

EIDGENÖSSISCHE TECHNISCHE HOCHSCHULE ZÜRICH ETHZ

Machine learning for multiscale thermal simulation of powder-bed additive manufacturing

by

Ole Müller

A semester thesis submitted in partial fulfillment for the
degree of Master of Science in Mechanical Engineering

Supervised by Prof. Dr. Eduardo Mazza and Dr. Ehsan Hosseini

in the

Experimental Continuum Mechanics Lab

Department of Mechanical and Process Engineering

in cooperation with

EMPA

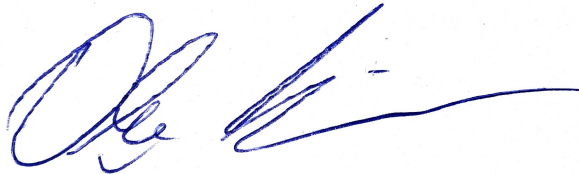
High Temperature Integrity Group (HTIG)

October 2021

Declaration of Authorship

I, OLE MÜLLER, declare that this thesis titled, ‘Machine learning for multiscale thermal simulation of powder-bed additive manufacturing’ and the work presented in it are my own. I confirm that:

- This work was done wholly while in candidature for a masters degree at ETH Zürich.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given.



Signed: Ole Müller

Place, Date: Kloten, 28.09.2021

What is the largest (rational) number n such that there are positive integers p, q, r such that

$$1 - \frac{1}{p} - \frac{1}{q} - \frac{1}{r} = \frac{1}{n}$$

- The Answer to the ultimate Question of Life, the Universe, and Everything!

EIDGENÖSSISCHE TECHNISCHE HOCHSCHULE ZÜRICH ETHZ

Abstract

High Temperature Integrity Group (HTIG)
Experimental Continuum Mechanics Lab

MSc in Mechanical Engineering

by Ole Müller

Fabrication of high quality parts through Laser Powder Bed Fusion (LPBF) requires accurate optimization of the process conditions. A viable approach for finding these is simulation, which additionally provides a detailed understanding. However, the computational cost is too high and poses a barrier for this approach. A multiscale thermal simulation strategy is recently proposed by Ghanbari et al. which merges the outcomes at small-millimeter-scale computationally cheap Finite-Element (FE) simulations for analysis at relatively large simulation domain for the LPBF process at a reduced cost. This thesis evaluated the effectiveness of data driven methods to act as alternative to FE simulations at the small-scale and further reduce the computational cost of thermal analysis of large LPBF parts. Two approaches are evaluated. Firstly a coupled principal component analysis (PCA) dimensionality reduction with a polynomial chaos expansion (PCE) model trained on a 60 simulation dataset. As well as a unsupervised physics informed neural net (PINN), which solves a simplified problem without the use of said dataset. Additionally the capability of PINNs to create general models with variable material parameters is evaluated. All evaluated surrogates produce steady state predictions with relative errors below 5%.

Acknowledgements

Firstly, I would like to thank Prof. Edoardo Mazza for giving me the opportunity to write this thesis at the Experimental Continuum Mechanics Lab and allowing me to conclude my Master at ETH Zurich with this work. I am very grateful to Dr. Ehsan Hosseini and Pooriya Ghanbari for the weekly meetings, their supervision and help through these months. Also i want to thank Prof. Siddharta Mishra and Roberto Molinaro, who agreed to be part of the effort and were always available to answer questions about the code base or the theory of physics informed neural networks. I want to express my gratitude to my fiance who was patient with me working late into the night, speaking code for dinner, as well as my family who was supportive through the ups and downs of the thesis.

Contents

Declaration of Authorship	i
Abstract	iii
Acknowledgements	iv
List of Figures	vii
1 Introduction	1
1.1 Metal Additive Manufacturing	1
1.1.1 Challenges in Simulating Additive Manufacturing	3
1.2 Adaptive Meshing	4
1.3 Multi-scale modeling	5
1.4 Surrogate Modeling	6
1.4.1 Dimensionality Reduction	7
1.4.1.1 PCA	7
1.4.1.2 Kernel PCA (KPCA)	8
1.4.2 Polynomial Chaos Expansion (PCE)	10
1.4.3 Neural Approach	11
1.4.3.1 Perceptron	11
1.4.3.2 Artificial Neural Networks (ANN)	12
1.4.3.3 Autoencoders	14
1.4.3.4 Physics Informed Neural Networks (PINN)	14
2 Experimental setup	17
2.1 The Problem Statement	17
2.2 The Dataset	19
2.3 PCA+PCE setup	19
2.4 PINN	20
2.4.1 Benchmark Testing	22
2.4.1.1 1D Heat equation with sinusoidal initial condition	22
2.4.1.2 2D Heat equation with Dirichlet and von-Neumann boundary conditions	22
2.4.2 2D Heat equation with Static Gaussian heat source	24
2.4.3 Implementation of the problem statement	24
2.4.4 The simplified Problem statement	26
2.4.5 Solution Approaches	28
2.4.5.1 Scaling of PDE components	28

2.4.5.2	Ensemble training	28
2.4.5.3	Adding support points	29
2.4.5.4	Adaptive Sampling Strategy	29
2.4.6	Ensuring correct initial conditions	32
3	Results and Discussion	33
3.1	PCA + PCE	33
3.2	PINN - Physics Informed Neural Networks	34
3.2.1	Parametric Model	37
4	Conclusion and Outlook	41
5	Appendix: Code Base	44
	Bibliography	44

List of Figures

1.1	Working Principle of LPBF process	2
1.2	Modeling ranges of the LPBF process	3
1.3	Multi-Physics in LPBF [Wikipedia]	3
1.4	2D Simulation Mesh analysis	5
1.5	Optimized Mesh	5
1.6	Typical structure of surrogate modeling	6
1.7	The Taxonomy of Dimensionality Reduction Techniques	7
1.8	Intuitive representation of PCA	8
1.9	Pitfalls of plain PCA vs. applying a Kernel	9
1.10	Typical structure of a perceptron	11
1.11	Basic Structure of ANN	12
1.12	Basic Structure of an autoencoder	14
1.13	Basic Structure of PINN	15
2.1	Modelled conductivity of HastelloyX	17
2.2	Goldak Heat Source Representation (elliptic)	18
2.3	1D Prediction vs Exact Temperature	22
2.4	Predicted and FEM Solution to the Boundary Problem	23
2.5	Prediction Error vs FEM	23
2.6	Predicted and FEM Solution to the static source Problem	24
2.7	Prediction Error vs FEM	24
2.8	Smoothed conductivity of HastelloyX	25
2.9	Smoothed phase transition	26
2.10	Failed PINN prediction of the complete problem	26
2.11	Failed PINN prediction of the simplified problem	28
2.12	Probability Density of a Triangular distribution	30
2.13	The adaptive sampling for capturing the Source	31
2.14	Comparison of initial predictions	32
3.1	Limited Error of the 6 variable vs. full field approach	33
3.2	Maximum Error of the 6 variable vs. full field approach	34
3.3	Relative difference of the prediction vs. FEM	35
3.4	Maximum temperature difference PINN vs. FEM	36
3.5	Maximum predicted temperature PINN vs. FEM	36
3.6	Temperature Evolution Analysis of different Conductivities	38
3.7	Parametrized model performance vs. selected FEM results	39

Chapter 1

Introduction

Additive manufacturing (AM) technologies allow the creation of virtually any geometry from metal, polymers and other materials. AM has been rapidly advancing into various industrial sectors, including aerospace, automotive, medical, architecture, arts and design, food, and construction [1]. However there are quite some challenges that prevail before AM is becoming common place in our daily world. This thesis will focus on the AM of metals. What is stopping the ubiquitous application of metal AM processes is the influence of many different factors in making a part robust, stable and durable. In order to understand these challenges, we first have to understand what entails the AM-process for metals.

1.1 Metal Additive Manufacturing

Metals have been commonly formed by reshaping or subtractive means. Due to poor mechanical properties after casting of metals, it is often just the first step of a longer manufacturing process. With the invention of AM this changed. This new process offers the potential to save between 30 and 50% of the cost of machining aerospace titanium structural components [2]. Albeit they do require a short heat treatment for the reduction of residual stresses due to thermal expansion and contraction during manufacturing, as well as a short post processing to remove support structures. In this thesis we will focus on Laser Powder Bed Fusion (LPBF). There are other ways to create full metal parts additively such as Binder Jetting, Direct Energy Deposition and Material Jetting, these will however not be part of this investigation. In the Figure 1.1 one can see the working principle of the LPBF-Process:

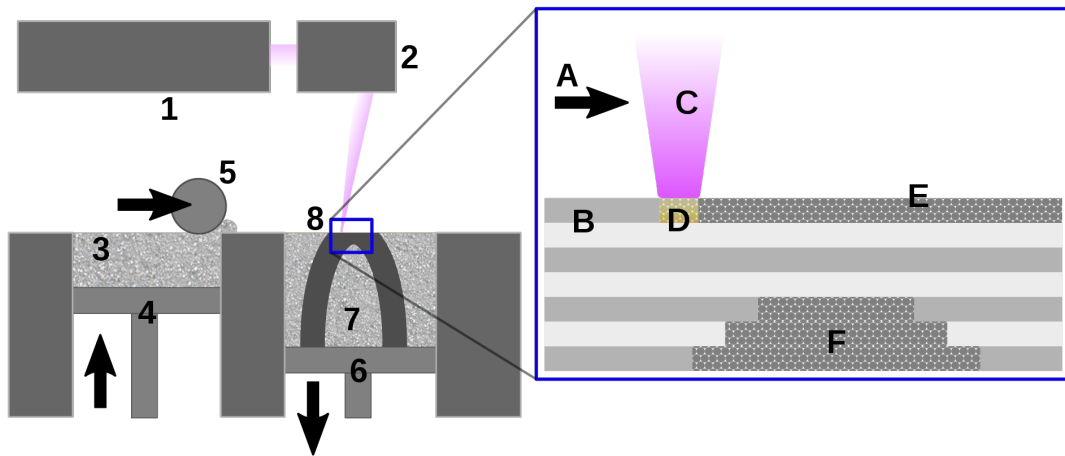


FIGURE 1.1:

Working Principle of LPBF process: [\[Wikipedia\]](#)

- 1) LASER is generated
 - 2) Mirrors redirect the energy
 - 3) Fresh powder reservoir
 - 4) A piston pushes the new powder up
 - 5) A spreader takes the powder and distributes it on the build platform
 - 6) A piston attached to the build platform moves downwards the same amount 4) moves down
 - 7) Unmolten powder under the part geometry
 - 8) Impact location of the Laser
-
- A) Scanning direction of the Laser
 - B) Molten and hardened metal
 - C) LASER-Beam
 - D) Melt-pool
 - E) Fresh powder layer
 - F) Unmolten powder from previous layers

The powder remaining at 7) can be recycled and reused for the manufacturing of other parts [3]. This way AM reduces waste and has a smaller material requirement than manufacturing the same part with subtractive means [4]. Currently, the best parameters for single tracks are often determined through laborious experimental study prior to conducting experiments to create high density parts. [5] However, this does not guarantee the optimal process parameters for all part geometries, and needs to be repeated for new materials. The obvious approach would be to simulate the manufacturing process using different parameters and derive the best settings for a high quality part. Nevertheless, figure 1.1 does not capture the multi-physics nature of the process. There are a multitude of very complex multi-physical phenomena accompanying the manufacturing process [6]. In order to receive continuously good quality parts, these need to be understood and controlled.

1.1.1 Challenges in Simulating Additive Manufacturing

The process of LPBF can be represented on three different levels of detail: Micro-, Meso- and Macro-scale as can be seen in figure 1.2 from [7]

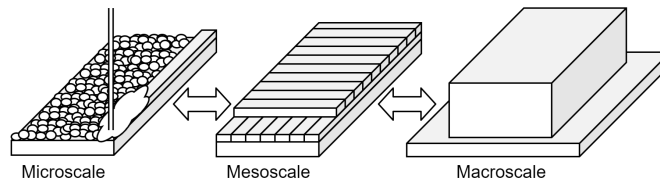


FIGURE 1.2: Modeling ranges of the LPBF process

The challenge with AM is that the relevant physical processes happen in the micro-scale in the range of $\text{nm}-\mu\text{m}$, while the desired part is of macro-scale ($\text{cm}-\text{m}$) proportions. In order to have consistently high quality parts, one has to understand the dynamics at the finer levels.

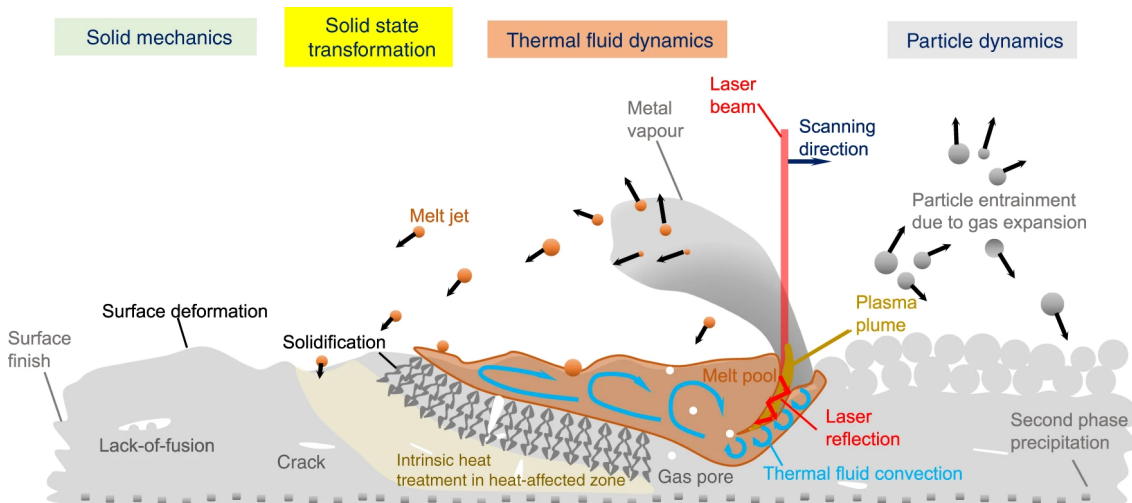


FIGURE 1.3: Multi-Physics in LPBF [Wikipedia]

As can be seen on figure 1.3 there are a multitude of different physics phenomena in the micro-scale that are affecting each other. Experiments and simulations aimed at understanding the micro-scale focus on resolving the interactions of individual powder particles in domains with sizes of several μm . The limited investigated space allows for the inclusion of many of the shown effects. This is too expensive to model in larger domains, but provides valuable insights for simplifying the model at the meso-scale. Various computational methods have been used to estimate the micro-scale behavior: cellular automata [8], phase-field modeling [9] and Monte Carlo simulations [10] to simulate the crystalline microstructure lattice Boltzmann method [11], fluid and particle dynamics (CFD)[12], [13] to model the multiphysics of melt pool dynamics in the micro-scale and thus help

modeling the process in the meso-scale. To include all of the micro-scale effects into one meso-scale simulation, would be beyond the scope of this master thesis and is in fact ongoing research [14][6][15]. Instead, we focus only on the thermal heat transfer within the solid and simplify the other aspects accordingly. The thermal distribution over time is the major influence on the material behavior. Because many material properties (such as density, surface tension, heat conductivity, heat capacity, and thermal diffusivity) are temperature dependent, temperature and its gradients are the most relevant quantities. The quality of the final component is determined by the thermodynamic, hydrodynamic, and mechanical impacts induced by these qualities. [7] They directly determine residual stresses within the part due to expansion and contraction [16], and influence crystal growth [17]. An additional challenge is the required temporal resolution of the process. When the laser reaches a given location, the temperature changes rapidly ($40\mu\text{s}$ depending on the scanning speeds) from ambient (25°C - 250°C) to peak temperature (3000°C). In order to resolve these steep temperature gradients, time steps need to be very small. For solving an FEM simulation with a $3\mu\text{m}$ uniform Cartesian mesh, required time steps are pushed to the low ns range [18]. Thus, simulating the meso-scale requires a fine resolution and small time steps for accurate results. Nonetheless, it cannot be used to simulate the manufacturing of a part geometry. The entire manufacturing process often takes several hours. Resolving an hour in the required time resolution would consume enormous computational resources. However, to perform a parameter study, this would need to be repeated for every set of parameters to be evaluated. Therefore, when performing a sensitivity analysis or uncertainty quantification of a set of parameters the computational cost of one evaluation has to be brought down.

1.2 Adaptive Meshing

As explained above, a very fine mesh resolution is required for correct results due to the high gradients. However, as shown by in Figure 1.4 from [19] much of this fine mesh is not required in the regions with low temperature gradients.

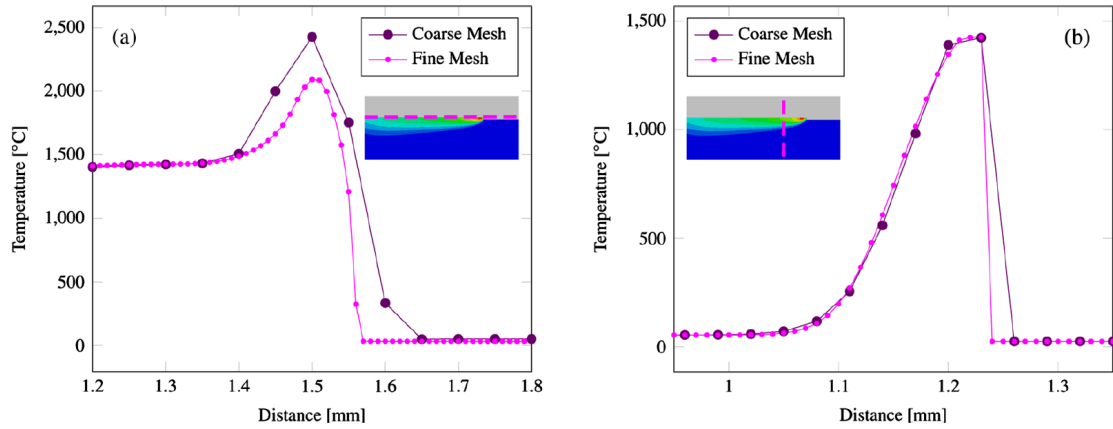


FIGURE 1.4: 2D Simulation Mesh analysis

With proper care one can formulate a mesh design that is taking into account the temperature gradients and is optimized for minimum element count. This has been done in previous work and the results can be seen in Figure 1.5 from [20]. Here the LASER moves along a very fine mesh (with $10\mu\text{m}$ resolution) for a few mm. All the while the low gradients are resolved on larger mesh elements.

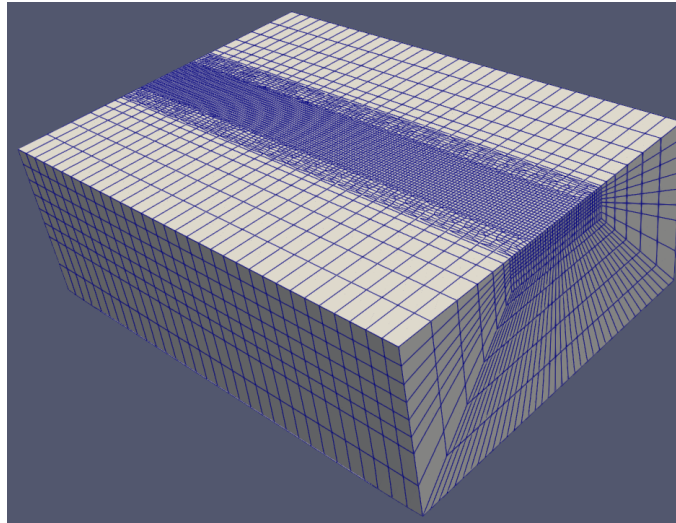


FIGURE 1.5: Optimized Mesh

1.3 Multi-scale modeling

In reality, the desired geometries are not only short straight lines. To simulate the manufacturing of any geometry, the same idea that was before applied to space, is now applied to time. In regions where a fine temporal resolution is required, i.e. the actual melt pool location, a fine "local" simulation is evaluated. The "global" simulation serves as a way

to estimate the initial temperature distribution of the local model after it shifted its position along the laser path. The challenge for the solution of the local model is posed by the changing boundary condition. It is not possible solve the local model only once and then apply the result to every desired location of the global model. The metal powder has a much lower heat conductivity than solid metal. Therefore, in regions close to an edge, wall or hole, the temperature distribution is changing quite dramatically. One local model can be evaluated on good computational hardware (Compute-Cluster EULER CPU XeonGold5118 with 14 threads) in around 13h. Thus, for an entire geometry it would still take an unfeasible amount of resources.

1.4 Surrogate Modeling

Surrogate models statistically tie input data to output data gathered by conducting a complex system simulation. These models are most commonly used when the link between input and output data is unclear, or when the relationship is exceedingly complex and a simpler relationship with reasonable accuracy is sought [21]. The computational cost during use of the surrogate model should be lower than the conventional one. Typical surrogate models include Kriging [22], Support Vector Machines (SVM) [23], Radial Basis Functions (RBF) [24], low-rank tensor approximations [25], Polynomial Chaos Expansion [26][20] and Neural Networks [27]. The usual procedure involves reducing the high dimensional input and output to a lower dimension using a dimensionality reduction. The problem is then solved in a lower dimensional space and then reconstructed to the desired output space, as can be seen in figure 1.6. This dramatically reduces computational cost from several hours of evaluation to a second or less, given enough training data. If the error after reconstruction is below the acceptance threshold, the surrogate model is a good choice for reducing the computational cost.

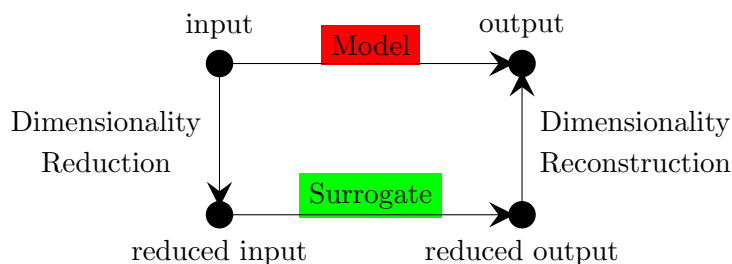


FIGURE 1.6: Typical structure of surrogate modeling

1.4.1 Dimensionality Reduction

In order to reduce the complexity of inputs it is important to understand the different dimensions of said input. In this report we looked at two convex fully spectral reductions: PCA and kernel PCA, as well as a nonconvex neural network method. The entire taxonomy of dimensionality reduction can be seen in figure 1.7 from [28].

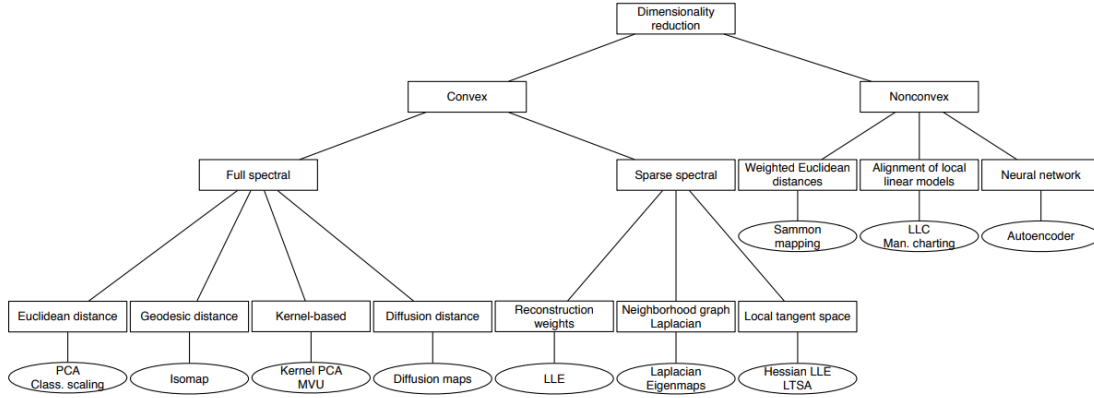


FIGURE 1.7: The Taxonomy of Dimensionality Reduction Techniques

1.4.1.1 PCA

To perform a principal component analysis (PCA) one begins by centering the input data around 0 and scaling it to unit variance.

$$\begin{aligned}
 \text{Original Data: } & \bar{\mathbf{x}}^n \in \mathbb{R}^D \\
 \text{mean:} & \mu_{\bar{\mathbf{x}}} \in \mathbb{R}^D \\
 \text{variance:} & \sigma_{\bar{\mathbf{x}}} \in \mathbb{R}^D \\
 \text{scaling:} & \mathbf{x}^n = (\bar{\mathbf{x}}^n - \mu_{\bar{\mathbf{x}}}) / \sigma_{\bar{\mathbf{x}}}
 \end{aligned}$$

$$\text{input } X = \{\mathbf{x}^0, \mathbf{x}^1, \dots, \mathbf{x}^N\}; \quad \mathbb{E}[\mathbf{x}^n] = 0; \quad \mathbf{x}^n \in \mathbb{R}^D \quad (1.1)$$

PCA is able to identify linear correlations in the data and implies statistical dependency.

$$\mathbb{E}[\mathbf{x}_i^n \mathbf{x}_j^n] \neq 0 \quad (1.2)$$

If there were no statistical dependency: $\mathbb{E}[\mathbf{x}_i^n \mathbf{x}_j^n] = \mathbb{E}[\mathbf{x}_i^n] \mathbb{E}[\mathbf{x}_j^n] = 0$. Commonly PCA is performed using the covariance method. For this the covariance matrix is calculated:

$$C = \frac{1}{N-1} X^T X; \quad C \in \mathbb{R}^{D \times D} \quad (1.3)$$

And then eigenvector composition is performed on the covariance matrix:

$$C = V\Lambda V^{-1}; \quad \Lambda = \text{diag}(\{\lambda_i | i \in \{1, \dots, D\}\}); \quad V^{-1} = V^T \in \mathbb{R}^{D \times D} \quad (1.4)$$

Then the resulting eigenvectors (columns of V) are sorted to decreasing order.

$$\mathbf{y}^n = V^T \mathbf{x}^n; \quad (W \triangleq V^T) \quad (1.5)$$

Principal components of X are the eigenvalues of the covariance matrix C . To reduce the dimensionality, or project to a lower dimensional space, one only keeps a given number $P < D$ of eigenvectors that correspond to the largest eigenvalues to create $\hat{V} \in \mathbb{R}^{D \times P}$. Using \hat{V} a new reduced dataset can be created:

$$z^n = \hat{V} x^n \quad (1.6)$$

In order to get an intuitive understanding of what is happening during the PCA, it is sensible to reduce the initial dimensions so something easy to understand. In figure 1.8, the initial input dimensions are in 2D being reduced to one principal component. (Animation only plays in Adobe Acrobat reader)

FIGURE 1.8: Intuitive representation of PCA

1.4.1.2 Kernel PCA (KPCA)

PCA performs a matrix decomposition into eigenvectors, this is a linear transformation. Any projection is always a linear combination of the original dimensions. There exist several datasets (e.g. figure 1.9) where PCA struggles to capture the correct behavior, as

they can only be captured by nonlinear combinations. The concept of KPCA is to apply the PCA to an altered covariance matrix. Instead of using the matrix defined in eq. (1.3) we apply a kernel. Many functions can be kernels [29], they need to be an inner product in a suitable space and they must be symmetric:

$$\forall x, x' \in X : k(x, x') = \phi(x)^T \phi(x') = \phi(x')^T \phi(x) = k(x', x) \quad (1.7)$$

Also their kernel (gram) matrix needs to be positive semidefinite:

$$K = \begin{bmatrix} k(x_1, x_1) & \dots & k(x_1, x_n) \\ \vdots & & \vdots \\ k(x_n, x_1) & \dots & k(x_n, x_n) \end{bmatrix} \quad (1.8)$$

This means

$$(i) \quad \forall x \in \mathbb{R}^n : x^T K x \leq 0 \quad (1.9)$$

$$(ii) \quad \forall \text{ Eigenvalues } \lambda(K) \leq 0 \quad (1.10)$$

The advantages are that many nonlinear features can be successfully reduced. However, the result cannot be interpreted intuitively as PCA can, as the transformations are nonlinear. As can be seen in figure 1.9 [©Sebastian Raschka]

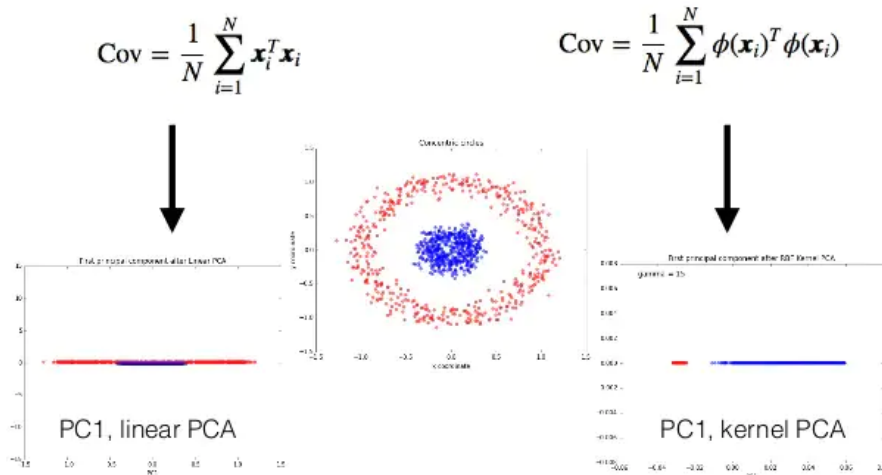


FIGURE 1.9: Pitfalls of plain PCA vs. applying a Kernel

Additionally, a reconstruction to normal space is not guaranteed. It requires hyperparameter tuning, selection of the adequate kernel and more computational complexity. This can only be achieved with hyperparameter tuning which requires an extensive amount of training data.

1.4.2 Polynomial Chaos Expansion (PCE)

PCE is the actual surrogate model. It takes the reduced input and is a way to model processes in terms of orthogonal polynomials. It can be thought of as PCA over an orthogonal vector space of polynomials rather than reals. It allows handling of any distribution that has a finite second moment, i.e. finite variance. The general form is:

$$Y = \sum_{i=0}^{\infty} y_i \Phi_i(X) \quad (1.11)$$

Where X is a random variable, and Φ is a polynomial with input variable ξ [30]. PCE can create a model that approximates the behavior of the given distribution as long as its second moment is finite and the polynomial Φ_i is orthogonal to the probability density function of $X_i \in \mathbb{R}$. If X_i comes from a Gaussian, then ϕ_i would have to be a Hermite polynomial fulfilling as per definition.

$$\langle H_i(X)H_j(X) \rangle = \int_{-\infty}^{\infty} H_i(X)H_j(X)e^{-\frac{X^2}{2}} dX \quad (1.12)$$

Then PCE can represent any finite variance random variable Y as a series of Hermite polynomials about a Gaussian random variable x

$$Y = \sum_{i=0}^{\infty} y_i H_i(X) \quad (1.13)$$

Every distribution of random variables has their orthogonal polynomials (i.e. uniform distribution is orthogonal to Legendre polynomials; exponential distribution is orthogonal to Laguerre polynomials). For real life application an infinite sum cannot be computed so a truncation scheme has to be applied so that

$$Y = \sum_{i \in S} y_i \Phi_i(X) \quad (1.14)$$

with S being the set of considered multi-indices in the truncated expansion:

$$S^p = \{i \in \mathbb{N} : |i| \leq p\} \quad (1.15)$$

and p the maximum degree of considered polynomials. In this work UQLab is used for any PCE related analysis. The UQLab library [31] which is implemented for MATLAB has different truncation schemes already implemented as well as the optimization required to find the best coefficients. PCE chooses the most likely combination of outputs and is able to give confidence intervals for every prediction. As it is a supervised machine learning technique, the predictive performance becomes better with increasing available data. This

means most weight is on feature engineering and parameter optimization to gain the best surrogate model for the available limited data set.

1.4.3 Neural Approach

”What fires together wires together” [32] is the basic logic of the perceptron network, a unit imitating a biological neuron. In 1958 it was originally devised for image recognition but due to poor performance it stagnated for many years. [33] With the improvement of multilayer perceptrons and the rise of GPU-hardware the algorithm had a dramatic performance improvement. Single layer perceptrons can only approximate linearly separable patterns. Adding more layers enables more complexity. Multilayer Perceptrons are usually referred to as Artificial Neural Networks (ANN).

1.4.3.1 Perceptron

The basic building block of an ANN is still the perceptron or slight variations thereof.

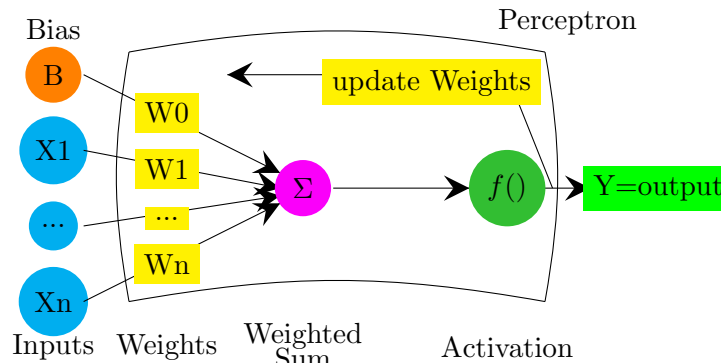


FIGURE 1.10: Typical structure of a perceptron

The perceptron sums up all the inputs and applies its activation function $\sigma()$ to the sum.

$$I(n) = \sum_{i=1}^k w_i(n)x_i(n) = \mathbf{x}^T(n)\mathbf{w}(n) \quad (1.16)$$

If $\sigma()$ is the sign function then the perceptron is a binary classifier, with T being the decision threshold

$$y = \sigma(I - T) = \begin{cases} +1 & \text{if } I \geq T \\ -1 & \text{if } I < T \end{cases} \quad (1.17)$$

$$y(n) = \mathbf{x}^T(n)\mathbf{w}(n) = \mathbf{w}^T(n)\mathbf{x}(n) \quad (1.18)$$

Then it updates the weights depending on the difference to the desired output, where β is the learning rate.

$$\begin{aligned}\mathbf{w}(n+1) &= \mathbf{w}(n) + \beta \mathbf{x}(n)^T \mathbf{w}(n) \mathbf{x}(n) \\ &= \mathbf{w}(n) + \beta \mathbf{x}(n)^T \mathbf{x}(n) \mathbf{w}(n) \\ &= \mathbf{w}(n) + \beta R(n) \mathbf{w}(n)\end{aligned}\tag{1.19}$$

$R(n)$ is here the covariance matrix. The above equation can be written in continuous form.

$$\frac{\mathbf{w}(n+1) - \mathbf{w}(n)}{\beta} \approx \frac{d\mathbf{w}}{dt} = R(n) \mathbf{w}(n)\tag{1.20}$$

A single linear perceptron tends to extract the principal component from a stationary input vector sequence, emulating PCA [34]. The original structure of perceptrons is usually used as a classifier, however combining different activation and loss functions enables also regression problems to be solved.

1.4.3.2 Artificial Neural Networks (ANN)

One can put several perceptrons or nodes in series and/or parallel to create a network, an artificial neural network. The output of nodes of one layer (parallel perceptrons) is fed as input into the next layer of nodes.

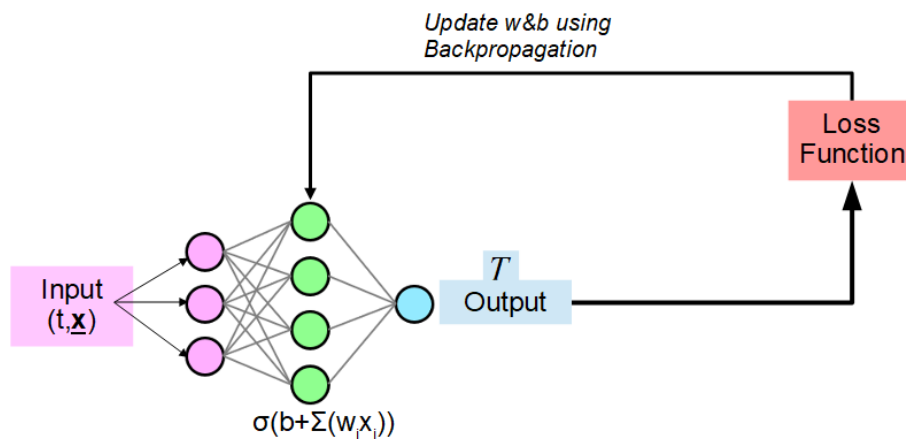


FIGURE 1.11: Basic Structure of ANN

While perceptrons have a preferred binary output as 1 or 0, the activation functions of ANN-nodes are chosen such, that the output is graded between 1 and 0 (tanh, ReLU, sin etc.). This allows for backpropagation of the weights using a given loss or error function as shown by Rumelhart [35]. In order to understand the best learning direction one needs to understand the gradient of the errors which are calculated using backpropagation. The

combined input x_j to the node j is a function of the outputs y_i of the previous layer, connected to j with the weights w_{ji} similar to (1.8):

$$x_j = \sum_i y_i w_{ji} \quad (1.21)$$

The output y_j is depending on the activation function $\sigma()$

$$y_j = \sigma(x_j) \quad (1.22)$$

Thus when a given solution d exists one can calculate the error E for a given number of data points c :

$$E = \frac{1}{2} \sum_c \sum_j (y_{j,c} - d_{j,c})^2 \quad (1.23)$$

This is also called the forward pass through the ANN. To find the gradients a backward pass is performed. One starts by computing the differentiation for a particular data point

$$\frac{\partial E}{\partial y_j} = y_j - d_j \quad (1.24)$$

then using the chain rule one can derive

$$\frac{\partial E}{\partial x_j} = \frac{\partial E}{\partial y_j} \frac{\partial y_j}{\partial x_j} \quad (1.25)$$

This means one knows what change is necessary in the input x_j in order to decrease the error. But the input itself compounded of outputs from previous nodes and of weights. In order to have the dependency of the error on a part of the ANN that is changeable, one needs the derivative with respect to w_{ji} , the weight of the connection from j to i :

$$\begin{aligned} \frac{\partial E}{\partial w_{ji}} &= \frac{\partial E}{\partial x_j} \frac{\partial x_j}{\partial w_{ji}} \\ &= \frac{\partial E}{\partial x_j} y_i \end{aligned} \quad (1.26)$$

Once the best direction to reduce the errors is known for every weight it is changed proportional to the accumulated gradient over all input output cases.

$$\Delta w = -\epsilon \frac{\partial E}{\partial w} \quad (1.27)$$

As Rumelhart stated, gradient descent is not as efficient as methods using second derivative (e.g. Newton Method and other using a Jacobian), but is much simpler and parallelizable. This allows for speedup when using Graphical (GPUs) or Tensor processing Units (TPUs). Gradient descent can however be significantly improved by employing an acceleration in the optimization space. Thus weights with a steep gradient will experience

bigger changes also in successive updates

$$\Delta w(t) = -\epsilon \frac{\partial E}{\partial w(t)} + \alpha \Delta w(t-1) \quad (1.28)$$

With t being an incrementing counter for every update, α and ϵ are tuning parameters.

1.4.3.3 Autoencoders

Autoencoders are a special type of neural network, that due to its architecture serves as encoder (dimensionality reduction) and decoder (reconstruction), as can be seen in figure 1.12 from [36]. To use an autoencoder the network is first trained on the data set and then split apart afterwards to encode and decode. There exists some difficulty of building an effective interface between PCE and autoencoder. The encoder would need to be trained in combination with the PCE inputs and the decoder with PCE outputs.

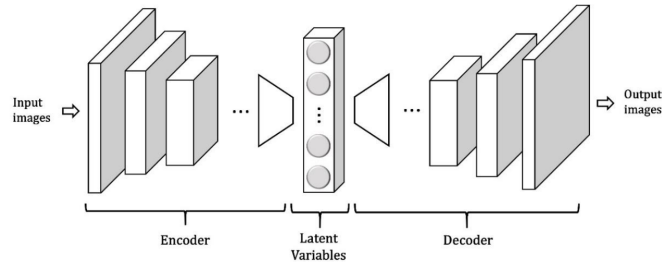


FIGURE 1.12: Basic Structure of an autoencoder

In order to train any ANN efficiently it needs a high amount of high quality data. If it is designed as a deep network (more than two node layers), the data does not need to be engineered, as it is able to do its own feature selection [37]. But it is difficult to generate enough data, if one data point requires more than 12h to generate using FEM simulation.

1.4.3.4 Physics Informed Neural Networks (PINN)

In order to reduce the need for training data, an alternative strategy 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 back-propagation. As seen in the equations above, for the correct update of ANN weights the derivative of the output with respect to the inputs is always computed. Thus 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. This is composed using the chain rule.

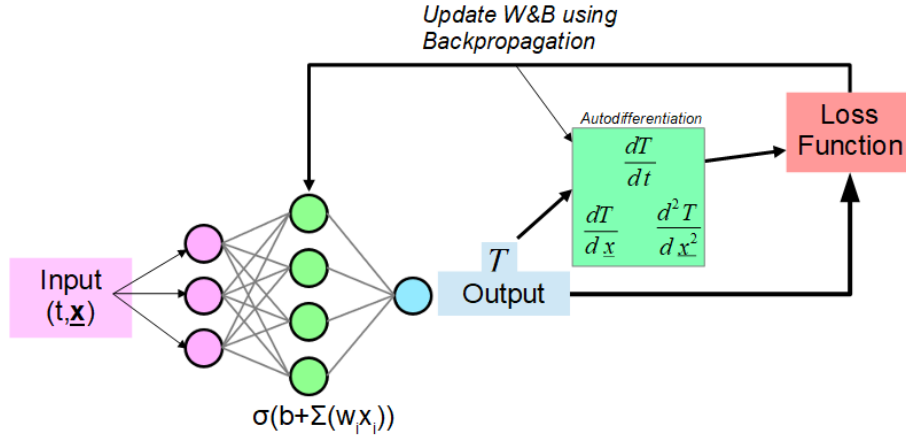


FIGURE 1.13: Basic Structure of PINN

In the case of this thesis one is interested in the solution of a special case of heat equation. This partial differential equation (PDE) can be written in the form

$$\rho c_p \frac{dT}{dt} = \frac{d}{dx} \left(k \frac{dT}{dx} \right) + q \quad (1.29)$$

As both x and t are inputs and u is the output of the network, equation (1.26) can be solved by the network. The network still needs to be trained to solve it correctly, and for this a adequate loss function has to be formed. As described above the loss or error function is important to find the gradient for updating the weights. One way to derive a loss function the correct solution of the PDE is by rewriting it as a residual:

$$E_{res} = \rho c_p \frac{dT}{dt} - \left(\frac{d}{dx} \left(k \frac{dT}{dx} \right) + q \right) \quad (1.30)$$

The loss also needs to be defined for the boundary and initial conditions.

$$E_{init} = T_{init} - T \quad (1.31)$$

and for a fixed temperature Dirichlet boundary

$$E_{boundary} = T_{boundary} - T \quad (1.32)$$

but also a gradient von-Neumann boundary condition can be implemented due to the auto-differentiation ability of the network

$$E_{boundary} = f_{boundary} - \frac{dT}{dx} \quad (1.33)$$

where f is representative for the given flux. A big advantage of this process is that the network generates its own training data according to the given loss functions. In the

case that some additional data is available, be it in the form of experiments or FEM-simulations, they can be fed in using another loss. This new loss is not dependent on any predetermined mesh as conventional methods are and can be evaluated at any given location.

$$E_{data} = T_{data} - T \quad (1.34)$$

During the training process the error will decrease and the PDE-residual gets minimized. It was shown that the generalization error decreases if the training error decreases [38]. So ensuring a low training error is important for ensuring a good prediction performance.

Chapter 2

Experimental setup

2.1 The Problem Statement

As touched upon in the introduction, the goal of this thesis is to find a surrogate model for the thermal simulation of moving LASER heat source for a domain in the millimeter range.

The thermal simulation considered in this thesis is using HastelloyX as a material. Its latent heat of 276kJ/kg is applied between 1260 and 1660°C, it has a specific heat of 605 $\frac{\text{J}}{\text{kgK}}$, a density of 8220 $\frac{\text{kg}}{\text{m}^3}$ and a temperature dependent conductivity as shown in figure 2.1:

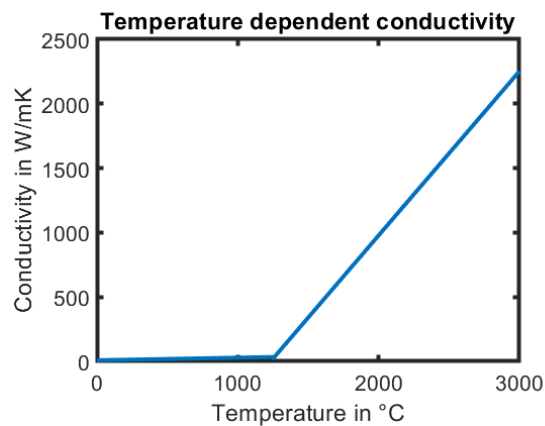


FIGURE 2.1: Modelled conductivity of HastelloyX

This representation of conductivity was chosen to simplify the convective heat transfer inside the molten metal. This way only conductive heat transfer could be used inside the domain. It also made the simulation convergence more stable (for more detail see [20]). There are cubic regions in the domain filled with powder that have a different conductivity

of $0.124 \frac{W}{mK}$ at $25^\circ C$ which increases to $0.169 \frac{W}{mK}$ at its liquidus temperature of $1260^\circ C$. If heated beyond this point, the powdered material switches to become "dense" material, i.e. it is no longer powder and follows the curve above.

The top layer of the domain is covered by a $0.03mm$ thick powder layer and a LASER is moving in a straight line along the center of the top surface in x-direction. The LASER process parameters can be seen in table 2.1.

Process Parameter	Value
Laser Source	spherical Goldak
Laser beam diameter d	$0.055mm$
Laser Power P	$200W$
Absorption η	70%
Penetration depth c	$0.1mm$
Starting Point	$x_0 = -1$
Laser Speed v	$900 \frac{mm}{sec}$
Domain size	$x \in [-1, 1.8]$
Domain size	$y \in [-1, 1]$
Domain size	$z \in [-1, 0.03]$

TABLE 2.1: Process Parameters

A Goldak heat source model [39] is a way to represent the Gaussian distribution of the absorbed energy of a Laser source and can be seen in figure 2.2 from [40].

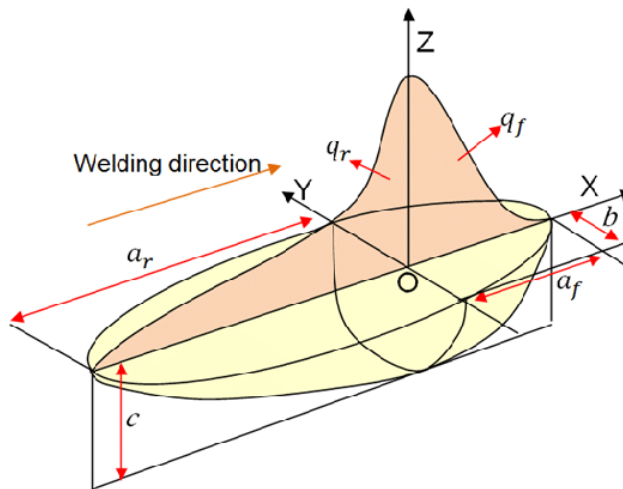


FIGURE 2.2: Goldak Heat Source Representation (elliptic)

$$q(x, y, z, t) = \frac{6\sqrt{3}\eta P}{\sigma^3\pi\sqrt{\pi}} \exp\left(-3\frac{(x+vt)^2 + y^2}{\sigma^2}\right) \exp\left(-3\frac{z^2}{c^2}\right) \quad (2.1)$$

Where σ is the radius and v the speed of the Goldak source.

At the top surface a convective flux and radiative heat transfer were applied, where the ambient temperature is always 25°C, the heat transfer coefficient is $25\frac{\text{W}}{\text{m}^2\text{K}}$ and the emissivity of the material is equal to its absorptivity. The rest of the domain boundaries were considered insulated.

The surrogate must be able to accurately predict the temperature distribution of the steady state for different powder arrangements. Steady state here means that the shape of the temperature field is only experiencing a translation along the laser path and no change in geometry. Accurately means that the maximum error in the domain is below a given threshold t

$$E = \frac{|T_{Pred} - T_{FEM}|}{\max(T_{FEM})} \quad (2.2)$$

$$\max(E) < t \approx 5\% \quad (2.3)$$

2.2 The Dataset

Building on the effort of the previous students [20], we had access to a dataset consisting of two times 30 FEM simulations implementing the described problem. One half consists of powder pockets in the corners of a 2.8x2x1mm domain, the other half of powder pockets in the middle of the domain. The FEM simulations have been cut in the beginning to only leave the steady state solution of the heat source moving across the domain. Both the heat-up and the cool-down phase were cut away. This made the training more easy and focussed on the part of interest. The simulations have been using the mesh defined in figure 1.5 and were the base for all training data.

2.3 PCA+PCE setup

In the previous work the FEM simulations had been replaced as a combination of PCA dimensionality reduction and PCE surrogate. The first step was to establish the performance of this approach and recreate the results. Previous work had represented powder pockets as six values, three for the center coordinates of the pocket and three values for the size in each of the coordinate axis. PCA was then used to find the six eigenvalues of the entire dataset that would represent it the best. It can be considered an informed dimensionality reduction as existing knowledge about the shape of the powder (cuboid)

was used to reduce it down to six values. PCA just looked in the 6 dimensional space for the best combination of eigenvectors to represent it. The six dimensional input of the PCE then gave as output the temperature evolution of the entire domain.

The alternative is an algorithmic dimensionality reduction using PCA to determine the eigenvectors of the dataset and only choosing those vectors corresponding to an eigenvalue above a certain threshold, as described in the introduction. Human intuition fails in high dimensions [41], so it is unlikely to choose the right variables that best reduce a dataset. In the case of powder pockets, the mesh described in 1.5 consists of over 70'000 elements, each either powder or dense. If PCA is applied and then reduced, a set of 16 eigenvectors remains. This is better for the PCE to work with as it can draw more information from this, but it is still a small number of inputs to allow a fast surrogate. Additionally it allows for different shapes of powder distributions besides cubic.

PCA was also used to reconstruct the surrogate predictions. For this the training output was reduced, and the same vectors were applied to the PCE output for reconstruction. Also a short venture into Kernel PCA was taken. It was quickly clear that hyper parameters have to be learnt for this dimensionality reduction method. Otherwise there cannot any reasonable reconstruction which is paramount in the combined approach of PCA and PCE or even a guaranteed reduction of the data. This requires large amounts of data. Thus, one might as well turn to neural networks for modeling the process and have the added benefit of having a model with changing parameters. As described in the introduction one can think about using a combination of auto-encoders and PCE. As shown in the results, that PCE is able to model the process down to a couple percent of error given PCA as dimensionality reduction. The shortcoming of auto-encoders is similar to Kernel PCA, as there are many hyper-parameters to be learned. It cannot be simply reversed, one has to learn the best set of weights and biases for encoding and decoding. For this however the existing data set is not large enough. There are some established ways to deal with small data sets like transfer learning [42], but this is not applicable here. An alternative is as described above to use unsupervised methods, that learn the desired behavior without the need of a training set. Examples of this are theory-guided auto encoders [36] or physics informed neural networks [43].

2.4 PINN

The reason for trying out PINN was the cost of generating new data points. PINNs are able, under the right model setup, to create their own dataset by sampling the PDE loss from a given collocation point. As described before, it is thus an unsupervised or semi-supervised method, if support points are used. The networks were initialized using

the Xavier or Glorot initialization as it is superior to an uniform initialization especially for deep networks [44]. As neural networks can predict better when they are solving in non-dimensional variables, the PDE was converted to variables in the order of magnitude between [-1,1] [45].

$$\frac{du}{dt} = \frac{d}{dx} \left(k \frac{du}{dx} \right) + q \quad (2.4)$$

$$\begin{aligned} u &\in (0, \hat{U}) \\ t &\in (0, \hat{T}) \\ x &\in (0, \hat{X}) \\ q &\in (0, \hat{Q}) \end{aligned} \quad (2.5)$$

Where the variables with hat are the maximum expected variables. So we can transform

$$\frac{du}{dt} = \frac{d\bar{u}}{d\bar{t}} * \frac{\hat{U}}{\hat{T}} \quad (2.6)$$

with

$$\begin{aligned} \bar{u} &\in (0, 1) \\ \bar{t} &\in (0, 1) \end{aligned} \quad (2.7)$$

As the domain boundaries are already within the acceptable margin for the network only time and temperature has to be normalized for performance. Equation (2.3) is put in (2.1):

$$\frac{d\bar{u}}{d\bar{t}} * \frac{\hat{U}}{\hat{T}} = \frac{d}{dx} \left(k \hat{U} \frac{d\bar{u}}{dx} \right) + q \quad (2.8)$$

\hat{U} is constant, so can be taken out of the space derivative.

$$\frac{d\bar{u}}{d\bar{t}} = \cancel{\hat{U}} \frac{d}{dx} \left(k \frac{d\bar{u}}{dx} \right) \frac{\hat{T}}{\cancel{\hat{U}}} + q * \frac{\hat{T}}{\hat{U}} \quad (2.9)$$

The PINN is then calculating and reducing the residual of equation (2.8)

$$\mathcal{R} = \frac{d\tilde{u}}{d\bar{t}} - \frac{d}{dx} \left(k \frac{d\tilde{u}}{dx} \right) \hat{T} - \frac{q\hat{T}}{\hat{U}} \quad (2.10)$$

Here \tilde{u} is the prediction of the PINN. U_{max} is be the expected maximum temperature of the output and can be tuned. In the case of the original problem statement it was set to 2400.

2.4.1 Benchmark Testing

In order to ensure correctness of the neural approach, the validity of the predictions was confirmed at several steps of complexity. This allowed a steady progression of the code while being sure that the predicted results are correct.

2.4.1.1 1D Heat equation with sinusoidal initial condition

To establish the functionality of the PINN a 1D model was employed. This used a sinusoidal initial condition and Dirichlet boundaries enforcing the value 0. This was easily learned and can be seen in fig. 2.3

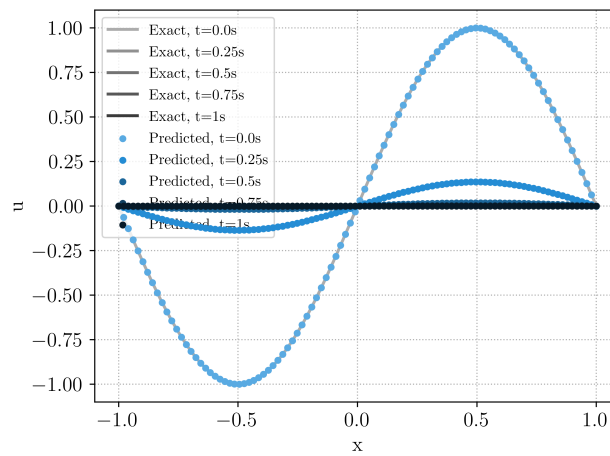


FIGURE 2.3: 1D Prediction vs Exact Temperature

It is apparent that this is not a very hard problem to solve, but it is important to increase complexity slowly to make sure the code is in right working order.

2.4.1.2 2D Heat equation with Dirichlet and von-Neumann boundary conditions

Next a 2d domain was simulated and the boundary condition on one side was switched from Dirichlet to von-Neumann, deploying an insulated boundary on the bottom side. It is important to test the functionality of insulated boundaries as in the final simulation all boundaries will have von-Neumann boundaries. The domain was initialized with a fixed value of one. As is visible on fig. 2.4 the network prediction looks closely like the conventional FEM prediction.

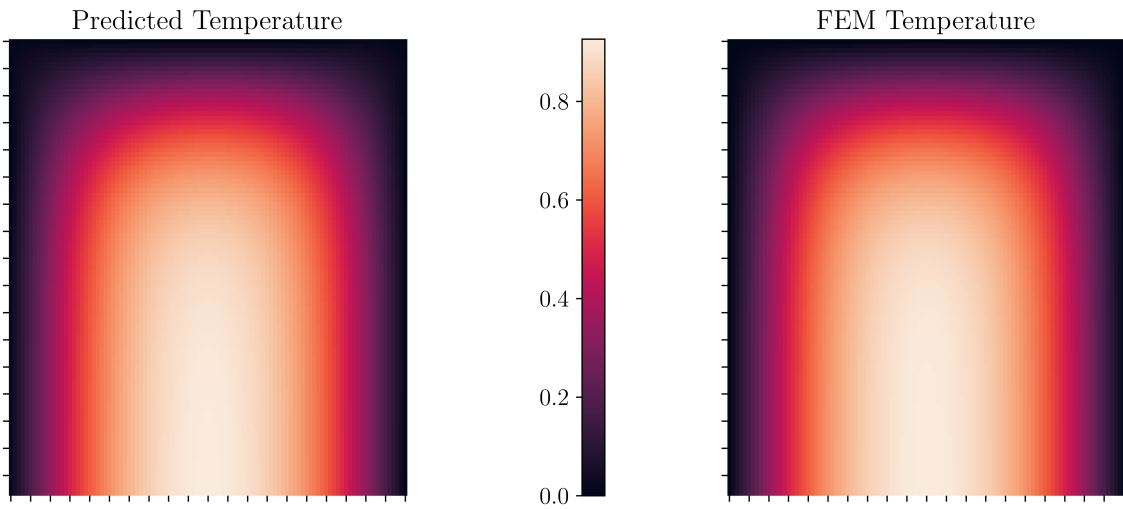


FIGURE 2.4: Predicted and FEM Solution to the Boundary Problem

Directly comparing PINN with the FEM resulted in a 2% error as can be seen in fig. 2.5. It is clear that 2% is an acceptable albeit large error for such a simple problem, but in order to converge to a smaller error one needs to employ hyper-parameter tuning using Ensemble training. It is not urgent to get the error down further for such a simple case, as it the goal was to show that the network can "accurately" predict the physics of the problem. Notable is the asymmetry of the prediction, as the PINN is using gradient descent to reduce the residual, the network might converge to asymmetric local optima.

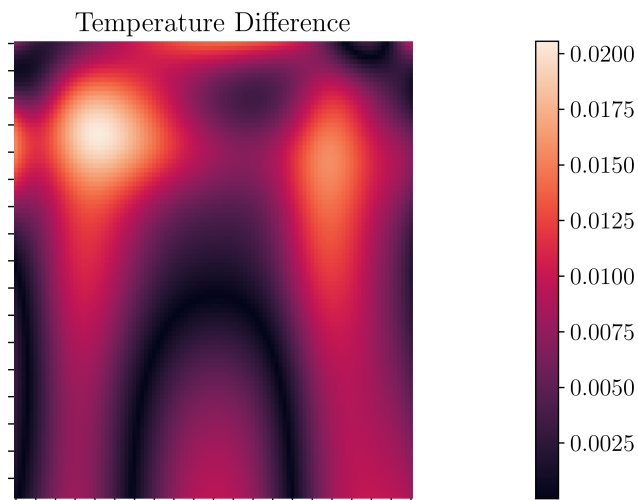


FIGURE 2.5: Prediction Error vs FEM

2.4.2 2D Heat equation with Static Gaussian heat source

The next step was to compare the performance when a static local heat source is introduced. This is an important step as in the desired model a similar source will be moving in time. One can see in fig. 2.6 that the behavior is captured well and the performance is acceptable with some temperature values are deviating by 0.8° or 3% as can be seen in figure 2.7.

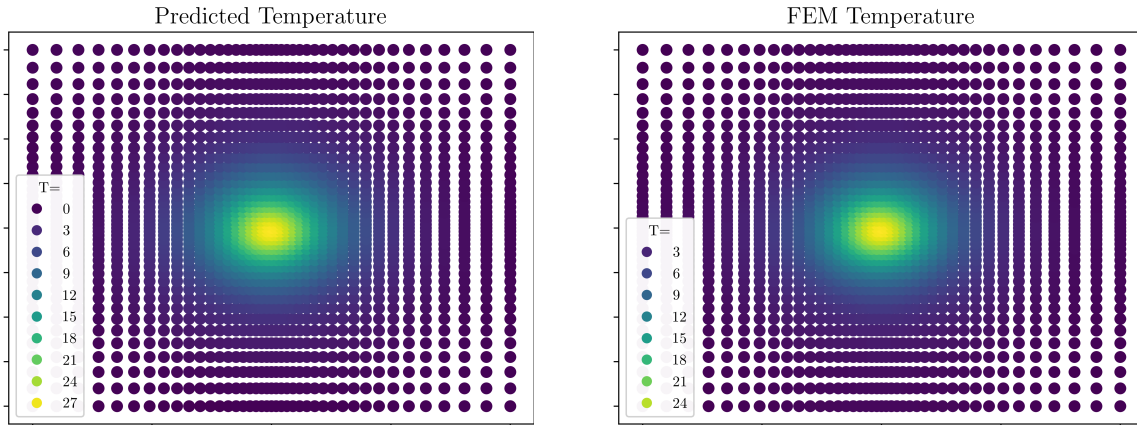


FIGURE 2.6: Predicted and FEM Solution to the static source Problem

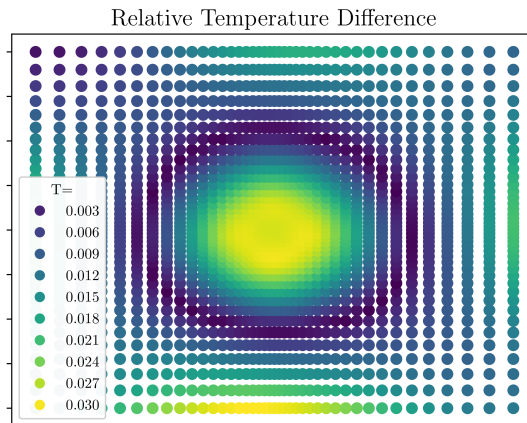


FIGURE 2.7: Prediction Error vs FEM

2.4.3 Implementation of the problem statement

After preliminary benchmark testing the complete problem statement above was implemented in python. This was build upon the PINN infrastructure by Mishra and Molinaro

as in [43]. An environment was created for the PINN to start its training. ANN have difficulties solving any problem that is not differentiable, so every variable in any dimension had to be smooth in the implementation. For the temperature dependent conductivity in figure 2.1 this was done using a four point cubic Bezier curve. This allows for the enforcing of a given slope at the end points (P1 and P4) and a smooth transition between slopes. [46]

$$\begin{aligned}
 P(t) = & (1 - t)^3 * P_1 \\
 & + 3t(1 - t)^2 * P_2 \\
 & + 3t^2(1 - t) * P_3 \\
 & + t^3 * P_4
 \end{aligned} \tag{2.11}$$

Where the range of $t \in [0, 1]$ defines the progression of the interpolated curve. P1 was chosen as 25°, P2=1200°, P3=1300° and P4 as 2000°, this resulted in the conductivity shown in figure 2.8.

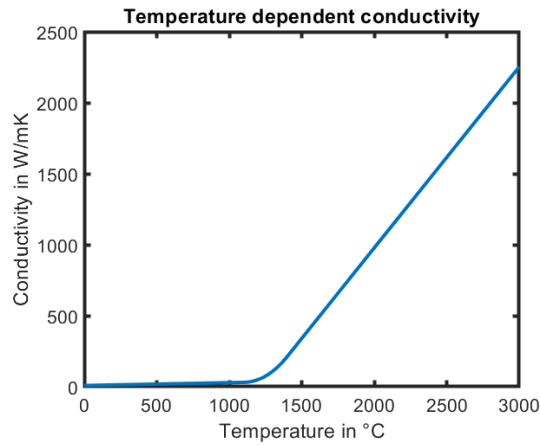


FIGURE 2.8: Smoothed conductivity of HastelloyX

Also, the phase boundary between powder and dense was smoothed using a logistic function as a value of 0 referred to powder state and a value of 1 referred to the dense state. When reaching the liquidus temperature of 1260°C the powder should be completely molten and have transitioned to a dense state.

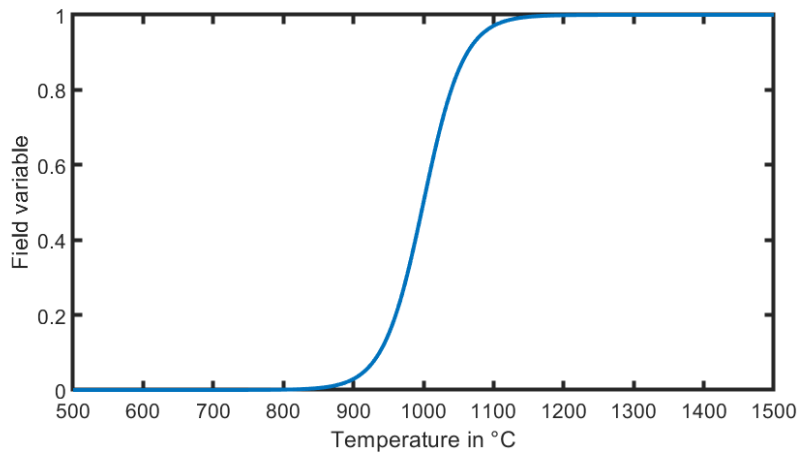


FIGURE 2.9: Smoothed phase transition

Convective Boundary condition are currently not possible with the infrastructure, so the top surface was implemented as insulated. This would have been fixed later if the setup was performing good otherwise. Sadly this was not the case. The results were completely nonsensical, returning negative temperatures:

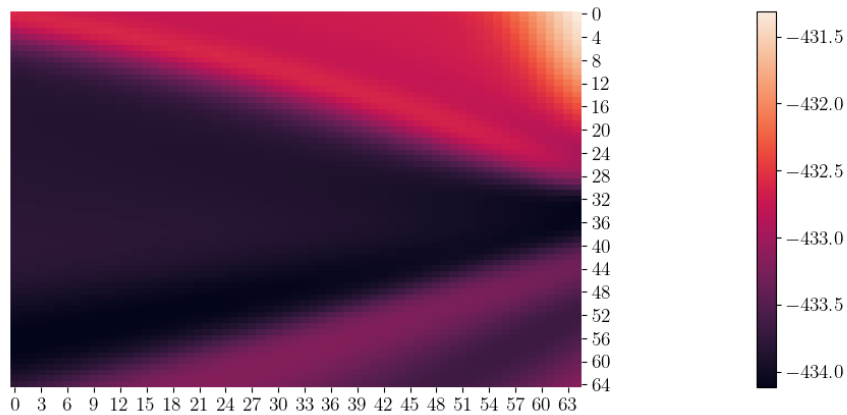


FIGURE 2.10: Failed PINN prediction of the complete problem

2.4.4 The simplified Problem statement

The model had to be simplified to identify the critical element that prevented the convergence of the simulation. The material parameters were simplified and all temperature dependencis were eliminated.

Material Parameter	Value
Density	8220 $\frac{\text{kg}}{\text{m}^3}$
Specific Heat	605 $\frac{\text{J}}{\text{kgK}}$
Constant Conductivity	100 $\frac{\text{W}}{\text{mK}}$

TABLE 2.2: Simplified Material Parameters

Also, the process parameters were changed in order to be able to debug easier and find the problematic parts of the simulation.

Process Parameter	Value
Laser Source	spherical Goldak
Goldak Radius σ	0.05mm
Penetration depth c	0.05mm
Laser Power P	150W
Absorption η	50%
Starting Point	$x_0 = 0$
Laser Speed v	1000 $\frac{\text{mm}}{\text{sec}}$
Domain size	$x \in [-1, 1.8]$
Domain size	$y \in [-1, 1]$
Domain size	$z \in [-1, 0.03]$

TABLE 2.3: New Process Parameters

The PINN still had to be able to simulate the transient heating phase, in contrast to the PCE model, that was able to be trained on steady state FEM simulations. The PINN needs to recreate the entire physical process in order to reach a steady state. However, even when simulating a completely dense domain, with no temperature dependency and simplified parameters, the results did not converge to a sensible result. As can be seen in figure 2.11, where the predicted values are plotted against the correct ones.

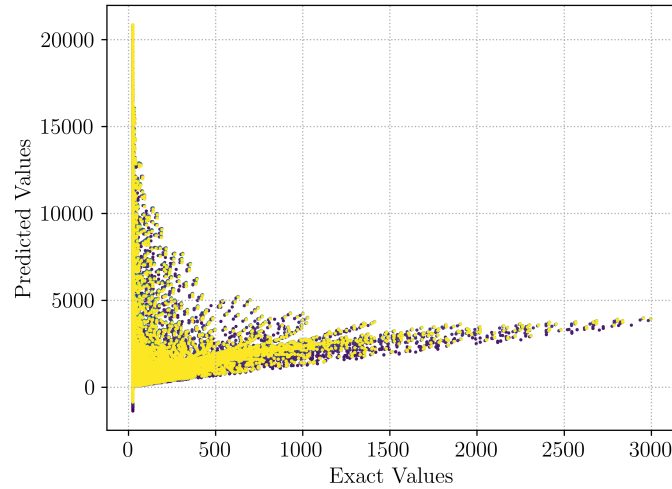


FIGURE 2.11: Failed PINN prediction of the simplified problem

2.4.5 Solution Approaches

2.4.5.1 Scaling of PDE components

Due to the nature of training a network, it is important that all components of a loss function have the same magnitude, as otherwise some components might grow in importance while others fall behind. If component A has 100x the effect on the outcome of the loss than component B, then any change in component B is meaningless. In order to ensure the scales of all components in equation 2.10 are of the same magnitude, the value of the maximum temperature \hat{U} and maximum time \hat{T} are changed to ensure a homogeneous magnitude at training start. The effect of this approach was visible in the temperature output, but did not fix the convergence issues. Hence the issue must be somewhere else.

2.4.5.2 Ensemble training

A neural network is influenced by a number of factors called hyper parameters. This includes the network architecture i.e. layers and neurons per layer, the choice of optimizer, activation function and regularizer as well as the balancing factor in the loss function that groups PDE loss with the boundary losses. It is impossible to guess the right combination on the first try. In ensemble training the parameters are changed within a fixed domain to find the best model architecture. The different parameter sets are then compared against a common metric using cross validation. The assumption was that possibly the chosen PINN architecture cannot learn or even approximate the problem, thus a better architecture has to be found. Cycling through over 200 different settings, we did not see a dramatic performance spike but chose the setup with the best predictive performance

compared to FEM. This is a six layer network with 24 neurons per layer, a tanh-activation function using the LFBGS-optimizer.

2.4.5.3 Adding support points

Adding support points of the tested FEM simulation does not bring any benefit, except to see that the network is practically able to approximate the desired function, as it causes an overfit to the test set. Yet being able to approximate a function is no guarantee that the network can actually learn it [41]. However, one can use an analytic function to generate support points for the PINN that are cheap to generate. This would allow the network to improve its performance by finding a better path through the loss surface. Its weights and biases can then be used to initialize a new network that can be trained without support points. For this the Rosenthal equation was chosen. The equation generates an approximate solution for a PDE with a moving heat source. It works with constant conductivity and is thus suited for the simplified problem statement [47]. The Rosenthal equation can be seen in (2.15).

$$\zeta(x, t) = x + v * t \quad (2.12)$$

$$r(x, y, z, t) = \sqrt{\zeta^2 + y^2 + z^2} \quad (2.13)$$

$$a = \frac{k}{\rho * c_p} \quad (2.14)$$

$$T(x, y, z, t) = T_0 + \frac{P * \eta}{2 * \pi * k * r} * \exp\left(-\frac{v * (\zeta + r)}{2 * a}\right) \quad (2.15)$$

Where k is the material conductivity, a the diffusivity, v is the laser speed, q is the laser power, n the absorptivity, t is the current timestep and x, y, z the evaluated locations.

Using Rosenthal support was a good path for helping the network to the right region of the loss surface. It did reduce the error slightly, but did not manage to reduce it below the acceptance threshold of 5%. It resulted in some artifacts on the result, as the support loss was working against the PDE-loss. This is due to the fact that the Rosenthal equation is just an approximation and not an exact solution of the PDE. When looking into the ability of the network to solve exactly the equation for which the Rosenthal equation delivers the solution, another problem became visible.

2.4.5.4 Adaptive Sampling Strategy

The region affected by the laser is very small in comparison with entire domain. The laser affected volume is only 0.00454% of the entire domain. Thus if sampled uniformly,

only 23 of half a million points would sample the laser. As the PINN is only trained on points that were sampled from the model infrastructure, it is important that enough points from the affected area are sampled consistently over time. By printing out the maximum sampled source term from training to training, it was clear that there was a big stochastic influence, as it varied by up to 10%. This showed clearly that there were inconsistencies and a better sampling strategy had to be found. The standard sampling had been a Sobol low discrepancy sampling method, as this leads to greater accuracy than on random points [48]. First only exchanging the sampling in y-direction with a normal distribution centered on 0 and a standard deviation of 0.1 such that the sampled points would be mostly in the center. This reduced the maximum sampled source variability to 5% but this was still not acceptable. The next step was moving to an adaptive sampling, where the center of the distribution was on the laser location in the x-axis. For the ease of implementation a triangular distribution with the limits at the domain boundaries was used.

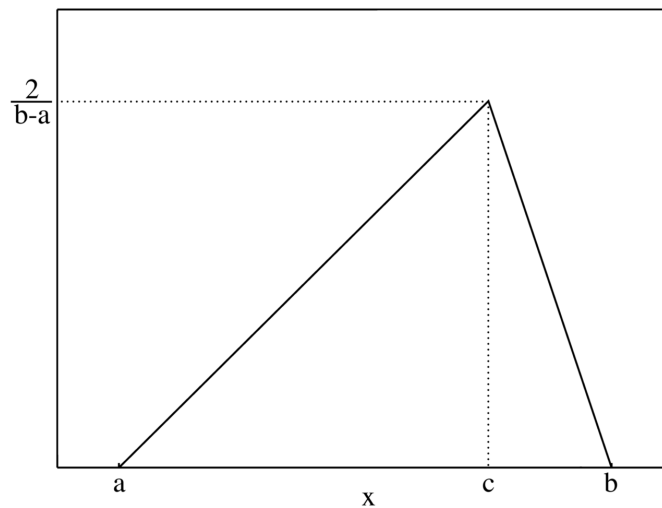


FIGURE 2.12: Probability Density of a Triangular distribution

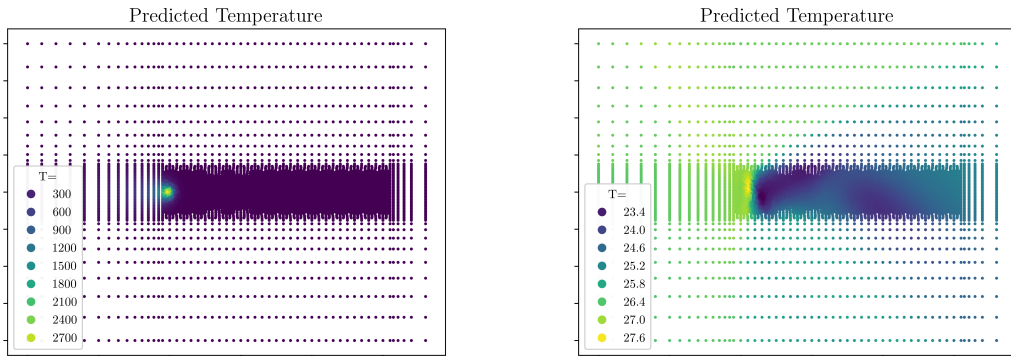
Additionally a half-normal distribution with a shifted mode at 0.03 is used for the z-axis, as most points are needed at the surface of the domain. This results in a collocation point distribution that can be seen animated below (animation works only on Adobe Acrobat reader). The orange dot represents the laser center. The collocation points are binned together for each of the frames, as they are smoothly distributed through time.

FIGURE 2.13: The adaptive sampling for capturing the Source

This brought down the prediction error dramatically to around 3-5%, and even eliminated the need for any support points. Thus effectively it is again an unsupervised method. However it did not consistently predict the correct maximum temperatures, maximum predictions fluctuated with $\pm 80\text{K}$ over time. As we saw such a big impact due to changing the sampling strategy, more effort was put on optimizing the sampling strategy to remove this inconsistency. Adding a few points directly in the geometric center of the laser spaced out at regular intervals removed network ability to train with as little as 10 points. These added points in or near the center will be called center bias from now on. If noise was added to the center bias, the easier the network could train. The maximum predicted temperature were slightly more consistent if a center bias was used once the network managed to converge. In this report consistent maximum predicted temperatures are the main measure for the consistency of the network predictions depending on the sampling strategy, as it strictly depends on the knowledge of the network on the spatial and temporal distribution of the source. From the FEM simulations we know the maximum temperature should be steady state and thus not fluctuate at all.

2.4.6 Ensuring correct initial conditions

Another issue came up with the initial conditions, as the initial-loss wants to enforce a constant temperature of 25°C , but the PDE-loss has a high power source active at $t = 0$, these two losses are working against each other. In order to arrange the two losses some expedient solutions have been implemented. Firstly, the time the simulation starts has been moved to $t = -0.1$ and the source term is activated at $t = 0$. This way there is some temporal leeway for the two losses. Additionally, the source is ramped up using a logistic function over the first 5% of the time. This allows the differentiation of the source term, as is required by the network. These workarounds do not suffice to enable a homogeneous initial condition, they do however ease the disconnect between the two competing loss functions. Instead of temperature discrepancies of $+2000^{\circ}\text{C}$, the differences are now in the order of $\pm 10^{\circ}\text{C}$



(A) Initial prediction without measures

(B) Initial prediction using all expedient measures

FIGURE 2.14: Comparison of initial predictions

Chapter 3

Results and Discussion

3.1 PCA + PCE

As described above, the predictive performance of a six variable input (location and size of powder cuboid) to PCE was compared with a 16 Variable input. The 16 variables were derived from the entire field i.e. an 70'000 element set using PCA. To ensure a good generality, the data set was used to create 30 different splits for training and testing. This ensures that there is no bias in the training set to allow for better prediction. As one can see in figure 3.1 there exist very adversarial splits for the six variable input. In general only the entire field input fulfills the quality requirement for the surrogate model, as the error is above 5% for the six available input. The graphic only shows the error for FEM temperatures above 300°C as the majority of the domain is at 25°C, it is much easier to predict low temperatures.

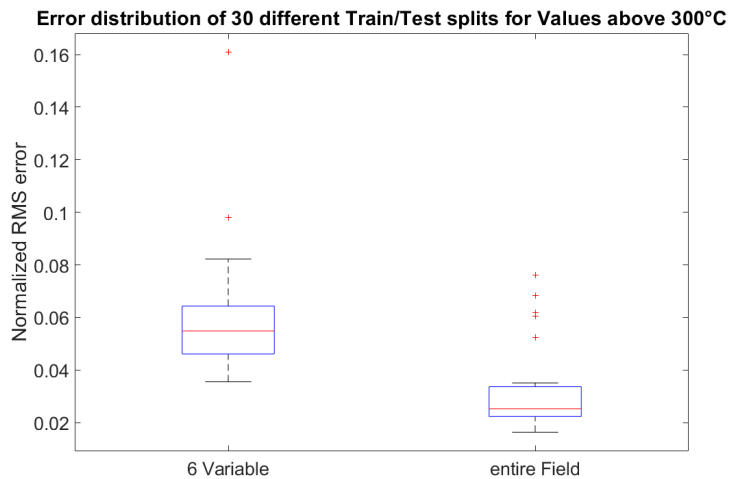


FIGURE 3.1: Limited Error of the 6 variable vs. full field approach

Alternatively, the maximum prediction error in the entire domain shows a similar image. The variance for the entire field approach is even lower. The maximum temperature in the domain is 2400°C so the lowest maximum error is around 4.17%, the average maximum error is below 5% and the maximum error in all the splits for the entire field is at 8%. This can be counted as an outlier and half of the total maximum error of the 6 variable approach.

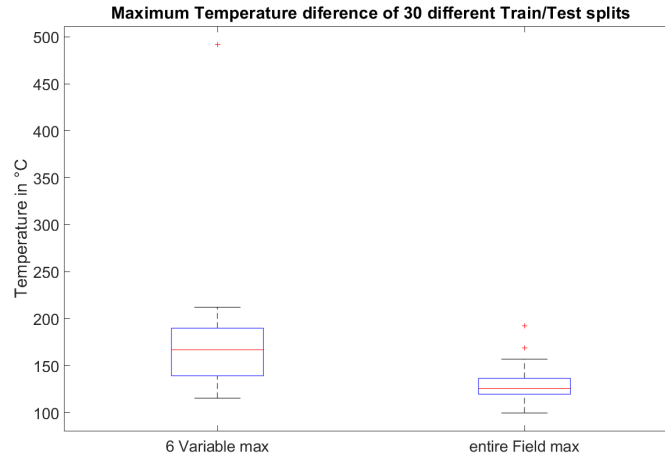


FIGURE 3.2: Maximum Error of the 6 variable vs. full field approach

The results from the Kernel PCA are not very informative as a hyper parameter optimization was missing to get the best reduction and reconstruction performance. Unfortunately, a server syncing malfunction deleted the codes and further result of this evaluation, while work had already progressed much further into programming physics informed neural networks. As this seemed more valuable than recreating the lost code, no other results can be shown for the PCA and PCE model at this point.

3.2 PINN - Physics Informed Neural Networks

The performance of the simplified problem is not as easily compared as the PCE performance, as the PINN also had to predict the transient heat-up phase. An accurate transient prediction was part of the performance metric. Usually the L2-norm is chosen to compare how well a network predicts:

$$L2_{relative} = \frac{\sqrt{\frac{\sum_{i=1}^N (T_{i,Pred} - T_{i,FEM})^2}{N}}}{\sqrt{\frac{\sum_{i=1}^N (T_{i,FEM})^2}{N}}} \quad (3.1)$$

In this case the relative L2-norm shows 7.37% error, so above the performance threshold. However, this is comparing all errors over all time steps over the entire domain, normalized with the average temperature over all time steps and entire domain. This means that the error is normalized not with the maximum temperature, but with some diminished value. The simulation should predict the high values accurately. That is why the following animation 3.3 shows the difference normalized with the maximum FEM temperature value in that time step. (Animation works only on Adobe Acrobat reader)

FIGURE 3.3: Relative difference of the prediction vs. FEM

In the steady state domain the results are fulfilling the acceptance criterion, with errors of less than $\pm 2.5\%$. However, one can see that especially during the first frames the error values are above the acceptance threshold. This is due to the fact that the PINN had enormous difficulties with enforcing the initial conditions and accurately predicting the transient period. This is also visible in the following figure 3.4, where the network was reinitialized ten times to ensure a retraining stability.

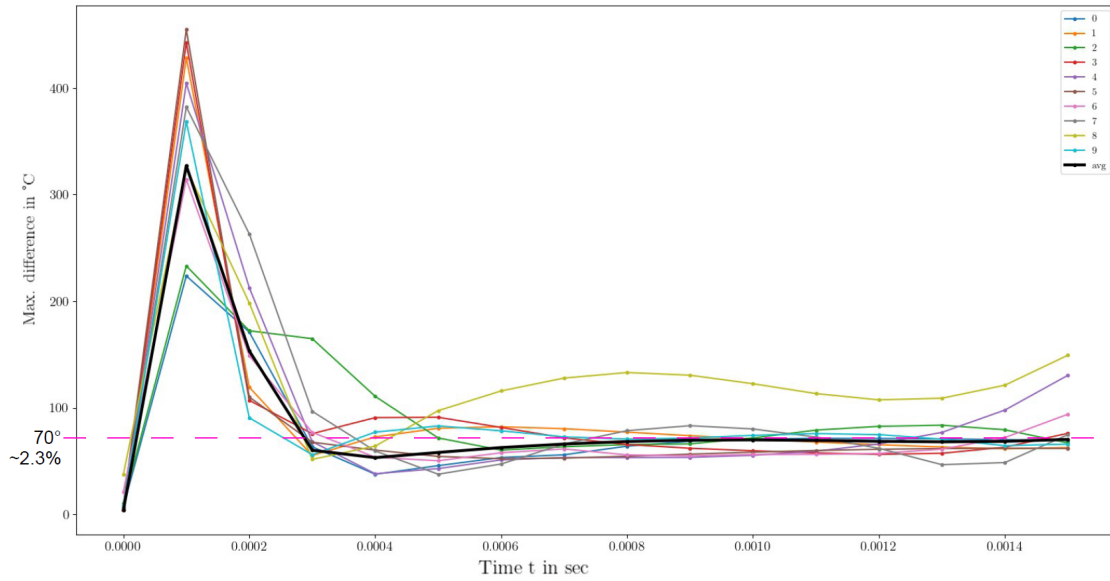


FIGURE 3.4: Maximum temperature difference PINN vs. FEM

The average maximum prediction error in the steady state is 70°C or 2.3% relative error. It is easily visible that the first few frames have maximum temperature differences in the domain that are not suitable for a proper transient surrogate model. If one looks at the maximum predicted temperatures in the domain the error becomes lower.

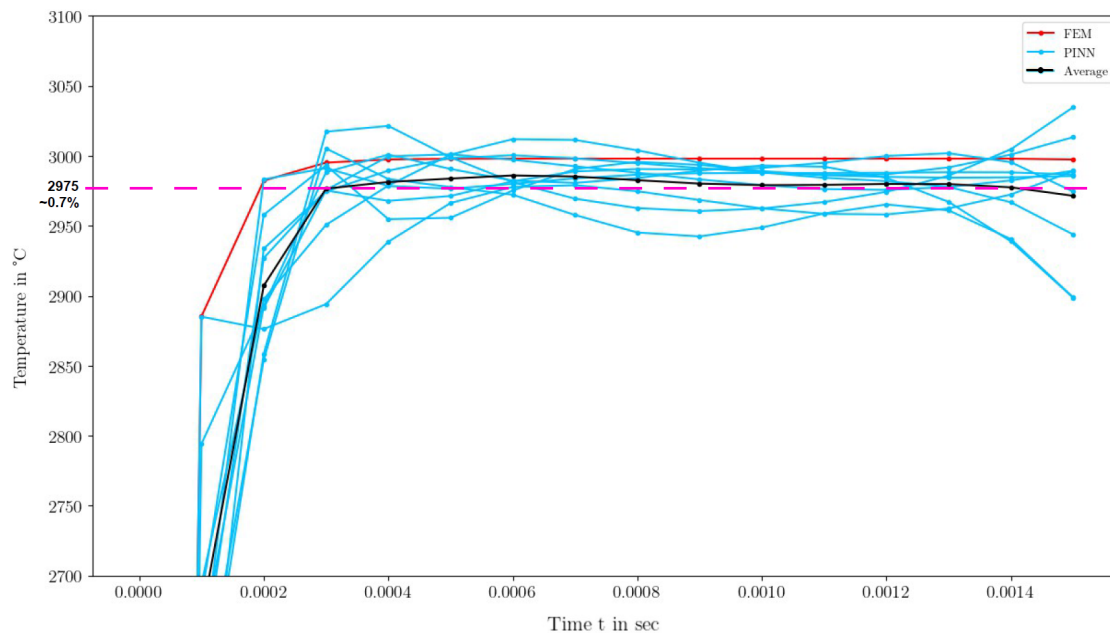


FIGURE 3.5: Maximum predicted temperature PINN vs. FEM

It is visible that despite adding a center bias as described in the Experimental Setup section, there is still some fluctuation in the maximum predicted temperature and further work is required.

3.2.1 Parametric Model

A big advantage of the PINN concept is that it is possible to make process or material parameters an input variable in a certain range. This allows uncertainty quantification and sensitivity analysis, without the need of rerunning the FEM simulations and generating large result files in the order of Gbyte. The machine learning model has a size of only 2Mbyte but can evaluate any point within the temporal, spacial and parametric domain. This was tried for conductivity in the range of $k \in [10, 200]$ and the model fulfilled the acceptance threshold even when looking at the relative L2-norm from equation 3.1 of the steady state, i.e. the first 4 frames were removed from analysis.

Conductivity in $\frac{W}{mK}$	Relative L2-Norm
10	0.1016
50	0.0423
100	0.0347
150	0.0458
200	0.1935

TABLE 3.1: Relative L2-norm of different evaluated conductivities at steady state

It is clearly visible that at the range limits the L2 error spikes. This is due to the fact that no special sampling strategy was chosen for the parameter space, thus there might be less sampling points at the boundaries and the performance is poorer than for the center values. However, just as above, the relative L2-norm is not normalized with the maximum value but with a skewed average. When normalizing with the maximum value of the FEM, all compared conductivities have less than 3% error for the steady state. Only for $k = 10 \frac{W}{mK}$ is the error close to the acceptance threshold at 5%. The main advantage of parametric models is the ability for sensitivity analysis, such as shown in figure 3.6 where the temperature evolution of the point at $x = 0.43, y = 0, z = 0.03$ is shown.

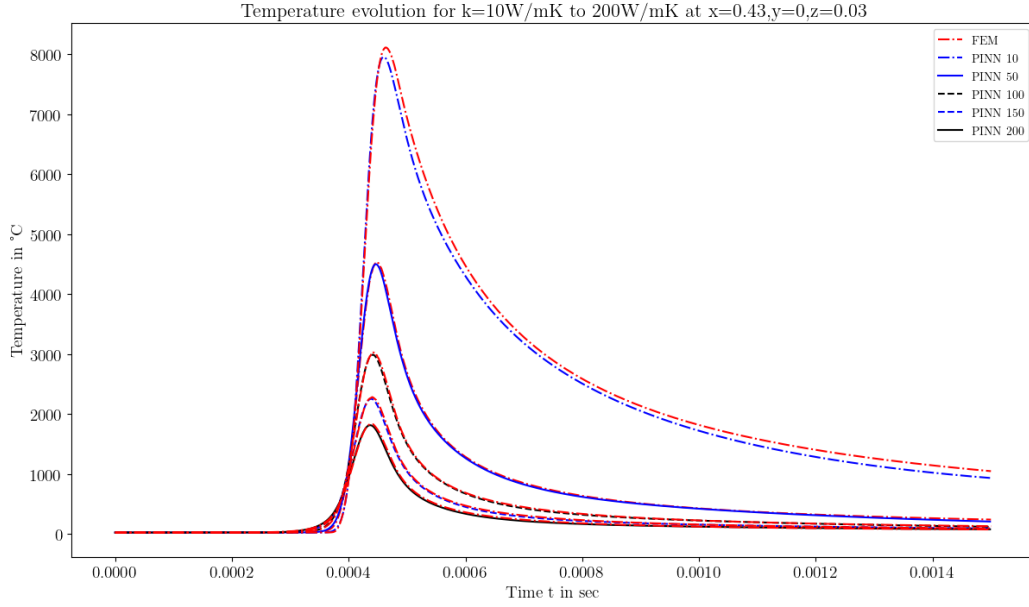


FIGURE 3.6: Temperature Evolution Analysis of different Conductivities

One can see that the PINN prediction closely matches the FEM simulation, with the exception of $k = 10 \frac{W}{mK}$, where the PINN did not capture the maximum temperature. This might be due to several reasons, most likely the sampling did not capture the source correctly at this location and thus underpredicted. The PINN performance can also be seen in the "Predicted vs. Exact" figure 3.7, where the color of the points refers to the progression in time. It is easy to see that the transient performance gets worse the lower the conductivity is, and thus the higher temperatures the simulation has to predict. Once the simulation has reached the steady state, the performance seems nominal except for the last image. Here once again the edge of the defined parameter space at $k = 10 \frac{W}{mK}$ shows poor transient performance over the domain. Noteworthy is here the changing scale from 1750°C maximum temperature at $k = 200 \frac{W}{mK}$ to 8000°C for $k = 10 \frac{W}{mK}$.

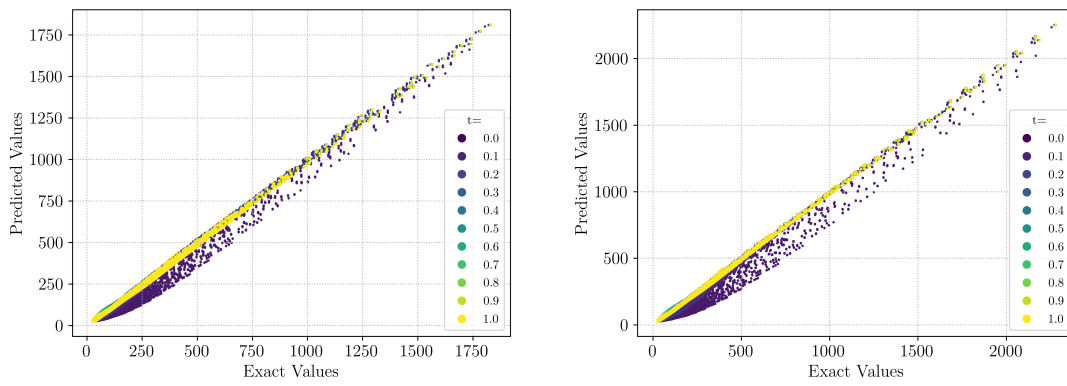
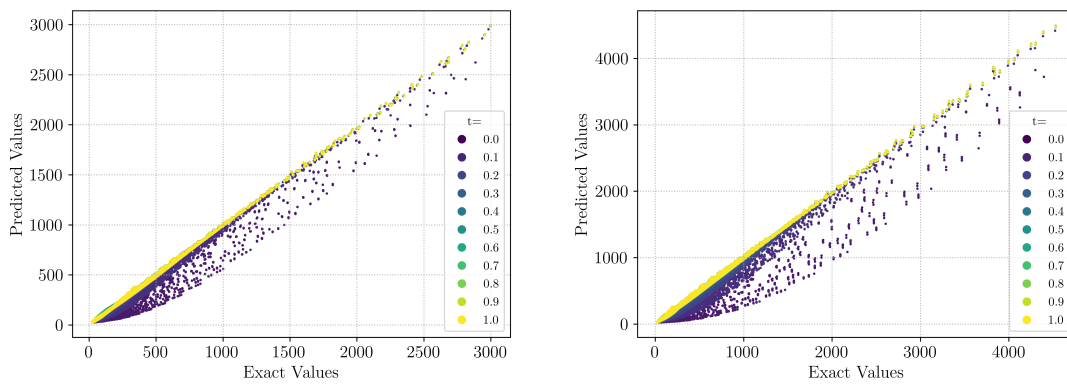
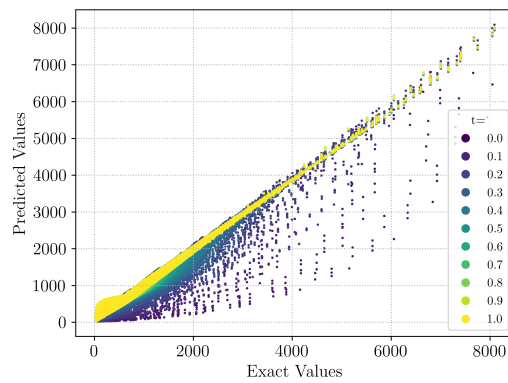
(A) $k=200$ and $k=150$ (B) $k=100$ and $k=50$ (C) $k=10$

FIGURE 3.7: Parametrized model performance vs. selected FEM results

In conclusion though, the parametrized model fulfilled the acceptance threshold across all evaluated conductivities, and even has the ability to evaluate any conductivity in the given range without any further training. Which leads to the question of training cost:

Simulation	Computational cost
5 FEM (15 output frames for entire domain)	CPU: 31.6h
5 FEM (1000 output frames for 1 point)	CPU: 30h
All FEM simulations used	CPU: +60h
PINN Training	CPU: 9.4h
PINN Training	GPU: 0.4h

TABLE 3.2: Computational cost for FEM vs. PINN

PINN evaluations can be performed anywhere within the definition space and are not bound by predefined evaluations of conductivity. Each parameter can be smoothly changed. If one would use a CPU to train the PINN it would outperform even 5 evaluations using FEM (for a given time step size). This means the ratio of 3.4 : 1 in computational time would grow even larger if more values were compared. However, if one would use the more fitting GPU architecture the ratio grows to 78.8 : 1, assuming that GPU could bring a 4x speedup in computational time for FEM simulations [49][50] then the ratio would still come to 18.7 : 1. These numbers show the big promise that parametrized PINN can bring for scientific analysis.

Chapter 4

Conclusion and Outlook

The goal of this thesis was to improve on the surrogate model from the previous thesis. The surrogate model is supposed to dramatically reduce the computational cost required for calculating the thermal history of the powder under a laser pass in the millimeter domain. Due to the manufacturing process, not the entirety of the domain is filled with dense material, some part of it is filled with powder. Powder has a much lower thermal conductivity than solid material, so the surrogate model must be able to generalize for different powder distributions within the domain. The previous work had shown the promising capabilities of a combination of dimensionality reduction coupled with a polynomial chaos expansion (PCE) that models the model output as distribution and selects the most likely result. PCE is a supervised method that treats the model as a blackbox and associates a selected input with a desired output. It uses preexisting (simulation) data to create a predictive model using optimization of the PCE parameters. Due to the complex nature of the problem, the dimension does not only need to be reduced but also reconstructed after the PCE. Linear Principal Component Analysis (PCA) allows to reconstruct data using the outputs of the training data. More efficient kernel based reconstruction techniques might exist but they require large amounts of data to train the correct set of hyper parameters for accurate results. A combination of Kernel PCA for dimensionality reduction and linear PCA for reconstruction could be a possibility that has not been tested. As it is easier to obtain a good set of hyper parameters for dimensionality reductions than for reconstruction. This thesis has shown that it is beneficial to not make any assumptions regarding the best choice input variables. It is best to let PCA draw the most information from the raw data and hand this to the PCE as input. A pre-informed abstraction might unintentionally remove data from the model and make it harder to train. This shows that feature engineering is the key to a good prediction [41]. In this thesis on average the model reached relative mean square errors below 3%. However, the blackbox approach of PCE always requires training data from simulations or experiments.

This implies, that once the simulation parameters of interest are changing, a new training data set needs to be created. This means added computational costs.

The second question that this thesis asked is if it is possible to find an unsupervised machine learning algorithm that accurately solves the problem statement. The relatively new algorithm called "Physics Informed Neural Networks" (PINN) uses the autodifferentiation feature of Artificial Neural Nets(ANN) to solve the partial differential equation (PDE) that is also solved by the FEM-solver. This way the PINN treats the PDE-solver as blackbox and the PDE-residual as loss. It then approximates the best solver by reducing the PDE-residual. The advantage of this is, that no training data is needed at all, only an implementation of the PDE. The PINN then samples inputs to the PDE and trains itself. The general infrastructure of the problem statement PDE was implemented. However not all features of the infrastructure could be tested. The predictive problems at the beginning of the process required a simplified problem statement. This removed temperature dependency and spacial dependency of material parameters. More research has to be go into what is missing or needs to be adapted to make them work. My evaluation is that spacial dependency is probably easier to learn, as it is dependent on an input parameter. Temperature is an output parameter, so dependencies can only be iteratively evaluated. That is to say, that the output gets worse as a result of being wrong, making learning difficult. Phase transition using latent heat, convective and radiative boundary conditions are also not implemented as all of them are temperature dependent phenomena. Mishra and Molinaro have shown that PINN can learn radiative transfer [43]. So one might experiment with adding several ANN or PINN together that each solve parts of the PDE and one PINN that assembles all results. Alternatively, a transfer weight initialization of another model that is trained on a simpler problem might aid the difficulties in training. Also support of an approximate solution has the same effect, albeit a continued training without the approximate support is needed in order to remove artifacts trained on by the approximation.

Using a simplified problem was a good practice however as this unveiled the importance of a proper sampling strategy. Simply sampling equally all over the domain requires very large amounts of points for a dense cover. The laser source has a very small spread compared to the dimension of the domain, and within the laser there is a gaussian distribution of power density. This means the location with highest power is even smaller. However, enough evaluations at this location are necessary for the PINN to learn the behavior of the laser source. Once this was discovered, an adaptive sampling strategy was implemented. This made the PINN able to predict with a maximum error of 3% without any training data. The strategy samples more points where the laser center is and moves with the laser. The predictive performance of different distributions needs to be evaluated. The addition of a center bias allows for the use of two different distributions, one for the larger

domain and one close to the laser center. One might also think about an adaptive sampling density that changes with expected temperature. Higher temperatures point to the existence of the laser in that region, and more points would be sampled from that region. This could be done with a variation of the Rosenthal equation. Once this was solved, a model was created that took a material parameter (conductivity) as input. When implementing powder pockets, the location of the powder has to become a network input, so it is important to gain experience with parametrized models. Additionally, it might give further insight on what happens when dealing with variable conductivity. The result was satisfying as all evaluated conductivities fulfilled the acceptance criteria of max. 5% relative error and showed the promise of cheap sensitivity analysis. This is due to the fact that a parametric model can smoothly change the parameter value and repeat analysis on the entire domain without the need for additional training. It needs to be evaluated what maximum the number of parameters or degrees of freedom (DoF) is that still produces accurate results and what domain size they can cover. Each DoF also needs a custom sampling strategy that samples the salient points. When spacially dependent conductivity (powder pockets) become enabled, using support data of different powder pocket locations would further decrease the error. Currently the PINN has always been trained on the maximum points possible with the available GPU hardware. However, it needs to be shown what the required number of collocation, initial and boundary points is for best performance. Lastly, the predictions need to be compared to analytical melt pool size estimates [51] and experimental data from literature [52].

Chapter 5

Appendix: Code Base

As the code for the PINN is quite segmented, it is not conducive for understanding to attach it all here. All the code can however be found on the gitlab of this project [gitlab.ethz.ch/olemuell/laser-thermal-pinn]. It is a python code, based on pytorch.

My main contribution, the equation models (both the fixed and parametrized version) that form the learning environment for the PINN can be found in the folder "EquationModels". The file "GeneratorPoints" contains the adaptive sampling strategy. This far it is hard coded for the defined problem. To start the learning process from scratch the file "PINN2" has to be executed.

As this project is still ongoing with the purpose to publish a paper, the code might still change slightly after the submission of this thesis.

Bibliography

- [1] Ans Al Rashid, Shoukat Alim Khan, Sami G. Al-Ghamdi, and Muammer Koç. Additive manufacturing: Technology, applications, markets, and opportunities for the built environment. *Automation in Construction*, 118:103268, 10 2020. ISSN 09265805. doi: 10.1016/j.autcon.2020.103268.
- [2] William E. Frazier. Metal additive manufacturing: A review. *Journal of Materials Engineering and Performance*, 23:1917–1928, 4 2014. ISSN 15441024. doi: 10.1007/s11665-014-0958-z. URL <https://link.springer.com/article/10.1007/s11665-014-0958-z>.
- [3] Pablo Zapico, Sara Giganto, Joaquín Barreiro, and Susana Martínez-Pellitero. Characterisation of 17-4ph metallic powder recycling to optimise the performance of the selective laser melting process. *Journal of Materials Research and Technology*, 9: 1273–1285, 3 2020. ISSN 2238-7854. doi: 10.1016/J.JMRT.2019.11.054.
- [4] Richard O’leary, Rossi Setchi, Paul Prickett, Gareth Hankins, and Nick Jones. An investigation into the recycling of ti-6al-4v powder used within slm to improve sustainability. *Sustainable Design and Manufacturing*, pages 377–388, 2015. ISSN 2051-6002. URL <http://www.inimpact.org>.
- [5] Hong-Chuong Tran, Yu-Lung Lo, and Trong-Nhan Le. A strategy to determine the optimal parameters for producing high density part in selective laser melting process, 2019.
- [6] Michal Krzyzanowski and Dmytro Svyetlichnyy. A multiphysics simulation approach to selective laser melting modelling based on cellular automata and lattice boltzmann methods. *Computational Particle Mechanics*, 2021. doi: 10.1007/S40571-021-00397-Y.
- [7] Matthias Markl and Carolin Köhner. Multiscale modeling of powder bed-based additive manufacturing additive manufacturing (am): a process of joining materials to directly build up objects from 3d virtual prototypes, usually layer upon layer (1). 2016. doi: 10.1146/annurev-matsci-070115-032158. URL www.annualreviews.org.

- [8] Yanping Lian, Zhengtao Gan, Cheng Yu, Dmitriy Kats, Wing Kam Liu, and Gregory J. Wagner. A cellular automaton finite volume method for microstructure evolution during additive manufacturing. *Materials and Design*, 169:107672, 5 2019. ISSN 0264-1275. doi: 10.1016/J.MATDES.2019.107672. URL <https://www.scholars.northwestern.edu/en/publications/a-cellular-automaton-finite-volume-method-for-microstructure-evol>.
- [9] Dehao Liu and Yan Wang. Mesoscale multi-physics simulation of solidification in selective laser melting process using a phase field and thermal lattice boltzmann model. *Proceedings of the ASME Design Engineering Technical Conference*, 1, 2017. doi: 10.1115/DETC2017-67633.
- [10] Theron M. Rodgers, Jonathan D. Madison, and Veena Tikare. Simulation of metal additive manufacturing microstructures using kinetic monte carlo. *Computational Materials Science*, 135:78–89, 7 2017. ISSN 0927-0256. doi: 10.1016/J.COMMATSCI.2017.03.053.
- [11] Alberto Cattenone, Simone Morganti, and Ferdinando Auricchio. Basis of the lattice boltzmann method for additive manufacturing. *Archives of Computational Methods in Engineering 2019 27:4*, 27:1109–1133, 6 2019. ISSN 1886-1784. doi: 10.1007/S11831-019-09347-7. URL <https://link.springer.com/article/10.1007/s11831-019-09347-7>.
- [12] Yi Zhang. Multi-scale multi-physics modeling of laser powder bed fusion process of metallic materials with experiment validation. *Theses and Dissertations Available from ProQuest*, 1 2018. URL <https://docs.lib.purdue.edu/dissertations/AAI10844033>.
- [13] P. Bidare, I. Bitharas, R. M. Ward, M. M. Attallah, and A. J. Moore. Fluid and particle dynamics in laser powder bed fusion. *Acta Materialia*, 142:107–120, 1 2018. ISSN 1359-6454. doi: 10.1016/J.ACTAMAT.2017.09.051.
- [14] Dehao Liu and Yan Wang. Multiphysics simulation of nucleation and grain growth in selective laser melting of alloys. 2020. doi: 10.1115/1.4046543. URL https://asmedigitalcollection.asme.org/computingengineering/article-pdf/20/5/051002/6529773/jcise_20_5_051002.pdf.
- [15] Peyman Ansari, Asif Ur Rehman, Fatih Pitir, Salih Veziroglu, Yogendra Kumar Mishra, Oral Cenk Aktas, and Metin U. Salamci. Selective laser melting of 316l austenitic stainless steel: Detailed process understanding using multiphysics simulation and experimentation. *Metals*, 11, 7 2021. doi: 10.3390/MET11071076.

- [16] Mostafa Yakout, M. A. Elbestawi, and Stephen C. Veldhuis. A study of the relationship between thermal expansion and residual stresses in selective laser melting of ti-6al-4v. *Journal of Manufacturing Processes*, 52:181–192, 4 2020. ISSN 1526-6125. doi: 10.1016/J.JMAPRO.2020.01.039.
- [17] Zhiyuan Liu, Dandan Zhao, Pei Wang, Ming Yan, Can Yang, Zhangwei Chen, Jian Lu, and Zhaoping Lu. Additive manufacturing of metals: Microstructure evolution and multistage control. *Journal of Materials Science & Technology*, 100:224–236, 7 2021. ISSN 1005-0302. doi: 10.1016/J.JMST.2021.06.011.
- [18] Saad A. Khairallah and Andy Anderson. Mesoscopic simulation model of selective laser melting of stainless steel powder. *Journal of Materials Processing Technology*, 214:2627–2636, 11 2014. ISSN 0924-0136. doi: 10.1016/J.JMATPROTEC.2014.06.001.
- [19] P. Gh Ghanbari, E. Mazza, and E. Hosseini. Adaptive local-global multiscale approach for thermal simulation of the selective laser melting process. *Additive Manufacturing*, 36:101518, 12 2020. ISSN 2214-8604. doi: 10.1016/J.ADDMA.2020.101518.
- [20] Natalia Smatsi, Prof. Dr. E. Mazza, Dr. E. Hosseini, and P. Gh. Ghanbari. Surrogate modelling for 3d multiscale thermal simulations of powder-bed additive manufacturing. 2020.
- [21] Sarah E. Davis, Selen Cremaschi, and Mario R. Eden. Efficient surrogate model development: Optimum model form based on input function characteristics. *Computer Aided Chemical Engineering*, 40:457–462, 1 2017. ISSN 1570-7946. doi: 10.1016/B978-0-444-63965-3.50078-7.
- [22] Yicheng Zhou and Zhenzhou Lu. An enhanced kriging surrogate modeling technique for high-dimensional problems. *Mechanical Systems and Signal Processing*, 140:106687, 6 2020. ISSN 0888-3270. doi: 10.1016/J.YMSSP.2020.106687.
- [23] Qiuqing Pan and Daniel Dias. An efficient reliability method combining adaptive support vector machine and monte carlo simulation. *Structural Safety*, 67:85–95, 7 2017. ISSN 0167-4730. doi: 10.1016/J.STRUSAFE.2017.04.006.
- [24] Tong Zhou, Yongbo Peng, and Jie Li. An efficient reliability method combining adaptive global metamodel and probability density evolution method. *Mechanical Systems and Signal Processing*, 131:592–616, 9 2019. ISSN 0888-3270. doi: 10.1016/J.YMSSP.2019.06.009.
- [25] Katerina Konakli and Bruno Sudret. Global sensitivity analysis using low-rank tensor approximations. *Reliability Engineering & System Safety*, 156:64–83, 12 2016. ISSN 0951-8320. doi: 10.1016/J.RESS.2016.07.012.

- [26] Fabian Keller, E Mazza, E Hosseini, and S Marelli. Surrogate modelling for multiscale thermal simulation of powder-bed additive manufacturing. 2020.
- [27] Felix W Baumann, André Sekulla, Michael Hassler, Benjamin Himpel, and Markus Pfeil. Trends of machine learning in additive manufacturing. *Int. J. Rapid Manufacturing*, 7, 2018.
- [28] P O Box, Laurens Van Der Maaten, Eric Postma, and Jaap Van Den Herik. Dimensionality reduction: A comparative review. *Tilburg centre for Creative Computing*, 2009. URL <http://www.uvt.nl/ticc>.
- [29] Prof. Andreas Krause. Lecture notes in "introduction to machine learning", March 2020.
- [30] Emily Gorcenski. Conference slides of "polynomial chaos: A technique for modeling uncertainty", July 2017.
- [31] Stefano Marelli and Bruno Sudret. Uqlab: A framework for uncertainty quantification in matlab. *Vulnerability, Uncertainty, and Risk*, 2014. URL https://www.academia.edu/27090215/UQLab_A_Framework_for_Uncertainty_Quantification_in_Matlab.
- [32] D. O. (Donald Olding) Hebb. The organization of behavior: a neuropsychological theory. 1949. URL https://books.google.com/books/about/The_Organization_of_Behavior.html?hl=de&id=uyV5AgAAQBAJ.
- [33] F Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain 1. *Psychological Review*, 65:19–27, 1958.
- [34] Erkki Oja. Simplified neuron model as a principal component analyzer. *Journal of Mathematical Biology* 1982 15:3, 15:267–273, 1982. ISSN 1432-1416. doi: 10.1007/BF00275687. URL <https://link.springer.com/article/10.1007/BF00275687>.
- [35] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning representations by back-propagating errors. *Nature*, 323:533–536, 10 1986.
- [36] Nanzhe Wang, Haibin Chang, and Dongxiao Zhang. Theory-guided auto-encoder for surrogate construction and inverse modeling. 11 2020. URL <http://arxiv.org/abs/2011.08618>.
- [37] Vahid Nasir and Farrokh Sassani. A review on deep learning in machining and tool monitoring: methods, opportunities, and challenges. *International Journal of Advanced Manufacturing Technology*, 115:2683–2709, 8 2021. doi: 10.1007/S00170-021-07325-7.

- [38] Siddhartha Mishra and Roberto Molinaro. Estimates on the generalization error of physics-informed neural networks for approximating a class of inverse problems for pdes. *IMA Journal of Numerical Analysis*, 00:1–42, 6 2021. doi: 10.1093/IMANUM/DRAB032. URL <https://academic.oup.com/imajna/advance-article/doi/10.1093/imanum/drab032/6297946>.
- [39] John Goldak, Aditya Chakravarti, and Malcolm Bibby. A new finite element model for welding heat sources. *Metallurgical Transactions B*, pages 299–305, 1984.
- [40] Z. Samad, N. M. Nor, and E. R.I. Fauzi. Thermo-mechanical simulation of temperature distribution and prediction of heat-affected zone size in mig welding process on aluminium alloy en aw 6082-t6. *IOP Conference Series: Materials Science and Engineering*, 530, 2019. doi: 10.1088/1757-899X/530/1/012016.
- [41] Pedro Domingos. A few useful things to know about machine learning. *Communications of the ACM*, 55:78–87, 2012. doi: 10.1145/2347736.2347755.
- [42] Enzo Tartaglione, Carlo Alberto Barbano, Claudio Berzovini, Marco Calandri, and Marco Grangetto. Unveiling covid-19 from chest x-ray with deep learning: A hurdles race with small data. *International Journal of Environmental Research and Public Health*, 17:1–17, 9 2020. doi: 10.3390/IJERPH17186933.
- [43] Siddhartha Mishra and Roberto Molinaro. Physics informed neural networks for simulating radiative transfer. *Journal of Quantitative Spectroscopy and Radiative Transfer*, 270:107705, 8 2021. ISSN 0022-4073. doi: 10.1016/J.JQSRT.2021.107705.
- [44] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. 2015.
- [45] Timo Stöttner. Why data should be normalized before training a neural network. *Towards Data Science*, 2019. URL <https://towardsdatascience.com/why-data-should-be-normalized-before-training-a-neural-network-c626b7f66c7d>.
- [46] Wojciech Matusik. M.i.t. class. *Computer Graphics*, 837.
- [47] Daniel Rosenthal. Mathematical theory of heat distribution during welding and cutting. *Welding Journal*, 20:220–234, 1941.
- [48] Kjetil O. Lye, Siddhartha Mishra, and Deep Ray. Deep learning observables in computational fluid dynamics. *Journal of Computational Physics*, 410:109339, 2020. ISSN 0021-9991. doi: <https://doi.org/10.1016/j.jcp.2020.109339>. URL <https://www.sciencedirect.com/science/article/pii/S0021999120301133>.

- [49] Attila Kakay, Elmar Westphal, and Riccardo Hertel. Speedup of fem micromagnetic simulations with graphical processing units. *IEEE Transactions on Magnetics*, 46: 2303–2306, 6 2010. doi: 10.1109/TMAG.2010.2048016.
- [50] Chao Wang, Bin Zhu, Liang Wang, Yi Lin Wang, and Yi Sheng Zhang. Gpu accelerated finite element simulation for ultra-high strength steel quenching. *Advanced Materials Research*, 842:337–340, 2014. ISSN 1662-8985. doi: 10.4028/WWW.SCIENTIFIC.NET/AMR.842.337. URL <https://www.scientific.net/AMR.842.337>.
- [51] Mahyar Khorasani, Amir Hossein Ghasemi, Martin Leary, William O’Neil, Ian Gibson, Laura Cordova, and Bernard Rolfe. Numerical and analytical investigation on meltpool temperature of laser-based powder bed fusion of in718. *International Journal of Heat and Mass Transfer*, 177:121477, 10 2021. ISSN 0017-9310. doi: 10.1016/J.IJHEATMASSTRANSFER.2021.121477.
- [52] Shahriar Imani Shahabad, Zhidong Zhang, Ali Keshavarzkermani, Usman Ali, Yahya Mahmoodkhani, Reza Esmaeilizadeh, Ali Bonakdar, and Ehsan Toyserkani. Heat source model calibration for thermal analysis of laser powder-bed fusion. *The International Journal of Advanced Manufacturing Technology 2020 106:7*, 106:3367–3379, 1 2020. ISSN 1433-3015. doi: 10.1007/S00170-019-04908-3. URL <https://link.springer.com/article/10.1007/s00170-019-04908-3>.