POLITECNICO DI TORINO

Department of Mechanical and Aerospace Engineering

Master's Degree in Biomedical Engineering

March 2022

Minimally-invasive Temperature Monitoring in Hyperthermia Treatment of Internal Tumors via Detailed Numerical Phantoms

Advisor
Prof. Giuseppe Vecchi
Co-Advisors
Dr. Rossella Gaffoglio
Dr. Giorgio Giordanengo

Candidate
Mariachiara Sinesi, s279879

Dedicated to my Family

Abstract

The aim of thermal therapies in cancer treatment is to significantly alter the temperature of the target (tumor) region without damaging the healthy tissues. Microwave hyperthermia (HT) consists in heating locally the region of interest up to 42-44°C by means of an antenna applicator, while keeping the heat in the surrounding area at a tolerable level for the tissues and therefore for the patient himself. Hyperthermia is used in association with conventional cancer therapies, being able to sensitize tumors to radiotherapy and chemotherapy (cancer therapies are more effective at the same dose) without adding toxicity.

In order to improve the clinical outcome of hyperthermia treatments, a patient-specific treatment planning is needed, where a numerical model of the patient is created, the baseline thermal and dielectric parameters (i.e., the values found in the Literature) are assigned to the different tissues, the antenna feedings are optimized using an EM simulation solver, and the corresponding temperature map is obtained by solving the bio-heat equation. Unfortunately, tissues parameters are characterized by great uncertainty, which can significantly affect the simulated temperature maps. Hence, for safety reasons, invasive temperature probes inserted into closed-tip catheters are necessary during treatment to provide some (limited) temperature measurements.

The present Master Thesis is part of a research aim to monitor and predict the temperature in, and around a cancer volume during HT in a minimally invasive method, obtaining a 3D temperature map from a single catheter (fiber-optic) thermometer. The specific focus is in simulating the operation in silico, with a highly realistic phantom of the neck region; simulations were performed with the highly specialized commercial software Sim4Life, and with temperature-reconstruction algorithms in MATLAB.

For the treatment of internal tumors, the hyperthermia applicator is usually composed by an array of antennas, properly fed to focus the specific absorption rate (SAR) – and hence the temperature increase – on the tumor region. An accurate model of the device has been built for a human phantom in Sim4Life. Then, the microwave heating was focused within the tumor region, through a SAR-based optimization of the antenna feedings. The influence of thermal parameters such as perfusion rate and thermal conductivity, as well as the dielectric ones, i.e., permittivity and electrical conductivity, on the resulting temperature maps is analysed by solving the bioheat equation in Sim4Life for different values of the constituent parameters, changed by means of a Python interface. Once found the most critical parameters, a numerical reconstruction is implemented to obtain a more reliable temperature map of the whole region of interest starting from a set of few known temperature values – mimicking those experimentally known along the catheters during the clinical treatment.

Acknowledgement

I would like to thank Prof. Giuseppe Vecchi, Dr.ssa Rossella Gaffoglio and Dr. Giorgio Giorganengo. In particular Rossella, despite the difficulties and the need to work remotely, given the exceptional period, was always kind and helpful in supporting me during the thesis work.

Thanks to my father Sabino (*Il Serpente dagli occhiali*) who followed me in all my studies helping me with his infinite knowledge, to my mother Michela(*La Vipera*) who took care of me even when I didn't deserve it, to my sister Alice Gaia (*La Cervicapra*) for the best laughs and night-time conversations, and all my relatives who supported my choices every day sharing my dreams.

Thanks to Luca (*Patatino 1*) who put up with me and my craziness every day, during the good times but especially during the bad times. Thanks also for being close to me and supporting me: between shit roommates, pandemic and war this long journey, which seemed to never end, was beautiful and fun especially because you were with me.

Thanks to all my colleagues and friends, in particular Giulia companion of a thousand adventures, to Pietro and Elena for the beautiful shared experiences, to Michele, Dario and Gabriele who has been several times an inspiration to me. I hope we will remain friends even in the future.

Finally, thanks to all the people I met along my way, even for just a minute, because they helped make me who I am.

Contents

List of Figures

List of Tables

Acronyms

API	Application Programming Interface
CAD	Computer-Aided Design
CT	Computer Tomography
EM	Electromagnetic
FDTD	Finite-Difference Time-Domain
H&N	Head and Neck
HT	Hyperthermia Treatment
HTP	Hyperthermia Treatment Planning
HTQ	Hotspot-to-Tumor-Quotient
MRI	Magnetic Resonance Imaging
MW	Microwave
RT	Radiotherapy
SAR	Specific Absorption Rate
SAT	Subcutaneous Adipose Tissue
S4L	Sim4Life
SVD	Singular Value Decomposition
ViP	Virtual Population

Introduction

Thermal therapy purpose is to improve cancer treatments by means of an alteration of temperature in the target tissue. One of the most used thermal treatments is microwave hyperthermia, that locally produces an increase of the tumor temperature up to 42-44°C by the exposure to non-ionizing electromagnetic radiation (radiofrequency or microwaves) [1–3].

Nowadays Hyperthermia Treatments (HT) have shown good clinical results in combination with other cancer treatments as radiotherapy or chemotherapy [3–5]: in this context, hyperthermia acts as a potent enhancer ensuring the same clinical outcome with a lower dose of ionizing radiations and/or drugs.

Microwave (MW) hyperthermia is one of the most promising adjuvant thermal therapies and it is also currently employed in the clinic for deep cancer treatment. The temperature increase on the tumor target is achieved by exploiting constructive wave interference from a certain number of antennas arranged in a circular array [6, 7]. Since overheating of the patient's skin is inevitable, to avoid this, a proper cooling system - the water-bolus - is inserted between the antenna applicator and the patient's body. The water-bolus, a doughnut-shape bag fill with water, performs not only this function, but also enhances the efficiency of the energy coupling into the patient [8].

Currently, patients with advanced carcinomas in the head and neck region (H&N) have a dismal prognosis due to the delicate anatomical sites where they are located: tumors grow adjacent to vulnerable structures such as the spinal cord, nerves and lymph nodes. These kinds of tumor are treated with surgery, radiotherapy, chemotherapy or a combination of them. Even so, most of patients die due to the inability to resect the whole tumor as well as the high toxicity of the therapies, therefore the need to improve loco-regional treatments. It was clinically demonstrated that HT provides significant benefits without adding toxicity [9, 10].

The possibility to plan and control the transferred heating by means of an hyperthermia treatment planning (HTP) increases the effectiveness of the treatments [6, 11]. The HTP allows patient-specific treatment planning thanks to the reconstruction of a 3D patient model from CT and MRI scans; this model is then imported in proper simulation software - together with the model of the antenna applicator - where optimization routines are employed to find the antenna coefficients that maximize the energy deposition in the tumor target.

Simulations of the temperature map in the patient cannot be considered as sufficient per se, being affected by the uncertainty introduced by tissue parameters (both dielectric and thermal). Hence, during treatment, it is necessary to directly control the temperature in some critical points using fiber-optics thermometers, inserted into closed-tip catheters.

However the catheter must be carefully inserted in the proximity of these sensible structures by avoiding undesirable injuries [12]. Hence, this procedure creates great discomfort to the patient and only provides temperature measurements along the direction of the catheter. For this reason, methods that allow to obtain more reliable temperature simulations of the patient during treatment with a minimum number of catheters are urgently needed.

More recently, research has been directed towards the possibility to implement non-invasive thermometry via magnetic resonance imaging (MRI); this method, however, requires the therapy to take place inside an MRI scanner, which would imply high costs preventing a widespread use in the clinic [13].

The aim of hyperthermia treatments is to increase temperature in the tumor target, so optimizing the temperature distribution would seem the recommended approach; however, there is not a systematic study assessing the relation between the predicted temperature and the clinical outcome [6, 14]. On the other hand, a study by Lee et al. [15] demonstrated that a relation exists between the Specific Absorption Rate (SAR) and clinical outcome. The advantages of using SAR to optimize HT are the direct connection with Maxwell's equations, the ability to control the performance both computationally and metrologically, the shorter computational time required with respect to temperature optimization [6, 14]. The deviation between temperature and SAR distributions are to be attributed to the thermal boundary conditions, such as external cooling and the air flow in the respiratory tracts. However, these shifts can be mitigated by additional temperature-based refinement add-on methods [6].

Numerous studies have been carried out to design a device that is site-specific and enables a good control of the power absorption pattern [16]. Based on these studies the HYPERcollar applicator was developed; this device enables to apply HT to tumor located in the H&N region both deeply and laterally [17]. In order to investigate the energy deposition efficiency at the centre of the tumor, numerical simulations were performed as function of antennas position, number of antennas and operating frequencies [18].

A key role, to allow an accurate and safely procedure for the patient during treatment, is the temperature monitoring: in HT is paramount to ensure damage to the tumor portion plus a reasonable safety margin while preserving healthy surrounding anatomical structures.

The goal of the present work is to implement a numerical procedure that allows to mitigate the errors introduced by the uncertainty that affects the tissues parameters, in order to increase the reliability of the simulated temperature maps. This will be achieved by means of algorithms and simulations performed with Sim4Life, the Python interface provided by Sim4Life and MATLAB.

First, an antenna applicator similar to the HYPERcollar, composed of a circular array of 8 antennas, aimed at focusing the electromagnetic radiation on a tumor target placed inside the neck of the realistic human phantom Yoon-sun, was studied and analysed in Sim4Life.

Then the optimization of the SAR using Sim4Life, the Python interface and MATLAB was performed in order to find the antenna feedings providing a maximum SAR focusing on the tumor target, with minimum overheating in the surrounding healthy tissues. The bioheat equation was solved in Sim4Life to find the temperature map corresponding to the optimized SAR profile.

More attention was paid to choose the right parameters and tissues to use during the analysis. Indeed, the human body is a very complex system, in term of elements that compose it and the interaction between them, so different event may occur that modify the results. The application of supra-physiological temperatures, like it happens during hyperthermia, causes changes at the molecular, cellular, and structural level, with corresponding changes in tissue function and in thermal, mechanical and dielectric tissue properties. Therefore, analyses of the temperature maps variation, when the thermal and dielectric tissue parameters are changed according to the ranges found in the Literature, and of the parameters, to determine which affected the most the temperature maps, were carried out. The different values used for the considered parameters are organized in two grids (multigrids): the multigrid A takes in input 4 fixed point and a defined number of quasi-random point (Sobol sequence) and the multigrid B considers only quasi-random point.

Then, a numerical reconstruction was implemented in MATLAB, aimed at improving the reliability of the temperature map starting from a set of few known temperature values. This procedure is based on the construction of a basis of temperature simulations corresponding to different (random) choices of the constituent parameters. In order to simulate a realistic temperature sampling, a fictitious (numerical) catheter was designed and used to extract, from the temperature maps obtained before, single temperature values along a chosen axis. This procedure, employed for both the multigrids, was repeated several times by changing some parameters such as number of points, direction and axis chosen for the insertion of the catheter to analyse several conditions. Furthermore, it was verified that the catheter's insertion did not injure sensitive structures such as blood vase, nerve or lymph nodes.

Finally, a Singular Value Decomposition (SVD) was performed to evaluate the information obtained from the 2 multigrids: in particular, a comparison was performed to determine the number of parameters' combinations needed to gain the same information from both of them.

Overview of the thesis

The thesis contents are organized in 5 chapters and 2 appendices.

Chapter 1 reports an introduction to thermal therapy history and types, the hyperthermia treatments definition, characteristics and state of the art, an in-depth study on focus of the microwave hyperthermia, a short description of the Hyperthermia Treatment Planning (HTP) and an overview of the mathematical equations used (Maxwell and Pennes).

Chapter 2 describes the model implemented in the computational software Sim4Life, the circular array of antennas in combination with the human phantom, the Specific Absorption Rate (SAR) and Hotspot-to-Tumor Quotient (HTQ) optimization. In detail, this chapter reports a short introduction of the software Sim4Life used as electromagnetic and thermal solver, an overview of the phantom and array of antennas employed, an explanation of how was obtained the SAR optimization and the correspondence between SAR and thermal maps.

In Chapter 3, the Python scripts written to automate the data extraction from Sim4Life are presented. In particular, this chapter reports the Python scripts implemented to vary the thermal and dielectric properties of some tissues in the human phantom and automatically generate the corresponding temperature maps. This allows to analyse the influence of the constituent parameters on the temperature map, in order to determine which are the most relevant.

In Chapter 4, the temperature maps corresponding to the variation of some of the most relevant tissue parameters are used as a basis to improve the temperature reconstruction of some target fields (i.e., temperature maps corresponding to a random choice of the tissue parameters), starting from few known values. These values are extracted along different directions, reproducing the direction of insertion of the catheters in the clinical practice. The information gained has been evaluated by means of Singular Value Deposition (SVD) analysis.

In Chapter 5, comments on the results, critical aspects and possible future developments are presented. Finally, the Appendices report the MATLAB and the Python scripts implemented.

Chapter 1

Head and Neck Cancer Therapy

Cancer is a disease caused when cells divide uncontrollably and spread into surrounding tissues. There are many different approaches for treating cancer, depending on the type of cancer, how advanced it is, what types of treatment are available, and what are the expected goals of the treatment. One of the most delicate cases is represented by the advanced carcinomas in the Head and Neck (H&N) region due to due to their sensitive anatomical position. Therefore, the interest in developing and improving loco-regional treatments has grown. For the treatment of internal tumors, the combination of traditional treatments, such as radiotherapy and chemotherapy, with hyperthermia treatments has been demonstrated to be effective in terms of dose (same outcome at a lower dose) as well as no adding further toxicity [4].

1.1 Thermal Therapies

The aim of thermal therapy is to maximize the efficacy of treatments, without damaging the heathy tissues.

Thermal treatments have been used since ancient times for therapeutic purposes such as Roman term, febrile therapies in XIX and XX or more recently the development of radiobiology laboratories [3]. Then experiments on cellular culture in vitro and in vivo demonstrate that an increase in temperature, above 41-42°C, produce inhibition of the normal cell function up to the eventual cell death. So, the efficiency of thermal treatments was associated with the cellular death in terms of thermal dosimetry. It was also observed that the time of treatments for patients was shortened with even higher temperatures.

Nowadays exists several types of thermal treatments that can be classified according to the range of temperatures reached and the duration of treatment [3]:

- *Diathermy*, temperature increase up to 41°C for 5-20 minutes in several session; it is mostly used for physiotherapy and rheumatic pathologies treatments.
- *Hyperthermia*, temperature increase between 42-44°C for almost 60 minutes; is applied in combination with radiotherapy and chemotherapy for cancer treatments.

- *Thermal Ablation*, temperature increase above 45°C for 5-15 minutes; allows an accurate and localized ablation. It is usually employed to destroy cell within a defined tumor region.

The working temperatures and therapy duration reported above do not represent clear-cut divisions between thermal therapies but provide an idea of the parameters used to achieve a certain biological effect.

1.2 Hyperthermia

In the Nineties technological development of non-ionizing radiation helped the research into thermal therapy and the growth of its clinical application [3]. It has been demonstrated that hyperthermia, caused by the exposure to non-ionizing electromagnetic radiation (like radiofrequency or microwaves), sensitize tumours to radiotherapy and chemotherapy (cancer therapies are more effective at the same dose), without adding toxicity. The comparison in terms of clinical outcome between oncology patients treated with radiotherapy alone vs. thermo-radiotherapy (radiotherapy plus hyperthermia treatment) has demonstrated how hyperthermia provides significant benefits for various tumour sites, such as breast, cervix, head and neck, rectum, urinary bladder, esophagus, cutaneous melanoma and choroidal melanomas [4, 5].

1.2.1 State of the art

Currently, electromagnetic heating is one of the main techniques employed for hyperthermia treatments (HT) thanks to its non-invasive nature that leads to a safe procedure for patients. The Microwave (MW) frequency range between $10^8 - 10^{10}$ Hz is characterised by a non-ionizing radiation that avoids health hazard. In addition, MW heating is more instantaneous and uniform than other processes.

HT can be split out in superficial hyperthermia, where the heat is limited to a volume of tissue close to the applicator, and deep hyperthermia, which aims at concentrating the heating in the tumor area. In general, higher MW frequencies provide localized heating of skin and superficial tissues, while the lower frequencies are more frequently used to heat larger and deeper regions in the body. Furthermore, power deposition decreases exponentially as microwaves penetrate through the tissue [7]. An effective HT aims at increasing the temperature within the tumor and this goal is often better achieved with deep hyperthermia. In the latter case the heating approach can be performed in two ways [7]:

- *External heating*: a circular array of antennas is placed around the patient to create a constructive wave interference and a definite electromagnetic (EM) field to heat the target region. The critical parameters, that need to be optimized, in a phased-array system are the amplitudes, the phases and the frequencies of the antennas [9, 25].

- *Internal heating*: intraluminal or intracavitary applicators are employed to heat areas limited to 5-10 mm radius. The needle-shaped antennas are implanted in the tissue or inserted in catheters.

In the years, several devices have been developed following the above heating approaches. For the H&N region, Paulides et al. [22] have developed the HYPERcollar, a novel applicator for hyperthermia treatment that enables a good control of the power absorption pattern as well as the possibility to apply HT deeply and laterally. This device is made up of two circular arrays of 6 antennas operating at 434 MHz and the water-bolus, a doughnut-shaped bag filled with circulating demineralized water at constant temperature of about 20-25°C [9] (Fig.1.1). The latter is located between the patient and the applicator having the dual function to couple the electromagnetic field into the patient and to control the skin surface temperature, avoiding superficial hotspots.

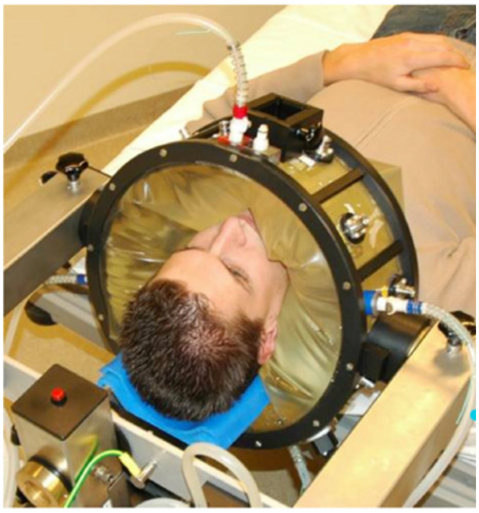


Figure 1.1: HYPERcollar for H&N tumor treatment [24]

In order to increase temperature in the tumor region, the feedings (phases and amplitudes) of the antennas within the applicator have to be properly optimized. This can be performed by optimizing the temperature distribution or the Specific Absorption Rate (SAR). The first method seems the recommended approach, however there is not a systematic study assessing the relation between the predicted temperature and the clinical outcome [6, 14]. On the other hand, Lee et al. demonstrated that the second method can be related with the clinical outcome and present several advantages: the direct connection with Maxwell equations, the ability to control the performance both computationally and metrologically, the shorter computational time and the faster optimization [15]. However, SAR-based optimization approach, considered in this thesis, is affected by the thermal boundary conditions: external cooling and the air flow in the respiratory tracts can shifts the maximum in temperature maps, but this can be mitigated by add-on methods [6]. Both SAR-based and temperature-based methods are affected by the uncertainty characterizing thermal tissue parameters, i.e., perfusion rate and thermal conductivity.

1.2.2 Hyperthermia Treatment Planning (HTP)

During hyperthermia treatments, the heat generated induces a biological damage due to the inability of the tissue to dissipate the excess of energy at the same time it is supplied. Therefore, physiological changes are involved, in particular the blood perfusion, the vascular permeability and the metabolic activity are modified [3]. Among them, the most important is the blood perfusion due to its role in the abduction and dissipation of heat within the tissues. Temperature-dependent perfusion is very heterogeneous between the tissues and is also related to age, gender, lifestyle and diseases. Indeed, vessels in tumor are leakier compared to healthy tissues and that leads to higher blood flow, but on the other hand, they present a more chaotic structure that causes less nutrient diffusion and oxygen concentration. For these reasons, tumors are usually more sensitive to hyperthermia treatments even at lower temperatures [3].

The excessive temperature increase in healthy thermo-sensitive tissues surrounding the target area could cause serious damage to the patient. For this reason it is very important to implement a sort of real time monitoring of the temperature during the whole treatment. The hyperthermia treatment planning (HTP) is needed to guarantee the safety of the patient as well as to achieve a patient-dedicated treatment [4, 8, 10] (Fig.1.2). Firstly, a radiotherapy computerised tomography (RT CT) scan or MRI images of the patient, in treatment position, are made using immobilisation devices such as mask and headrest. Next, a full 3D patient phantom is created by means of a manual or automatic segmentation of the different tissue types. Then, to represent the real HT set-up, the patient-specific 3D model is positioned into the applicator model. The tissue specific dielectric and thermal parameters are assigned to each tissue to feature a realistic behaviour,while the feedings (amplitudes and phases) of the antenna applicator are properly selected by means of optimization techniques to focus the heat in the tumor target volume. Finally, the complex electric field distribution is computed by numerically solving Maxwell's equations (e.g., using the finite-difference time-domain (FDTD) scheme) in the region of interest, in order to predict the specific absorption rate (SAR) distribution.

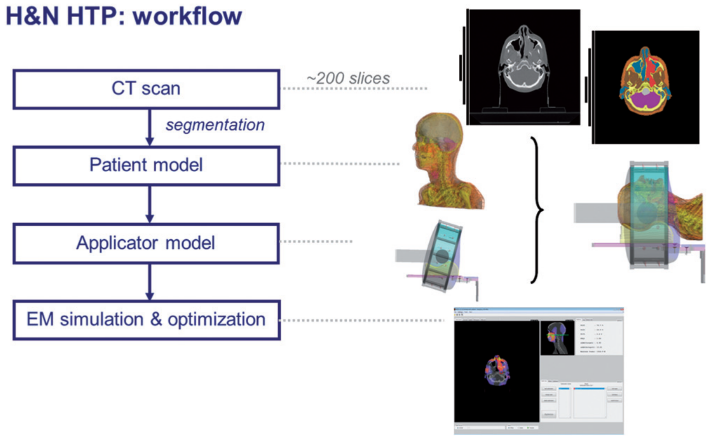


Figure 1.2: H&N HTP workflow [8]

1.3 Mathematical Model

Mathematical modelling is important to plan, predict and optimize treatments efficiently. Indeed, it helps to determine the correct dose and transmitted energy, to analyse the physiological parameters more easily, to monitor and predict the clinical results.

Regardless of the method used to achieve tissue heating, it is necessary to solve the bio-heat equation or Pennes equation:

$$\rho C \frac{\partial T}{\partial t} = k\nabla^2 T + q_s \tag{1.1}$$

where ρ is the tissue mass density [kg m^{-3}], C is the heat capacity [J kg^{-1} K^{-1}], k is the thermal conductivity [W m^{-1} K^{-1}], and q_s [W m^{-3}], is the source term. The latter can be written as $q_s = q_{hs} + q_m + q_p$, where [3]:

- q_{hs} is the heat source term;
- q_m is the metabolic heat generation term (heat generated by metabolic reactions);
- q_p is the heat loss due to blood perfusion.

In this case, the electromagnetic heating is considered as heat source term:

$$q_{hs} = \frac{\sigma}{2}|\mathbf{E}_0|^2 \tag{1.2}$$

where σ is the electrical conductivity in [S m^{-1}] and $|\mathbf{E}_0|^2$ is the electric field intensity [V m^{-1}]; while the metabolic heat generation term can be assumed negligible ($q_m \approx 0$) to simplify the problem. Considering that the heat source does not change with time, (1.1) can be rewritten as follows:

$$\rho C \frac{\partial T}{\partial t} = k\nabla^2 T + q_{hs} + q_p \tag{1.3}$$

This is a simple equation that enables to mimics the biological system's behaviour, but it also has limitations: first, it considers all the heat released from arteries as absorbed locally by the tissue, not taking into account the vein heat; then, it rates the arteries temperature equal to the body temperature [3]. All of these factors produce a blood perfusion overestimation, that could be calculated as follows:

$$q_p = -\omega_{bl} C_{bl} \rho_{bl} (T - T_{bl}) \tag{1.4}$$

where ω_{bl} is the blood perfusion rate [ml kg^{-1} K^{-1}], C_{bl} is the specific heat of blood [J kg^{-1} K^{-1}], ρ_{bl} is the mass density of blood and T_{bl} is the arterial blood temperature [K] [3]. The minus sign in (1.4) represents what the blood does, i.e., it compensates temperature variations: if the temperature increases, it takes away the heat, if the temperature decreases, it carries the heat in.

In order to determine the heat source term q_{hs} in the bio-heat equation 1.3, is mandatory to solve the electromagnetic equations (Maxwell's equations). Microwave energy is converted in heat within tissue due to dielectric losses and this requires modelling of wave propagation since MW wavelengths in tissue are in the cm range [19]. The "Maxwell-Ampere equation" describes how conduction current as well as displacement current affect the magnetic field:

$$\nabla \times \mathbf{H} = \mathbf{J} + \frac{\partial \mathbf{D}}{\partial t} \tag{1.5}$$

where $\mathbf{H}$ denotes the magnetic field strength [A m^{-1}], $\mathbf{J}$ the current density [A m^{-2}], $\mathbf{D}$ the electric flux density [A s m^{-2}], and $\frac{\partial \mathbf{D}}{\partial t}$ the displacement current density [A m^{-2}]. The "Maxwell-Faraday equation" states that changes in the magnetic field $\mathbf{B}$ [V s m^{-2}] influence the electric field strength $\mathbf{E}$ [V m^{-1}]

$$\nabla \times \mathbf{E} = -\frac{\partial \mathbf{B}}{\partial t} \tag{1.6}$$

The Gauss's laws for the electric field and the magnetic field are describing the facts that the magnetic field $\mathbf{B}$ is without any source and that the electrical charge density ρ [A s m^{-3}] is the source for the electric displacement:

$$\nabla \cdot \mathbf{D} = \rho \tag{1.7}$$

$$\nabla \cdot \mathbf{B} = 0 \tag{1.8}$$

The electromagnetic field carries a certain amount of energy that is perceptible only as it interacts with matter (this interaction depends on the ratio between the dimension of the object and the wavelength).

The energy balance in a defined volume (Ω) in which the electromagnetic field propagates can be calculate as follows:

$$p_{diss,\Omega}(t) + \frac{d}{dt} W_\Omega(t) = p_{\partial\Omega}(t) \tag{1.9}$$

where $p_{diss,\Omega}$ [W] is the power dissipated in the volume, W_Ω is the stored energy in [Ω] and $p_{\partial\Omega}$ is the energy transport through the surface of [Ω]. The power dissipation term, that corresponds to the heat source, can be written as:

$$\frac{p_{diss(t)}}{d\Omega} = \mathbf{E} \cdot \sigma \mathbf{E} = \sigma |\mathbf{E}(t)|^2 \tag{1.10}$$

or in the harmonic form:

$$\frac{p_{diss(t)}}{d\Omega} = \frac{\sigma}{2} |\mathbf{E}|^2 \tag{1.11}$$

where $|\mathbf{E}|$ is the electric field's magnitude (peak value) [V m^{-1}].

When (1.11) is normalized by the tissue density (ρ), it is referred to as the Specific Absorption Rate (SAR) [W kg^{-1}]:

$$SAR = \frac{\sigma}{2\rho} |\mathbf{E}|^2 \tag{1.12}$$

The specific absorption rate can can be defined as the power absorbed per unit of mass of a tissue in the interaction with the electromagnetic wave [3]. Even if SAR does not relate directly with temperature, it is demonstrated that there is a relation between SAR optimization and clinical outcome [14].

For modelling of MW hyperthermia, it is necessary to know the relations between the vectors **H**, **D**, **B**, **J**, and **E**, and the tissue properties, such as permittivity ϵ $[\mathrm{A\,s\,V^{-1}\,m^{-1}}]$ and the electrical conductivity σ $[\mathrm{A\,V^{-1}\,m^{-1}}]$.

1.4 Parameters Dependence

In order to achieve a more realistic model, it is important to consider the temperature variation due to the physiological thermoregulation depending on electrical and thermal tissue properties. The knowledge of temperature-dependent electrical properties of biological tissues is significant to calculate the deposition of electromagnetic energy. Therefore these properties strongly influence the resulting temperature and SAR maps.

The dielectric permittivity ϵ $[\mathrm{F\,m^{-1}}]$ and the conductivity σ $[\mathrm{S\,m^{-1}}]$ are expressed as a function of the electric field E according to the following equations:

$$\epsilon = \frac{D}{E} \tag{1.13}$$

$$\sigma = \frac{J}{E} \tag{1.14}$$

The thermal conductivity k $[\mathrm{W\,m^{-1}\,K^{-1}}]$ describes how well the tissue conducts heat and is important to estimate the biological heat transport correctly.

$$k = \frac{Q}{\nabla^2 T} \tag{1.15}$$

Another parameter necessary to generate valid results is the perfusion ω $[\mathrm{ml\,kg^{-1}\,min^{-1}}]$. Indeed, blood flow strongly affects temperature distribution due to both the vascular system of the body and the vasculature within the tumor. The distribution of tumor's vasculature is quite heterogeneous between different tumors and in the tumor itself, but in general they have reduced blood flow. Therefore, tumors have limited perfusion (compared to normal tissues) and are not able to remove the heat efficiently, which may lead to higher temperatures during heating [19].

Thermal and dielectric tissue parameters of normal and tumor tissues, used as input for the electromagnetic and thermal model, are usually taken from the Literature. However, the reported values vary substantially due to their great uncertainty that depends both on the reliability and availability of the Literature values and the assumption made beforehand [22, 26]. In particular, the these parameters values found in the Literature derive from *ex-vivo* measurements, while it would be better to know their value from *in-vivo* measurements in order to achieve more realistic results. In Chapter 4 is presented an alternative method used to mitigate this lack of information due to the difficulty in deriving the *in-vivo* values [27].

Chapter 2

Numerical Simulation

A numerical simulation is the starting point to study and understand the behaviour of systems, due to its ability to reproduce the real world. Thanks to the possibility to simplify and split up complex problems, as well as to avoid ethical and economic difficulties, numerical simulations are widely used to predict the system's behaviour, test different conditions, promote comparisons and support decision making.

In order to produce the most accurate simulation it is important to use suitable mathematical equations, create realistic model and assess the most relevant parameters. In this thesis the previous statements were achieved by means of understanding the bio-heat and electromagnetic equations and applying proper numerical methods to solve them, reproducing human phantom's structure, devices geometry and boundary conditions, and choosing the most relevant tissues and related parameters.

2.1 Sim4Life

"*Sim4Life is a revolutionary simulation platform, combining computable human phantoms with the most powerful physics solvers and the most advanced tissue models, for directly analysing biological real-world phenomena and complex technical devices in a validated biological and anatomical environment*" [28].

In other words, this software contains computable human phantoms based on the Virtual Population ViP3.0 models of the IT'IS Foundation at ETH Zurich, an electromagnetic full wave solver (P-EM-FDTD) combined with a biothermal solver that allow to solve the Penn equation (1.3), integrated tissue models that enable the modelling and analysis of physiological processes and an advanced modelling tool set. The computable phantoms are functionalized for prediction of real-world biological and physiological phenomena for any defined patient population. The electromagnetic full solver it is extensively validated and documented according to the IEEE/IEC 62704-1 standard as well as by comparisons with measured data. The modelling tool set allows advanced and interactive CAD modelling and high-quality surface modelling (from segmented image data). Furthermore, Sim4Life provides a very sophisticated mesher, in order to have an high-fidelity discretization, an always update database of all tissues properties and specialized tools to handle specific tasks (upon user request) [28].

For the purpose of this work, Sim4Life was used as electromagnetic (EM-FDTD) and thermal solver. In order to optimize the phase and amplitude of the array of antennas and consequently the SAR, it was created an EM-FDTD simulation, while the thermal solver was then employed to solve the bioheat equation in the realistic phantom, using the optimized SAR map as input source term.

2.2 Patch Antenna

The IEEE Standard Definitions of Terms for Antennas (IEEE Std 145–1983) [29] defines the antenna as "*a means for radiating or receiving radio waves*". The use of patch and dipole antennas within clinical environment is common because they are simple and have a larger bandwidth. Patch (or microstrip) antennas are made up of a thin metallic layer (patch) deposited on a grounded substrate of thickness h_{sub} and relative permittivity ϵ_r. In the considered case a patch with rectangular shape was chosen; the design and the values of the different parameters have been optimized in order to improve the performance, as shown in Fig.2.1 and Tab.2.1.

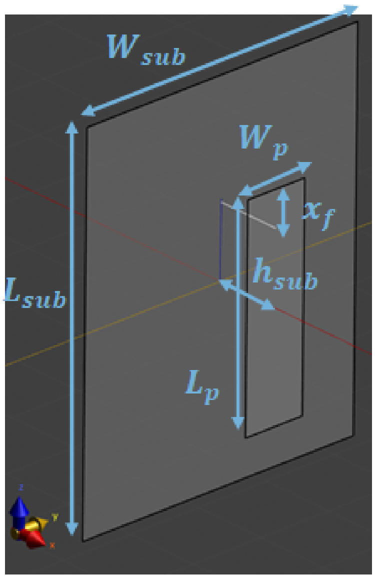


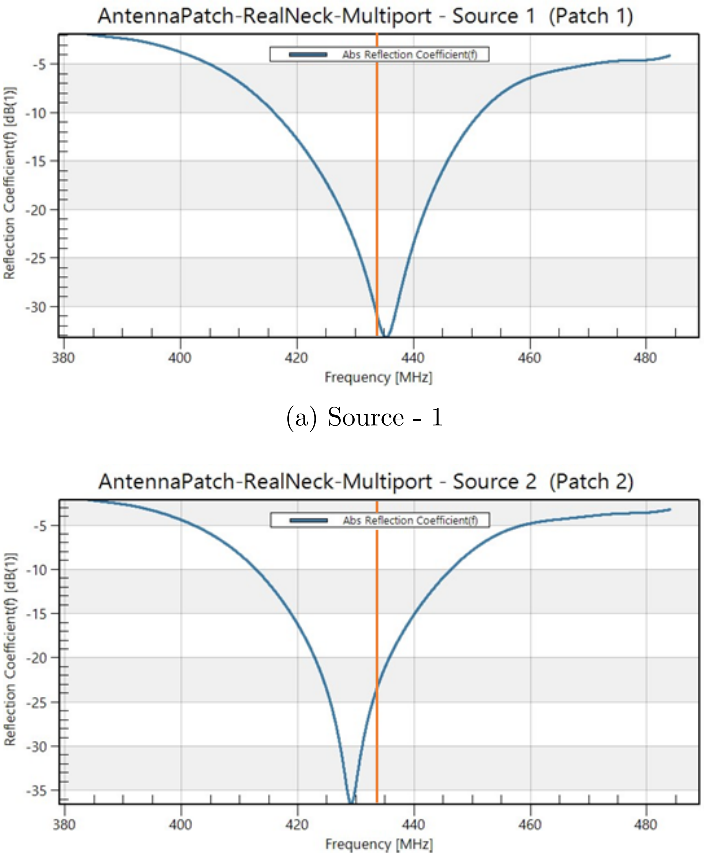
Figure 2.1: Patch antenna design

Table 2.1: Patch antenna dimensions

Parameter	Dimension (mm)
W_{sub}	40
L_{sub}	50
W_p	7.21
L_p	31
x_f	4.96
h_{sub}	8.4

In addition, a circular array of 8 equidistant antennas was built as a device for transmitting energy in the form of electromagnetic waves. This set-up, according to Henke et al. [18], ensures the optimum arrangement to maximize the interference of the transversal waves [25] for a symmetrical geometry as the neck region. The operating frequency was fixed at 434 MHz to avoid hotspots in other tissues as well as to increase the efficiency of the applicator in the target region. Indeed, lower frequencies (100–144 MHz) result in superficial absorption only, whereas higher frequencies (700-915 MHz) produce multiple hotspots [18].

To verify the correctness of the implemented antenna setup, the reflection coefficient S11 of each antenna in the array was analysed. Figures 2.2a and 2.2b report the modulus of the reflection coefficient for the antennas 1 and 2, respectively, showing a minimum (resonance) at the operating frequency (434 MHz) and a sufficiently wide bandwidth around it. This check has been carried out in Sim4Life by suppling a unitary power to a chosen antenna while keeping the others switched off.



(a) Source - 1

(b) Source - 2

Figure 2.2: Modulus of the reflection coefficient (S_{11})

The reflection coefficient represents the power reflected from the input antenna, hence, is equal to the ratio between the amplitude of the reflected signal to that of the incident signal. Therefore, a narrow minimum in dB means that almost all the power is transmitted at a defined frequency.

2.3 Human Phantom

As it was already said, to faithfully replicate the real system a detailed model is needed. For this purpose, a great help is provided by the Virtual Population within Sim4Life. The ViP models are a set of detailed high-resolution anatomical model created from magnetic resonance data of volunteers. These models are useful for biophysical and biomedical modelling and simulation, including medical implant safety assessments, as well as to characterize a specific population.

For this work it was used Yoon-Sun (Fig.2.3), a female ViP model that presents the characteristics reported in Table 2.2 (where BMI is the Body Mass Index):

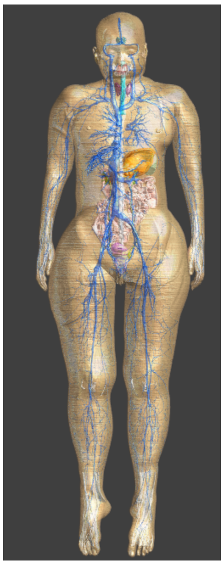


Figure 2.3: Phantom model Yoon-sun

Table 2.2: Phantom characteristics

Age	26	[year]
Height	1.52	[m]
Weight	54.6	[kg]
BMI	23.6	[$\mathrm{kg\,m^{-2}}$]

Then, a tumor model was created inside Yoon-Sun's model, and precisely in the neck region of the phantom, in order to identify the target region, correctly estimate the thermal dose and consequently perform an accurate hyperthermal treatment. In particular, the tumor, modelled using the advance modelling tool offered in Sim4Life, is an ellipsoid with a volume of 4041.44 mm^3 and overall dimensions 18.07 mm, 16.26 mm and 26.25 mm, along the x, y, z axes respectively. Next, the entire volume was divided into a finite number of cubic elements (voxels) used in the numerical calculations performed by Sim4Life.

Finally, the complete set-up was built, including the human phantom, Yoon-Sun, and the HT applicator i.e., the circular array of antennas (Fig.2.4).

Furthermore, using the electromagnetic solver (FDTD) in Sim4Life, it was analysed the E, J and SAR filed distribution generated from the array of antennas: through a Multiport simulation Sim4Life solves the Maxwell's equations by turning on only one antenna at time, with an initial power supply normalized to 1W, and turning off all the others. Then the derived results where combined with the Simulation Combiner which add the effect of

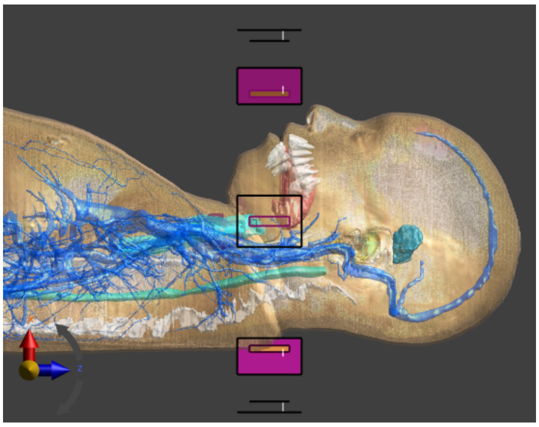
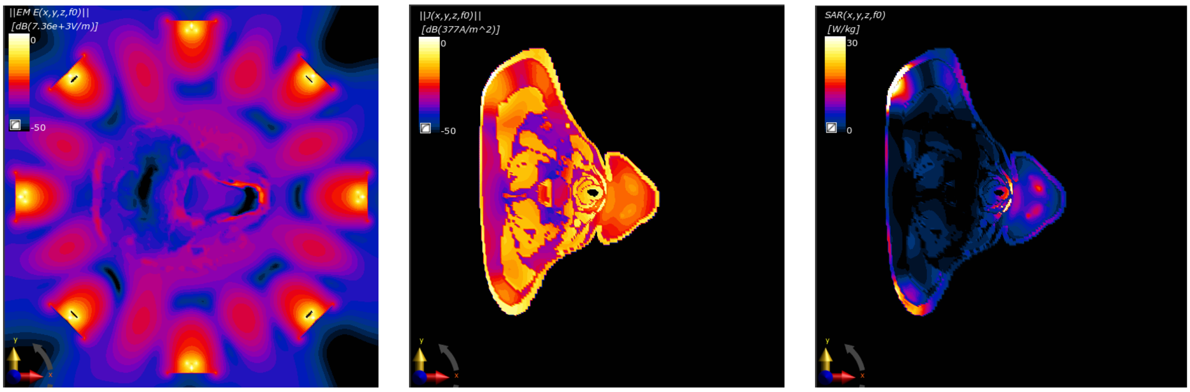


Figure 2.4: Final set-up (treatment position)

each antenna, without phase shifting. Hence the field profile obtained are obviously not focused on the tumor, but at the centre of the array. Below in Figure 2.5 are presented the fields distribution of E, J and SAR on xy plane, considered the more relevant, with a power supply of 20W for a better view of the results.

In order to check the correct reading of the results, the fields data were exported from Sim4Life to MATLAB and displayed.



(a) $|\mathbf{E}|$

(b) $|\mathbf{J}|$

(c) SAR

Figure 2.5: Slice view of $|\mathbf{E}|$, $|\mathbf{J}|$ and SAR fields in the xy plane at 20W.

As can be seen, there is a hotspot near the shoulder that can be related to a numerical peak as well as to a heat concentration. This phenomenon is common due to the proximity between the antennas and the patient skin but thanks to the water-bolus interposition it should not determine overheating problems when the bioheat equation will be solved. Moreover, none of the analysed fields displayed a peak in the target region: optimization is required.

2.4 SAR-based Optimization

The head and neck region is characterized by many tissue transitions, several ducts containing air and large cooling vessels. Therefore, in the clinical set-up, the thermal dose is determined by a huge amount of parameters with separate and combined influences.

As mentioned in section 1.2.1 the SAR-based approach was chosen to optimize the feedings, i.e., phase and amplitude, of the antennas. The SAR can be correlated with the tumor surface area and depth, depending on the hyperthermia applicator characteristic [15].

On the other hand, the influence on the SAR pattern of various parameters, such as inaccuracies in the geometry of the array, uncertainty in the anatomy of the phantom etc., can be seen as distortions on the local SAR pattern with respect to the reference configuration.

The causes of the perturbations on the SAR distribution in the region of interest are clustered in three groups:

- Anatomy related distortion;
- Antenna array related distortion;
- Water-bolus related distortion.

These parameters can be investigated independently allowing a better understanding of their, sometimes opposing, effects.

In this thesis only the first two points, anatomy and array related distortion, will be examined. In particular, will be studied the effect of the uncertainty of the phantom parameters on the temperature maps: this is introduced by the thermal parameters when solving the biothermal equation (1.3) and from the dielectric parameters at the level of the SAR maps obtained by solving the Maxwell equations (Equations (1.5) to (1.8)) and then the biothermal equation by means of using the previous results as source term.

The SAR optimization was carried out by maximizing the electric field concentration in the target region and, as a consequence, the SAR profile and the temperature.

The SAR field defined in section 1.3 (1.12) has the following expression:

$$SAR = \frac{\sigma}{2\rho}|\mathbf{E}|^2 \tag{2.1}$$

where $\mathbf{E}$ is the total electric field generated by the antenna array:

$$\mathbf{E} = \sum_{i=1}^{N} \tilde{\nu_n} \cdot \mathbf{E}_n \tag{2.2}$$

In (2.2) the $\mathbf{E}_n$ is the electric filed generated by the Nth antenna of the array when all the other antennas are off (as described before in section 2.3) and $\tilde{\nu_n}$ are the coefficients that need to be optimized. For the phase optimization $\tilde{\nu_n}$ are expressed as:

$$\tilde{\nu}_n = C \cdot \nu_0 \cdot e^{i\varphi_n} \tag{2.3}$$

where φ_n are the phases of the antennas included in the range [0,2π], $\nu_0 = \sqrt{2R_0P_0}$ with $R_0 = 50\Omega$ and $P_0 = 20W$, while:

$$C = \frac{1}{\sqrt{N}} \tag{2.4}$$

is a coefficient for the power normalization and N is the number of antenna in the array. On the other hand for the optimization of phase and amplitude the $\tilde{\nu_n}$ can be written as follow:

$$\tilde{\nu}_n = C \cdot \nu_0 \cdot \zeta_n \cdot e^{i\varphi_n} \tag{2.5}$$

where ζ_n included in the range [0,1] are the coefficients to vary to optimize the amplitude of each antenna in the array and C is:

$$C = \frac{1}{\sqrt{\sum_n {\zeta_n}^2}} \tag{2.6}$$

Hence to maximize the SAR field, the antenna feedings have to be chosen wisely.

2.4.1 Phase and Phase-Amplitude Optimization

In this thesis, first were optimized only the phases of the antennas (2.3) and then both amplitudes and phases were included in the optimization process (2.5). The optimized feedings are shown in Tab.2.3 and Tab.2.4 for phase-only and phase-amplitude optimization, respectively.

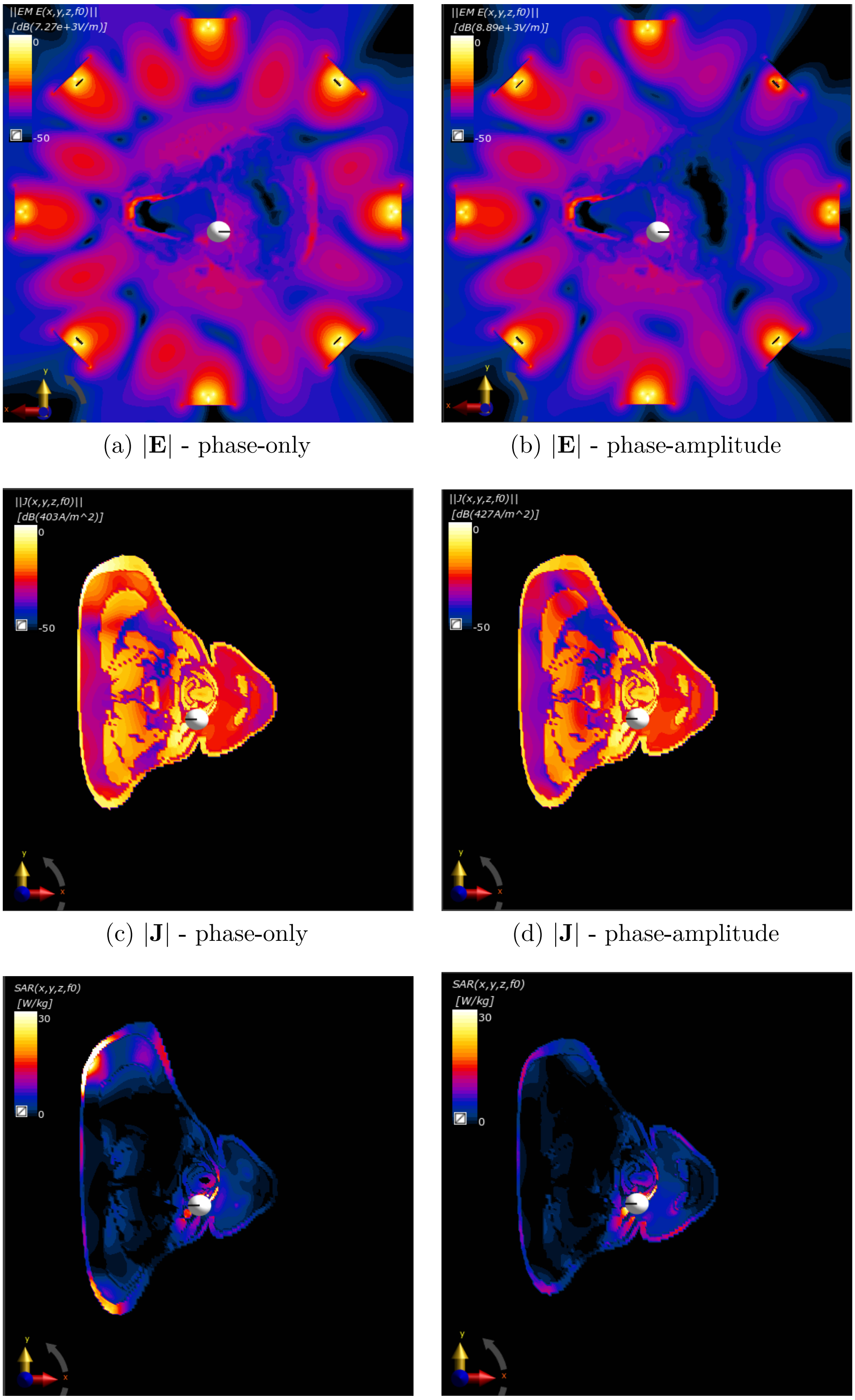
Table 2.3: Phase-only optimization

Antenna	φ
(n.)	[deg]
1	28.37
2	29.55
3	0
4	337.40
5	294.88
6	296.55
7	301.05
8	0

Table 2.4: Phase-amplitude optimization

Antenna	φ	ζ^2
(n.)	[deg]	[W]
1	27.02	0.86
2	32.51	0.74
3	0	0.54
4	0	0.04
5	307.25	0.35
6	293.51	0.41
7	311.57	0.99
8	0	0.75

Then these values, normalized at 20W power, were replaced in the analysis settings within Sim4Life in order to visualize the optimized fields profile.



(a) $|\mathbf{E}|$ - phase-only

(b) $|\mathbf{E}|$ - phase-amplitude

(c) $|\mathbf{J}|$ - phase-only

(d) $|\mathbf{J}|$ - phase-amplitude

(e) SAR - phase-only

(f) SAR - phase-amplitude

Figure 2.6: Comparison of $|\mathbf{E}|$, $|\mathbf{J}|$ and SAR field, in the xy plane, obtained by optimizing the phases only (left column) and both amplitudes and phases (right column). The white element identifies the tumor.

The SAR optimization is successful when there are no internal hotspot, the radiation is focused on the target region and the temperature reaches its peak within the tumor. The comparison between the fields obtained by maximizing only the phases and both amplitudes and phases (Fig.2.6), shows that the optimization performed with phase and amplitude achieves better results. In detail, the hotspot placed on the shoulder can be avoided only with the second optimization and the focusing of the fields in the target region (indicated by the green ellipsoid in the images and representing the tumor) is more evident. In addition, there is an increase in the fields levels on the target region that points out a less dissipation of the supplied power (better use).

Subsequently a Python script will be implemented which, in post processing, extracts the fields relating to the switching on of the different antennas (the so-called stand-alone fields (2.2)), evaluates them in certain regions of the phantom (tumor and healthy tissues) and saves them as .mat file to be processed later in MATLAB.

2.4.2 Hotspot-to-Tumor Quotient (HTQ)

In order to focalize the SAR on the tumor, minimizing the power deposited in the surrounding healthy tissues, the Hotspot to Tumor Quotient (HTQ) is minimized. The latter is defined as follow:

$$HTQ = \frac{< SAR_{V1} >}{< SAR_{target} >} \tag{2.7}$$

where $< SAR_{V1} >$ is the average SAR in the 1% of healthy volume V1 with the highest SAR, while $< SAR_{target} >$ is the average SAR evaluated in the target region.

The HTQ is the objective function of the optimization algorithm implemented in MATLAB in which the phases φ_n of the individual antennas and subsequently also the amplitudes ζ_n are optimized, as described before in Equations (2.3) and (2.5). In particular, in this thesis, the HTQ is the fitness function used in order to highlight which set of parameters, between the power supply of the antennas for the phase-only and the phase-amplitude, allow a better optimization, hence has the lowest value of HTQ.

A fitness function is a particular type of objective function that is used to summarise, as a single figure of merit, how close a given design solution is to achieving the set aims. Fitness functions are used in each optimization method to guide simulations towards optimal design solutions.

In Figure 2.7 are compared the fitness function, calculated in MATLAB, relating to the optimization of only the phases (blue line) and to the optimization of phases and amplitudes (green line) of the antennas. As can be seen there is a sharp decrease in both the curve until both of them reach a plateau, which is clearly lower in the optimization with phase and amplitude (green line).

The greater slope of the curve and the minimum value achieved ensure a correct optimization, hence the HTQ is minimized as a greater specific absorption of the tumor is desired.

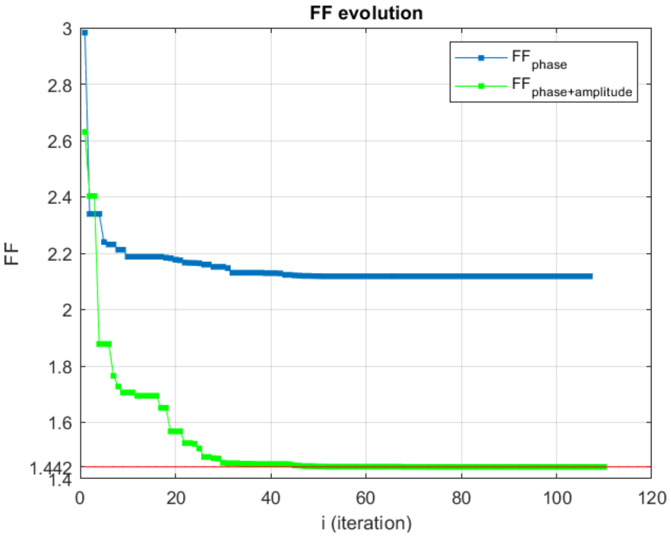


Figure 2.7: Comparison between the fitness functions for the phase-only (blue) and the phase-amplitude (green) optimizations.

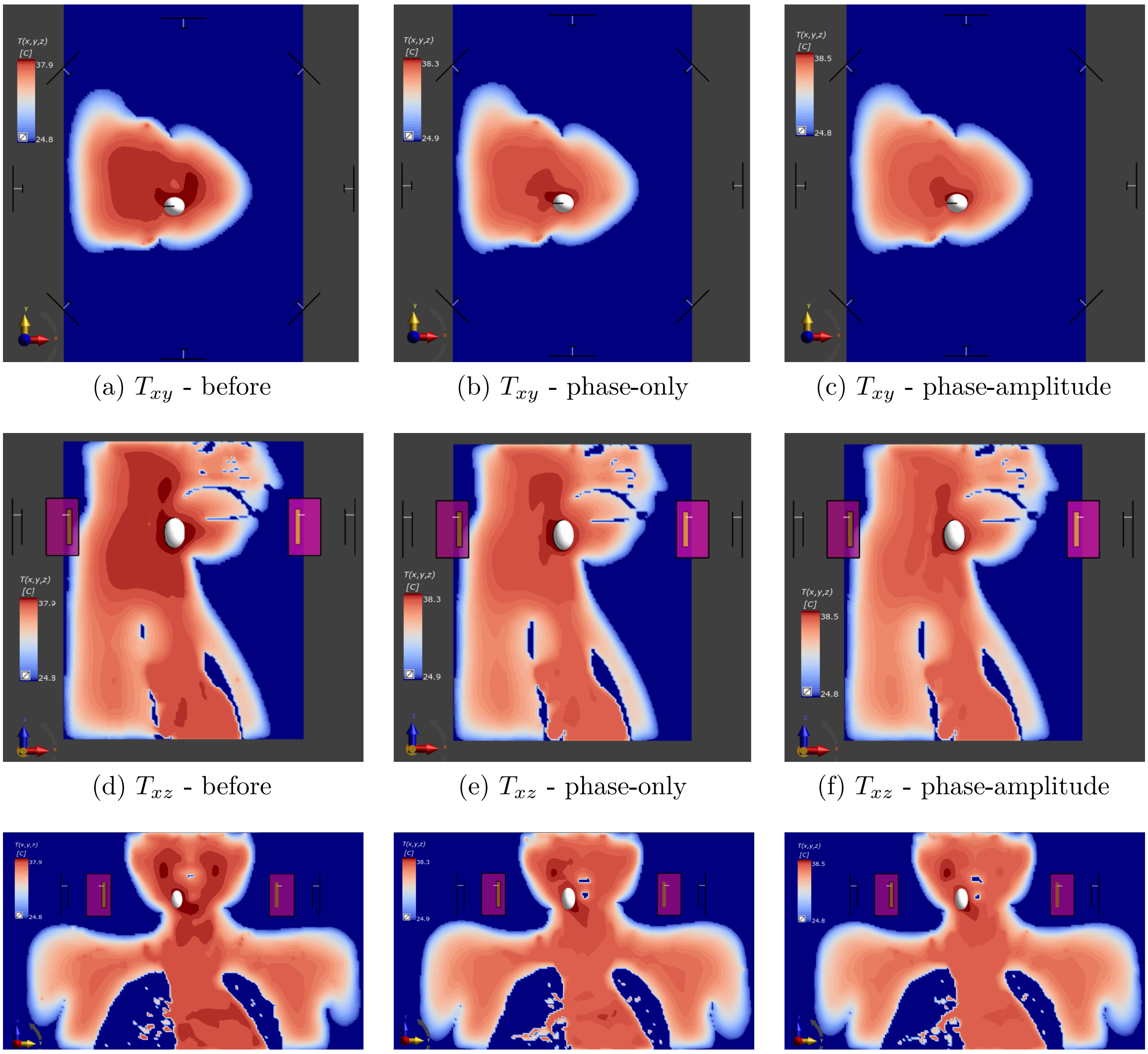
2.5 Temperature Maps

Now, it is necessary to verify that the temperature inside the tumor corresponds to the value required for hyperthermia treatment. For this reason, the temperature maps corresponding to the optimized SAR profiles reported before (with a normalization power for the array of 20W) are extracted and compared. In Fig.2.8 presents the temperature maps obtained *before the optimization* (fig.a), *after the phase-only optimization* (fig.b) and *after the phase and amplitude optimization* (fig.c). Scrolling through the various cases, it is possible to notice a focalization of the temperature field around the tumor, that is necessary to avoid damages to the surrounding healthy tissues and cause less pain to the patient. On the other hand, regarding the temperature value in the centre of the tumor, it is observed that the latter is too low (38.5°C) compared to the value requested for hyperthermia treatment (43-44°C).

In order to achieve the desired temperature within the tumor the absorbed power needs to be increased, due to the relation between temperature and power. After several tests (Fig.2.9) it has been found that an acceptable power supply corresponds to a value of 80W which generates a maximum temperature of 44.6°C in the target area.

To complete the analyses and verify that the new power chosen is adequate, the $|\mathbf{E}|$, $|\mathbf{J}|$ and SAR fields (in the xy plane considered the most relevant) and the temperature maps for the xz and yz planes, shown below (Fig.2.10), have been displayed. All the analyses carried out so far have been reproduced in MATLAB after exporting them from Sim4Life.

The presented images show that the power chosen and the phase-amplitude optimization carried out is correct as there is a focus of heat in the region of the tumor in all planes (xy, xz and yz) and a sufficient temperature (44.6 °C) for heat treatment. The comparison with the values in MATLAB verified that the data exported from Sim4Life can be correctly



(a) T_{xy} - before (b) T_{xy} - phase-only (c) T_{xy} - phase-amplitude

(d) T_{xz} - before (e) T_{xz} - phase-only (f) T_{xz} - phase-amplitude

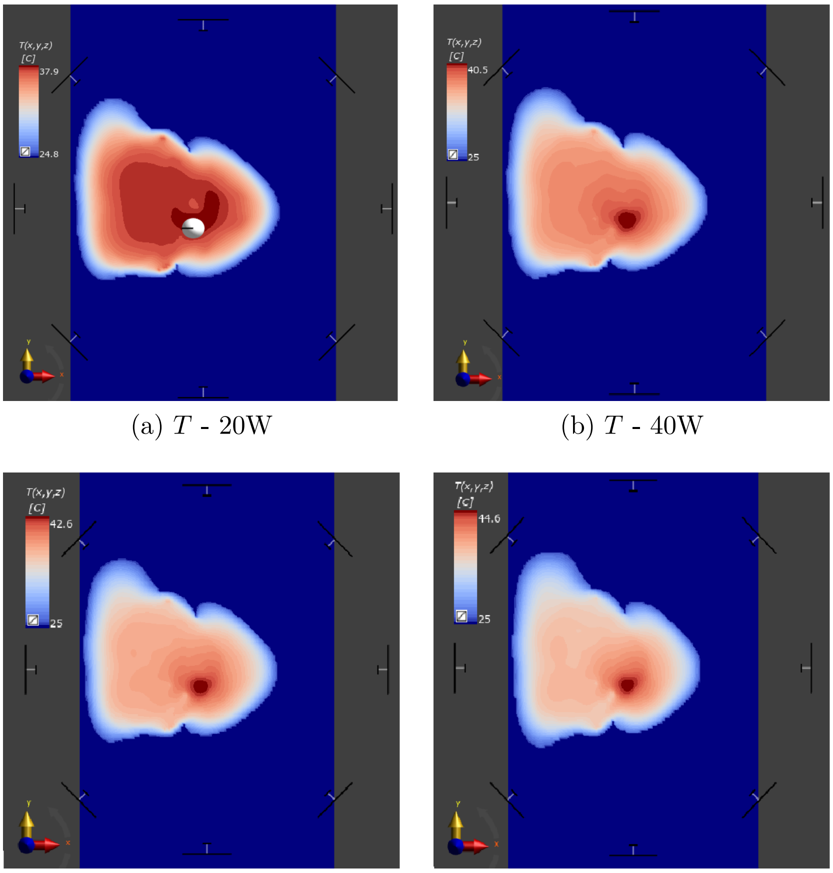
(g) T_{yz} - before (h) T_{yz} - phase-only (i) T_{yz} - phase-amplitude

Figure 2.8: Comparison of the temperature maps corresponding to the SAR profiles obtained before the optimization, after the phase-only optimization and after the phase-amplitude optimization, in each plane for a total input power of 20W.

read and manipulated them in MATLAB. (Fig.2.11).

Accordingly, the phase and amplitude optimization is successful since it was able to remove hotspots already present as well as to prevent internal ones, decrease the radiation in non-treated tissues and increase the maximum temperature in the target area. In addition, it is demonstrated that the power value was an important parameter to consider in order to obtain an appropriate temperature value within the tumor, and therefore an effective hyperthermia treatment.

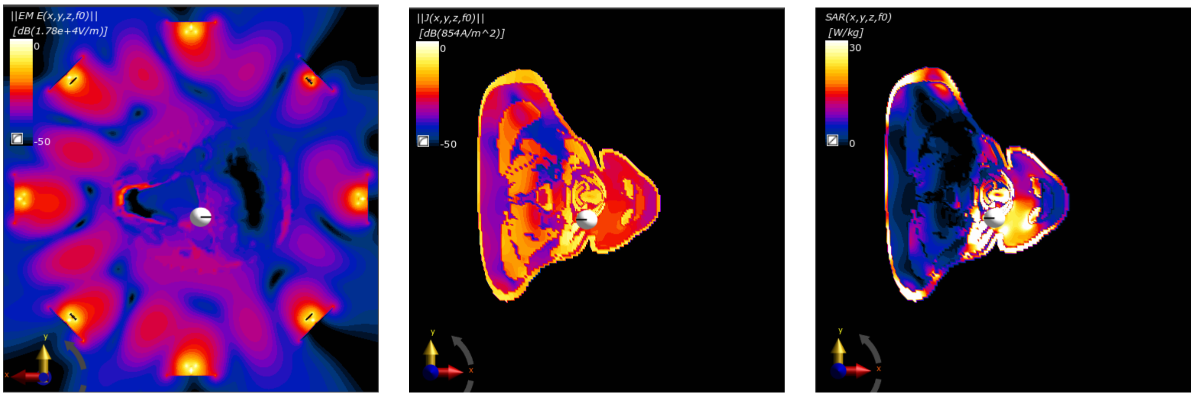
From now on all the simulations conducted in Sim4Life will be performed with the phase and amplitude values optimized at a power of 80W.



(a) T - 20W (b) T - 40W

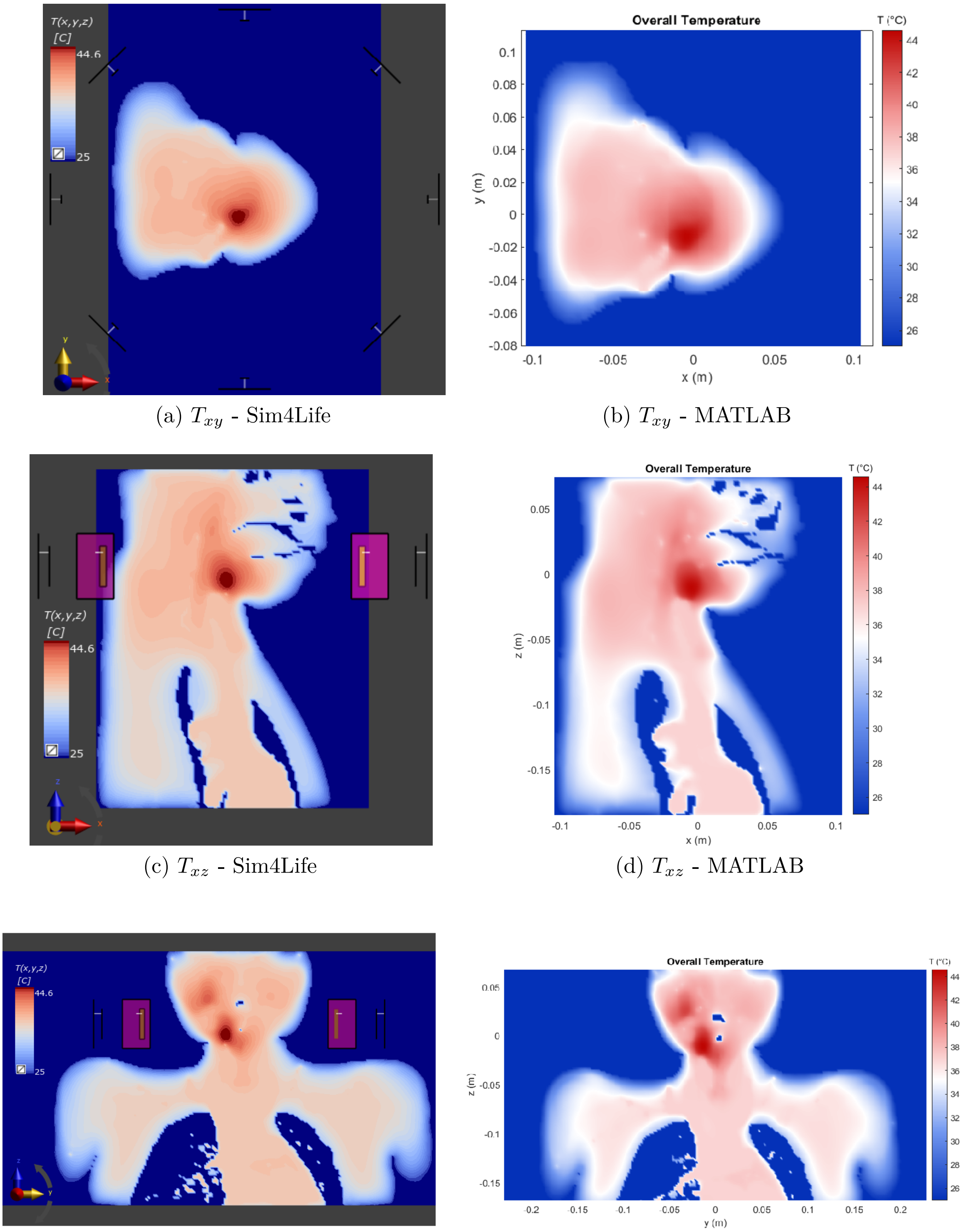
(c) T - 60W (d) T - 80W

Figure 2.9: Influence of power on the maximum temperature value, slice view in the xy plane for the phase and amplitude optimization (the white ellipsoid identifies the tumor).



(a) $|\mathbf{E}|$ (b) $|\mathbf{J}|$ (c) SAR

Figure 2.10: $|\mathbf{E}|$,$|\mathbf{J}|$ and SAR fields for a total input power of the array of 80W, slice view in the xy plane (the white ellipsoid identifies the tumor).



(a) T_{xy} - Sim4Life

(b) T_{xy} - MATLAB

(c) T_{xz} - Sim4Life

(d) T_{xz} - MATLAB

(e) T_{yz} - Sim4Life

(f) T_{yz} - MATLAB

Figure 2.11: Temperature maps at 80W, slice view in the xy, xz and yz plane from Sim4Life and MATLAB.

Chapter 3

Control via Python Scripting

"*Sim4Life features Python, a powerful scientific scripting language offering a vast range of third-party programs. The Sim4Life Python application programming interface (API) can be used to parametrize and automatize tasks, e.g., geometric modelling, simulation setup, or post-processing, and to build custom tools and independent applications.*" [28].

The Python interface is very useful to provide full interoperability and automation in order to launch simulations, perform parametrization and optimization and to automatically extract several parameters and result data of interest. In addition, it can be used to generate and share customized reports or analyses. This interface is also quite easy to understand and employ due to various routine for performing frequent tasks and the extensive API browser provided.

3.1 Data Extraction

So far, the SAR optimization was performed by means of a manual work of inserting and extracting data: in particular it was necessary to run the electromagnetic simulation in Sim4Life, then manually limit the field extraction to defined regions of the phantom using the "mask filer tool" of Sim4Life, that allows to mask a 3D field. This is obtained by assigning zero or a specified replacement value to values outside the mask and extract each field through MATLAB exporter. The extracted data is then imported in MATLAB and used by the optimizer to find the phases and amplitudes for the antennas of the array. The optimized feeding coefficients are then provided to the array in Sim4Life in order to properly combine the fields and visualize the resulting optimized SAR maps on the phantom.

3.1.1 Mask Filter

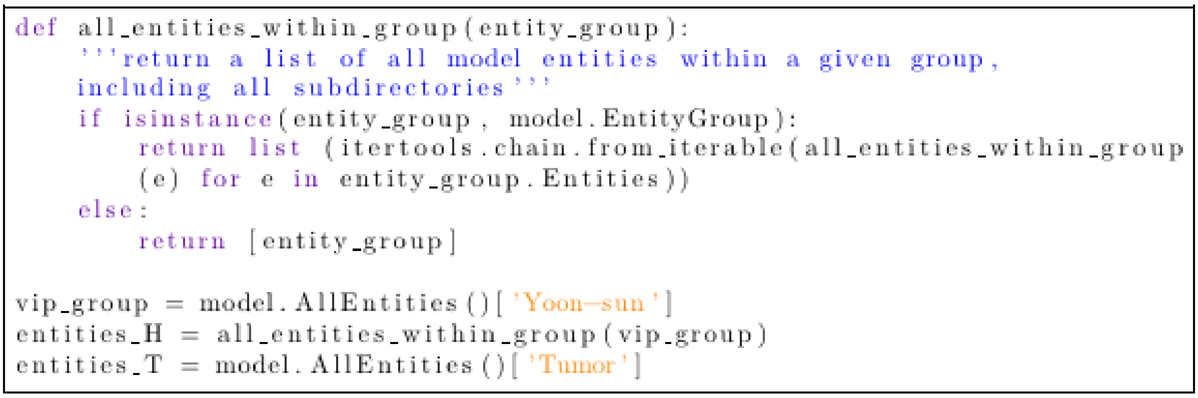
For this thesis purpose, the mask filter was used to mask out each antenna, the background and the internal air, therefore considering only the tumor and the healthy tissues. Hence, two different mask filters for healthy tissues and the tumor were created in order to analyse healthy tissues from diseased tissues separately. This protocol needs to perform 16 mask filters for the E field (2 for each antenna) and 4 mask filters for the J and SAR field (2 for each field J and SAR) extraction. The latter are necessary to calculate σ and ρ:

$$\sigma = \frac{|\mathbf{J}|}{|\mathbf{E}|} \tag{3.1}$$

$$\rho = \frac{1}{2SAR}\sigma|\mathbf{E}|^2 \tag{3.2}$$

In order to automate and speed up the process mentioned, a Python script was written which aims to automatically create mask filters for healthy tissues and tumor. Below are shown (as figure) some relevant parts of the script used, while the complete script is reported in Appendix B.

First, the model entities needed for the mask filter were chosen, in particular were selected all the Yoon-sun's healthy tissues and the tumor, therefore grouped under the name of entities_H and entities_T respectively (Fig.3.1). After, a new analysis branch was created by means of selecting the results of the EM-Multiport simulation of interest, adding a new EM-port and EM-sensor extractor (Overall Field), and choosing the field (E, J and SAR). Finally, the mask filter was performed.

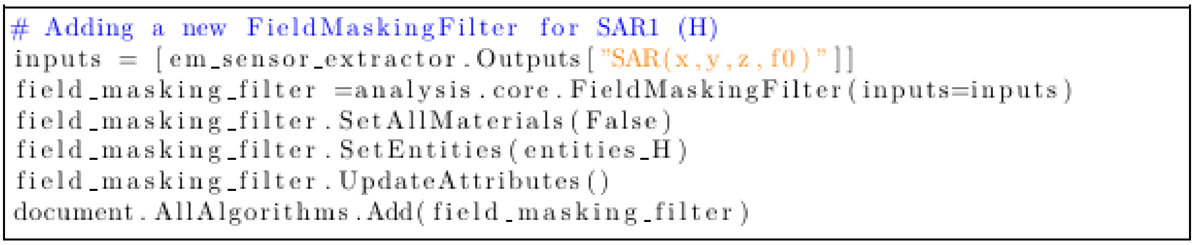


```
def all_entities_within_group(entity_group):
    '''return a list of all model entities within a given group,
    including all subdirectories'''
    if isinstance(entity_group, model.EntityGroup):
        return list (itertools.chain.from_iterable(all_entities_within_group
        (e) for e in entity_group.Entities))
    else:
        return [entity_group]

vip_group = model.AllEntities()['Yoon-sun']
entities_H = all_entities_within_group(vip_group)
entities_T = model.AllEntities()['Tumor']
```

Figure 3.1: Selection of tissues for mask filter.

The mask filter was applied using the command ”$field\_masking\_filter = analysis.core.FieldMaskingFilter(inputs = inputs)$”, then all the unwanted materials were deselected while the desired materials were ticked again.

In Fig.3.2 an extract of the original script, in which the SAR field mask filter is applied, is reported. These few lines were repeated in order to consider the tumor or the healthy tissues in the different fields.



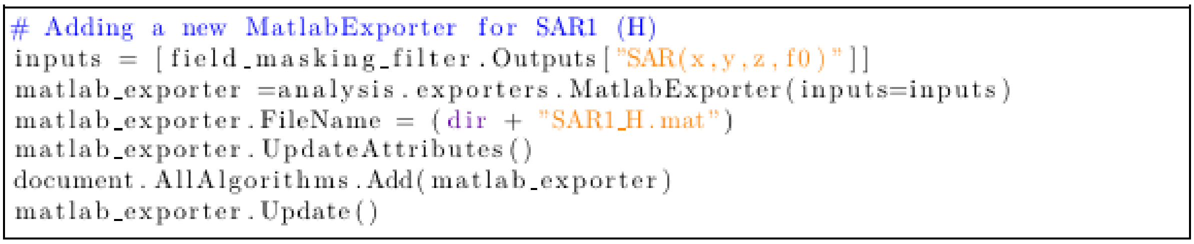
```
# Adding a new FieldMaskingFilter for SAR1 (H)
inputs = [em_sensor_extractor.Outputs["SAR(x,y,z,f0)"]]
field_masking_filter =analysis.core.FieldMaskingFilter(inputs=inputs)
field_masking_filter.SetAllMaterials(False)
field_masking_filter.SetEntities(entities_H)
field_masking_filter.UpdateAttributes()
document.AllAlgorithms.Add(field_masking_filter)
```

Figure 3.2: Mask filter in the Healthy region for the SAR field.

3.1.2 MATLAB Exporter

Once the mask filter has been applied, the next step is to extract the fields in format .mat in order to process this data in MATLAB and obtain the phase-only and phase-amplitude parameters optimized for each antenna, as in section 2.4. In these case the line used to export the data files was: "$matlab\_exporter = analysis.exporters.MatlabExporter(inputs = inputs)$".

The last line in Fig.3.3 is another fundamental command to insert in the script, because it allows you to update the results and actually export the file externally.



```
# Adding a new MatlabExporter for SAR1 (H)
inputs = [field_masking_filter.Outputs["SAR(x,y,z,f0)"]]
matlab_exporter =analysis.exporters.MatlabExporter(inputs=inputs)
matlab_exporter.FileName = (dir + "SAR1_H.mat")
matlab_exporter.UpdateAttributes()
document.AllAlgorithms.Add(matlab_exporter)
matlab_exporter.Update()
```

Figure 3.3: MATLAB exporter in the Healthy region for the SAR field.

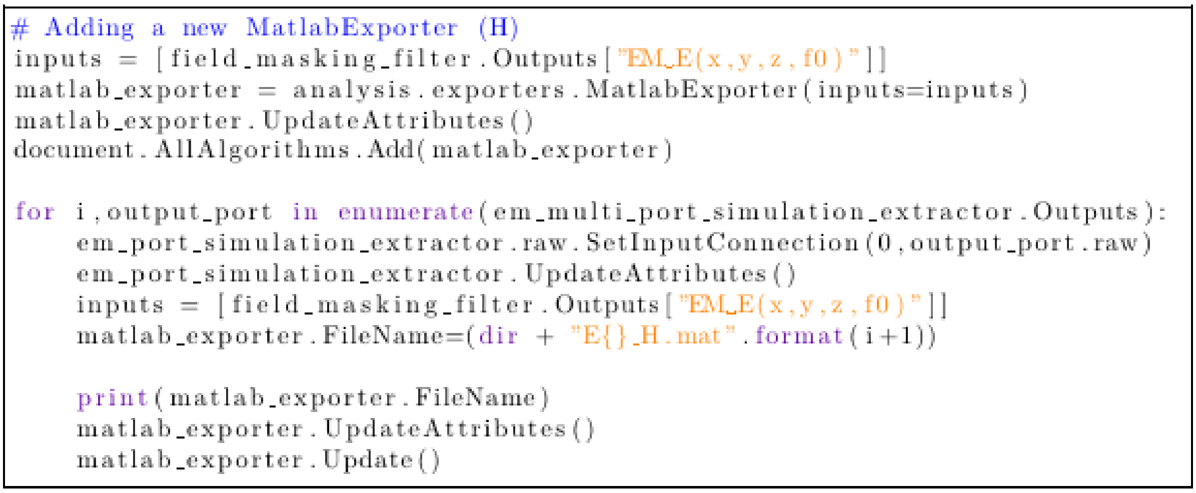
3.1.3 E Field

The methods shown above work and are effective when the mask filter and MATLAB extraction is performed a limited number of times, as in the case of J and SAR (only 2 cases are considered).

For the electric field E, that needs to be accounted several times, two times for each antenna (8 times for the healthy tissues and 8 times for the tumor tissues), a cunning route was employed, as shown in Fig.3.4.

In this case a for loop was used in order to iterate the same command for each antenna and hence shorten and optimize the script.

"$em\_port\_simulation\_extractor.raw.SetInputConnection(0, output\_port.raw)$" is the main line that allows to achieve this result. In particular, it gets raw internal implementation object, i.e. each source, and connect each port (hence each antenna).



```
# Adding a new MatlabExporter (H)
inputs = [field_masking_filter.Outputs["EM_E(x,y,z,f0)"]]
matlab_exporter = analysis.exporters.MatlabExporter(inputs=inputs)
matlab_exporter.UpdateAttributes()
document.AllAlgorithms.Add(matlab_exporter)

for i,output_port in enumerate(em_multi_port_simulation_extractor.Outputs):
    em_port_simulation_extractor.raw.SetInputConnection(0,output_port.raw)
    em_port_simulation_extractor.UpdateAttributes()
    inputs = [field_masking_filter.Outputs["EM_E(x,y,z,f0)"]]
    matlab_exporter.FileName=(dir + "E{}_H.mat".format(i+1))

    print(matlab_exporter.FileName)
    matlab_exporter.UpdateAttributes()
    matlab_exporter.Update()
```

Figure 3.4: Short route to apply the mask filter and MATLAB extraction for the E field of the 8 antennas.

Subsequently proceeding as in section 2.3, it was verified that the optimized phase and amplitude values obtained with the manual procedure and the one automated by the Python script were the same. Hence the Python script works and can be used to automatically and quickly derive field values.

3.2 Parameters Variation

"Besides validation of the numerical model, any medical application evaluated by mathematical simulation needs the careful assessment of all particular variables and their influence on the therapeutic effect and efficiency of results." [20].

As already said, to achieve an effective hyperthermia treatment it is necessary to consider thermal and dielectric properties of tissues, which affect the heat transfer and impact absorption of electromagnetic energy respectively. Therefore, these properties are crucial to pre-clinical and clinical application and need to be investigated.

In this section a quantitative study, aiming at assessing the influence of some important model parameters on the simulation process and results, is also performed.

3.2.1 Thermal Parameters

Heat transport in biological tissues may occur due to conductive, convective or radiative mechanisms. These heat transfer mechanisms can be characterized via several parameters, but in this thesis only blood perfusion ω [$\mathrm{ml\,kg^{-1}\,min^{-1}}$] and thermal conduction k [$\mathrm{W\,m^{-1}\,K^{-1}}$] are considered. Blood flow strongly affects temperature distribution due to convective heat transfer between the tissue and circulating blood during heating; while the thermal conductivity describes how well the tissue conducts heat.

The values of these parameters were assigned to 4 different tissues, *Muscle*, *Fat + SAT* (where SAT identifies the Subcutaneous Adipose Tissue), *Skin* and *Tumor* according to the material database in [30], using a tumor perfusion value 1.85 times higher than the rest value for the muscle perfusion (Tab.3.1 and Tab.3.2).

Table 3.1: Perfusion.

$\boldsymbol{\omega}$ [ml kg^{-1} min^{-1}]	Muscle	Fat	Skin	Tumor
Average	39.1	33	106	72.3
Minimum	39.1	33	106	72.3
Maximum	442	255	175	848

Table 3.2: Thermal conductivity.

$\boldsymbol{k}$ [W m^{-1} K^{-1}]	Muscle	Fat	Skin	Tumor
Average	0.49	0.21	0.37	0.51
Minimum	0.40	0.18	0.32	0.48
Maximum	0.56	0.5	0.5	1.5

In order to evaluate the tissues behaviour in a defined range of variation, perfusion and thermal conductivity ranges were split in equidistant values in MATLAB, then these values were changed in Sim4Life, the simulation was run, thermal maps were generated and exported in MATLAB to produce several plots.

The range was defined starting from the average value according to the material database. Since real perfusion values are supposed to be much greater than the baseline values, only maximum thresholds were fixed according to the indications reported in [21]. These values, written in Excel, were extracted via MATLAB script (Appendix A) and then they were divided starting from the baseline value, up to the maximum and minimum value by applying the following variation:

$$\omega_i = \overline{\omega} + x\% \cdot \Delta\omega \tag{3.3}$$

where $\overline{\omega}$ is the average value, $\Delta\omega = \omega_{max} - \omega_{min}$ and $x =$ 0,10,20,...,100.

$$\begin{aligned} k_{i+} &= \overline{k} + x_{+}\% \cdot \Delta k_{+} \\ k_{i-} &= \overline{k} - x_{-}\% \cdot \Delta k_{-} \end{aligned} \tag{3.4}$$

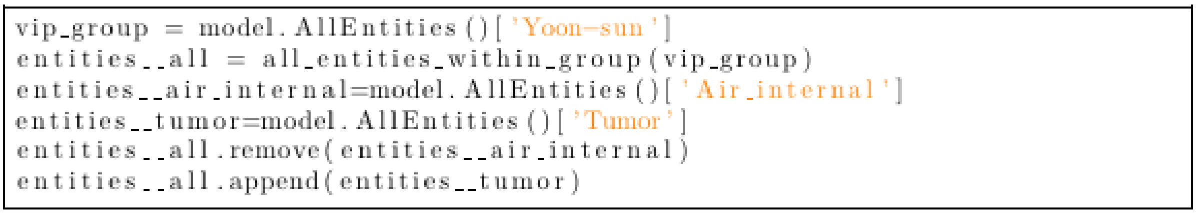
where $\overline{k}$ is the average value, $\Delta k_{+} = k_{max} - \overline{k}$ and $x_{+} =$ 0,10,20,...,100, while for the backwards value $\Delta k_{-} = \overline{k} - k_{min}$ and $x_{-} =$ 20,40,60,80,100.

It can be noted that 11 forward values were selected for perfusion, while for thermal conductivity were considered 11 forward values, as for perfusion, and 5 backward values for a total of 16 values. It is important to note that, when a parameter is changed according to the above-reported equations, the other parameters are left to their baseline average values.

The next steps in order to produce thermal maps were to replace these values for each tissue (once at time) in Sim4Life's thermal simulation, then run the simulation, apply the mask filter, generate thermal maps and finally export them as MATLAB file.

Again in this case, changing the values manually one at a time was not considered the best solution; for this reason, a Python script was created (Appendix B) of which some extracts are shown as images below.

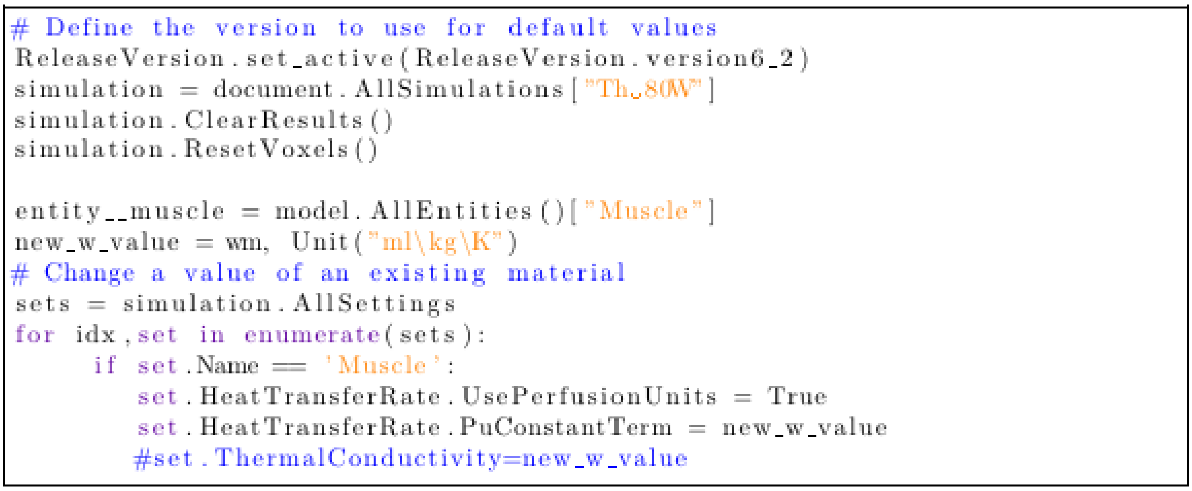
First the model entities needed for the mask filter were chosen as before. In particular were selected all the Yoon-sun's healthy tissues, taking care to remove the background and internal air because they are outside the computation domain of the bioheat equation (1.3), then was added the Tumor entity (Fig.3.5).



```
vip_group = model.AllEntities()['Yoon-sun']
entities__all = all_entities_within_group(vip_group)
entities__air_internal=model.AllEntities()['Air_internal']
entities__tumor=model.AllEntities()['Tumor']
entities__all.remove(entities__air_internal)
entities__all.append(entities__tumor)
```

Figure 3.5: Entities needed for the mask filter.

Using the command "*coord* = *numpy.loadtxt*("*C* : /*Users*/.../*Range.txt*")" the MATLAB file was loaded containing the perfusion or thermal conductivity values to be considered for each tissue.

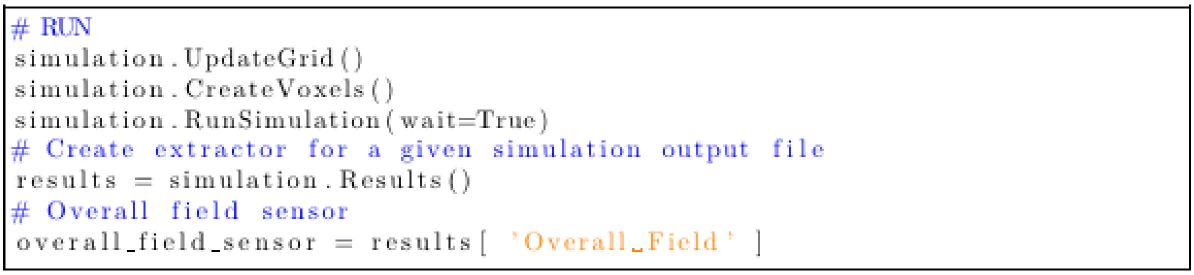


```
# Define the version to use for default values
ReleaseVersion.set_active(ReleaseVersion.version6_2)
simulation = document.AllSimulations["Th␣80W"]
simulation.ClearResults()
simulation.ResetVoxels()

entity__muscle = model.AllEntities()["Muscle"]
new_w_value = wm, Unit("ml\kg\K")
# Change a value of an existing material
sets = simulation.AllSettings
for idx,set in enumerate(sets):
      if set.Name == 'Muscle':
         set.HeatTransferRate.UsePerfusionUnits = True
         set.HeatTransferRate.PuConstantTerm = new_w_value
         #set.ThermalConductivity=new_w_value
```

Figure 3.6: Command lines needed to reset the simulation and change the parameters. The last commented line shows how to modify the thermal conductivity.

Then, a *for* loop was written which takes into consideration all the values by replacing them one at a time within the properties of the considered tissue, making the simulation overwrite each time and repeating the same commands for the construction of the temperature maps. After, it was necessary to delete the results and reset the voxels of the simulation of interest. These operations are mandatory because after the voxels have been created, all the simulations settings become locked, so updating whatever setting is forbidden. This is done to also keep consistency between the actual grid and the corresponding settings, like materials, min and max frequency, sensors, etc., that contributed to generate it. Then, the entity is identified and a new value is assigned to it (Fig.3.6).

It is important to check that the units of measurement set by default in Sim4Life correspond to those of the Literature: in our case the perfusion units did not correspond, so it was necessary to add this line $set.HeatTransferRate.UsePerfusionUnits = True$. Once the data was updated, the thermal simulation was run again and a new branch was created in the analysis section as in Fig.3.7. Finally, as described before (section 3.1.1 and 3.1.2) the mask filter was applied and the thermal maps were exported as file MATLAB. The thermal maps obtained represent the entire three-dimensional grid.



```
# RUN
simulation.UpdateGrid()
simulation.CreateVoxels()
simulation.RunSimulation(wait=True)
# Create extractor for a given simulation output file
results = simulation.Results()
# Overall field sensor
overall_field_sensor = results[ 'Overall␣Field' ]
```

Figure 3.7: Command lines needed to restart the simulation and create a new analysis branch.

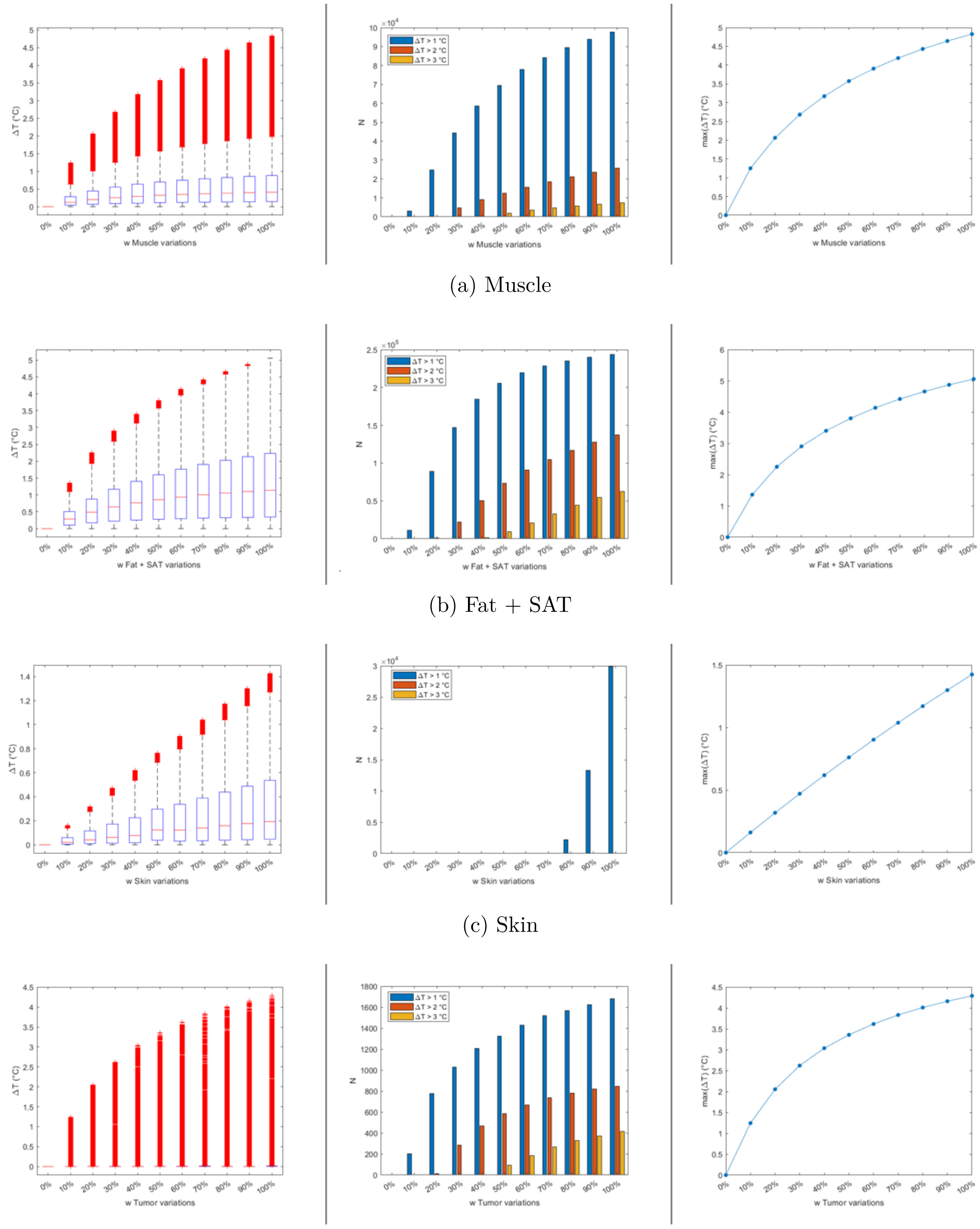
In order to estimate the temperature maps variation as consequence of the thermal parameter's variation, three statistical plots were created: a boxplot, a bar diagram and a simple trend line (Fig3.8 and Fig.3.9). In detail, the boxplot plots one box for each column of the considered matrix, where the central mark indicates the median, and the bottom and top edges of the box indicate the 25th and 75th percentiles, respectively. The whiskers extend to the most extreme data points not considered outliers, and the outliers are plotted individually using the '+' marker symbol [32]. The bar diagram reports the number of points in the region of interest corresponding to a temperature variation ΔT greater than 1°C, 2°C and 3°C; while the 2D-plots reports the maximum value of ΔT_{var} in the region of interest.

The temperature variation (ΔT_{var}) was calculated as difference between the baseline thermal maps (T_b) and the temperature map corresponding to the variation of a parameter:

$$\Delta T_{var}(\mathbf{r}) = |T_b(\mathbf{r}) - T_{var}(\mathbf{r})| \tag{3.5}$$

where $\mathbf{r}$ indicates the spatial grid points in the human phantom, for which the bioheat equation is computed.

Then all the 2D-plots were combined to visually show which of the 4 tissues (*Muscle, Fat + SAT, Skin and Tumor*) was the most relevant for perfusion (Fig.3.10) as well as thermal conductivity (Fig.3.11).



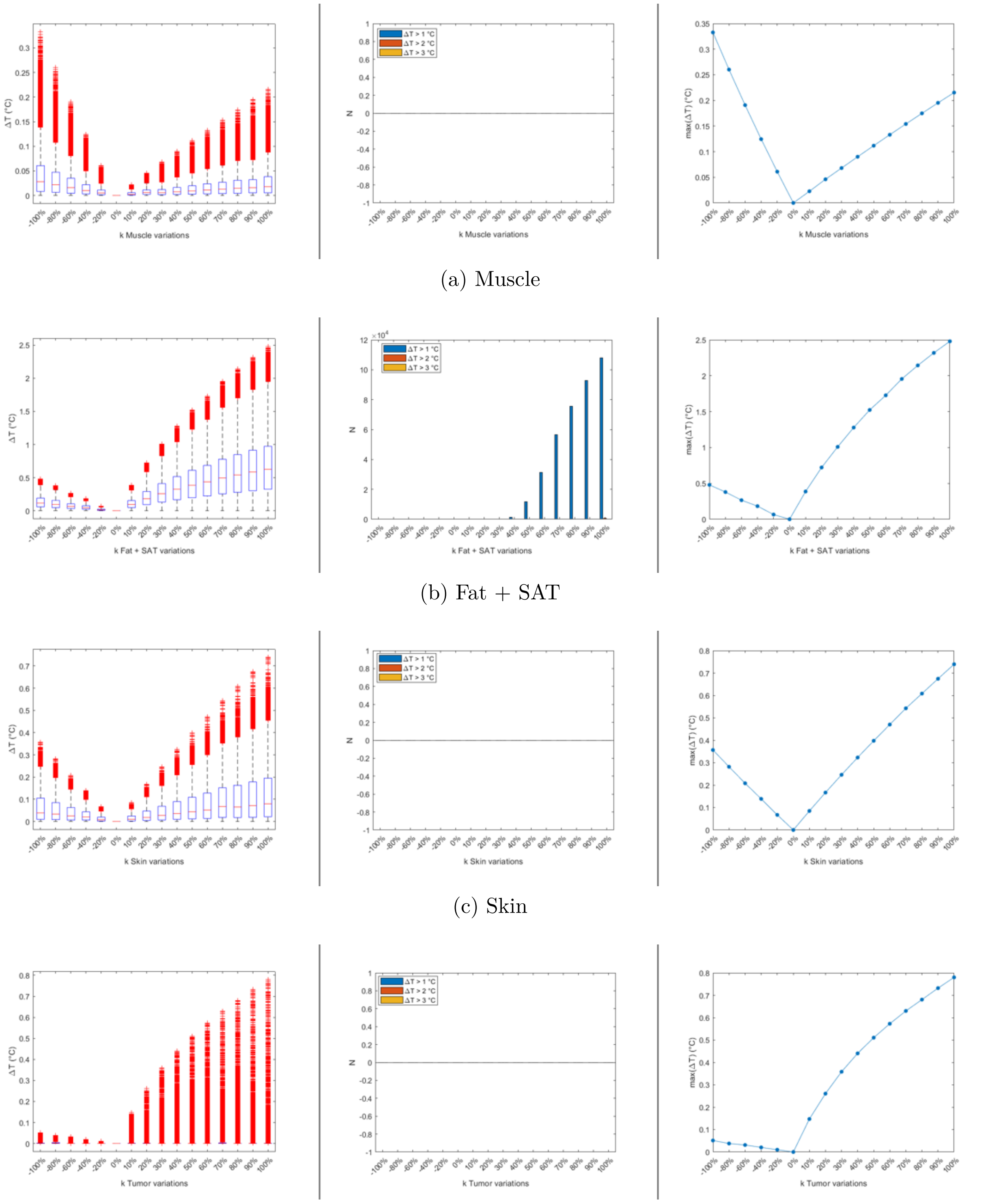
(a) Muscle

(b) Fat + SAT

(c) Skin

(d) Tumor

Figure 3.8: Boxplot statistics, histogram and maximum temperature variation corresponding to different percentage changes in the muscle, fat, skin, and tumor thermal conductivity with respect to the baseline value.



(a) Muscle

(b) Fat + SAT

(c) Skin

(d) Tumor

Figure 3.9: Boxplot statistics, histogram and maximum temperature variation corresponding to different percentage changes in the muscle, fat, skin, and tumor thermal conductivity with respect to the baseline value.

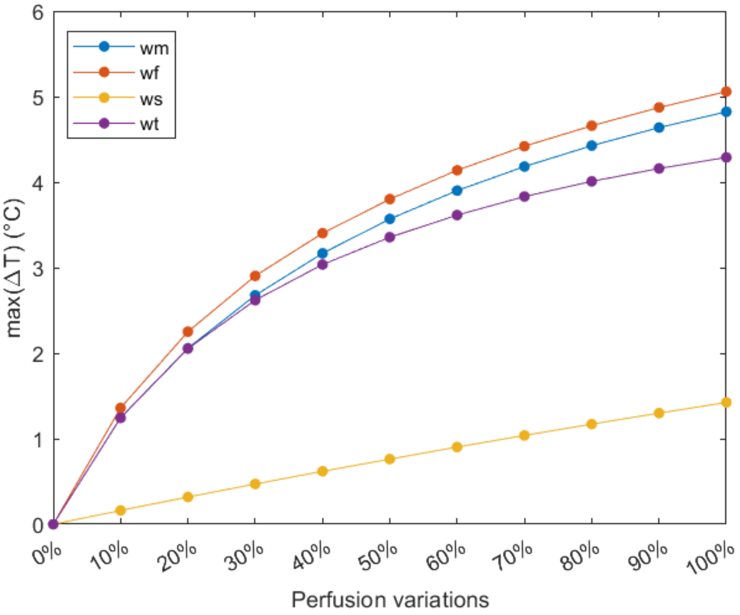


Figure 3.10: Maximum temperature variation in the region of interest corresponding to different percentage changes in the muscle, fat, skin, and tumor perfusion with respect to the baseline value.

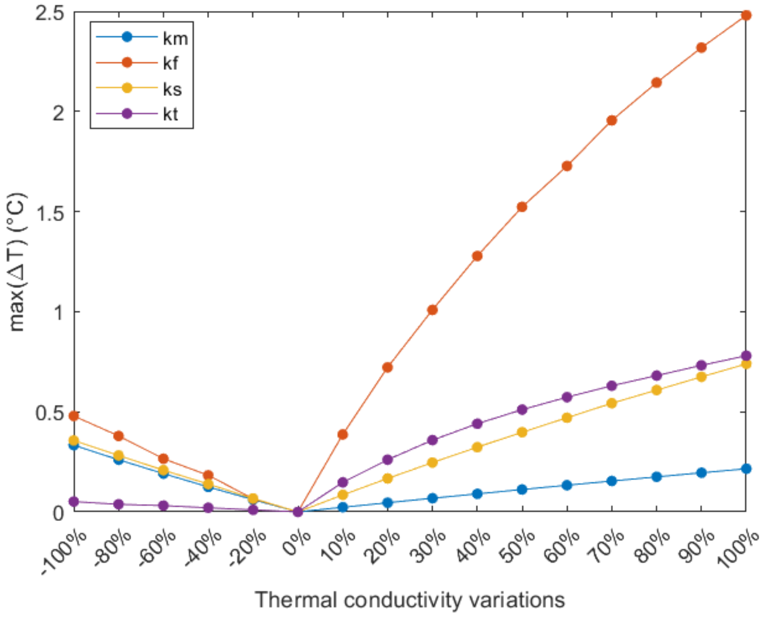


Figure 3.11: Maximum temperature variation in the region of interest corresponding to different percentage changes in the muscle, fat, skin, and tumor thermal conductivity with respect to the baseline value.

It is important to note that together with the *Fat (Average Infiltrated)* tissue the subcutaneous fat tissue (*SAT*) is also considered due to the database information: the *SAT* tissue reports the same values as the *Average Infiltered* one so it was necessary to vary it too during the analysis.

In all cases there is an increasing trend of the curve corresponding to an increase in the value of the parameter with respect to the baseline starting point. Furthermore, as it can be seen for both the perfusion variation and the thermal conductivity, the *Fat + SAT* tissue is the predominant one followed by *Muscle* and *Tumor* tissues in the first case, by *Skin* and *Tumor* in the second case.

In Chapter 4, the temperature reconstruction will be presented for the initial simple case in which only two parameters are varied. Since the perfusion provides the most critical variations (with respect to the baseline map), as shown in Fig.3.10, we decided to consider this parameter for two different tissues: *Tumor* and *muscle* (a further development should include the *Fat + SAT* too, according to Fig.3.10).

3.2.2 Dielectric Parameters

The knowledge of electrical properties of biological tissues is important to calculate the deposition of electromagnetic energy. The heat generation in biological tissues due to dissipation of electromagnetic energy is dependent on the relative permittivity ϵ_r and the electrical conductivity σ [S m^{-1}]. Furthermore, at microwave frequencies, dielectric properties are primarily determined by water content.

Also for this case, these parameters were assigned to *Muscle*, *Fat + SAT*, *Skin* and *Tumor* tissues according to the database [30], they are reported in Tab.3.3 and Tab.3.4.

Table 3.3: Relative permittivity.

$\boldsymbol{\epsilon_r}$	Muscle	Fat	Skin	Tumor
Average	56.9	11.6	49.4	59
Minimum	54	11	46.9	56
Maximum	59.7	12.2	51.9	62

Table 3.4: Electrical conductivity.

$\boldsymbol{\sigma}$ [S m^{-1}]	Muscle	Fat	Skin	Tumor
Average	0.805	0.082	0.681	0.89
Minimum	0.765	0.078	0.647	0.846
Maximum	0.845	0.086	0.715	0.935

For relative permittivity and electrical conductivity variation ranges were considered as follows:

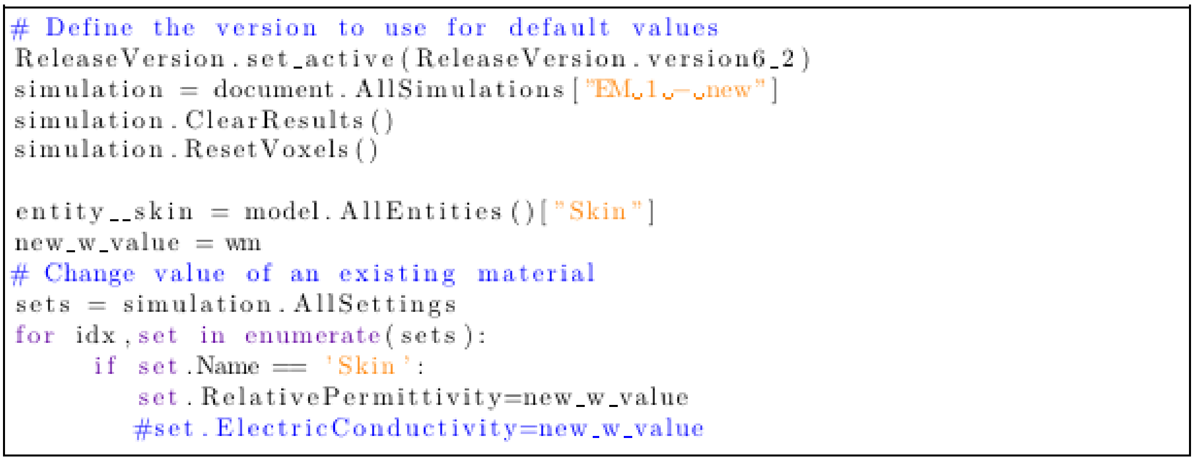
$$\begin{aligned} \epsilon_{min} &= \overline{\epsilon}_r - 5\% \cdot \overline{\epsilon}_r \\ \epsilon_{max} &= \overline{\epsilon}_r + 5\% \cdot \overline{\epsilon}_r \end{aligned} \tag{3.6}$$

$$\begin{aligned} \sigma_{min} &= \overline{\sigma} - 5\% \cdot \overline{\sigma} \\ \sigma_{max} &= \overline{\sigma} + 5\% \cdot \overline{\sigma} \end{aligned} \tag{3.7}$$

Hence in this case only two parameter's variation for each tissue were considered.

The next steps, aimed at analysing the tissues behaviour as well as study how the heat is delivered to tissues during microwave treatment, are the same as already described in the previous section: these values were changed in Sim4Life, the simulation was run, thermal maps were generated and exported in MATLAB to produce comparison plots.

In this case the python script used differs from the previous one for: the simulation considered to vary the parameters, in this case the parameters must be changed in the electromagnetic node (EM FDTD). The resulting SAR map employed during the subsequent thermal simulation must then be updated for each parameters variation. Figures 3.12 to 3.14 report some extracts of the script, regarding the differences described; while the complete script can be found in the Appendix B.

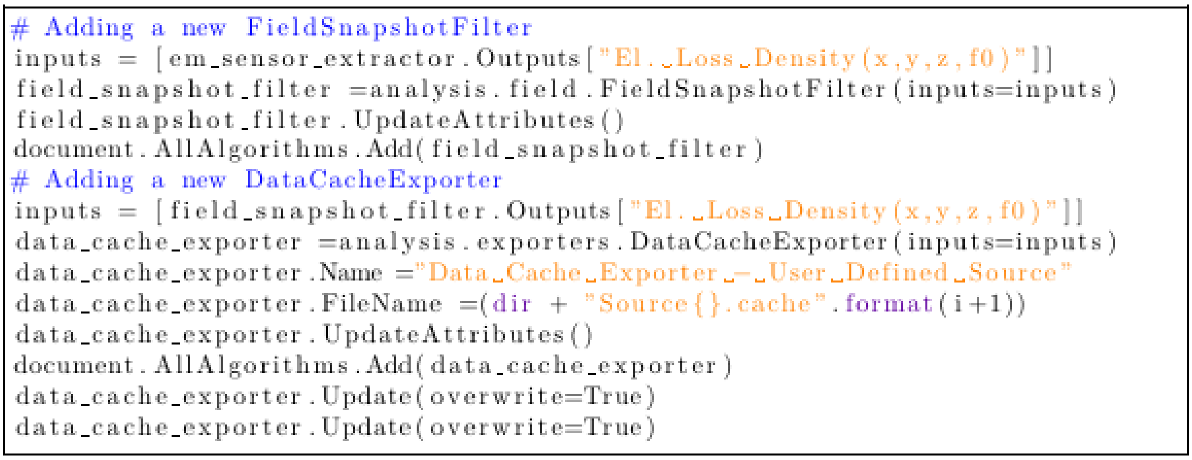


```
# Define the version to use for default values
ReleaseVersion.set_active(ReleaseVersion.version6_2)
simulation = document.AllSimulations["EM␣1␣—␣new"]
simulation.ClearResults()
simulation.ResetVoxels()

entity__skin = model.AllEntities()["Skin"]
new_w_value = wm
# Change value of an existing material
sets = simulation.AllSettings
for idx,set in enumerate(sets):
      if set.Name == 'Skin':
         set.RelativePermittivity=new_w_value
        #set.ElectricConductivity=new_w_value
```

Figure 3.12: Command lines needed to change the dielectric parameter. The last commented line shows how to modify the electrical conductivity.

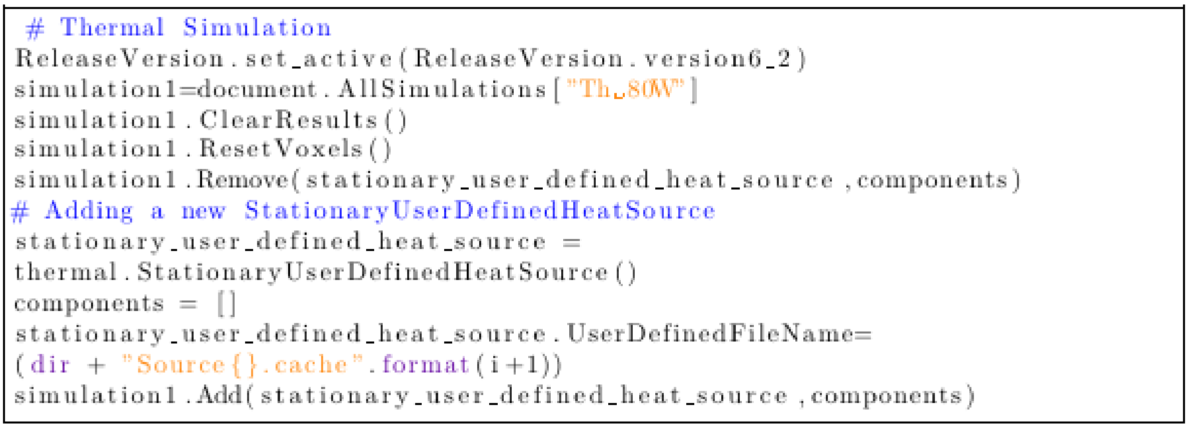
As done before, the thermal maps were extracted and exported in MATLAB in order to calculate the temperature variation ΔT_{var} and then produce two plots to compare the permittivity and electrical conductivity variation of each tissue(Figures 3.15 and 3.16).

From an overall view of the comparison between the boxplots of the tissues, it is possible to observe that the variation of the dielectric permittivity and conductivity does not significantly affect the temperature inside the tissues.



```
# Adding a new FieldSnapshotFilter
inputs = [em_sensor_extractor.Outputs["El.␣Loss␣Density(x,y,z,f0)"]]
field_snapshot_filter =analysis.field.FieldSnapshotFilter(inputs=inputs)
field_snapshot_filter.UpdateAttributes()
document.AllAlgorithms.Add(field_snapshot_filter)
# Adding a new DataCacheExporter
inputs = [field_snapshot_filter.Outputs["El.␣Loss␣Density(x,y,z,f0)"]]
data_cache_exporter =analysis.exporters.DataCacheExporter(inputs=inputs)
data_cache_exporter.Name ="Data␣Cache␣Exporter␣-␣User␣Defined␣Source"
data_cache_exporter.FileName =(dir + "Source{}.cache".format(i+1))
data_cache_exporter.UpdateAttributes()
document.AllAlgorithms.Add(data_cache_exporter)
data_cache_exporter.Update(overwrite=True)
data_cache_exporter.Update(overwrite=True)
```

Figure 3.13: Extraction of the Source.



```
 # Thermal Simulation
ReleaseVersion.set_active(ReleaseVersion.version6_2)
simulation1=document.AllSimulations["Th␣80W"]
simulation1.ClearResults()
simulation1.ResetVoxels()
simulation1.Remove(stationary_user_defined_heat_source,components)
# Adding a new StationaryUserDefinedHeatSource
stationary_user_defined_heat_source =
thermal.StationaryUserDefinedHeatSource()
components = []
stationary_user_defined_heat_source.UserDefinedFileName=
(dir + "Source{}.cache".format(i+1))
simulation1.Add(stationary_user_defined_heat_source,components)
```

Figure 3.14: Command lines needed to add the Source in the Thermal simulation.

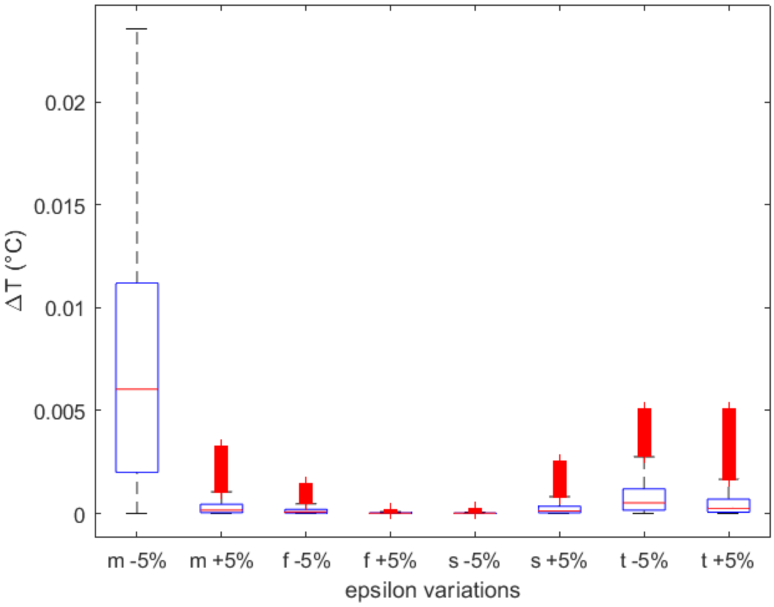


Figure 3.15: Boxplot statistics corresponding to the variation of the relative permittivity for the different considered tissues.

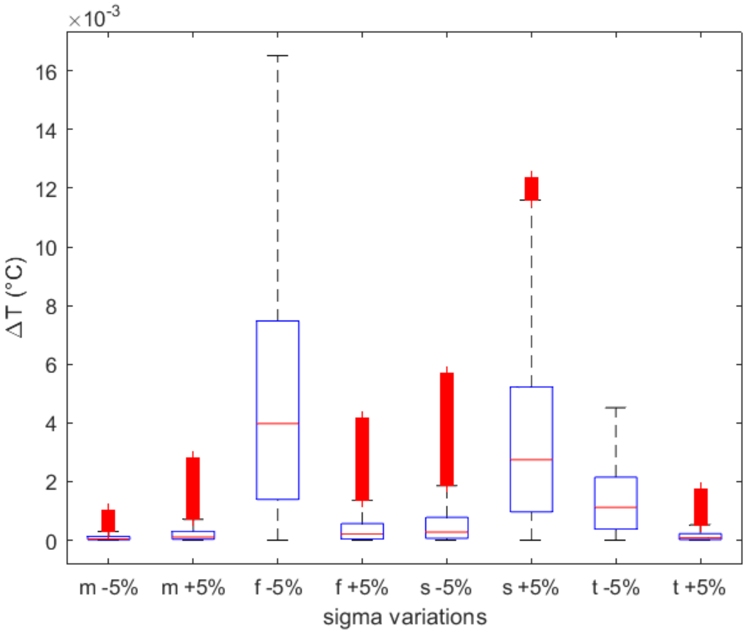


Figure 3.16: Boxplot statistics corresponding to the variation of the electrical conductivity for the different considered tissues.

Chapter 4

Temperature Monitoring

Temperature monitoring is considered to be particularly beneficial for adjusting delivered energy settings during treatment as well as ensure damage to the tumor region while preserving healthy surrounding anatomical structures. Hence, an accurate measurement of tissue temperature may be particularly beneficial to improve the treatment outcome, because it can be used as a clear end-point to achieve complete tumor regression and minimize recurrence [22, 31].

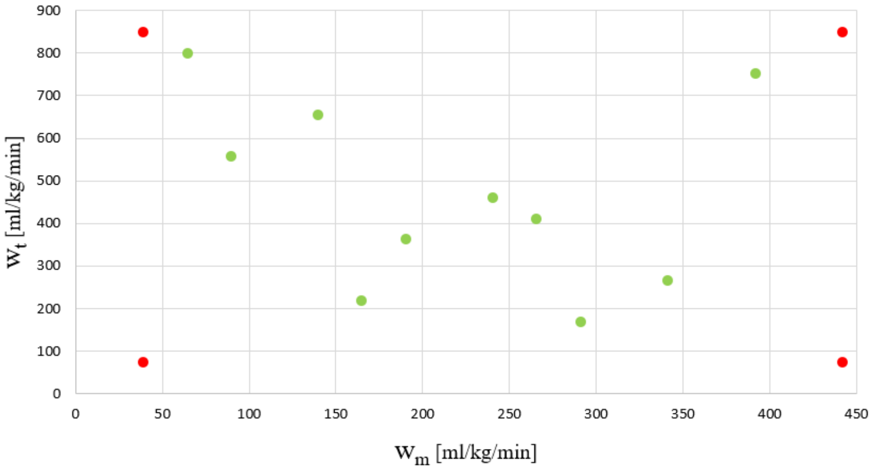
For the H&N region this gains further relevance due to the presence of vulnerable structure such as the spinal cord, nerves, lymph nodes and glands. Therefore, it is important to prevent cytotoxic temperature in these susceptible areas to improve the procedure's safety and efficacy. As explained in Chapter 1, simulations play a fundamental role in hyperthermia treatment planning. However, due to the uncertainties characterizing tissue parameters (as emphasized in Chapter 3), temperature control cannot be based exclusively on simulations. For this reason, fiber-optic sensors are inserted into closed-tip catheters during treatment to directly control the temperature in some points in an invasive way. Improving the reliability of simulations will limit the number of catheters and hence the safety of the treatment itself.

4.1 Numerical Reconstruction

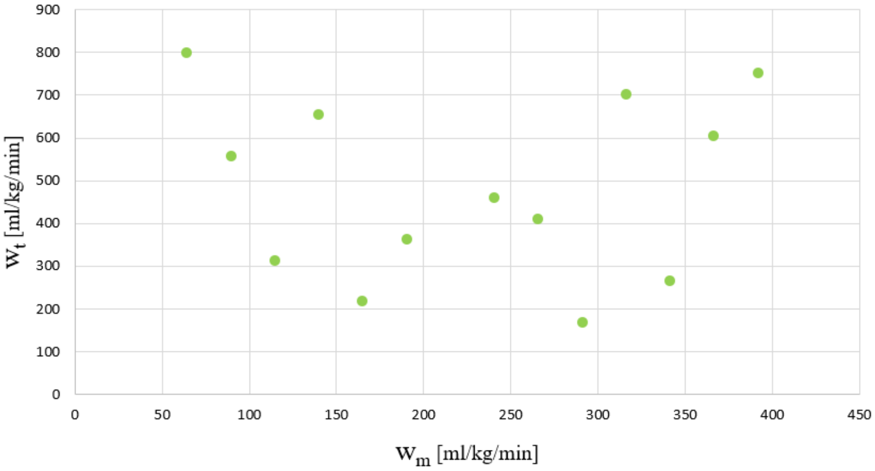
The goal is to generate a pseudo-random grid of M points in a N-dimensional space, where N indicates the number of parameters that we decide to vary. This grid will then be used in Sim4Life to obtain the temperature maps for each point of the grid considered. Finally, the extracted maps will be used to produce the S_{var} matrix of the temperatures and the S_{cath} matrix of the temperatures along the catheter's direction.

Before starting with the reconstruction of the temperature maps, it is necessary to define and understand which are the parameters and the tissues that most affect the temperature variation in order to gain greater information, while shortening the computational time. From the previous analyses it has been seen that the thermal parameters have a greater influence on the temperature variation than the dielectric parameters. Furthermore, among the thermal parameters the perfusion of *Fat + SAT*, *Muscle* and *Tumor* tissues overtake the others. In this thesis, we decided to implement the reconstruction by considering a 2-dimensional parameters' space, where the considered parameters are the muscle and tumor perfusions.

Thus, the grid is composed by $N = 2$ columns representing the perfusion of the *Muscle* and the *Tumor* (parameters we decide to vary) and M rows. In this thesis, two types of multidimensional grids have been generated: *Multigrid A* and *Multigrid B*. The first grid has dimensions 14×2 where the first 4 rows indicates the extreme bounds of the grid, assuming that the perfusion of muscle and tumor vary in the ranges reported in Table 3.1 and the remaining 10 rows are the quasi-random values from a ND-Sobol sequence. On the other hand, the second grid consider only 13 quasi-random values from the ND-Sobol sequence and the average value (as first row) to produce a 14×2 grid.



(a) Multigrid A



(b) Multigrid B

Figure 4.1: Points distribution in the plane for *Multigrid A* and *Multigrid B*.

In Fig.4.1 is displayed the points distribution of the 2 multigrids: the red points in fig.a are the extreme bounds of the grid and are fixed, while the green points (fig.a and fig.b) are the quasi-random points produced by the Sobol-sequence. However, in both cases the Sobol sequence was used so the values are well distributed due to a finer uniform partitions of the unit interval and then a reorder of the coordinates in each dimension.

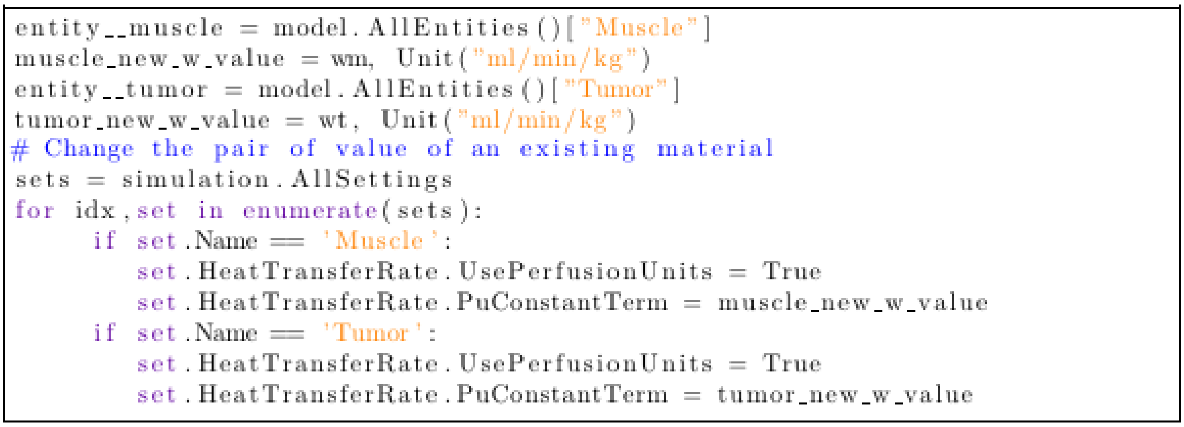
In order to generate these grids, a MATLAB script (Appendix A) was developed where ND-Sobol sequence was employed to produce the pseudo-random values starting from the average values and the variation ranges. The values for each grid (Tab.4.1 and Tab.4.2) were substituted in the perfusion properties of each tissue in Sim4life's simulation, using a Python script. This time the pairs of values (ω_m,ω_t) are replaced simultaneously (Fig.4.2). Then the thermal simulation is run, the mask filter is applied in order to remove the background and internal air, and finally the temperature maps, for each pair of values, are extracted as MATLAB format.

Table 4.1: Multigrid A.

$\boldsymbol{\omega_m}$	$\boldsymbol{\omega_t}$
[ml kg^{-1} min^{-1}]	[ml kg^{-1} min^{-1}]
39.1	72.3
442	72.3
39.1	848
442	848
240.55	460.15
139.83	654.08
341.28	266.23
89.46	557.11
290.91	169.26
190.19	363.19
391.64	751.04
64.28	799.52
265.73	411.67
165.01	217.74

Table 4.2: Multigrid B.

$\boldsymbol{\omega_m}$	$\boldsymbol{\omega_t}$
[ml kg^{-1} min^{-1}]	[ml kg^{-1} min^{-1}]
39.1	72.3
240.55	460.15
139.83	654.08
341.28	266.23
89.46	557.11
290.91	169.26
190.19	363.19
391.64	751.04
64.28	799.52
265.73	411.67
165.01	217.74
366.46	605.59
144.64	314.71
316.09	702.56



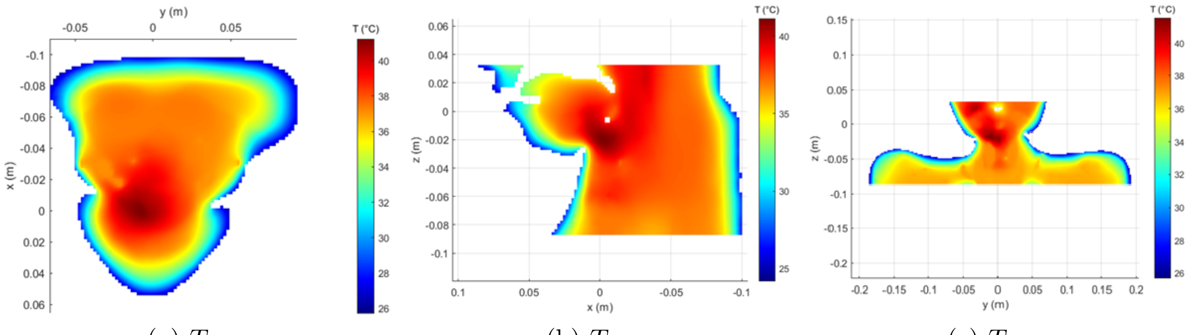
```
entity__muscle = model.AllEntities()["Muscle"]
muscle_new_w_value = wm, Unit("ml/min/kg")
entity__tumor = model.AllEntities()["Tumor"]
tumor_new_w_value = wt, Unit("ml/min/kg")
# Change the pair of value of an existing material
sets = simulation.AllSettings
for idx,set in enumerate(sets):
      if set.Name == 'Muscle':
         set.HeatTransferRate.UsePerfusionUnits = True
         set.HeatTransferRate.PuConstantTerm = muscle_new_w_value
      if set.Name == 'Tumor':
         set.HeatTransferRate.UsePerfusionUnits = True
         set.HeatTransferRate.PuConstantTerm = tumor_new_w_value
```

Figure 4.2: Command lines needed to change the two parameters ω_m and ω_t simultaneously.

4.2 Temperature Maps

New MATLAB scripts (Appendix A) are written to display the temperature maps extracted and to create the S_{var} and S_{cath} matrices. This procedure was employed for both *MultigridA* and *MultigridB* in order to monitor the temperature along the catheter and evaluate the statistics behaviour.

One of the temperature maps extracted, shown below in Fig.4.3, is plotted in xy, xz and yz plane on the slice that corresponds to the temperature maximum.



(a) $T_{14}xy$

(b) $T_{14}xz$

(c) $T_{14}yz$

Figure 4.3: Temperature map corresponding to the 14th element in the Multigrid A: slice view in xy, xz and yz plane.

The *Svar* matrix (Appendix B) is a matrix in which the nth column reports the temperature map obtained in the whole region of interest when the nth pair of perfusion values (ω_m,ω_t) is assigned to the corresponding tissues. The *Svar* matrix is defined for the *MultigridA*:

$$[S_{var}] = [[\mathbf{T_b}], [\mathbf{T_2}], [\mathbf{T_3}], [\mathbf{T_4}], [\mathbf{T_{Sobol,t=1}}], [\mathbf{T_{Sobol,t=2}}], ..., [\mathbf{T_{Sobol,t=n}}]] \tag{4.1}$$

where $\mathbf{T}$ is the temperature field: the first 4 columns correspond to the extreme bounds of the perfusion ranges and the other columns are temperature maps corresponding to a Sobol extraction of the parameters (ω_m, ω_t). While, for the *MultigridB* in the *Svar* matrix the first column is the baseline map and the other columns are temperature maps corresponding to a Sobol extraction of the parameters (ω_m, ω_t).

In order to create this matrix, for the *MultigridA*, the first 4 columns were fixed, then it was considered a map corresponding to the 5th pair of perfusion values in Tab.4.1. This map is supposed unknown, and it was used as a target field for the reconstruction:

$$\mathbf{T_{target}} = \begin{bmatrix} T_{Sobol,t}(\mathbf{r_1}) \\ \vdots \\ T_{Sobol,t}(\mathbf{r_M}) \end{bmatrix} \tag{4.2}$$

where M is the number of points in the region of interest. This target field is then used to create the $\boldsymbol{\beta}$ matrix:

$$\boldsymbol{\beta} = pinv([S_{var}]|_{1...N_{cath}}, 10^{-3})\mathbf{T}_{target}|_{1...N_{cath}} \tag{4.3}$$

where the $[S_{var}]|_{1...N_{cath}}$ is the matrix whose elements correspond to the coordinates of the catheter:

$$[S_{var}]|_{1...N_{cath}} = S_{cath} \tag{4.4}$$

Now it is possible to obtain the reconstructed temperature map as follows:

$$[\mathbf{T_{rec}}] = [S_{var}][\boldsymbol{\beta}] \tag{4.5}$$

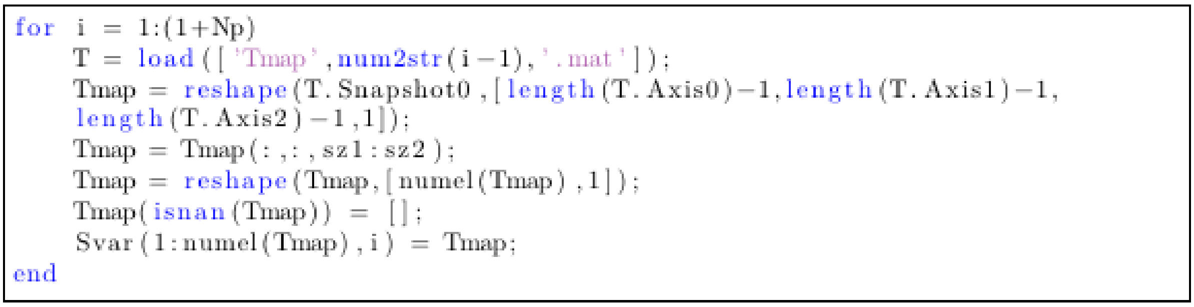
Finally, the target field was added to the matrix *Svar*, increasing the dimension of this matrix, and this procedure was repeated for another target field corresponding to another point in the multigrid.

The same procedure was performed for the *MultigridB*, considering the baseline map and the first 3 Sobol extractions in Tab.4.2 as extreme bounds.

To perform the *Svar* matrix in MATLAB it was necessary to reshape the temperature maps and remove all the NaN values, as shown in Fig.4.4.

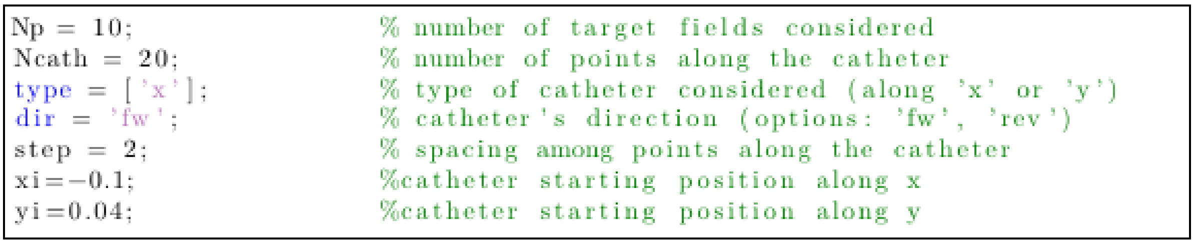
On the other hand, the *Scath* matrix (Appendix B) is a submatrix of *Svar* that presents only the temperature values along the catheter (4.4). Hence, the columns are the temperature maps as for the *Svar* matrix, while the number of rows depends on the number of points on the catheter to be considered.

In order to create this submatrix, it was necessary to: identify the starting point, *xi* and *yi* coordinates from which to start the catheter, direction of insertion of the catheter (along the *x* or *y* axis), type of insertion (forward or reverse), number of points on the



```
for  i  =  1:(1+Np)
     T  =  load(['Tmap',num2str(i-1),'.mat']);
     Tmap =  reshape(T.Snapshot0 ,[length(T.Axis0)-1,length(T.Axis1)-1,
     length(T.Axis2)-1 ,1]);
     Tmap =  Tmap(: ,: , sz1 : sz2 );
     Tmap =  reshape(Tmap,[numel(Tmap) ,1]);
     Tmap(isnan(Tmap))  =  [];
     Svar (1:numel(Tmap) , i )  =  Tmap;
end
```

Figure 4.4: Command line in order to create the *Svar* matrix



```
Np = 10;                   % number of target fields considered
Ncath = 20;                % number of points along the catheter
type = ['x'];              % type of catheter considered (along 'x' or 'y')
dir = 'fw';                % catheter's direction (options: 'fw', 'rev')
step = 2;                  % spacing among points along the catheter
xi=-0.1;                   %catheter starting position along x
yi=0.04;                   %catheter starting position along y
```

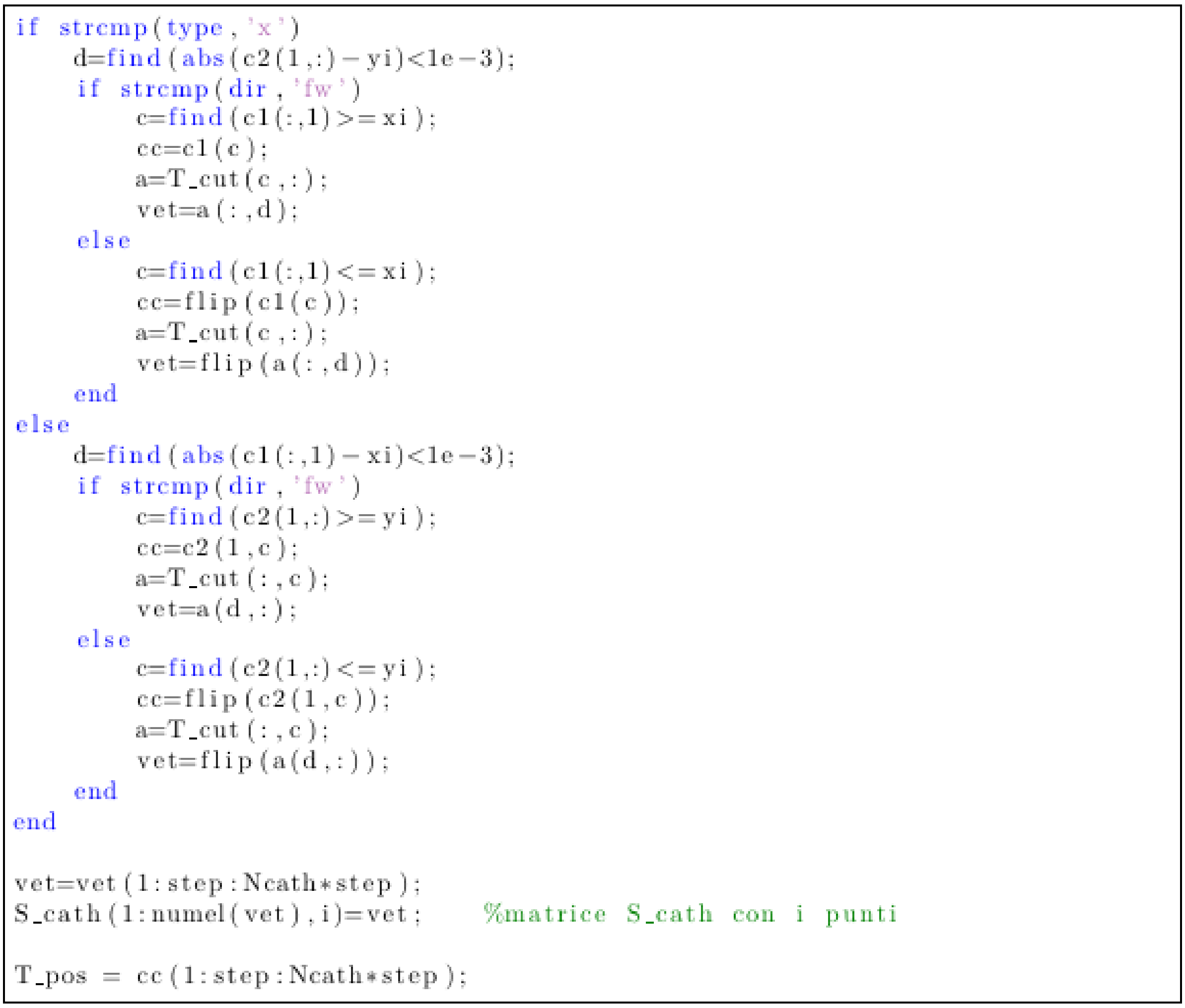
Figure 4.5: Inputs needed to configure catheter insertion.

catheter, i.e. temperature along the catheter and finally the step between them. All these are indications that are provided in input by the user (Fig.4.5).

Then, as shown in Fig.4.6, was saved an *Scath* matrix with all temperature values, along the chosen axis, starting from the initial coordinates up to the number of points defined previously and a vector *Tpos* that has all the temperature position (coordinates). The latter served to graph the catheter on the temperature map.

For this thesis purpose different directions of catheter insertion, different number of points and different spacing between points were considered, in order to monitor the temperature in a sufficient number of cases. In particular temperature maps were reconstructed for $N = 20$ points along the negative y axis, $N = 10$,20,40 points along the positive x axis and $N = 20$ points along the negative x axis. For all these cases were also produce some statistics plot to evaluate the temperature variation ΔT: the *Statistics - 1* plot shows the temperature variation in different target field, the *Statistics - 2* plot compares the baseline and the the 10th target field, the last plot *Statistics - 3* presents the trend line of each target field for maximum and median ΔT.

The results obtained (temperature maps and statistical plots) for all the cases just described are presented in the following sections 4.2.1 and 4.2.2, respectively for *Multigrid A* and *Multigrid B*. Furthermore, the xy plane will be considered as the reference plane for a clear visualization of the insertion of the catheter.



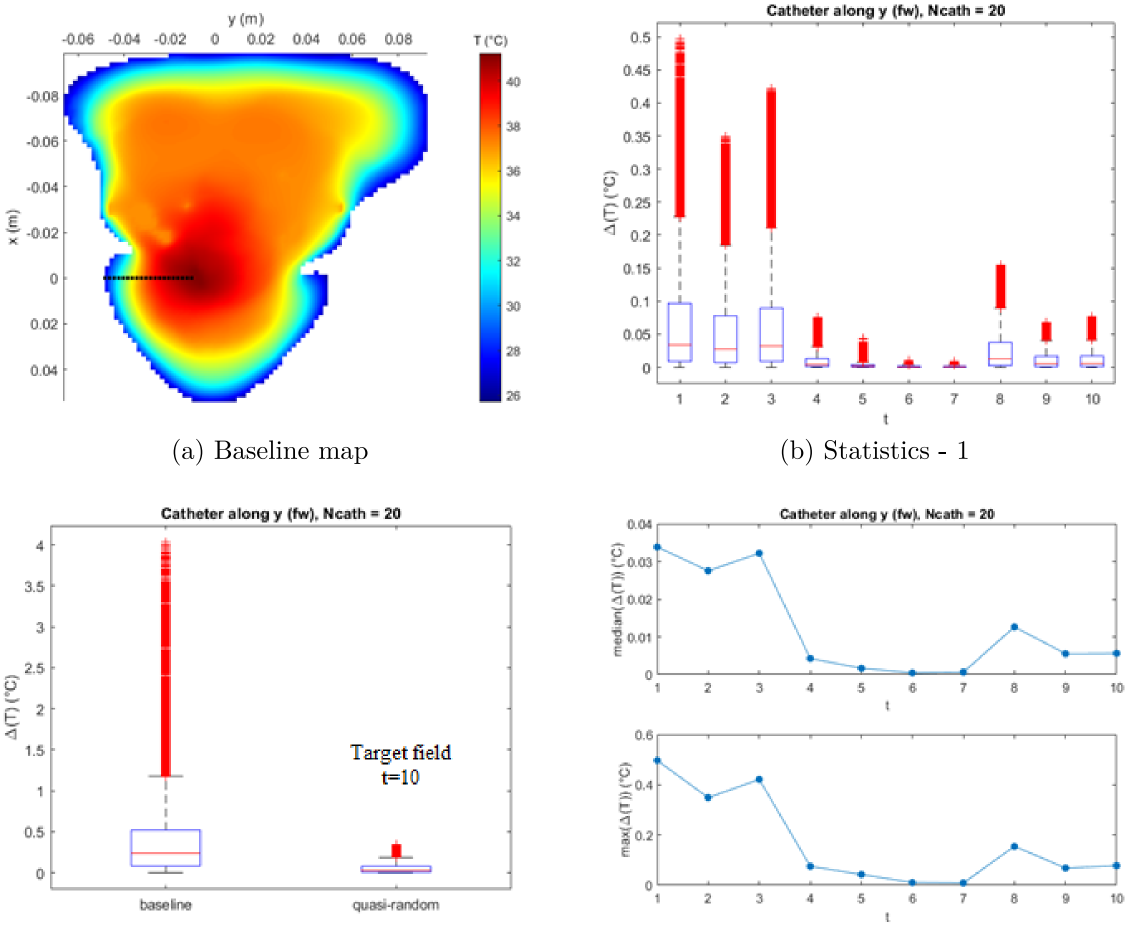
```
if strcmp(type,'x')
    d=find(abs(c2(1,:)-yi)<1e-3);
    if strcmp(dir,'fw')
        c=find(c1(:,1)>=xi);
        cc=c1(c);
        a=T_cut(c,:);
        vet=a(:,d);
    else
        c=find(c1(:,1)<=xi);
        cc=flip(c1(c));
        a=T_cut(c,:);
        vet=flip(a(:,d));
    end
else
    d=find(abs(c1(:,1)-xi)<1e-3);
    if strcmp(dir,'fw')
        c=find(c2(1,:)>=yi);
        cc=c2(1,c);
        a=T_cut(:,c);
        vet=a(d,:);
    else
        c=find(c2(1,:)<=yi);
        cc=flip(c2(1,c));
        a=T_cut(:,c);
        vet=flip(a(d,:));
    end
end

vet=vet(1:step:Ncath*step);
S_cath(1:numel(vet),i)=vet;    %matrice S_cath con i punti

T_pos = cc(1:step:Ncath*step);
```

Figure 4.6: Command lines for the creation of the *Scath* matrix

4.2.1 Multigrid A

Case 1:

- initial coordinates: $x_i = 0$m, $y_i = -0.049$m;
- direction: along y axis;
- type: forward;
- $N = 20$;
- step: 2mm.



(a) Baseline map

(b) Statistics - 1

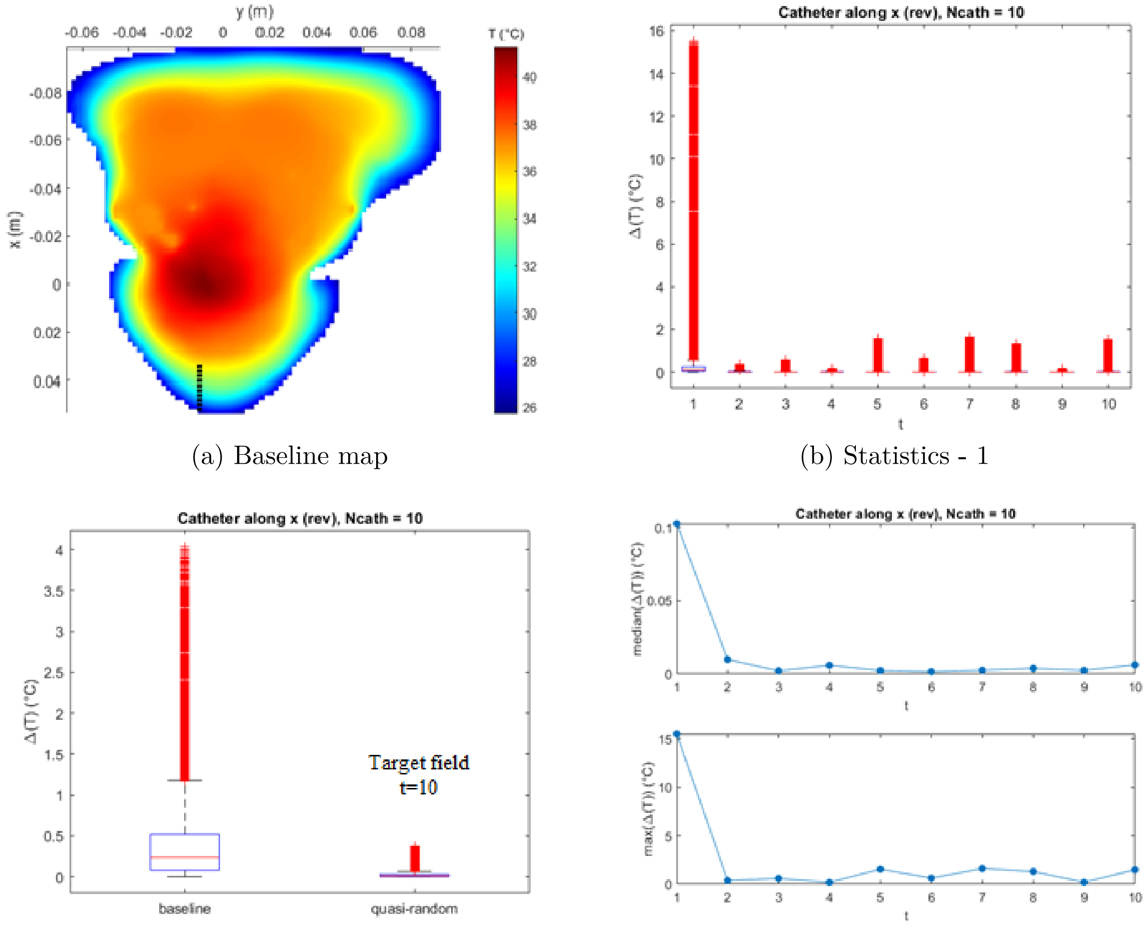
(c) Statistics - 2

(d) Statistics - 3

Figure 4.7: Multigrid A - Case 1, insertion of the catheter in the y_{fw} direction for $N = 20$.

Case 2:

- initial coordinates: $x_i = 0.053$m, $y_i = -0.01$m;
- direction: along x axis;
- type: forward;
- $N = 10$;
- step: 2mm.



(a) Baseline map

(b) Statistics - 1

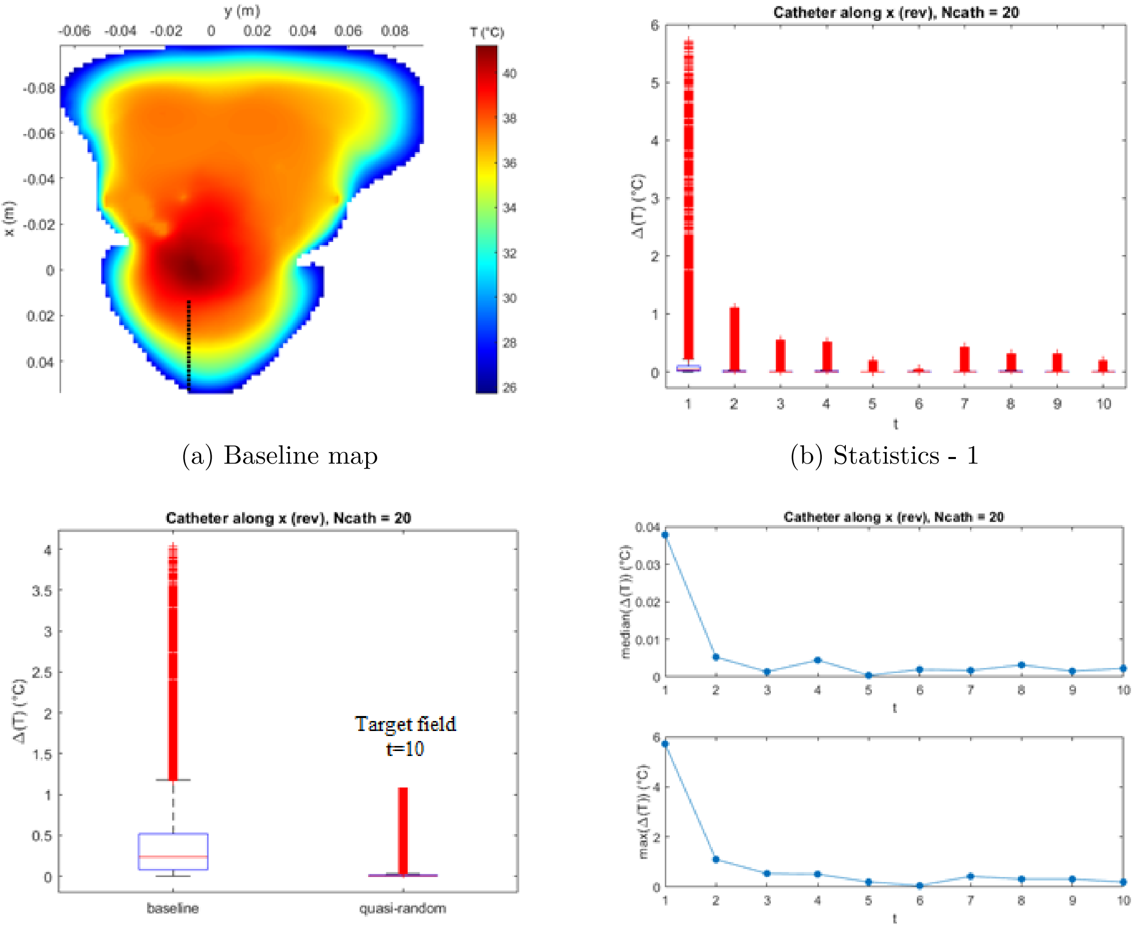
(c) Statistics - 2

(d) Statistics - 3

Figure 4.8: Multigrid A - Case 2, insertion of the catheter in the x_{fw} direction for $N = 10$.

Case 3:

- initial coordinates: $x_i = 0.053$m, $y_i = -0.01$m;
- direction: along x axis;
- type: forward;
- $N = 20$;
- step: 2mm.



(a) Baseline map

(b) Statistics - 1

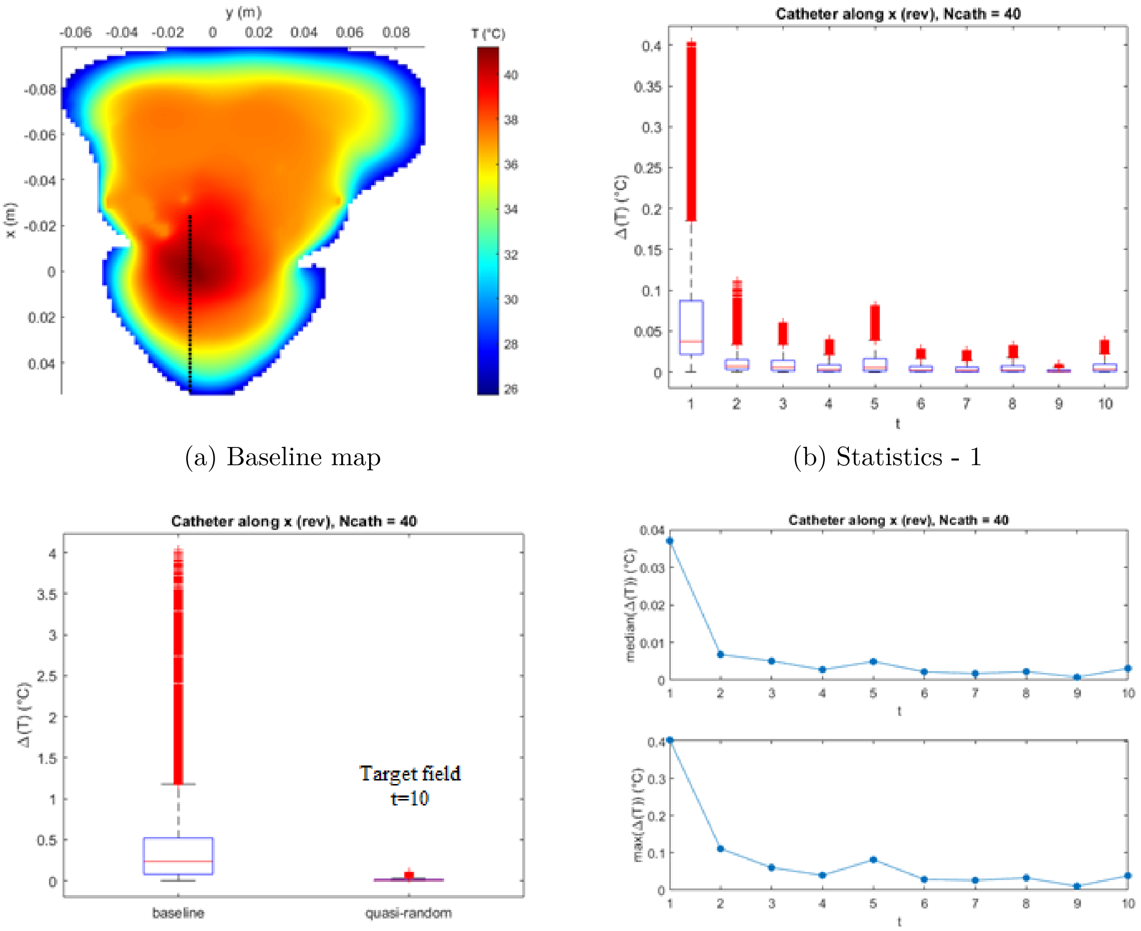
(c) Statistics - 2

(d) Statistics - 3

Figure 4.9: Multigrid A - Case 3, insertion of the catheter in the x_{fw} direction for $N = 20$.

Case 4:

- initial coordinates: $x_i = 0.053$m, $y_i = -0.01$m;
- direction: along x axis;
- type: forward;
- $N = 40$;
- step: 2mm.



(a) Baseline map

(b) Statistics - 1

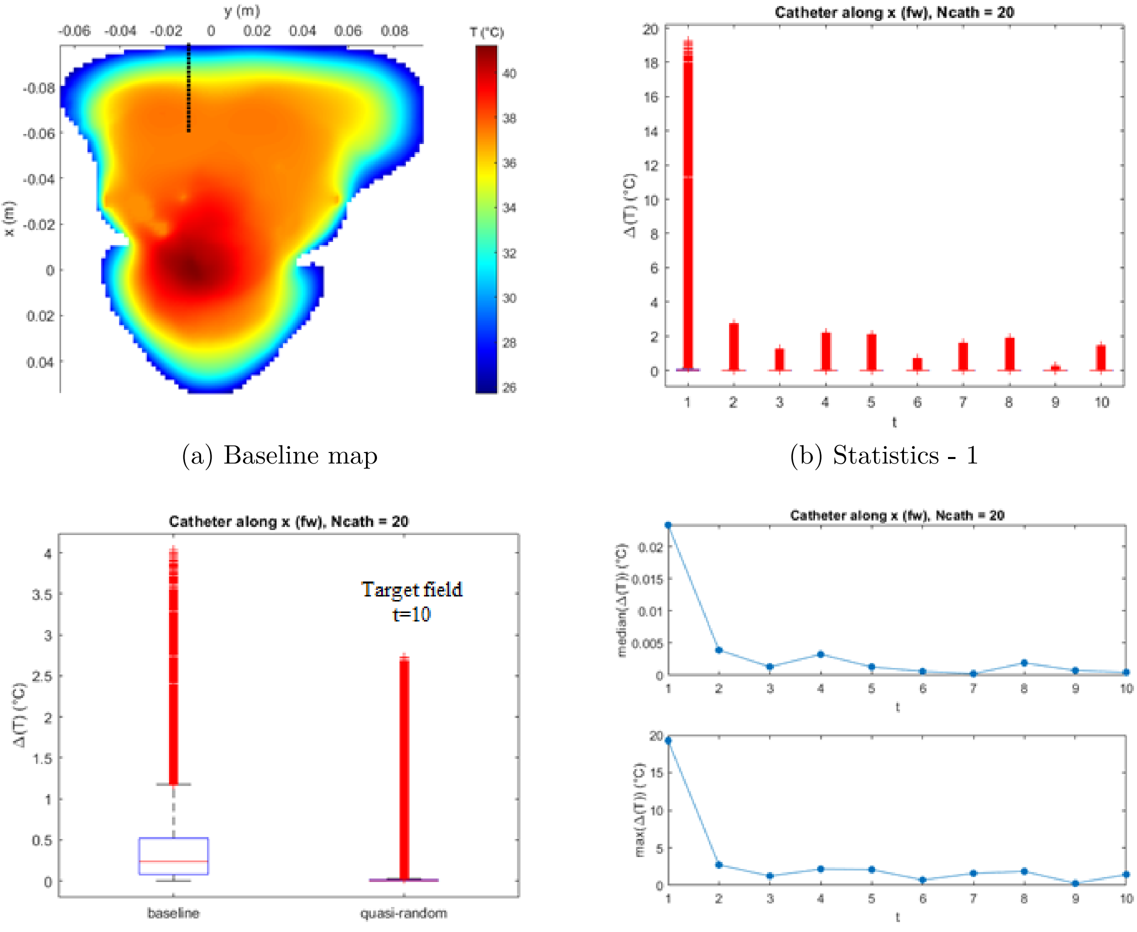
(c) Statistics - 2

(d) Statistics - 3

Figure 4.10: Multigrid A - Case 4, insertion of the catheter in the x_{fw} direction for $N = 40$.

Case 5:

- initial coordinates: $x_i = -0.1$m, $y_i = -0.01$m;
- direction: along x axis;
- type: reverse;
- $N = 20$;
- step: 2mm.



(a) Baseline map

(b) Statistics - 1

(c) Statistics - 2

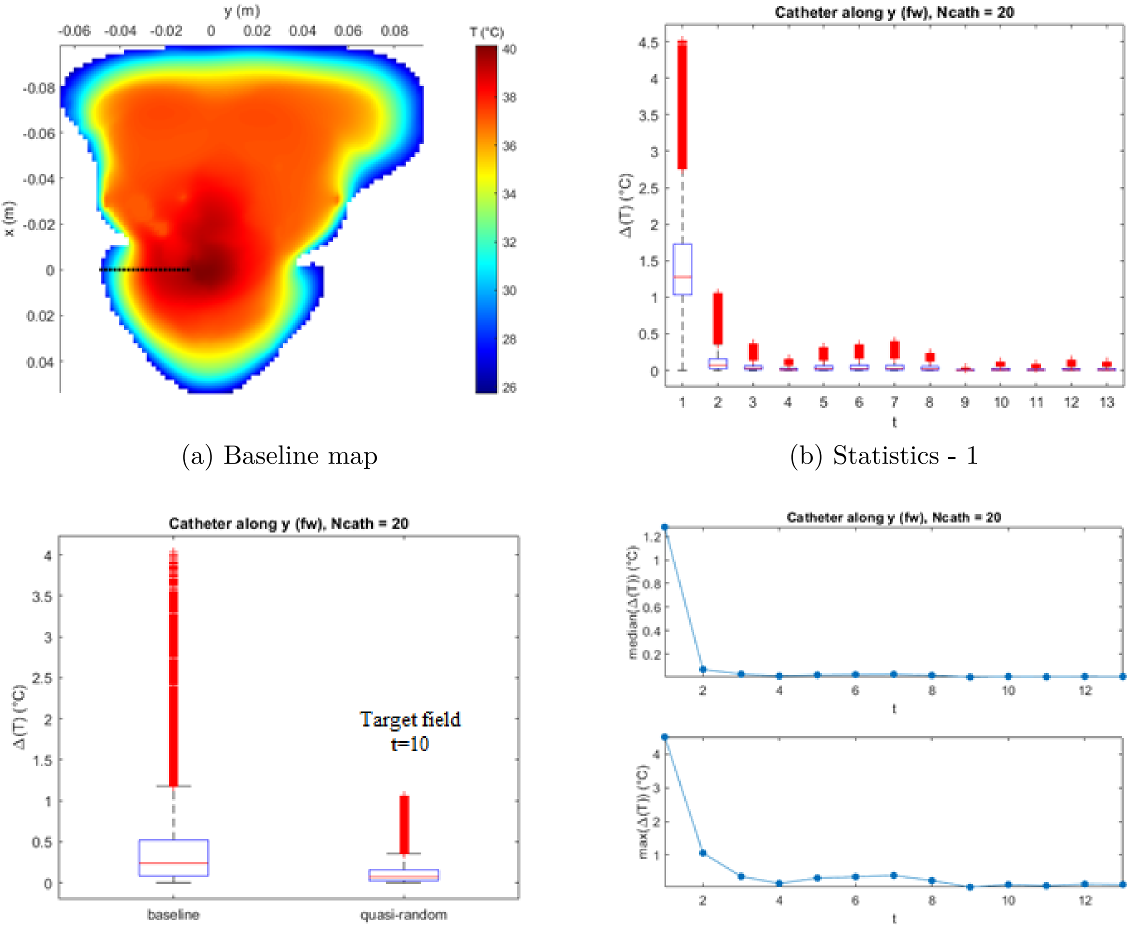
(d) Statistics - 3

Figure 4.11: Multigrid A - Case 5, insertion of the catheter in the x_{rev} direction for $N = 20$.

4.2.2 Multigrid B

Case 1:

- initial coordinates: $x_i = 0$m, $y_i = -0.049$m;
- direction: along y axis;
- type: forward;
- $N = 20$;
- step: 2mm.



(a) Baseline map

(b) Statistics - 1

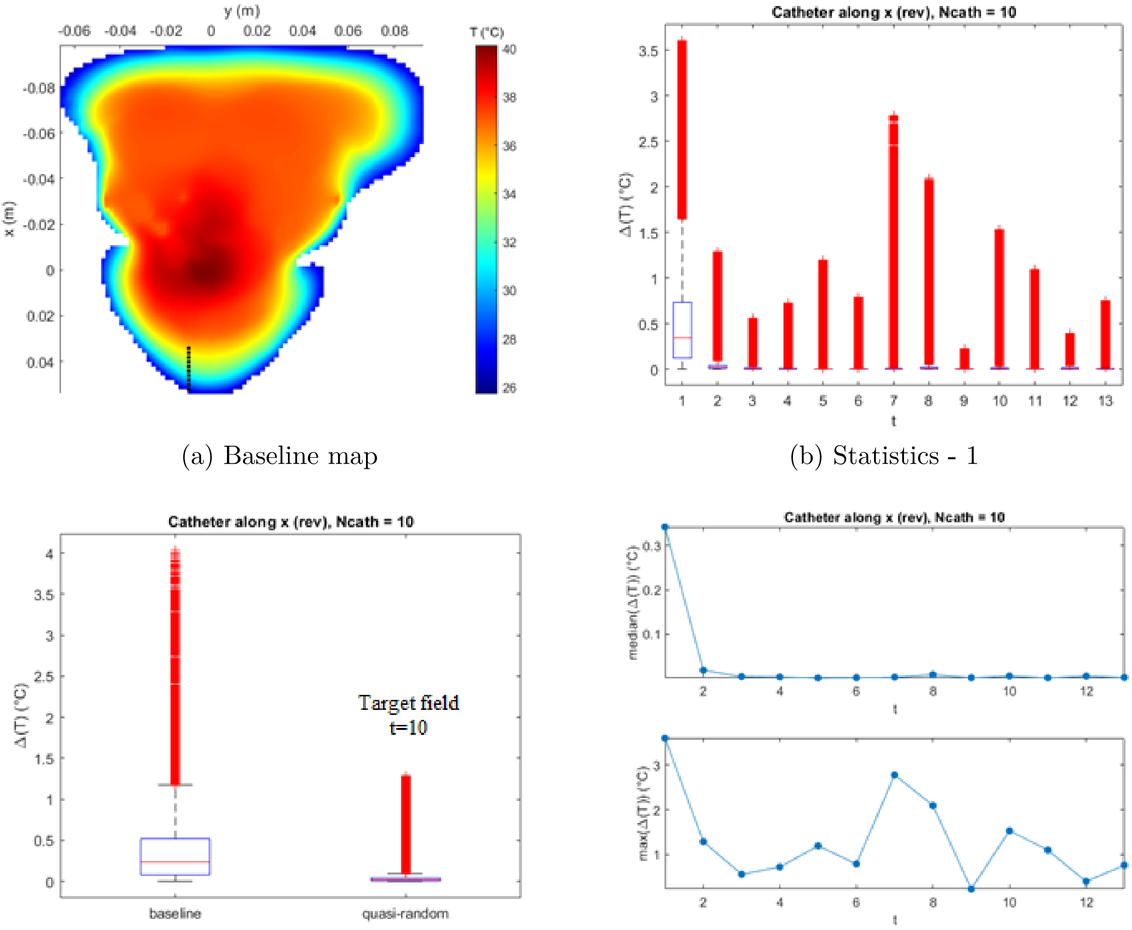
(c) Statistics - 2

(d) Statistics - 3

Figure 4.12: Multigrid B - Case 1, insertion of the catheter in the y_{fw} direction for $N = 20$.

Case 2:

- initial coordinates: $x_i = 0.053$m, $y_i = -0.01$m;
- direction: along x axis;
- type: forward;
- $N = 10$;
- step: 2mm.



(a) Baseline map

(b) Statistics - 1

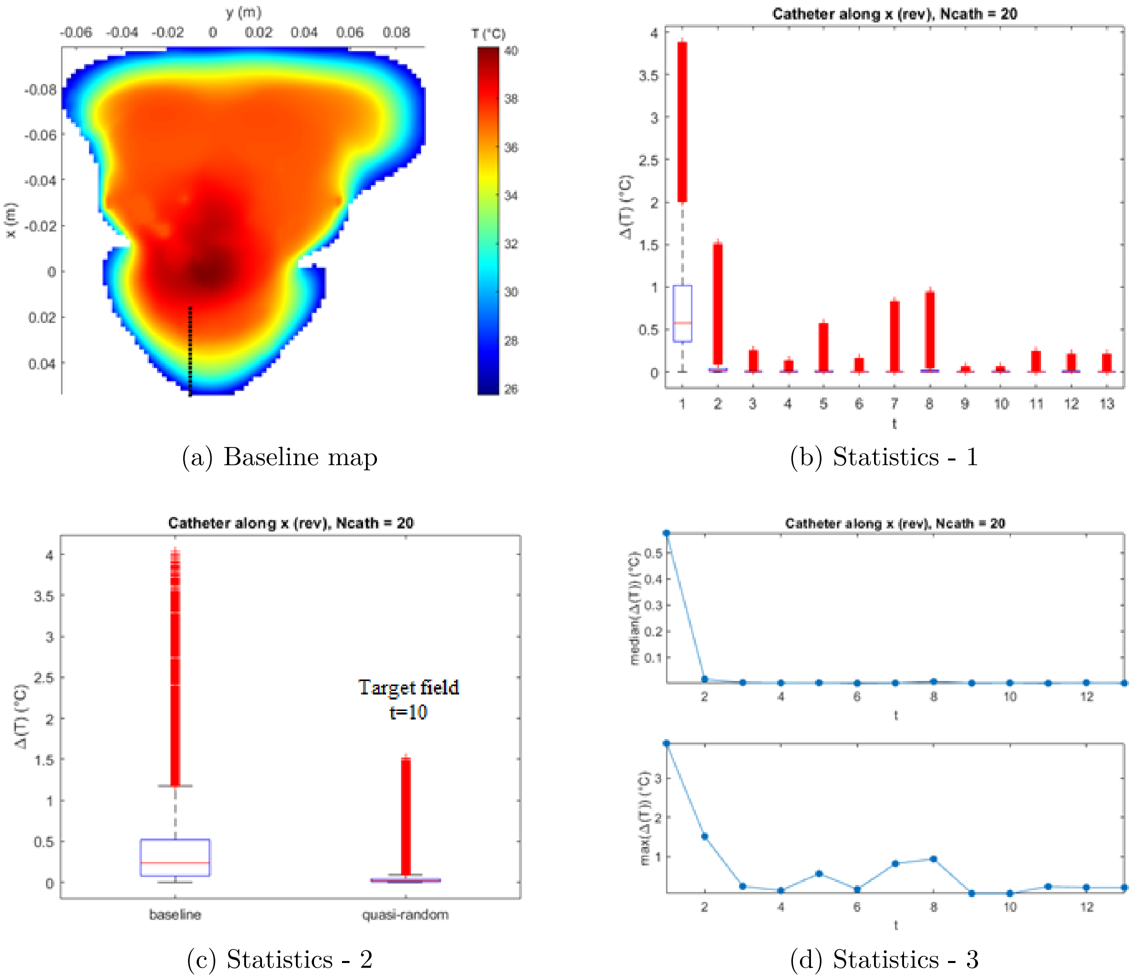
(c) Statistics - 2

(d) Statistics - 3

Figure 4.13: Multigrid B - Case 2, insertion of the catheter in the x_{fw} direction for $N = 10$.

Case 3:

- initial coordinates: $x_i = 0.053$m, $y_i = -0.01$m;
- direction: along x axis;
- type: forward;
- $N = 20$;
- step: 2mm.



(a) Baseline map

(b) Statistics - 1

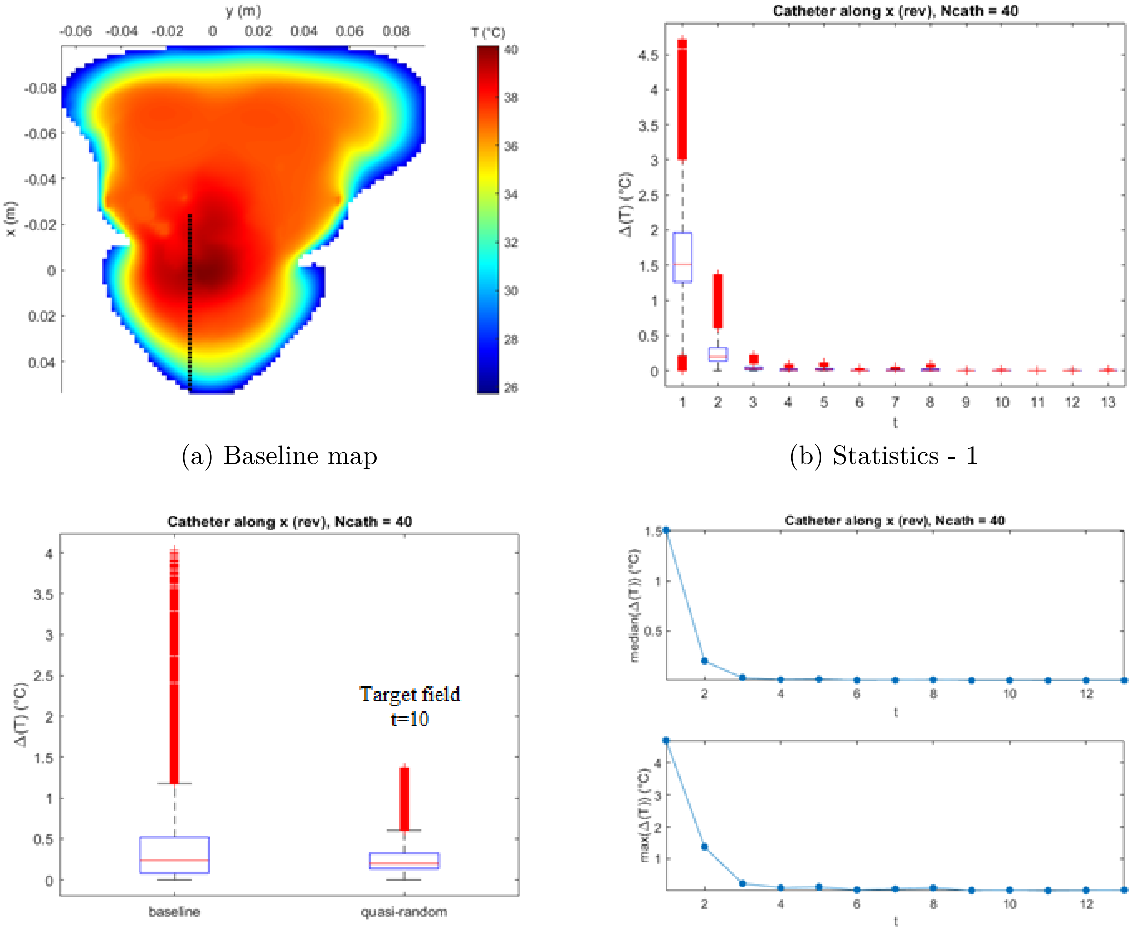
(c) Statistics - 2

(d) Statistics - 3

Figure 4.14: Multigrid B - Case 3, insertion of the catheter in the x_{fw} direction for $N = 20$.

Case 4:

- initial coordinates: $x_i = 0.053$m, $y_i = -0.01$m;
- direction: along x axis;
- type: forward;
- $N = 40$;
- step: 2mm.



(a) Baseline map

(b) Statistics - 1

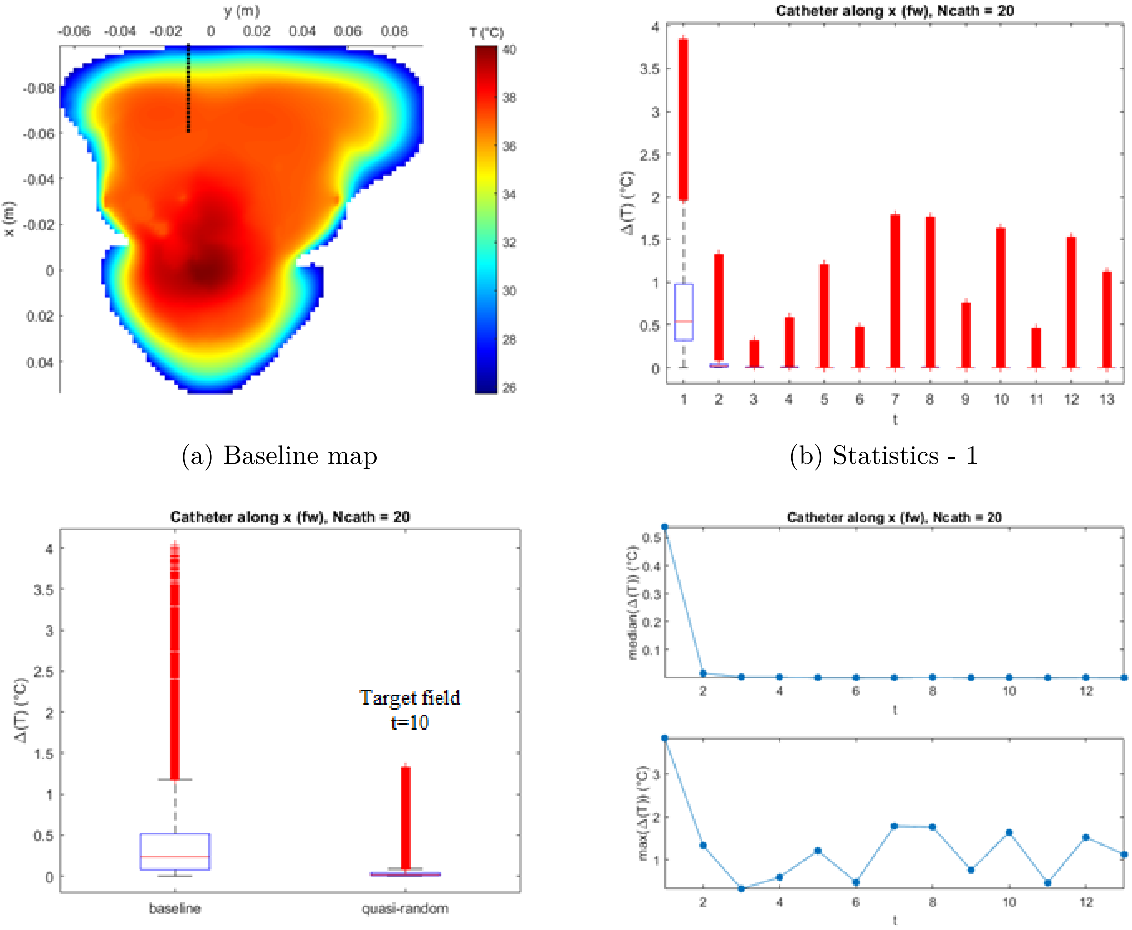
(c) Statistics - 2

(d) Statistics - 3

Figure 4.15: Multigrid B - Case 4, insertion of the catheter in the x_{fw} direction for $N = 40$.

Case 5:

- initial coordinates: $x_i = -0.1$m, $y_i = -0.01$m;
- direction: along x axis;
- type: reverse;
- $N = 20$;
- step: 2mm.



(a) Baseline map

(b) Statistics - 1

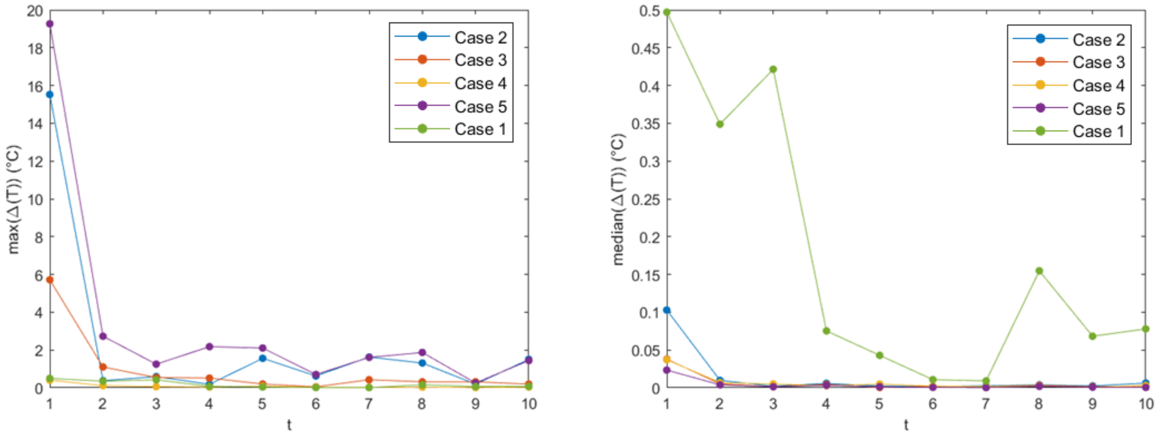
(c) Statistics - 2

(d) Statistics - 3

Figure 4.16: Multigrid B - Case 5, insertion of the catheter in the x_{rev} direction for $N = 20$.

4.2.3 Comparisons and Checks

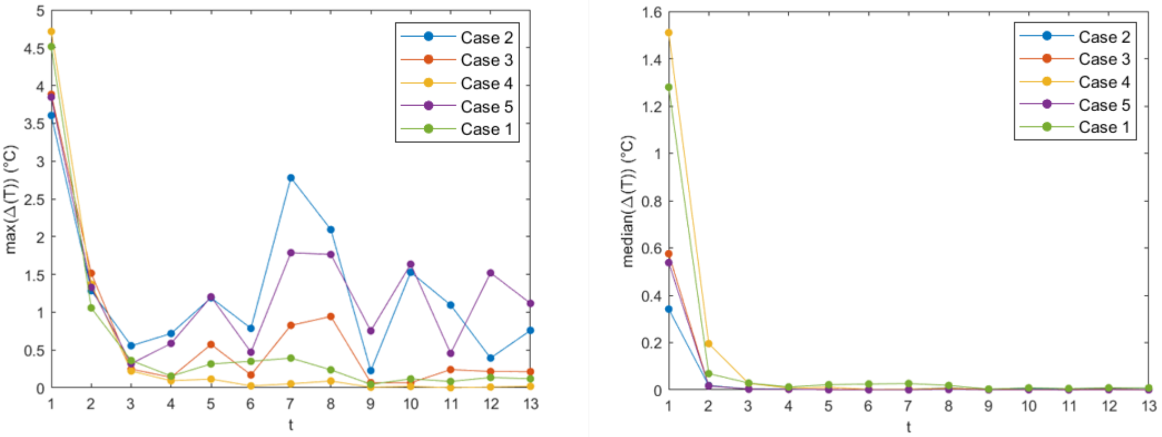
In both cases the statistical results for the maximum and median temperature variation were compared (Fig.4.17 and Fig.4.18). As can be seen, there is a decreasing trend of all the curves due to the greater number of points considered along the catheter.



(a) Max

(b) Median

Figure 4.17: Comparison Case 1, Case 2, Case 3, Case 4, Case 5 for the *Multigrid A*.

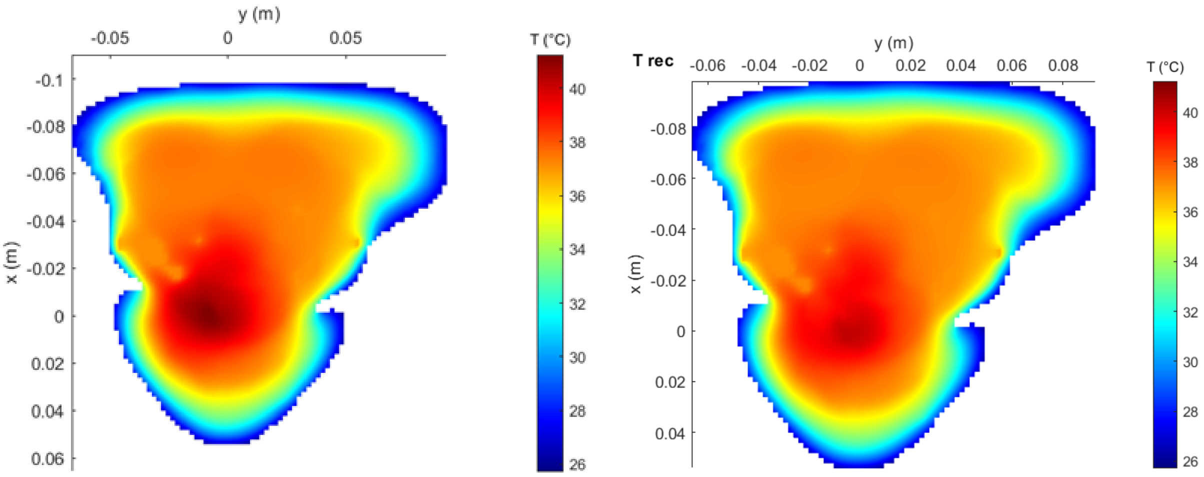


(a) Max

(b) Median

Figure 4.18: Comparison of Case 1, Case 2, Case 3, Case 4, Case 5 for the *Multigrid B*.

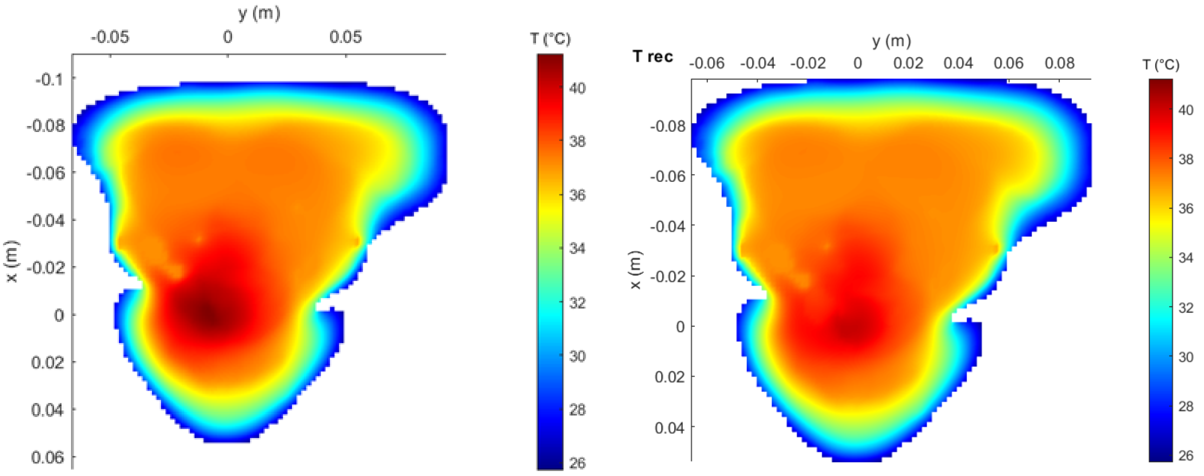
Finally, the temperature maps with respect to the baseline case (average values) and the temperature maps reconstructed 4.5, for the *Multigrid A* (Fig.4.19) and *Multigrid B* (Fig.4.20), of the 10th target field were compared.



(a) T_10

(b) T_rec

Figure 4.19: Comparison between the initial temperature map and the reconstructed map for the *Multigrid A*.



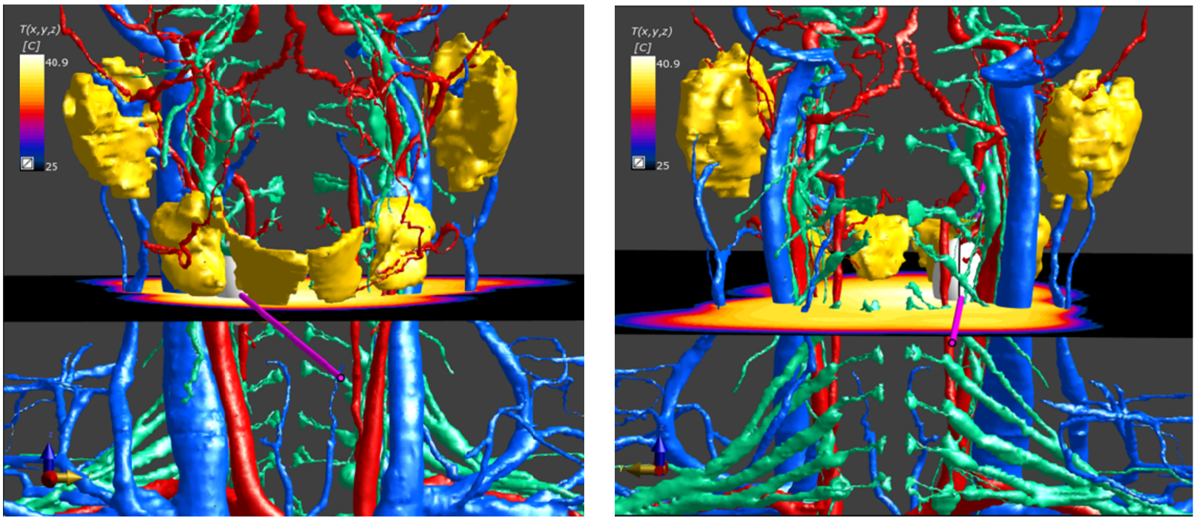
(a) T_10

(b) T_rec

Figure 4.20: Comparison between the initial temperature map and the reconstructed map for the *Multigrid B*.

The reconstruction implemented gives good results only if a sufficient number of maps in the *Svar* matrix are used (ie, at least 6). In fact, it can be seen from the graphs that only from the third target field onwards the performance is good regardless of the position of the catheter. The success of the reconstruction can also be seen from the comparison of the thermal maps: in both cases the temperature map of the reconstructed target field matches quite well the *Tb* map.

A final check was performed to verify that the catheter insertion position is not damaging vital tissues or organs, such as blood vessels, nerves and lymph nodes. This check was carry out by simulating the insertion of the catheter in Sim4Life: a 1 mm diameter cylinder, as displayed in Fig.4.21, was designed and positioned in correspondence with the coordinates and the chosen axis. It can be seen that the insertion of the catheter in the examined position is adequate and safe for the analysed subject (Yoon-sun).



(a) Front

(b) Back

Figure 4.21: Simulation of catheter insertion. The catheter is the pink cylinder and the tumor is the white element.

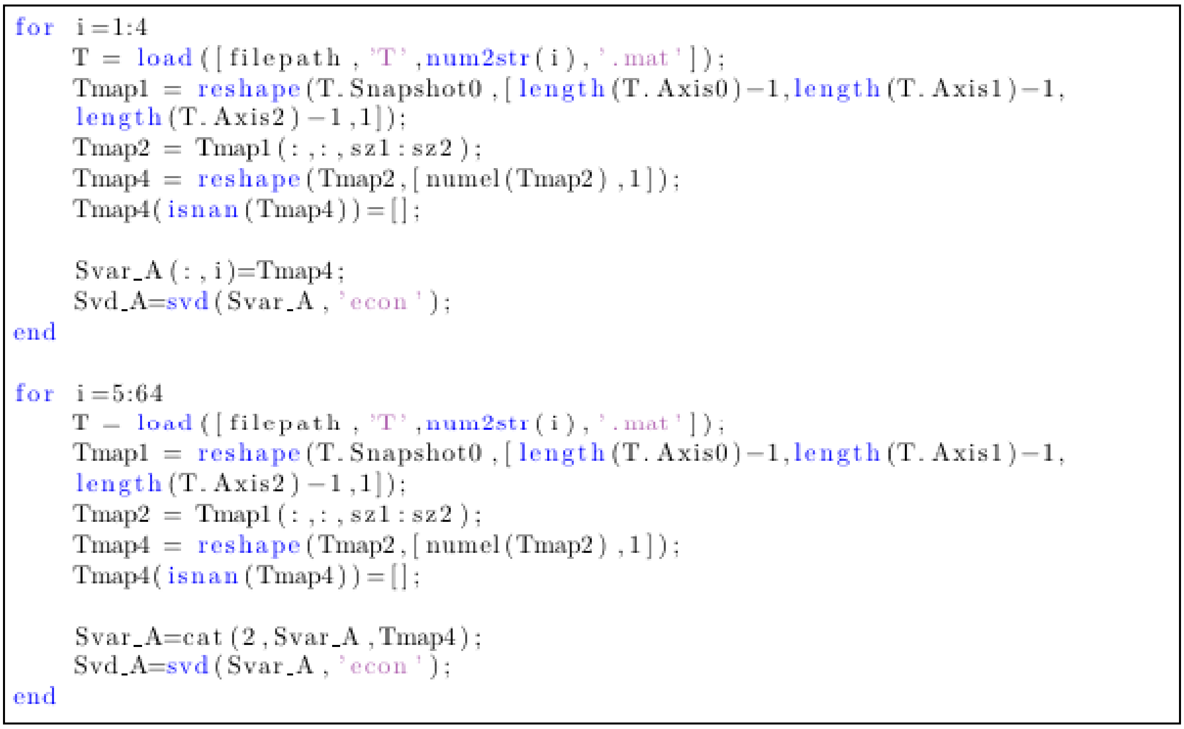
4.3 SVD analysis

"Singular Value Decomposition (SVD) is a widely used technique to decompose a matrix into several component matrices, exposing many of the useful and interesting properties of the original matrix. Using SVD, we can determine the rank of the matrix, quantify the sensitivity of a linear system to numerical error, or obtain an optimal lower-rank approximation to the matrix." [23]

In this thesis work, the SVD analysis aims evaluating the information contained in the *Svar* matrix, in order to assess how many columns are needed to obtain maximum information.

To achieve this goal, starting from the previous analysis, a multigrid of type A was built with 4 fixed values (corresponding to the extreme bounds of the perfusion values for muscle and tumor) and the extraction of 100 Sobol points.

Then, using the Python script reported in Fig.4.22, these pair of values were replaced in the setting of each tissues (*Muscle* and *Tumor*) and then the Sim4Life simulation was run. Again the temperature maps extracted were employed to create a $Svar_A$ matrix with 104 columns in MATLAB. As shown in Fig.4.22 the first 4 columns are fixed, while the other 60 columns are added once at time. The SVD analysis is performed each time a new column is added and the lower singular value is reported as a new green point in the plot of Fig. 4.23.



```
for i=1:4
    T = load([filepath,'T',num2str(i),'.mat']);
    Tmap1 = reshape(T.Snapshot0,[length(T.Axis0)-1,length(T.Axis1)-1,
    length(T.Axis2)-1,1]);
    Tmap2 = Tmap1(:,:,sz1:sz2);
    Tmap4 = reshape(Tmap2,[numel(Tmap2),1]);
    Tmap4(isnan(Tmap4))=[];

    Svar_A(:,i)=Tmap4;
    Svd_A=svd(Svar_A,'econ');
end

for i=5:64
    T = load([filepath,'T',num2str(i),'.mat']);
    Tmap1 = reshape(T.Snapshot0,[length(T.Axis0)-1,length(T.Axis1)-1,
    length(T.Axis2)-1,1]);
    Tmap2 = Tmap1(:,:,sz1:sz2);
    Tmap4 = reshape(Tmap2,[numel(Tmap2),1]);
    Tmap4(isnan(Tmap4))=[];

    Svar_A=cat(2,Svar_A,Tmap4);
    Svd_A=svd(Svar_A,'econ');
end
```

Figure 4.22: Command lines to built a new $Svar_A$ matrix.

In Fig.4.23, the result of this analysis shows that the curve reaches a plateau around the 50th column. From here on, any other extra column does not provide any additional information.

Then the same procedure was adopted for the *Multigrid B*, in which the 4 initial values are not considered, so the $Svar_B$ matrix is made up of only 60 quasi-random values (Fig.4.24). However, as for the *Multigrid A*, also the result of this analysis shows that the curve reaches a plateau around the 50th column.

In Fig.4.25 is displayed the comparison between the eigenvalues of the SVD_A and SVD_B matrix, taking 64 columns in both cases. As can be seen there is the same information content, so adding more columns does not lead to any improvement.

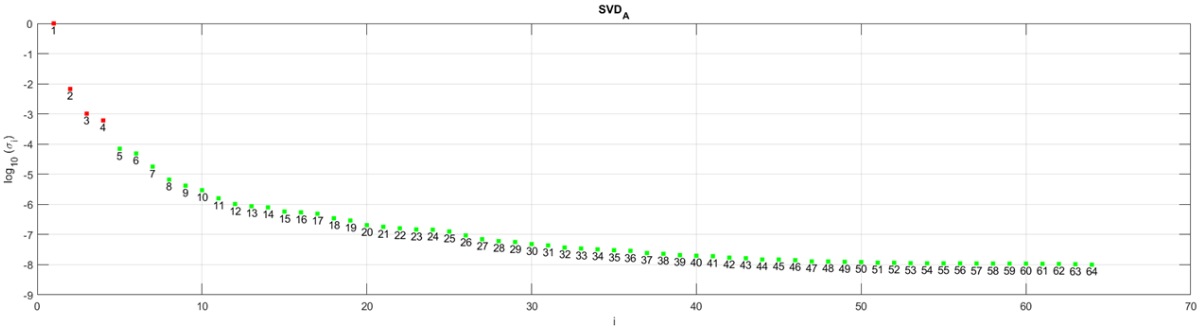


Figure 4.23: SVD of the *Multigrid A*.

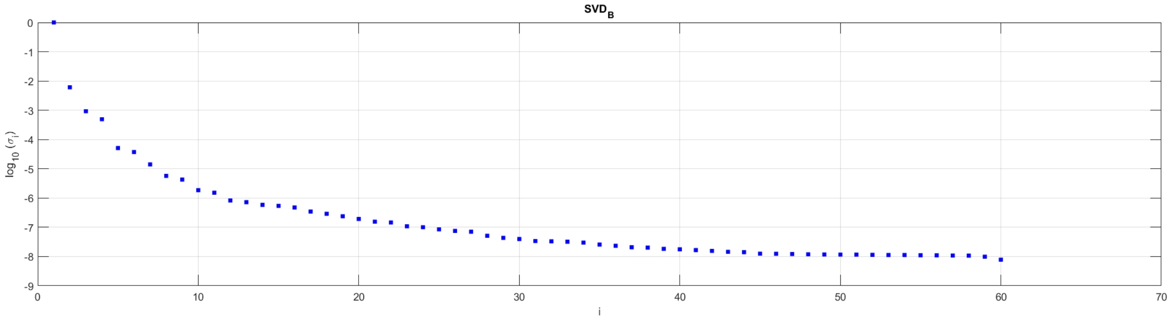


Figure 4.24: SVD of the *Multigrid B*.

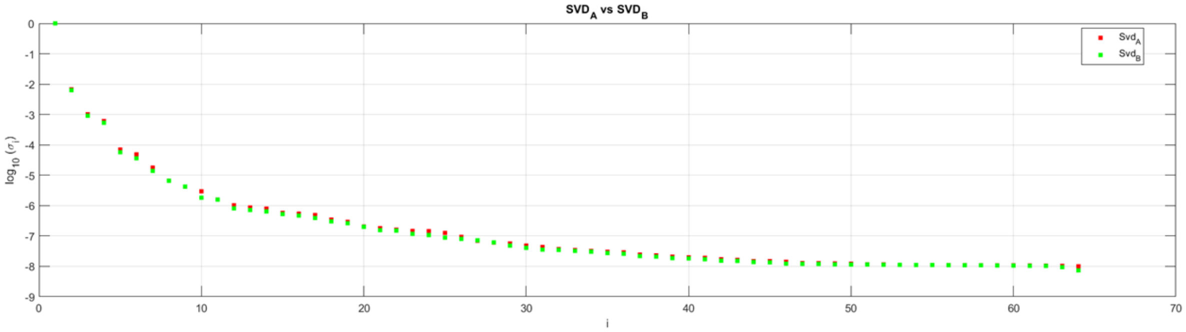


Figure 4.25: Comparison between the SVD_A and SVD_B.

Chapter 5

Conclusions

In this thesis was investigated the tissue properties that most influence the thermoregulatory response of H&N tissues and the performance of the hyperthermia treatments, as well as establish the accuracy of temperature simulations through minimally invasive temperature monitoring. Moreover, to address the scope of this work numerical simulation is been implemented allowing to perform analysis without the ethical and economical difficulties that apply to experimental methods. The final aim was to present a method to improve the temperature reconstruction in the region of interest using few known temperature values (corresponding, in the clinical practice, to the temperature measurements along the direction of insertion of a single catheter).

To achieve this goal the simulation software Sim4Life was employed, that combines a realistic and high-definition human phantom with a powerful physics solver, in order to reach and analyse the biological real-world phenomena. Then, thanks to the use of the Python application programming interface (API) it was possible to automate the data replacement process within the simulation, reducing the number of operator-dependent actions and saving times.

To analyse the influence of the tissues parameters on the temperature maps, only 4 types of tissues (*Muscle*, *Fat + SAT*, *Skin* and *Tumor*) have been considered. This choice was made by reasoning on the fact that the considered tissues are the dominant in the treated region.

To implement the temperature map reconstruction in a first simple case, we only considered the variation of the muscle and tumor perfusion. Hence, we created a basis for the reconstruction made of different (random) choices of these parameters. The first grid was created using the points at the extreme bounds of the parameters ranges plus other combinations extracted using a quasi-random Sobol sequence. The other grid was created only using the quasi-random points. The temperature reconstruction implemented using the two multigrids and different positions of the catheters points, as well as an SVD analysis performed on the two basis, show how both the two multigrids can be successfully used for the envisioned reconstruction.

The reconstruction of more reliable temperature and the monitoring of the latter during treatments can be considered a highly useful tool that will improve the outcomes of thermal treatments, enhancing the accuracy of predicting and monitoring pre-clinical and clinical procedures and treatments, as well as for controlling the amount of damage being imparted to tissue during the procedure. Furthermore, the efficacy of hyperthermia treatment planning tools in therapy management itself can be strengthened by feedback in the form of accurately measured tissue temperatures.

Future Developments

In this work, the temperature reconstruction has been implemented only considering two tissue parameters, i.e., the perfusion of muscle and tumor. In future works, it would be interesting to include other parameters of several kinds of tissue present in the H&N region. This will lead to the creation of a N-dimensional multigrid of parameters (being N the total number of parameters considered) and to a reconstruction of the temperature map from few known points that will almost completely mitigate (in principle) the error introduced in the model by the parameters' uncertainty. This will increase the required computational time, but it will be limited to a pre-treatment stage. Also, the analyses carried out are stationary, while they should be transient to be more realistic.

Moreover, it would be interesting to validate the numerical results obtained with experimental analyses both on fictitious models as well as on patients' data. Another future development could be keeping track of temperature changes over time, hence perform real-time monitoring, which accounts for valuable information to the clinician performing the procedure.

Another future development could be to use the numerical analysis presented as a basis for studying other body regions (i.e., breast, abdomen) and other populations, characterized by different human phantoms.

The final goal for the future is to improve the temperature monitoring during hyperthermia treatments in the clinical practice, in a minimally invasive way, i.e., using the implemented method and few measurements points. This will require the construction of a patient-specific library of temperature maps in a pre-treatment stage, and be used in real time during treatment together with the few measurement points.

MATLAB scripts

Fields Extraction

```matlab
% import the electric fields from Sim4Life and build proper
% structures for the healthy and tumor regions (E_health, E_tum)
% with the electric field components (.X,.Y,.Z).
folder = 'Fields_extract';     %files folder
filepath = [pwd,'\'];
N = 8;                          % number of antennas in the array

E_tum.X = [];
E_tum.Y = [];
E_tum.Z = [];
E_health.X = [];
E_health.Y = [];
E_health.Z = [];

for i = 1:N
        E = load([filepath,folder,'\E',num2str(i),'_H.mat']);
        E_health.X = [E_health.X,E.Snapshot0(:,1)];
        E_health.Y = [E_health.Y,E.Snapshot0(:,2)];
        E_health.Z = [E_health.Z,E.Snapshot0(:,3)];
        E = load([filepath,folder,'\E',num2str(i),'_T.mat']);
        E_tum.X = [E_tum.X,E.Snapshot0(:,1)];
        E_tum.Y = [E_tum.Y,E.Snapshot0(:,2)];
        E_tum.Z = [E_tum.Z,E.Snapshot0(:,3)];
end

E_health.X(isnan(E_health.X(:,1)),:) = [];
E_health.Y(isnan(E_health.Y(:,1)),:) = [];
E_health.Z(isnan(E_health.Z(:,1)),:) = [];
E_tum.X(isnan(E_tum.X(:,1)),:) = [];
E_tum.Y(isnan(E_tum.Y(:,1)),:) = [];
E_tum.Z(isnan(E_tum.Z(:,1)),:) = [];

save('E_health','-struct','E_health');
save('E_tum','-struct','E_tum');
```

Thermal Parameters Variation

```
% Plots for the estimation of the temperature maps variation as
% a consequence of the parameters variation.
% T_var is a matrix where the columns report the difference
% delta_T=|T_base-Tn| for the nth parameters variation. The row index
% runs on the total number of points in the reginon of interest.
filename = 'Parameters_v1_new.xlsx';
filepath = [pwd,'\',filename];

% Type of parameters: [k,w]
param = 2;
[num,txt,raw]=xlsread(filepath,param);
tissue_name=txt(2:5,1);

% Type of tissues: [Muscle,Fat,Skin,Tumor]
tissue = 1;

Range=range_value(filepath,param,tissue);

%%PLOT
sz1=50;
sz2=111;
filepath='C:\Users\...\Twm\';

T = load([filepath,'T',num2str(6),'.mat']);
Tmap1 = reshape(T.Snapshot0,[length(T.Axis0)-1,length(T.Axis1)-1,
length(T.Axis2)-1,1]);
Tmap2 = Tmap1(:,:,sz1:sz2);
Tmap4 = reshape(Tmap2,[numel(Tmap2),1]);
Tmap4(isnan(Tmap4))=[];
Tb=Tmap4;

for i=1:length(Range)
    T = load([filepath,'T',num2str(i),'.mat']);
    Tmap1 = reshape(T.Snapshot0,[length(T.Axis0)-1,length(T.Axis1)-1,
    length(T.Axis2)-1,1]);
    Tmap2 = Tmap1(:,:,sz1:sz2);
    Tmap4 = reshape(Tmap2,[numel(Tmap2),1]);
    Tmap4(isnan(Tmap4))=[];

    Tall(:,i)=Tmap4;
    T_var(:,i)=abs(Tb-Tall(:,i));
end

figure()
boxplot(T_var)
set(gcf,'color','w')
xlabel(['k␣',char(tissue_name(tissue)),'␣variations'])
ylabel('\Delta{T}␣(C)')
```

```
omega_var=length(Range);
M = zeros(omega_var,3);
M_stat = zeros(omega_var,2);

for i = 1:omega_var
    M(i,1) = numel(find(T_var(:,i)>1));
    M(i,2) = numel(find(T_var(:,i)>2));
    M(i,3) = numel(find(T_var(:,i)>3));
     M_stat(i,1) = mean(T_var(:,i));
     M_stat(i,2) = max(T_var(:,i));
end

figure()
hBar = bar(M);
xticks(0:16)
xlabel(['k␣',char(tissue_name(tissue)),'␣variations'])
ylabel('N')
legend('\Delta{T}␣>␣1␣C','\Delta{T}␣>␣2␣C','\Delta{T}␣>␣3␣C',
'Location','best')
set(gcf,'color','w')

figure()
plot(1:omega_var,M_stat(:,2),'.-','MarkerSize',18)
xticks(1:16)
xlabel(['k␣',char(tissue_name(tissue)),'␣variations'])
ylabel('max(\Delta{T})␣(C)')
set(gcf,'color','w')
```

Multigrid A

```
% Multi-dimensional pseudo-random Sobol grid (type A): including both the
% red Cartesian bound points and the Sobol green points.
% The parameters are organized in an external Excel file.

function MGrid = MultiGrid_Sobol_A(filepath,param,tissue,S)

    %%% Grid of fixed bounds points (red points) %%%
    table = cell(1,numel(param));

    for i = 1:numel(param)
        table{i} = xlsread(filepath,param(i),'A1:D5');
        table{i} = table{i}(tissue,:);
    end

    ranges = cell(numel(tissue),numel(param));
    avg = cell(numel(tissue),numel(param));

    for i = 1:numel(tissue)
        for j = 1:numel(param)
            ranges{i,j} = [table{j}(i,2),table{j}(i,3)];
            avg{i,j}=[table{j}(i,1)];
        end
    end

    output = cell(1,numel(param)*numel(tissue));
    [output{:}] = ndgrid(ranges{:});
    avg_output = cell(1,numel(param)*numel(tissue));
    [avg_output{:}] = ndgrid(avg{:});

    results = [];
    avg_res=[];
    for i = 1:length(output)
        results = [results output{1,i}(:)];
        avg_res=[avg_res avg_output{1,i}(:)];
    end

    %%% Quasi random Sobol sequence (green points) %%%
    q = qrandstream('sobol',numel(param)*numel(tissue),'Skip',1);
    X = rand(q,S,numel(param)*numel(tissue));
    RF = reshape([ranges{:}],2,numel(param)*numel(tissue)).';

    for i = 1:length(RF)
        X(:,i) = RF(i,1)+range(RF(i,:)).*X(:,i);
    end

    %%% Final grid %%%
    MGrid = [avg_res;results; X];
end
```

Multigrid B

```
% Multi-dimensional pseudo-random Sobol grid (type B): including only
% the Sobol green points.
% The parameters are organized in an external Excel file.

function MGrid = MultiGrid_Sobol_B(filepath,param,tissue,S)

   table = cell(1,numel(param));

   for i = 1:numel(param)
       table{i} = xlsread(filepath,param(i),'A1:D5');
       table{i} = table{i}(tissue,:);
   end

   ranges = cell(numel(tissue),numel(param));

   for i = 1:numel(tissue)
       for j = 1:numel(param)
           ranges{i,j} = [table{j}(i,2),table{j}(i,3)];
       end
   end

   %%% Quasi random Sobol sequence (green points) %%%
   q = qrandstream('sobol',numel(param)*numel(tissue),'Skip',1);
   X = rand(q,S,numel(param)*numel(tissue));
   RF = reshape([ranges{:}],2,numel(param)*numel(tissue)).';

   for i = 1:length(RF)
       X(:,i) = RF(i,1)+range(RF(i,:)).*X(:,i);
   end

   %%% Final grid %%%
   MGrid = X;
end
```

S_var Matrix

```
%This function generates the basis matrix Svar with the four fixed
% columns and other additional Np columns computed varying the perfusion of
% muscle and tumor according to the pseudo-random Sobol sequence.

function Svar_matrix(Np,sz1,sz2)

    Svar = size(4e6,1+Np);

    for i = 1:(1+Np)
         T = load(['Tmap',num2str(i-1),'.mat']);
         Tmap = reshape(T.Snapshot0,[length(T.Axis0)-1,length(T.Axis1)-1,
         length(T.Axis2)-1,1]);
         Tmap = Tmap(:,:,sz1:sz2);
         Tmap = reshape(Tmap,[numel(Tmap),1]);
         Tmap(isnan(Tmap)) = [];
         Svar(1:numel(Tmap),i) = Tmap;
    end

    Svar = Svar(1:numel(Tmap),:);
    save('Svar.mat','Svar');
end
```

S_cath Matrix

```
function S_cath(Np,Ncath,dir,type,step,xi,yi)
    Scath = size(4e6,1+Np);
    for i = 1:(1+Np)
        T = load(['Tmap',num2str(i-1),'.mat']);
        T_map = reshape(T.Snapshot0,[length(T.Axis0)-1,length(T.Axis1)-1,
        length(T.Axis2)-1,1]);
        x = (T.Axis0(1:(length(T.Axis0)-1))+T.Axis0(2:length(T.Axis0)))./2;
        y = (T.Axis1(1:(length(T.Axis1)-1))+T.Axis1(2:length(T.Axis1)))./2;
        z = (T.Axis2(1:(length(T.Axis2)-1))+T.Axis2(2:length(T.Axis2)))./2;

        [X,Y,Z] = ndgrid(x,y,z);

        zt = 87;
        c1 = squeeze(X(:,:,zt));
        c2 = squeeze(Y(:,:,zt));
        T_cut = squeeze(T_map(:,:,zt));

        figure()
        s = surf(c1,c2,T_cut);
        set(s,'edgecolor','none');
        h = colorbar;
        title(h,'T␣(C)')
        colormap jet
        shading interp
        axis([min(x) max(x) min(y) max(y)])
        set(gcf,'color','w')
        xlabel('x␣(m)')
        ylabel('y␣(m)')
        axis equal
        grid off
        view(0,90)
        camroll(-90)

        if strcmp(type,'x')
            d=find(abs(c2(1,:)-yi)<1e-3);
            if strcmp(dir,'fw')
                c=find(c1(:,1)>=xi);
                cc=c1(c);
                a=T_cut(c,:);
                vet=a(:,d);
            else
                c=find(c1(:,1)<=xi);
                cc=flip(c1(c));
                a=T_cut(c,:);
                vet=flip(a(:,d));
            end
        else
            d=find(abs(c1(:,1)-xi)<1e-3);
            if strcmp(dir,'fw')
```

```
                c=find(c2(1,:)>=yi);
                cc=c2(1,c);
                a=T_cut(:,c);
                vet=a(d,:);
            else
                c=find(c2(1,:)<=yi);
                cc=flip(c2(1,c));
                a=T_cut(:,c);
                vet=flip(a(d,:));
            end
        end

        vet=vet(1:step:Ncath*step);
        S_cath(1:numel(vet),i)=vet;     %matrice S_cath con i punti

        T_pos = cc(1:step:Ncath*step);

        hold on
        if strcmp(type,'y')
            plot3(xi.*ones(numel(T_pos),1),T_pos,1e3.*ones(numel(T_pos),1),
            'k.','MarkerSize',8)
        else
            plot3(T_pos,yi.*ones(numel(T_pos),1),1e3.*ones(numel(T_pos),1),
            'k.','MarkerSize',8)
        end
    end
    save('Scath.mat','Scath')
end
```

SVD Analysis

```
filename = 'Parameters_v1_new.xlsx';
filepath = [pwd,'\',filename];

% Type of parameters: [epsr,sigma,k,w]
param = 4;
% Type of tissues: [Muscle,Fat,Skin,Tumor]
tissue = [1,4];

S = 100;                  %number of Sobol quasi-random points

MGrid = SVD_Sobol_A(filepath,param,tissue,S);
MGrid=MGrid(2:end,:);
save([pwd,'\MGrid.txt'],'MGrid','-ascii','-tabs');

filepath=('C:\Users\...\');
sz1=50;
sz2=111;

figure()
for i=1:4
    T = load([filepath,'T',num2str(i),'.mat']);
    Tmap1 = reshape(T.Snapshot0,[length(T.Axis0)-1,length(T.Axis1)-1,
    length(T.Axis2)-1,1]);
    Tmap2 = Tmap1(:,:,sz1:sz2);
    Tmap4 = reshape(Tmap2,[numel(Tmap2),1]);
    Tmap4(isnan(Tmap4))=[];

    Svar_A(:,i)=Tmap4;
    Svd_A=svd(Svar_A,'econ');
end

for i=5:64
    T = load([filepath,'T',num2str(i),'.mat']);
    Tmap1 = reshape(T.Snapshot0,[length(T.Axis0)-1,length(T.Axis1)-1,
    length(T.Axis2)-1,1]);
    Tmap2 = Tmap1(:,:,sz1:sz2);
    Tmap4 = reshape(Tmap2,[numel(Tmap2),1]);
    Tmap4(isnan(Tmap4))=[];

    Svar_A=cat(2,Svar_A,Tmap4);
    Svd_A=svd(Svar_A,'econ');
end
plot(1:64,log10(Svd_A(1:64)./max(Svd_A)),'.','MarkerSize',12,'color','r');
hold on
xlabel('i')
ylabel('log_{10}␣(\sigma_i)')
title('Quasi-random␣vs␣Random')
grid on
```

```
for i=5:64
    T = load([filepath,'T',num2str(i),'.mat']);
    Tmap1 = reshape(T.Snapshot0,[length(T.Axis0)-1,length(T.Axis1)-1,
    length(T.Axis2)-1,1]);
    Tmap2 = Tmap1(:,:,sz1:sz2);
    Tmap4 = reshape(Tmap2,[numel(Tmap2),1]);
    Tmap4(isnan(Tmap4))=[];

    Svar_60(:,i-4)=Tmap4;
    Svd_60=svd(Svar_60,'econ');
end
plot(1:60,log10(Svd_60(1:60)./max(Svd_60)),'.','MarkerSize',12,'color','b');
hold on

for i=5:64+4   %rimuovo i 4 punti cardinali e considero altri 4 punti random
    T = load([filepath,'T',num2str(i),'.mat']);
    Tmap1 = reshape(T.Snapshot0,[length(T.Axis0)-1,length(T.Axis1)-1,
    length(T.Axis2)-1,1]);
    Tmap2 = Tmap1(:,:,sz1:sz2);
    Tmap4 = reshape(Tmap2,[numel(Tmap2),1]);
    Tmap4(isnan(Tmap4))=[];

    Svar_B(:,i-4)=Tmap4;
    Svd_B=svd(Svar_B,'econ');
end
plot(1:64,log10(Svd_B(1:64)./max(Svd_B)),'.','MarkerSize',12,'color','g');

%Aggiungo 4 colonne alla volta alla matrice Svar_B e confronto con quelle
%precedenti.

for k=4:4:36
    hold on
    in=64+k;    %punti iniziali considerati: 64 valori senza considerare
    %i 4 cardinali quindi 68.

    for i=in:in+4
        T = load([filepath,'T',num2str(i),'.mat']);
        Tmap1 = reshape(T.Snapshot0,[length(T.Axis0)-1,length(T.Axis1)-1,
        length(T.Axis2)-1,1]);
        Tmap2 = Tmap1(:,:,sz1:sz2);
        Tmap4 = reshape(Tmap2,[numel(Tmap2),1]);
        Tmap4(isnan(Tmap4))=[];

        Svar_D=cat(2,Svar_B,Tmap4);
        Svd_D=svd(Svar_D,'econ');
     end
     txt = ['Svd_{B+',num2str(k),'}'];
     plot(1:20,log10(Svd_D(1:20)./max(Svd_D)),'LineWidth',1.5,'DisplayName',
     txt);
     hold on
end
```

Python scripts

Field Extraction

```
import numpy
import s4l_v1.analysis as analysis
import s4l_v1.document as document
import s4l_v1.model as model
import s4l_v1.units as units
from s4l_v1 import ReleaseVersion
from s4l_v1 import Unit
import itertools

def all_entities_within_group(entity_group):
        '''return a list of all model entities within a given group,
        including all subdirectories'''
    if isinstance(entity_group, model.EntityGroup):
        return list (itertools.chain.from_iterable(all_entities_within_group
        (e) for e in entity_group.Entities))
    else:
        return [entity_group]

vip_group = model.AllEntities()['Yoon-sun']
entities_H = all_entities_within_group(vip_group)
entities_T = model.AllEntities()['Tumor']

# Name of the directory where the files will be saved
dir = u"C:\\Users\\...\\"



# HEALTHY REGION standalone electric fields Ei_H
try:
    # Define the version to use for default values
    ReleaseVersion.set_active(ReleaseVersion.version6_2)
    # Creating the analysis pipeline
    # Adding a new EmMultiPortSimulationExtractor
    simulation = document.AllSimulations["EM_1"]
    em_multi_port_simulation_extractor =simulation.Results()
    #Create the postprocessing pipeline once
    output_port=em_multi_port_simulation_extractor.Outputs[0]
    # Adding a new EmPortSimulationExtractor
    em_port_simulation_extractor = analysis.extractors.
    EmPortSimulationExtractor(inputs=[output_port])
    em_port_simulation_extractor.UpdateAttributes()
    document.AllAlgorithms.Add(em_port_simulation_extractor)
    # Adding a new EmSensorExtractor (H)
    em_sensor_extractor = em_port_simulation_extractor["Overall␣Field"]
    em_sensor_extractor.FrequencySettings.ExtractedFrequency= u"All"
    em_sensor_extractor.SurfaceCurrent.SurfaceResolution=0.001, units.Meters
    document.AllAlgorithms.Add(em_sensor_extractor)
    # Adding a new MatlabExporter (H)
    inputs = [field_masking_filter.Outputs["EM␣E(x,y,z,f0)"]]
    matlab_exporter = analysis.exporters.MatlabExporter(inputs=inputs)
    matlab_exporter.UpdateAttributes()
    document.AllAlgorithms.Add(matlab_exporter)

    for i, output_port in enumerate(em_multi_port_simulation_extractor.Outputs):
      em_port_simulation_extractor.raw.SetInputConnection(0,output_port.raw)
      em_port_simulation_extractor.UpdateAttributes()
      inputs = [field_masking_filter.Outputs["EM␣E(x,y,z,f0)"]]
      matlab_exporter.FileName=(dir + "E{}_H.mat".format(i+1))

      print(matlab_exporter.FileName)
      matlab_exporter.UpdateAttributes()
      matlab_exporter.Update()

except Exception as exc:
    import traceback
    traceback.print_exc(exc)
    # Reset active version to default
    ReleaseVersion.reset()
    raise(exc)




# TUMOR REGION standalone electric fields Ei_T
try:
    # Define the version to use for default values
    ReleaseVersion.set_active(ReleaseVersion.version6_2)
    # Creating the analysis pipeline
    # Adding a new EmMultiPortSimulationExtractor
    simulation = document.AllSimulations["EM␣1"]
    em_multi_port_simulation_extractor = simulation.Results()
    #Create the postprocessing pipeline once
    output_port=em_multi_port_simulation_extractor.Outputs[0]
    # Adding a new EmPortSimulationExtractor
    em_port_simulation_extractor =analysis.extractors.
    EmPortSimulationExtractor(inputs=[output_port])
    em_port_simulation_extractor.UpdateAttributes()
    document.AllAlgorithms.Add(em_port_simulation_extractor)
    # Adding a new EmSensorExtractor (T)
    em_sensor_extractor =
    em_port_simulation_extractor["Overall␣Field"]
    em_sensor_extractor.FrequencySettings.ExtractedFrequency= u"All"
    em_sensor_extractor.SurfaceCurrent.SurfaceResolution=0.001, units.Meters
    document.AllAlgorithms.Add(em_sensor_extractor)
    # Adding a new FieldMaskingFilter (T)
    inputs = [em_sensor_extractor.Outputs["EM_E(x,y,z,f0)"]]
    field_masking_filter =analysis.core.FieldMaskingFilter(inputs=inputs)
    field_masking_filter.SetAllMaterials(False)
    field_masking_filter.SetEntities([entities_T])
    field_masking_filter.UpdateAttributes()
    document.AllAlgorithms.Add(field_masking_filter)
    # Adding a new MatlabExporter (T)
    inputs = [field_masking_filter.Outputs["EM_E(x,y,z,f0)"]]
    matlab_exporter =
    analysis.exporters.MatlabExporter(inputs=inputs)
    matlab_exporter.UpdateAttributes()
    document.AllAlgorithms.Add(matlab_exporter)

    for i, output_port in enumerate(em_multi_port_simulation_extractor.Outputs):
      em_port_simulation_extractor.raw.SetInputConnection(0,output_port.raw)
      em_port_simulation_extractor.UpdateAttributes()

      inputs = [field_masking_filter.Outputs["EM_E(x,y,z,f0)"]]
      matlab_exporter.FileName=(dir + "E{}_T.mat".format(i+1))

      print(matlab_exporter.FileName)
      matlab_exporter.UpdateAttributes()
      matlab_exporter.Update()

except Exception as exc:
    import traceback
    traceback.print_exc(exc)
    # Reset active version to default
    ReleaseVersion.reset()
    raise(exc)




# J e SAR extraction
try:
    # Define the version to use for default values
    ReleaseVersion.set_active(ReleaseVersion.version6_2)
    # Creating the analysis pipeline
    # Adding a new EmMultiPortSimulationExtractor
    simulation = document.AllSimulations["EM_1"]
    em_multi_port_simulation_extractor = simulation.Results()
    #Create the postprocessing pipeline once
    output_port=em_multi_port_simulation_extractor.Outputs[0]
```

```
# Adding a new EmPortSimulationExtractor
em_port_simulation_extractor =analysis.extractors.
EmPortSimulationExtractor(inputs=output_port)
em_port_simulation_extractor.UpdateAttributes()
document.AllAlgorithms.Add(em_port_simulation_extractor)
# Adding a new EmSensorExtractor (T)
em_sensor_extractor =em_port_simulation_extractor["Overall Field"]
em_sensor_extractor.FrequencySettings.ExtractedFrequency= u"All"
em_sensor_extractor.SurfaceCurrent.SurfaceResolution=0.001, units.Meters
document.AllAlgorithms.Add(em_sensor_extractor)
# Adding a new FieldMaskingFilter for SAR1 (H)
inputs = [em_sensor_extractor.Outputs["SAR(x,y,z,f0)"]]
field_masking_filter =analysis.core.FieldMaskingFilter(inputs=inputs)
field_masking_filter.SetAllMaterials(False)
field_masking_filter.SetEntities(entities_H)
field_masking_filter.UpdateAttributes()
document.AllAlgorithms.Add(field_masking_filter)
# Adding a new FieldMaskingFilter for SAR1 (T)
inputs = [em_sensor_extractor.Outputs["SAR(x,y,z,f0)"]]
field_masking_filter_2 =analysis.core.FieldMaskingFilter(inputs=inputs)
field_masking_filter_2.SetAllMaterials(False)
field_masking_filter_2.SetEntities([entities_T])
field_masking_filter_2.UpdateAttributes()
document.AllAlgorithms.Add(field_masking_filter_2)
# Adding a new FieldMaskingFilter for J1 (H)
inputs = [em_sensor_extractor.Outputs["J(x,y,z,f0)"]]
field_masking_filter_3 =analysis.core.FieldMaskingFilter(inputs=inputs)
field_masking_filter_3.SetAllMaterials(False)
field_masking_filter_3.SetEntities(entities_H)
field_masking_filter_3.UpdateAttributes()
document.AllAlgorithms.Add(field_masking_filter_3)
# Adding a new FieldMaskingFilter for J1 (T)
inputs = [em_sensor_extractor.Outputs["J(x,y,z,f0)"]]
field_masking_filter_4 =analysis.core.FieldMaskingFilter(inputs=inputs)
field_masking_filter_4.SetAllMaterials(False)
field_masking_filter_4.SetEntities([entities_T])
field_masking_filter_4.UpdateAttributes()
document.AllAlgorithms.Add(field_masking_filter_4)
# Adding a new MatlabExporter for SAR1 (H)
inputs = [field_masking_filter.Outputs["SAR(x,y,z,f0)"]]
matlab_exporter =analysis.exporters.MatlabExporter(inputs=inputs)
matlab_exporter.FileName = (dir + "SAR1_H.mat")
matlab_exporter.UpdateAttributes()
document.AllAlgorithms.Add(matlab_exporter)
matlab_exporter.Update()
# Adding a new MatlabExporter for SAR1 (T)
inputs=[field_masking_filter_2.Outputs["SAR(x,y,z,f0)"]]
matlab_exporter_2 =analysis.exporters.MatlabExporter(inputs=inputs)
matlab_exporter_2.FileName = (dir + "SAR1_T.mat")
matlab_exporter_2.UpdateAttributes()
document.AllAlgorithms.Add(matlab_exporter_2)
```

```
    matlab_exporter_2.Update()
    # Adding a new MatlabExporter for J1 (H)
    inputs = [field_masking_filter_3.Outputs["J(x,y,z,f0)"]]
    matlab_exporter_3 =analysis.exporters.MatlabExporter(inputs=inputs)
    matlab_exporter_3.FileName = (dir + "J1_H.mat")
    matlab_exporter_3.UpdateAttributes()
    document.AllAlgorithms.Add(matlab_exporter_3)
    matlab_exporter_3.Update()
    # Adding a new MatlabExporter for J1 (T)
    inputs = [field_masking_filter_4.Outputs["J(x,y,z,f0)"]]
    matlab_exporter_4 =analysis.exporters.MatlabExporter(inputs=inputs)
    matlab_exporter_4.FileName = (dir + "J1_T.mat")
    matlab_exporter_4.UpdateAttributes()
    document.AllAlgorithms.Add(matlab_exporter_4)
    matlab_exporter_4.Update()

except Exception as exc:
    import traceback
    traceback.print_exc(exc)
    # Reset active version to default
    ReleaseVersion.reset()
    raise(exc)
```

Thermal Parameter Variation

```
import numpy
import math
import s4l_v1.document as document
import s4l_v1.materials.database as database
import s4l_v1.model as model
import s4l_v1.simulation.thermal as thermal
import s4l_v1.analysis as analysis
import s4l_v1.units as units
from s4l_v1 import ReleaseVersion
from s4l_v1 import Unit
import itertools

def all_entities_within_group(entity_group):
         '''return a list of all model entities within a given group,
         including all subdirectories'''
    if isinstance(entity_group, model.EntityGroup):
         return list (itertools.chain.from_iterable(all_entities_within_group
         (e) for e in entity_group.Entities))
    else:
         return [entity_group]

vip_group = model.AllEntities()['Yoon-sun']
entities__all = all_entities_within_group(vip_group)
entities__air_internal=model.AllEntities()['Air_internal']
entities__tumor=model.AllEntities()['Tumor']
entities__all.remove(entities__air_internal)
entities__all.append(entities__tumor)

dir = u"C:\\Users\\...\\"

coord = numpy.loadtxt("C:/Users/.../Range.txt")
print (coord)
size=coord.size

for i in range (0, size):
    wm = coord[i]

    print(wm)

    # Define the version to use for default values
    ReleaseVersion.set_active(ReleaseVersion.version6_2)
    simulation = document.AllSimulations["Th_80W"]
    simulation.ClearResults()
    simulation.ResetVoxels()

    entity__muscle = model.AllEntities()["Muscle"]
    new_w_value = wm, Unit("ml\kg\K")
    # Change a value of an existing material
    sets = simulation.AllSettings
```

```
for idx,set in enumerate(sets):
    if set.Name == 'Muscle':
        set.HeatTransferRate.UsePerfusionUnits = True
        set.HeatTransferRate.PuConstantTerm = new_w_value
        #set.ThermalConductivity=new_w_value

# RUN
simulation.UpdateGrid()
simulation.CreateVoxels()
simulation.RunSimulation(wait=True)
# Create extractor for a given simulation output file
results = simulation.Results()
# Overall field sensor
overall_field_sensor = results[ 'Overall␣Field' ]

# EXPORT MATLAB
inputs = [overall_field_sensor.Outputs[”T(x,y,z)”]]
field_masking_filter =analysis.core.FieldMaskingFilter(inputs=inputs)
field_masking_filter.SetAllMaterials(False)
field_masking_filter.SetEntities(entities__all)
field_masking_filter.UpdateAttributes()
document.AllAlgorithms.Add(field_masking_filter)
inputs = [field_masking_filter.Outputs[”T(x,y,z)”]]
matlab_exporter =analysis.exporters.MatlabExporter(inputs=inputs)
matlab_exporter.Name=(”Matlab␣{}”.format(i+1))
matlab_exporter.FileName = (dir + ”T{}.mat”.format(i+1))
matlab_exporter.UpdateAttributes()
document.AllAlgorithms.Add(matlab_exporter)
matlab_exporter.Update(overwrite=True)
matlab_exporter.Update(overwrite=True)
```

Dielectric Parameter Variation

```
import numpy
import math
import s4l_v1.document as document
import s4l_v1.materials.database as database
import s4l_v1.model as model
import s4l_v1.simulation.thermal as thermal
import s4l_v1.analysis as analysis
import s4l_v1.units as units
from s4l_v1 import ReleaseVersion
from s4l_v1 import Unit
import itertools

def all_entities_within_group(entity_group):
        '''return a list of all model entities within a given group,
        including all subdirectories'''
    if isinstance(entity_group, model.EntityGroup):
        return list(itertools.chain.from_iterable(all_entities_within_group
        (e) for e in entity_group.Entities))
    else:
        return [entity_group]

vip_group = model.AllEntities()['Yoon-sun']
entities__all = all_entities_within_group(vip_group)
entities__air_internal=model.AllEntities()['Air_internal']
entities__tumor=model.AllEntities()['Tumor']
entities__all.remove(entities__air_internal)
entities__all.append(entities__tumor)

dir = u"C:\\Users\\...\\"

coord = numpy.loadtxt("C:/Users/.../Range.txt")
print (coord)
size=coord.size

for i in range (0, size):
    wm = coord[i]

    print(wm)

    # Define the version to use for default values
    ReleaseVersion.set_active(ReleaseVersion.version6_2)
    simulation = document.AllSimulations["EM_1 - new"]
    simulation.ClearResults()
    simulation.ResetVoxels()

    entity__skin = model.AllEntities()["Skin"]
    new_w_value = wm
    # Change value of an existing material
    sets = simulation.AllSettings
```

```
for idx,set in enumerate(sets):
    if set.Name == 'Skin':
        set.RelativePermittivity=new_w_value
        # set.ElectricConductivity=new_w_value

# RUN
simulation.UpdateGrid()
simulation.CreateVoxels()
simulation.RunSimulation(wait=True)
simulation = document.AllSimulations["EM␣1␣-␣new"]
simulation_extractor = simulation.Results()

# Adding a new EmSensorExtractor
em_sensor_extractor=simulation_extractor["Overall␣Field"]
em_sensor_extractor.FrequencySettings.ExtractedFrequency= u"All"
em_sensor_extractor.SurfaceCurrent.SurfaceResolution=0.001, units.Meters
document.AllAlgorithms.Add(em_sensor_extractor)
# Adding a new FieldSnapshotFilter
inputs = [em_sensor_extractor.Outputs["El.␣Loss␣Density(x,y,z,f0)"]]
field_snapshot_filter =analysis.field.FieldSnapshotFilter(inputs=inputs)
field_snapshot_filter.UpdateAttributes()
document.AllAlgorithms.Add(field_snapshot_filter)
# Adding a new DataCacheExporter
inputs = [field_snapshot_filter.Outputs["El.␣Loss␣Density(x,y,z,f0)"]]
data_cache_exporter =analysis.exporters.DataCacheExporter(inputs=inputs)
data_cache_exporter.Name ="Data␣Cache␣Exporter␣-␣User␣Defined␣Source"
data_cache_exporter.FileName =(dir + "Source{}.cache".format(i+1))
data_cache_exporter.UpdateAttributes()
document.AllAlgorithms.Add(data_cache_exporter)
data_cache_exporter.Update(overwrite=True)
data_cache_exporter.Update(overwrite=True)

# Thermal Simulation
ReleaseVersion.set_active(ReleaseVersion.version6_2)
simulation1=document.AllSimulations["Th␣80W"]
simulation1.ClearResults()
simulation1.ResetVoxels()
simulation1.Remove(stationary_user_defined_heat_source,components)
# Adding a new StationaryUserDefinedHeatSource
stationary_user_defined_heat_source =
thermal.StationaryUserDefinedHeatSource()
components = []
stationary_user_defined_heat_source.UserDefinedFileName=
(dir + "Source{}.cache".format(i+1))
simulation1.Add(stationary_user_defined_heat_source,components)

# RUN
simulation1.UpdateGrid()
simulation1.CreateVoxels()
simulation1.RunSimulation(wait=True)
```

```
# Create extractor for a given simulation output file
results1 = simulation1.Results()
# overall field sensor
overall_field_sensor1 = results1[ 'Overall␣Field' ]

# EXPORT MATLAB
inputs = [overall_field_sensor1.Outputs["T(x,y,z)"]]
field_masking_filter =analysis.core.FieldMaskingFilter(inputs=inputs)
field_masking_filter.SetAllMaterials(False)
field_masking_filter.SetEntities(entities__all)
field_masking_filter.UpdateAttributes()
document.AllAlgorithms.Add(field_masking_filter)
inputs = [field_masking_filter.Outputs["T(x,y,z)"]]
matlab_exporter =analysis.exporters.MatlabExporter(inputs=inputs)
matlab_exporter.Name=("Matlab␣{}".format(i+1))
matlab_exporter.FileName = (dir + "T{}.mat".format(i+1))
matlab_exporter.UpdateAttributes()
document.AllAlgorithms.Add(matlab_exporter)
matlab_exporter.Update(overwrite=True)
matlab_exporter.Update(overwrite=True)
```

Perfusion Variation

```
import numpy
import math
import s4l_v1.document as document
import s4l_v1.materials.database as database
import s4l_v1.model as model
import s4l_v1.simulation.thermal as thermal
import s4l_v1.analysis as analysis
import s4l_v1.units as units
from s4l_v1 import ReleaseVersion
from s4l_v1 import Unit
import itertools

def all_entities_within_group(entity_group):
    '''return a list of all model entities within a given group,
    including all subdirectories'''
    if isinstance(entity_group, model.EntityGroup):
        return list(itertools.chain.from_iterable(all_entities_within_group
        (e) for e in entity_group.Entities))
    else:
        return [entity_group]

vip_group = model.AllEntities()['Yoon-sun']
entities__all = all_entities_within_group(vip_group)
entities__air_internal=model.AllEntities()['Air_internal']
entities__tumor=model.AllEntities()['Tumor']
entities__all.remove(entities__air_internal)
entities__all.append(entities__tumor)

dir = u”C:\\Users\\...\\”

coord = numpy.loadtxt(”C:/Users/.../MGrid.txt”)
print (coord)
num_rows, num_columns = coord.shape

for i in range (0, num_rows):
    wm = coord[i,0]
    wt = coord[i,1]
    print(wm, wt)

    # Define the version to use for default values
    ReleaseVersion.set_active(ReleaseVersion.version6_2)
    simulation = document.AllSimulations[”Th_80W”]
    simulation.ClearResults()
    simulation.ResetVoxels()

    entity__muscle = model.AllEntities()[”Muscle”]
    muscle_new_w_value = wm, Unit(”ml/min/kg”)
    entity__tumor = model.AllEntities()[”Tumor”]
    tumor_new_w_value = wt, Unit(”ml/min/kg”)
```

```
# Change the pair of value of an existing material
sets = simulation.AllSettings
for idx,set in enumerate(sets):
    if set.Name == 'Muscle':
        set.HeatTransferRate.UsePerfusionUnits = True
        set.HeatTransferRate.PuConstantTerm = muscle_new_w_value
    if set.Name == 'Tumor':
        set.HeatTransferRate.UsePerfusionUnits = True
        set.HeatTransferRate.PuConstantTerm = tumor_new_w_value

# RUN
simulation.UpdateGrid()
simulation.CreateVoxels()
simulation.RunSimulation(wait=True)

# Create extractor for a given simulation output file
results = simulation.Results()
# overall field sensor
overall_field_sensor = results[ 'Overall␣Field' ]

# EXPORT MATLAB
inputs = [overall_field_sensor.Outputs["T(x,y,z)"]]
field_masking_filter = analysis.core.FieldMaskingFilter(inputs=inputs)
field_masking_filter.SetAllMaterials(False)
field_masking_filter.SetEntities(entities__all)
field_masking_filter.UpdateAttributes()
document.AllAlgorithms.Add(field_masking_filter)
inputs = [field_masking_filter.Outputs["T(x,y,z)"]]
matlab_exporter = analysis.exporters.MatlabExporter(inputs=inputs)
matlab_exporter.Name=("Matlab␣{}".format(i))
matlab_exporter.FileName = (dir + "T{}.mat".format(i))
matlab_exporter.UpdateAttributes()
document.AllAlgorithms.Add(matlab_exporter)
matlab_exporter.Update(overwrite=True)
```

Bibliography

[1] J. van der Zee, *Heating the patient: a promising approach?*, Annals of Oncology, 13:1173-1184, 2002.

[2] M. H. Falk and R. D. Issels, *Hyperthermia in oncology*, International Journal of Hyperthermia, Vol. 17, N. 1, 1-18, 2001.

[3] U. Lucia, *Elementi di ingegneria delle terapie termiche*, Torino, Edizioni C.L.U.T., 2017.

[4] N.R. Datta, S. Gómez Ordóñez, U.S. Gaipl, M.M. Paulides, H. Crezee, J. Gellermann, D. Marder, E. Puric, S. Bodis, *Local hyperthermia combined with radiotherapy and-/or chemotherapy: Recent advances and promises for the future*, Cancer Treatment Reviews, 41:742–753, 2015.

[5] M. R. Horsman, J. Overgaard, *Hyperthermia: a Potent Enhancer of Radiotherapy*, Clinical Oncology, 19:418-426, 2007.

[6] Rossella Gaffoglio, Marco Righero, Giorgio Giordanengo, Marcello Zucchi, and Giuseppe Vecchi, Fellow, IEEE, *Fast Optimization of Temperature Focusing in Hyperthermia Treatment of Sub-Superficial Tumors*, IEEE Journal of Electromagnetics RF and Microwaves in Medicine and Biology, Vol. 5, N. 3, 2021.

[7] M.M. Paulides, H. Dobsicek Trefna, S. Curto, D.B. Rodrigues, *Recent technological advancements in radiofrequency and microwave-mediated hyperthermia for enhancing drug delivery*, Advanced Drug Delivery Reviews, 163–164 (2020) 3–18.

[8] Zef Rijnen, Jurriaan F. Bakker, Richard A.M. Canters, Paolo Togni, Gerda M. Verduijn, Peter C. Levendag, Gerard C. Van Rhoon & Margarethus M. Paulides, *Clinical integration of software tool VEDO for adaptive and quantitative application of phased array hyperthermia in the head and neck*, International Journal of Hyperthermia, 29(3):181–193, 2013.

[9] Margarethus M. Paulides, Gerda M. Verduijn and Netteke Van Holthe, *Status quo and directions in deep head and neck hyperthermia*, Radiation Oncology, 11:21, 2016.

[10] R. Gaffoglio, *Hyperthermia Treatment Planning: a state-of-the-art overview*, Polytechnic University of Torino, I-10129.

[11] HP Kok, P. Wust, PR Stauffer, F Bardati, GC van Rhoon and J. Crezee, *Current state of the art of regional hyperthermia treatment planning: a review*, Radiation Oncology, 10:196, 2015.

[12] Francesca De Tommasi, Carlo Massaroni, Rosario Francesco Grasso, Massimiliano Carassiti and Emiliano Schena, *Temperature Monitoring in Hyperthermia Treatments of Bone Tumors: State-of-the-Art and Future Challenges*, Sensors, vol. 21, no. 5470, 2021.

[13] F. Adibzadeh, K. Sumser, S. Curto, D. T. B. Yeo, A. A. Shishegar and M. M. Paulides, *Systematic Review of Pre-clinical and Clinical Devices for Magnetic Resonance Guided Radiofrequency Hyperthermia*, International Journal of Hyperthermia, vol. 37, no. 1, 2020.

[14] G. G. Bellizzi, T. Drizdal, G. C. van Rhoon, L. Crocco, T. Isernia and M. M. Paulides, *The Potential of Constrained SAR Focusing for Hyperthermia Treatment Planning: Analysis for the Head & Neck Region*, Physics in Medicine and Biology, 2018.

[15] Henry K. Lee, M.D., Andrew G. Antell, M.D., Carlos A. Perez, M.D., William L. Straube, M.S., Ganeshan Ramachandran, B.S., Robert J. Myerson, Ph.D., M.D.. Bahman Emami, M.D., Ernesto P. Molmenti, M.D., Alyson Buckner, M.D. And Mary A. Locket-T, M.A., *Superficial hyperthermia and irradiation for recurrent breast carcinoma of the chest wall: prognostic factors in 196 tumors*, Int. J. Radiation Oncology Riol. Phys., Vol. 4tJ. No. 2. pp. 365-575. 1998.

[16] M. Paulides, J. Bakker, R Jansen, R Levendag, J. van der Zee, G. van Rhoon, *A novel head and neck applicator for targeted hyperthermia*, ERASMUS MC, Radiotherapy, Rotterdam, The Netherlands.

[17] MM Paulides, Z Rijnen, P Togni, RF Verhaart, T Drizdal, D de Jong, M Franckena, GM Verduijn, GC Van Rhoon, *Clinical Introduction of Novel Microwave Hyperthermia Technology: the HYPERcollar3D Applicator for Head and Neck Hyperthermia*, Dept. Radiation Oncology, Erasmus MC Cancer Institute, Rotterdam, The Netherlands.

[18] Margarethus M. Paulides, M.Sc., Stefan H. J. A. Vossen, M.Sc., Ph.D., Adrianus P. M. Zwamborn, M.Sc., Ph.D., And Gerard C. Van Rhoon, Ph.D., *Theoretical investigation into the feasibility to deposit RF energy centrally in the head-and-neck region* , Int. J. Radiation Oncology Biol. Phys., Vol. 63, No. 2, pp. 634–642, 2005.

[19] Christian Rossmann & Dieter Haemmerich, *Review of Temperature Dependence of Thermal Properties, Dielectric Properties, and Perfusion of Biological Tissues at Hyperthermic and Ablation Temperatures*, Critical Reviews in Biomedical Engineering, 42(6):467–492, 2014.

[20] Mihaela Morega, SM IEEE, Alexandru M. Morega, SM IEEE, Alina Săndoiu, *Sensitivity to Parameters Variation in Numerical Simulation of Microwave Thermotherapy*, The 9th International Symposium On Advanced Topics In Electrical Engineering, May 7-9, 2015.

[21] René F. Verhaart, Gerda M. Verduijn, Valerio Fortunati, Zef Rijnen, Theo van Walsum, Jifke F. Veenland & Margarethus M. Paulides, *Accurate 3D temperature dosimetry during hyperthermia therapy by combining invasive measurements and patient-specific simulations*, Int. J. Hyperthermia, 31(6):686–692, 2015.

[22] René F. Verhaart, Zef Rijnen, Valerio Fortunati, Gerda M. Verduijn, Theo van Walsum, Jifke F. Veenland, Margarethus M. Paulides, *Temperature simulations in hyperthermia treatment planning of the head and neck region*, Strahlenther Onkol, 190:1117–1124, 2014.

[23] Liao Chengwang, *Singular Value Decomposition In Active Monitoring Data Analysis*, Handbook of Geophysical Exploration: Seismic Exploration, Volume 40, 2010.

[24] P. Togni , Z. Rijnen, W.C.M. Numan, R.F. Verhaart, J.F. Bakker, G.C. van Rhoon and M.M. Paulides, *Electromagnetic redesign of the HYPERcollar applicator: toward improved deep local head-and-neck hyperthermia*, Physics in Medicine Biology, 58:5997–6009, 2013.

[25] Margarethus M. Paulides, Jurriaan F. Bakker, Nicolas Chavannes, Member, IEEE, and Gerard C. Van Rhoon, Member, IEEE, *A Patch Antenna Design for Application in a Phased-Array Head and Neck Hyperthermia Applicator*, Ieee Transactions On Biomedical Engineering, VOL. 54, NO. 11, 2007.

[26] J. F. Bakker, M. M. Paulides, E. Neufeld, A. Christ, N. Kuster and G. C. van Rhoon, *Children and adults exposed to electromagnetic fields at the ICNIRP reference levels: theoretical assessment of the induced peak temperature increase*, Phys. Med. Biol., 56:4967–4989, 2011.

[27] S. Gabriely, R.W. Lau and C. Gabriel, *The dielectric properties of biological tissues: II. Measurements in the frequency range 10 Hz to 20 GHz*, Physics Medicine Biology, 41:2251–2269, 1996.

[28] *https://zmt.swiss/sim4life/*

[29] Ralph E. Hiatt, Life Fellow IEEE, *The IEEE Antennas and Propagation Society 1949-1982*, IEEE Transactions On Antennas And Propagation, Vol. Ap-32, No. 8, 1984.

[30] P. A. Hasgall et al., *IT'IS database for thermal and electromagnetic parameters of biological tissues*, 2018

[31] Emiliano Schena, Daniele Tosi, Paola Saccomandi, Elfed Lewis and Taesung Kim, *Fiber Optic Sensors for Temperature Monitoring during Thermal Treatments: An Overview*, Sensors, Vol. 16, No. 1144, 2016;

[32] *https://it.mathworks.com/help/stats/boxplot.html*