Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Empa
Materials Science and Technology

Physics informed neural network for thermal simulation of laser powder-bed fusion process

Sarven Yardim

Master Thesis

Seminar for Applied Mathematics, ETH Zürich
Experimental Continuum Mechanics,
High Temperature Integrity Group (HTIG) Group, EMPA

Supervised by Prof. Dr. Siddhartha Mishra, Dr. Ehsan Hosseini and Roberto Molinaro

January 2021

Abstract

In Laser Powder Bed Fusion (LPBF) Additive Manufacturing process, it is important to know the crucial factors affecting the production of robust, stable, and durable manufacturing parts. In order to find the best manufacturing parameters, thermal simulation of Additive Manufactured parts has to be conducted. However, thermal simulation of the laser powder-bed fusion (LPBF) additive manufacturing process is a challenging task. It involves calculating a highly localized transient temperature field generated by a moving laser with a typical radius of 30-100 micrometers and a velocity of 100-1000 mm/s, which raises the temperature by around $2000 - 3000°C$ within a couple of microseconds. Reliable Finite Element (FE) models of these processes require fine space and time discretization in the order of micrometers and microseconds, respectively. On the other hand, LPBF builds are typically in the range of centimeters, and their manufacturing takes hours. This discrepancy between the scale of conventional LPBF simulations and actual parts makes component-scale simulations inaccessible. Therefore, a balance between the accuracy and the computational cost has to be made, and as a result, in the context of this thesis, the main idea has been the use of Neural Networks as a surrogate model to act as an alternative to FE simulations. The Physics Informed Neural Networks (PINNs) have been implemented in order to solve the aforementioned problem without the need for training data. Additionally, the effectiveness of the PINN algorithm has been evaluated in the context of temperature dependent material properties by utilization of the Transfer Learning Approach. All models created with the help of PINN have produced predictions with relative errors $\lessapprox 20\%$ in comparison to FE simulations.

Acknowledgements

Firstly, I would like to thank Prof. Siddharta Mishra for giving me the opportunity to write this thesis at the Seminar for Applied Mathematics and allowing me to complete my Master's at ETH Zurich with this work. I am extremely grateful to Dr. Ehsan Hosseini and Roberto Molinaro for their constant support and supervision through these months. Special thanks to Pooriya Ghanbari for sharing his expertise with me regarding Abaqus AM Modeler. I would also like to express my gratitude to my colleague Ole Müller for helping me with the PINN algorithm. Finally, I would also like to thank my friends, Jian, Ibrahim, Patrik, and Xialong, for their nice discussions during lunch breaks and their moral support during these six months.

Contents

List of Figures

List of Tables

Nomenclature

Acronyms

ETH	Swiss Federal Institute of Technology
EMPA	Swiss Federal Laboratories for Materials Science and Technology
AM	Additive Manufacturing
LPBF	Laser Powder Bed Fusion
ANN	Artificial Neural Network
SGD	Stochastic Gradient Descent
DEM	Discrete element method
PDE	Partial Differential Equation
PINN	Physics-informed Neural Network

Symbols

t	Time	[s]
x	Position	[m]
v	Velocity	[m/s]
ρ	Density	$[\mathrm{kg\,m^{-3}}]$
k	Thermal conductivity	$[\mathrm{W\,m^{-1}\,K^{-1}}]$
μ	Dynamic viscosity	$[\mathrm{kg\,m^{-1}\,s}]$
ν	Kinematic viscosity	$[\mathrm{m^2\,s^{-1}}]$
∇	Gradient operator	$[\mathrm{m^{-1}}]$
q	Source Term	$[\mathrm{J\,m^3\,s}]$
c_p	Specific heat capacity	$[\mathrm{J\,kg^{-1}\,K^{-1}}]$
T	Temperature	[K]
u	velocity	$[\mathrm{m\,s^{-1}}]$
ζ_n	Stepping parameter	
δ_{kl}	Kronecker delta	
w	Weights	
L	Loss function	
η	Learning rate	

Chapter 1

Introduction

Additive Manufacturing (AM) is an emerging technology with the promise of taking an important and prominent role in the future of design and construction thanks to its advantages compared to conventional manufacturing methods. AM has rapidly accelerated production and innovation in various industries including, aerospace, automotive, medical, architecture, arts and design, food, and construction [Al Rashid et al., 2020]. Even though AM has several benefits over traditional manufacturing processes, there are still many challenges attributed to the AM before it can become the future of manufacturing. In this thesis, special emphasis will be put on the AM of metals, and most importantly the crucial factors affecting the production of robust, stable, and durable manufacturing parts. In the following subsections, the metal AM process, and challenges associated with the simulation of a specific metal Additive Manufacturing process called the Laser Powder Bed Fusion (LPBF) process will be explained.

1.1 Metal Additive Manufacturing

Metallic parts have been commonly fabricated by convention subtractive (machining, milling, etc.) means. However, with the invention of metal AM, a new method of manufacturing metals by joining materials to make objects from 3D model data, usually layer upon layer, as opposed to subtractive manufacturing methodologies, has been born [Frazier, 2014, ISO/ASTM, 2013]. In terms of mechanical properties, metallic materials fabricated by AM, have comparable static mechanical properties to the materials produced

by traditional manufacturing techniques. Furthermore, the high cooling rates achieved during the AM process reduce partitioning and favor reduced grain sizes. It is also observed through manufacturing processes that AM provided 30 to 50% cost savings in the production of certain structural components [Frazier, 2014]. There are several ways to produce metal parts additively, such as Direct Energy Deposition, Binder Jetting, Material Jetting, and LPBF; however, in this thesis, particular attention will be given to the LPBF Additive Manufacturing Process.

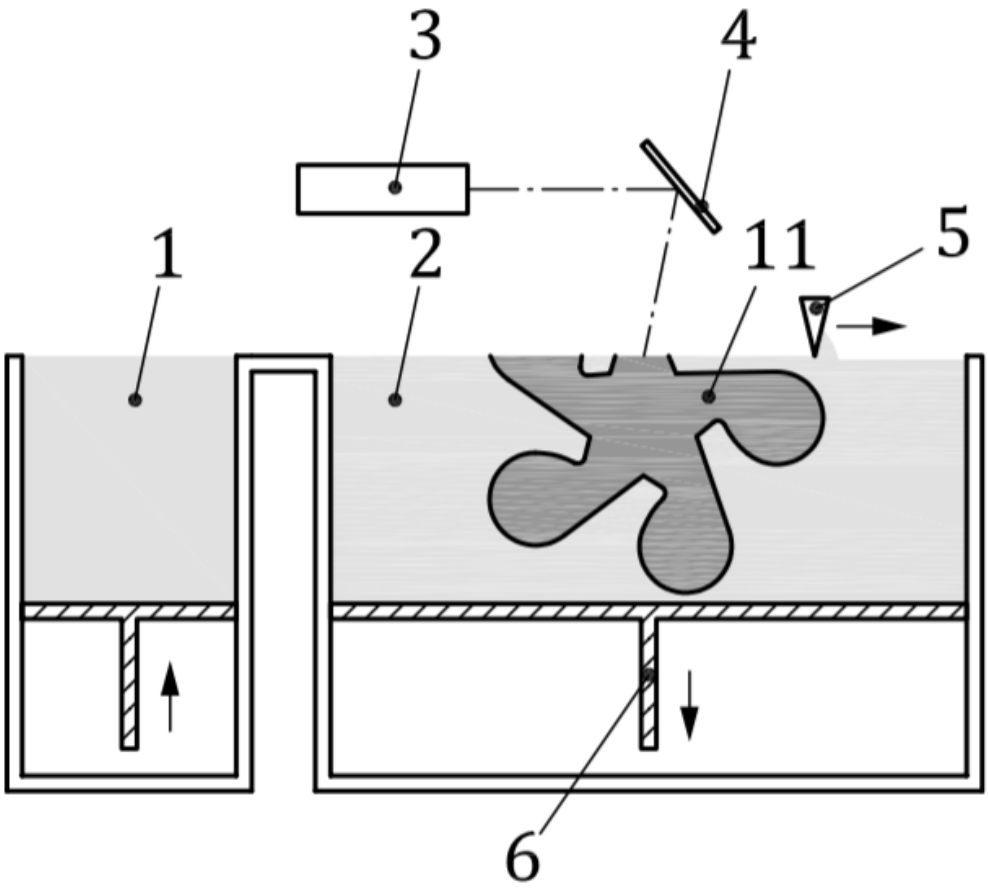

Figure 1.1: Working Principle of LPBF-Process [ISO, 2016]

1) Powder feeding system

2) Powder material distributed in a powder bed laser

3) Laser

4) Tilted mirror with focus

5) Powder spreading device

6) Build platform

In Figure 1.1, the working principle of the LPBF-Process is shown. As it can be seen from Figure 1.1, in LPBF AM Process, the laser beam interacts with the powder particles and by absorption of energy by powder particles, the temperature increases and particles start melting locally. Afterward, the thermal conduction mechanism takes place, which helps to distribute the heat through the deposited material. Finally, thermal conduction transfers some of the heat from the system to the substrate and therefore it dissipates some energy. Since there is a hot melt pool, the other thermal dissipation mechanism that happens during this time is radiation. There are also some elements that evaporate and leave the system. When they abandon the system, they also take some energy with them. This is also another heat dissipation mechanism. In the LPBF-Process, there is a gas atmosphere that also induces thermal convention to take some energy from the system and bring it to the gas environment. The other important mechanism that has to be considered is the phase transition, for example, latent heat of solidification or evaporation. These are the mechanisms more relevant for the thermal part. Since during the LPBF-Process, the liquid phase, molten metal, is created, there are some mechanisms important to describe the motion of the melt pool such as gravity, buoyancy, also wetting and capillarity effect resulting from the surface tension of the material, and finally the Marangoni effect. In summary, AM alloys have a complex thermal history involving directional heat extraction, and repeated melting and rapid solidification. Typically, AM manufactured alloys also experience repeated solid state phase transformations. These factors introduce complexities not typically found in conventional processes [Frazier, 2014].

There are several different approaches used for the melt pool simulation for Metal AM. In the first approach, the Discrete element method (DEM) is employed for the creation of powder bed, and Computational Fluid Dynamics (CFD) is used for following the melt pool motion and the temperature distribution inside the deposited material, as well. In the second method, the Rain Model is used to create the powder bed, and the Lattice Boltzmann method is utilized for conducting the fluid dynamical calculations. Both of these approaches can predict the different types of defect formation (porosity, lack of fusion, and balling, etc.). These approaches can also give predictions in terms of temperature distribution within the melt pool and at the area vicinity of the melt pool. However, there are several challenges concerning these approaches. For these types of simulations,

extremely fine element size and extremely fine time increments have to be used. As a result, these approaches result in a very high computational cost, and hence, only a very small simulation domain can be considered for simulations. Due to this reason, these types of simulations are not ideal for the calculation of the temperature profile within the whole component. In reality, the temperature profile and temperature evolution during the AM process are very important. Purely from the temperature profile, one might have some indications about the risk of damage formation (porosity, lack of fusion, etc.). Thermal profiles can also be used for the prediction of residual stress and distortion of AM material. Another important aspect of the thermal profile is that if knowledge about the mechanical performance of AM components is required, information about the microstructure is needed. To create a predictive model for microstructure, the input data from the temperature profile will be crucial. In summary, the main importance of thermal modeling is the generation of the input data for other types of simulations. Thermal simulation of the laser powder-bed fusion (LPBF) additive manufacturing process is a challenging task. It involves calculating a highly localized transient temperature field generated by a moving laser with a typical radius of 30-100 µm and a velocity of 100-1000 mm/s. Reliable finite element models of these processes require fine space and time discretization in the order of micrometers and microseconds, respectively. On the other hand, LPBF builds are typically in the range of centimeters, and their manufacturing takes hours. This discrepancy between the scale of conventional LPBF simulations and actual parts makes component-scale simulations inaccessible. In Figure 1.4 the dominant physical phenomena during melting are illustrated in a partially molten powder bed. The temperature distribution in the powder bed and on the melt pool surface is indicated by color-coding (blue indicates cooler temperatures, and red indicates warmer temperatures). The semitransparent melt pool surface enables the visualization of melt pool dynamics by velocity arrows. The bottom of the melt pool is visualized in white and the beam source in semitransparent red.

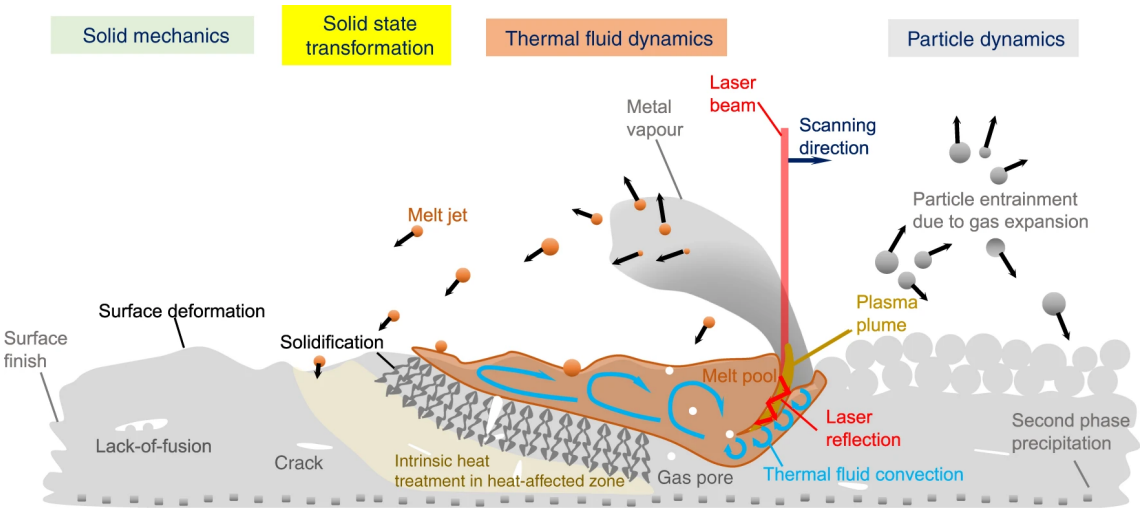

Figure 1.2: Multi-Physics Phenomena in LPBF-Process [Panwisawas et al., 2020]

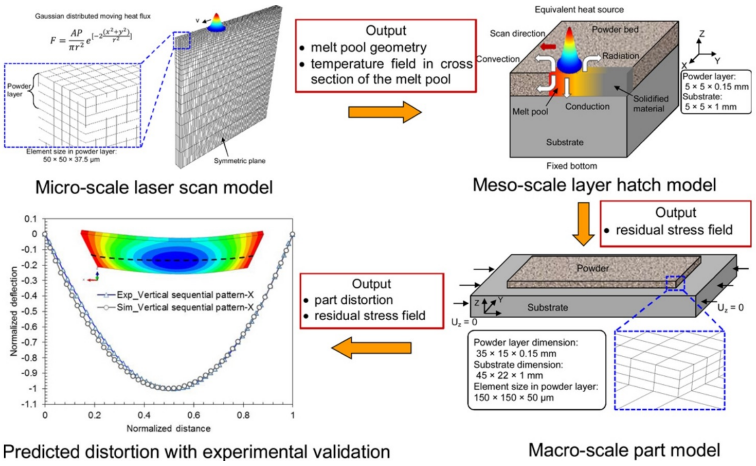

Figure 1.3: Multiscale Modeling Approach of the LPBF-Process [Li et al., 2015]

1.1.1 Challenges in Simulating Additive Manufacturing

1.1.2 Continuum Thermal Simulation

The main idea behind the Continuum Thermal Simulation is that the fluid part of the problem is not taken into account, and as a result, the creation and the motion of a melt pool are not considered. Therefore, all the fluid dynamical mechanisms are neglected, and hence the computational cost of the problem decreases, and as a result, a larger simulation domain can be simulated. As the name suggests, the main assumption behind the Continuum Thermal Simulation is that the particles are considered as a continuum medium. In other words, in order to simplify the problem the whole powder bed with discrete powder particles is represented by a continuum medium, and a set of equivalent

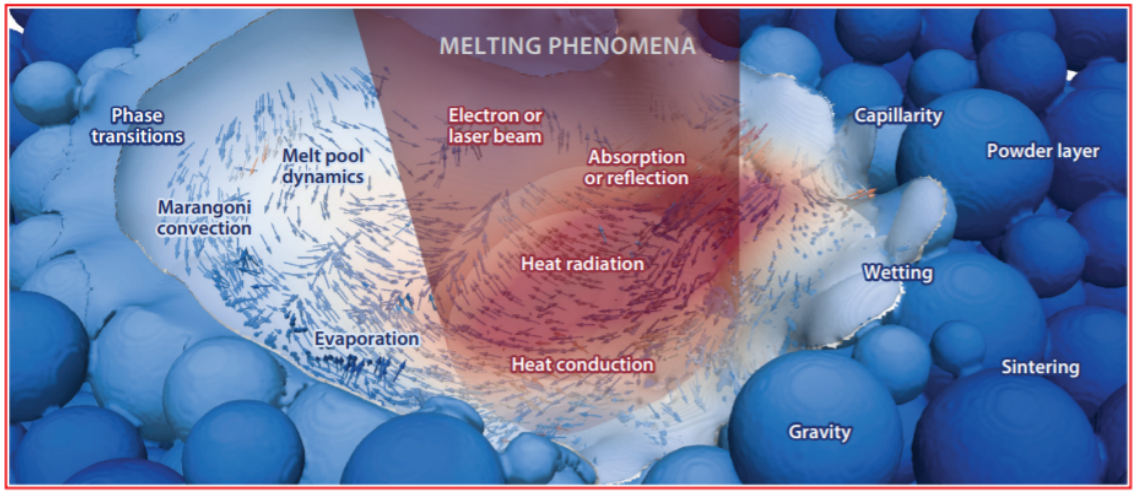

Figure 1.4: The Dominant Physical Phenomena During Melting in the LPBF-Process [Markl and Körner, 2016]

properties for this layer is considered. The whole layer is not only filled with material, but also with gas. Specifically, in the parts of the powder bed, where there is porosity (filled with gas), a lower thermal conductivity has to be considered since the thermal conductivity of the powder bed is less than the thermal conductivity of the layer with solid material. The second assumption is that the formation of the fluid phase inside the simulation domain is not considered, and therefore the Continuum-based Thermal Modeling omits all fluid-related mechanisms such as gravity, buoyancy, wetting, and capillarity, Marangoni effect, and evaporation of the material.

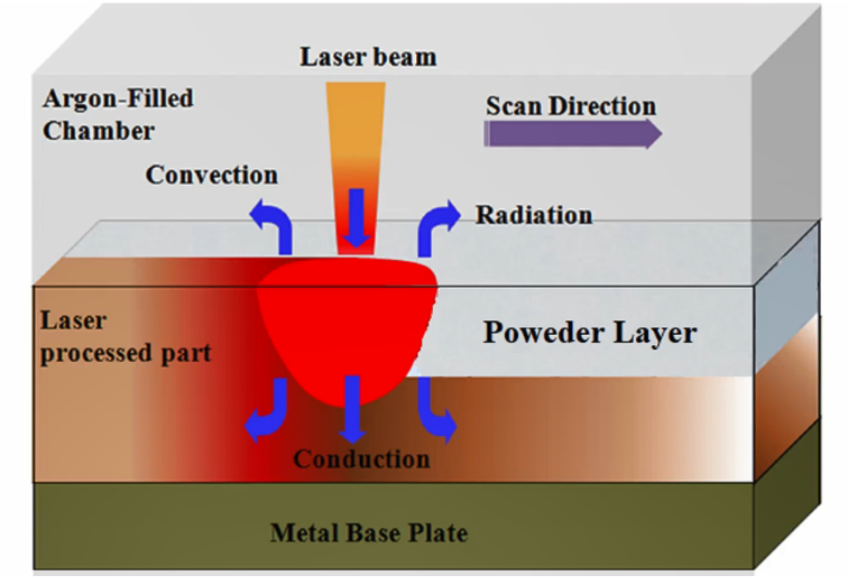

Figure 1.5: Heat Transfer Mechanisms during the LPBF-Process [Yuan and Gu, 2015]

Figure 1.5 gives an overview of the heat transfer mechanisms, which will be considered in this thesis as a part of Continuum Thermal Modeling. As it is shown in Figure 1.5, the powder layer is represented by a continuum medium. With this model, the interaction of the beam source with the powder bed is still considered. The thermal conduction, which helps to distribute the temperature within the part and also to the substrate is also considered. The thermal radiation from the surface of the part is also taken into account. Furthermore, when required (in the case of the gas-filled chamber) the thermal convention is taken into consideration. It is also important to consider the latent heat of melting/solidification, which has a significant effect on the temperature inside the melt pool. However, when it comes to the latent heat of evaporation, it is not possible to consider it in this type of calculation, since when the material evaporates, it will leave the system, and in Continuum-based Thermal Modeling removal of any mass from the system is not considered. Therefore, the evaporation mechanism is not taken into account in this kind of calculation. Another important mechanism that helps improvement of the temperature uniformity inside the melt pool, such as the movement of the melt pool due to the Marangoni effect or buoyancy is not considered since in this type of calculation the effect of melt pool motion on the thermal field inside the melt pool is not included. However, researchers artificially increase the thermal conductivity of the liquid phase to increase the heat transfer, and therefore they consider the effect of melt pool motion.

There are several studies that show that the way the energy source (heat source) is defined will affect the predicted melt pool shape, melt pool size, temperature profile and distributions, cooling rate, and also temperature gradients. It is known that temperature gradients are important for the prediction of the residual stress and cooling rates are essential for the prediction of the microstructure. Therefore, it can be concluded that the establishment of an appropriate heat source model will eventually affect all the predictions in the subsequent simulations. A 3-dimensional heat source formulation is required to consider the penetration of the laser beam into the powder bed. As it can be seen in Figure 1.6, Zhang et al. have examined consideration of 8 different types of representation for the energy distribution [Zhang et al., 2019], and concluded that semi-ellipsoid heat source model, which was proposed firstly by Goldak et al. [Goldak et al., 1984], in the LPBF simulation, shows good agreement with the experimental results. As it is shown in Figure

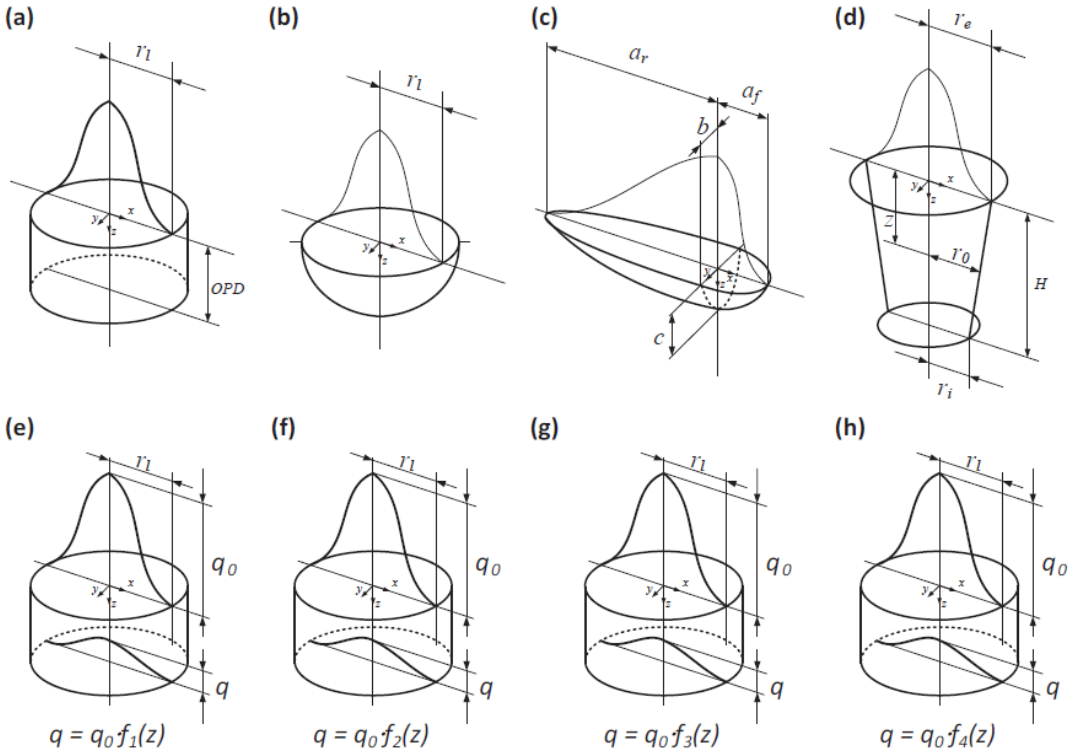

Figure 1.6: The schematic of the heat source models, (a) cylindrical shape; (b) semi-spherical shape; (c) semi-ellipsoidal shape; (d) conical shape, (e) radiation transfer method; (f) ray-tracing method; (g) linearly decaying method; (h) exponentially decaying method [Zhang et al., 2019]

1.6 (c), often the front part of the ellipsoid is considered to be different from the rear part. Thus the double ellipsoidal power density equation was proposed. The front part of the ellipsoid can be expressed as,

$$q_f(x,y,z) = f_f \frac{2^{5/2}\beta P}{\pi^{3/2} a_f bc} \exp\left[-2\left(\frac{x^2}{{a_f}^2} + \frac{y^2}{b^2} + \frac{z^2}{c^2}\right)\right] \tag{1.1}$$

where β is the absorptivity of laser beam, P is the laser power (W). The rear part of the ellipsoid can be written as follows,

$$q_r(x,y,z) = f_r \frac{2^{5/2}\beta P}{\pi^{3/2} a_r bc} \exp\left[-2\left(\frac{x^2}{{a_r}^2} + \frac{y^2}{b^2} + \frac{z^2}{c^2}\right)\right] \tag{1.2}$$

where a_f and a_r are the semi-axes of the front and rear ellipsoids. It should be noted that $f_f + f_r = 2$.

It is very important to specify the governing equations that are of significant importance

in this thesis.

$$\frac{\partial E}{\partial t} + \nabla.(\vec{u}E) = \nabla.(\alpha \nabla E) + \dot{Q} \tag{1.3}$$

$$\partial E = C_p \partial T \tag{1.4}$$

$$\alpha = \frac{k}{\rho C_p} \tag{1.5}$$

The Equation 1.3 indicates that the evolution of the energy state in the control volume elements ($\frac{\partial E}{\partial t}$) depends on the difference in the energy of the material which comes in or goes away from the control volume ($\nabla.(\vec{u}E)$ - the convective thermal energy), the amount of energy which might be transferred through the boundaries of the control volume ($\nabla.(\alpha \nabla E)$ - thermal energy diffusion) and also the energy generation within the control volume elements ($\dot{Q}$ - beam deposited energy). Since in this simulation framework, there is no mass flow/transfer, the convective thermal energy can be neglected and as a result, the formula becomes simpler. With the consideration of the Equation 1.4, which states that the variation in the energy state of the system depends on the heat capacity (C_p) and the variation of the temperature, and simple considering the definition of the thermal conductivity (Equation 1.5, where k is the thermal conductivity and ρ is the density), and with the assumption of constant thermal conductivity and heat capacity, the Equation 1.3 can be rewritten in the form of Equation as follows,

$$\rho C_p \frac{\partial T}{\partial t} = \frac{\partial}{\partial x}\left(\frac{\partial T}{\partial x}\right) + \frac{\partial}{\partial y}\left(\frac{\partial T}{\partial y}\right) + \frac{\partial}{\partial z}\left(\frac{\partial T}{\partial z}\right) + \dot{q} \tag{1.6}$$

$$k\frac{\partial T}{\partial n} + h(T - T_0) + \sigma\epsilon(T^4 - T_0^4 \tag{1.7}$$

and the Fourier type Heat Conduction Equation (Equation 1.6) can be derived. This is the differential equation that is solved during the Continuum-based Thermal calculation and in order to solve this Partial Differential Equation (PDE) a set of initial and boundary conditions are required. Often as an initial condition a uniform temperature distribution

within the body is considered ($T(x, y, z, 0) = T_0$ - T_0 is the ambient temperature). As a boundary condition, usually a constant temperature at the bottom of the substrate ($T(x, y, 0, t) = T_0$) might be considered. At the free surfaces the effect of heat radiation (Equation 1.7, where n is the vector normal to the surface, h is the convective heat transfer coefficient, σ is the Stefan-Boltzmann constant, ϵ is the emissivity) can be implemented.

It is also crucial to understand what specific considerations in terms of material properties have to be taken for Continuum-based Thermal Simulation. For this type of analysis, three material properties are needed, density, heat capacity, and thermal conductivity. From the Equation 1.6, it can be seen that the first term is the material density. Since in this type of simulation, a quite wide range of temperature values are observed, and because of the existence of different material phases such as powder, solid, and liquid, the evolution of the density as a function of temperature for different phases should be known. In general, the density of a material decreases with the increasing temperature values. In fact, an equation can be derived comprising the density, temperature, and thermal expansion of the material. However, it is observed that when the transition from solid to liquid phase occurs, a drastic change of the density (a sudden jump in the density value) might be noticed as shown in Figure 1.7.

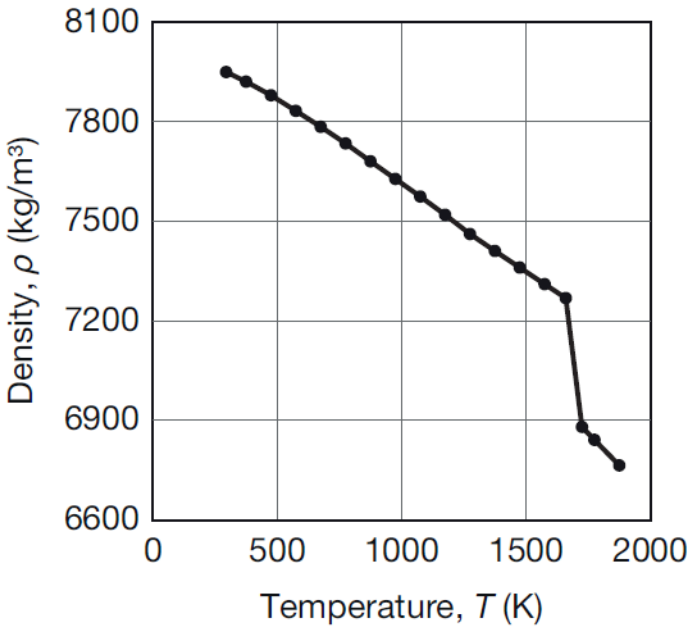

Figure 1.7: Temperature dependent density of 316L stainless steel [Piscopo et al., 2019]

Another important point is that the density of the powder material also should be taken into account. It is possible to write down the density on powder layer based on the Equation below,

$$\phi = \frac{\rho_{solid} - \rho_{powder}}{\rho_{solid}} \tag{1.8}$$

where ϕ represents the relative porosity, and often a relative porosity of 40-60% is considered for powder layer.

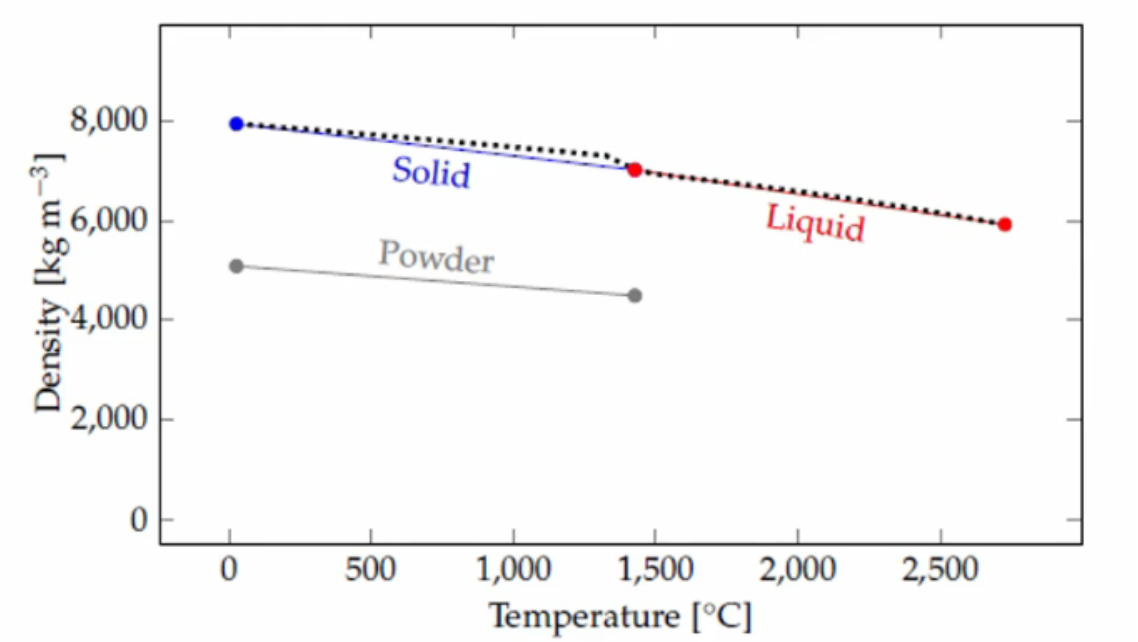

Figure 1.8: Temperature dependent density values of different phases [Kim et al., 1975]

However, Gh Ghanbari et al. have shown that it might not be a good idea to consider different densities for powder and solid material because of the fact that when the density of the material is changed at a certain point, the mass conservation is violated in the system. In other words, since the volume of the elements is constant, changing the density of the material results in the changing of mass of the system. Another issue is that the thickness of the powder layer is larger than the thickness of the deposited material. In this type of calculation, the change of the thickness of the material cannot be considered, since the thickness of the powder layer and the thickness of the deposited material are assumed to be the same. Therefore, it is concluded that it might be more reasonable to consider the same density values for powder and solid material. Furthermore, at the solidification/melting

point, there is a jump in the density of the powder when the transition from powder to liquid occurs, and this by itself causes some convergence issues due to the fact that all numerical schemes work with a set of derivatives, and therefore they face difficulties for consideration of a drastic change/jump in material properties.

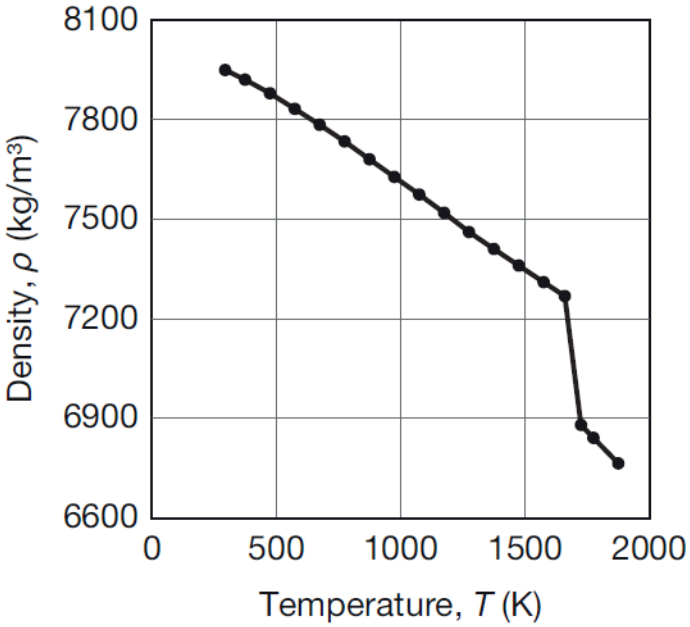

Figure 1.9: Temperature dependent heat capacity of 316L steel [Piscopo et al., 2019]

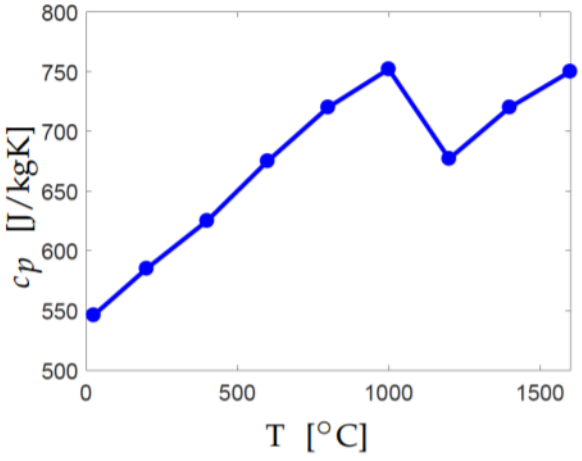

Figure 1.10: Temperature dependent heat capacity of Ti-6Al-4V [Ranjan et al., 2020]

The second material property is the specific heat. It is the amount of heat that is needed to give to a unit mass of the material in order to raise its temperature by one degree Celsius. Its definition is shown in Equation 1.9. Again the evolution of specific heat

at different temperature values has to be considered. Often, the specific heat increases by temperature, as shown in Figure 1.9. There is a drastic change in the specific heat of the material at the point where the transition from solid to liquid phase happens. The underlying physics behind why the specific heat increases with the temperature and why the specific heat is different for solid and liquid phases have to be investigated. Specific heat is a measure of the ability of a material to store internal energy. Materials can store energy in different ways, but mainly by increasing the kinetic energy of their molecules through vibrational, rotational, and translational movements of their molecules. The possibility of having these motions depends on the energy state of the material. It means that at low-temperature values, when the internal energy of the material is low, there are only a few degrees of freedom for molecules to move, and therefore there is only a possibility of translation. By increasing the temperature, more degrees of freedom are added, and as a result, molecules can also rotate. It means that materials have a higher ability to absorb energy. By increasing the temperature, the degrees of freedom are also increased, but at a certain point, all energy levels are populated, and therefore the heat capacity remains constant. It is usually observed that the heat capacity of the liquid is not sensitive to the temperature because of the fact that the liquid phase is the state where all degrees of freedom are activated, and materials are not able to introduce new degrees of freedom for absorption of energy. The other fact is that the specific heat varies for different phases. The specific heat of the liquid phase differs from the solid material or even in the solid material, there might be a different crystallographic structure and as a result, a different capacity might be observed. For example, as it is shown in Figure 1.10, in the case of Ti-6Al-4V, at around 900°C Ti-6Al-4V switches from α phase to β phase, and since the crystallographic structure is different, all properties including the specific heat of the material are different. For the powder phase, a different specific value is not considered since the specific heat is defined for the unit of mass, and it does not depend on the porosity level. Therefore, the same specific heat value is considered for both solid and powder phases.

$$C_p = \frac{\partial E}{\partial t} \tag{1.9}$$

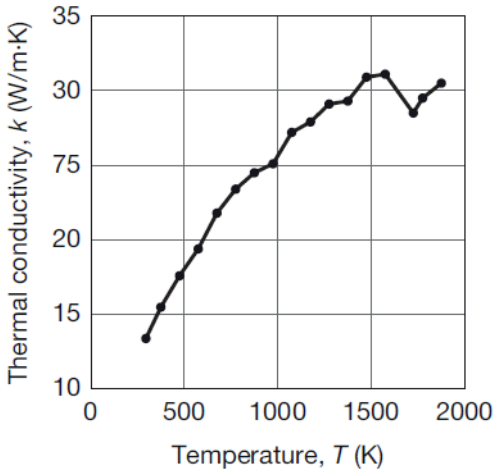

Figure 1.11: Temperature dependent thermal conductivity values of 316L stainless steel [Piscopo et al., 2019]

The final material property that has to be considered is thermal conductivity. As it is shown in Figure 1.11, from the variation of thermal conductivity over temperature for 316L stainless steel, it can be seen that thermal conductivity increases, then decreases at the melting point and increases again. In Figure 1.12, the evolution of thermal conductivity of different materials have been shown, it can be observed from Figure 1.12 is that for some materials thermal conductivity increases with temperature, on the other hand, for other materials, the thermal conductivity decreases with temperature. Therefore, it can be concluded that thermal conductivity has a more complex behavior than other material properties. To understand the main reason behind the behavior of thermal conductivity, first, the physics behind the mechanism for heat transfer in solid materials has to be understood. There are two main mechanisms for heat transfer in solid materials. Heat in a solid material is transferred either by lattice vibration or by the movement of free electrons. For pure metals, the main heat transfer/heat conduction mechanism is the motion of the free electrons, and therefore, it is expected that heat conductivity follows a very similar behavior to that of electrical conductivity. With the increase in the temperature, the vibrations of atoms inside the material increase, and as a result, atoms obstruct the movement of free electrons, and therefore it is expected that by increasing temperature, the electrical and thermal conductivity of the material decreases. However, in the case of alloys or non-metals, a different phenomenon occurs. For alloys and non-metals, the lattice vibrations and generated waves are responsible for the conduction of heat and therefore,

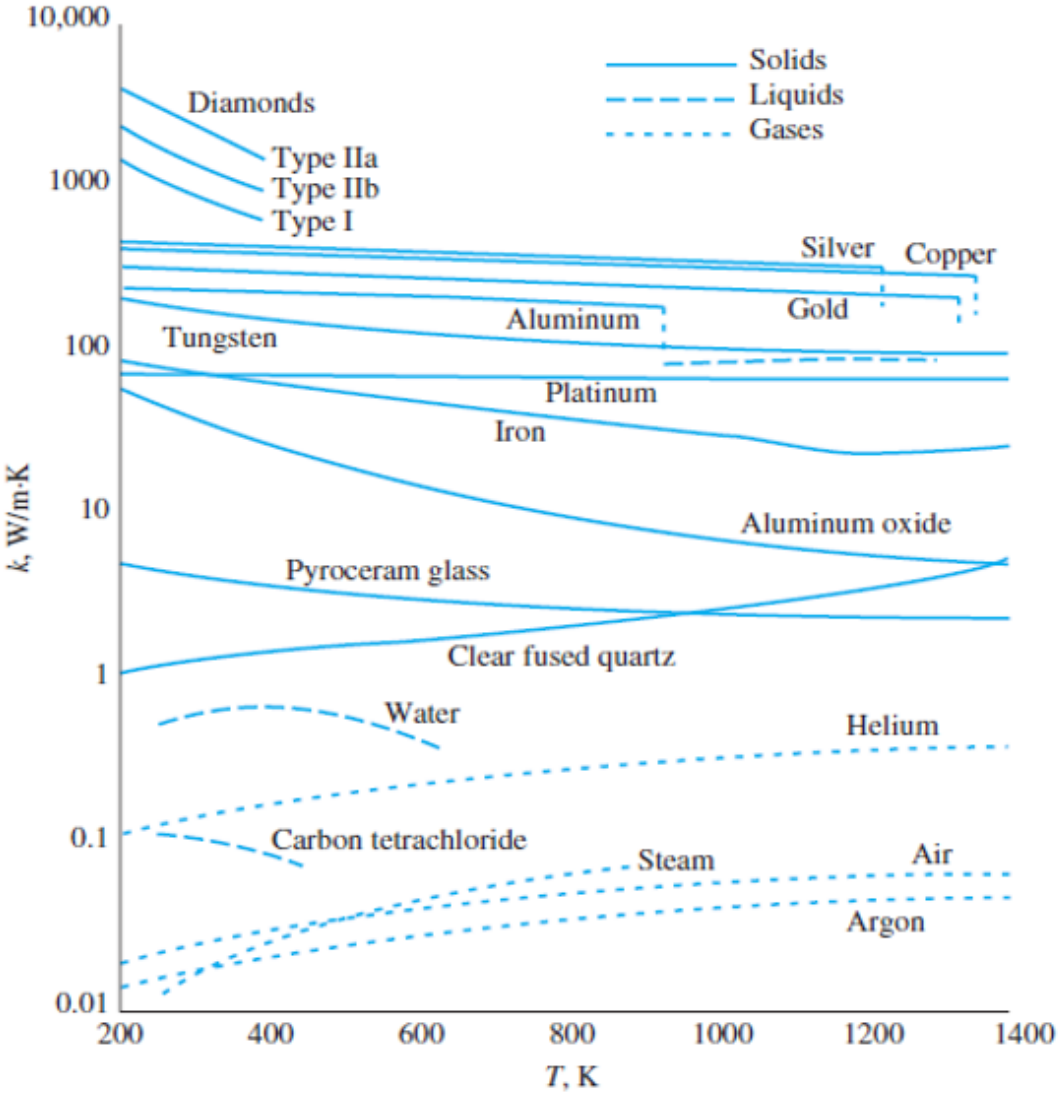

Figure 1.12: Thermal conductivity of different materials [Kumar, 2019]

at higher temperature values, more intensified vibrations of atoms are expected and as a result, higher thermal conductivity values are observed due to the increase in the heat transfer rate. The reason behind the low thermal conductivity values observed in liquids is that, for heat transfer by vibration waves, an ordered structure/crystalline structure is needed; however, in the liquid phase, there is no ordered structure and therefore, the heat transfer due to lattice vibrations is significantly reduced, and heat conductivity drops. In the case of the powder bed, some sources propose to use a mixture rule to assume the powder bed as the combination of metal particles and the air, and therefore considering the 40-60% porosity level, it is assumed that the thermal conductivity of the powder phase

is 40 to 60% of the solid phase. However, this relationship has never been confirmed by the experimental observations. During experiments, it has been observed that powder conductivity does not only depend on density but also powder size distribution and shape. The general conclusion is that the conductivity of the powder phase is much less than 40-60%, and as a rule of thumb it is considered that the powder conductivity is only 1% of the solid phase. This means that, at the melting temperature, there is a jump in the thermal conductivity, the powder transfers into a liquid phase, and since the liquid phase has a higher thermal conductivity, there will be a drastic change in the value of thermal conductivity. As it is mentioned before, numerical methods experience difficulties in the case of such drastic changes, but unfortunately, so far no solution has been proposed in order to solve this issue.

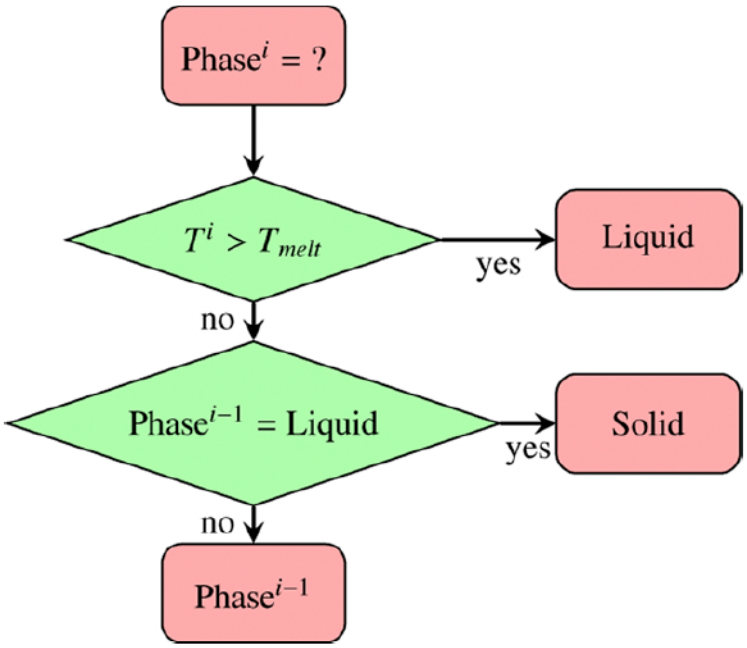

Figure 1.13: Schematic representation of the employed phase transition rule for simulation of the LPBF process [Gh Ghanbari et al., 2020]

It is mentioned before that the LPBF process includes the transformation of powder particles to molten metal upon beam exposure and solidification of liquid metal upon cooling. Therefore, any material point in the simulation domain can take one of the three states of powder, liquid, or solid. In order to determine the phase in the simulation domain, a set of user-defined subroutines is created and provided to the Finite Element

(FE) packages. In these subroutines, as it is shown in Figure 1.13, initially, the temperature values of each material point of which the respective phase wants to be known, is calculated. If the temperature value of that material point at the current time is above the melting temperature, then the material is in the liquid state. However, if the temperature is less than the melting point, there exist two conditions. If the previous state of the material is either powder or solid, the material keeps its state, but there might be the condition that the material is in the liquid phase in the previous increment and at the current increment, the temperature is below the melting temperature, then the formation of the solid phase should be considered. The liquid phase from the previous time increment now switches to the solid-state. Within this type of calculation, the evaporation and the formation of the gas phase are neglected since the mass transfer is not included in the problem definition of these simulations.

There are several challenges faced in the Thermal Simulation for Metal Additive Manufacturing. The challenge is about the generation of the material data. For this kind of calculation, the material properties over a wide range of temperature values for powder, liquid, and solid materials are needed. Experimental measurement of material properties is challenging and expensive, in particular for the liquid phase. Often, researchers conduct a few experiments for the solid phase and then try to have an idea about material properties in that range. For the temperature values, researchers either try to extrapolate, use educated guesses or simply use theoretical values, and as a result, this approach has several negative impacts on the accuracy of the predictions. The other problem that is encountered in the Thermal Modelling of MAM is that simulations might face difficulties in convergence. The reason is that, in the simulation domain, fast temperature evolutions and high gradients in the temperature field at the area close to the melt pool are observed. Furthermore, there are some physical phenomena in the system, such as thermal radiation, which introduce non-linearity. It is known that FE simulations work perfectly for linear problems; however, when the non-linearity of the problem increases, it will be more difficult for simulations to converge. As an extreme situation, in the case of a jump in material properties when the transition from powder to liquid state happens, the numerical scheme will face challenges with this drastic change. The final challenge is that in the Thermal Modelling of MAM, very small time increments and very fine element sizes have to be used

in order to accommodate large temperature gradients and fast temperature evolutions. As a result, the computational cost of the model is increased significantly. When the computational cost of the LPBF process is examined, with the consideration of the melt pool, and also heating/cooling rates, an element size of 10-20 µm and time increments smaller than 1 ms have to be used. If a print of 2 cm cube is considered, it means that 10^9 elements and 10^8 time increments are needed. Performing such a simulation is not possible for most computer systems. Therefore, there are several methods to reduce the computational cost of the simulations. There exist various proposed approaches to reduce the computational cost. Mainly five different approaches will be explained in the following paragraphs. The first two approaches are very rough approximations, but they can significantly reduce the computational cost. The next two are approaches that try to refer to the problem regarding the requirement for very fine element sizes. Finally, the last one tries to solve the problem related to the requirement of small-time increments, and also fine elements.

The first approach is called Approximate Approaches or Lumped Heating. In this approach, neither the in-plane movement of the laser beam nor the deposition of individual layers is taken into account. Instead, what is done is that a metal layer is defined (a metal layer can include one to several physical layers), and equivalent heat flux to the whole layer is applied at the same time. This approach significantly reduces the computational cost of the analysis, and one reason is that in this type of calculation, large temperature gradients and fast temperature evolution rates are neglected; therefore, there is no need to use fine elements or small time increments. However, for this reduced computational cost, there is a pay-off. The result of this approach is not that reliable. In this type of calculation, the peak temperature cannot be predicted, and the information about temperature gradient and cooling/heating rates is lost. This method is used by research in particular for the prediction of residual stress.

The second approach is Steady-state Eulerian Framework. Often, a Lagrangian system is used for the simulation of the AM which means that the mesh is bonded to the material, and the heat source moves through the simulation domain and deposits material. This problem can be approached differently, and it can be assumed that there is a mesh which material can flow through and there is a fixed location for the heat source. Fixing the

location of the heat source makes it possible to use fine elements only at the area in the vicinity of the heat source, and use much larger elements size for the rest of the simulation domain. As a result, the number of degrees of freedom in the problem is reduced. Furthermore, when this problem is solved, it is observed that a steady-state solution will develop. The temperature within the system will not change anymore. Therefore, this approach can significantly reduce the computational cost, both by decreasing the number of degrees of freedom and also by replacing a transient solution with a steady-state solution. However, in the case of the application of this method to the AM, some problems arise. First of all, in AM, any strict steady-state solution does not develop. The temperature profile at each location of the deposited material is different. The second limitation is that this type of approach is applicable only to simple geometries; however, AM is often applied to complex geometries, and therefore again this approach is not usable. In conclusion, this approach is quite effective for the simulation of welding and direct energy deposition of very simple geometries, but when it comes to processes such as LPBF this approach is not applicable.

The third approach is Adaptive Remeshing. This approach is a well-known approach for dealing with the problems regarding the development of high gradients at some parts of the model. This approach has an adaptive algorithm that detects these high gradients and starts adding more elements in these areas where high gradients are observed. This algorithm has been implemented for the simulation of MAM, which means that there is an area in the simulation domain with high mesh density moving with the heat source in order to capture the large thermal gradients. Whenever the remeshing algorithm activates, the results from the previous mesh have to be mapped to the newly generated mesh. The remeshing and remapping processes are computationally expensive, and therefore, the number of remeshing steps has to be optimized. This type of approach helps to decrease the number of degrees of freedom in the system, fundamentally to reduce the number of elements in the model; however, for this calculation, the small-time increments have still to be used.

The next approach is Stepwise Element Coarsening. Here, the fine level of discretization for the recently deposited layer is kept, but for the rest of the model, the size of elements is increased, and that makes it possible to keep the number of degrees of freedom in the

simulation domain rather low. This approach is not implemented into FE software packages and therefore should be programmed by the user. The difference between this approach and the one with Adaptive Remeshing is that in this approach the number of remeshing steps is much less. Therefore, the cost of remapping the result between the two meshes decreases.

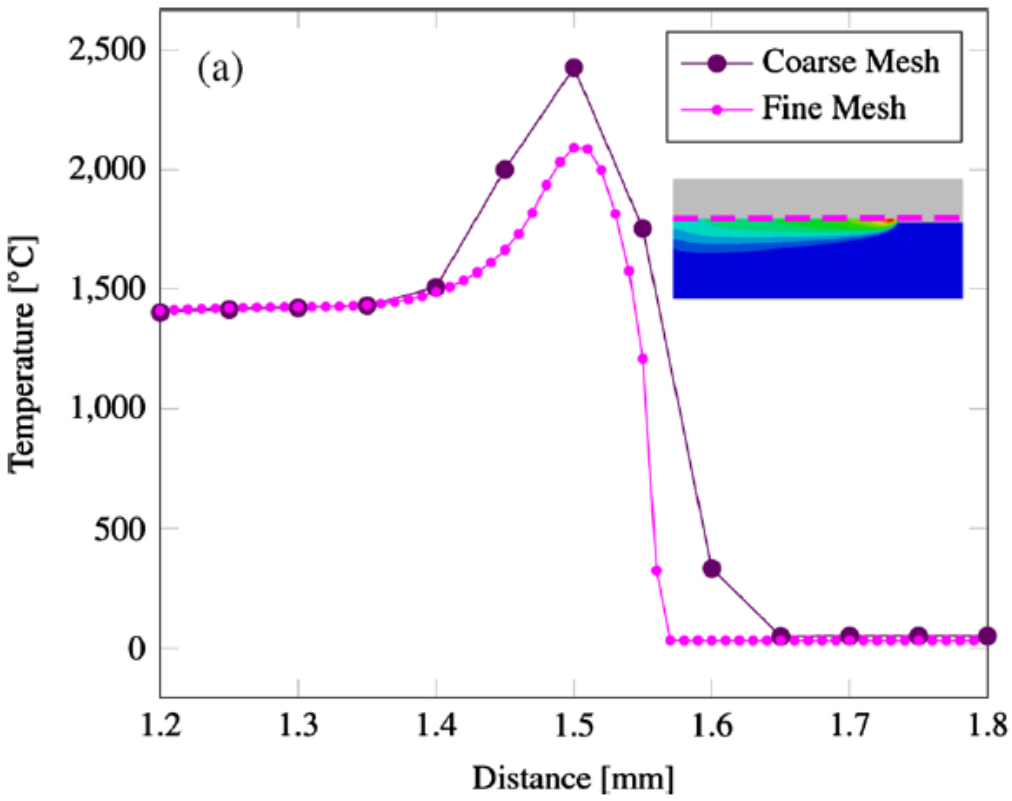

Figure 1.14: Comparison of temperature profiles for fine and coarse mesh

The final approach is the Adaptive Local-global Approach which is under development at Swiss Federal Laboratories for Materials Science and Technology (EMPA) by Pooriya Gh Ghanbari as a part of an ongoing Ph.D. project [Gh Ghanbari et al., 2020]. The main idea behind this approach is to reduce the computational cost with a minimum loss in terms of accuracy. The idea is that even with a coarse mesh, an accurate solution can be achieved at some area far from the melt pool. In Figure 1.14 the results of a fine mesh solution and the result of a coarse mesh solution are shown [Gh Ghanbari et al., 2020]. It can be seen that 100-200 µm away from the center of the melt pool, the result of two calculations merges. Therefore, the main logic behind this approach is to run a coarse

mesh analysis for the whole geometry of the model and correct it with a set of local fine mesh solutions which will result in an accurate temperature distribution at the area close to the melt pool. This type of analysis, similar to Adaptive Remeshing argues that a high level of discretization is only required for the area close to the melt pool, and for the rest of the simulation domain a much coarser discretization can be used. Therefore, the approach breaks down the simulation into two scales, a global scale that uses large elements and large time increments to simulate the whole geometry, and it is acknowledged that the temperature predictions by the global scale at the vicinity of the melt pool are all wrong. Then in the second scale, which is called local scale, fine elements and also small time increments are utilized in order to recalculate the temperature profile at the vicinity of the laser location, and there are many of these local calculations since they have to follow the location of the laser. In the end, the result of the global calculation is combined with the outcome of the numerous local calculations to a single database, and therefore the thermal analysis, which is valid for the whole geometry is achieved. In this approach, there is a link between the global solution and local simulations, and the link is that the local models import the boundary temperatures from the global simulation. There is also a link between different local simulations. For the local models, the result of the previously solved local calculation is defined as the initial temperature for the new local calculation. Within the local model, there is a non-uniform mesh distribution with high mesh density in the vicinity of the beam location. This approach allows the user to create a final output database with the desired mesh matrix and time increment density. The Multiscale Adaptive Local-global Approach decreases the computational cost of the simulations while the introduced errors are below 1%. However, it should still be noted that even with the Multiscale Adaptive Local-global Approach the computational cost is still quite significant, and therefore in order to reduce the computational cost different approaches are under development.

One of these approaches, which is investigated in this thesis is the use of Machine Learning algorithms. As it is mentioned before, in the Multiscale Adaptive Local-global Approach many local calculations have to be performed. All these local calculations share the same mesh distribution, the same geometry, and the same amount of movement of the laser beam, and they are only slightly different in terms of initial and boundary conditions. Therefore, in reality, there is a correlation between the predicted temperature

profile and the condition at the boundary of these local calculations. This enables the use of data-driven Machine Learning algorithms in creating a surrogate model in order to spot this correlation. Surrogate models statistically link input data to output data, which are acquired through the complex system simulation. These models are commonly utilized when the link between input and output data is not known, or when the relationship is extremely complex, and a simpler relationship with reasonable accuracy is needed [Davis et al., 2017]. Typical surrogate models include Kriging [Zhou and Lu, 2020], Support Vector Machines (SVM) [Pan and Dias, 2017], Radial Basis Functions (RBF) [Zhou et al., 2019], low-rank tensor approximations [Konakli and Sudret, 2016], Polynomial Chaos Expansion [Keller et al., 2020], [Smatsi et al., 2020], and Neural Networks [Baumann et al., 2018]. In this thesis, mainly the Neural Networks, in particular, a novel Network Network approach based on Physics-informed Neural Networks will be investigated.

1.2 Artificial Neural Network

Machine learning is a process of data analysis that enables machines such as computers to learn from large data set without being explicitly programmed. Deep learning is a sub-field of machine learning used for the implementation of Machine Learning algorithms and mimics the human brain. Therefore, the purpose of deep learning is to make a machine learn and work in a similar way that the human brain actually learns from experience [Lecun et al., 2015, Haykin, 1999, Goodfellow et al., 2016].

Artificial Neural Network (ANN), which comprises neurons, and these neurons behave in a similar way as neurons in the human brain. The neurons receive large numerical data, which consists of input features, transmit this data to the other neurons, and at the end of this process, the desired values are extracted. This process is also called as the learning process or the training process. A simple Artificial Neural Network consists of mainly three layers. Input layers are the layers, where the input features are provided. Hidden layers, which are located between the input and output layers and construct the main body of the ANN. They are the layers used to get the intended output by applying suitable mathematical operations to the input data. Finally, the output layers are the layers where the output data are obtained.

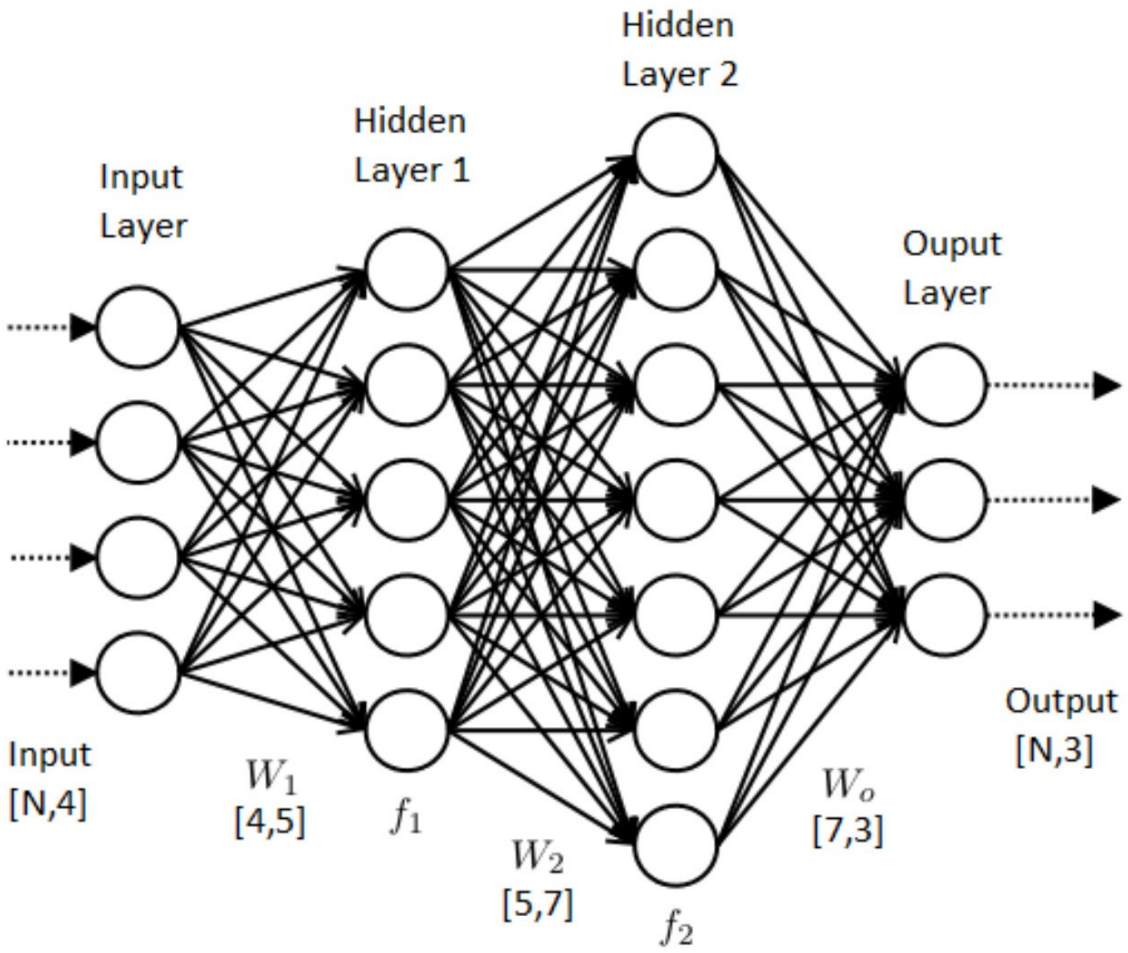

Figure 1.15: The outline of the Artificial Neural Network [`https://medium.com`]

A schematic outline of the Artificial Neural Network, which presents the layer structure described above, is shown in Figure 1.15. To understand the process performed in the hidden layers, it is crucial to acknowledge the motivation behind the lines depicted in Figure 1.15 and the importance of the weights. Since the ANN mimics the behavior of the human brain, it is helpful to consider in a biological manner. In the human body, when the input features (the real-time visual data) are acquired from the eye, they pass through the neurons, and each neuron does some processing to generate an output that the human can comprehend in a meaningful way. In a similar way, in the ANN, there are some weights assigned, shown as W in Figure 1.15, to these input features, and there is also a concept of activation function which plays a significantly important role. When the weights are assigned, these weights will be passed to the hidden neurons, and then once it is passed to the hidden neurons, there will be two types of operation that happen inside

the hidden neurons. In the first operation, the summation of the products of the weights and input features is performed. Once the summation is calculated, a parameter called bias is added to the summation. The bias is used to adjust the output of this summation and can be considered as a threshold that determines whether a neuron will produce a meaningful result or not. In the second operation, the result of the final summation is passed through the activation function, such that these neurons can be activated with the help of higher weights and this activation function. This can again be explained in a biological sense with an example that when a needle inserted into a human body, the neurons in the corresponding insertion site are activated so that the humans can feel the pain in that part of their body and can respond to that particular stimuli. In the ANN, the activation functions have the same purpose, which is activating the neurons with suitable mathematical operations. This particular operation that is with respect to weights and activation functions are applied in each and every neuron. At the end of this process, the output z of the neurons is passed to the output layer. After it is getting passed to the output layer, there is also a weight assigned to that particular output layer, and these particular weights will also get multiplied by z, and then an activation function will be applied to z again. As an outcome of this learning process, also known as forward propagation, the estimated parameters, which can be used for different applications, will be extracted from the output layer.

There are various activation functions described in the machine learning literature. The Sinusoidal, and Tanh, which are used in the context of this thesis, will be explained in this section, and they are the most commonly used activation functions in neural networks. The Tanh activation function, also known as Threshold activation function, which is sketched in Figure 1.16, will transform the value of an input into an output value ranging between -1 and 1. On the other hand, the Sinusoidal activation function, shown in 1.17, is a function that is based on sine function, and therefore, shows a periodic behavior.

It is important to note that the output of the ANN, the predicted parameters, need to be compared with the actual or target parameters. To compare these parameters, a function called Loss Function is employed. The Loss can be considered as an error, as in the case of the numerical analysis. The Mean Squared Error is one of the most widely used Loss Functions, which calculates the average (mean) of the squared difference between the actual parameters

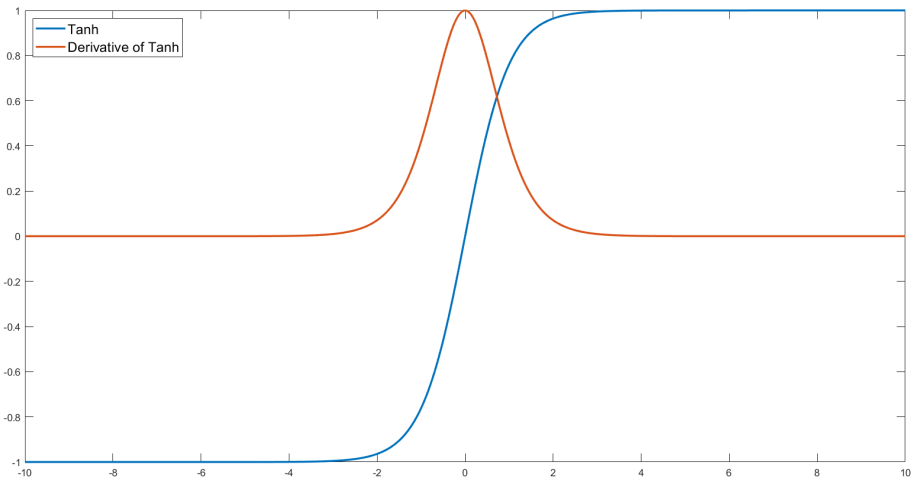

Figure 1.16: The figure of Tanh activation function and its derivative

and predicted parameters. It is expected that this Loss value has to be reduced in order to achieve an accurate learning process. Decreasing the value of Loss can be attained by updating the weights of the neurons with the help of optimizers. There are several optimizers, which will be explained later in greater detail, used in the context of ANN. Some of them are Gradient Descent, Stochastic Gradient Descent (SGD), Adam, Adagrad, AdaDelta, and RMSProp [Kingma and Ba, 2015, Duchi et al., 2012, Zeiler, 2012, Ruder, 2016], and each optimizer will try to reduce the Loss value. In order to reduce this value, a process called Backpropagation is performed in which the weights of the neurons are updated. As seen in Equation 1.10, the weights will get updated in such a way that the product of the learning rate and the gradient of the Loss Function with respect to the corresponding weights that are updated is subtracted from the previous value of the weights. The learning rate, which is similar to the stepping parameter in the context of Newton iterations, takes a value between 0 and 1 and determines the speed of the movement toward a global minimum of a function. Once the weights are updated, the forward propagation will be conducted again, unless, for some number of epochs, also known as iterations (one forward propagation plus one backward propagation), this Loss value can be completely reduced with the help of optimizers. The optimizers are used to find out the derivative of the Loss Function with

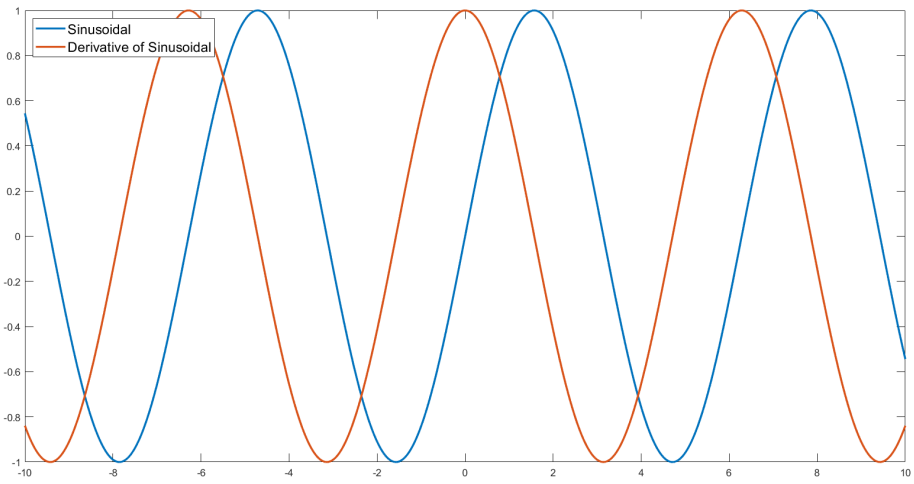

Figure 1.17: The figure of Sinusoidal activation function and its derivative

respect to the weights.

$$w_{i_{new}} = w_{i_{old}} - \eta \frac{\partial L}{\partial w_i} \tag{1.10}$$

Since the ANN may contain several hidden layers and several neurons, the backpropagation algorithm is performed by computing the gradient of the Loss Function with respect to the specific weights by means of the chain rule, calculating the derivative of the Loss Function with respect to the output values of each neuron and multiplying this with the derivative of the output values with respect to the specified weights due to the fact that output of each neuron depends on the weights assigned to that particular neuron.

It is also important to understand the concepts of Dropout and Regularization in the scope of ANN. In the case of the neural network having multiple layers and several neurons, there will be a huge number of weight parameters. These high amounts of weight and bias parameters will cause an issue called overfitting, which means that the model exactly predicts all the training data, however, it fails to predict the future data or the test data that have not been observed yet. Overfitted models have a low bias (error in the training data), but on the other hand, they have higher variance (error in the test data), which means that the data are scattered largely across the test data. There are two ways to solve this overfitting problem. One of them is using regularizers such as L1 and L2, which add a penalty term to the existing Loss Function to penalize for the large

values of weights. L1 regularizer is used to penalize the sum of absolute values of the weights, while L2 regularization is utilized in order to penalize the sum of squares of the weights. The penalty term is a function of weights, and therefore, this penalty term will be significantly large due to the large values of weights. As a consequence of the large penalty term, the Loss Function will also become relatively large. Optimizers will try to update the weights such that they can get smaller values, and thus Loss Function can be minimized thanks to the reduced value of penalty term with the smaller values of weights. The other method to solve the overfitting problem is implementing Dropout regularization [Srivastava et al., 2014]. In this method, a dropout ratio between 0 and 1 is selected, which helps to randomly drop (remove) some of the features from the input layer and some of the neurons from the hidden layers based on this ratio. The Dropout regularization allows the neural network to use the neurons more efficiently and not to rely on any specific neuron.

In this section, it is important to address the better ways to initialize weights and the various techniques that researchers usually apply in the deep learning model with respect to the weight initialization. Choosing the weights in a proper way will prevent the neural network from encountering vanishing and exploding gradient problems. The first thing that should be emphasized is that the weights should be selected as small values in order not to cause an exploding gradient problem. Secondly, weights should not be initialized to the same values. If the weights are set to the same values, the neurons will be performing in the same way, which means that the neurons will be learning from different features in a similar way, and this is not a desired feature from neurons in the ANN. As it is mentioned, each neuron learns new information from input features. Therefore, there should be enough difference between the weights such that the learning process to be performed more efficiently in a way that neurons do not give the same output. There are several ways to initialize the weights. In the first technique, the weights are initialized using Uniform Distribution, as shown in Equation 1.11, where fan_{in} represents the inputs that are passed to a particular neuron. In this method, the weights are sampled from a Uniform Distribution, which describes a probability distribution between a lower bound and an upper bound in which all outcomes have an equal probability. Sampling from Uniform Distribution is suitable for use in combination with the Sinusoidal and Tanh activation functions. The second technique is Xavier/Glorot Distribution, which has two

subclasses, known as Xavier Normal and Xavier Uniform. In Xavier Normal, also called Glorot Uniform, the weights are sampled from a Normal Distribution with a zero mean and a specific standard deviation, which is defined in Equation 1.12, where fan_{out} indicates the outputs from a particular neuron. In Xavier Uniform, also known as Glorot Uniform, the weights are initialized from a Uniform Distribution within limits shown in Equation 1.13. Xavier/Glorot Distribution techniques are also suitable for use in combination with the Sinusoidal and Tanh activation functions.

$$w \sim Uniform\left[\frac{1}{-\sqrt{fan_{in}}}, \frac{1}{\sqrt{fan_{in}}}\right] \tag{1.11}$$

$$w \sim N\left(0, \sigma\right), \qquad \sigma = \sqrt{\frac{2}{fan_{in} + fan_{out}}} \tag{1.12}$$

$$w \sim Uniform\left[-\sqrt{\frac{6}{fan_{in} + fan_{out}}}, \sqrt{\frac{6}{fan_{in} + fan_{out}}}\right] \tag{1.13}$$

As it is discussed before, in this section, the different types of optimizers will be explained in greater detail. It is important to remember that the main purpose of the optimizers is to update all the weights in such a way that the relative difference between the actual and predicted parameters can be reduced. To begin with, the main goal of the Gradient Descent optimizer is to find the global minimum of the function by iteratively moving in the direction of the steepest slope with the help of a negative gradient of the function at the present point. Unless the global minimum cannot be reached, the updating of the weights will be performed by means of backpropagation. In the weight update formula, which is given in Equation 1.10, the most important part is to find out the derivative of the Loss Function with respect to specific weights. If all the data points of the training data set are considered in order to find the derivative of the Loss Function with respect to particular weights, the technique that will be used for convergence is called Gradient Descent. If only a single data point at a time is considered, then the technique is known as Stochastic Gradient Descent. If a random subset of the training data set is considered, the technique is identified as Mini Batch Stochastic Gradient Descent, where batch specifies the subset of the training data set. In the case of the training data sets having a large

number of records, the computational cost of the Gradient Descent optimizer will be significantly larger. Therefore, the purpose of Mini Batch Stochastic Gradient Descent is, instead of using the whole data set, working on a subset of the training data set so that computational cost can be reduced, and iterations can be much faster due to less data being investigated. However, because of the stochastic nature of the Stochastic Gradient Descent and Mini Batch Stochastic Gradient Descent Optimizer, the convergence will take much more time compared to the Gradient Descent method. One of the most important problems encountered in the optimization process is that because of the large number of the weight parameters, hidden layers, and neurons, a non-convex Loss Function may be observed. This Loss Function may have several local minimum and saddle points where the derivative of the Loss Function is zero. Therefore, it is highly possible that optimizers may capture the local minimum instead of converging to the global minimum, and this will cause the update of the weights not being performed efficiently. To solve this problem, there are different optimizers that are developed by researchers such as Stochastic Gradient Descent with Momentum, Adam, Adagrad, AdaDelta, and RMSProp. Stochastic Gradient Descent with Momentum is an enhanced version of Stochastic Gradient Descent, and it tries to remove the noisy data encountered in Stochastic Gradient Descent caused by stochastic nature of the optimizer [Qian, 1999]. In the Stochastic Gradient Descent with Momentum optimizer, a momentum term which is derived using the Exponentially Weighted Moving Average added into the Stochastic Gradient Descent algorithm. Exponentially Weighted Moving Average is an approach used for assigning weights in a sequence in an exponentially decreasing order. In summary, the purpose of Stochastic Gradient Descent with Momentum is to reduce the noise in the data and increase the speed of movement in the direction of the steepest slope in the optimization process. This momentum term also helps the optimizer to reach the global minimum point of the Loss Function instead of capturing the local minimum. Furthermore, in the Gradient Descent, Stochastic Gradient Descent, and Mini Batch Stochastic Gradient Descent the learning rate is the same for all the neurons along with all the hidden layers, and in all of these optimizers, the weights are getting updated by using the same learning rate at every epoch. Therefore, the idea behind the Adagrad, Adaptive Gradient Descent, optimizer is to use different learning rates for each neuron along with the hidden layers at different epochs and to update the weights accordingly.

Whenever there is a data set that will be solved using an ANN, in that data set, there will be two types of features. The first type is called Dense, and the second feature is identified as Sparse. In the Sparse feature, most of the values are zero. In the case of the Dense feature, most of the values are non-zero. When a learning rate is initialized at the beginning of the training process, and both Sparse and Dense features exist in the training data set, by using this initial learning rate, it is not possible to update the weights based on these Dense and Sparse features properly. For the Sparse feature, it is required to have a different learning rate since most of the values here are zero. In the case of the Dense feature, it is also required to have a different learning rate. Therefore, the main concept behind the Adagrad optimizer is to use different learning rates with respect to different features, different weights, and different epochs. Based on Adagrad optimizer, the previous weight update formula, given in Equation 1.10, takes a new form, as shown in Equation 1.14, where t species the t-th epoch, and the η'_t indicates the adaptive learning rate. According to the idea behind Adagrad optimizer, the η'_t value should change with respect to weights and different epochs. The η'_t is defined in Equation 1.15, where ϵ represents a small positive number in order to avoid division by zero. The α_t, which is described in Equation 1.16, will get high values as the iteration goes on since it depends on the sum of the square of the Loss Function with respect to weights over the number of epochs. Because α_t is in the denominator in Equation 1.15, the η'_t will take small values due to high values of α_t, which means that the learning rate will start to decrease as the iterative process goes on. When the learning rate decreases, according to the weight update formula given in Equation 1.14, the weights will decrease slowly, which signifies that the convergence process will be performed very efficiently. Based on Sparse and Dense features, the Adagrad optimizer is able to apply different learning rate parameters so that the optimization process can reach the global minimum effectively. In conclusion, when the Adagrad optimizer is implemented, the learning rate will change with respect to different layers and different features. As the iteration continues, the learning rate will keep changing. That is why it is called the Adaptive Gradient Descent optimizer. However, one of the disadvantages of Adagrad optimizer is that it is possible that the α_t can become a very high number, and as a result, the learning rate can decrease significantly, indicating that the convergence process can take an extremely long time. To fix the issue of the

learning rate becoming significantly small, the AdaDelta and RMSProp optimizers are introduced. In AdaDelta and RMSProp optimizers, in order to prevent the learning rate from getting very small numbers, the α_t is replaced with a parameter called Exponential Weighted Average, E, which is given in Equation 1.17 and the idea behind it is very similar to the Exponentially Weighted Moving Average described in Stochastic Gradient Descent with Momentum optimizer. This Weighting function, with the help of parameter γ, which is commonly chosen in the range 0.9-0.95, tries to restrict the square of the gradient of the Loss Function taking extremely large values so that the learning rate cannot decrease to significantly small values in each iteration. Finally, the Adam optimizer, which is one of the most prevalent optimizers for deep learning, uses both the Momentum concept in the Stochastic Gradient Descent with Momentum optimizer and the idea of the Adaptive learning rate similar to AdaDelta and RMSProp.

$$w_t = w_{t-1} - \eta'_t \frac{\partial L}{\partial w_{t-1}} \tag{1.14}$$

$$\eta'_t = \frac{\eta}{\sqrt{\alpha_t + \epsilon}} \tag{1.15}$$

$$\alpha_t = \sum_{i=1}^{t} \left(\frac{\partial L}{\partial w_i} \right)^2 \tag{1.16}$$

$$E_t = \gamma E_{t-1} + (1-\gamma) \left(\frac{\partial L}{\partial w_t} \right)^2 \tag{1.17}$$

1.3 Physics Informed Neural Networks (PINN)

In order to eliminate the need for training data, an alternative method has been found to make the supervised ANN semi- or even unsupervised. Physics-informed Neural Networks are algorithms that use the auto-differentiation feature of backpropagation [Raissi et al., 2017], [Raissi et al., 2019]. During the gradient calculation in the backpropagation stage, one can calculate the gradients of the output with respect to the inputs by using the chain rule. Therefore, at any step, one has access to the values of the derivative from the network output with respect to the network input without extra computational

effort, and this can be used to implement the PDE, the Heat Equation, that is of particular interest in the context of this thesis. This algorithm can be used to reduce the required training data near 0. Instead of using labeled training data, PINNs utilize collocation points in order to calculate the PDE loss, boundary, and initial points to evaluate the boundary and initial loss, respectively. In order to understand how the loss function has been built up, in the following section, equations describing the loss function have been shown using the example of the Heat Equation. An example of PINN in the case of Heat Equation is shown in Figure 1.18. As it can be seen in Equation 1.18, the PDE loss has 3 components, the temperature gradient over time, the conduction term, and the heat source which is evaluated all over the domain. The PDE loss is evaluated at the black points, also known as collocation points, as shown in Figure 1.19 for the 1D case. In order to solve the PDE, one also needs the boundary conditions and the initial condition. For Neumann insulated or Dirichlet with a given temperature boundary conditions, the boundary loss (Equation 1.19) is evaluated at the edge locations, grey dots, seen in Figure 1.19. Similarly, the initial loss is evaluated at the red dots when the time is equal to 0. What is special about PINN is that it is a meshless method, and therefore, one can easily include support data from experiments or FE simulations that are not prebound by any predetermined mesh. All these loss functions can be combined into one total loss function by giving it two hyperparameters, one is called λ_{PDE} and the other one is called $\lambda_{Variables}$. Here, it has been decided to define two different hyperparameters, because it is important that in the loss function all components have the same magnitude, in other words, some components should not have higher importance than other components. Furthermore, the most crucial point is that these hyperparameters can be tuned later using Ensemble Training in order to find a balance between the PDE loss, and initial and boundary conditions.

$$L_{PDE} = \rho c_p \frac{dT}{dt} - k\frac{d^2T}{dx^2} - k\frac{d^2T}{dy^2} - k\frac{d^2T}{dz^2} - q(x,y,z) \tag{1.18}$$

$$\begin{aligned} L_{Boundary\,Neumann} &= \frac{dT(\underline{x} \in b)}{d\underline{x}} \\ L_{Boundary\,Dirichlet} &= (\underline{x} \in b) - T \end{aligned} \tag{1.19}$$

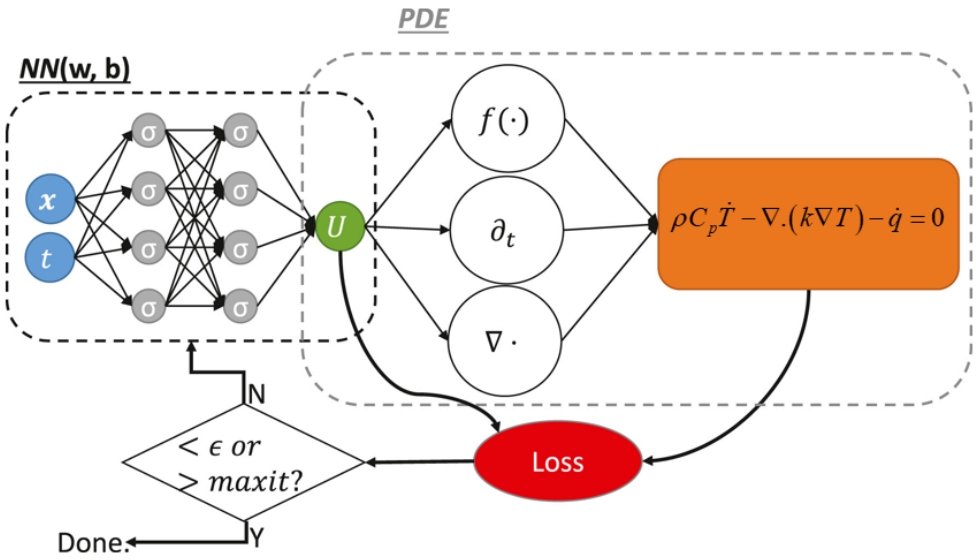

Figure 1.18: An example of PINN in the case of Heat Equation

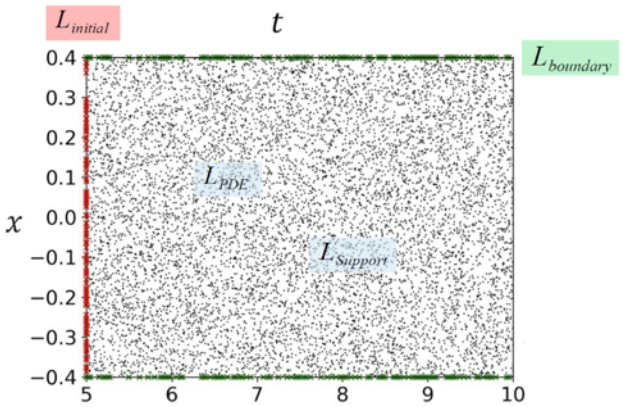

Figure 1.19: Initial, Collocation and Boundary Points [Alber et al., 2019]

$$L_{Initial} = T(t=0) - T_0 \tag{1.20}$$

$$L_{Support} = T(t, \underline{x}) - T_{data}(t, \underline{x}) \tag{1.21}$$

$$Total\ Loss = \lambda_{PDE} L_{PDE} + \lambda_{Variables}[L_{Boundary} + (L_{Initial} + L_{Support}] \tag{1.22}$$

The PINN algorithm that has been used in this thesis was based on the code developed by [Mishra and Molinaro, 2021].

Chapter 2

Simulation Setup, Results and Discussion

As it is mentioned before, in this thesis, the main focus is using Neural Networks, mainly Physics-informed Neural Networks, as a surrogate model for the thermal simulation of the LPBF additive manufacturing process.

2.1 The Problem Statement

The original problem statement, as shown in Figure 2.1 contains a moving heat source with multiple tracks and multiple layers in the 3D simulation domain. However, during the course of this thesis, the original problem definition has changed, and the main focus has been on the investigation of the case with a moving heat source with a single track and a layer that includes temperature-dependent material properties. The powder phase has been neglected for computational simplicity. The material that has been considered in the simulation setup is Steel, and it has the density of 8220 $\frac{kg}{m^3}$, and the temperature dependent conductivity and temperature dependent heat capacity as shown in Figure 2.2 and Figure 2.3, respectively.

The process parameters that have been used in the context of this thesis have been summarized in the table below. It is important to note that in order to represent the Laser Heat Source, the Goldak heat source model described in the previous chapter has been used.

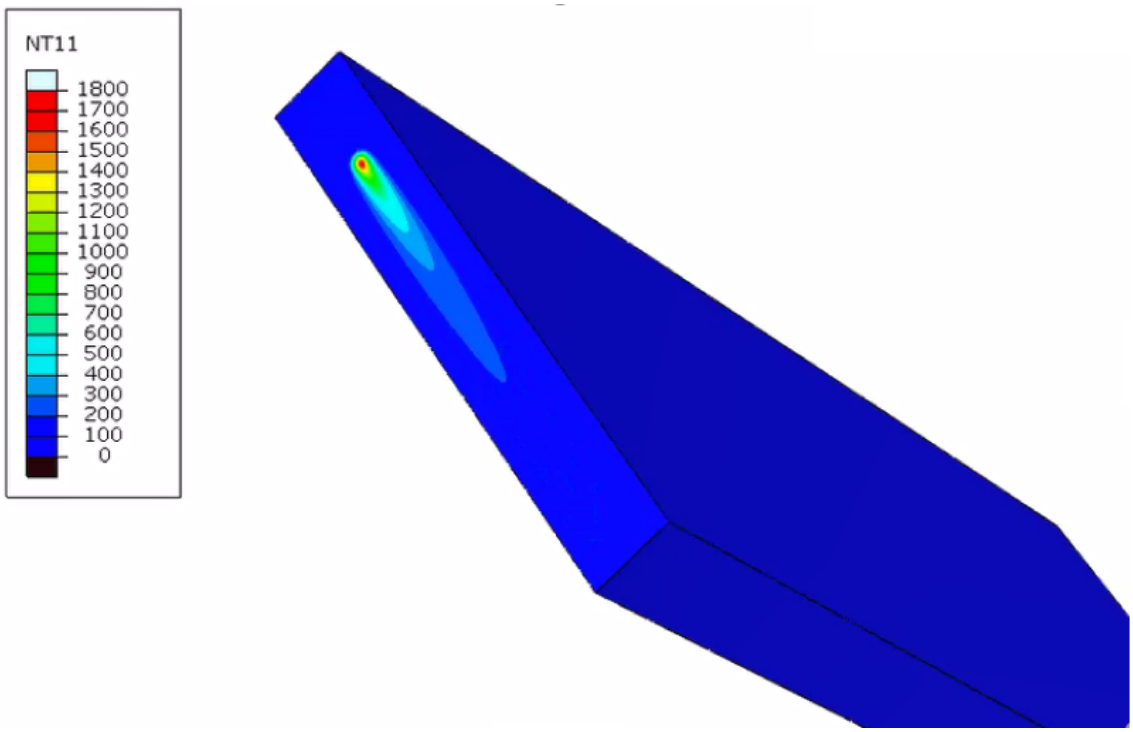

Figure 2.1: The original problem statement

Process Parameters	**Value**
Domain Size in x direction	$x \in [-1, 1.8]mm$
Domain Size in y direction	$y \in [-1, 1]mm$
Domain Size in z direction	$z \in [-1, 0.03]mm$
Laser Source	spherical Goldak
Goldak Radius σ	0.05mm
Penetration Depth in all directions	0.05mm
Laser Power	200W
Absorption β	50%
Starting Point	$x_0 = 0$
Laser Speed	1000 $\frac{mm}{s}$

Table 2.1: Process Parameters

2.2 Scaling of the PDE Components

Before starting using the PINN algorithm, it is important to scale the loss functions and all the variables in the range of -1 to 1, since Neural Networks can train better when variables in the domain are in the range of -1 to 1. The loss function has been scaled in the domain -1 to 1 with respect to time and temperature since in the simulation domain that is of particular interest in the context of this thesis, the temperature values go up to 3000 degrees and time values go up to 0.0015s. For the spatial domain, no specific

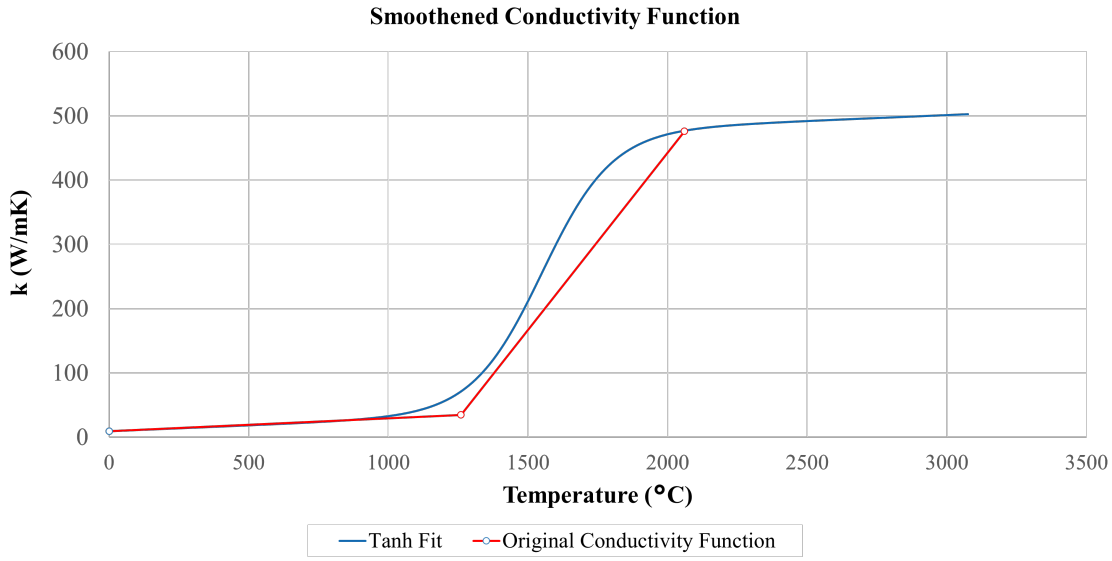

Figure 2.2: Temperature Dependent Conductivity

normalization has been considered since the spatial dimensions are already in the domain that the network can handle. However, it is important to mention that, for some cases, there is the benefit of also non-dimensionalizing the spatial domain. As it can be seen from the following equations, at the end of non-dimensionalizing, Equation 2.9 has been obtained. Here, the source term, the q, is the only component that is affected by the T_{max} normalization factor, the maximum value of the temperature. In order to make the heat source have the same scale as the other components, the time gradient, and the conduction term, and also to make the network easier to train, specific considerations had to be taken in to find a suitable T_{max} value, even though T_{max} has been known from the FE simulations.

$$\frac{\partial E}{\partial t} + \nabla.(\vec{u}E) = \nabla.(\alpha \nabla E) + \dot{Q} \tag{2.1}$$

The convective thermal energy term in the above equation can be neglected, and the equation can be rewritten using the following definitions. It is important to mention that, here, temperature-dependent conductivity and heat capacity has been utilized.

$$\partial E = C_p \partial T \tag{2.2}$$

$$\alpha = \frac{k}{\rho C_p} \tag{2.3}$$

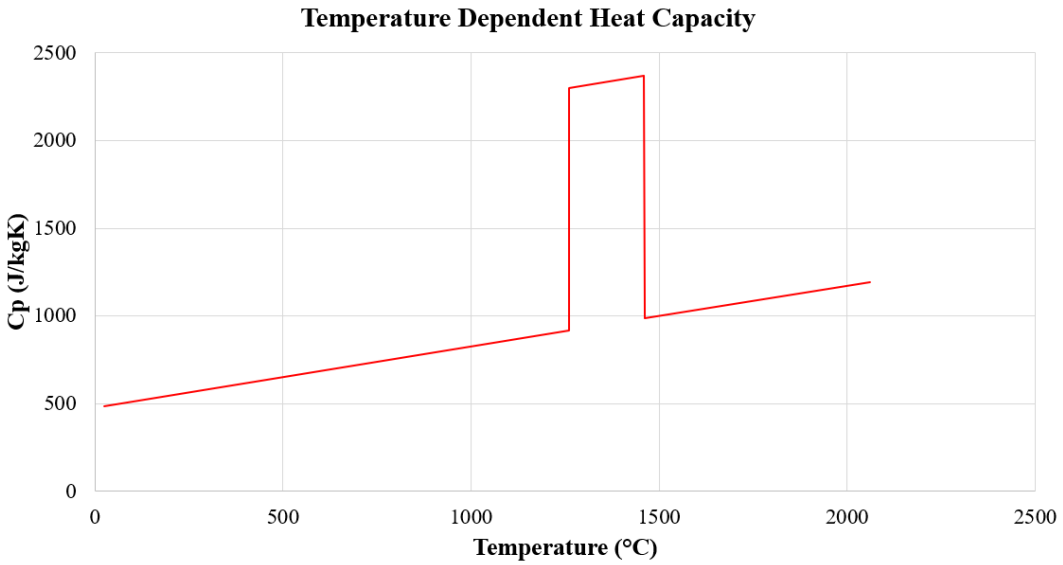

Figure 2.3: Temperature Dependent Heat Capacity

$$\begin{aligned} \rho\frac{dT}{dt} &= \frac{d}{dx}\left(\frac{k}{c_p}\frac{dT}{dx}\right) + \frac{d}{dy}\left(\frac{k}{c_p}\frac{dT}{dy}\right) + \frac{d}{dz}\left(\frac{k}{c_p}\frac{dT}{dz}\right) + q \\ \rho\frac{dT}{dt} - \frac{d}{dx}\left(\frac{k}{c_p}\frac{dT}{dx}\right) - \frac{d}{dy}\left(\frac{k}{c_p}\frac{dT}{dy}\right) - \frac{d}{dz}\left(\frac{k}{c_p}\frac{dT}{dz}\right) - q &= 0 \end{aligned} \tag{2.4}$$

$$\begin{aligned} T &\in (0, T_{max}) \\ t &\in (0, t_{max}) \end{aligned} \tag{2.5}$$

The above equations have been non-dimensionalized using the relationships

$$\begin{aligned} T &= T_{max}T' \\ t &= t_{max}t' \end{aligned} \tag{2.6}$$

where T' representing the non-dimensional temperature and t' representing the non-dimensional time.

$$\begin{aligned} T' &\in (0, 1) \\ t' &\in (0, 1) \end{aligned} \tag{2.7}$$

$$\begin{aligned} T &= T_{max}T' \\ t &= t_{max}t' \end{aligned} \tag{2.8}$$

$$\begin{aligned}
&\frac{T_{max}}{t_{max}}\frac{dT'}{dt} - \frac{T_{max}}{\rho}\left[\frac{d}{dx}\left(\frac{k}{c_p}\frac{dT'}{dx}\right) + \frac{d}{dy}\left(\frac{k}{c_p}\frac{dT'}{dy}\right) + \frac{d}{dz}\left(\frac{k}{c_p}\frac{dT'}{dz}\right)\right] - \frac{1}{\rho}q = 0 \\
&\frac{dT'}{dt} - \frac{t_{max}}{\rho}\left[\frac{d}{dx}\left(\frac{k}{c_p}\frac{dT'}{dx}\right) + \frac{d}{dy}\left(\frac{k}{c_p}\frac{dT'}{dy}\right) + \frac{d}{dz}\left(\frac{k}{c_p}\frac{dT'}{dz}\right)\right] - \frac{t_{max}}{\rho T_{max}}q = 0
\end{aligned} \tag{2.9}$$

2.3 Test Cases

In order to understand what the PINN algorithm is capable of, first, several different test cases have been investigated with an increasing level of complexity. This approach has given a clear idea about how to approach the problem that is of the main focus in the context of this thesis.

2.3.1 2D Case without Heat Source and without any temperature dependent material properties

First, a very simple 2D rectangular block without Heat Source and without any temperature-dependent material properties, meaning constant conductivity, heat capacity, and density, has been established. As shown in Figure 2.4, Neumann boundary conditions have been employed in three walls of the rectangular block, and Dirichlet boundary condition with a constant temperature of $1°C$ has been employed at the bottom wall. The temperature range has been kept in the range of 0 to 1 in order to make this case even more simple. In Figure 2.5, the thermal simulation result using both FE and PINN has been shown. As mentioned before, the FE simulation result has been considered as the ground truth, and the PINN result has been compared with the FE result, as shown in Figure 2.6. Here, a pointwise error definition has been used. According to this definition, the temperature value calculated using PINN for each node at FE has been compared with the temperature value extracted from FE for that respective node. The L2 error and Relative error definitions can be seen in Equations 2.10 and 2.11. These definitions have also been used in the following cases. As it can be seen from Figure 2.6, where the results of the last time step have been shown, the PINN has successfully predicted the temperature values with a relative error of 0.4%.

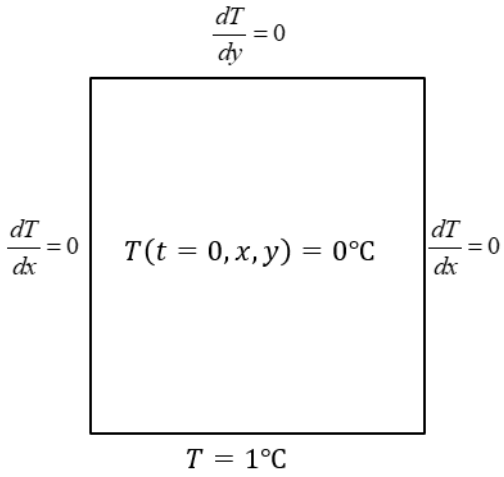

Figure 2.4: 2D Case without Heat Source and without any temperature dependent material properties

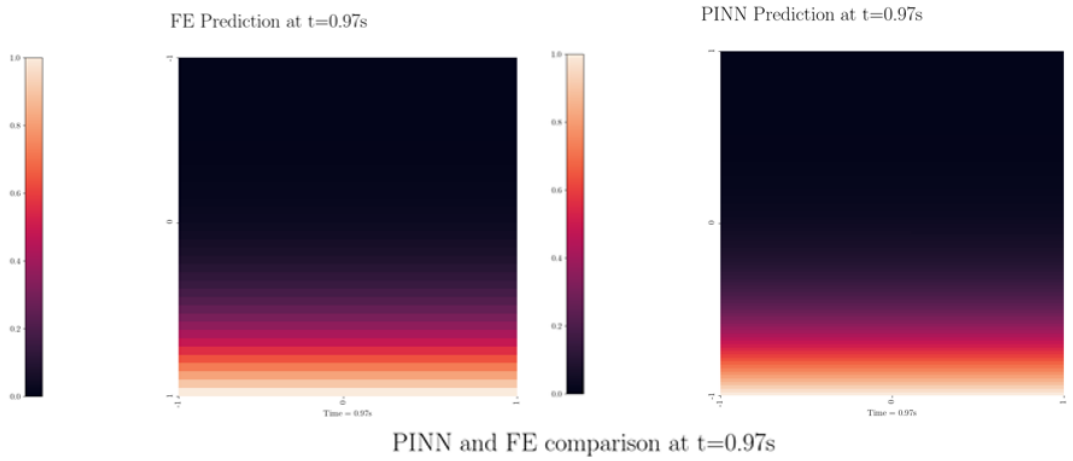

Figure 2.5: FE and PINN comparison for the 2D Case without Heat Source and without any temperature dependent material properties

$$L2\ Error = \sqrt{\sum_{i=1}^{N} (T_{i,PINN} - T_{i,FEM})^2} \tag{2.10}$$

$$Relative\ Error = \frac{\sqrt{\sum_{i=1}^{N} (T_{i,PINN} - T_{i,FEM})^2}}{\sqrt{\frac{\sum_{i=1}^{N} (T_{i,FEM})^2}{N}}} \tag{2.11}$$

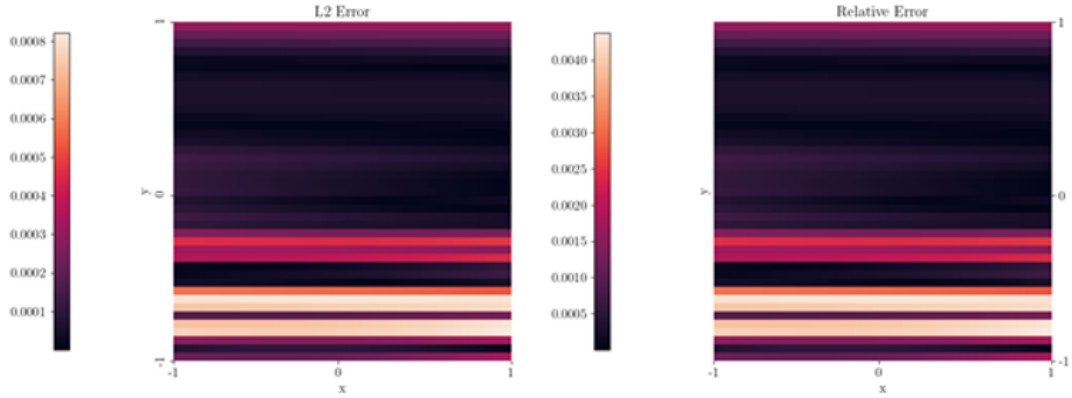

Figure 2.6: L2 and Relative Error for the 2D Case without Heat Source and without any temperature dependent material properties

2.3.2 2D Case with a diffused Heat Source and constant material properties

Secondly, as shown in Figure 2.7, a slightly more complicated case where a diffused heat source that increases the maximum temperature in the simulation domain up to 3500 degrees has been implemented at one of the walls of the 2D rectangular block. Similar to the previous test case, constant material properties have been employed. All the walls have been insulated, in other words, Neumann boundary conditions have been employed in all the walls of the rectangular block. As it can be seen from Figures 2.8 and 2.9, the PINN has managed to estimate the temperature values with a relative error of 0.8%.

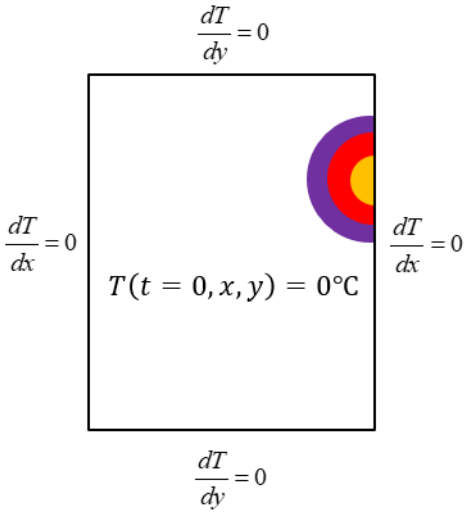

Figure 2.7: 2D Case with a diffused Heat Source and constant material properties

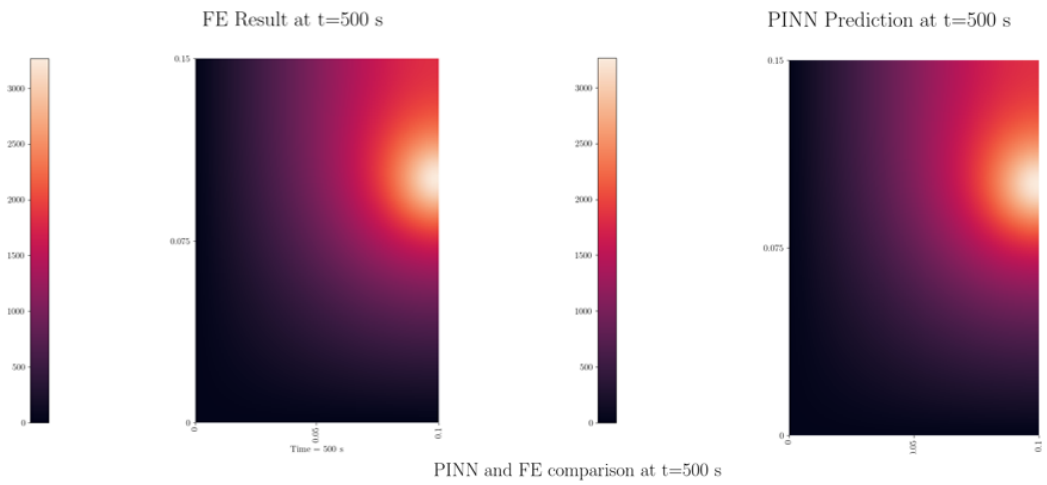

Figure 2.8: FE and PINN comparison at for the 2D Case with a diffused Heat Source and constant material properties

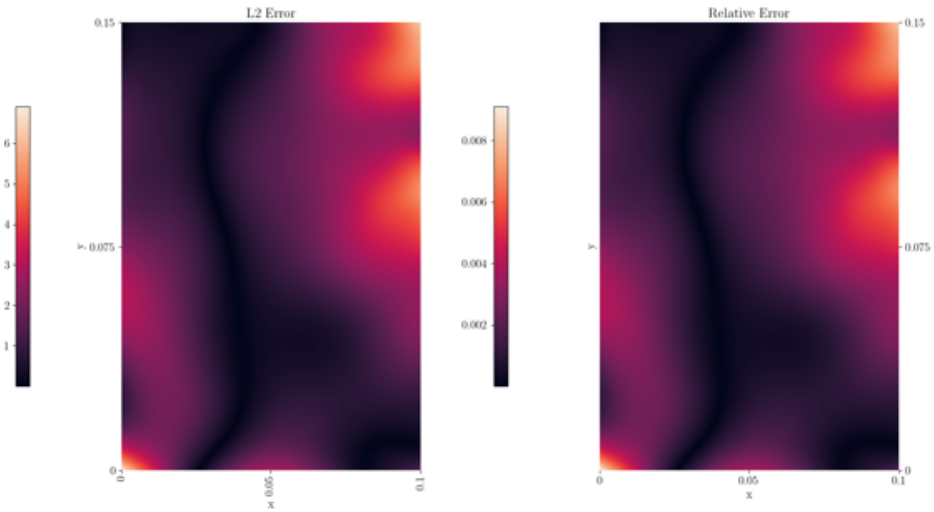

Figure 2.9: L2 and Relative Error comparison for the 2D Case with a diffused Heat Source and constant material properties

2.3.3 2D Case with time-dependent localized Heat Source and without any temperature-dependent material properties

In the third test case, again no temperature dependent material properties have been used; however, a localized Heat Source, similar to the laser source in the LPBF process, has been implemented at the center of the simulation domain, as shown in Figure. As it can be seen from Figure, the heat source has been active only for a very small part of the entire time domain (only 5% of the entire time domain). It is also important to note that the heat source has a very sharp gradient at the beginning and it causes the temperature values to go up

to 5500°C in 5 milliseconds, as shown in the animation at the following link `https://drive.google.com/file/d/1G-n6rvRAUiNc69jgTXBblKYOlmAalS6r/view?usp=sharing`. As soon as the heat source gets deactivated, the cooling phase starts, and temperature values go down to 100°C. In Figures 2.11 and 2.12, the results of FE and PINN prediction at 0.005s (at the time when maximum temperature occurs) can be observed. As it can be seen from the Figures 2.11 and 2.12, the maximum temperature that PINN predicts and the maximum temperature that FE shows is almost identical, but when the Relative Error data has been examined in Figure 2.12, it can be observed that at a certain part of the domain, the relative error goes up to 70%.

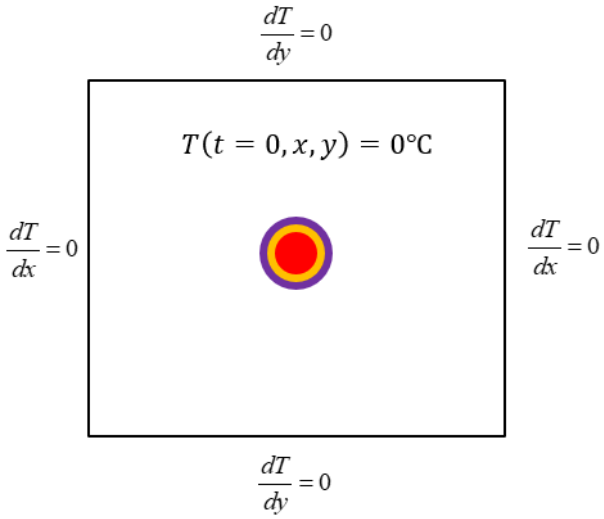

Figure 2.10: 2D Case with time-dependent localized Heat Source and without any temperature-dependent material properties

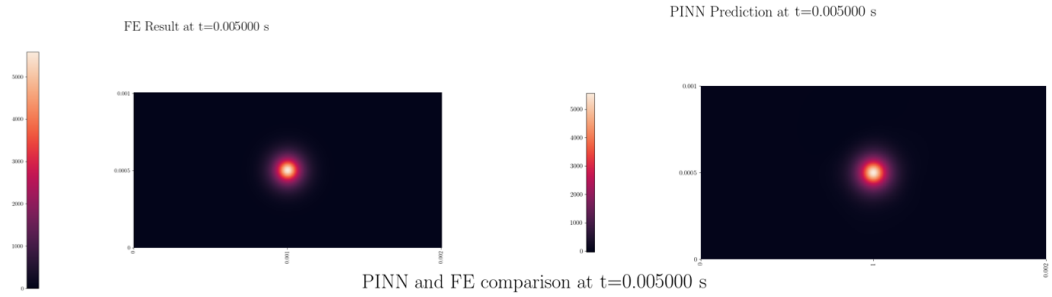

Figure 2.11: FE and PINN comparison at 0.005s for the 2D Case with time-dependent localized Heat Source and without any temperature-dependent material properties

When the results at the final time step (at 0.2s) have been investigated in Figures 2.13 and 2.14, it can be seen a similar trend as before in terms of the Relative Error.

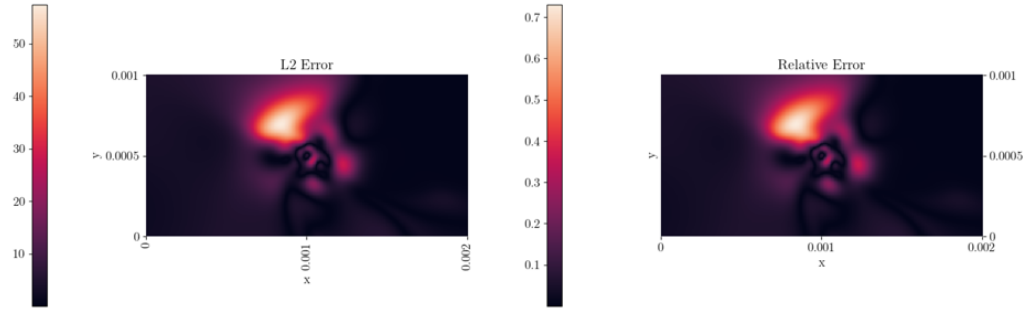

Figure 2.12: L2 and Relative Error comparison at 0.005s for the 2D Case with time-dependent localized Heat Source and without any temperature-dependent material properties

In this case, it can additionally be observed from Figures 2.13 and 2.14 that the PINN prediction is completely different than the FE result. The PINN shows issues predicting the temperature profile shape and the maximum temperature value accurately. Moreover, it even shows negative temperature results. Therefore, it can be concluded that the PINN prediction is absolutely unrealistic in this scenario.

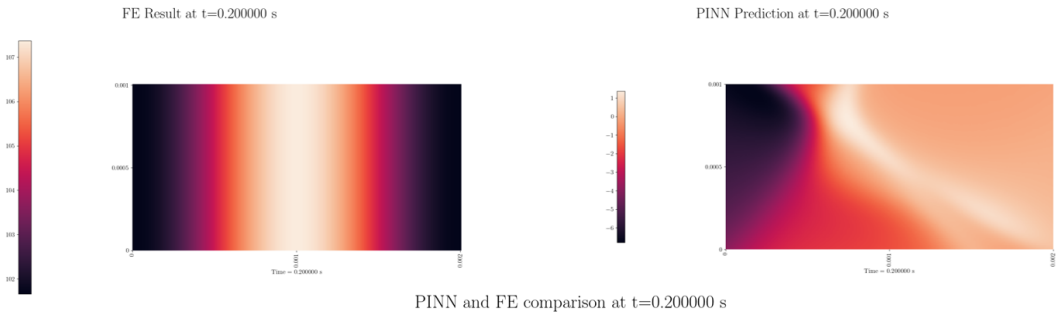

Figure 2.13: FE and PINN comparison at 0.2s for the 2D Case with time-dependent localized Heat Source and without any temperature-dependent material properties

In order to solve the issue encountered in the context of localized heat sources, a new collocation point sampling strategy has been developed. The algorithm of the new collocation point distribution can be seen below. According to this algorithm, more sampling points at the center of the domain have been accumulated in order to capture the physics of the heat source more accurately. At the same time, these bulk of points concentrated at the center of the domain has been spread out slowly when the heat source gets deactivated since the heat source is only active for 5 milliseconds. As it be can be seen in the Algorithm 1, the collocation points first have been generated in radial coordinates, and then converted into cartesian coordinates. The animation of this new collocation

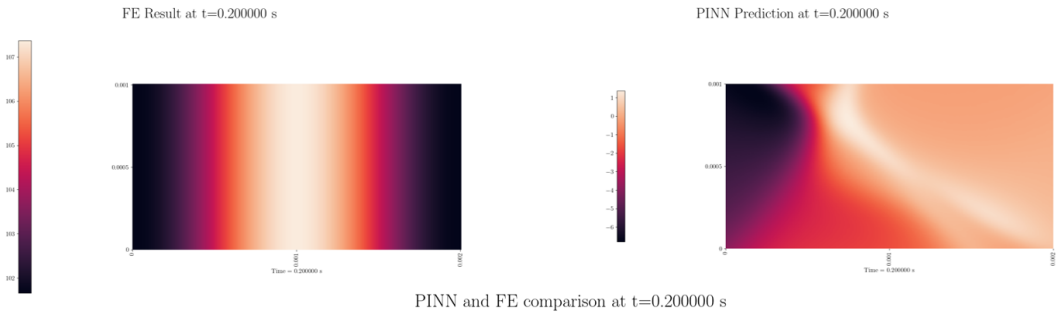

Figure 2.14: L2 and Relative Error comparison at 0.2s for the 2D Case with time-dependent localized Heat Source and without any temperature-dependent material properties

point distribution can be found at the following link `https://drive.google.com/file/d/1yDG7f_chOxxFMcemEqDEKsjWw3tC1k-2/view?usp=sharing`.

Algorithm 1 Algorithm for Time Dependent (not Moving) Collocation Points in 2D

$N_{samples} = 10N_{samples}$
for $k = 1 \rightarrow N_{samples}$ **do**
    $a = 1$
    $N_{random} \rightarrow$ sampled from Sobol sequence
    $radius[k,:] \rightarrow N_{random}[0]$
    $theta[k,:] \rightarrow 2\pi N_{random}[1]$
    $time[k,:] \rightarrow N_{random}[2]$
    $radius[k,:] = radius[k,:]^{a(1-time[k,:])+0.5}$
    $x[k,:] = \sqrt{2} radius[k,:] cos(theta[k,:])$
    $x[k,:] = x[k,:] + x_{locationofthelaser}$
    $y[k,:] = \sqrt{2} radius[k,:] sin(theta[k,:])$
    $y[k,:] = y[k,:] + y_{locationofthelaser}$
    Concatenate time, x and y
    Delete coordinates <0 and >1

After the implementation of this new collocation point sampling strategy, the PINN has been trained again in order to understand whether the new sampling strategy will result in highly accurate temperature values. In Figures 2.15 and 2.16, it can be seen that at time point 0.005s the Relative Error has been reduced from 70% to 8%. When the results at the end of time steps have been examined, it be can be seen from the Figures 2.17 and 2.18 is that the Relative Error has also been reduced, but it can be observed that the PINN still does not predict the temperature profile shape accurately, even though it predicts the temperature range very precisely. In order to solve this problem, Extended PINN, XPINNs, which is a domain decomposition approach in the PINN framework, has been implemented [Jagtap and Karniadakis, 2020]. In this approach, one can divide the space-time domain into several subdomains, and in each subdomain, a separate neural network can be employed. Similar to PINN, the boundary and initial conditions for each subdomain have to be implemented; however, in addition to the boundary and initial conditions,

the interface conditions which are used to stitch the decomposed domains together have to be employed. Since PINN has issues predicting the temperature profile at the final time step, XPINNs have been implemented by using the decomposition of the time domain into several subdomains. However, it has been observed that XPINNs had issues in the implementation of the Neumann boundary conditions, and therefore due to time constraints, the study of XPINNs had to be left for future research.

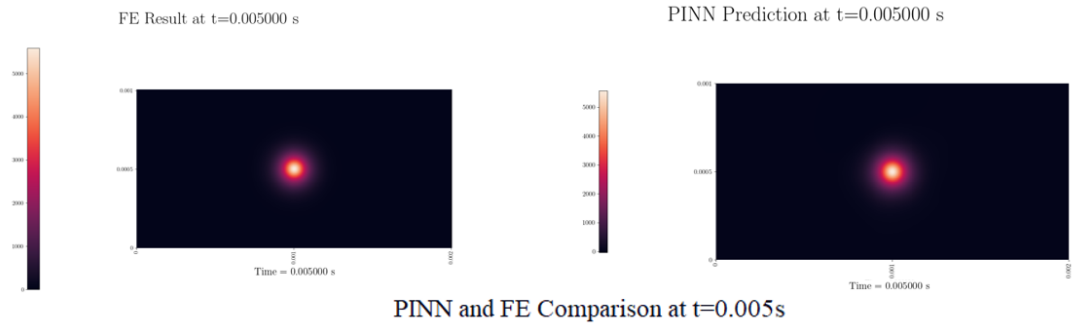

Figure 2.15: FE and PINN comparison at 0.005s for the 2D Case with time-dependent localized Heat Source and without any temperature-dependent material properties after new Collocation Point distribution

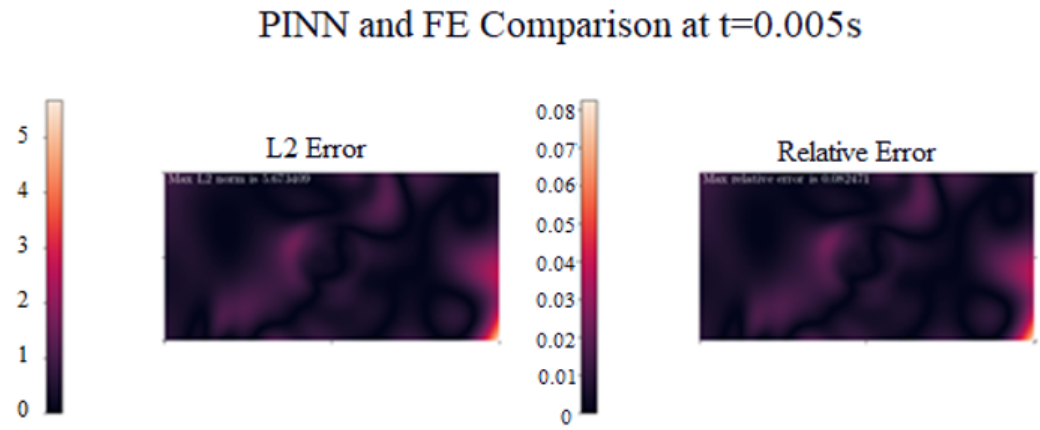

Figure 2.16: L2 and Relative Error comparison at 0.005s for the 2D Case with time-dependent localized Heat Source and without any temperature-dependent material properties after new Collocation Point distribution

2.4 The Simplified Problem Statement and Transfer Learning Approach

In the previous test cases that have been shown, no temperature-dependent material properties have been taken into account. When the temperature-dependent conductivity and heat capacity

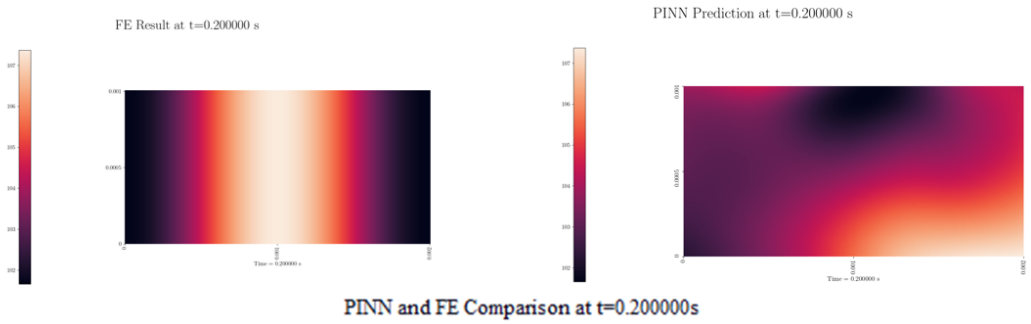

Figure 2.17: FE and PINN comparison at 0.2s for the 2D Case with time-dependent localized Heat Source and without any temperature-dependent material properties after new Collocation Point distribution

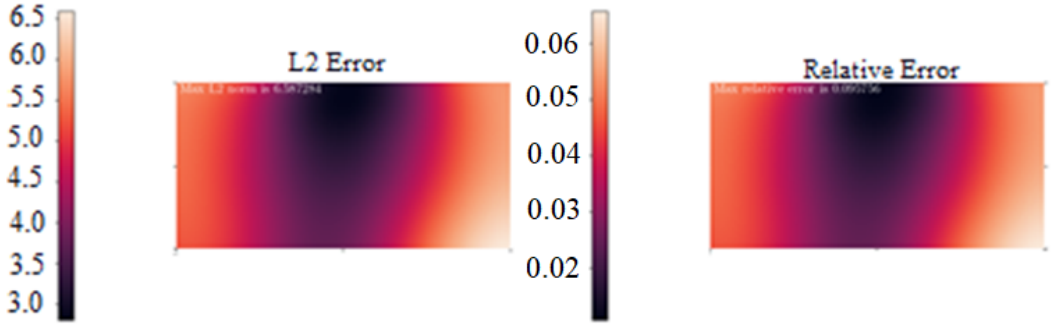

Figure 2.18: L2 and Relative Error comparison at 0.2s for the 2D Case with time-dependent localized Heat Source and without any temperature-dependent material properties after new Collocation Point distribution

have been implemented into the PINN algorithm; the network has struggled to train. As it can be seen in Figure 2.2 and Figure 2.3, there are some kinks and very sharp gradients in the graphs of conductivity and heat capacity lines, and usually, Neural Networks struggle to handle these kinds of kinks and sharp gradients or jumps, and as a result, the PINN prediction was absolutely unrealistic. Then in order to solve this issue, a step-by-step approach involving Transfer Learning has been developed.

In this approach, first, the network has been trained without any temperature-dependent material properties, so constant conductivity and heat capacity. At the end of the training, hyperparameter optimization/ensemble training has been conducted in order to find the best hyperparameters, best neural network architecture (number of layers, and neurons). Then the Transfer Learning approach has been used, and the model has been transferred, or in other words, the weights and biases from the network that has been trained with no temperature-dependent properties has been loaded to a model with temperature-dependent conductivity and constant heat capacity. It is important to mention that, in this approach, the exact temperature-dependent properties that has been showed in Figure 2.2 and Figure 2.3, has not been used. Instead, approximate functions or smooth fit

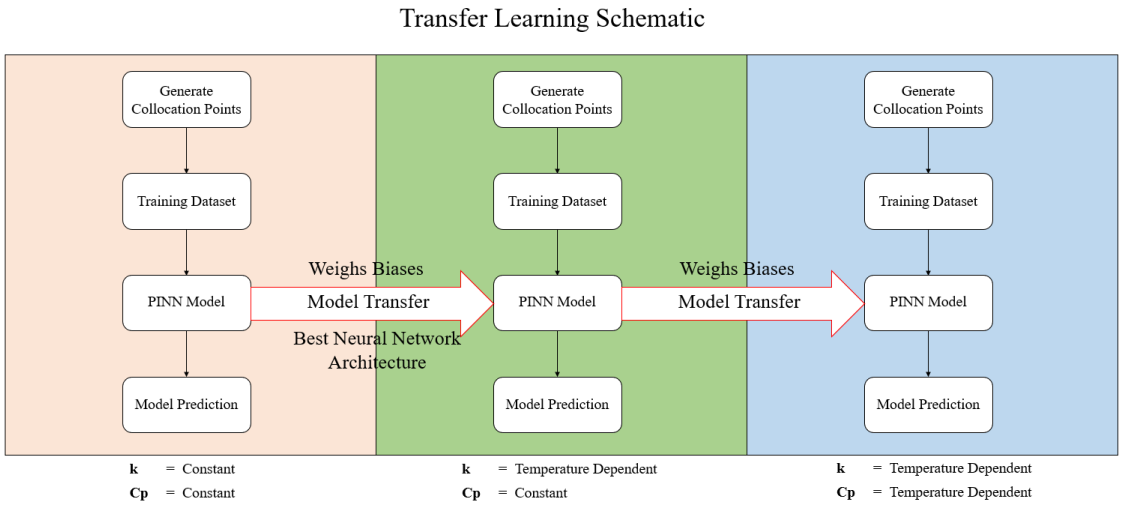

Figure 2.19: Transfer Learning Approach

that approximate these temperature-dependent properties have been used in order to avoid kinks and sharp gradients. As it can be seen from Figure 2.2, a Tanh function has been used in order to approximate the original conductivity function. Similarly, as shown in Figure 2.3, for the heat capacity, a Gaussian fit has been used in order to approximate the original heat capacity function. After the training of the second model has been done, the weights and biases of this network have been transferred to the network with temperature-dependent conductivity and heat capacity. The entire process used in this approach has been summarized in Figure 2.19. The one disadvantage of this approach is that the Ensemble Training in order to find the best Neural Network architecture in the 2nd and 3rd model cannot be used, because the weights and biases from the 1st model have been loaded to the 2nd model, and as a result same number of layers and neurons have to be used. However, one can still do Ensemble Training, but there are only very few hyperparameters that can be tuned, such as the lambda values that have been shown in Equation 1.22.

There has been another approach that has been used when the temperature dependent material properties were utilized, which did not yield any meaningful results. According to this approach, two different Neural Networks, as it can be seen in this Figure 2.22, one for conductivity, and one for heat capacity, have been used in order to find a smooth fit to the conductivity and heat capacity lines, in other words in order to smoothen the kinks and sharp gradients. The output of these two networks has been supplied into the original PINN algorithm. However, this approach did not work out. The reason behind the failure of this approach might be the complexity of using two Neural Networks, which depend on the outcome of the PINN algorithm.

The results of the first step (when no temperature material properties have been used) of the Transfer Learning approach at 5.5 milliseconds in Figures 2.23 and 2.24 have been shown. It can be seen from the Figures that a high accuracy has been achieved. Similarly, at 0.2s, as shown in

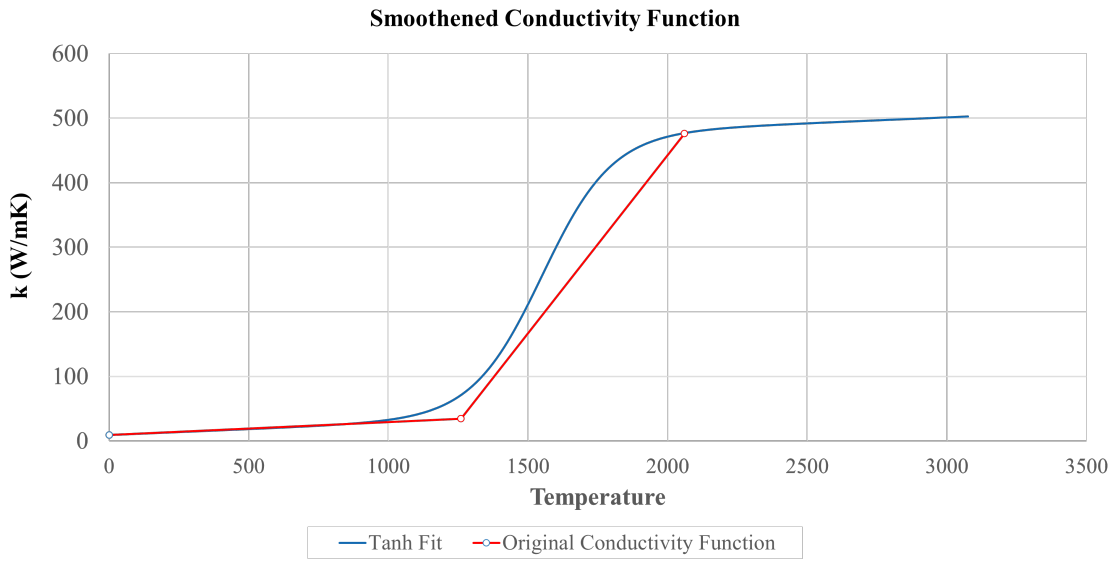

Figure 2.20: Smoothened Conductivity Function

Figures 2.25 and 2.26, low Relative Error-values have been observed.

When the results at the time when the maximum temperature values occur have been looked upon for the 2nd step of the Transfer Learning Approach, it can be concluded from Figures 2.27 and 2.28 that even though PINN has managed to predict the maximum temperature values quite accurately; at a certain part of the domain, there has been a relative error of 50%; however, when the L2 error Figure have been examined, it can be realized that the maximum discrepancy between the FE and PINN is only 30 degrees. Therefore, for these temperature values, it can be said that a 30-degree difference is acceptable. As it is mentioned before, Transfer Learning does not allow the utilization of Ensemble Training in order to find the best number of layers and neurons for the Neural Network, and therefore improving the accuracy of the results is a challenging task. As it can be seen from Figures 2.29 and 2.30, at the final time step a similar trend as before has been observed; however, in this case, the relative error is much lower, around 20%, and the maximum L2 error is 11 degrees. Therefore, again it can be concluded that the result is acceptable. Here, even though PINN has predicted the maximum temperature values with a decent accuracy, it can be observed that PINN has again struggled to predict the temperature profile shape at the final time step as before.

When the results of the 3rd step at the time when maximum temperature occurs have been checked from Figures 2.31 and 2.32, it can be realized that there is a similar trend as before and PINN has accurately predicted the maximum temperature; however, the relative error is again around 50% at a certain of the domain. Since the maximum L2 error is also around 30-degrees, here it can also be concluded that the PINN prediction is reasonable. As it can be seen from Figures 2.33 and 2.34, at the final time step, similar results as in the case of the 2nd step have been

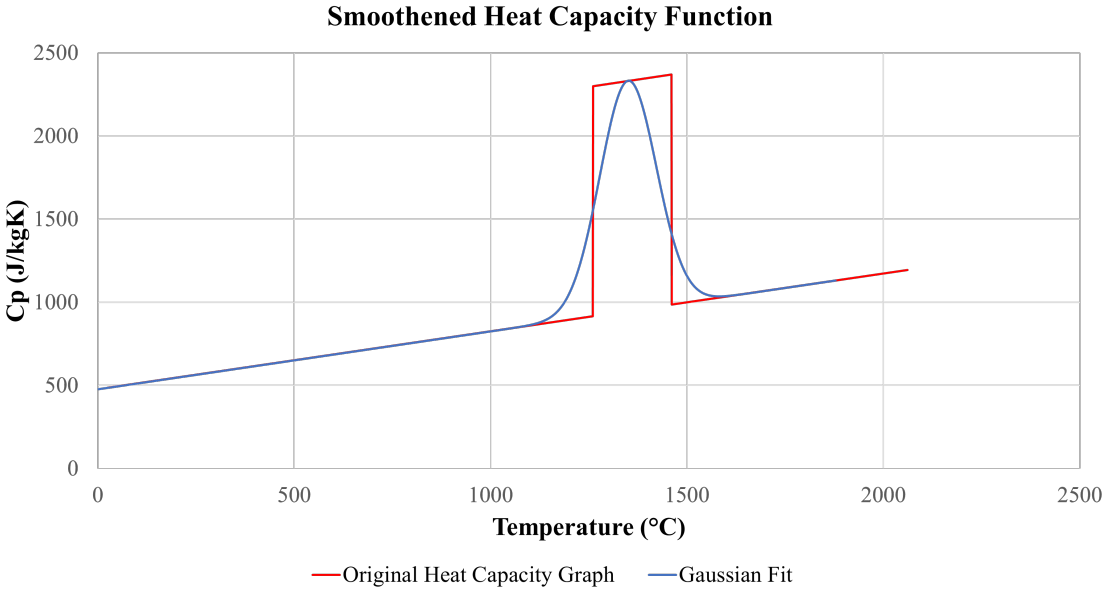

Figure 2.21: Smoothened Heat Capacity Function

observed. Again, there is a maximum relative error of 20%, 11-degree L2 error, and similar to the previous cases, PINN has struggled to predict the temperature profile shape accurately.

In order to improve the accuracy of the PINN algorithm several strategies have been developed. It has been observed that at the beginning of the training process, the Neural Network has sometimes produced negative Temperature predictions, and since the material properties are dependent on the network outcome, these negative temperature values have caused wrong predictions in conductivity and heat capacity. Therefore, in order to avoid the PINN predicting negative temperature values, two additional new loss functions L_{max} and L_{min} have been implemented, as shown in Equations 2.12 and 2.13. These new loss functions have helped the Network not to predict any negative temperature values and also values greater than the normalization temperature T_{max}. The 2nd approach that has been tried instead of implementing new functions into loss function was implementing a mask function which avoids the prediction of temperature values below 0. Both approaches have helped Neural Network to train better when temperature dependent material properties have been utilized.

$$L_{max} = max(max(temperature) - 1, 0) \tag{2.12}$$

$$L_{min} = max(-min(temperature), 0) \tag{2.13}$$

In order to improve the accuracy of the results, different sampling strategies for collocation points have also been studied. In the first sampling strategy, Ole Müller, MSc student in Mechanical Engineering at ETH Zürich, has used Moving Triangular Distribution that samples on the x-axis,

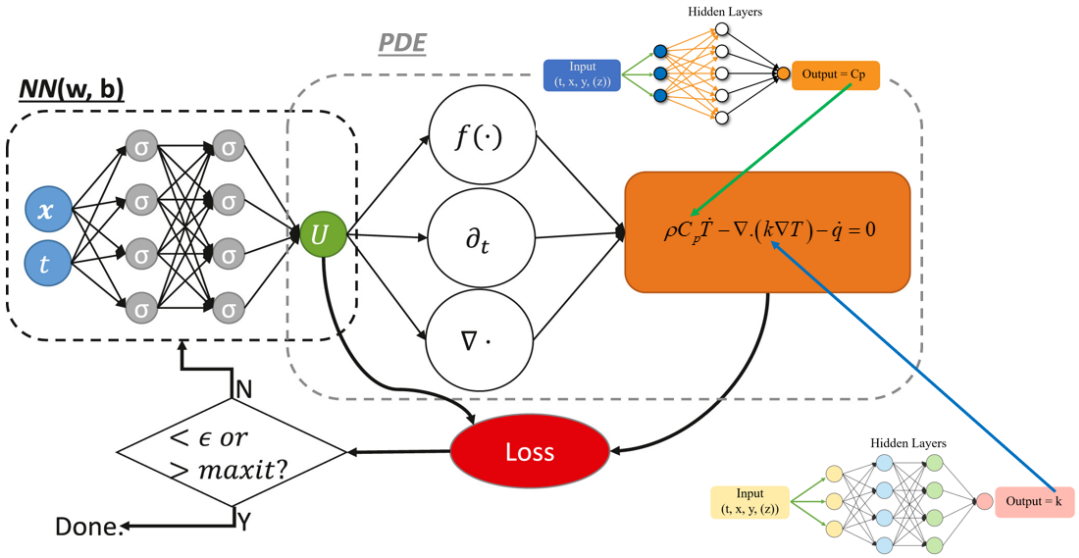

Figure 2.22: Using two different Neural Networks for conductivity and heat capacity

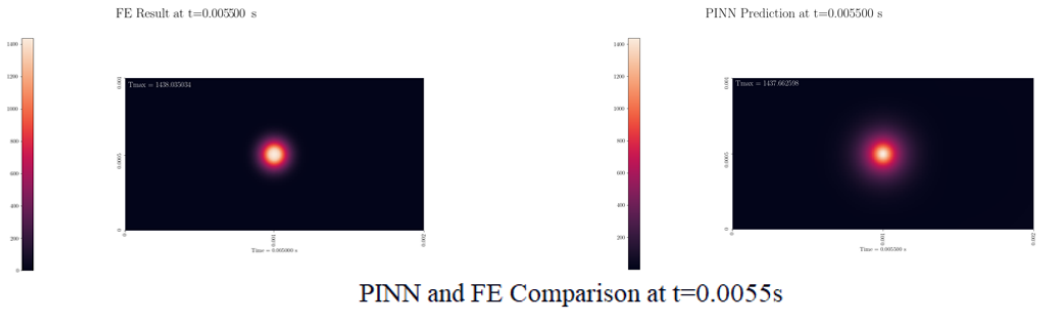

Figure 2.23: FE and PINN comparison of the first step of the Transfer Learning approach at 5.5 milliseconds

Static Normal Distribution on the y-axis and Shifted Normal Distribution on the z-axis, in order to make sure that most of the collocations points are allocated at the point where the PDE loss is evaluated, or in other words where the laser is acting on the material. He has shown that this collocation point distribution strategy has brought down the error; however, with this approach, there have been still some inconsistencies at the maximum temperature prediction where the laser is acting. Then additional points directly in the geometric center of the laser, so-called central bias points, have been added in order to make sure that the laser has been sampled at the highest gradient points all over the time domain. He has shown that this approach brought down the error dramatically [Mueller et al., 2021]. This sampling strategy has been summarized in Figure 2.35

In the context of this thesis, similar to the 2D case that has been shown in Algorithm 1, in the 3D case, the random points have been generated in spherical coordinates, and then converted into Cartesian coordinates. Here, an equation has been derived in order to sample more points at the laser location, where the laser is acting on the material, and to follow the movement

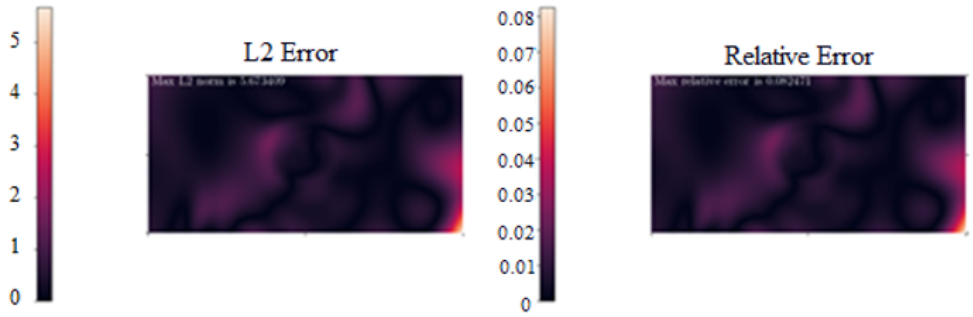

Figure 2.24: L2 and Relative Error comparison of the first step of the Transfer Learning approach at 5.5 milliseconds

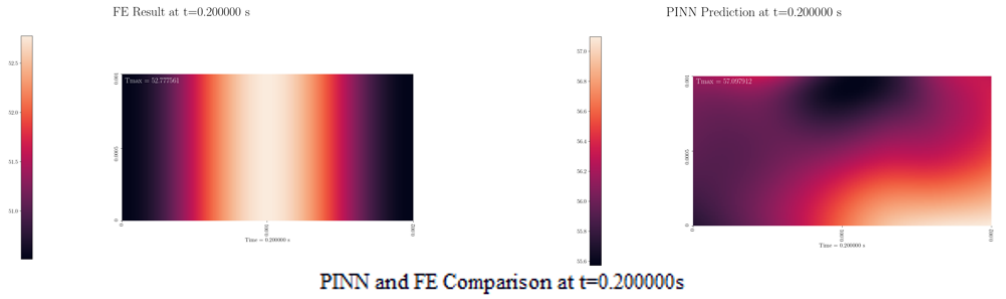

Figure 2.25: FE and PINN comparison of the first step of the Transfer Learning approach at the end of the time step

of the laser consistently. The algorithm for the 3D case is shown in 2 and the animations of these collocation points can be found at the following links `https://drive.google.com/file/d/1hBIq8Nc2nCh4tGaeOmO5WZJSbst9Ky4k/view?usp=sharing` and `https://drive.google.com/file/d/1rqORzYzQueaRD32gwa1c2g3VjQJeGymn/view?usp=sharing`.

At the following links `https://drive.google.com/file/d/1SLquPQTpSOQts9Dj_jKkYw4Z11E8dS88/view?usp=sharing` and `https://drive.google.com/file/d/18SNuStaWHfqMRxWZu15P7OuLgigRFHX3/view?usp=sharing`, one can find the animations of PINN and FE predictions for the 3D Case with temperature-dependent properties with a moving heat source (for the laser speed 1000 $\frac{mm}{s}$). Here it can be concluded that PINN has accurately predicted the maximum temperature value that has been observed in the FE result.

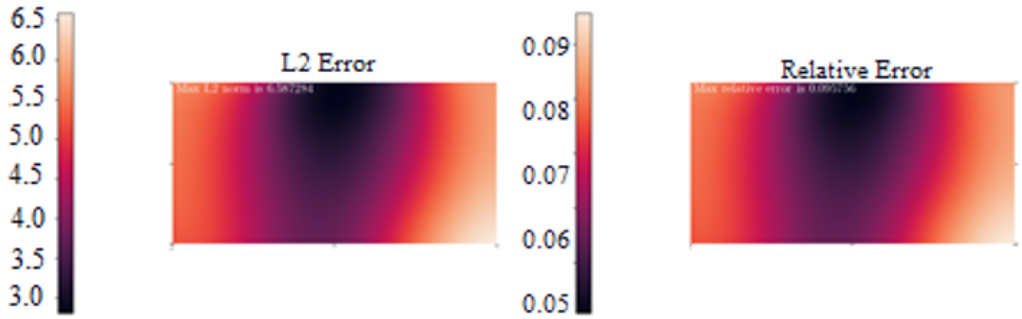

Figure 2.26: L2 and Relative Error comparison of the first step of the Transfer Learning approach at the end of the time step

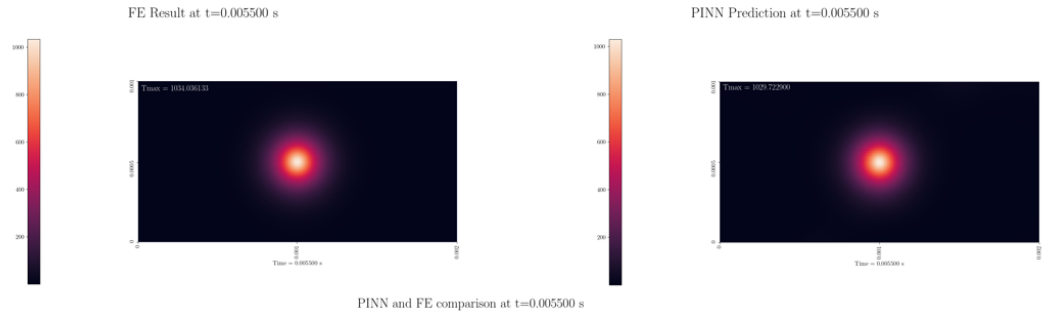

Figure 2.27: FE and PINN comparison of the second step of the Transfer Learning approach at 5.5 milliseconds

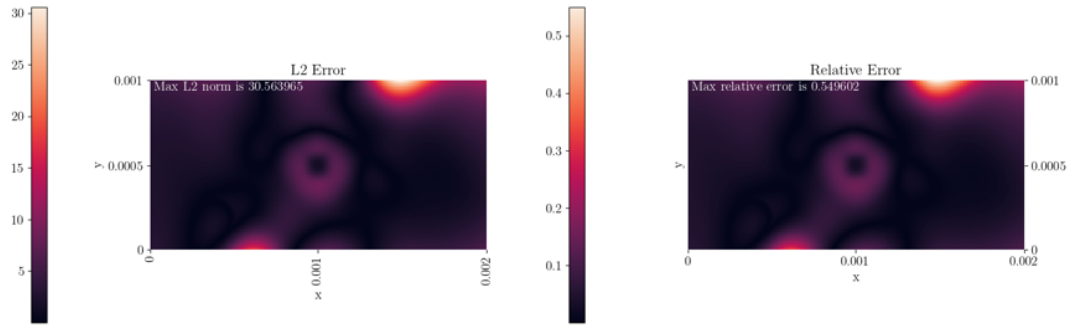

Figure 2.28: L2 and Relative Error comparison of the second step of the Transfer Learning approach at 5.5 milliseconds

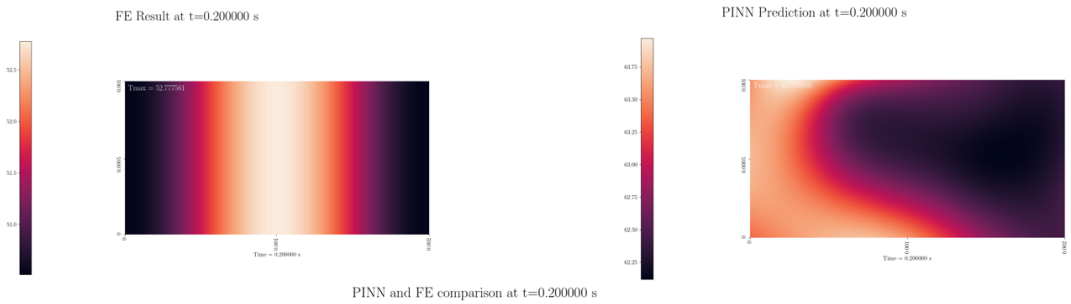

Figure 2.29: FE and PINN comparison of the second step of the Transfer Learning approach at the end of the time step

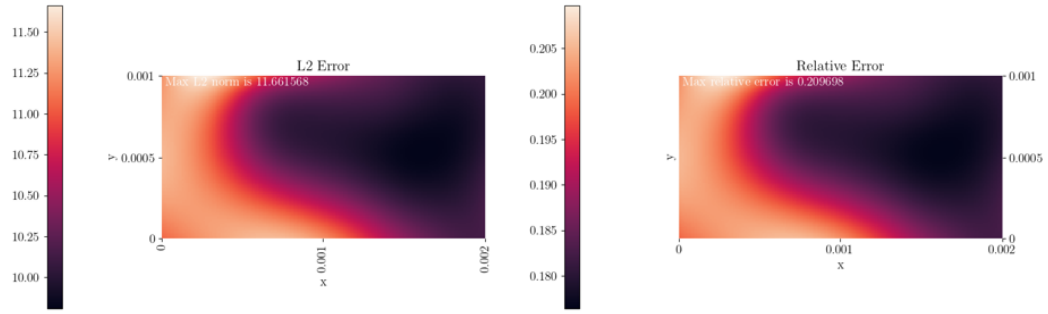

Figure 2.30: L2 and Relative Error comparison of the second step of the Transfer Learning approach at the end of the time step

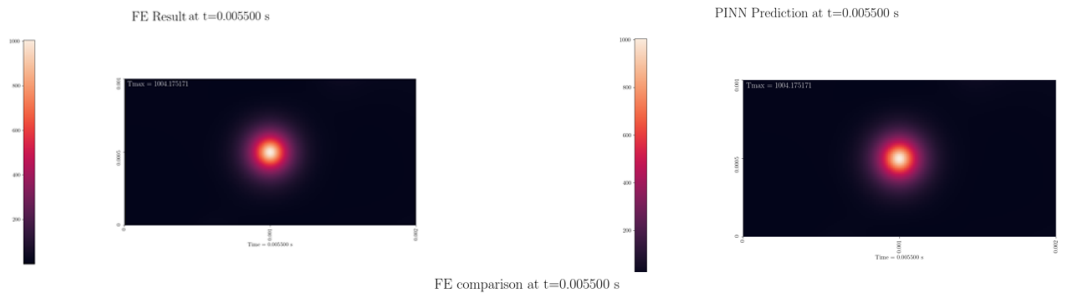

Figure 2.31: FE and PINN comparison of the third step of the Transfer Learning approach at 5.5 milliseconds

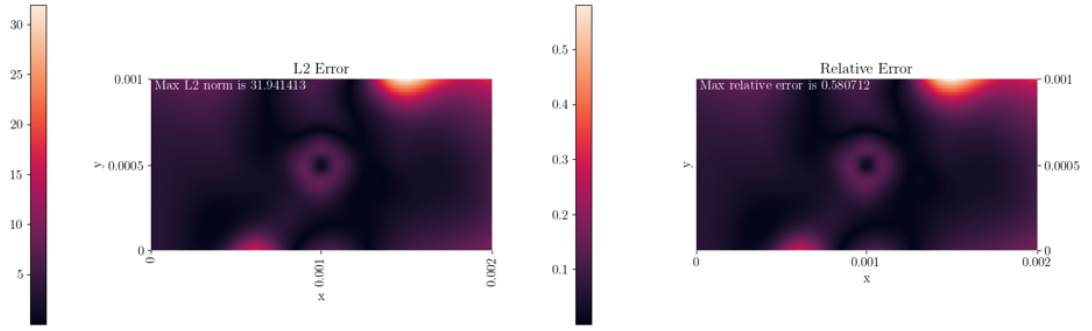

Figure 2.32: L2 and Relative Error comparison of the third step of the Transfer Learning approach at 5.5 milliseconds

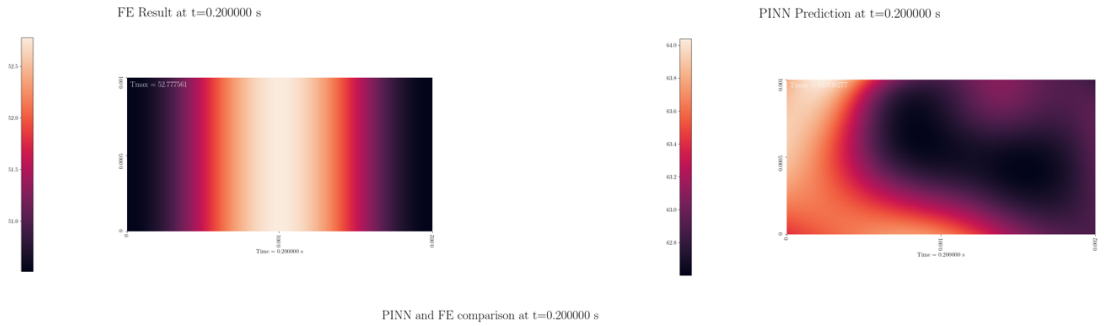

Figure 2.33: FE and PINN comparison of the third step of the Transfer Learning approach at the end of the time step

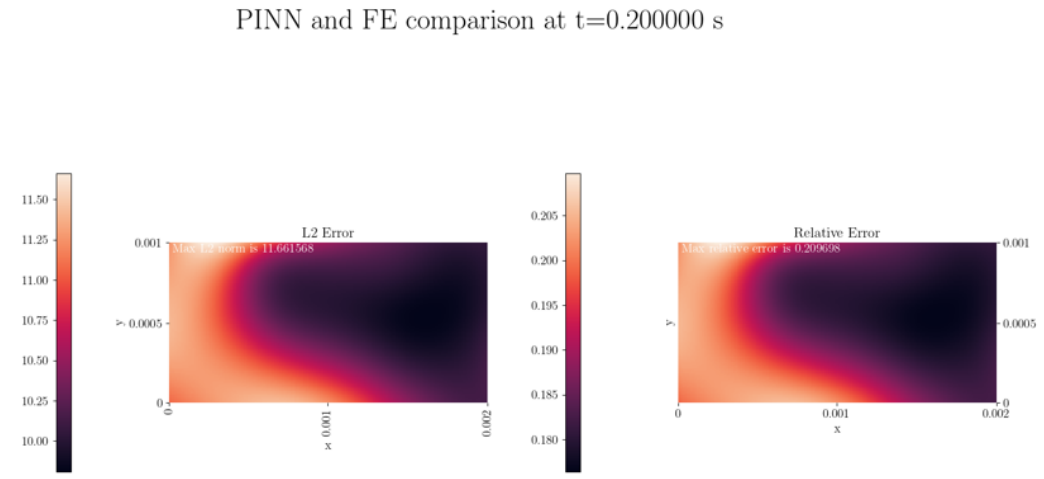

Figure 2.34: L2 and Relative Error comparison of the third step of the Transfer Learning approach at the end of the time step

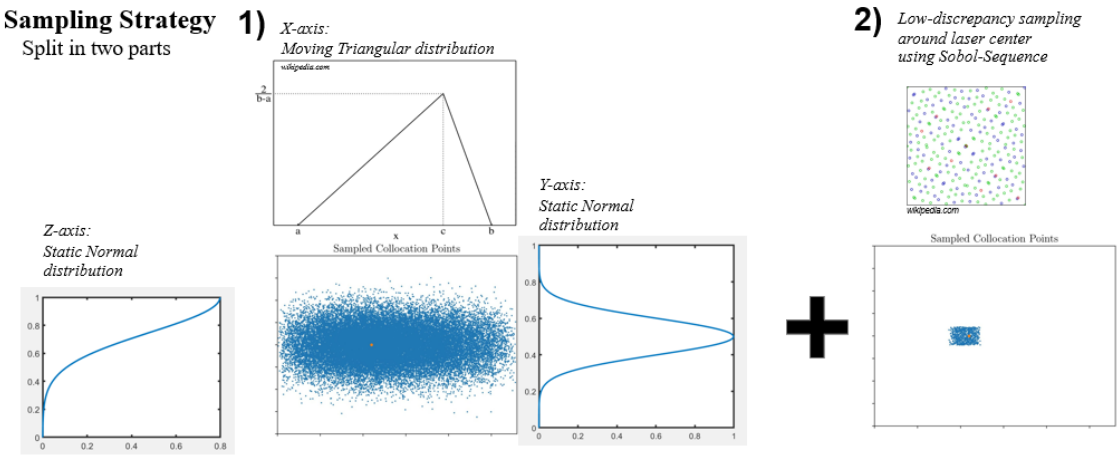

Figure 2.35: Sampling strategy developed by Ole Müller

Algorithm 2 Algorithm for Collocation Points for 3D Moving Heat Source Case

$N_{samples} = 10N_{samples}$
for $k = 1 \rightarrow N_{samples}$ **do**
    $a = 1$
    $N_{random} \rightarrow$ sampled from Sobol sequence
    $radius[k,:] \rightarrow N_{random}[0]$
    $theta[k,:] \rightarrow 2\pi N_{random}[1]$
    $time[k,:] \rightarrow N_{random}[2]$
    $phi[k,:] \rightarrow N_{random}[3]$
    $radius[k,:] = radius[k,:]^{a(1-time[k,:])+0.5}$
    $x[k,:] = \sqrt{2}radius[k,:]cos(phi[k,:])cos(theta[k,:])$
    $x[k,:] = x[k,:] + x_{locationofthelaser}$
    $y[k,:] = \sqrt{2}radius[k,:]cos(phi[k,:])sin(theta[k,:])$
    $y[k,:] = y[k,:] + y_{locationofthelaser}$
    $z[k,:] = \sqrt{2}radius[k,:]sin(phi[k,:])$
    $z[k,:] = z[k,:] + z_{locationofthelaser}$
    Concatenate time, x, y and z
    Delete coordinates <0 and >1

Chapter 3

Conclusion and Outlook

In this thesis, the potential of the Physics Informed Neural Networks (PINNs) has been investigated. This new algorithm uses the use auto-differentiation feature of the backpropagation stage of Artificial Neural Networks (ANN) to solve the Partial Differential Equation (PDE) that is also evaluated by FEM-solver. In the context of this thesis FE simulation results have been considered as the ground truth, and results obtained by PINN have always been compared with the FE simulation results in terms of Relative Error. The main focus during this study has been on the investigation of the rectangular plate with a moving heat source with a single track and a layer that includes temperature-dependent material properties (conductivity and heat capacity). It has been shown that the PINN algorithm has given satisfactory predictions for simple cases with a Relative Error below 1% without using any training data. However, when the temperature-dependent material properties have been implemented, some problems have started to arise. Since material properties depend on the output of the PINN, implementing temperature-dependent material properties has been a challenging task. In other words, any wrong evaluations of temperature output during the training process of PINN could iteratively affect the calculation of temperature-dependent properties. Several strategies have been tested in order to find a good way to incorporate temperature-dependent properties into the algorithm. In the first strategy, two different Artificial Neural Networks, one for conductivity, and one for heat capacity have been used in order to calculate the functions of these properties, and the output of these two networks has been supplied into the original PINN algorithm. However, this approach did not yield any meaningful results. The second strategy, which also constitutes the main approach that has been used during the course of this thesis, was using Transfer Learning Approach to transfer the weights and biases from a simple model with no temperature-dependent material properties to a more complicated model with temperature dependent conductivity and heat capacity. This approach has helped PINN to produce results

with a Relative Error $\lessapprox 20\%$ in comparison to FE simulation. The main disadvantage of this approach is that since the network architecture has been transferred from a simpler model to a more complicated model, Transfer Learning does not allow the utilization of Ensemble Training in order to change the Neural Network architecture, and therefore improving the accuracy of the results is a challenging task. Multiple strategies have been tested in order to improve the accuracy of the results. This includes implementing new loss functions into the PINN algorithm. However, out of all these strategies, changing the sampling method had the most impact. Simply allocating more points where the laser center is in order to calculate the high gradient points more precisely has brought down the error. For future work, different sampling strategies should be examined in order to find the best method to sample the high gradient points. Moreover, the influence of appropriate approximate functions for temperature-dependent material properties on the training should be investigated. As mentioned before, what is special about the PINN is that one can also include support data from experiments or FEM simulations that are not prebound by any predetermined mesh. In this thesis, the effect of FE support points has not been investigated, and therefore, it could be important for future studies to evaluate the impact of FE support points to understand the effect of support points in terms of accuracy. In this thesis, the powder pockets have been neglected in order to simplify the problem statement. Therefore, the effect of powder pockets (spatially changing material properties) should be studied in the future. Finally, the influence of the number of collocation, initial, and boundary points on the training should be evaluated.

Bibliography

[Al Rashid et al., 2020] Al Rashid, A., Khan, S. A., G. Al-Ghamdi, S., and Koç, M. (2020). Additive manufacturing: Technology, applications, markets, and opportunities for the built environment. *Automation in Construction*, 118(May):103268.

[Alber et al., 2019] Alber, M., Buganza Tepole, A., Cannon, W., De, S., Dura-Bernal, S., Garikipati, K., Karniadakis, G., Lytton, W., Perdikaris, P., Petzold, L., and Kuhl, E. (2019). Multiscale modeling meets machine learning: What can we learn?

[Baumann et al., 2018] Baumann, F. W., Sekulla, A., Hassler, M., Himpel, B., and Pfeil, M. (2018). Trends of machine learning in additive manufacturing. *International Journal of Rapid Manufacturing*, 7(4):310.

[Davis et al., 2017] Davis, S. E., Cremaschi, S., and Eden, M. R. (2017). *Efficient Surrogate Model Development: Optimum Model Form Based on Input Function Characteristics*, volume 40. Elsevier Masson SAS.

[Duchi et al., 2012] Duchi, J. C., Bartlett, P. L., and Wainwright, M. J. (2012). Randomized smoothing for (parallel) stochastic optimization. *Proceedings of the IEEE Conference on Decision and Control*, 12:5442–5444.

[Frazier, 2014] Frazier, W. E. (2014). Metal additive manufacturing: A review. *Journal of Materials Engineering and Performance*, 23(6):1917–1928.

[Gh Ghanbari et al., 2020] Gh Ghanbari, P., Mazza, E., and Hosseini, E. (2020). Adaptive local-global multiscale approach for thermal simulation of the selective laser melting process. *Additive Manufacturing*, 36(July):101518.

[Goldak et al., 1984] Goldak, J., Chakravarti, A., and Bibby, M. (1984). A New Finite Element Model for Welding Heat Sources. *Metallurgical Transactions B*, 15(June 1984):299–305.

[Goodfellow et al., 2016] Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep Learning*. MIT Press. `http://www.deeplearningbook.org`.

[Haykin, 1999] Haykin, S. (1999). *Neural Networks: A Comprehensive Foundation*. International edition. Prentice Hall.

[ISO, 2016] ISO (2016). Additive manufacturing - General principles - Part 2: Overview of process categories and feedstock.

[ISO/ASTM, 2013] ISO/ASTM (2013). Additive Manufacturing - General Principles Terminology (ASTM52900). *Rapid Manufacturing Association*, pages 10–12.

[Jagtap and Karniadakis, 2020] Jagtap, A. D. and Karniadakis, G. E. (2020). Extended physics-informed neural networks (xpinns): A generalized space-time domain decomposition based deep learning framework for nonlinear partial differential equations. *Communications in Computational Physics*, 28(5):2002–2041.

[Keller et al., 2020] Keller, F., Mazza, E., Hosseini, E., and Marelli, S. (2020). Surrogate modelling for multiscale thermal simulation of powder-bed additive manufacturing.

[Kim et al., 1975] Kim, S. D., Baker, C. G., and Bergougnou, M. A. (1975). Phase holdup characteristics of three phase fluidized beds. *The Canadian Journal of Chemical Engineering*, 53(1):134–139.

[Kingma and Ba, 2015] Kingma, D. P. and Ba, J. L. (2015). Adam: A method for stochastic optimization. *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*, pages 1–15.

[Konakli and Sudret, 2016] Konakli, K. and Sudret, B. (2016). Global sensitivity analysis using low-rank tensor approximations. *Reliability Engineering and System Safety*, 156:64–83.

[Kumar, 2019] Kumar, V. (2019). What is the effect of temperature on thermal conductivity of solids, liquids and gases?

[Lecun et al., 2015] Lecun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. *Nature*, 521(7553):436–444.

[Li et al., 2015] Li, C., Fu, C. H., Guo, Y. B., and Fang, F. Z. (2015). A multiscale modeling approach for fast prediction of part distortion in selective laser melting. *Journal of Materials Processing Technology*, 229:703–712.

[Markl and Körner, 2016] Markl, M. and Körner, C. (2016). Multiscale Modeling of Powder Bed-Based Additive Manufacturing. *Annual Review of Materials Research*, 46(April):93–123.

[Mishra and Molinaro, 2021] Mishra, S. and Molinaro, R. (2021). Physics informed neural networks for simulating radiative transfer. *Journal of Quantitative Spectroscopy and Radiative Transfer*, 270:1–25.

[Mueller et al., 2021] Mueller, O., Mazza, E., and Hosseini, E. (2021). Machine learning for multiscale thermal simulation of powder-bed additive manufacturing.

[Pan and Dias, 2017] Pan, Q. and Dias, D. (2017). An efficient reliability method combining adaptive Support Vector Machine and Monte Carlo Simulation. *Structural Safety*, 67:85–95.

[Panwisawas et al., 2020] Panwisawas, C., Tang, Y. T., and Reed, R. C. (2020). Metal 3D printing as a disruptive technology for superalloys. *Nature Communications*, 11(1):1–4.

[Piscopo et al., 2019] Piscopo, G., Atzeni, E., and Salmi, A. (2019). A hybrid modeling of the physics-driven evolution of material addition and track generation in laser powder directed energy deposition. *Materials*, 12(7).

[Qian, 1999] Qian, N. (1999). On the Momentum Term in Gradient Descent Learning Algorithms Acknowledgments. *Learning*, 5213.

[Raissi et al., 2019] Raissi, M., Perdikaris, P., and Karniadakis, G. (2019). Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686–707.

[Raissi et al., 2017] Raissi, M., Perdikaris, P., and Karniadakis, G. E. (2017). Physics Informed Deep Learning (Part II): Data-driven Discovery of Nonlinear Partial Differential Equations. (Part I):1–22.

[Ranjan et al., 2020] Ranjan, R., Ayas, C., Langelaar, M., and van Keulen, F. (2020). Fast detection of heat accumulation in powder bed fusion using computationally efficient thermal models. *Materials*, 13(20):1–25.

[Ruder, 2016] Ruder, S. (2016). An overview of gradient descent optimization algorithms. pages 1–14.

[Smatsi et al., 2020] Smatsi, N., Mazza, E., Hosseini, E., and Gh Ghanbari, P. (2020). Surrogate modelling for 3d multiscale thermal simulations of powder-bed additive manufacturing.

[Srivastava et al., 2014] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958.

[Yuan and Gu, 2015] Yuan, P. and Gu, D. (2015). Molten pool behaviour and its physical mechanism during selective laser melting of TiC/AlSi10Mg nanocomposites: Simulation and experiments. *Journal of Physics D: Applied Physics*, 48(3):35303.

[Zeiler, 2012] Zeiler, M. D. (2012). ADADELTA: An Adaptive Learning Rate Method.

[Zhang et al., 2019] Zhang, Z., Huang, Y., Rani Kasinathan, A., Imani Shahabad, S., Ali, U., Mahmoodkhani, Y., and Toyserkani, E. (2019). 3-Dimensional heat transfer modeling for laser powder-bed fusion additive manufacturing with volumetric heat sources based on varied thermal conductivity and absorptivity. *Optics and Laser Technology*, 109(May 2018):297–312.

[Zhou et al., 2019] Zhou, T., Peng, Y., and Li, J. (2019). An efficient reliability method combining adaptive global metamodel and probability density evolution method. *Mechanical Systems and Signal Processing*, 131:592–616.

[Zhou and Lu, 2020] Zhou, Y. and Lu, Z. (2020). An enhanced Kriging surrogate modeling technique for high-dimensional problems. *Mechanical Systems and Signal Processing*, 140:106687.

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Declaration of originality

The signed declaration of originality is a component of every semester paper, Bachelor's thesis, Master's thesis and any other degree paper undertaken during the course of studies, including the respective electronic versions.

Lecturers may also require a declaration of originality for other written papers compiled for their courses.

I hereby confirm that I am the sole author of the written work here enclosed and that I have compiled it in my own words. Parts excepted are corrections of form and content by the supervisor.

Title of work (in block letters):

PHYSICS INFORMED NEURAL NETWORK FOR THERMAL SIMULATION OF LASER POWDER-BED FUSION PROCESS

Authored by (in block letters):

For papers written by groups the names of all authors are required.

Name(s):	**First name(s):**
YARDIM	SARVEN

With my signature I confirm that

- I have committed none of the forms of plagiarism described in the 'Citation etiquette' information sheet.
- I have documented all methods, data and processes truthfully.
- I have not manipulated any data.
- I have mentioned all persons who were significant facilitators of the work.

I am aware that the work may be screened electronically for plagiarism.

Place, date	**Signature(s)**
ZURICH, 14/01/2022	

For papers written by groups the names of all authors are required. Their signatures collectively guarantee the entire content of the written paper.