horizon is extended since the quality of the data has been dramatically reduced, 20 contaminated data points for each operating point, i.e. a TUS beam intensity measured at a single sensor for each phase (cooling, heating). Nevertheless, despite of the mitigation of the ensemble methods performance, the performance criteria obtained are always kept under satisfactory thresholds.

6.2.2 Linear Weight Norm

The LWN, see Section (2.5), measures the network complexity, which ideally should be kept simplest as possible, admitting that the structure is powerful enough to have the job done, i.e. we should use the minimum resources possible to meet the requirements inherent to the task. This way the designer tries to enforce a protecting barrier in the learning process that filters the noise dynamics from being learned by the network parameters.

Table (6.2) exposes the average network complexity as dictated by the ASMOD algorithm used to create and evolve the structures. This figure reflects the computational

costs associated with each network (or ensemble) when making a prediction. Higher LWN are translated in more heavy computational time, which in an embedded system application with limited resources must be considered.

	LWN	
Typology	**uncorrupted**	**corrupted**
SPSI	09.0	39.2
SPMI	18.2	67.9
MPMI (7)	63.0	33.0
MPMI (60)	18.0	16.0

TABLE 6.2: Average linear weight norm obtained at each stage of the typology universe, regarding models built both with the original and corrupted data sets.

The evolution of the average complexity is better illustrated in Figure (6.2).

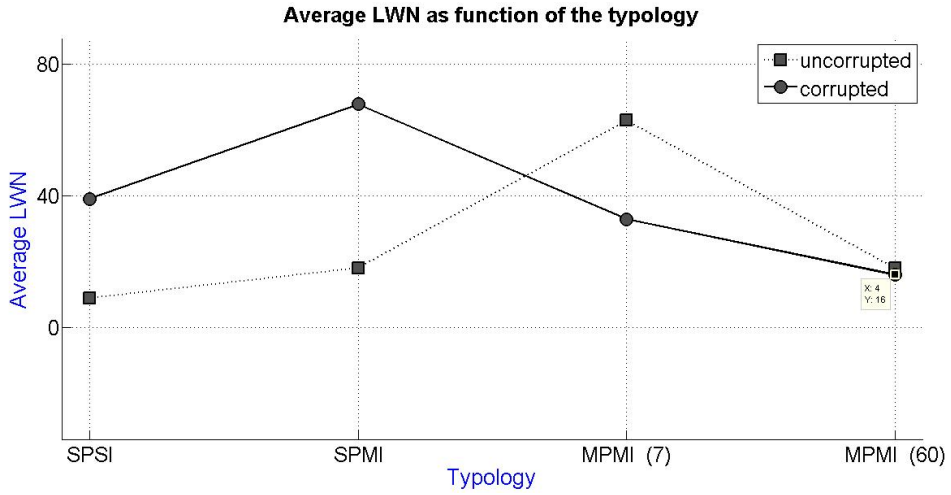

FIGURE 6.2: Graphical illustration of the average LWN calculated for all models typologies, using uncorrupted and Gaussian contaminated data. Solid line: values obtained with uncorrupted data; Dashed line: vales obtained with corrupted data.

The dashed line (uncorrupted) exhibits a continuous increasing the in linear weight norm as we move towards the most complex typology (MPMI). It should be mentioned that the first two typology admitted a prediction horizon of $h = 1$ second, while in the first MPMI typology considered this horizon was extended to by a factor of seven to $h = 7$ seconds. This means the data sets forming the dynamics of each operating point were reduced seven times accordingly. Still as the designing passed from SPMI to MPMI with $h = 7$ the average LWN experienced a sharp increase. A further increase in the prediction horizon to $h = 60$ alleviates, in a large extension, the number of resources needed by the structure to accommodate the knowledge embedded in the data set, which is now dramatically reduce by approximately a factor of 10. Curiously by doing so, we land

near the complexity levels calculated for the initial models typology. This is a very good indicator, since it gives the designer the possibility to further increase the complexity (reality) of the modelling environment, i.e. the approach is **scalable** to variations with respect to the **modelling environment**, a highly desirable feature.

With the same resources, i.e. structures of similar complexity, thus similar computational cost, we were able to model the most simple model typology as well as the more complex typology considered, which naturally was submitted to a data reducing (prediction horizon extension), but nevertheless with comparable performance figures. A LWN of 18 means that with only 18 *neurons* (basis functions in the BSNNs scope), it was possible to learn 20 different operating points (5 sensors $\times$ 5 TUS beam intensities), having 40 data points each.

Noticeable is also the intimate relation existing between the LWN and the data volume size, where the **non parametric** nature of neural networks is well exposed. Increasing the data volume comes with a similar increase in the linear weight norm of the network, i.e. more parameters to adapt in the learning process, which can sentence the feasibility of the implementation.

Concerning the results obtained for the models trained with corrupted data (solid line in Figure 6.2), one can observe in the first two typologies more complex network structures which are result of the ensemble formation, which linearly adds the individually LWN_i regarding each network N_i that composes the ensemble. This is the price to pay to embrace the performance boosts brought in using the ensembles methods. Trivially for a network ensemble composed of N networks, the average additional complexity introduced when using these methods would be of $N \times \mu_{LWN}$, where μ_{LWN} is the linear weight norm of an average network that enters the ensemble. Here it is also visible the decay in the ensemble methods performance when the model typology advances to MPMI with $h = 7$ seconds. A single KTB[2] model was chosen instead of the ensemble, since it didn't worth the additional overhead.

[2]Keep-the-best

6.2.3 Maximum Absolute Error

Another crucial specification error bounding in our forecasting network. Biomedical applications often require strict bounds that need to be strictly fulfilled if the system is to be used with real patients. Thus we should study the maximum absolute errors obtained and conclude if they are in compliance with the norms required by the biomedical organizations, which currently classify the MRI[3] has the gold standard with 0.5 $^{o}C/cm^{3}$ of temperature resolution. Table (6.3) exposes the maximum absolute errors obtained in this work for all typologies.

	Maximum absolute error ^{o}C			
	uncorrupted		corrupted	
Typology	Average	Max	Average	Max
SPSI	0.1212	0.3230	0.0831	0.3039
SPMI	0.1379	0.3252	0.0564	0.1666
MPMI (7)	0.0460	0.0477	0.3004	0.3264
MPMI (60)	0.0475	0.0510	0.2345	0.2629

TABLE 6.3: Average maximum absolute error obtained at each stage for each model typology, regarding models built both with the original and corrupted data sets.

The results show the maximum absolute errors were consistently kept under 0.33 ^{o}C, which results in a very comfortable margin, thus we can trust in the forecasting provided by the network, i.e. the system is *reliable*. Is also interesting to note that, as far as the environment complexity was increased, we detected no increase in the maximum errors registered. This is a very meritorious indicator that suggests the reliability of the system can scale along side with the modelling environment complexity.

6.2.4 Ensembles methods

A great deal of effort was indeed projected on various attempts to enhance the performance of the models created. In any biomedical application the designing hierarchy always has the patient as the first concern, followed by the actual technical implementation feasibility. We deeply believe that any improvement can help the usefulness of the application, creating more impact in the patient. Our efforts on this point reflected these beliefs.

[3]Magnetic Resonance Imaging

	uncorrupted (%)			**corrupted(%)**		
typology	SA	ES	NDEO	SA	ES	NDEO
SPSI	-2.86	-0.12	-13.1	-98.8	30.1	58.7
SPMI	1.86	-3.31	-3.20	-76.6	54.3	72.2
MPMI (7)	4.96	5.67	0.01	4.98	3.27	6.64
MPMI (60)	-17.56	-2.71	4.42	8.24	4.34	-160

TABLE 6.4: Average performance enhancements obtained when the various ensemble methods were applied. Note that these results are obtained in comparison with the KTB approach, i.e. the best trained single model. Note that a negative value means a deteoration in the performance, i.e. the performance got worst when compared with the KTB approach,

Moving on to the actual results obtained, Table (6.4) exposed the enhancements experienced through the typologies, in both experiments considered. As we saw previously the contamination of the data definitely impacts the enhancements, so it would be convenient to analyze both experiments separately.

The results obtained aren't, by any means, outstanding ones at first glance but nevertheless they shouldn't be neglected, they form a starting point suggesting the enhancements are possible, and we can use them to redirect focus to the key points involved in combining the results of an ensemble.

Starting with the uncorrupted data experience, the three methods applied are visually better confronted in Figure (6.3).

Using uncorrupted data means that all the models were trained with the same data sets. Even though one could make changes regarding the data division into training, validation and test sets, the training examples were too much similar for two consequent built models to be uncorrelated. Consequently, any ensemble created with models positively correlated lacks the sufficient ambiguity levels that allow the ensemble methods to boost the accuracy of the predictions. These thoughts are reflected Figure (6.3), where we see that no method at any instant was able to achieve the 10% mark. Actually the plotting gives the feeling that the average is situated around the 0% line. This clearly suggests that the ensemble mechanism are completely useless under such circumstances and are there merely adding unjustified overhead to the system. Despite creating the models with a different number of input lags, the results were not satisfactory and another mechanisms need to be applied in order to force a negative correlation among the

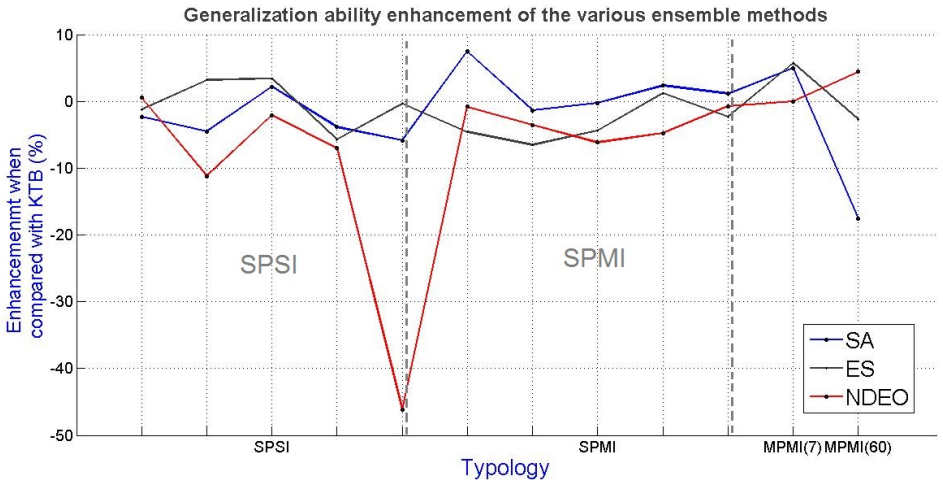

FIGURE 6.3: Ensemble methods performance enhancements of the various methods when compared with the KTB paradigm, regarding experiences where the models were build using uncorrupted data. SPSI and SPMI model typology both set 5 discrete points in the plot, whereas MPMI typologies just have one point. This is due to the fact that for the last typology all of the data universe had to be used, resulting in just one possible operating point: all sensors at all TUS beam intensities.

ensemble.

A second experience consisted in contaminating the data used to train the model using samples from a Gaussian process (Gaussian noise), following the methodology detailed in Section (4.3.2). Training models with independently contaminated data sets can act as a negative correlation agent in the ensemble and, at the same time, assess the robustness of the modeling approach. The results regarding the performance enhancements obtained from the use of the various ensemble methods are illustrated in Figure (6.4).

Regarding the first two models typologies (SPSI and SPMI) the enhancements obtained using NDEO and ES were surprisingly high, while the SA kept walking an oscillatory path, exhibiting a curve with contants negative performances. However concerning the last model typology considered (MPMI), all methods fail to provide any kind of performance enhancement. This observation is partly explained by the increased modeling task complexity when we join the spatial dimension together with the TUS beam intensity dimension. As a consequence, the volume of data available was increased, which in turn migrates the decorrelation effect brought to the ensemble by the contamination process. Increasing the volume of the data weakens this decorrelation agent, whose ability to increase the ambiguity levels among the ensembles is highly affected. Another

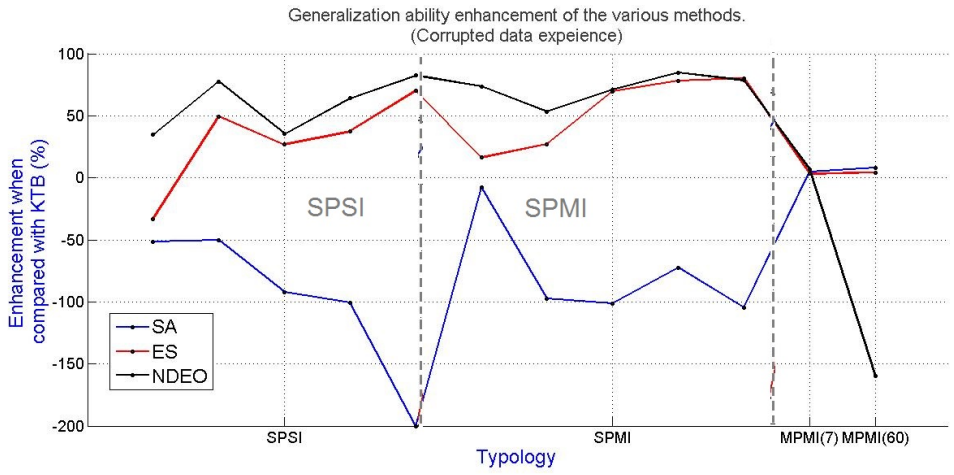

FIGURE 6.4: Ensemble methods performance enhancements of the various methods when compared with the KTB paradigm, regarding experiences where the models were build using uncorrupted data. SPSI and SPMI model typology both set 5 discrete points in the plot, whereas MPMI typologies just have one point. This is due to the fact that for the last typology all of the data universe had to be used, resulting in just one possible operating point: all sensors at all TUS beam intensities.

consequence of hardening the modeling task is success rate that one can consistently build good models. It was studied in Chapter (2) that for an ensemble to be successful, two conditions must be met: the individuals performances of the models must be comparable and similar; and the ambiguity level among the ensemble must be high, i.e. the models should be uncorrelated. Hardening the modelling task makes it more difficult to consistently build comparable models. This last fact partly explains the performance mitigation observed in the last model typology. Figure (6.4) shows that the enhancements performances dropped to the 0% line, a result consistent with the previous observations, since a 0% performance basically means the method is doing nothing in the system, besides introducing unjustified overhead in the system.

Combining models indeed consists of a hard task that requires a great deal of effort and art from the designer. this methods should only be applied if the consequent performance improvements justify the additional effort.

6.3 Concluding remarks

This thesis intended to study the feasibility in applying an innovative approach to estimate the temperature propagation during thermal therapies, in a non invasive way. As indicated, the reference in this field is imposed by the temperature resolution obtained with MRI techniques, 0.5 $^{o}C/cm^{3}$. It was proposed to estimate the temperature evolution using predictive models using b-splines neural networks evolved by the ASMOD algorithm.

Initially the data used to construct the models was characterized to provide the reader the possibility to assess if the data is trustworthy and representative of the physical phenomena intended to model. The modelling environment complexity was gradually increased which resulted in three different models typologies: SPSI, MPSI, MPMI. For each one of the different typologies the relevant features to be taken as input variables were defined along with the network structures associated with the typology.

Ensembles of neural networks were also studied in an attempt to enhance the prediction accuracy of the system. Three methods were assessed:

- Simple average (SA). The average of the individual predictions is taken as the final prediction.
- Evolutionary strategy (ES). Again the average of the individual predictions is taken as the final input however each individual network N_i is affected by a weight ω_i. The weight vector ω was evolved by using a evolutionary strategy.
- Neural dynamic ensemble optimization (NDEO). This thesis proposed an alternative method to combine the individual predictions by altering the combining mechanism. A second layer formed by a b-spline neural network takes all the individual predictions as inputs, $o_1 \dots o_N$ where N is the ensemble size and generates an output o_f, which is taken as the final prediction.

A clear division was made between the heating and cooling dynamics involved in a typical thermal therapy. This division resulted in the creation of two distinct models that model

the two different dynamics observed. Two experiments were always considered regarding the data used for training, validation and testing:

- Uncorrupted data. This data set is composed of the original data collected in the conditions exposed in this work.
- Corrupted data. After a contamination process, where Gaussian noise was added to the original set, the corrupted data was used to train and validate the models.

Using corrupted data to train and validate the models provides two different analysis perspectives. For one side the robustness of the system was assessed and it helps the designer to ascertain if the structure modelling power is in adequate level for the task. This last assessment is possible by observing the model behaviour in the test set. Ideally the model should only learn the dynamics of the phenomena intended to model and filter all external dynamics derived from the various possible noise sources. On the other side it alleviates the need for acquiring high quality data, which can only be captured using an invasive technique. A reliable temperature estimation method can be used to collect all the data needed to create models of complex environments.

Several models were developed for estimating the temperature curves in a non invasive way. We found that the modeling approach applied was capable of providing highly accurate predictive models with maximum absolute errors in the test case less than $0.33\ ^{o}C$, this is, below the $0.33\ ^{o}C$ threshold. This observation holds in the experiments using Gaussian contaminated data, which evidences the robustness of the approach. A second crucial observation is that the performance figures obtained remain comparable when the modeling environment complexity is increased, suggesting a modelling approach with the desirable scalability.

The performance figures were obtained using relatively simple models, which might be crucial for applications with scarce resources or that require real time responses. It was observed the average model complexity evolved at a slow pace with the modelling environment complexity, which means the system complexity can be managed as the environment approaches ideal conditions.

Combining BSNNs by forming neural network ensembles creates a potential performance enhancement mechanism, if the designing is appropriated. However we noted that a great deal of effort by the designer is needed to create the favorable conditions on which combining individual forecasting entities might pay off.

When compared to the state of art, the BSNN structures over-perform the maximum absolute error obtained using MRI, which is a very impressive result. Obviously the environments on which MRI operates are far more complex than the ones studied in this work. However we observed a modelling approach with very good indicators concerning scalability in response to increases in the complexity of the modeling environment. Together with neural network ensemble methods the systems can be forced to be more accurate and robust. We conclude that the approach followed in this thesis is feasible, and future research is highly recommended.

6.4 Reflections about the solution derived and AI

Over the last years artificial intelligence has been extensively to solve problems from a wide diversity of areas. Biological mimicry like neural networks (NN) or evolutionary computing (EC) are attractive concepts. However, one should bear in mind that despite the attractiveness present in these concepts, they are very naive attempts to model human biological mechanisms. We do not fully comprehend the deep essence behind the learning ability of our brains, neither we understand how evolution works at molecular level. As a result both NN or EC consists of *bulldozer* forms of intelligence, i.e. they work by brute force. Lately we observe a tendency to extensively apply this techniques to a variety of problems and test it if the mechanism is suitable to the problem, which consists of a bad practice. From an engineering perspective, the methodology applied when solving a problem should start by formulating a proper problem *definition* followed by the derivation of a suitable *representation*, so then we can approach the problem by deriving an algorithm or a mechanism to solve it. During this work we tried to follow this methodology, an effort was made to define and represent the problem in an convenient way, then we proposed an approach to solve the TSF problem[4].

[4]Time Series Forecasting (TSF)

Also important when facing a successfully approach, is to be able to see where the credit lies. Were neural networks the key ingredient in this work? Assuredly not. The credit relies almost entirely on the properties of B-splines and their amazing function approximating power. These mathematical constructions coined by *Isaac Jacob Schoenberg*, make use of their local control to fit curves with a great deal of flexibility. Concerning neural networks, one can argue that NN were useful in proving a suitable structure for the adaptation of the control points of the splines (weights) to approximate the temperature propagation function. A typical *hill climbing* problem, which could be solved by a variety of mechanisms. Nevertheless neural networks undoubtedly are a powerful tool with a wide range of potential applications. Note that the models were built without any knowledge about the underlying physics that govern the phenomena of temperature propagation, which was allowed by the learning paradigm conceptualized in neural networks.

As a final remark over AI, personally I believe that we should address the question: "What if God was an engineer?". After all humans have a remarkable good problem-solving ability. By deeply understanding the steps involved when we solve a particular problem, it is possible to model our intelligence and create powerful algorithms. Therefore I believe the future of AI will move away from bulldozer intelligence, moving towards to resemble and mimic the incredibly efficient algorithms used by our brain in a more intimate way.

6.5 Future research

To conclude this work a few future research guide lines can be pointed out. We believe the feasibility of the modelling approach has been proved to an extend that is reasonable to classify it as appropriate for the problem at hand. The next steps include:

- Increase the modeling environment complexity, where ideally the environment conditions match perfectly the human tissue ones.
- Apply the ensemble decorrelation method, that was derived but not employed, in Section (2.8.2.3).

- Use a different medium to acquire data, besides from the homogeneous phantom exposed in Chapter (3). Ideally real living tissues should be used.

Data extraction should be done in environments that resemble the conditions found on human tissue in a more intimate way. This will result in more realistic models that can indeed be used in biomedical instrumentation practice. Regarding the neural network ensembles techniques we suggest applying the work derived in Section (2.8.2.3), which we believe that can be used to further increase the ambiguity levels among the ensemble. Furthermore different temperature estimation methods can be tested.

Nevertheless we believe the extension of the models typologies to $2-D$ and $3-D$ spatial dimensions should be the immediate step. These models with an increased complexity would provide the last assessment regarding the feasibility of the intelligent modelling approach proposed to estimate the evolution of temperature during thermal therapies. This extension requires one more additional input to the network, for a $2-D$ model typology, and two additional inputs in the $3-D$ case. In Chapter(4) we derived the following expression to calculate the input variable that informs the network of the operating position in the $1-D$ space, formed by the line of sensors:

$$\theta_i = \arctan\left(\frac{D}{N_i * 5mm - 5mm}\right)$$

Where N_i is the number of the operating sensor and D is the distance from the line formed by the array of sensors which is parallel to the face of the transducer. The extension to a $3-D$ space can trivially be done by using *spherical coordinates*, Figure (6.5).

After structuring these models typologies, their generalization ability should be assessed. If the results are satisfactory we can proceed to gather an extensive amount of real data that would be used to train $3-D$ typology models that biomedical applications can benefit from.

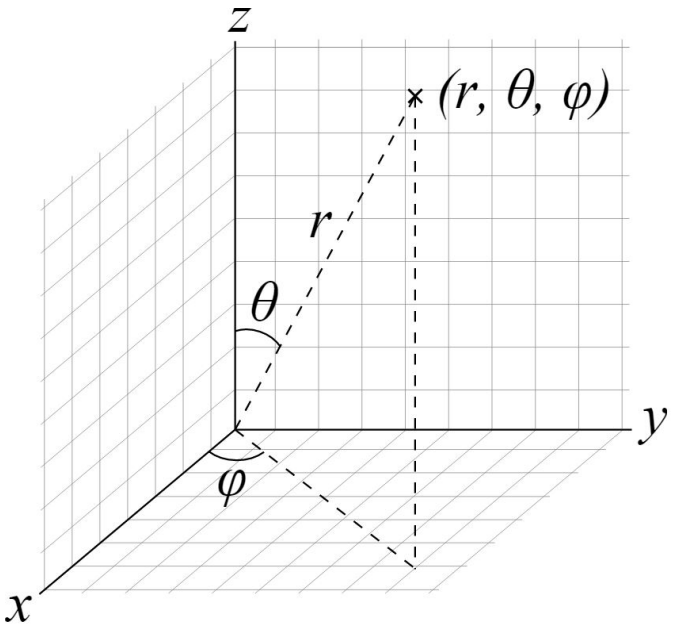

FIGURE 6.5: Spherical coordinates (r, θ, φ) as commonly used in physics: radial distance r, polar angle θ (theta), and azimuthal angle ϕ (phi). Taken from [3]

A

B-splines under the light of the convolution operation.

We define the **convolution** of two functions $f(t)$ and $g(t)$ as:

$$(f * g)(t) = \int_{-\infty}^{\infty} f(s)g(t-s)ds \tag{A.1}$$

Let us define also a *base* function as a rectangular window given by:

$$base(t) = \begin{cases} 1 & 0 \leq t < 1 \\ 0 & otherwise \end{cases} \tag{A.2}$$

Which is analogous to the first order b-spline $B_{i,1}$, defined in Chapter (2). Any piecewise constant function can be given by the following linear combination:

$$p(t) = \sum_{i=-\infty}^{\infty} p_i base(t) \tag{A.3}$$

With $p_i \in \mathbb{R}$ being the control points.

A B-spline basis function of degree n can be obtained by convolving a BS of degree $n-1$ with the rectangular function *base*. To exemplify this process, consider the case of a BS of order 2, given by:

$$B_2 = \int_{-\infty}^{\infty} base(s)base(t-s)ds \tag{A.4}$$

This operation is illustrated in Figure(A.1).

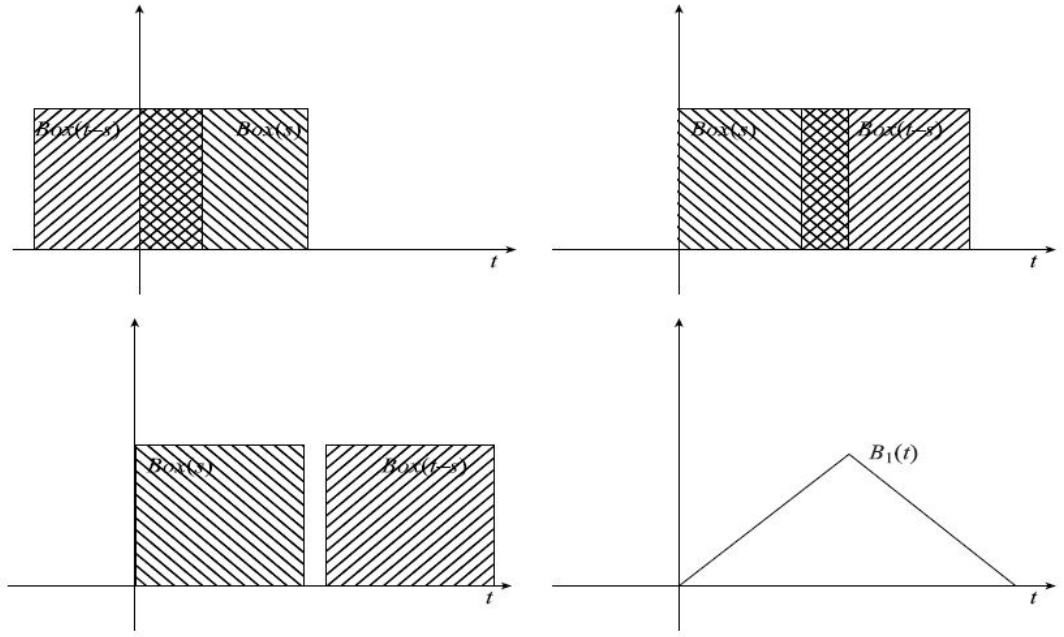

FIGURE A.1: Defining a BS of order 2 through the use of convolution operation. Adapted from [9]

The remarkable property of convolution is that each time a function is convolved with a *base* function, its smoothness increases. This process can then be seen as a *moving average* operation.

This definition of B-splines agrees with the properties derived before:

- $B_k(t)$ is a piecewise polynomial of order k. Each convolution increases the degree by 1.
- $B_k(t)$ has a support of length k.

- $B_k(t)$ is C^{k-1} continuous. Note that $base(t)$ is C^0 continuous, each convolution increases the smoothness by one unit.

Constructing B-splines basis functions using this procedure can be useful to derive insight about the construction process and properties of this curves. However, for practical implementations *Cox* [21] algorithm is used.

B

Data division concerning MPMI model typology.

This appendix characterizes the data division scheme applied in the construction of models belonging to MPMI typology.

Case 1:

Model 1 (2 lags):

- Test: Sensor 3 $1.5W/cm^2$, Sensor 5 $1.8W/cm^2$.
- Validation: Sensor 2 $1.5W/cm^2$, Sensor 4 $0.5W/cm^2$, Sensor 4 $1.5W/cm^2$, Sensor 5 $1.5W/cm^2$.
- Training: the remainder.

Model 2 (3 lags):

- Test: Sensor 2 $1.5W/cm^2$, Sensor 5 $1.8W/cm^2$.

- Validation: Sensor 2 $1.0W/cm^2$, Sensor 3 $0.5W/cm^2$, Sensor 3 $1.5W/cm^2$, Sensor 4 $1.5W/cm^2$.
- Training: the remainder.

Model 3 (4 lags):

- Test: Sensor 2 $0.5W/cm^2$, Sensor 5 $0.5W/cm^2$.
- Validation: Sensor 3 $1.0W/cm^2$, Sensor 4 $1.5W/cm^2$, Sensor 5 $1.0W/cm^2$, Sensor 5 $1.8W/cm^2$.
- Training: the remainder.

Model 4 (5 lags):

- Test: Sensor 3 $0.5W/cm^2$, Sensor 5 $1.8W/cm^2$.
- Validation: Sensor 2 $1.0W/cm^2$, Sensor 3 $1.0W/cm^2$, Sensor 3 $1.5W/cm^2$, Sensor 4 $1.5W/cm^2$.
- Training: the remainder.

Case 2:

Model 1 (3 lags):

- Test: Sensor 3 $1.5W/cm^2$, Sensor 5 $1.8W/cm^2$.
- Validation: Sensor 2 $1.5W/cm^2$, Sensor 4 $0.5W/cm^2$, Sensor 4 $1.5W/cm^2$, Sensor 5 $1.5W/cm^2$.
- Training: the remainder.

Model 2 (4 lags):

- Test: Sensor 2 $1.5W/cm^2$, Sensor 5 $1.8W/cm^2$.
- Validation: Sensor 2 $1.0W/cm^2$, Sensor 3 $0.5W/cm^2$, Sensor 3 $1.5W/cm^2$, Sensor 4 $1.5W/cm^2$.

- Training: the remainder.

Model 3 (5 lags):

- Test: Sensor 2 $0.5W/cm^2$, Sensor 5 $0.5W/cm^2$.
- Validation: Sensor 3 $1.0W/cm^2$, Sensor 4 $1.5W/cm^2$, Sensor 5 $1.0W/cm^2$, Sensor 5 $1.8W/cm^2$.
- Training: the remainder.

Model 4 (6 lags):

- Test: Sensor 3 $0.5W/cm^2$, Sensor 5 $1.8W/cm^2$.
- Validation: Sensor 2 $1.0W/cm^2$, Sensor 3 $1.0W/cm^2$, Sensor 3 $1.5W/cm^2$, Sensor 4 $1.5W/cm^2$.
- Training: the remainder.

C

Extended results obtained with respect to SPSI typology models.

This appendix covers the remaining modeling environments regarding SPSI typology models, extending the results exposed in Section(**??**). The following operating points are covered:

- $1.8W/cm^2$, Sensor 1
- $0.5W/cm^2$, Sensor 2
- $1.5W/cm^2$, Sensor 3
- $1.8W/cm^2$, Sensor 4
- $1.0W/cm^2$, Sensor 5

Model environment: TUS Intensity ($1.8W/cm^2$), Sensor (1)

We advance to model the dynamics concerning the data collected at Sensor 1, with a TUS beam intensity of $1.8W/cm^2$. Note that this environment holds the temperature curve with the highest slope, since this sensor was the closest to the TUS device and $1.8W/cm^2$ is the stronger beam intensity considered.

Using the original data, plotted in Figure (C.1), four distinct models were created. The 70/30/10 data division scheme was one more applied. Four models with a number of lags ranging between 2 and 5 were built, whose performance descriptors are presented in Table (C.1). The output of the top performer model in the validation test (model 3), is shown in Figure (C.2).

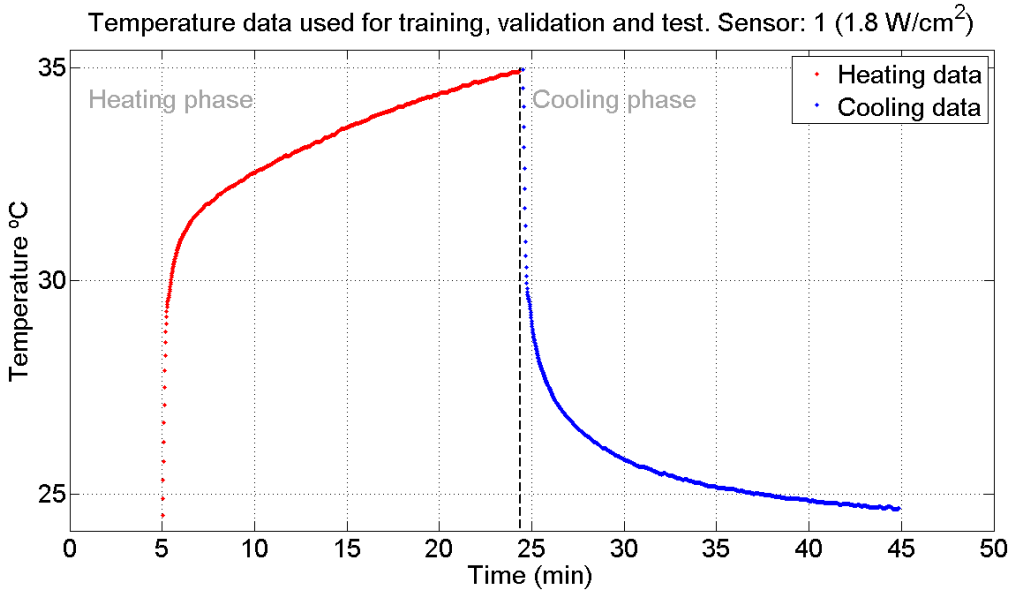

FIGURE C.1: Uncorrupted original data set used for SPSI model training, validation and test. Collected from the *homogeneous phantom* experimental setup. TUS intensity: $1.8W/cm^2$. Sensor 1.

TABLE C.1: Performance descriptors obtained concerning models with different number of input lags. The models presented were selected using the KTB approach. SPSI, Sensor 1 (1.8 W/cm^2). Data used: uncorrupted.

Model 1(2 lags)		
Criterion	Heating	Cooling
BIC	-6791	-6446
MSE	2.2471e-04	6.7879e-04
$MSRE$	7.8987e-06	2.2620e-05
MSE_v	2.0730e-03	3.6516e-03
$MSRE_v$	6.5780e-05	1.2737e-04
MSE_t	2.3620e-03	2.2251e-03
$MSRE_t$	7.2190e-05	8.4658e-05
M_{ae}	0.4954	0.4899
LWN	7	9
SR	e	e

Model 2(3 lags)		
Criterion	Heating	Cooling
BIC	-7221	-7017
MSE	1.7294e-04	2.5843e-04
$MSRE$	6.0751e-06	8.6586e-06
MSE_v	6.4084e-04	2.4284e-03
$MSRE_v$	1.9587e-05	8.5608e-05
MSE_t	1.8174e-03	9.8672e-04
$MSRE_t$	5.4516e-05	3.8387e-05
M_{ae}	0.1686	0.4785
LWN	7	13
SR	e	e

Model 3(4 lags)		
Criterion	Heating	Cooling
BIC	-6760	-8761
MSE	2.5599e-04	3.2923e-05
$MSRE$	9.3108e-06	1.1814e-06
MSE_v	7.1296e-04	5.9979e-04
$MSRE_v$	2.1932e-05	2.2135e-05
MSE_t	2.1568e-03	5.0391e-04
$MSRE_t$	6.5406e-05	1.9619e-05
M_{ae}	0.2806	0.2686
LWN	7	11
SR	e	e

Model 4(5 lags)		
Criterion	Heating	Cooling
BIC	-8863	-7076
MSE	1.9477e-05	2.1771e-04
$MSRE$	6.1079e-07	7.0953e-06
MSE_v	3.9683e-04	9.8528e-04
$MSRE_v$	1.2028e-05	3.5972e-05
MSE_t	1.1031e-03	8.0515e-04
$MSRE_t$	3.3125e-05	3.1135e-05
M_{ae}	0.1347	0.3230
LWN	11	11
SR	n	e

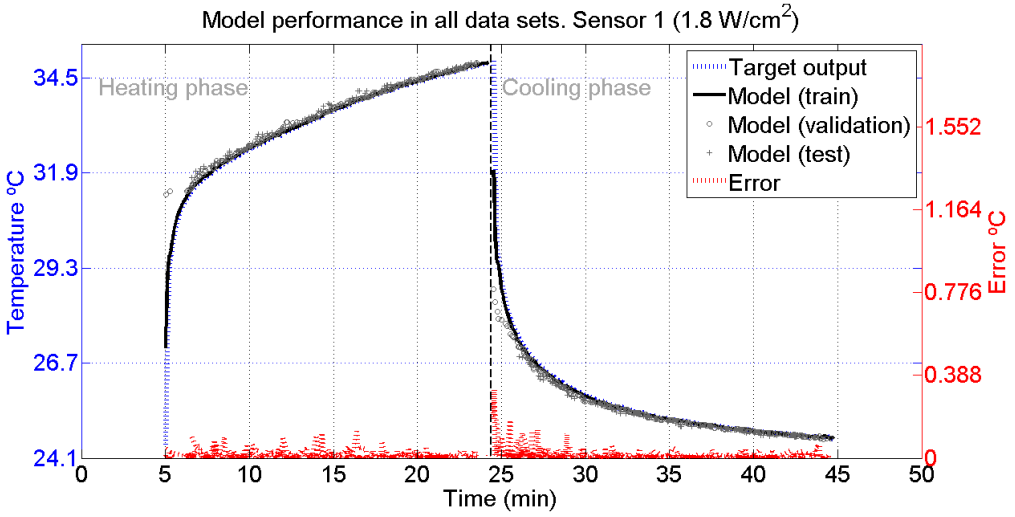

FIGURE C.2: Output curve of model 3 through the whole data set, selected using the KTB approach. The blue line represents the desired behaviour and the model's training output is given by the black line. The error line is red, circle and cross markers represents the model's validation and test output respectively. SISP,*homogeneous phantom* experimental setup. TUS intensity: 1.8W/cm^2. Sensor 1. Data used: uncorrupted.

The model's output curve exhibits a behaviour close to desired in the three data sets. The region where the error is more accentuated is concentrated in the starting region of the cooling model. This could be explained due to the clipping temperature fall registered in this zone, a more challenging dynamic for the model to learn. Nevertheless a 7 ^oC fall was registered in this zone, thus errors around 0.38 ^oC are assuredly reasonable under this conditions.

The application of the ensemble methods followed the KTB paradigm. The four previous models were combined using three different methods, as discussed in Section (4.4). The performance criteria obtained using this combination schemes are reflected in Table (C.2), whereas Table (C.3) compares this previous results with the KTB approach.

From this results exposed in this tables we can observe that the results resemble the ones obtained in the previous model environment. The ES optimized ensemble method managed to increase the generalization performance when a high level of correlation is present in the ensemble, due to the similarity among the training data sets used to construct the models. Howsoever, a KTB model with 5 lags outperforms any of the ensemble approaches, making the ensemble overhead in the system unjustified.

Following this results the data set was once more corrupted, Figure (C.3), and the experience was repeated.

70% of the noisy data was used in the training phase and 30% in the validation phase. The complete original uncurrupted data set was used for testing. The four KTB models constructed performed in compliance with Table (C.4), and Figure (C.4) illustrates model's 1 performance through all the data sets. Observe that even with the presence of noise the models managed to learn the process dynamics. Model 2 kept the error under 0.3 ^oC in both phases.

Moving to the ensembles approaches, the generalization error is presented in Table (C.5) and comparison with KTB models is done in Table (C.6).

Inspecting the KTB comparison table it is clear that the NDEO approach outperformed all of the other ensembles methods consistently as expected. The coherency of this results should be assessed during the analysis of the next modelling environments.

TABLE C.2: Performance comparison between all methodologies employed. SPSI, Sensor 1 (1.8 W/cm^2). Data used: uncorrupted.

Model 1	Heating			Cooling		
Criterion	MSE_v	MSE_t	M_{ae}	MSE_v	MSE_t	M_{ae}
KTB	2.0730e-03	2.3620e-03	0.4954	3.6516e-03	2.2251e-03	0.4899
Ensemble (SA)	2.0406e-03	2.1188e-03	0.4954	2.9760e-03	1.7309e-03	0.5816
Ensemble optimized (ES)	2.0251e-03	1.9398e-03	0.4954	2.8938e-03	1.6796e-03	0.5801
NDEO	2.0352e-03	1.7070e-03	0.4927	3.0915e-03	1.7240e-03	0.5390

Model 2	Heating			Cooling		
Criterion	MSE_v	MSE_t	M_{ae}	MSE_v	MSE_t	M_{ae}
KTB	6.4084e-04	1.8174e-03	0.1686	2.4284e-03	9.8672e-04	0.4785
Ensemble (SA)	5.9821e-04	1.6223e-03	0.1701	2.2143e-03	9.7272e-04	0.4867
Ensemble optimized (ES)	5.7131e-04	1.4663e-03	0.1738	1.9611e-03	9.8346e-04	0.4867
NDEO	5.5763e-04	1.2953e-03	0.1758	2.4271e-03	9.8698e-04	0.4783

Model 3	Heating			Cooling		
Criterion	MSE_v	MSE_t	M_{ae}	MSE_v	MSE_t	M_{ae}
KTB	7.1296e-04	2.1568e-03	0.2806	5.9979e-04	5.0391e-04	0.2686
Ensemble (SA)	6.3139e-04	1.9486e-03	0.4021	4.7517e-04	5.1349e-04	0.1351
Ensemble optimized (ES)	6.3751e-04	1.7850e-03	0.3184	3.7418e-04	4.9101e-04	0.1234
NDEO	7.1579e-04	2.1577e-03	0.2770	5.9968e-04	5.0371e-04	0.2685

Model 4	Heating			Cooling		
Criterion	MSE_v	MSE_t	M_{ae}	MSE_v	MSE_t	M_{ae}
KTB	3.9683e-04	1.1031e-03	0.1347	9.8528e-04	8.0515e-04	0.3230
Ensemble (SA)	4.5530e-04	1.3270e-03	0.1456	8.8166e-04	8.1418e-04	0.3252
Ensemble optimized (ES)	4.2602e-04	1.2131e-03	0.1408	6.7043e-04	7.7236e-04	0.4045
NDEO	4.0233e-04	1.1350e-03	0.1339	1.8668e-03	8.6399e-04	0.3824

TABLE C.3: Generalization error obtained with the ensemble approaches in comparison with the KTB model selection. SPSI, Sensor 1 (1.8 W/cm^2). Data used: uncorrupted.

Model 1 (2 Lags)	**Heating**	**Cooling**	**Average comparison**
Ensemble (SA)	10.29 %	22.21 %	16.25 %
Ensemble optimized (ES)	17.88 %	24.52 %	21.20 %
NDEO	27.73 %	22.52 %	25.12 %

Model 2 (3 Lags)	**Heating**	**Cooling**	**Average comparison**
Ensemble (SA)	10.73 %	1.42 %	6.08 %
Ensemble optimized (ES)	19.31 %	0.33 %	9.82 %
NDEO	28.73 %	-0.03 %	14.35 %

Model 3 (4 Lags)	**Heating**	**Cooling**	**Average comparison**
Ensemble (SA)	9.65 %	-1.90 %	3.88 %
Ensemble optimized (ES)	17.24 %	2.56 %	9.90 %
NDEO	-0.04 %	0.04 %	-0.00 %

Model 4 (5 Lags)	**Heating**	**Cooling**	**Average comparison**
Ensemble (SA)	-20.30 %	-1.12 %	-10.71 %
Ensemble optimized (ES)	-9.97 %	4.07 %	-2.95 %
NDEO	-2.89 %	-7.31 %	-5.10 %

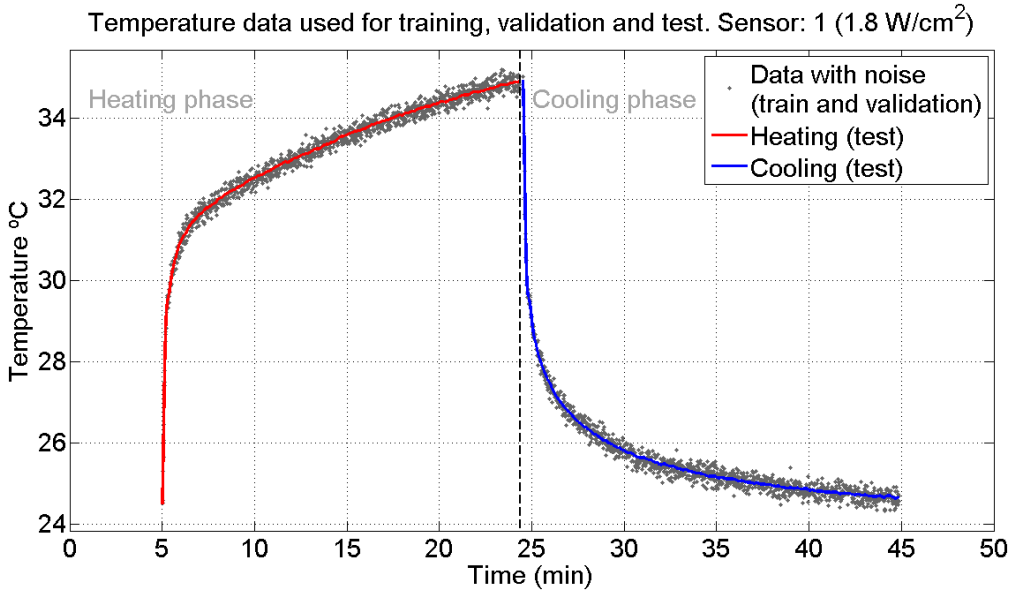

FIGURE C.3: Corrupted data after the addition of random Gaussian noise. The noisy data is used for training and validation. The Uncorrupted, noise free data is used to test the model. Collected from the *homogeneous phantom* experimental setup. TUS intensity: $1.8W/cm^2$. Sensor 1.

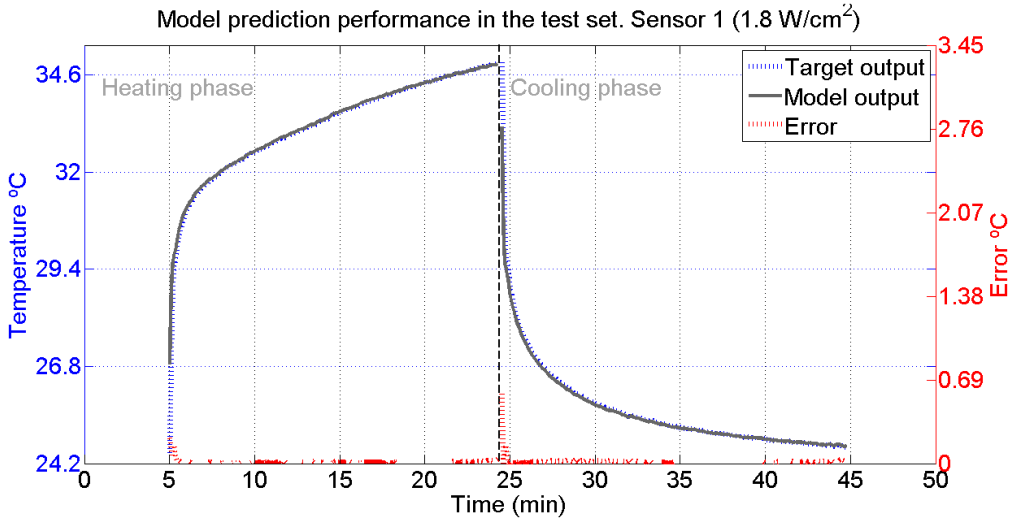

FIGURE C.4: Behaviour of model 1 in the test set, selected using the KTB approach. The blue line represents the desired behaviour and the model's test output is given by the black line. The error line is red. SISP,*homogeneous phantom* experimental setup. TUS intensity: $1.0 W/cm^2$. Sensor 1.

TABLE C.4: Performance descriptors obtained concerning models with different number of input lags. The models presented were selected using the KTB approach. SPSI, Sensor 1 (1.8 W/cm^2). Data used: corrupted.

Model 1(2 lags)		
Criterion	Heating	Cooling
BIC	-2670	-2918
MSE	5.0467e-02	3.7290e-02
$MSRE$	1.5235e-03	1.4579e-03
MSE_v	5.7721e-02	3.6602e-02
$MSRE_v$	1.7917e-03	1.4361e-03
MSE_t	1.3214e-03	1.5154e-03
$MSRE_t$	4.4962e-05	5.9574e-05
M_{ae}	0.6046	0.1516
LWN	7	10
SR	n	n

Model 2(3 lags)		
Criterion	Heating	Cooling
BIC	-2987	-3173
MSE	3.1574e-02	2.9984e-02
$MSRE$	9.5135e-04	1.1697e-03
MSE_v	3.6938e-02	3.6449e-02
$MSRE_v$	1.1221e-03	1.4192e-03
MSE_t	4.9403e-04	7.9429e-04
$MSRE_t$	1.6678e-05	3.0305e-05
M_{ae}	0.3039	0.1633
LWN	11	13
SR	n	n

Model 3(4 lags)		
Criterion	Heating	Cooling
BIC	-2870	-3128
MSE	2.9364e-02	2.8613e-02
$MSRE$	8.8425e-04	1.1199e-03
MSE_v	3.3262e-02	3.2984e-02
$MSRE_v$	1.0029e-03	1.2897e-03
MSE_t	1.0691e-03	6.1345e-04
$MSRE_t$	3.6781e-05	2.0903e-05
M_{ae}	0.3352	0.4173
LWN	16	15
SR	n	n

Model 4(5 lags)		
Criterion	Heating	Cooling
BIC	-2964	-3137
MSE	2.7338e-02	2.5822e-02
$MSRE$	8.2509e-04	1.0125e-03
MSE_v	2.8911e-02	2.9427e-02
$MSRE_v$	8.7105e-04	1.1492e-03
MSE_t	3.9962e-04	7.6348e-04
$MSRE_t$	1.3465e-05	2.5392e-05
M_{ae}	0.2092	0.5751
LWN	20	20
SR	e	n

TABLE C.5: Performance comparison between all methodologies employed. SPSI, Sensor 1 (1.8 W/cm^2)

Model 1	Heating			Cooling		
Criterion	MSE_v	MSE_t	M_{ae}	MSE_v	MSE_t	M_{ae}
KTB	5.7721e-02	1.3214e-03	0.6046	3.6602e-02	1.5154e-03	0.1516
Ensemble (SA)	4.2733e-02	9.6271e-04	0.6046	2.3128e-02	5.9889e-04	0.1763
Ensemble optimized (ES)	4.1223e-02	9.7371e-04	0.6046	2.2774e-02	6.4189e-04	0.2395
NDEO	4.8945e-02	7.2289e-04	0.5208	2.3533e-02	8.7936e-04	0.3325

Model 2	Heating			Cooling		
Criterion	MSE_v	MSE_t	M_{ae}	MSE_v	MSE_t	M_{ae}
KTB	3.6938e-02	4.9403e-04	0.3039	3.6449e-02	7.9429e-04	0.1633
Ensemble (SA)	2.7269e-02	6.4787e-04	0.3688	2.5973e-02	5.8052e-04	0.1763
Ensemble optimized (ES)	2.6456e-02	6.5888e-04	0.3688	2.5459e-02	6.2356e-04	0.2395
NDEO	3.0951e-02	6.2310e-04	0.3594	2.5291e-02	5.7101e-04	0.4638

Model 3	Heating			Cooling		
Criterion	MSE_v	MSE_t	M_{ae}	MSE_v	MSE_t	M_{ae}
KTB	3.3262e-02	1.0691e-03	0.3352	3.2984e-02	6.1345e-04	0.4173
Ensemble (SA)	2.6018e-02	5.3087e-04	0.2856	2.5681e-02	5.7890e-04	0.1763
Ensemble optimized (ES)	2.6224e-02	5.4189e-04	0.2856	2.5167e-02	6.2196e-04	0.2395
NDEO	2.5770e-02	7.1164e-04	0.2532	2.5263e-02	4.2762e-04	0.3669

Model 4	Heating			Cooling		
Criterion	MSE_v	MSE_t	M_{ae}	MSE_v	MSE_t	M_{ae}
KTB	2.8911e-02	3.9962e-04	0.2092	2.9427e-02	7.6348e-04	0.5751
Ensemble (SA)	2.2154e-02	4.6079e-04	0.2238	2.3771e-02	5.5384e-04	0.1390
Ensemble optimized (ES)	2.1988e-02	4.7182e-04	0.1364	2.3794e-02	5.9695e-04	0.2395
NDEO	2.3049e-02	3.5993e-04	0.1465	2.5410e-02	2.5918e-04	0.3496

TABLE C.6: Generalization error obtained with the ensemble approaches in comparison with the KTB model selection. SPSI, Sensor 1 (1.8 W/cm^2). Data used: corrupted.

Model 1 (2 Lags)	**Heating**	**Cooling**	**Average comparison**
Ensemble (SA)	27.15 %	60.48 %	43.81 %
Ensemble optimized (ES)	26.31 %	57.64 %	41.98 %
NDEO	45.29 %	41.97 %	43.63 %

Model 2 (3 Lags)	**Heating**	**Cooling**	**Average comparison**
Ensemble (SA)	-31.14 %	26.91 %	-2.11 %
Ensemble optimized (ES)	-33.37 %	21.50 %	-5.94 %
NDEO	-26.13 %	28.11 %	0.99 %

Model 3 (4 Lags)	**Heating**	**Cooling**	**Average comparison**
Ensemble (SA)	50.34 %	5.63 %	27.99 %
Ensemble optimized (ES)	49.31 %	-1.39 %	23.96 %
NDEO	33.43 %	30.29 %	31.86 %

Model 4 (5 Lags)	**Heating**	**Cooling**	**Average comparison**
Ensemble (SA)	-15.31 %	27.46 %	6.08 %
Ensemble optimized (ES)	-18.07 %	21.81 %	1.87 %
NDEO	9.93 %	66.05 %	37.99 %

Model environment: TUS Intensity ($0.5W/cm^2$), Sensor (2)

It is unpractical to present here in graphical/table form, the results obtained for all the environments considered (SISP, 5 sensors, 4 intensities, $5 \times 4 = 20$ environments). Therefore just one intensity for each sensor is presented to not over strain the reader. This section now deals with the data collected at Sensor 2, with a TUS beam intensity of $0.5W/cm^2$, the weakest intensity used.

Again the first experiment deals with the original data, shown in Figure (C.5). Due to the weak intensity waves ($0.5W/cm^2$), one can observe a short temperature range during the session. The temperature evolution was smooth, without abrupt changes. It is expected the model's output to follow the desired behaviour without difficulties due to the smooth dynamics present in the target data.

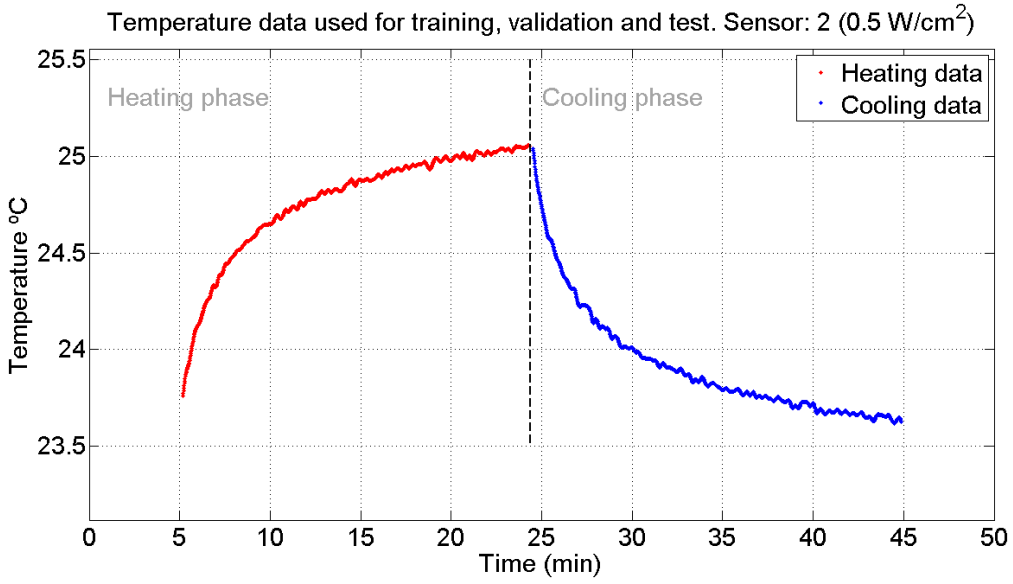

FIGURE C.5: Uncorrupted data set used for SPSI model training, validation and test. Collected from the *homogeneous phantom* experimental setup. TUS intensity: $0.5W/cm^2$. Sensor 2.

In Table (C.7) the performance figures are shown for each one of the models considered.

Even with relatively simple models (model 1, LWN below 10) the performance descriptors obtained are satisfactory. The actual output of model 1 is confronted with the desired output in Figure (C.6).

TABLE C.7: Performance descriptors obtained concerning models with different number of input lags. The models presented were selected using the KTB approach. SPSI, Sensor 2 ($0.5\ W/cm^2$). Data used: uncorrupted.

Model 1(2 lags)		
Criterion	Heating	Cooling
BIC	-9683	-9841
MSE	8.3907e-06	1.0153e-05
$MSRE$	3.3908e-07	4.2321e-07
MSE_v	8.3087e-05	2.0447e-04
$MSRE_v$	3.3540e-06	8.4180e-06
MSE_t	3.9231e-04	3.1247e-04
$MSRE_t$	1.5896e-05	1.3004e-05
M_{ae}	0.1261	0.1079
LWN	8	7
SR	n	n

Model 2(3 lags)		
Criterion	Heating	Cooling
BIC	-9974	-9943
MSE	4.8377e-06	9.9041e-06
$MSRE$	1.9576e-07	4.1187e-07
MSE_v	2.8040e-04	1.7870e-04
$MSRE_v$	1.1450e-05	7.3778e-06
MSE_t	2.9018e-04	3.5948e-04
$MSRE_t$	1.1697e-05	1.4986e-05
M_{ae}	0.1141	0.1078
LWN	11	7
SR	n	n

Model 3(4 lags)		
Criterion	Heating	Cooling
BIC	-9508	-9449
MSE	6.3819e-06	1.1046e-05
$MSRE$	2.5920e-07	4.6035e-07
MSE_v	1.3821e-04	2.1642e-04
$MSRE_v$	5.6219e-06	8.9162e-06
MSE_t	3.0584e-04	2.2076e-04
$MSRE_t$	1.2368e-05	9.2289e-06
M_{ae}	0.0916	0.0935
LWN	11	7
SR	n	n

Model 4(5 lags)		
Criterion	Heating	Cooling
BIC	-9805	-9744
MSE	4.6846e-06	1.0216e-05
$MSRE$	1.8938e-07	4.2586e-07
MSE_v	1.8043e-04	1.9783e-04
$MSRE_v$	7.3163e-06	8.1850e-06
MSE_t	3.5063e-04	2.8460e-04
$MSRE_t$	1.4166e-05	1.1857e-05
M_{ae}	0.0883	0.0998
LWN	11	7
SR	n	n

The ensemble methods performance criteria are calculated in Table (C.8) and compared with the KTB approach in Table (C.9).

The observations resemble the previous ones concerning the original data experience. The ensemble optimized (ES) outperforms the others methods, and for this environment it even outperformed the KTB model with 5 lags, which was not verified in the last two environments. The ensemble optimized by an evolutionary strategy seems to appear as a very robust combination scheme when the ambiguity among the ensemble for some reason can't be forced to reasonable levels.

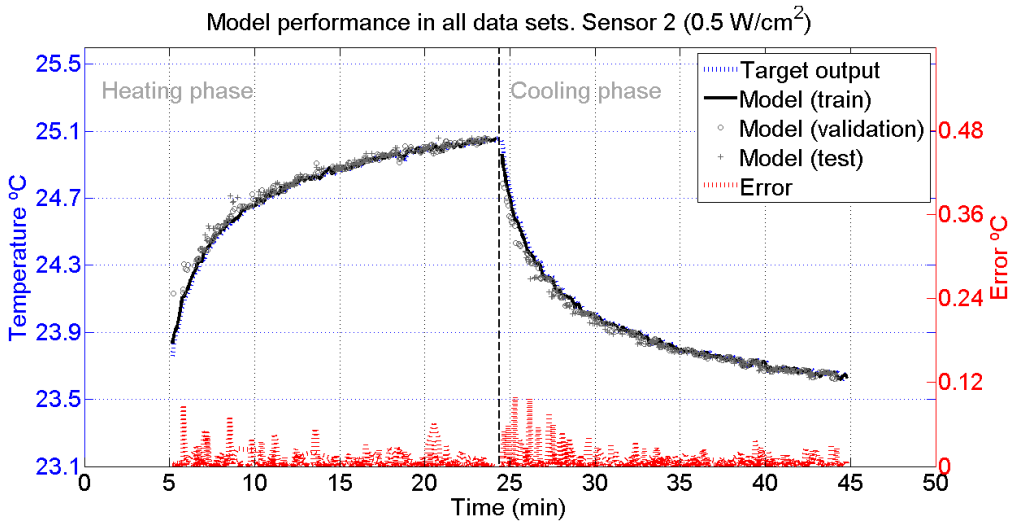

FIGURE C.6: Behaviour of the model 1 through the whole data set, selected using the KTB approach. The blue line represents the desired behaviour and the model's training output is given by the black line. The error line is red, circle and cross markers represents the model's validation and test output respectively. SISP,*homogeneous phantom* experimental setup. TUS intensity: $0.5W/cm^2$. Sensor 2. Data used: uncorrupted.

Passing to the second experience, the original data was corrupted, Figure (C.7), and used in the constructions of the models. One more the uncorrupted original data was left to be used to assess the generalization error of the models.

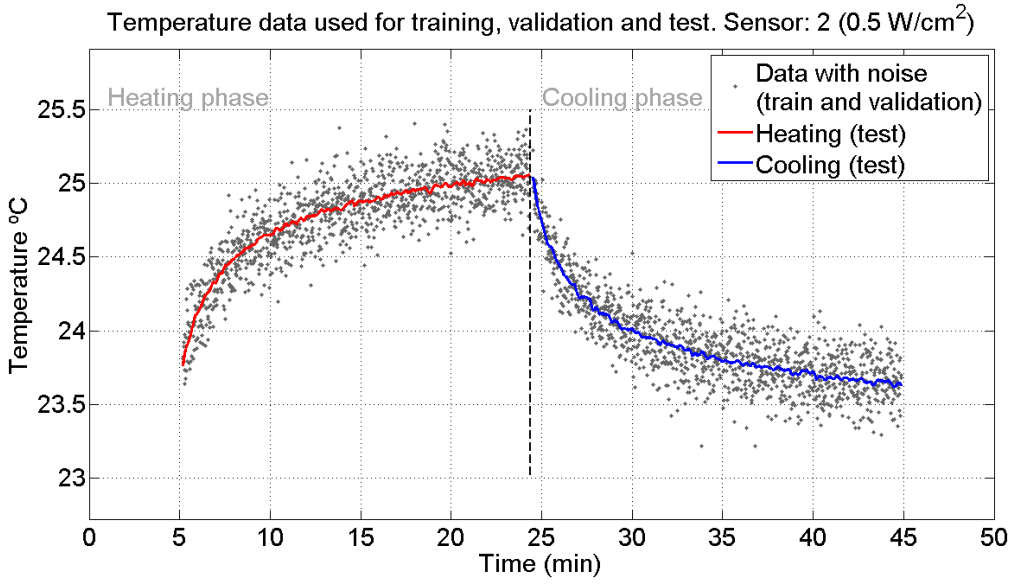

FIGURE C.7: Data after the addition of random Gaussian noise. The noisy data is used for training and validation. The Uncorrupted, noise free data is used to test the model. Collected from the *homogeneous phantom* experimental setup. TUS intensity: $0.5W/cm^2$. Sensor 2.

TABLE C.8: Performance comparison between all methodologies employed. SPSI, Sensor 2 (0.5 W/cm^2). Data used: uncorrupted.

Model 1	Heating			Cooling		
Criterion	MSE_v	MSE_t	M_{ae}	MSE_v	MSE_t	M_{ae}
KTB	8.3087e-05	3.9231e-04	0.1261	2.0447e-04	3.1247e-04	0.1079
Ensemble (SA)	9.5170e-05	4.2876e-04	0.1172	2.0348e-04	3.1160e-04	0.1079
Ensemble optimized (ES)	8.1716e-05	3.8777e-04	0.1232	1.8871e-04	2.9605e-04	0.1079
NDEO	1.0896e-04	4.7972e-04	0.1134	2.0447e-04	3.1274e-04	0.1080

Model 2	Heating			Cooling		
Criterion	MSE_v	MSE_t	M_{ae}	MSE_v	MSE_t	M_{ae}
KTB	2.8040e-04	2.9018e-04	0.1141	1.7870e-04	3.5948e-04	0.1078
Ensemble (SA)	2.6794e-04	2.6002e-04	0.1301	1.7813e-04	3.5917e-04	0.1076
Ensemble optimized (ES)	2.4644e-04	2.3305e-04	0.1301	1.6682e-04	3.3563e-04	0.1060
NDEO	2.8040e-04	2.9018e-04	0.1141	1.7884e-04	3.5990e-04	0.1078

Model 3	Heating			Cooling		
Criterion	MSE_v	MSE_t	M_{ae}	MSE_v	MSE_t	M_{ae}
KTB	1.3821e-04	3.0584e-04	0.0916	2.1642e-04	2.2076e-04	0.0935
Ensemble (SA)	1.3240e-04	2.8259e-04	0.0962	2.1681e-04	2.2146e-04	0.0936
Ensemble optimized (ES)	1.2942e-04	2.7634e-04	0.0962	2.0833e-04	2.0658e-04	0.0924
NDEO	1.3820e-04	3.0584e-04	0.0916	2.1631e-04	2.1975e-04	0.0941

Model 4	Heating			Cooling		
Criterion	MSE_v	MSE_t	M_{ae}	MSE_v	MSE_t	M_{ae}
KTB	1.8043e-04	3.5063e-04	0.0883	1.9783e-04	2.8460e-04	0.0998
Ensemble (SA)	1.5779e-04	3.0122e-04	0.0907	1.9899e-04	2.8494e-04	0.1002
Ensemble optimized (ES)	1.4051e-04	2.4495e-04	0.0927	1.8555e-04	2.6678e-04	0.0959
NDEO	1.8028e-04	3.5067e-04	0.0882	1.9782e-04	2.8459e-04	0.0998

To speed up the results presentation we omit the individual performance descriptors of each one of the four models, however an individual KTB model's output is presented Figure (C.8), which clearly shows a model capable of consistently keeping the error below 0.16 ^oC. Table (C.10) depicts the performance of all the approaches, KTB and ensemble, whereas Table (C.11) compares the ensemble generalization performance with the single model approach.

TABLE C.9: Generalization error obtained with the ensemble approaches in comparison with the KTB model selection. SPSI, Sensor 2 (0.5 W/cm^2). Data used: uncorrupted.

Model 1 (2 Lags)	**Heating**	**Cooling**	**Average comparison**
Ensemble (SA)	-9.29 %	0.28 %	-4.51 %
Ensemble optimized (ES)	1.16 %	5.25 %	3.21 %
NDEO	-22.28 %	-0.09 %	-11.18 %

Model 2 (3 Lags)	**Heating**	**Cooling**	**Average comparison**
Ensemble (SA)	10.39 %	0.09 %	5.24 %
Ensemble optimized (ES)	19.69 %	6.63 %	13.16 %
NDEO	0.00 %	-0.12 %	-0.06 %

Model 3 (4 Lags)	**Heating**	**Cooling**	**Average comparison**
Ensemble (SA)	7.60 %	-0.32 %	3.64 %
Ensemble optimized (ES)	9.65 %	6.42 %	8.03 %
NDEO	-0.00 %	0.46 %	0.23 %

Model 4 (5 Lags)	**Heating**	**Cooling**	**Average comparison**
Ensemble (SA)	14.09 %	-0.12 %	6.99 %
Ensemble optimized (ES)	30.14 %	6.26 %	18.20 %
NDEO	-0.01 %	0.00 %	-0.01 %

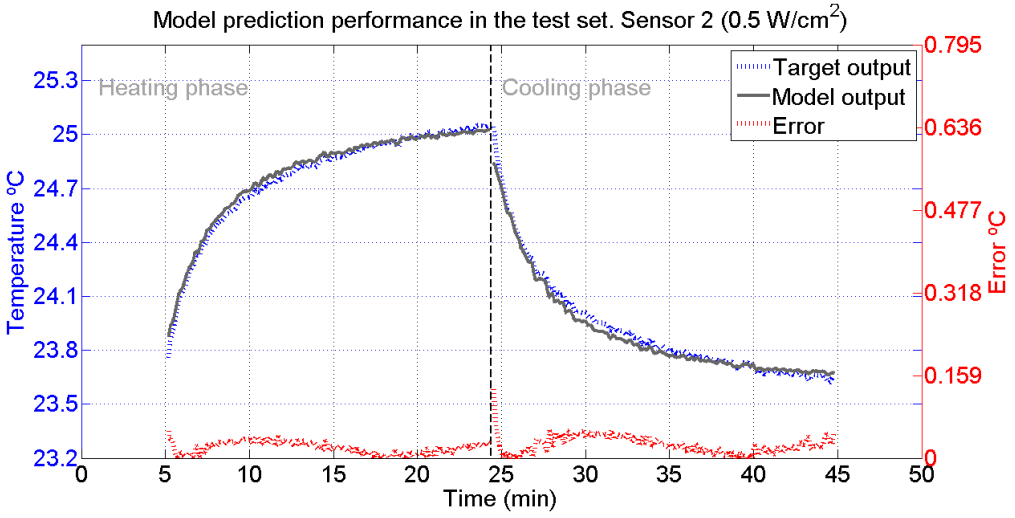

FIGURE C.8: Model's behaviour in the test set, selected using the KTB approach. The blue line represents the desired behaviour and the model's test output is given by the black line. The error line is red. SISP,*homogeneous phantom* experimental setup. TUS intensity: $0.5W/cm^2$. Sensor 2. Data used: corrupted.

Both NDEO and ES approach provided an interesting increases in the generalization ability of the predictive system, albeit the NDEO results were more noticeable. Nevertheless the major difference resides in the fact the the NDEO presented consistent and

TABLE C.10: Performance comparison between all methodologies employed. SPSI, Sensor 2 (0.5 W/cm^2). Data used: corrupted.

Model 1	Heating			Cooling		
Criterion	MSE_v	MSE_t	M_{ae}	MSE_v	MSE_t	M_{ae}
KTB	3.5841e-02	2.9191e-03	0.2343	3.3888e-02	3.2803e-03	0.1580
Ensemble (SA)	2.3659e-02	1.2563e-03	0.2343	2.2021e-02	1.4065e-03	0.1580
Ensemble optimized (ES)	2.3391e-02	1.9272e-04	0.2343	2.1709e-02	7.5176e-04	0.1580
NDEO	2.2994e-02	1.5556e-04	0.2041	2.1657e-02	1.3158e-04	0.0791

Model 2	Heating			Cooling		
Criterion	MSE_v	MSE_t	M_{ae}	MSE_v	MSE_t	M_{ae}
KTB	2.7263e-02	1.3652e-03	0.1093	3.3331e-02	1.3292e-03	0.1628
Ensemble (SA)	2.0816e-02	1.2096e-03	0.1629	2.5912e-02	1.3872e-03	0.1558
Ensemble optimized (ES)	1.9834e-02	1.4508e-04	0.1629	2.5749e-02	7.3191e-04	0.1558
NDEO	1.9903e-02	2.7594e-04	0.0664	2.5457e-02	2.6231e-04	0.1506

Model 3	Heating			Cooling		
Criterion	MSE_v	MSE_t	M_{ae}	MSE_v	MSE_t	M_{ae}
KTB	2.7125e-02	8.5288e-04	0.0506	2.7821e-02	8.4833e-04	0.1310
Ensemble (SA)	2.2336e-02	1.1875e-03	0.1042	2.3003e-02	1.3684e-03	0.1408
Ensemble optimized (ES)	2.0912e-02	1.2208e-04	0.1042	2.2829e-02	7.1257e-04	0.1408
NDEO	2.0560e-02	1.4694e-04	0.1212	2.3454e-02	2.3622e-04	0.0906

Model 4	Heating			Cooling		
Criterion	MSE_v	MSE_t	M_{ae}	MSE_v	MSE_t	M_{ae}
KTB	2.8055e-02	4.9664e-04	0.0514	2.9124e-02	8.9033e-04	0.1326
Ensemble (SA)	2.4454e-02	1.1791e-03	0.0847	2.5940e-02	1.3532e-03	0.1312
Ensemble optimized (ES)	2.3008e-02	1.1271e-04	0.0276	2.5483e-02	6.9686e-04	0.1160
NDEO	2.3016e-02	7.4112e-05	0.0261	2.5007e-02	1.3858e-04	0.1034

similar results in both heating and cooling phases, whereas the ES results were more oscillatory. The SA method lacked intelligence when combining the individual model's output and hence has failed to improve the generalization ability of the system. This last method continues to exhibit a highly oscillatory behaviour.

TABLE C.11: Generalization error obtained with the ensemble approaches in comparison with the KTB model selection.SPSI, Sensor 2 (0.5 W/cm^2). Data used: corrupted.

Model 1 (2 Lags)	**Heating**	**Cooling**	**Average comparison**
Ensemble (SA)	56.96 %	57.12 %	57.04 %
Ensemble optimized (ES)	93.40 %	77.08 %	85.24 %
NDEO	94.67 %	95.99 %	95.33 %

Model 2 (3 Lags)	**Heating**	**Cooling**	**Average comparison**
Ensemble (SA)	11.40 %	-4.36 %	3.52 %
Ensemble optimized (ES)	89.37 %	44.94 %	67.15 %
NDEO	79.79 %	80.27 %	80.03 %

Model 3 (4 Lags)	**Heating**	**Cooling**	**Average comparison**
Ensemble (SA)	-39.23 %	-61.30 %	-50.27 %
Ensemble optimized (ES)	85.69 %	16.00 %	50.84 %
NDEO	82.77 %	72.15 %	77.46 %

Model 4 (5 Lags)	**Heating**	**Cooling**	**Average comparison**
Ensemble (SA)	-137.41 %	-51.99 %	-94.70 %
Ensemble optimized (ES)	77.30 %	21.73 %	49.52 %
NDEO	85.08 %	84.44 %	84.76 %

Model environment: TUS Intensity ($1.5W/cm^2$), Sensor (3)

The environment that follows concerns the data collected at sensor 3, while appyling a TUS beam intensity of $1.5W/cm^2$. Starting with the uncorrupted data experiment, the data is presented in Figure (C.9) and the models assessment is made in Table (C.12). The output calculated for model 4 was chosen to be confronted with the target output, Figure (C.10).

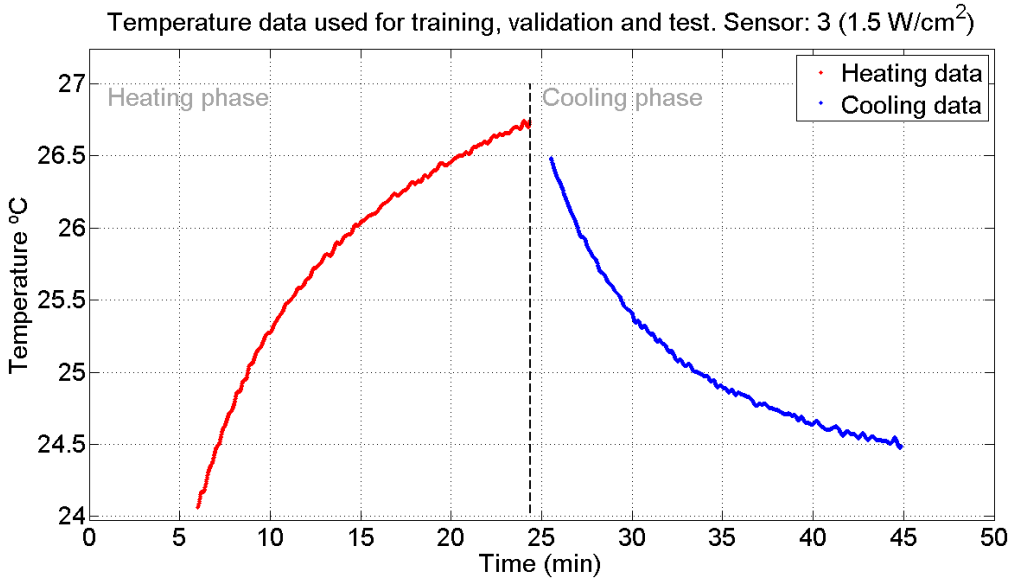

FIGURE C.9: Uncorrupted data set used for SPSI model training, validation and test. Collected from the *homogeneous phantom* experimental setup. TUS intensity: $1.8W/cm^2$. Sensor 1.

TABLE C.12: Performance descriptors obtained concerning models with different number of input lags. The models presented were selected using the KTB approach. SPSI, Sensor 3 (1.5 W/cm^2). Data used: uncorrupted.

Model 1(2 lags)		
Criterion	Heating	Cooling
BIC	-8849	-9131
MSE	1.2578e-05	1.1339e-05
$MSRE$	4.9196e-07	4.5182e-07
MSE_v	5.2066e-04	2.6519e-04
$MSRE_v$	2.0656e-05	1.0423e-05
MSE_t	8.5115e-04	5.6234e-04
$MSRE_t$	3.3523e-05	2.2224e-05
M_{ae}	0.1847	0.1085
LWN	8	7
SR	n	n

Model 2(3 lags)		
Criterion	Heating	Cooling
BIC	-8892	-9372
MSE	1.3840e-05	1.2625e-05
$MSRE$	5.4051e-07	5.0342e-07
MSE_v	3.1580e-04	2.0304e-04
$MSRE_v$	1.2405e-05	8.0318e-06
MSE_t	1.4739e-03	6.2714e-04
$MSRE_t$	5.7685e-05	2.4858e-05
M_{ae}	0.2047	0.0995
LWN	7	7
SR	n	n

Model 3(4 lags)		
Criterion	Heating	Cooling
BIC	-9122	-9301
MSE	9.6193e-06	9.5467e-06
$MSRE$	3.7757e-07	3.7755e-07
MSE_v	2.4512e-04	2.9941e-04
$MSRE_v$	9.5317e-06	1.1859e-05
MSE_t	9.9152e-04	6.9524e-04
$MSRE_t$	3.8679e-05	2.7411e-05
M_{ae}	0.1886	0.2012
LWN	11	11
SR	n	n

Model 4(5 lags)		
Criterion	Heating	Cooling
BIC	-8967	-8748
MSE	1.0870e-05	1.4025e-05
$MSRE$	4.2529e-07	5.5754e-07
MSE_v	2.6030e-04	1.9462e-04
$MSRE_v$	1.0222e-05	7.7023e-06
MSE_t	6.9525e-04	5.4587e-04
$MSRE_t$	2.7085e-05	2.1667e-05
M_{ae}	0.1313	0.0725
LWN	11	7
SR	n	n

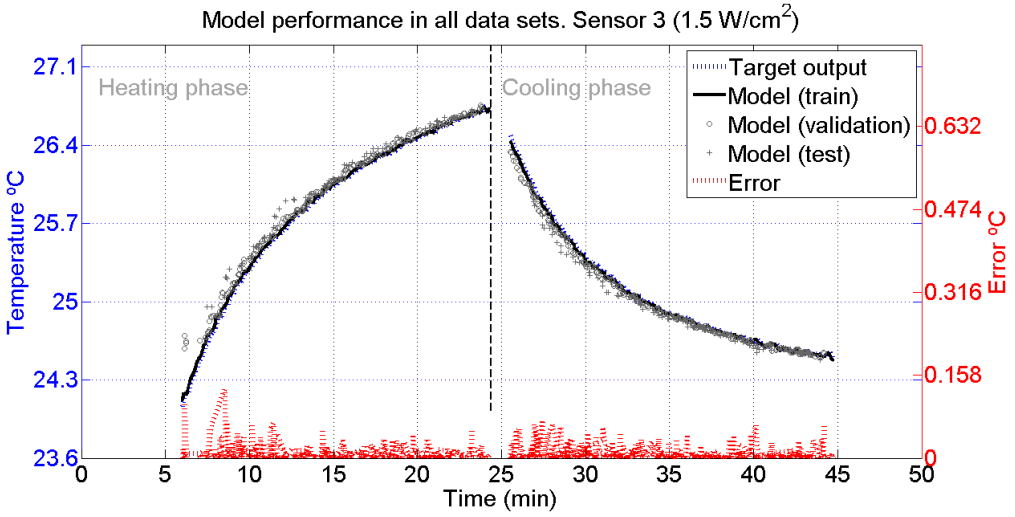

FIGURE C.10: Behaviour of the model through the whole data set, selected using the KTB approach. The blue line represents the desired behaviour and the model's training output is given by the black line. The error line is red, circle and cross markers represents the model's validation and test output respectively. SISP,*homogeneous phantom* experimental setup. TUS intensity: 1.5W/cm^2. Sensor 3.

TABLE C.13: Performance comparison between all methodologies employed. SPSI, Sensor 3 (1.5 W/cm^2). Data used: uncorrupted.

Model 1	Heating			Cooling		
Criterion	MSE_v	MSE_t	M_{ae}	MSE_v	MSE_t	M_{ae}
KTB	5.2066e-04	8.5115e-04	0.1847	2.6519e-04	5.6234e-04	0.1085
Ensemble (SA)	4.9962e-04	7.2831e-04	0.1847	2.5833e-04	5.4071e-04	0.1079
Ensemble optimized (ES)	4.9814e-04	7.1582e-04	0.1847	2.5703e-04	5.3684e-04	0.1078
NDEO	4.9746e-04	7.0383e-04	0.1845	2.8227e-04	5.7111e-04	0.1066

Model 2	Heating			Cooling		
Criterion	MSE_v	MSE_t	M_{ae}	MSE_v	MSE_t	M_{ae}
KTB	3.1580e-04	1.4739e-03	0.2047	2.0304e-04	6.2714e-04	0.0995
Ensemble (SA)	3.0433e-04	1.3555e-03	0.2057	1.9490e-04	6.1836e-04	0.0999
Ensemble optimized (ES)	3.0543e-04	1.3448e-03	0.2058	1.9314e-04	6.1934e-04	0.0999
NDEO	3.2595e-04	1.3475e-03	0.2068	2.0879e-04	7.0691e-04	0.1002

Model 3	Heating			Cooling		
Criterion	MSE_v	MSE_t	M_{ae}	MSE_v	MSE_t	M_{ae}
KTB	2.4512e-04	9.9152e-04	0.1886	2.9941e-04	6.9524e-04	0.2012
Ensemble (SA)	2.2064e-04	9.4997e-04	0.1890	2.6643e-04	6.5813e-04	0.2030
Ensemble optimized (ES)	2.1917e-04	9.3224e-04	0.1882	2.6617e-04	6.5446e-04	0.2028
NDEO	2.4512e-04	9.9152e-04	0.1886	2.9943e-04	6.9526e-04	0.2012

Model 4	Heating			Cooling		
Criterion	MSE_v	MSE_t	M_{ae}	MSE_v	MSE_t	M_{ae}
KTB	2.6030e-04	6.9525e-04	0.1313	1.9462e-04	5.4587e-04	0.0725
Ensemble (SA)	2.1240e-04	7.0022e-04	0.1339	1.9029e-04	5.1807e-04	0.0709
Ensemble optimized (ES)	2.1273e-04	6.9135e-04	0.1325	1.8929e-04	5.1173e-04	0.0703
NDEO	2.6030e-04	6.9525e-04	0.1313	2.0788e-04	5.4489e-04	0.0722

The ensemble approaches assessment, and generalization ability comparison with the KTB scheme are presented in Table (C.13) and (C.14), respectively.

The observations previously noted related to the uncorrupted data experience are now reinforced. The ES approach seems to be the only ensemble approach that returns a performance gain in situations with low ambiguity levels, as is the Model of models training with similar data like this one. Despite the ideal Model being the one where the designer can force a presence of high ambiguity levels in the ensemble, this is not always true. Designing situations arise where models cannot be uncorrelated, and the ES is proving itself to be a possible reasonable choice in such a situation.

TABLE C.14: Generalization error obtained with the ensemble approaches in comparison with the KTB model selection. SPSI, Sensor 3 (1.5 W/cm^2). Data used: uncorrupted.

Model 1 (2 Lags)	**Heating**	**Cooling**	**Average comparison**
Ensemble (SA)	14.43 %	3.85 %	9.14 %
Ensemble optimized (ES)	15.90 %	4.54 %	10.22 %
NDEO	17.31 %	-1.56 %	7.88 %

Model 2 (3 Lags)	**Heating**	**Cooling**	**Average comparison**
Ensemble (SA)	8.03 %	1.40 %	4.72 %
Ensemble optimized (ES)	8.76 %	1.24 %	5.00 %
NDEO	8.58 %	-12.72 %	-2.07 %

Model 3 (4 Lags)	**Heating**	**Cooling**	**Average comparison**
Ensemble (SA)	4.19 %	5.34 %	4.76 %
Ensemble optimized (ES)	5.98 %	5.87 %	5.92 %
NDEO	0.00 %	-0.00 %	-0.00 %

Model 4 (5 Lags)	**Heating**	**Cooling**	**Average comparison**
Ensemble (SA)	-0.72 %	5.09 %	2.19 %
Ensemble optimized (ES)	0.56 %	6.25 %	3.41 %
NDEO	0.00 %	0.18 %	0.09 %

Concerning the corrupted data experiment, Figure (C.11) illustrates the contamination process. The assessment of the four constructed networks is reflected in Table (C.15).

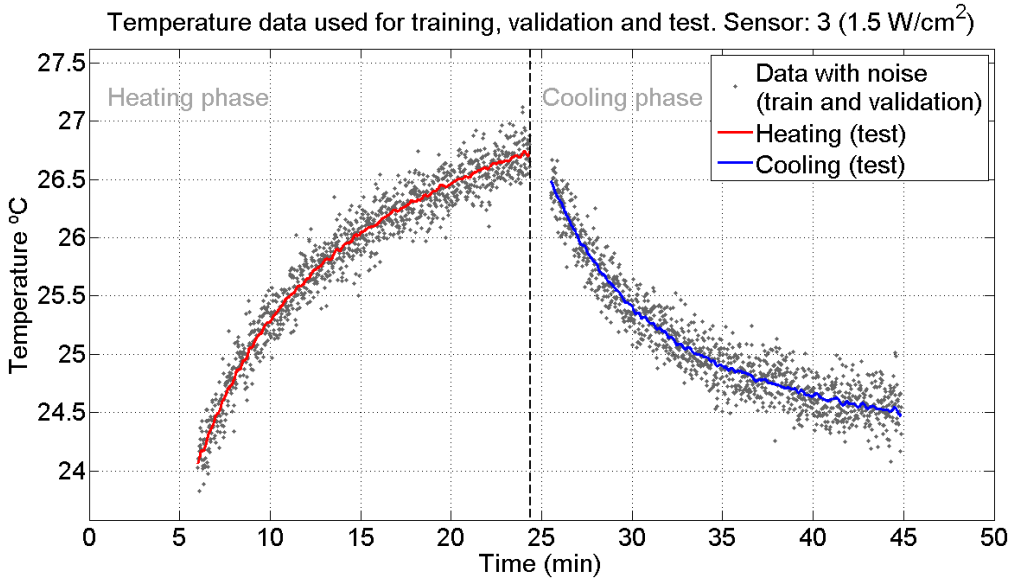

FIGURE C.11: Data after the addition of random Gaussian noise. The noisy data is used for training and validation. The Uncorrupted, noise free data is used to test the model. Collected from the *homogeneous phantom* experimental setup. TUS intensity: $1.8W/cm^2$. Sensor 1.

TABLE C.15: Performance descriptors obtained concerning models with different number of input lags. The models presented were selected using the KTB approach. SPSI, Sensor 3 (1.5 W/cm^2). Data used: corrupted.

Model 1(2 lags)		
Criterion	Heating	Cooling
BIC	-2637	-2788
MSE	3.8926e-02	3.9419e-02
$MSRE$	1.5084e-03	1.5679e-03
MSE_v	4.8054e-02	3.6015e-02
$MSRE_v$	1.8570e-03	1.4382e-03
MSE_t	1.6145e-03	1.7407e-03
$MSRE_t$	6.1732e-05	6.8998e-05
M_{ae}	0.1200	0.1734
LWN	8	7
SR	n	n

Model 2(3 lags)		
Criterion	Heating	Cooling
BIC	-2689	-2943
MSE	3.5113e-02	3.1417e-02
$MSRE$	1.3579e-03	1.2531e-03
MSE_v	3.2671e-02	3.6800e-02
$MSRE_v$	1.2645e-03	1.4740e-03
MSE_t	5.8995e-04	7.5874e-04
$MSRE_t$	2.2693e-05	3.0253e-05
M_{ae}	0.0751	0.0965
LWN	11	11
SR	n	n

Model 3(4 lags)		
Criterion	Heating	Cooling
BIC	-2781	-2953
MSE	2.9788e-02	2.8236e-02
$MSRE$	1.1551e-03	1.1278e-03
MSE_v	2.9115e-02	2.6645e-02
$MSRE_v$	1.1240e-03	1.0659e-03
MSE_t	3.0563e-04	4.1026e-04
$MSRE_t$	1.1784e-05	1.6447e-05
M_{ae}	0.0578	0.0559
LWN	15	17
SR	n	n

Model 4(5 lags)		
Criterion	Heating	Cooling
BIC	-2898	-3098
MSE	2.6102e-02	2.4820e-02
$MSRE$	1.0115e-03	9.8993e-04
MSE_v	2.7539e-02	3.3844e-02
$MSRE_v$	1.0635e-03	1.3559e-03
MSE_t	2.5606e-04	4.5118e-04
$MSRE_t$	9.9510e-06	1.8058e-05
M_{ae}	0.0580	0.0604
LWN	19	19
SR	n	n

The behaviour of model 2 was chose to be plotted through all the test set, Figure (C.12).

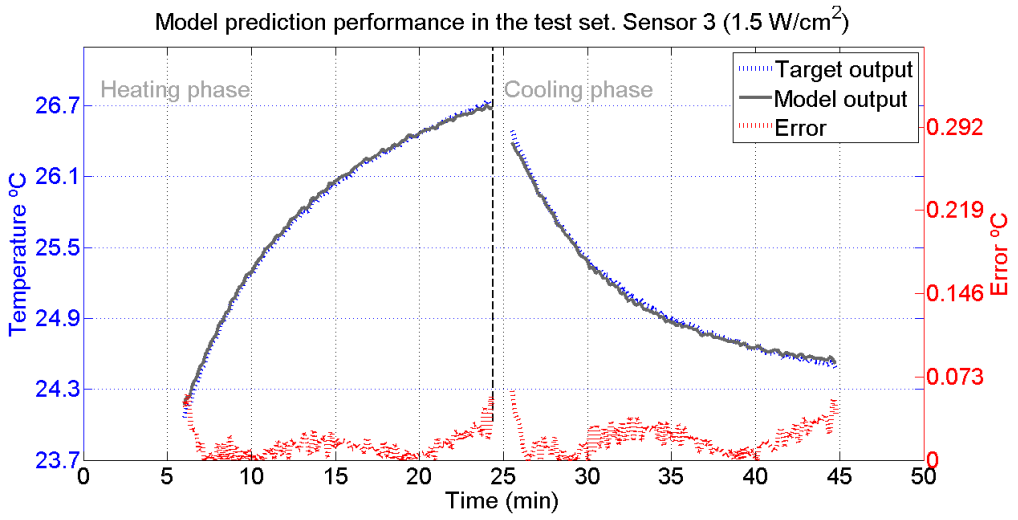

FIGURE C.12: Model's behaviour in the test set, selected using the KTB approach. The blue line represents the desired behaviour and the model's test output is given by the black line. The error line is red. SISP,*homogeneous phantom* experimental setup. TUS intensity: $1.5W/cm^2$. Sensor 3. Data used: corrupted.

Again the absolute errors were kept under a satisfactory threshold (0.07 ^oC). The ensemble results are shown in Table (C.16) and the comparison with the KTB scheme is highlighted in Table (C.17).

NDEO held the top performance place concerning generalization ability increase. However, the ES method also achieved very satisfactory and consistent results with ones previously obtained. By this time it is becoming obvious that the SA method is completely outperformed by both ES and NDEO methods. The lack of intelligence in the combination of the individual predictions gives rise to an unstable method that oscillates between high performance gains and drops.

TABLE C.16: Performance comparison between all methodologies employed. SPSI, Sensor 3 (1.5 W/cm^2). Data used: corrupted.

Model 1	Heating			Cooling		
Criterion	MSE_v	MSE_t	M_{ae}	MSE_v	MSE_t	M_{ae}
KTB	4.8054e-02	1.6145e-03	0.1200	3.6015e-02	1.7407e-03	0.1734
Ensemble (SA)	2.3705e-02	5.9379e-04	0.0825	2.2412e-02	7.5503e-04	0.1734
Ensemble optimized (ES)	2.3579e-02	2.0074e-04	0.0825	2.2509e-02	3.7683e-04	0.1734
NDEO	2.3596e-02	1.3958e-04	0.0563	2.2716e-02	1.2519e-04	0.1197

Model 2	Heating			Cooling		
Criterion	MSE_v	MSE_t	M_{ae}	MSE_v	MSE_t	M_{ae}
KTB	3.2671e-02	5.8995e-04	0.0751	3.6800e-02	7.5874e-04	0.0965
Ensemble (SA)	2.2022e-02	5.8814e-04	0.0772	2.7843e-02	7.2973e-04	0.1327
Ensemble optimized (ES)	2.2202e-02	1.9473e-04	0.0690	2.7278e-02	3.5120e-04	0.1327
NDEO	2.2401e-02	1.0990e-04	0.0212	2.6322e-02	2.3520e-04	0.0255

Model 3	Heating			Cooling		
Criterion	MSE_v	MSE_t	M_{ae}	MSE_v	MSE_t	M_{ae}
KTB	2.9115e-02	3.0563e-04	0.0578	2.6645e-02	4.1026e-04	0.0559
Ensemble (SA)	2.2418e-02	5.8435e-04	0.0772	2.1759e-02	7.1514e-04	0.0941
Ensemble optimized (ES)	2.1940e-02	1.9058e-04	0.0593	2.1509e-02	3.3629e-04	0.0941
NDEO	2.1866e-02	3.9712e-05	0.0176	2.1489e-02	7.6532e-05	0.0292

Model 4	Heating			Cooling		
Criterion	MSE_v	MSE_t	M_{ae}	MSE_v	MSE_t	M_{ae}
KTB	2.7539e-02	2.5606e-04	0.0580	3.3844e-02	4.5118e-04	0.0604
Ensemble (SA)	2.2948e-02	5.8167e-04	0.0772	2.7960e-02	7.0811e-04	0.0838
Ensemble optimized (ES)	2.2515e-02	1.8755e-04	0.0447	2.7559e-02	3.2893e-04	0.0446
NDEO	2.2964e-02	2.4210e-04	0.0399	2.7506e-02	2.7054e-04	0.0587

TABLE C.17: Generalization error obtained with the ensemble approaches in comparison with the KTB model selection. SPSI, Sensor 3 (1.5 W/cm^2). Data used: corrupted.

Model 1 (2 Lags)	**Heating**	**Cooling**	**Average comparison**
Ensemble (SA)	63.22 %	56.62 %	59.92 %
Ensemble optimized (ES)	87.57 %	78.35 %	82.96 %
NDEO	91.35 %	92.81 %	92.08 %

Model 2 (3 Lags)	**Heating**	**Cooling**	**Average comparison**
Ensemble (SA)	0.31 %	3.82 %	2.06 %
Ensemble optimized (ES)	66.99 %	53.71 %	60.35 %
NDEO	81.37 %	69.00 %	75.19 %

Model 3 (4 Lags)	**Heating**	**Cooling**	**Average comparison**
Ensemble (SA)	-91.19 %	-74.31 %	-82.75 %
Ensemble optimized (ES)	37.64 %	18.03 %	27.84 %
NDEO	87.01 %	81.35 %	84.18 %

Model 4 (5 Lags)	**Heating**	**Cooling**	**Average comparison**
Ensemble (SA)	-127.17 %	-56.95 %	-92.06 %
Ensemble optimized (ES)	26.75 %	27.10 %	26.93 %
NDEO	30.45 %	40.04 %	35.425 %

Model environment: TUS Intensity ($1.8W/cm^2$), Sensor (4)

Passing to the fourth sensor, a $1.8W/cm^2$ TUS beam intensity was considered. This consists on the fastest propagation experienced in this sensor. Starting with the uncorrupted data from Figure (C.13), the usual 70/20/10 data set division was employed, and four models constructed that gave rise to the performance figures present in Table (C.18).

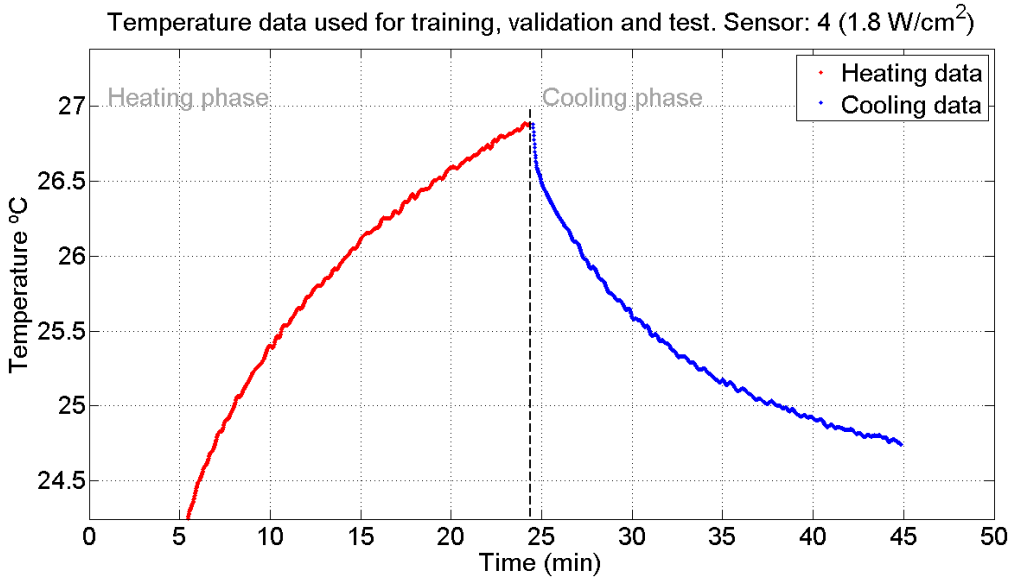

FIGURE C.13: Uncorrupted data set used for SPSI model training, validation and test. Collected from the *homogeneous phantom* experimental setup. TUS intensity: $1.8W/cm^2$. Sensor 4.

From the results shown it is obvious that the results are in compliance with the previous environments. The test error was sustained under reasonable results. The output curve illustrating the dynamic exhibit by the simplest model (2 input lags) is depicted in Figure (C.14). The error one again was kept under a strict low threshold ($0.19\ ^oC$).

TABLE C.18: Performance descriptors obtained concerning models with different number of input lags. The models presented were selected using the KTB approach. SPSI, Sensor 4 (1.8 W/cm^2). Data used: uncorrupted.

Model 1(2 lags)		
Criterion	Heating	Cooling
BIC	-8840	-10169
MSE	1.7185e-05	9.6939e-06
$MSRE$	6.7235e-07	3.7831e-07
MSE_v	3.5428e-04	2.2367e-04
$MSRE_v$	1.3991e-05	8.6654e-06
MSE_t	9.6507e-04	7.9555e-04
$MSRE_t$	3.7596e-05	3.1075e-05
M_{ae}	0.1299	0.1835
LWN	7	7
SR	n	n

Model 2(3 lags)		
Criterion	Heating	Cooling
BIC	-9317	-9806
MSE	1.3847e-05	8.0165e-06
$MSRE$	5.4468e-07	3.1057e-07
MSE_v	3.7732e-04	2.1936e-04
$MSRE_v$	1.4768e-05	8.5130e-06
MSE_t	8.3145e-04	3.8587e-04
$MSRE_t$	3.2449e-05	1.5243e-05
M_{ae}	0.1372	0.0963
LWN	7	11
SR	n	n

Model 3(4 lags)		
Criterion	Heating	Cooling
BIC	-9491	-9533
MSE	9.6204e-06	1.1763e-05
$MSRE$	3.7784e-07	4.5908e-07
MSE_v	3.8591e-04	1.5094e-04
$MSRE_v$	1.5142e-05	5.8703e-06
MSE_t	5.9893e-04	3.5579e-04
$MSRE_t$	2.3193e-05	1.3956e-05
M_{ae}	0.1359	0.0820
LWN	11	7
SR	n	n

Model 4(5 lags)		
Criterion	Heating	Cooling
BIC	-9680	-9724
MSE	8.5730e-06	9.7665e-06
$MSRE$	3.3542e-07	3.8153e-07
MSE_v	2.9307e-04	1.3316e-04
$MSRE_v$	1.1453e-05	5.2093e-06
MSE_t	6.4724e-04	4.1889e-04
$MSRE_t$	2.4965e-05	1.6414e-05
M_{ae}	0.1037	0.1623
LWN	11	7
SR	n	e

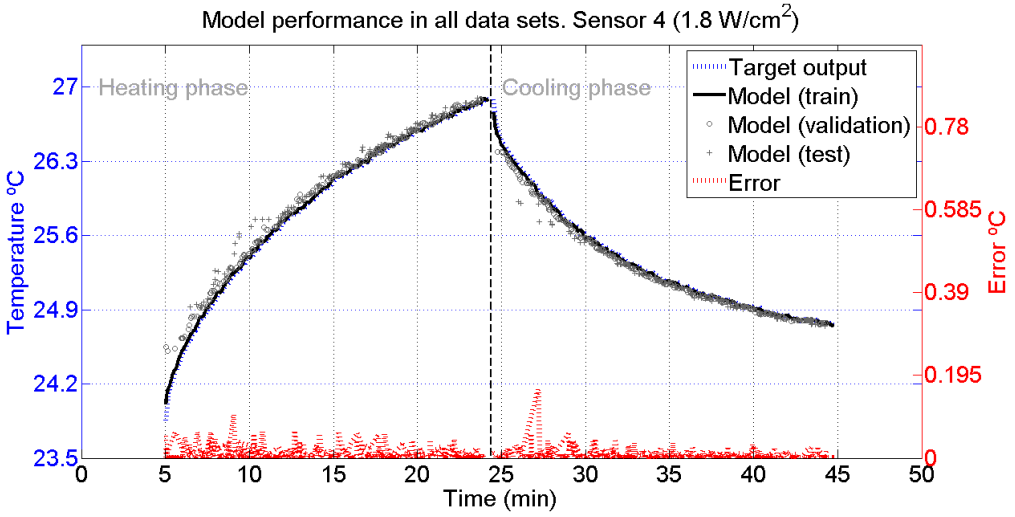

FIGURE C.14: Behaviour of the model through the whole data set, selected using the KTB approach. The blue line represents the desired behaviour and the model's training output is given by the black line. The error line is red, circle and cross markers represents the model's validation and test output respectively. SISP,*homogeneous phantom* experimental setup. TUS intensity: 1.8W/cm^2. Sensor 4. Data used: uncorrupted.

Ensemble approaches performed as shown in Table (C.19). The generalization ability of this ensembles methods is confronted with the test error calculated for KTB models in Table (C.19).

TABLE C.19: Performance comparison between all methodologies employed. SPSI, Sensor 4 (1.8 W/cm^2). Data used: uncorrupted.

Model 1	Heating			Cooling		
Criterion	MSE_v	MSE_t	M_{ae}	MSE_v	MSE_t	M_{ae}
KTB	3.5428e-04	9.6507e-04	0.1299	2.2367e-04	7.9555e-04	0.1835
Ensemble (SA)	3.3474e-04	8.6056e-04	0.1263	2.0814e-04	7.6898e-04	0.1830
Ensemble optimized (ES)	3.3513e-04	8.6610e-04	0.1263	2.0304e-04	7.7762e-04	0.1828
NDEO	3.6319e-04	8.5947e-04	0.1261	2.1875e-04	8.6067e-04	0.1827

Model 2	Heating			Cooling		
Criterion	MSE_v	MSE_t	M_{ae}	MSE_v	MSE_t	M_{ae}
KTB	3.7732e-04	8.3145e-04	0.1372	2.1936e-04	3.8587e-04	0.0963
Ensemble (SA)	3.6729e-04	6.8989e-04	0.1366	1.9731e-04	3.3727e-04	0.1167
Ensemble optimized (ES)	3.6659e-04	7.0011e-04	0.1366	1.9977e-04	3.3853e-04	0.1167
NDEO	5.1389e-04	6.5189e-04	0.1604	2.1898e-04	3.8426e-04	0.0967

Model 3	Heating			Cooling		
Criterion	MSE_v	MSE_t	M_{ae}	MSE_v	MSE_t	M_{ae}
KTB	3.8591e-04	5.9893e-04	0.1359	1.5094e-04	3.5579e-04	0.0820
Ensemble (SA)	3.6284e-04	6.3373e-04	0.1359	1.4329e-04	3.4095e-04	0.0835
Ensemble optimized (ES)	3.6050e-04	6.4558e-04	0.1359	1.4064e-04	3.3823e-04	0.0846
NDEO	3.8590e-04	5.9892e-04	0.1359	1.5317e-04	3.7540e-04	0.0875

Model 4	Heating			Cooling		
Criterion	MSE_v	MSE_t	M_{ae}	MSE_v	MSE_t	M_{ae}
KTB	2.9307e-04	6.4724e-04	0.1037	1.3316e-04	4.1889e-04	0.1623
Ensemble (SA)	2.7141e-04	7.0879e-04	0.1082	1.2650e-04	4.1121e-04	0.1621
Ensemble optimized (ES)	2.7158e-04	7.2467e-04	0.1087	1.2407e-04	4.1667e-04	0.1617
NDEO	2.9307e-04	6.4724e-04	0.1037	1.3574e-04	4.7759e-04	0.1608

Focusing in the fourth Model (Model 4), the results show that no method was capable of returning substantial performance gains. The generalization ability of the ensemble was always worst than the individual test set performance of model 4.

Figure (C.15) exhibits the Gaussian contamination process of the original data. This corrupted was split following a 70/30 ratio into training/validation data sets. The Uncorrupted data was used to assess the model generalization ability.

TABLE C.20: Generalization error obtained with the ensemble approaches in comparison with the KTB model selection. SPSI, Sensor 4 (1.8 W/cm^2). Data used: uncorrupted.

Model 1 (2 Lags)	**Heating**	**Cooling**	**Average comparison**
Ensemble (SA)	10.83 %	3.34 %	7.08 %
Ensemble optimized (ES)	10.26 %	2.25 %	6.25 %
NDEO	10.94 %	-8.19 %	1.38 %

Model 2 (3 Lags)	**Heating**	**Cooling**	**Average comparison**
Ensemble (SA)	17.03 %	12.59 %	14.81 %
Ensemble optimized (ES)	15.80 %	12.27 %	14.03 %
NDEO	21.60 %	0.42 %	11.01 %

Model 3 (4 Lags)	**Heating**	**Cooling**	**Average comparison**
Ensemble (SA)	-5.81 %	4.17 %	-0.82 %
Ensemble optimized (ES)	-7.79 %	4.93 %	-1.43 %
NDEO	0.00 %	-5.51 %	-2.76 %

Model 4 (5 Lags)	**Heating**	**Cooling**	**Average comparison**
Ensemble (SA)	-9.51 %	1.83 %	-3.84 %
Ensemble optimized (ES)	-11.96 %	0.53 %	-5.72 %
NDEO	0.00 %	-14.01 %	-7.01 %

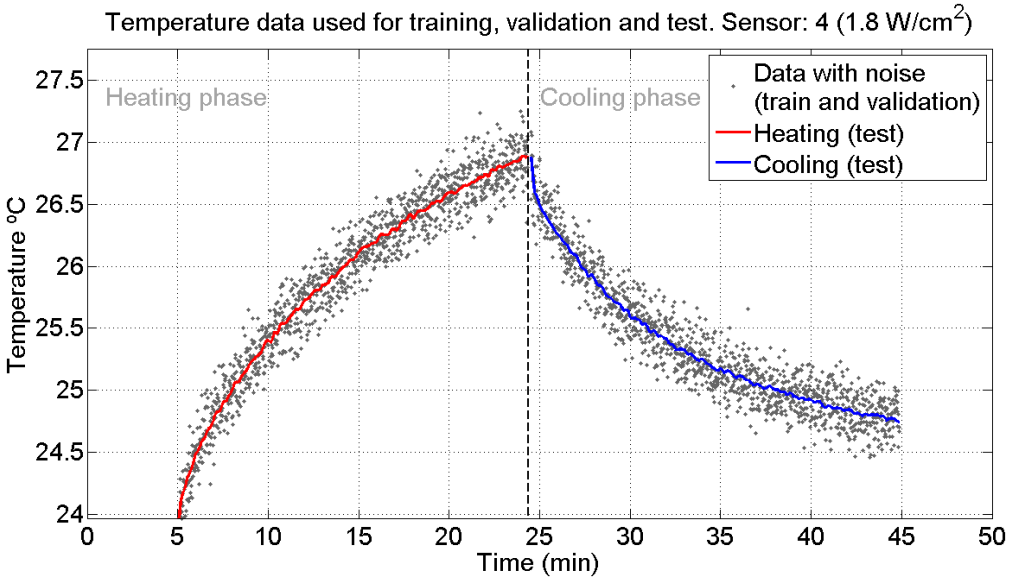

FIGURE C.15: Data after the addition of random Gaussian noise. The noisy data is used for training and validation. The Uncorrupted, noise free data is used to test the model. Collected from the *homogeneous phantom* experimental setup. TUS intensity: $1.8W/cm^2$. Sensor 4.

As before, four models were built. Their performance criteria are calculated in Table (C.21).

TABLE C.21: Performance descriptors obtained concerning models with different number of input lags. The models presented were selected using the KTB approach. SPSI, Sensor 4 (1.8 W/cm^2). Data used: corrupted.

Model 1(2 lags)		
Criterion	Heating	Cooling
BIC	-2722	-2933
MSE	4.1759e-02	4.0239e-02
$MSRE$	1.6158e-03	1.5854e-03
MSE_v	4.6206e-02	3.1819e-02
$MSRE_v$	1.7932e-03	1.2578e-03
MSE_t	1.7511e-03	1.7319e-03
$MSRE_t$	6.7297e-05	6.8271e-05
M_{ae}	0.1394	0.2170
LWN	8	8
SR	n	n

Model 2(3 lags)		
Criterion	Heating	Cooling
BIC	-2855	-3071
MSE	3.4336e-02	3.4695e-02
$MSRE$	1.3264e-03	1.3692e-03
MSE_v	3.2198e-02	2.7293e-02
$MSRE_v$	1.2429e-03	1.0749e-03
MSE_t	5.6628e-04	7.4621e-04
$MSRE_t$	2.1708e-05	2.9507e-05
M_{ae}	0.0897	0.1015
LWN	10	11
SR	n	n

Model 3(4 lags)		
Criterion	Heating	Cooling
BIC	-3133	-3172
MSE	2.6828e-02	2.6636e-02
$MSRE$	1.0357e-03	1.0529e-03
MSE_v	3.4918e-02	2.4605e-02
$MSRE_v$	1.3458e-03	9.7299e-04
MSE_t	2.5131e-04	3.7529e-04
$MSRE_t$	9.6362e-06	1.4830e-05
M_{ae}	0.0450	0.0800
LWN	16	16
SR	n	n

Model 4(5 lags)		
Criterion	Heating	Cooling
BIC	-2941	-3159
MSE	3.0080e-02	2.8557e-02
$MSRE$	1.1624e-03	1.1261e-03
MSE_v	2.7302e-02	2.6530e-02
$MSRE_v$	1.0530e-03	1.0476e-03
MSE_t	2.7102e-04	4.8167e-04
$MSRE_t$	1.0530e-05	1.9008e-05
M_{ae}	0.0485	0.1112
LWN	20	19
SR	n	n

The output curve of model 4 is confronted with the desired dynamic in the test set (complete original data set) in Figure (C.16). Once again, the error curve revealed a small magnitude through all the test. Ensemble methods performance is shown in Table (C.22) and the usual comparison with the KTB models is done in Table (C.23).

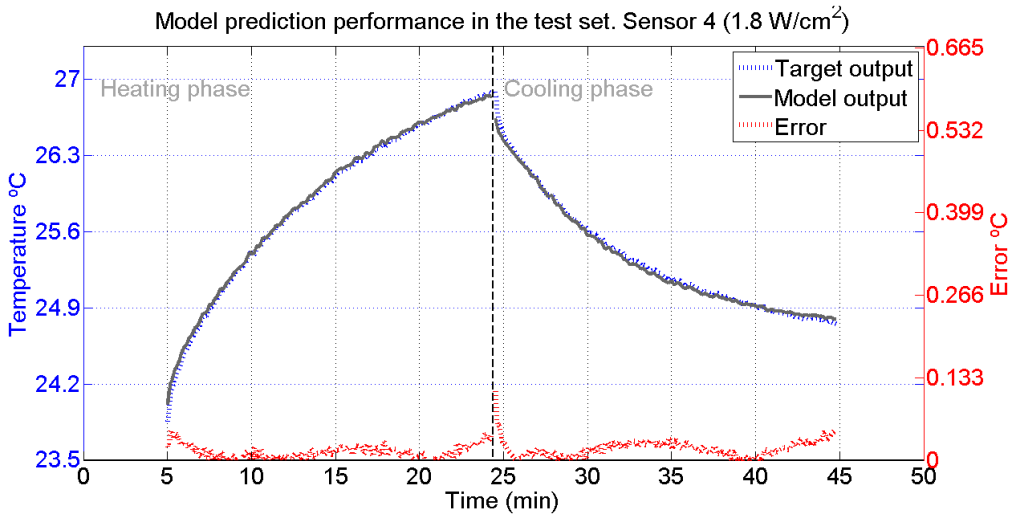

FIGURE C.16: Model's behaviour in the test set, selected using the KTB approach. The blue line represents the desired behaviour and the model's test output is given by the black line. The error line is red. SISP,*homogeneous phantom* experimental setup. TUS intensity: $1.8 W/cm^2$. Sensor 4. Data used: corrupted.

Both ES and NDEO generalization performance gains are considerable, however more accentuated in the later approach, whereas the simplest method (SA) still couldn't break from the oscillatory road it has been walking. We suspect this is due to the fact that the method does not possesses resources to differentiate the models. However it can be used as an assessment tool of ambiguity levels that are present in the ensemble.

TABLE C.22: Performance comparison between all methodologies employed. SPSI, Sensor 4 (1.8 W/cm^2). Data used: corrupted.

Model 1	Heating			Cooling		
Criterion	MSE_v	MSE_t	M_{ae}	MSE_v	MSE_t	M_{ae}
KTB	4.6206e-02	1.7511e-03	0.1394	3.1819e-02	1.7319e-03	0.2170
Ensemble (SA)	2.3016e-02	5.8406e-04	0.1394	1.8935e-02	7.8281e-04	0.2170
Ensemble optimized (ES)	2.2775e-02	2.3583e-04	0.1394	1.8983e-02	2.1496e-04	0.2170
NDEO	2.3142e-02	1.3573e-04	0.0884	1.9402e-02	9.1294e-05	0.1503

Model 2	Heating			Cooling		
Criterion	MSE_v	MSE_t	M_{ae}	MSE_v	MSE_t	M_{ae}
KTB	3.2198e-02	5.6628e-04	0.0897	2.7293e-02	7.4621e-04	0.1015
Ensemble (SA)	2.2720e-02	5.6778e-04	0.1112	2.0135e-02	7.4481e-04	0.1508
Ensemble optimized (ES)	2.2862e-02	2.1925e-04	0.1112	1.9858e-02	1.7649e-04	0.1508
NDEO	2.2623e-02	3.3615e-05	0.0231	2.0038e-02	1.3917e-04	0.0242

Model 3	Heating			Cooling		
Criterion	MSE_v	MSE_t	M_{ae}	MSE_v	MSE_t	M_{ae}
KTB	3.4918e-02	2.5131e-04	0.0450	2.4605e-02	3.7529e-04	0.0800
Ensemble (SA)	2.5410e-02	5.5758e-04	0.0700	2.0148e-02	7.2674e-04	0.1184
Ensemble optimized (ES)	2.5342e-02	2.0874e-04	0.0700	1.9944e-02	1.5796e-04	0.1184
NDEO	2.5602e-02	1.5233e-04	0.0956	1.9870e-02	4.2096e-05	0.0194

Model 4	Heating			Cooling		
Criterion	MSE_v	MSE_t	M_{ae}	MSE_v	MSE_t	M_{ae}
KTB	2.7302e-02	2.7102e-04	0.0485	2.6530e-02	4.8167e-04	0.1112
Ensemble (SA)	2.2721e-02	5.5382e-04	0.0665	2.2223e-02	7.1581e-04	0.1080
Ensemble optimized (ES)	2.2349e-02	2.0469e-04	0.0398	2.1771e-02	1.4656e-04	0.0367
NDEO	2.2054e-02	3.6352e-05	0.0189	2.1681e-02	5.7144e-05	0.0526

TABLE C.23: Generalization error obtained with the ensemble approaches in comparison with the KTB model selection. SPSI, Sensor 4 (1.8 W/cm^2). Data used: corrupted.

Model 1 (2 Lags)	**Heating**	**Cooling**	**Average comparison**
Ensemble (SA)	66.65 %	54.80 %	60.72 %
Ensemble optimized (ES)	86.53 %	87.59 %	87.06 %
NDEO	92.25 %	94.73 %	93.49 %

Model 2 (3 Lags)	**Heating**	**Cooling**	**Average comparison**
Ensemble (SA)	-0.26 %	0.19 %	-0.04 %
Ensemble optimized (ES)	61.28 %	76.35 %	68.82 %
NDEO	94.06 %	81.35 %	87.71 %

Model 3 (4 Lags)	**Heating**	**Cooling**	**Average comparison**
Ensemble (SA)	-121.87 %	-93.65 %	-107.76 %
Ensemble optimized (ES)	16.94 %	57.91 %	37.42 %
NDEO	39.38 %	88.78 %	64.08 %

Model 4 (5 Lags)	**Heating**	**Cooling**	**Average comparison**
Ensemble (SA)	-104.35 %	-48.61 %	-76.48 %
Ensemble optimized (ES)	24.47 %	69.57 %	47.02 %
NDEO	86.59 %	88.14 %	87.36 %

Model environment: TUS Intensity ($1.0W/cm^2$), Sensor (5)

The last SPSI environment is now considered, consisting of the data collected by the furthest sensor (5), relatively to the TUS device, while applying a TUS beam intensity of $1.0W/cm^2$. Analogously to the previous environments, the uncorrupted data is first considered, depicted in Figure (C.17). The fifth sensor was placed orthogonally at $20mm$ from the TUS beam central line, Figure (3.4) of Chapter(3), thus the heating experienced in this region was small, hence the temperature propagation dynamics are slow and smooth. Consequently the models are expected to follow the desired behaviour without a vast effort.

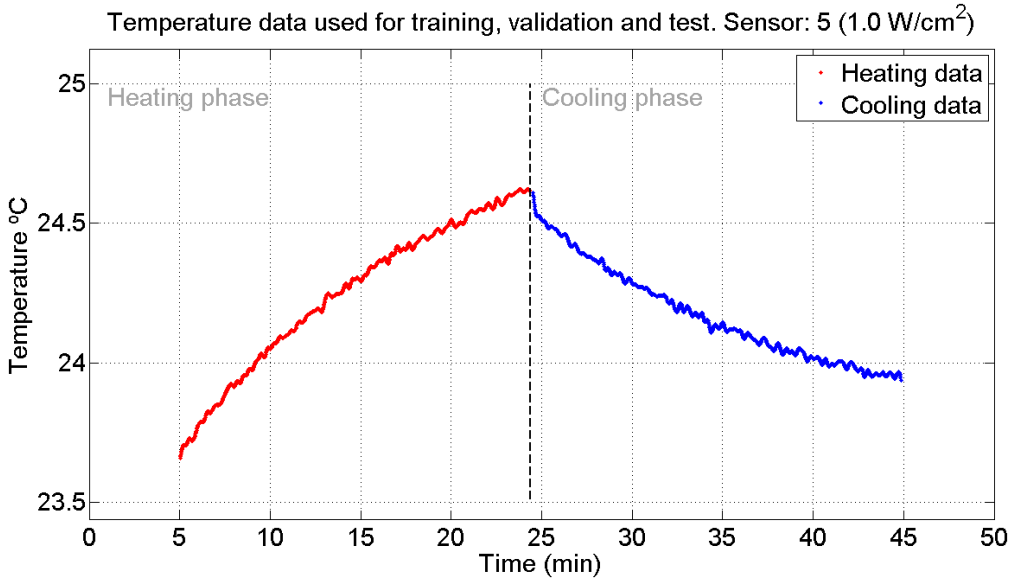

FIGURE C.17: Uncorrupted data set used for SPSI model training, validation and test. Collected from the *homogeneous phantom* experimental setup. TUS intensity: $1.0W/cm^2$. Sensor 5.

Four distinct models were constructed, exposed Table (C.24).

In Figure (C.18) the third model (Model 3) output curve is confronted with the desired one. We can observe two almost perfectly matched curves. It is by this time evident that the most defiant challenges are concentrated in the environments exhibiting fast and abrupt temperature evolutions, i.e. higher intensities and most importantly, spatial locations more close to the TUS device face.

The ensemble approaches performance indicators are shown in Table (C.25) and the generalization ability is compared with KTB models in Table (C.26).

TABLE C.24: Performance descriptors obtained concerning models with different number of input lags. The models presented were selected using the KTB approach. SPSI, Sensor 5 (1.0 W/cm^2). Data used: uncorrupted.

Model 1(2 lags)		
Criterion	Heating	Cooling
BIC	-9853	-9872
MSE	6.2375e-06	8.9043e-06
$MSRE$	2.5726e-07	3.6862e-07
MSE_v	9.5965e-05	8.5749e-05
$MSRE_v$	3.9692e-06	3.5519e-06
MSE_t	2.3481e-04	1.3382e-04
$MSRE_t$	9.7033e-06	5.5407e-06
M_{ae}	0.0795	0.0418
LWN	7	7
SR	n	n

Model 2(3 lags)		
Criterion	Heating	Cooling
BIC	-10817	-10362
MSE	2.7668e-06	7.5919e-06
$MSRE$	1.1411e-07	3.1433e-07
MSE_v	9.3611e-05	8.0271e-05
$MSRE_v$	3.8579e-06	3.3278e-06
MSE_t	2.6098e-04	1.8483e-04
$MSRE_t$	1.0734e-05	7.6457e-06
M_{ae}	0.0476	0.0432
LWN	11	7
SR	n	n

Model 3(4 lags)		
Criterion	Heating	Cooling
BIC	-10735	-10405
MSE	2.6602e-06	4.9640e-06
$MSRE$	1.0981e-07	2.0561e-07
MSE_v	7.9328e-05	8.3608e-05
$MSRE_v$	3.2562e-06	3.4647e-06
MSE_t	3.2457e-04	2.7561e-04
$MSRE_t$	1.3377e-05	1.1435e-05
M_{ae}	0.0618	0.0504
LWN	11	11
SR	n	n

Model 4(5 lags)		
Criterion	Heating	Cooling
BIC	-10245	-9927
MSE	5.2799e-06	8.3487e-06
$MSRE$	2.1774e-07	3.4576e-07
MSE_v	7.9055e-05	6.0168e-05
$MSRE_v$	3.2623e-06	2.4919e-06
MSE_t	2.0606e-04	1.3722e-04
$MSRE_t$	8.4744e-06	5.6819e-06
M_{ae}	0.0627	0.0537
LWN	7	7
SR	n	e

The KTB comparison revealed highly unstable results. Definitely data random division and different number of input lags used to train the networks are not sufficient measures to achieve the ambiguity levels required to explore the ensemble mechanism potentials. In a general way the three methods exhibit a fairly low performance.

Figure (C.19) depicts the corrupted using to construct the models for the second experiment. Their performance figures can be assessed in Table (C.27). The usual 70/30 division scheme separates the training set from the validation set, and testing is done over the whole original data set.

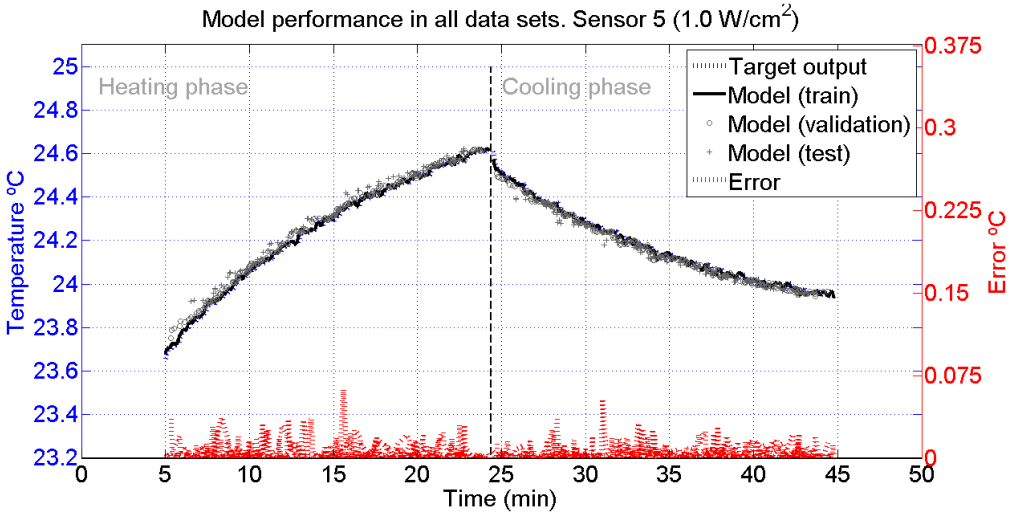

FIGURE C.18: Behaviour of the model through the whole data set, selected using the KTB approach. The blue line represents the desired behaviour and the model's training output is given by the black line. The error line is red, circle and cross markers represents the model's validation and test output respectively. SISP,*homogeneous phantom* experimental setup. TUS intensity: $1.0W/cm^2$. Sensor 5. Data used: uncorrupted.

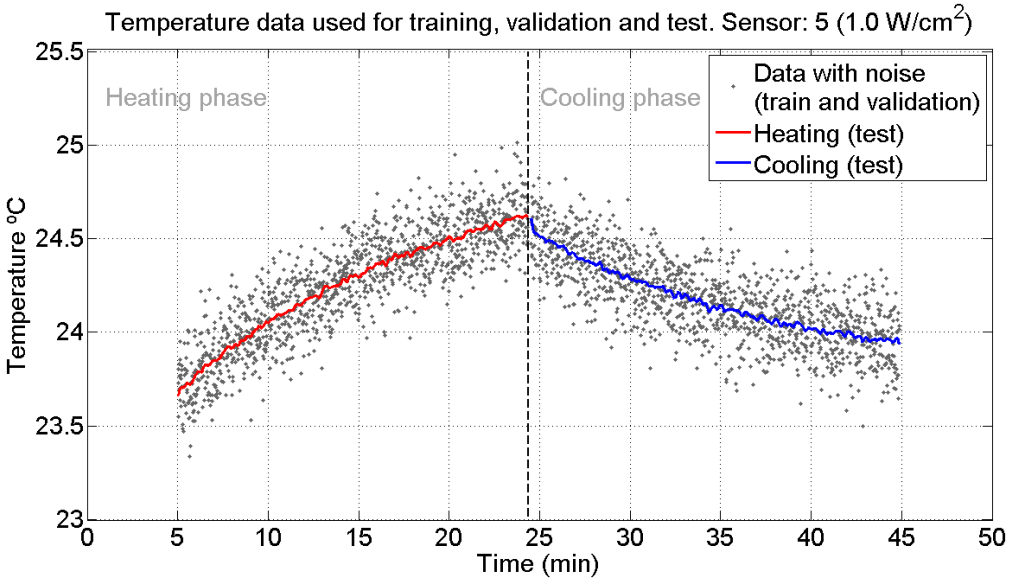

FIGURE C.19: Data after the addition of random Gaussian noise. The noisy data is used for training and validation. The Uncorrupted, noise free data is used to test the model. Collected from the *homogeneous phantom* experimental setup. TUS intensity: $1.0W/cm^2$. Sensor 5.

The output of model 3 over the test is shown in Figure (C.20).

TABLE C.25: Performance comparison between all methodologies employed. SPSI, Sensor 5 (1.0 W/cm^2). Data used: uncorrupted.

Model 1	Heating			Cooling		
Criterion	MSE_v	MSE_t	M_{ae}	MSE_v	MSE_t	M_{ae}
KTB	9.5965e-05	2.3481e-04	0.0795	8.5749e-05	1.3382e-04	0.0418
Ensemble (SA)	9.2791e-05	2.5343e-04	0.0793	8.6340e-05	1.3884e-04	0.0424
Ensemble optimized (ES)	9.1119e-05	2.3534e-04	0.0794	8.5387e-05	1.3441e-04	0.0420
NDEO	1.1362e-04	3.2665e-04	0.0792	1.1591e-04	1.9896e-04	0.0445

Model 2	Heating			Cooling		
Criterion	MSE_v	MSE_t	M_{ae}	MSE_v	MSE_t	M_{ae}
KTB	9.3611e-05	2.6098e-04	0.0476	8.0271e-05	1.8483e-04	0.0432
Ensemble (SA)	7.3989e-05	2.2148e-04	0.0428	8.3155e-05	1.9877e-04	0.0449
Ensemble optimized (ES)	7.0526e-05	2.2464e-04	0.0416	8.0441e-05	1.8727e-04	0.0432
NDEO	9.3589e-05	2.6087e-04	0.0476	1.1757e-04	3.0325e-04	0.0524

Model 3	Heating			Cooling		
Criterion	MSE_v	MSE_t	M_{ae}	MSE_v	MSE_t	M_{ae}
KTB	7.9328e-05	3.2457e-04	0.0618	8.3608e-05	2.7561e-04	0.0504
Ensemble (SA)	6.2215e-05	2.5668e-04	0.0605	6.3104e-05	1.9231e-04	0.0413
Ensemble optimized (ES)	6.0037e-05	2.4690e-04	0.0598	6.2946e-05	1.8573e-04	0.0398
NDEO	7.9303e-05	3.2452e-04	0.0618	8.3609e-05	2.7561e-04	0.0504

Model 4	Heating			Cooling		
Criterion	MSE_v	MSE_t	M_{ae}	MSE_v	MSE_t	M_{ae}
KTB	7.9055e-05	2.0606e-04	0.0627	6.0168e-05	1.3722e-04	0.0537
Ensemble (SA)	8.1700e-05	2.1279e-04	0.0640	5.9120e-05	1.4648e-04	0.0535
Ensemble optimized (ES)	7.7687e-05	2.0258e-04	0.0632	5.9496e-05	1.3916e-04	0.0538
NDEO	1.0499e-04	2.7270e-04	0.0654	7.5935e-05	2.1969e-04	0.0570

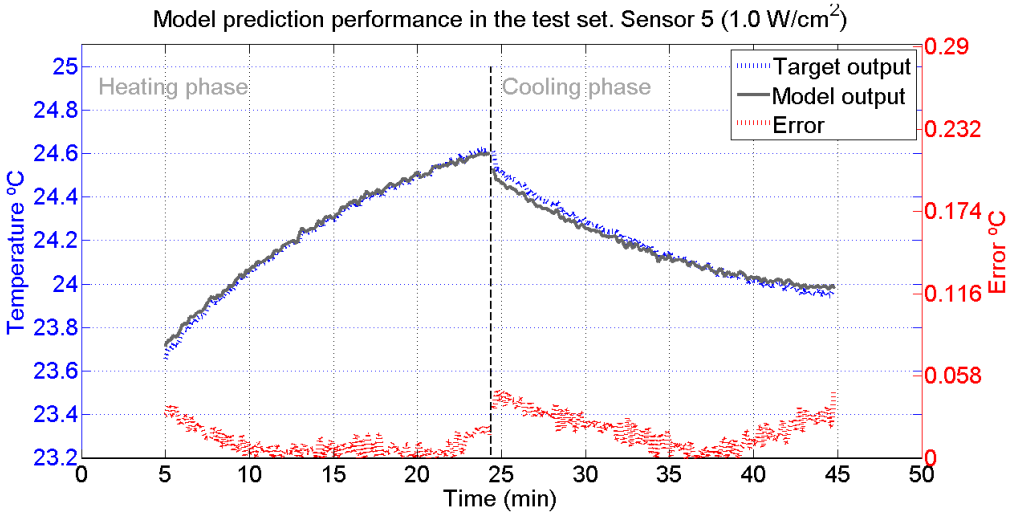

FIGURE C.20: Behaviour of model 3 in the test set, selected using the KTB approach. The blue line represents the desired behaviour and the model's test output is given by the black line. The error line is red. SISP,*homogeneous phantom* experimental setup. TUS intensity: $1.0W/cm^2$. Sensor 5. Data used: corrupted.

TABLE C.26: Generalization error obtained with the ensemble approaches in comparison with the KTB model selection. SPSI, Sensor 5 (1.0 W/cm^2). Data used: uncorrupted.

Model 1 (2 Lags)	**Heating**	**Cooling**	**Average comparison**
Ensemble (SA)	-7.93 %	-3.75 %	-5.84 %
Ensemble optimized (ES)	-0.22 %	-0.44 %	-0.33 %
NDEO	-39.11 %	-48.69 %	-43.90 %

Model 2 (3 Lags)	**Heating**	**Cooling**	**Average comparison**
Ensemble (SA)	15.13 %	-7.54 %	3.80 %
Ensemble optimized (ES)	13.93 %	-1.32 %	6.30 %
NDEO	0.04 %	-64.07 %	-32.01 %

Model 3 (4 Lags)	**Heating**	**Cooling**	**Average comparison**
Ensemble (SA)	20.92 %	30.22 %	25.57 %
Ensemble optimized (ES)	23.93 %	32.61 %	28.27 %
NDEO	0.01 %	0.00 %	0.01 %

Model 4 (5 Lags)	**Heating**	**Cooling**	**Average comparison**
Ensemble (SA)	-3.27 %	-6.75 %	-5.01 %
Ensemble optimized (ES)	1.69 %	-1.41 %	0.14 %
NDEO	-32.34 %	-60.10 %	-46.22 %

Observing the results, and confronting them with Figure (C.18), one can note that the maximum absolute error was reduced, even though the models were trained with highly noisy data, which proves the robustness of the modelling approaches taken. We classify the data as *highly noisy data* due to the fact that temperature range experienced in this sensor was narrow, $23.7 - 24.7\ ^{o}C$, and the Gaussian distribution from where the noise was taken from remained the same, which accentuates the power of the noise in the data, Figure (C.19).

Concerning the ensemble approaches, Table (C.28) exposed the calculated performance criteria values and Table (C.29) compares their generalization ability when compared with the KTB correspondent models.

When the modelling task is alleviated, i.e. smooth and slow dynamics, the ensembles methods returned high gains in the generalization ability, when compared with the KTB approaches. Assuredly the complexity of the forecasting task has impact in the enhancements gained when using this ensembles methods. This conclusion can be justified if we assume that hardening the dynamics involved in the modelling process leads to a smaller success rate of building good models. Therefore, in highly complex environments it is not trivial to consistently build good models to compose the ensemble. In contrast, if the

TABLE C.27: Performance descriptors obtained concerning models with different number of input lags. The models presented were selected using the KTB approach. SPSI, Sensor 5 (1.0 W/cm^2). Data used: corrupted.

Model 1(2 lags)		
Criterion	Heating	Cooling
BIC	-2752	-3062
MSE	3.7713e-02	3.6548e-02
$MSRE$	1.5551e-03	1.5105e-03
MSE_v	3.8728e-02	3.4810e-02
$MSRE_v$	1.5987e-03	1.4371e-03
MSE_t	1.9058e-03	4.9649e-03
$MSRE_t$	7.8911e-05	2.0472e-04
M_{ae}	0.1256	0.1670
LWN	7	8
SR	n	n

Model 2(3 lags)		
Criterion	Heating	Cooling
BIC	-3056	-3118
MSE	3.1093e-02	3.2258e-02
$MSRE$	1.2842e-03	1.3338e-03
MSE_v	3.4602e-02	3.1570e-02
$MSRE_v$	1.4272e-03	1.3091e-03
MSE_t	8.6358e-04	1.8765e-03
$MSRE_t$	3.5475e-05	7.7434e-05
M_{ae}	0.0738	0.0886
LWN	10	10
SR	n	n

Model 3(4 lags)		
Criterion	Heating	Cooling
BIC	-2982	-3223
MSE	2.8873e-02	2.7785e-02
$MSRE$	1.1926e-03	1.1484e-03
MSE_v	2.8571e-02	2.9675e-02
$MSRE_v$	1.1782e-03	1.2292e-03
MSE_t	3.6052e-04	3.4729e-04
$MSRE_t$	1.4825e-05	1.4315e-05
M_{ae}	0.0470	0.0510
LWN	15	15
SR	n	n

Model 4(5 lags)		
Criterion	Heating	Cooling
BIC	-3112	-3134
MSE	2.4869e-02	2.7266e-02
$MSRE$	1.0259e-03	1.1287e-03
MSE_v	2.3549e-02	2.6009e-02
$MSRE_v$	9.7095e-04	1.0764e-03
MSE_t	1.7014e-04	5.5672e-04
$MSRE_t$	7.0813e-06	2.2941e-05
M_{ae}	0.0374	0.0486
LWN	20	19
SR	n	n

dynamics are more easy to model, one can consistently build reliable models that enter the ensemble and, if the ambiguity levels are right, boost the performance enhancements one expects when using such techniques.

TABLE C.28: Performance comparison between all methodologies employed. SPSI, Sensor 5 (1.0 W/cm^2). Data used: corrupted.

Model 1	Heating			Cooling		
Criterion	MSE_v	MSE_t	M_{ae}	MSE_v	MSE_t	M_{ae}
KTB	3.8728e-02	1.9058e-03	0.1256	3.4810e-02	4.9649e-03	0.1670
Ensemble (SA)	2.3659e-02	6.5248e-04	0.1256	2.4576e-02	1.5052e-03	0.1670
Ensemble optimized (ES)	2.2847e-02	8.9711e-05	0.1256	2.3582e-02	1.6850e-04	0.1670
NDEO	2.2801e-02	1.2814e-04	0.1039	2.2938e-02	4.7537e-05	0.0921

Model 2	Heating			Cooling		
Criterion	MSE_v	MSE_t	M_{ae}	MSE_v	MSE_t	M_{ae}
KTB	3.4602e-02	8.6358e-04	0.0738	3.1570e-02	1.8765e-03	0.0886
Ensemble (SA)	2.4371e-02	6.3943e-04	0.0939	2.4556e-02	1.4835e-03	0.1257
Ensemble optimized (ES)	2.4012e-02	7.6183e-05	0.0939	2.2989e-02	1.4575e-04	0.1257
NDEO	2.4114e-02	1.8224e-04	0.0350	2.3562e-02	1.8295e-04	0.1012

Model 3	Heating			Cooling		
Criterion	MSE_v	MSE_t	M_{ae}	MSE_v	MSE_t	M_{ae}
KTB	2.8571e-02	3.6052e-04	0.0470	2.9675e-02	3.4729e-04	0.0510
Ensemble (SA)	2.4246e-02	6.3237e-04	0.0750	2.4305e-02	1.4718e-03	0.0986
Ensemble optimized (ES)	2.3217e-02	6.8635e-05	0.0750	2.3208e-02	1.3290e-04	0.0986
NDEO	2.3228e-02	3.0468e-05	0.0192	2.3388e-02	3.3558e-05	0.0178

Model 4	Heating			Cooling		
Criterion	MSE_v	MSE_t	M_{ae}	MSE_v	MSE_t	M_{ae}
KTB	2.3549e-02	1.7014e-04	0.0374	2.6009e-02	5.5672e-04	0.0486
Ensemble (SA)	2.0603e-02	6.2806e-04	0.0620	2.1851e-02	1.4650e-03	0.0805
Ensemble optimized (ES)	2.0067e-02	6.3838e-05	0.0230	2.1001e-02	1.2503e-04	0.0265
NDEO	1.9936e-02	4.5339e-05	0.0168	2.0933e-02	4.7459e-05	0.0208

TABLE C.29: Generalization error obtained with the ensemble approaches in comparison with the KTB model selection. SPSI, Sensor 5 (1.0 W/cm^2). Data used: corrupted.

Model 1 (2 Lags)	**Heating**	**Cooling**	**Average comparison**
Ensemble (SA)	65.76 %	69.68 %	67.72 %
Ensemble optimized (ES)	95.29 %	96.61 %	95.95 %
NDEO	93.28 %	99.04 %	96.16 %

Model 2 (3 Lags)	**Heating**	**Cooling**	**Average comparison**
Ensemble (SA)	25.96 %	20.94 %	23.45 %
Ensemble optimized (ES)	91.18 %	92.23 %	91.71 %
NDEO	98.90 %	90.25 %	94.58 %

Model 3 (4 Lags)	**Heating**	**Cooling**	**Average comparison**
Ensemble (SA)	-75.40 %	-323.80 %	-199.60 %
Ensemble optimized (ES)	80.96 %	61.73 %	71.35 %
NDEO	91.55 %	90.34 %	90.94 %

Model 4 (5 Lags)	**Heating**	**Cooling**	**Average comparison**
Ensemble (SA)	-269.15 %	-163.15 %	-216.15 %
Ensemble optimized (ES)	62.48 %	77.54 %	70.01 %
NDEO	73.35 %	91.48 %	82.41 %

D

Extended results obtained with respect to SPMI typology models.

This appendix covers the remaining modeling environments considered when modelling SPMI typology models, extending the results exposed in Section(5.3). The following operating points are covered here:

- Sensor 2, all TUS beam intensities ($0.5W/cm^2$, $1.0W/cm^2$, $1.5W/cm^2$, $1.8W/cm^2$)
- Sensor 5, all TUS beam intensities ($0.5W/cm^2$, $1.0W/cm^2$, $1.5W/cm^2$, $1.8W/cm^2$)

Model environment: Sensor (2), all TUS intensities

Moving along the array of sensors, we consider here the data collected by sensor 2, considering all intensities. This data is illustrated in Figure (D.1), after the completion of the Gaussian contamination process (corrupted data). We can discarded the first experiment (using the uncorrupted data) if we assume the second (corrupted) experiment

constitutes a more challenging task. Thus is reasonable to assume the results obtained with a corrupted data set to be at the very least comparable with the results one would obtain considering uncorrupted data.

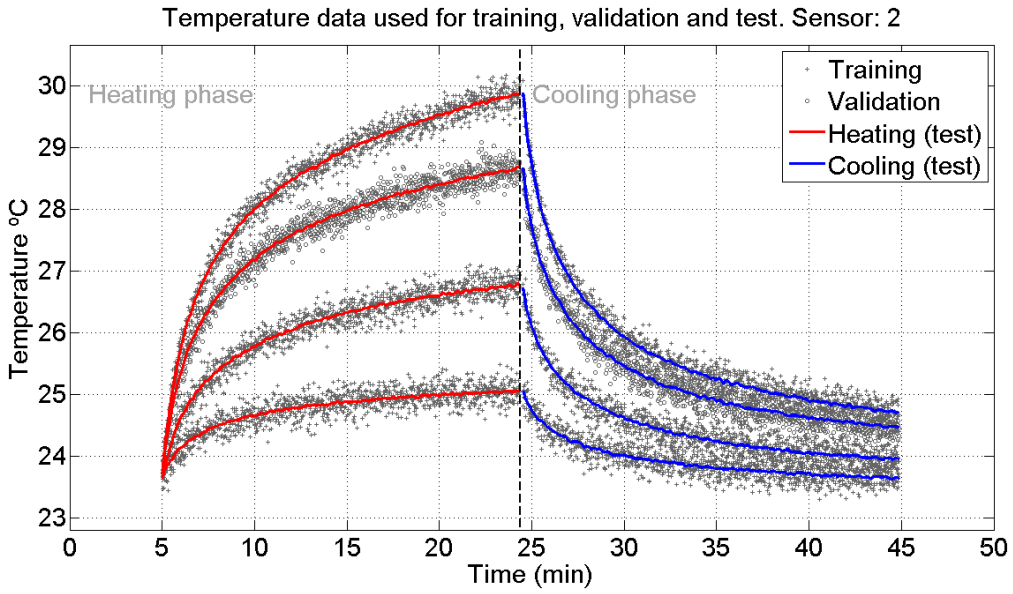

FIGURE D.1: Corrupted data after the completion of the Gaussian contamination process.. The noisy data is used for training and validation. The unaltered, noise free data is used to test the model. Collected from the *homogeneous phantom* experimental setup. The top curve corresponds to the strongest intensity. TUS intensity (from the shortest curve to the tallest curve): $0.5W/cm^2$, $1.0W/cm^2$, $1.5W/cm^2$ and $1.8W/cm^2$. Sensor 1.

Data collected at $0.5W/cm^2$, $1.0W/cm^2$ and $1.8W/cm^2$ was used for training and data collected at $1.5W/cm^2$ was used for validation. The model test assessment was done using the complete uncorrupted data set. After the construction of four distinct models, the performance criteria were calculated and exposed in Table (D.1).

Observing the performance descriptors, in particular the maximum absolute error M_{ae}, one can note a surprisingly low error threshold that the test error maintained. Albeit the data used for training and validation was contaminated with noise, the robustness of the models are verified, since the networks were able to fully comprehend the process dynamics. Furthermore note that the data collected at $1.5W/cm^2$ was used just for validation, which means that the network did not learned its parameters according to the validation patterns. Then the original uncorrupted data collected at $1.5W/cm^2$ was part of the test set. With these results, the models *interpolation* ability is successfully assessed, since the networks were able to predict one step ahead within a threshold lower then 0.05 ^oC (model 4). Furthermore the data that was indeed seen, was contaminated

TABLE D.1: Performance descriptors obtained concerning models with different number of input lags. The models presented were selected using the KTB approach. SPMI (Sensor 2). Data used: corrupted.

Model 1(2 lags)		
Criterion	Heating	Cooling
BIC	-10934	-12455
MSE	4.2035e-02	3.2463e-02
$MSRE$	1.5873e-03	1.3161e-03
MSE_v	4.5034e-02	3.2201e-02
$MSRE_v$	1.6322e-03	1.2775e-03
MSE_t	8.5816e-04	7.1760e-04
$MSRE_t$	3.3813e-05	2.9029e-05
M_{ae}	0.1912	0.1530
LWN	7	11
SR	e	n

Model 2(3 lags)		
Criterion	Heating	Cooling
BIC	-12071	-12648
MSE	2.9627e-02	3.0433e-02
$MSRE$	1.1268e-03	1.2361e-03
MSE_v	2.9403e-02	3.1046e-02
$MSRE_v$	1.0670e-03	1.2307e-03
MSE_t	1.3724e-04	4.1565e-04
$MSRE_t$	5.4154e-06	1.6847e-05
M_{ae}	0.0757	0.0789
LWN	15	15
SR	e	e

Model 3(4 lags)		
Criterion	Heating	Cooling
BIC	-12287	-12748
MSE	2.7490e-02	2.9199e-02
$MSRE$	1.0445e-03	1.1859e-03
MSE_v	2.9374e-02	2.7381e-02
$MSRE_v$	1.0639e-03	1.0857e-03
MSE_t	1.0497e-04	2.5406e-04
$MSRE_t$	4.0882e-06	1.0366e-05
M_{ae}	0.0536	0.0588
LWN	19	20
SR	n	n

Model 4(5 lags)		
Criterion	Heating	Cooling
BIC	-12265	-13231
MSE	2.7325e-02	2.5278e-02
$MSRE$	1.0385e-03	1.0265e-03
MSE_v	2.7823e-02	2.6139e-02
$MSRE_v$	1.0077e-03	1.0371e-03
MSE_t	9.9212e-05	1.7824e-04
$MSRE_t$	3.8351e-06	7.2434e-06
M_{ae}	0.0392	0.0561
LWN	23	24
SR	n	n

with noise, which reinforces the interpolation ability of the created BSNNs. Figures (D.2), (D.3), (D.4) and (D.5) depict the output of model 4 assessment over the complete test set.

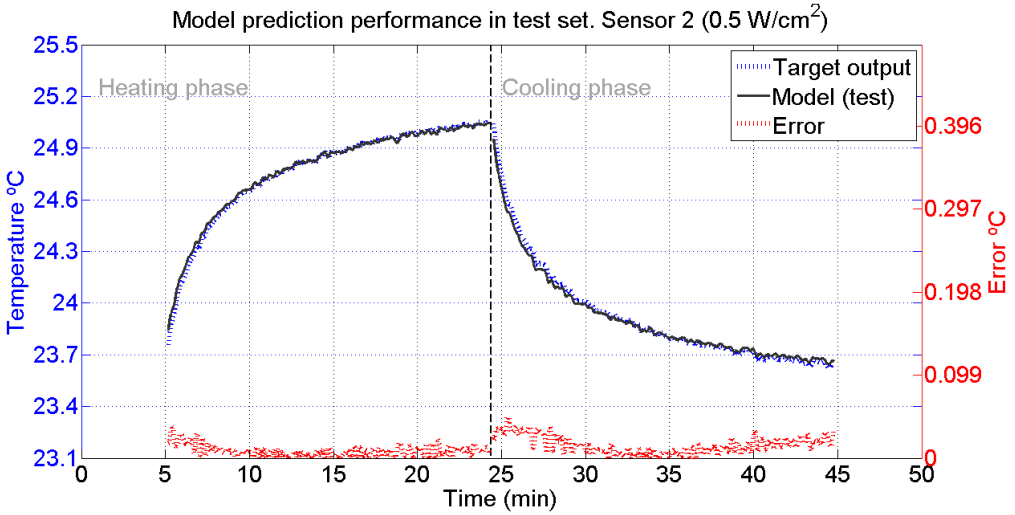

FIGURE D.2: Behaviour of model 4 in the test set (Sensor 2 $0.5W/cm^2$), selected using the KTB approach. The blue line represents the desired behaviour and the model's test output is given by the black line. The error line is red. SIMP, *homogeneous phantom* experimental setup. Data used: corrupted.

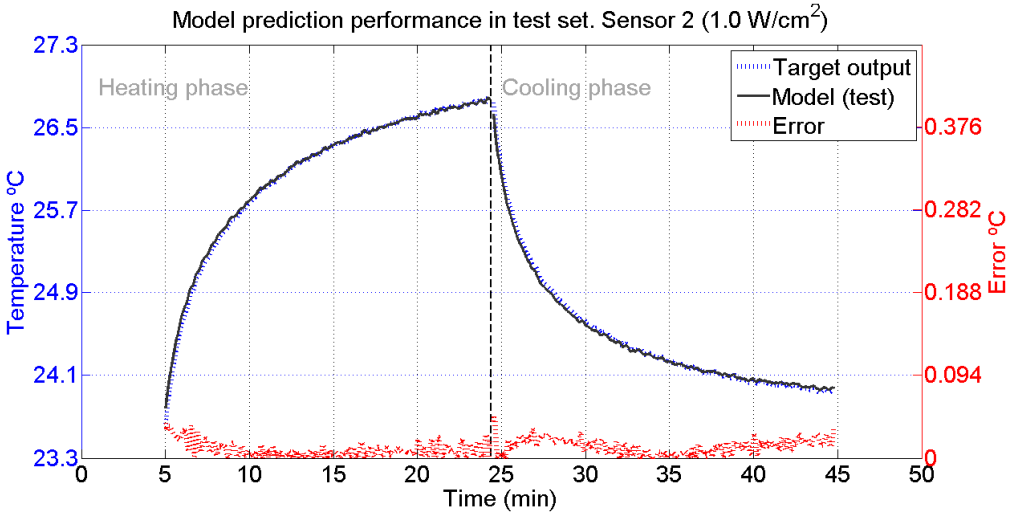

FIGURE D.3: Behaviour of model 4 in the test set (Sensor 2 $1.0W/cm^2$), selected using the KTB approach. The blue line represents the desired behaviour and the model's test output is given by the black line. The error line is red. SIMP, *homogeneous phantom* experimental setup. Data used: corrupted.

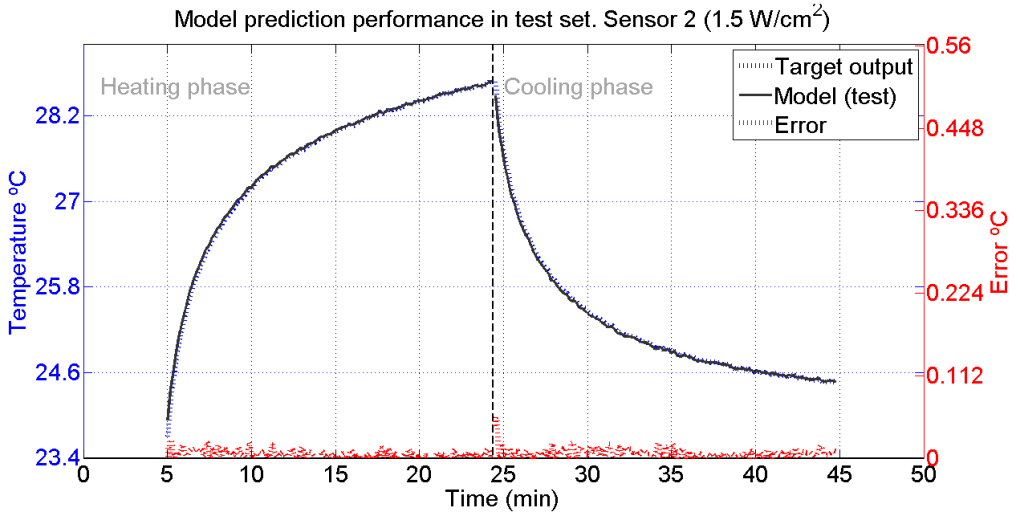

FIGURE D.4: Behaviour of model 4 in the test set (Sensor 2 $1.5W/cm^2$), selected using the KTB approach. The blue line represents the desired behaviour and the model's test output is given by the black line. The error line is red. SIMP, *homogeneous phantom* experimental setup. Data used: corrupted.

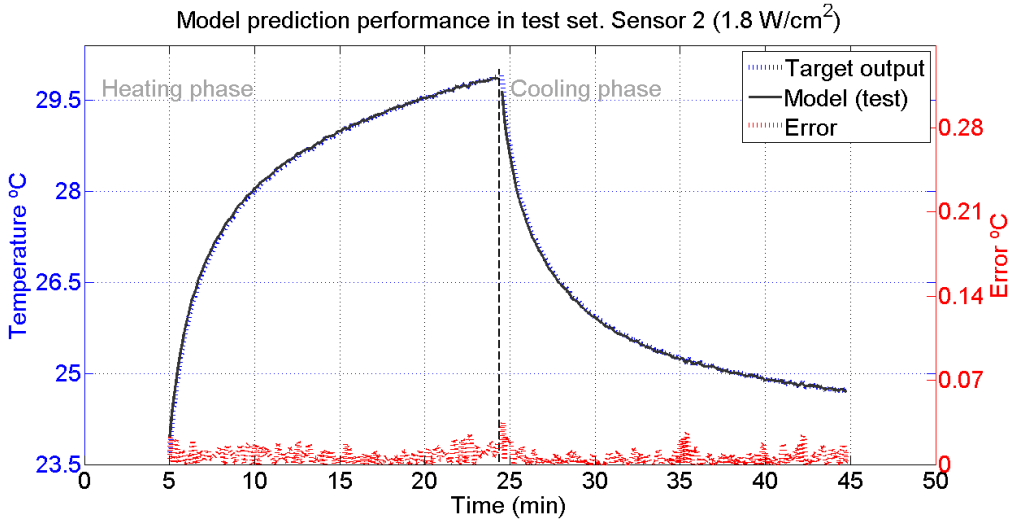

FIGURE D.5: Behaviour of model 4 in the test set (Sensor 2 $1.8W/cm^2$), selected using the KTB approach. The blue line represents the desired behaviour and the model's test output is given by the black line. The error line is red. SIMP, *homogeneous phantom* experimental setup. Data used: corrupted.

Albeit the contamination present in the data set used to train and validate the model, we observe a robust and successful learning of the process dynamics. The assessment of the ensemble methods follows. Table (D.2) presents the performance criteria calculated for this methods. Table (D.3) confronts the ensemble performance in the test set with the descriptors calculated for the KTB approach.

TABLE D.2: Performance comparison between all methodologies employed. SPMI (Sensor 2). Data used: corrupted.

Model 1	Heating			Cooling		
Criterion	MSE_v	MSE_t	M_{ae}	MSE_v	MSE_t	M_{ae}
KTB	4.5034e-02	8.5816e-04	0.1912	3.2201e-02	7.1760e-04	0.1530
Ensemble (SA)	2.3568e-02	2.4316e-04	0.1912	2.2742e-02	3.5411e-04	0.1530
Ensemble optimized (ES)	2.3380e-02	8.0554e-05	0.1912	2.2540e-02	2.0317e-04	0.1530
NDEO	2.3461e-02	4.5211e-05	0.0331	2.2621e-02	1.2487e-04	0.0698

Model 2	Heating			Cooling		
Criterion	MSE_v	MSE_t	M_{ae}	MSE_v	MSE_t	M_{ae}
KTB	2.9403e-02	1.3724e-04	0.0757	3.1046e-02	4.1565e-04	0.0789
Ensemble (SA)	2.2879e-02	2.1719e-04	0.1302	2.3114e-02	3.4762e-04	0.1077
Ensemble optimized (ES)	2.2807e-02	5.4445e-05	0.1302	2.2971e-02	1.9656e-04	0.1369
NDEO	2.2762e-02	5.3853e-05	0.0188	2.2985e-02	6.7114e-05	0.0580

Model 3	Heating			Cooling		
Criterion	MSE_v	MSE_t	M_{ae}	MSE_v	MSE_t	M_{ae}
KTB	2.9374e-02	1.0497e-04	0.0536	2.7381e-02	2.5406e-04	0.0588
Ensemble (SA)	2.3142e-02	2.0657e-04	0.1020	2.2142e-02	3.4410e-04	0.0663
Ensemble optimized (ES)	2.3032e-02	4.3681e-05	0.1020	2.1841e-02	1.9292e-04	0.1369
NDEO	2.3018e-02	5.7776e-05	0.0254	2.2052e-02	9.7936e-05	0.0510

Model 4	Heating			Cooling		
Criterion	MSE_v	MSE_t	M_{ae}	MSE_v	MSE_t	M_{ae}
KTB	2.7823e-02	9.9212e-05	0.0392	2.6139e-02	1.7824e-04	0.0561
Ensemble (SA)	2.3271e-02	2.0107e-04	0.0836	2.2812e-02	3.4274e-04	0.0649
Ensemble optimized (ES)	2.3105e-02	3.8042e-05	0.0217	2.2552e-02	1.9143e-04	0.1369
NDEO	2.3206e-02	5.2366e-05	0.0494	2.2722e-02	5.3971e-05	0.0558

Once again the NDEO approach outperformed the others methods by a considerable amount. Nevertheless the ES optimization scheme also provided a generalization performance enhancement comparable with the results obtained with this method when considering SPSI model typology. Thus the ES method is providing evidence of being scalable approach in terms of complexity. Ideally this method should maintain the same levels of enhancement regardless of the modelling environment complexity. Such a method would be highly desirable.

TABLE D.3: Generalization error obtained with the ensemble approaches in comparison with the KTB model selection. SPMI (Sensor 2). Data used: corrupted.

Model 1 (2 Lags)	**Heating**	**Cooling**	**Average comparison**
Ensemble (SA)	71.66 %	50.65 %	61.16 %
Ensemble optimized (ES)	90.61 %	71.69 %	81.15 %
NDEO	94.73 %	82.60 %	88.66 %

Model 2 (3 Lags)	**Heating**	**Cooling**	**Average comparison**
Ensemble (SA)	-58.25 %	16.37 %	-20.94 %
Ensemble optimized (ES)	60.33 %	52.71 %	56.52 %
NDEO	60.76 %	83.85 %	72.31 %

Model 3 (4 Lags)	**Heating**	**Cooling**	**Average comparison**
Ensemble (SA)	-96.80 %	-35.44 %	-66.12 %
Ensemble optimized (ES)	58.39 %	24.06 %	41.23 %
NDEO	44.96 %	61.45 %	53.20 %

Model 2 (4 Lags)	**Heating**	**Cooling**	**Average comparison**
Ensemble (SA)	-102.67 %	-92.29 %	-97.48 %
Ensemble optimized (ES)	61.66 %	-7.40 %	27.13 %
NDEO	47.22 %	69.72 %	58.47 %

Model environment: Sensor (5), all TUS intensities

To not exhaust the reader we shall pass directly to the last sensor (Sensor 5), the furthest sensor relatively to the TUS face, Figure (3.4. Nevertheless it should be mentioned that the results concerning sensor 3 and 4 are in the line of results being presented.

Figure (D.6) depicts the noisy data used to train ($0.5W/cm^2$, $1.5W/cm^2$ and $1.8W/cm^2$) and validate ($1.0W/cm^2$) the model. The whole uncorrupted data set was used in the test set.

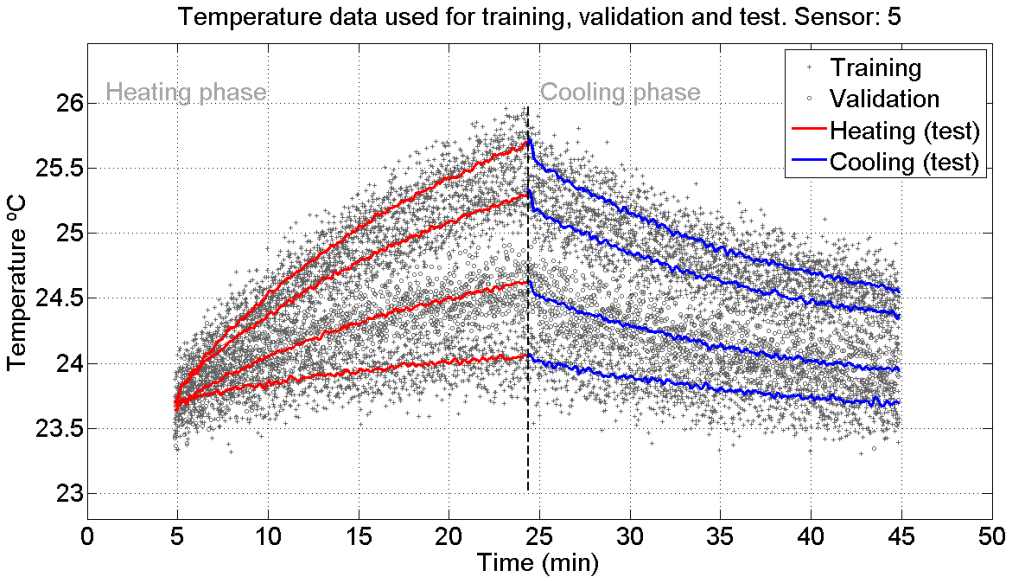

FIGURE D.6: Corrupted data after the completion of the Gaussian contamination process.. The noisy data is used for training and validation. The unaltered, noise free data is used to test the model. Collected from the *homogeneous phantom* experimental setup. The top curve corresponds to the strongest intensity. TUS intensity: $0.5W/cm^2$, $1.0W/cm^2$, $1.5W/cm^2$ and $1.8W/cm^2$. Sensor 1.

The performance of the four distinct build models is presented in Table (D.4).

The assessment of model 4 in the test set is illustrated through Figures (D.7), (D.8), (D.9) and (D.10). This plotting expose a successful learning process, where the erros were consistently kept under a 0.16 ^oC error threshold, despite the training phase use of the highly corrupted data.

TABLE D.4: Performance descriptors obtained concerning models with different number of input lags. The models presented were selected using the KTB approach. SPMI (Sensor 5). Data used: corrupted.

Model 1(2 lags)		
Criterion	Heating	Cooling
BIC	-12013	-12945
MSE	3.1806e-02	2.9228e-02
$MSRE$	1.2977e-03	1.1911e-03
MSE_v	3.5268e-02	3.2856e-02
$MSRE_v$	1.4527e-03	1.3623e-03
MSE_t	1.5687e-03	2.4346e-03
$MSRE_t$	6.4969e-05	1.0055e-04
M_{ae}	0.1764	0.1396
LWN	11	11
SR	n	n

Model 2(3 lags)		
Criterion	Heating	Cooling
BIC	-12158	-12871
MSE	3.0144e-02	2.9472e-02
$MSRE$	1.2305e-03	1.2036e-03
MSE_v	3.1034e-02	2.7756e-02
$MSRE_v$	1.2794e-03	1.1485e-03
MSE_t	9.3066e-04	1.4004e-03
$MSRE_t$	3.8475e-05	5.7846e-05
M_{ae}	0.1391	0.1126
LWN	15	15
SR	n	n

Model 3(4 lags)		
Criterion	Heating	Cooling
BIC	-12382	-13243
MSE	2.7934e-02	2.6327e-02
$MSRE$	1.1407e-03	1.0745e-03
MSE_v	2.7678e-02	2.8223e-02
$MSRE_v$	1.1399e-03	1.1694e-03
MSE_t	6.7565e-04	9.0539e-04
$MSRE_t$	2.7927e-05	3.7410e-05
M_{ae}	0.1178	0.0910
LWN	19	19
SR	n	e

Model 4(5 lags)		
Criterion	Heating	Cooling
BIC	-12607	-13109
MSE	2.5872e-02	2.6982e-02
$MSRE$	1.0576e-03	1.1025e-03
MSE_v	2.6563e-02	2.8463e-02
$MSRE_v$	1.0944e-03	1.1783e-03
MSE_t	3.9436e-04	6.1172e-04
$MSRE_t$	1.6334e-05	2.5294e-05
M_{ae}	0.0977	0.0842
LWN	23	23
SR	n	n

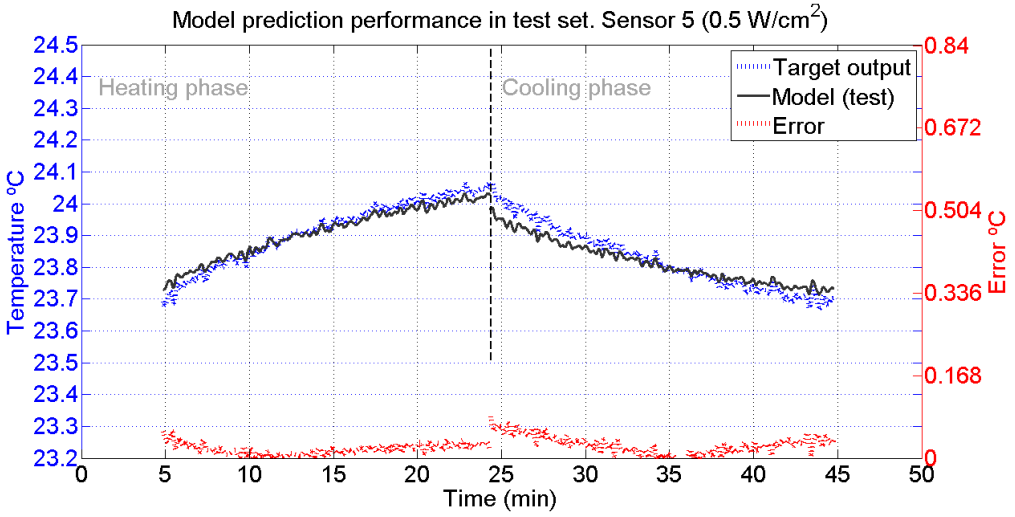

FIGURE D.7: Behaviour of model 4 in the test set (Sensor 5 $0.5W/cm^2$), selected using the KTB approach. The blue line represents the desired behaviour and the model's test output is given by the black line. The error line is red. SIMP, *homogeneous phantom* experimental setup. Data used: corrupted.

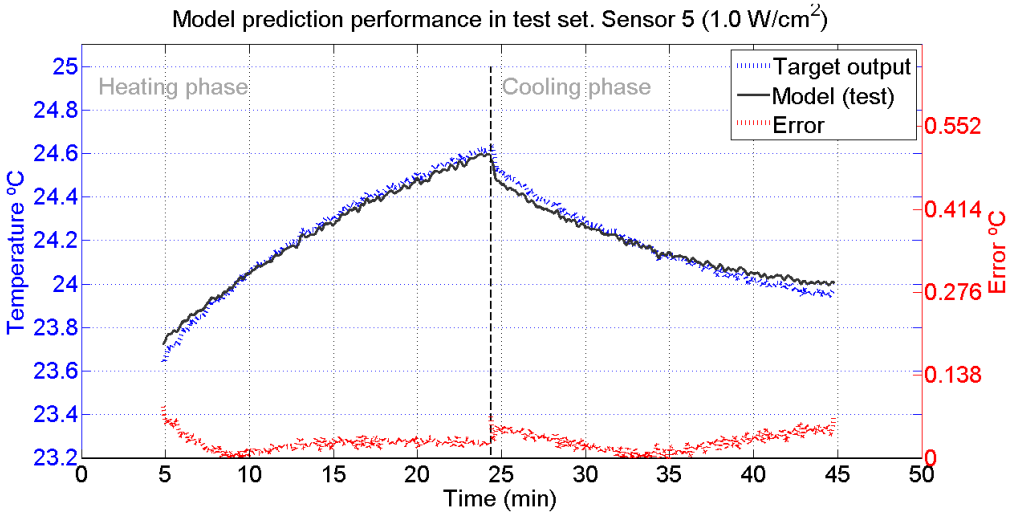

FIGURE D.8: Behaviour of model 4 in the test set (Sensor 5 $1.0 W/cm^2$), selected using the KTB approach. The blue line represents the desired behaviour and the model's test output is given by the black line. The error line is red. SIMP, *homogeneous phantom* experimental setup. Data used: corrupted.

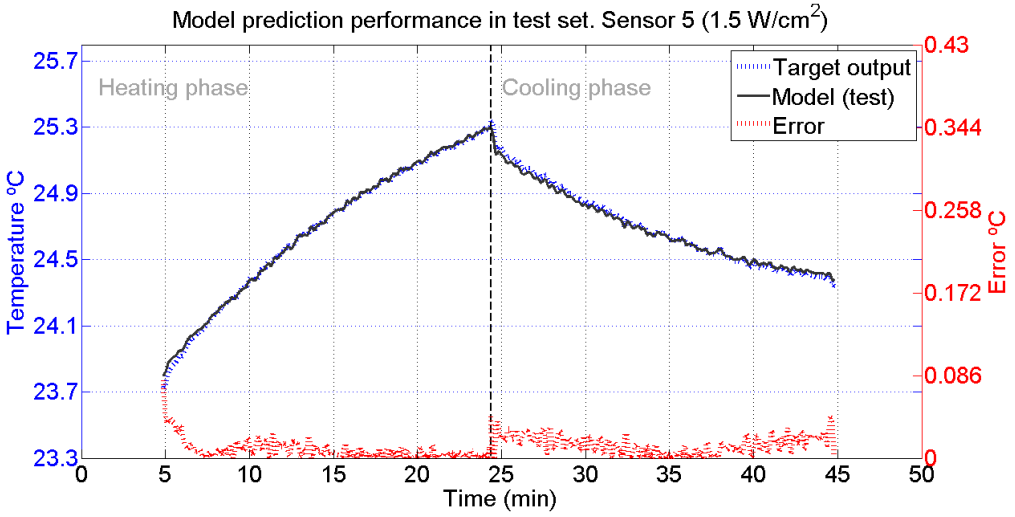

FIGURE D.9: Behaviour of model 4 in the test set (Sensor 5 $1.5 W/cm^2$), selected using the KTB approach. The blue line represents the desired behaviour and the model's test output is given by the black line. The error line is red. SIMP, *homogeneous phantom* experimental setup. Data used: corrupted.

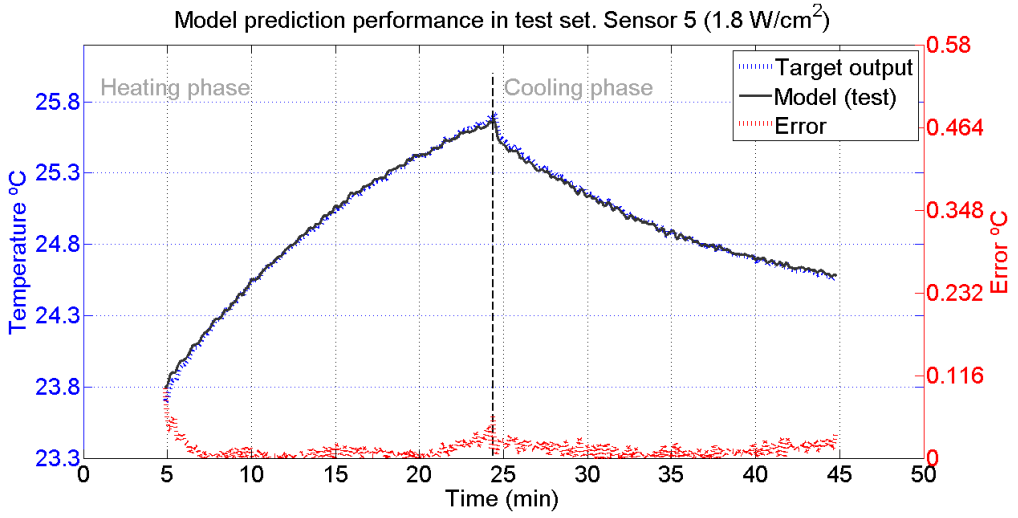

FIGURE D.10: Behaviour of model 4 in the test set (Sensor 5 $1.8W/cm^2$), selected using the KTB approach. The blue line represents the desired behaviour and the model's test output is given by the black line. The error line is red. SIMP, *homogeneous phantom* experimental setup. Data used: corrupted.

In an attempt to enhance the predictions, the discussed ensemble methods were applied, by forming an ensemble of four forecasting entities (models). The evaluation of these methods is reflected in Table (D.5) and the comparison with the KTB approach is made in Table (D.6).

This last SPMI environment revealed ES performance enhancements comparable with the ones obtained using NDEO. Both approaches provide a considerable gain in the forecasting performance. It's by this time evident that the NDEO approach requires perfect conditions for performance enhancements. *Perfect conditions* assumes a ensemble composed by uncorrelated models. Furthermore all this uncorrelated models should be good models. Achieving both these requirements might not be a trivial task.

TABLE D.5: Performance comparison between all methodologies employed. SPMI (Sensor 5). Data used: corrupted.

Model 1	Heating			Cooling		
Criterion	MSE_v	MSE_t	M_{ae}	MSE_v	MSE_t	M_{ae}
KTB	3.5268e-02	1.5687e-03	0.1764	3.2856e-02	2.4346e-03	0.1396
Ensemble (SA)	2.3769e-02	8.6071e-04	0.1761	2.6073e-02	1.2587e-03	0.1392
Ensemble optimized (ES)	2.2885e-02	1.1745e-04	0.1761	2.4082e-02	1.5402e-04	0.1392
NDEO	2.3053e-02	8.1659e-05	0.1184	2.4165e-02	1.0835e-04	0.0460

Model 2	Heating			Cooling		
Criterion	MSE_v	MSE_t	M_{ae}	MSE_v	MSE_t	M_{ae}
KTB	3.1034e-02	9.3066e-04	0.1391	2.7756e-02	1.4004e-03	0.1126
Ensemble (SA)	2.4135e-02	8.4286e-04	0.1577	2.3590e-02	1.2493e-03	0.1261
Ensemble optimized (ES)	2.2856e-02	9.8962e-05	0.1577	2.1473e-02	1.4369e-04	0.1261
NDEO	2.2839e-02	4.2434e-05	0.0358	2.2104e-02	2.4255e-04	0.0905

Model 3	Heating			Cooling		
Criterion	MSE_v	MSE_t	M_{ae}	MSE_v	MSE_t	M_{ae}
KTB	2.7678e-02	6.7565e-04	0.1178	2.8223e-02	9.0539e-04	0.0910
Ensemble (SA)	2.2858e-02	8.2880e-04	0.1435	2.5293e-02	1.2419e-03	0.1139
Ensemble optimized (ES)	2.1828e-02	8.4273e-05	0.1435	2.2873e-02	1.3541e-04	0.1139
NDEO	2.1894e-02	4.4359e-05	0.0327	2.2954e-02	6.4314e-05	0.0446

Model 4	Heating			Cooling		
Criterion	MSE_v	MSE_t	M_{ae}	MSE_v	MSE_t	M_{ae}
KTB	2.6563e-02	3.9436e-04	0.0977	2.8463e-02	6.1172e-04	0.0842
Ensemble (SA)	2.3395e-02	8.1724e-04	0.1302	2.7380e-02	1.2362e-03	0.1054
Ensemble optimized (ES)	2.2098e-02	7.2076e-05	0.0398	2.4606e-02	1.2884e-04	0.0639
NDEO	2.2362e-02	1.0087e-04	0.0404	2.4507e-02	1.0688e-04	0.0550

TABLE D.6: Generalization error obtained with the ensemble approaches in comparison with the KTB model selection. SPMI (Sensor 5). Data used: corrupted.

Model 1 (2 Lags)	**Heating**	**Cooling**	**Average comparison**
Ensemble (SA)	45.13 %	48.30 %	46.72 %
Ensemble optimized (ES)	92.51 %	93.67 %	93.09 %
NDEO	94.79 %	95.55 %	95.17 %

Model 2 (3 Lags)	**Heating**	**Cooling**	**Average comparison**
Ensemble (SA)	9.43 %	10.79 %	10.11 %
Ensemble optimized (ES)	89.37 %	89.74 %	89.55 %
NDEO	95.44 %	82.68 %	89.06 %

Model 3 (4 Lags)	**Heating**	**Cooling**	**Average comparison**
Ensemble (SA)	-22.67 %	-37.17 %	-29.92 %
Ensemble optimized (ES)	87.53 %	85.04 %	86.29 %
NDEO	93.43 %	92.90 %	93.17 %

Model 2 (4 Lags)	**Heating**	**Cooling**	**Average comparison**
Ensemble (SA)	-107.23 %	-102.09 %	-104.66 %
Ensemble optimized (ES)	81.72 %	60.02 %	70.87 %
NDEO	74.42 %	82.53 %	78.48 %

E

Results for MPMI model typology, prediction horizon $h = 7$ seconds.

As discussed in Section(5.4), the prediction horizon was extended, moving towards an assessment of the approach scalability, which can provide a great insight regarding the robustness of the modeling approach. We now extend the prediction horizon h to 7 seconds., an increase by a multiplicative factor of 7.

Temperature data points concerning all intensities and spatial locations are now considered. The original uncorrupted data, is illustrated in Figure (E.1).

We start by creating four distinct models using this unaltered data. Again the distinction among the models resides in the number of input lags. The data division applied, concerning trainning, validtion and test sets can be found in Appendix(B), Model 1.

The performance criteria calculated for the distinct models are presented in Table (E.1).

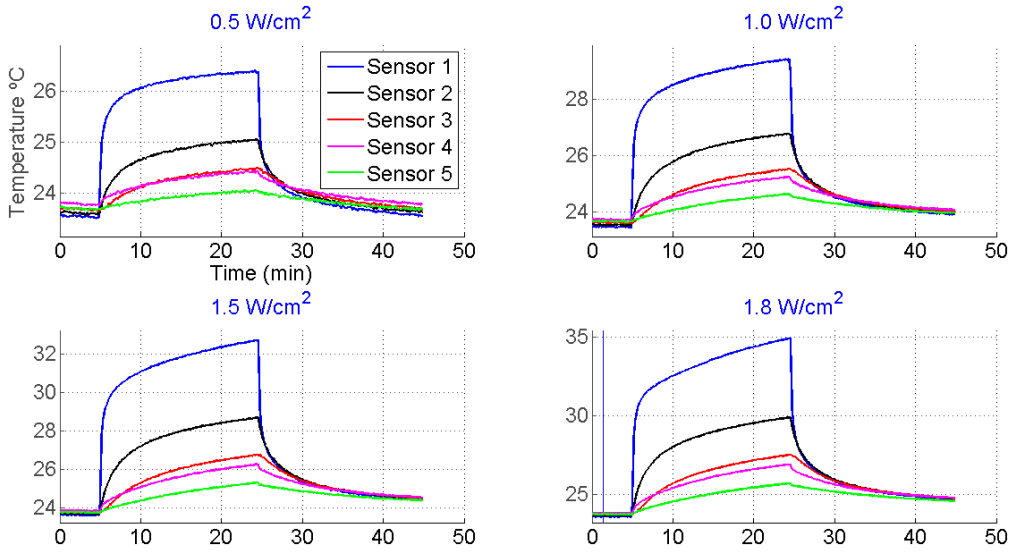

FIGURE E.1: Uncorrupted data set used for MPMI model training, validation and test. Collected from the *homogeneous phantom* experimental setup.

Albeit the increase of the prediction horizon to 7 seconds the performance descriptors obtained represent excellent figures. Concerning models 2, 3 and 4 the maximum absolute error M_{ae} obtained (including the test set) was kept below a threshold of 0.1 ^oC, which is a meritorious indicator about the predictions provided by the networks. Figures (E.2) and (E.3) expose the behaviour of model 4 in the test set (Sensor 3, $0.5W/cm^2$ and Sensor 5, $1.8W/cm^2$.)

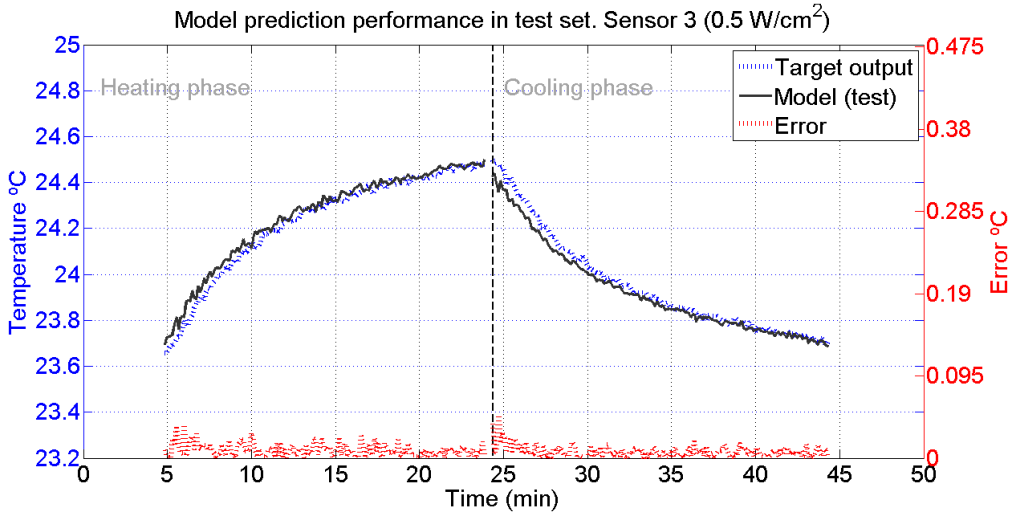

FIGURE E.2: Model's behaviour in the test set (Sensor 3 $0.5W/cm^2$), selected using the KTB approach. The blue line represents the desired behaviour and the model's test output is given by the black line. The error line is red. MIMP, *homogeneous phantom* experimental setup. Data used: uncorrupted.

TABLE E.1: Performance descriptors obtained concerning models with different number of input lags. The models presented were selected using the KTB approach. MPMI. Data used: uncorrupted.

Model 1(2 lags)		
Criterion	Heating	Cooling
BIC	-20531	-18390
MSE	2.4201e-04	8.6867e-04
$MSRE$	9.2701e-06	3.2581e-05
MSE_v	2.5127e-04	2.2165e-04
$MSRE_v$	9.9201e-06	8.7704e-06
MSE_t	2.3355e-04	2.3393e-04
$MSRE_t$	9.3110e-06	9.2870e-06
M_{ae}	0.1537	0.7401
LWN	28	8
SR	e	n

Model 2(3 lags)		
Criterion	Heating	Cooling
BIC	-20975	-23567
MSE	1.8933e-04	9.9935e-05
$MSRE$	7.1766e-06	4.0305e-06
MSE_v	2.1658e-04	1.1034e-04
$MSRE_v$	8.5972e-06	4.4736e-06
MSE_t	2.1444e-04	1.4472e-04
$MSRE_t$	8.5231e-06	5.6957e-06
M_{ae}	0.0920	0.0531
LWN	33	52
SR	e	n

Model 3(4 lags)		
Criterion	Heating	Cooling
BIC	-20875	-23671
MSE	1.8780e-04	8.9119e-05
$MSRE$	7.1088e-06	3.5921e-06
MSE_v	1.6165e-04	9.9556e-05
$MSRE_v$	6.4897e-06	4.0364e-06
MSE_t	1.8281e-04	1.0716e-04
$MSRE_t$	7.5338e-06	4.4885e-06
M_{ae}	0.0854	0.0482
LWN	33	59
SR	e	n

Model 4(5 lags)		
Criterion	Heating	Cooling
BIC	-21712	-23766
MSE	1.1461e-04	8.0350e-05
$MSRE$	4.3600e-06	3.2463e-06
MSE_v	1.0414e-04	8.9524e-05
$MSRE_v$	4.0603e-06	3.5912e-06
MSE_t	1.3007e-04	1.0946e-04
$MSRE_t$	5.3774e-06	4.5481e-06
M_{ae}	0.0444	0.0477
LWN	63	63
SR	e	n

These two figures represent network's ability to *interpolate* and *extrapolate* the data. The data concerning sensor 5 $1.8W/cm^2$ represents a test to the extrapolation ability of the network, since it consists in data captured in one extremity, while the data captured by Sensor 3 at $0.5W/cm^2$ serves as a test to the network's ability to interpolate since the network has been shown examples from both sides.

The ensemble approaches performance figures are shown in Table (E.2) and the generalization performance between the two paradigms (single model and model ensemble) can be observed in Table (E.3). Observing Table (E.1) one can notice that Model 4 consistently exhibits better performance figures than all the others models, hence the

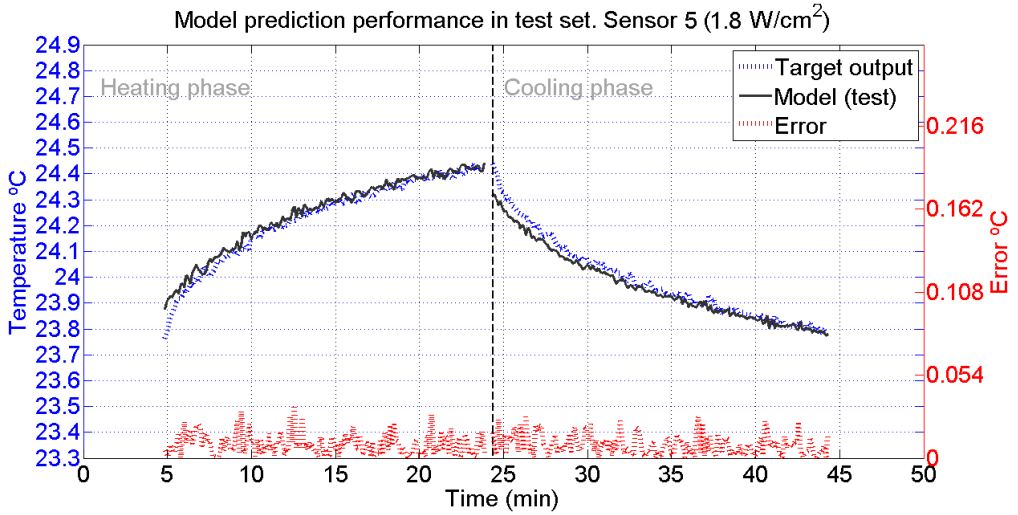

FIGURE E.3: Model's behaviour in the test set (Sensor 5 $1.8W/cm^2$), selected using the KTB approach. The blue line represents the desired behaviour and the model's test output is given by the black line. The error line is red. MIMP, *homogeneous phantom* experimental setup. Data used: uncorrupted.

ensemble approaches comparison reveals that this methods might not be justifiable as shown in Table (E.3), where in the last Model we do not observe a performance improvement, hence the overhead introduced in the system is not justifiable. This observations are in compliance with the remarks that an ensemble of networks requires individuals with comparable performances. Moreover by using the original data, the models might suffer from positive correlation.

Analogously to the previous model typologies, we now consider a noisy contaminated data set. The noise e_i was taken from a Gaussian distribution $e_i \sim \mathcal{N}(0, \sigma)$, Figure (4.12). The Gaussian contaminated data set is shown in Figure (E.4).

TABLE E.2: Performance comparison between all methodologies employed. MPMI. Data used: uncorrupted.

Model 1	Heating			Cooling		
Criterion	MSE_v	MSE_t	M_{ae}	MSE_v	MSE_t	M_{ae}
KTB	2.5127e-04	2.3355e-04	0.1537	2.2165e-04	2.3393e-04	0.7401
Ensemble (SA)	1.9166e-04	1.8801e-04	0.1537	1.3265e-04	1.3525e-04	0.7401
Ensemble optimized (ES)	1.6627e-04	1.4588e-04	0.1537	1.2393e-04	1.1528e-04	0.7401
NDEO	1.6607e-04	1.4305e-04	0.1520	1.2577e-04	1.3327e-04	0.7248

Model 2	Heating			Cooling		
Criterion	MSE_v	MSE_t	M_{ae}	MSE_v	MSE_t	M_{ae}
KTB	2.1658e-04	2.1444e-04	0.0920	1.1034e-04	1.4472e-04	0.0531
Ensemble (SA)	1.8044e-04	1.7488e-04	0.0673	1.0004e-04	1.2156e-04	0.1542
Ensemble optimized (ES)	1.4002e-04	1.3249e-04	0.0673	9.1472e-05	1.0148e-04	0.1542
NDEO	1.3978e-04	1.3046e-04	0.0686	8.9444e-05	9.8966e-05	0.0512

Model 3	Heating			Cooling		
Criterion	MSE_v	MSE_t	M_{ae}	MSE_v	MSE_t	M_{ae}
KTB	1.6165e-04	1.8281e-04	0.0854	9.9556e-05	1.0716e-04	0.0482
Ensemble (SA)	1.4136e-04	1.4379e-04	0.0673	8.8032e-05	1.0580e-04	0.0569
Ensemble optimized (ES)	1.1552e-04	1.0787e-04	0.0673	7.6662e-05	9.4693e-05	0.0569
NDEO	1.1669e-04	1.0722e-04	0.0683	7.9205e-05	9.3611e-05	0.0466

Model 4	Heating			Cooling		
Criterion	MSE_v	MSE_t	M_{ae}	MSE_v	MSE_t	M_{ae}
KTB	1.0414e-04	1.3007e-04	0.0444	8.9524e-05	1.0946e-04	0.0477
Ensemble (SA)	1.2552e-04	1.4288e-04	0.0534	9.0284e-05	8.7823e-05	0.0429
Ensemble optimized (ES)	1.0279e-04	1.2691e-04	0.0436	8.7702e-05	9.9710e-05	0.0457
NDEO	1.0409e-04	1.3003e-04	0.0444	8.9534e-05	1.0950e-04	0.0477

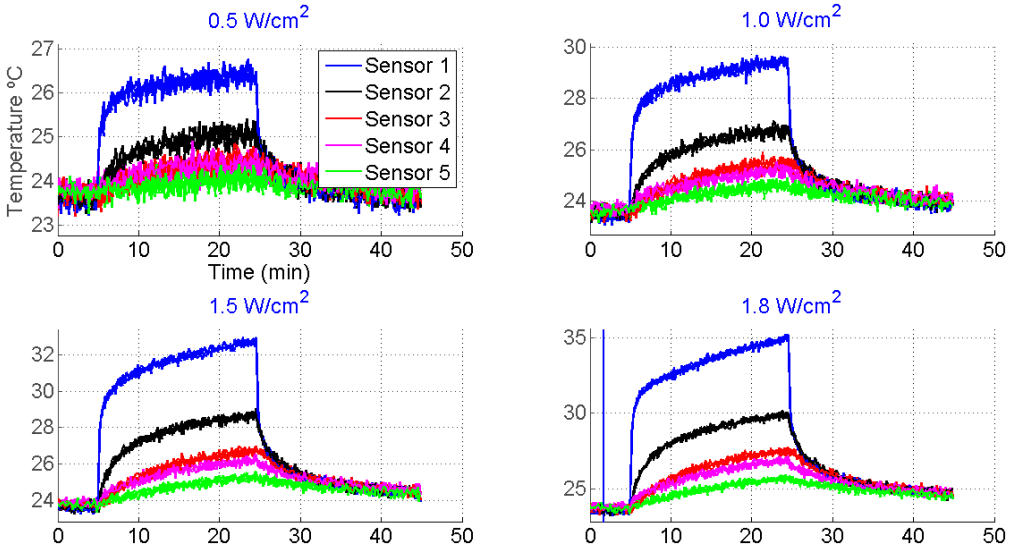

FIGURE E.4: Corrupted data obtained after the Gaussian contamination process. The corrupted data is used for training and validation. The uncorrupted original data is used to test the model. Collected from the *homogeneous phantom* experimental setup.

TABLE E.3: Generalization error obtained with the ensemble approaches in comparison with the KTB model selection. MPMI. Data used: uncorrupted.

Model 1 (2 Lags)	**Heating**	**Cooling**	**Average comparison**
Ensemble (SA)	19.50 %	42.18 %	30.84 %
Ensemble optimized (ES)	37.54 %	50.72 %	44.13 %
NDEO	38.75 %	43.03 %	40.89 %

Model 2 (3 Lags)	**Heating**	**Cooling**	**Average comparison**
Ensemble (SA)	18.45 %	16.00 %	17.23 %
Ensemble optimized (ES)	38.21 %	29.88 %	34.05 %
NDEO	39.16 %	31.62 %	35.39 %

Model 3 (4 Lags)	**Heating**	**Cooling**	**Average comparison**
Ensemble (SA)	21.35 %	1.27 %	11.31 %
Ensemble optimized (ES)	40.99 %	11.64 %	26.31 %
NDEO	41.35 %	12.65 %	27.00 %

Model 4 (5 Lags)	**Heating**	**Cooling**	**Average comparison**
Ensemble (SA)	-9.85 %	19.77 %	4.96 %
Ensemble optimized (ES)	2.42 %	8.91 %	5.67 %
NDEO	0.03 %	-0.03 %	-0.00 %

Using this noisy data, four models were again build and assessed. The data splitting regarding all models was done by merging the test set into the validation set. The data used to test the models was the complete uncorrupted data set, concerning all models and intensities, hence the models were assessed using the data shown in Figure (E.1).

The performance figures obtained for each one of the four built models is presented in Table (E.4).

Analyzing the performance indicators one can conclude that, despite the data contamination, the models were able to learn the process true dynamics, which proves the robustness of the modelling approach and evince the approximation power of BSNNs. The model was assessed using the complete original data set, i.e. all the curves present in Figure (E.1). One can observe that the maximum absolute errors M_{ae} were kept under simperingly low thresholds even when the networks are trained with highly contaminated data. For instance, Model 4 kept M_{ae} under 0.35 ^{o}C through all the test set, with a one step prediction horizon of 7 seconds. Figures (E.5), (E.6), (E.7), (E.8) and (E.9) demonstrate Model's behaviour in the test set for different intensities and spatial locations.

TABLE E.4: Performance descriptors obtained concerning models with different number of input lags. The models presented were selected using the KTB approach. MPMI. Data used: corrupted.

Model 1(2 lags)		
Criterion	Heating	Cooling
BIC	-9372	-9547
MSE	2.7037e-02	3.2030e-02
$MSRE$	1.0312e-03	1.2991e-03
MSE_v	2.6722e-02	2.9949e-02
$MSRE_v$	1.0429e-03	1.2001e-03
MSE_t	1.9728e-03	1.5113e-03
$MSRE_t$	7.5553e-05	5.9482e-05
M_{ae}	0.6428	0.7287
LWN	29	8
SR	n	n

Model 2(3 lags)		
Criterion	Heating	Cooling
BIC	-9784	-10066
MSE	2.2369e-02	2.5688e-02
$MSRE$	8.5683e-04	1.0447e-03
MSE_v	2.1470e-02	2.4964e-02
$MSRE_v$	8.3985e-04	1.0003e-03
MSE_t	1.8568e-03	1.0050e-03
$MSRE_t$	7.0785e-05	3.9572e-05
M_{ae}	0.4668	0.6075
LWN	33	13
SR	n	n

Model 3(4 lags)		
Criterion	Heating	Cooling
BIC	-9596	-10171
MSE	2.4560e-02	2.3859e-02
$MSRE$	9.2981e-04	9.7178e-04
MSE_v	2.7118e-02	2.4098e-02
$MSRE_v$	1.0561e-03	9.6533e-04
MSE_t	2.0776e-03	7.1778e-04
$MSRE_t$	7.7306e-05	2.8442e-05
M_{ae}	0.7875	0.3870
LWN	18	18
SR	n	n

Model 4(5 lags)		
Criterion	Heating	Cooling
BIC	-9551	-10386
MSE	2.2650e-02	2.1279e-02
$MSRE$	8.6211e-04	8.6492e-04
MSE_v	2.3695e-02	1.9927e-02
$MSRE_v$	9.2650e-04	7.9681e-04
MSE_t	1.2554e-03	6.5287e-04
$MSRE_t$	4.7779e-05	2.5945e-05
M_{ae}	0.2745	0.3264
LWN	43	23
SR	n	n

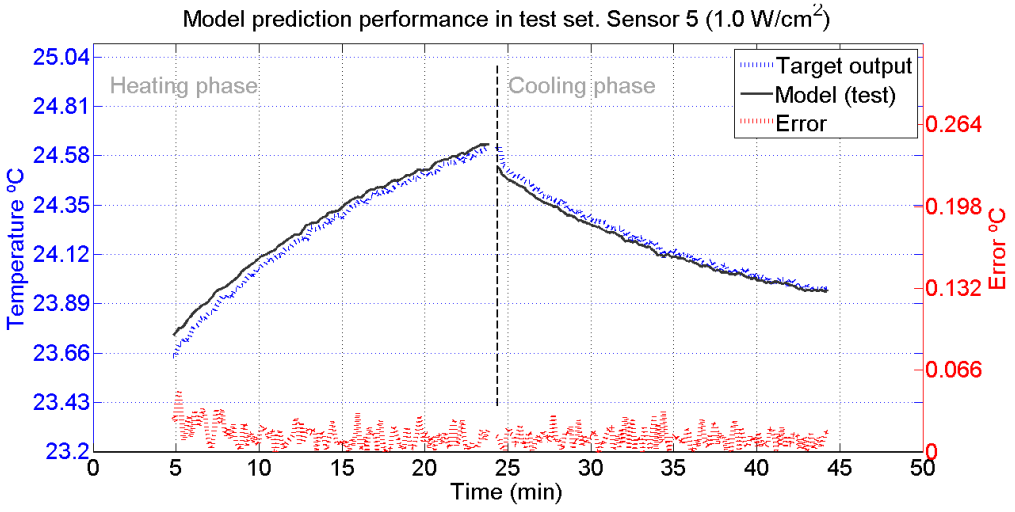

FIGURE E.9: Model's behaviour in the test set (Sensor 1 $1.8W/cm^2$), selected using the KTB approach. The blue line represents the desired behaviour and the model's test output is given by the black line. The error line is red. MIMP, *homogeneous phantom* experimental setup.

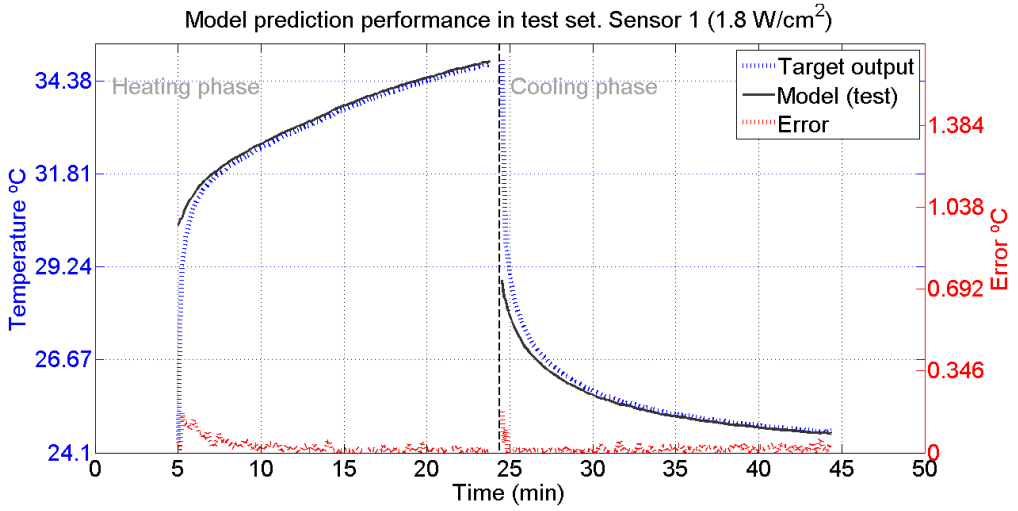

FIGURE E.5: Model's behaviour in the test set (Sensor 1 $1.8W/cm^2$), selected using the KTB approach. The blue line represents the desired behaviour and the model's test output is given by the black line. The error line is red. MIMP, *homogeneous phantom* experimental setup.

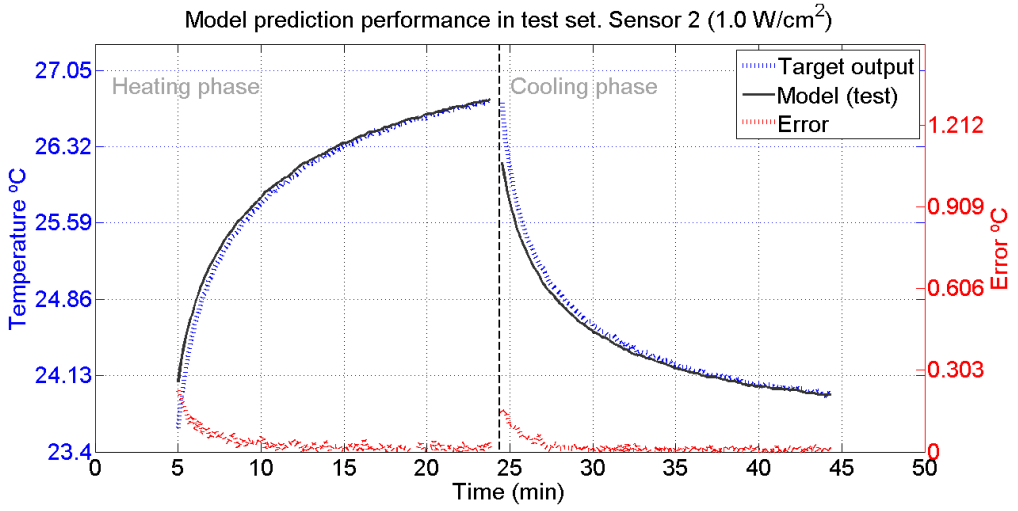

FIGURE E.6: Model's behaviour in the test set (Sensor 2 $1.0W/cm^2$), selected using the KTB approach. The blue line represents the desired behaviour and the model's test output is given by the black line. The error line is red. MIMP, *homogeneous phantom* experimental setup.

The test using data collected at Sensor 1 at $1.8W/cm^2$ represents the most challenging prediction zone, due the abrupt temperature evolution. However the network managed to follow the temperature variations, even with the increased prediction horizon. After the training of the four models, the ensembles approaches were applied. Their descriptors are assessed in Table (E.5) and the usual comparison with the KTB paradigm is done under to results shown in Table (E.6).

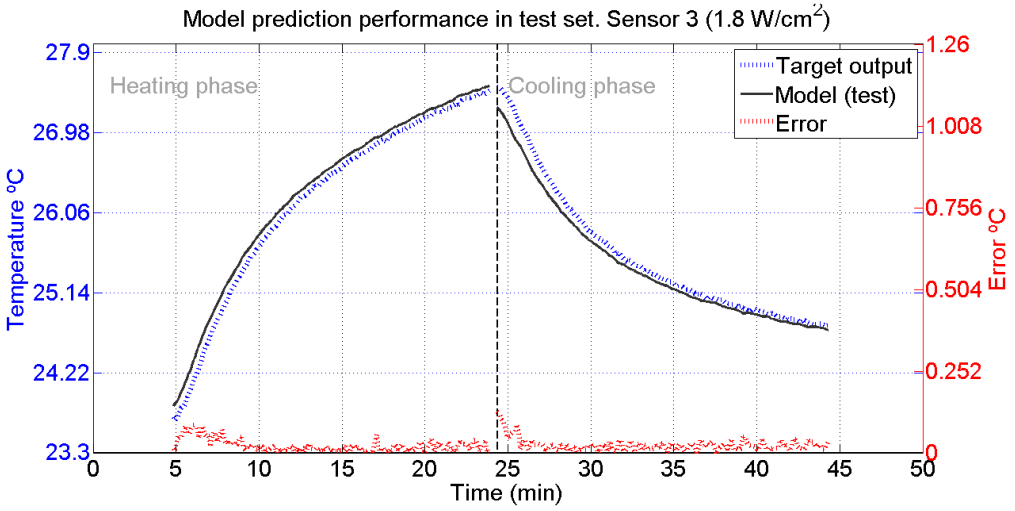

FIGURE E.7: Model's behaviour in the test set (Sensor 1 $1.8W/cm^2$), selected using the KTB approach. The blue line represents the desired behaviour and the model's test output is given by the black line. The error line is red. MIMP, *homogeneous phantom* experimental setup.

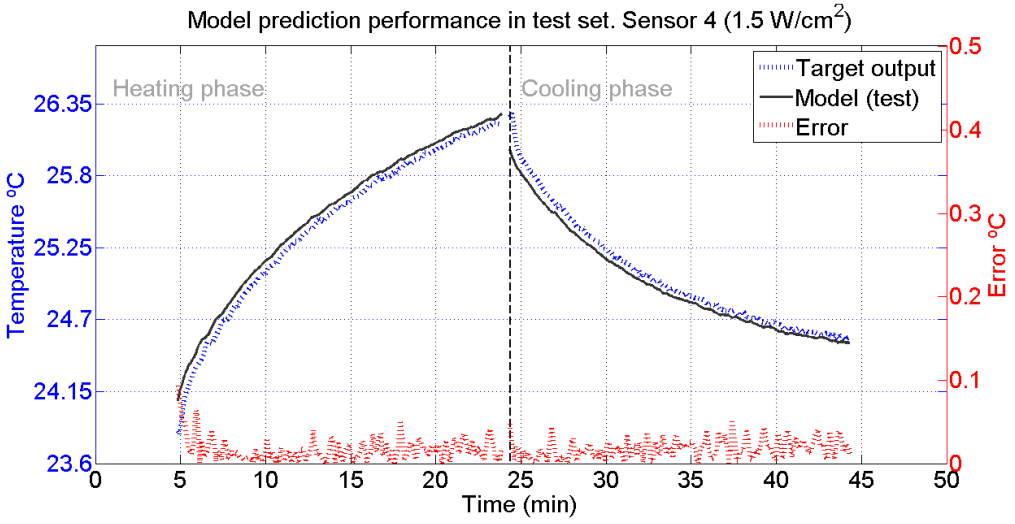

FIGURE E.8: Model's behaviour in the test set (Sensor 1 $1.8W/cm^2$), selected using the KTB approach. The blue line represents the desired behaviour and the model's test output is given by the black line. The error line is red. MIMP, *homogeneous phantom* experimental setup.

The generalization assessment of the ensemble's approaches revealed consistent performance improvements achieved by the three methods, with a tendency for better results when NDEO is applied. Nevertheless the performance gains encountered for this model typology are substantially worst when compared to the previous typologies. This fact is due to the outlier performance that a model admitting a higher number of input lags has over models that consider less inputs. As so, the ensemble has an individual whose

Table E.5: Performance comparison between all methodologies employed. MPMI

Model 1	Heating			Cooling		
Criterion	MSE_v	MSE_t	M_{ae}	MSE_v	MSE_t	M_{ae}
KTB	2.6722e-02	1.9728e-03	0.6428	2.9949e-02	1.5113e-03	0.7287
Ensemble (SA)	2.3894e-02	1.9472e-03	0.6428	2.1844e-02	1.1711e-03	0.7287
Ensemble optimized (ES)	2.3292e-02	1.9544e-03	0.6428	2.0864e-02	1.1893e-03	0.7287
NDEO	2.3503e-02	1.9949e-03	0.6372	2.0311e-02	6.0846e-04	0.3195

Model 2	Heating			Cooling		
Criterion	MSE_v	MSE_t	M_{ae}	MSE_v	MSE_t	M_{ae}
KTB	2.1470e-02	1.8568e-03	0.4668	2.4964e-02	1.0050e-03	0.6075
Ensemble (SA)	2.1529e-02	1.6396e-03	0.5090	2.3417e-02	8.1185e-04	0.3778
Ensemble optimized (ES)	2.0990e-02	1.6469e-03	0.5090	2.1850e-02	8.3013e-04	0.3778
NDEO	2.1293e-02	1.7106e-03	0.4637	2.1444e-02	8.5243e-04	0.4862

Model 3	Heating			Cooling		
Criterion	MSE_v	MSE_t	M_{ae}	MSE_v	MSE_t	M_{ae}
KTB	2.7118e-02	2.0776e-03	0.7875	2.4098e-02	7.1778e-04	0.3870
Ensemble (SA)	2.6551e-02	1.4333e-03	0.4229	2.5128e-02	6.9105e-04	0.3238
Ensemble optimized (ES)	2.6410e-02	1.4407e-03	0.4229	2.2239e-02	7.0943e-04	0.3238
NDEO	2.6694e-02	1.5387e-03	0.4925	2.1715e-02	6.3656e-04	0.3229

Model 4	Heating			Cooling		
Criterion	MSE_v	MSE_t	M_{ae}	MSE_v	MSE_t	M_{ae}
KTB	2.3695e-02	1.2554e-03	0.2745	1.9927e-02	6.5287e-04	0.3264
Ensemble (SA)	2.4220e-02	1.1765e-03	0.3531	2.2313e-02	6.2893e-04	0.2107
Ensemble optimized (ES)	2.3487e-02	1.1839e-03	0.3120	1.9942e-02	6.4741e-04	0.3210
NDEO	2.3695e-02	1.2551e-03	0.2745	1.9447e-02	5.6579e-04	0.2714

performance might not me comparable with the remainders, thus compromising the whole ensemble mechanism. However any performance gain, if its additional overhead is justified, are welcome in biomedical applications, which is a highly sensitive area, any improvement can make the difference.

TABLE E.6: Generalization error obtained with the ensemble approaches in comparison with the KTB model selection.

Model 1 (2 Lags)	**Heating**	**Cooling**	**Average comparison**
Ensemble (SA)	1.30 %	22.51 %	11.90 %
Ensemble optimized (ES)	0.93 %	21.31 %	11.12 %
NDEO	-1.12 %	59.74 %	29.31 %

Model 2 (3 Lags)	**Heating**	**Cooling**	**Average comparison**
Ensemble (SA)	11.69 %	19.22 %	15.46 %
Ensemble optimized (ES)	11.30 %	17.40 %	14.35 %
NDEO	7.87 %	15.18 %	11.53 %

Model 3 (4 Lags)	**Heating**	**Cooling**	**Average comparison**
Ensemble (SA)	31.01 %	3.72 %	17.37 %
Ensemble optimized (ES)	30.66 %	1.16 %	15.91 %
NDEO	25.94 %	11.32 %	18.63 %

Model 2 (4 Lags)	**Heating**	**Cooling**	**Average comparison**
Ensemble (SA)	6.28 %	3.67 %	4.98 %
Ensemble optimized (ES)	5.70 %	0.84 %	3.27 %
NDEO	0.02 %	13.34 %	6.68 %

Bibliography

[1] Dave Touretzky and Kornel Laskowski. *Neural Networks for Time Series Prediction*. Computer Sciences Department, University of Wisconsin-Madison, usa edition, 2006. 15-486/782:.

[2] Carl de Boor. B(asic)-spline basics, 2002.

[3] *MATLAB Curve Fitting Toolbox*. URL http://www.mathworks.com/products/curvefitting/.

[4] Prof. António Eduardo de Barros Ruano. *Artificial Neural Networks, p.147*. Centre for Intelligent Systems, University of Algarve, 2002.

[5] Ersu E. Tolle, H. Neurocontrol: Learning control systems inspired by neuronal architectures and human problem solving strategies. (ISBN 9780387550572), 1992.

[6] Simon Haykin. Neural networks: A comprehensive foundation. (ISBN 0-13-273350-1), 1998.

[7] Helder S. Duarte M. Graça Ruano. Estimating temperature in perfused tissue phantoms subject to ultrasound heating. pages p. 5,6.

[8] César Alexandre Domingues Teixeira. *Soft-computing techniques applied to invasive and non-invasive temperature estimation*. PhD thesis, Universidade do Algarve, 2008.

[9] Prof. Denis Zorin. *Bezier Curves and B-splines, Blossoming*. New York University, 2002.

[10] G. ter Haar. Therapeutic ultrasound. *Eur. J. Ultrasound*, 9(1):3–9, 1999.

[11] J. Civale I. Rivens, A. Shaw and H. Morris. Treatment monitoring and thermometry for therapeutic focused ultrasound. *Int. J. Hyperthermia*, 23(2):121–127, 2007.

[12] R. Seip and E. S. Ebbini. Invasive and non-invasive feedback for ultrasound phased array thermometry. *Ultrasonics Symposium IEEE*, 9(1):1821–1824, 1994.

[13] J. W. Trobaugh R. M. Arthur, W. L. Straube and E. G. Moros. Noninvasive temperature estimation of hyperthermia temperatures with ultrasound. *Int. J. Hyperthermia*, 21(6):589–600, 2005.

[14] Jerome H. Friedman. An overview of predictive learning and function approximation. *NATO ASI Series*, 136:1–61, 1994.

[15] Jiaguang Sun Huaiping Yang, Wenping Wang. Control point adjustment for b-spline curve approximation. *Elsevier*, 36(3):639–652, 2003.

[16] Timothy Irwin Mueller. *GEOMETRIC MODELLING WITH MULTIVARIATE B-SPLINES*. PhD thesis, The University of Utah, 1986.

[17] Fabio A. Guerra Leandro dos Santos Coelho. B-spline neural network design using improved differential evolution for identification of an experimental nonlinear process. *Elsevier*, 36(3):1513–1522, 2003.

[18] A. Ruano. *Applications of neural networks to control systems*. PhD thesis, UCNW, UK, 1992.

[19] Gerald E. Farin; Josef Hoschek; Myung-Soo Kim. *Handbook of Computer Aided Geometric Design*. Number ISBN 978-0-444-51104-1. Elsevier, 2002.

[20] I. J. SCHOENBERGC. Contributions to the problem of approximation of equidistant data by analytic functions,. *Quart. Appl. Math.*, 4:112–141, 1946.

[21] M.G. Cox. The numerical evaluation of b-splines. *J. Inst. Math. Appl.*, pages 134–139, 1972.

[22] Chris Harris Martin Brown. Neurofuzzy adaptive modelling and control. (ISBN 0-13-134453-6), 1994. p. 302.

[23] Mark Coppejans. *Breaking The Curse Of Dimensionality*. Deporatment of Economics, Duke University, usa edition, 2000.

[24] C. de Boor. *What is a multivariate spline?* Computer Sciences Department, University of Wisconsin-Madison, usa edition, 2000.

[25] Balázs Csanád Csáji. *Approximation with Artificial Neural Networks.* Faculty of Sciences; Eotovos Loránd University, Hungary, usa edition, 2000.

[26] Arrelwlh Itley Nachimuthu Karunanithdi and Yashwant K. Malaiy. Using neural networks in reliability prediction. 1992. IEEE software.

[27] Chris Harris Martin Brown. Neurofuzzy adaptive modelling and control. *Prentice Hall*, (ISBN 0-13-134453-6), 1994. section 8.3.1.

[28] Chris Harris Martin Brown. Neurofuzzy adaptive modelling and control. *Prentice Hall*, (ISBN 0-13-134453-6), 1994. p. 88,89.

[29] Chris Harris Martin Brown. Neurofuzzy adaptive modelling and control. *Prentice Hall*, (ISBN 0-13-134453-6), 1994. p.314,315.

[30] C. de Boor. *Splines as a linear combination of B-splines, a survey in Approximation Theory II.* Academic Press, new york edition, 1976. p. 1-47.

[31] C. de Boor. *On local linear functionals which vanish at all B-splines but one, in Theory of approximation with Applications (A. G. Law and B. H. Sahney).* Academic Press, new york edition, 1976. p. 120-146.

[32] T. Kavli. Asmod: An algorithm for adaptive spline modelling of observation data. *International Journal of Control*, 58(4):p. 947–968, 1993.

[33] Erik Weyer Tom Kavli. On asmod — an algorithm for empirical modelling using spline functions. *Neural Network Engineering in Dynamic Control Systems Advances in Industrial Control*, Springer:p. 83–104, 1995.

[34] Seymour Geisser. *Predictive Inference.* ADEE, EST, Universidade do Algarve. URL http://sapientia.ualg.pt/bitstream/10400.1/157/1/14_7.pdf.

[35] Robert T. CLEMEN. Combining forecasts: A review and annotated bibliography. *International Journal of Forecasting.*, 5(5):559–583, 19893.

[36] James M. Ragusa William Leigh, Russell Purvis. Forecasting the nyse composite index with technical analysis, pattern recognizer, neural network, and genetic algorithm: a case study in romantic decision support. *Decision Support Systems.*, 5 (32):361– 377, 2002.

[37] Zhang X and Hutchinson J (. Simple architectures on fast machines: practical issues in nonlinear time series prediction. *Time Series Prediction: Forecasting the Future and Understanding the Past.*, 5(52):219–2414, 2001.

[38] Makridakis S. The accuracy of extrapolation (time series) methods: results of a forecasting competition. *J Forecast.*, 5(1):111–153, 1982.

[39] A. Krogh A and J. Vedelsby. Neural network ensembles, cross validation, and active learning. *In Advances in Neural Information Processing Systems*, 7(38):231–238, 1995.

[40] Y. Yang. Combining forecasting procedures: Some theoretical results. *Econometric Theory.*, 5(20):176–222, 1984.

[41] RON KOHAVI ERIC BAUER. An empirical comparison of voting classication algorithms: Bagging, boosting, and variants. *Machine Learning.*, 5(38):1–38, 1998.

[42] GP Zhang. Time series forecasting with neural network ensembles: an application for exchange rate prediction. *Journal of the Operational Research Society.*, 5(52): 652–664, 2001.

[43] Zhi-Hua Zhou Jian-Xin Wu Yuan Jiang Shi-Fu Chen. Genetic algorithm based selective neural network ensemble. *Proceedings of the 17th International Joint Conference on Artificial Intelligencen*, 2(38):797–802, 2001.

[44] L Breiman. The heuristics of instability in model selection. *in press, Annals of Statistics.*, 5(38):1–38, 1996.

[45] C.W.J. Granger and R Ramanathan. Improved methods of forecasting. *Journal of Forecasting.*, 5(3):197–204, 1984.

[46] Dianhui Wan Monther Alhamdoosh. Fast decorrelated neural network ensembles with random weights. *Information Sciences*, 264(38):104–117, 2013.

[47] X. Yao Y. Liu. Ensemble learning via negative correlation. *Elsevier, Neural Networks*, 12(10):1399–1404, 1999.

[48] Eduardo Vazquez-Santacruz and Debrup Chakraborty. An ensemble of degraded neural networks. *Pattern Recognition Lecture Notes in Computer Science Volume*, 6718(38):278–287, 2011.

[49] C. R. S. Vieira S. Y. Sato, W. C. A. Pereira. Phantom para medição da faixa dinâmica de equipamentos de ultra-som biomédicos. 19.(3):156–166, Dezembro 2003.

[50] M. Graça Ruano C. A. Teixeira. Plataforma de aquisição de dados para a estimação não invasiva de temperatura. Portugal national patent nº 103530, 2010(3):156–166, Dezembro 2003.

[51] J. W. Trobaugh R. M. Arthur, W. L. Straube and E. G. Moros. Noninvasive temperature estimation of hyperthermia temperatures with ultrasound. *Int. J. Hyperthermia*, 21(6):589–600, 2005.

[52] A. Hartov P. M. Meaney, K. D. Paulsen and R. K. Crane. Microwave imaging for tissue assessment: initial evaluation in multitarget tissue-equivalent phantoms. *IEEE Trans. Biomed. Eng.*, 43(9):878–890, 1996.

[53] T. Fjield M. Buchanan D. Daum V. Colucci P. Lopath K. Hynynen, A. Chung and F. Jolesz. Feasibility of using ultrasound phased arrays for mri monitored noninvasive surgery. *IEEE Trans. Ultrason., Ferroelect., Freq. Contr.*, 43(6):878–890, 1996.

[54] P. VanBaren C. Simon and E. S. Ebbini. Two-dimensional temperature estimation using diagnostic ultrasound. *IEEE Trans. Ultrason., Ferroelect., Freq. Contr.*, 45 (4):1088–1099, 1998.

[55] R. Seip and E. S. Ebbini. Noninvasive estimation of tissue temperature response to heating fields using diagnostic ultrasound. *IEEE Trans. Biomed. Eng.*, 42(8): 828–839, 1995.

[56] H. Fukukita S. Ueno, M. Hashimoto and T. Yano. Ultrasound thermometry in hyperthermia. *Proc IEEE Ultrasonics Symposium,*, 3(8):1645–1652, 1990.

[57] J. D. Starman R. M. Arthur, W. L. Straube and E. G. Moros. Noninvasive temperature estimation based on the energy of backscattered ultrasound. *Med. Phys.*, 30(PART 6):1021–1109, 2003.

[58] Smith Eiben. *Introduction to Evolutionary Computing.* Number ISBN 0-13-134453-6 in Springer. Prentice Hall, natural computing series edition, 2003. p.71-85.