INVESTIGATION OF PROCESS-STRUCTURE RELATIONSHIP FOR ADDITIVE MANUFACTURING WITH MULTIPHYSICS SIMULATION AND PHYSICS-CONSTRAINED MACHINE LEARNING

A Dissertation
Presented to
The Academic Faculty

By

Dehao Liu

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in Mechanical Engineering in the
George W. Woodruff School of Mechanical Engineering

Georgia Institute of Technology
August 2021

INVESTIGATION OF PROCESS-STRUCTURE RELATIONSHIP FOR ADDITIVE MANUFACTURING WITH MULTIPHYSICS SIMULATION AND PHYSICS-CONSTRAINED MACHINE LEARNING

Approved by:

Dr. Yan Wang, Advisor
George W. Woodruff School of Mechanical Engineering
Georgia Institute of Technology

Dr. Tuo Zhao
H. Milton Stewart School of Industrial and Systems Engineering, School of Computational Science and Engineering
Georgia Institute of Technology

Dr. David L. McDowell
George W. Woodruff School of Mechanical Engineering, School of Materials Science and Engineering
Georgia Institute of Technology

Dr. Sudarsanam Suresh Babu
Energy and Environmental Sciences Division,
Oak Ridge National Laboratory
Department of Mechanical, Aerospace and Biomedical Engineering, Department of Materials Science and Engineering
University of Tennessee

Dr. Shreyes N. Melkote
George W. Woodruff School of Mechanical Engineering
Georgia Institute of Technology

Date Approved: July 2, 2021

ACKNOWLEDGEMENTS

I would like to express my deep and sincere gratitude to my advisor, Prof. Yan Wang for his continuous support and guidance of my Ph.D. study and related research. Without his patience, motivation, vision, and immense knowledge, it would not have been possible to complete my dissertation. He helps me to improve my research abilities and cultivates my research taste. He encourages me to solve hard problems rather than the low-hanging fruit. He teaches me how to make better choices in a long-term view by sharing his own experiences and thoughts.

I would like to thank the rest of my committee members: Prof. David L. McDowell, Prof. Shreyes N. Melkote, Prof. Tuo Zhao, and Prof. Sudarsanam Suresh Babu, for their valuable comments, but also for the hard questions which help me to widen my research from various aspects.

I would like to thank each member in our MSSE research group, Dr. Anh Tran, Dr. Yanglong Lu, Dr. Longchao Cao, Dr. Lijuan He, Dr. Ji-Hyeon Song, Dr. Jesse Sestito, Dr. Qi Zhou, Dr. Jie Liu, Michael Kempner, Zhehao Zhang, and Luka Malashkhia for their encouragement, feedback, and assistance during my Ph.D. study.

My sincere thanks also go to Dr. Larry Aagesen at Idaho National Lab and Dr. Elena Arvanitis at Siemens, who provided me an opportunity to join their teams as a graduate intern.

I would like to acknowledge my loving and supportive wife, Ziyuan Zhong, who has stood by me since the time we have known each other. I would like to thank my family: my parents and my siblings for supporting me spiritually throughout my life.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SUMMARY

Metal additive manufacturing (AM) is a group of processes by which metal parts are built layer by layer from powder or wire feedstock with high-energy laser or electron beams. The most well-known metal AM processes include selective laser melting, electron beam melting, and direct energy deposition. Metal AM can significantly improve the manufacturability of products with complex geometries and heterogeneous materials. It has the potential to be widely applied in various industries including automotive, aerospace, biomedical, energy, and other high-value low-volume manufacturing environments. However, the lack of complete and reliable process-structure-property (P-S-P) relationships for metal AM is still the bottleneck to produce defect-free, structurally sound, and reliable AM parts. There are several technical challenges in establishing the P-S-P relationships for process design and optimization. First, there is a lack of fundamental understanding of the rapid solidification process during which microstructures are formed and the properties of solid parts are determined. Second, the curse of dimensionality in the process and structure design space leads to the lack of data to construct reliable P-S-P relationships.

Simulation becomes an important tool to enable us to understand rapid solidification given the limitations of experimental techniques for in-situ measurement. In this research, a mesoscale multiphysics simulation model, called phase-field and thermal lattice Boltzmann method (PF-TLBM), is developed with simultaneous considerations of heterogeneous nucleation, solute transport, heat transfer, and phase transition. The simulation can reveal the complex dynamics of rapid solidification in the melt pool, such

as the effects of latent heat and cooling rate on dendritic morphology and solute distribution. The microstructure evolution in the complex heating and cooling environment in the layer-by-layer AM process is simulated with the PF-TLBM model.

To meet the lack-of-data challenge in constructing P-S-P relationships, a new scheme of multi-fidelity physics-constrained neural network (MF-PCNN) is developed to improve the efficiency of training in neural networks by reducing the required amount of training data and incorporating physical knowledge as constraints. Neural networks with two levels of fidelities are combined to improve prediction accuracy. Low-fidelity networks predict the general trend, whereas high-fidelity networks model local details and fluctuations. The developed MF-PCNN is applied to predict phase transition and dendritic growth. A new physics-constrained neural network with the minimax architecture (PCNN-MM) is also developed, where the training of PCNN-MM is formulated as a minimax problem. A novel training algorithm called Dual-Dimer method is developed to search high-order saddle points. The developed PCNN-MM is also extended to solve multiphysics problems. A new sequential training scheme is developed for PCNN-MMs to ensure the convergence in solving multiphysics problems. A new Dual-Dimer with compressive sampling (DD-CS) algorithm is also developed to alleviate the curse of dimensionality in searching high-order saddle points during the training. A surrogate model of process-structure relationship for AM is constructed based on the PF-TLBM and PCNN-MM. Based on the surrogate model, multi-objective Bayesian optimization is utilized to search the optimal initial temperature and cooling rate to obtain the desired dendritic area and microsegregation level. The developed PF-TLBM and PCNN-MM provide a systematic and efficient approach to construct P-S-P relationships for AM process design.

CHAPTER 1. INTRODUCTION

Metal additive manufacturing (AM) is a group of processes by which metal parts are built layer by layer from powder or wire feedstock with high-energy laser or electron beams. The most well-known metal AM processes include selective laser melting (SLM), electron beam melting (EBM), and direct energy deposition (DED). Metal AM can significantly improve the manufacturability of products with complex geometries and heterogeneous materials. It can be used to produce complex or customized parts, such as meta-materials [1], without the need for expensive tooling. It also allows us to control the chemical composition of a part locally by adjusting the mixture of different powders at each layer or even at individual voxel level [2]. Furthermore, metal AM enables tailoring microstructures and thereby mechanical properties of a part by the local control of crystallographic grain orientation [3]. It has the potential to be widely applied in various industries including automotive, aerospace, biomedical, energy, and other high-value low-volume manufacturing environments. In automotive, metal AM has been used for prototyping, rapid fabrication, and repair of industrial hardware such as dies, punches, and customized tooling. In aerospace, lighter and more durable fuel nozzles are directly produced by metal AM without assembly of multiple parts with significant cost savings. In biomedical applications, metal AM can significantly improve the integration and biocompatibility of medical implants. In the energy industry, mixing and swirling burner tips made by metal AM can save energy, extend component lifetime, and reduce system repair frequencies and downtime.

The microstructures of materials are the direct results of processing conditions, whereas the microstructures determine the material properties. The process-structure-property (P-S-P) relationships constitute the guiding principles of materials design. The essential task of material and process design is to establish the P-S-P relationships. The P-S-P relationships in metal AM processes are complex since many process parameters can be controlled locally at the fine-grained level. Experimental studies have shown that processing conditions such as the power of heat source, scanning speed, scanning patterns [3,4], powder layer thickness, hatch spacing, building directions [5], shielding gas [6], external electromagnetic field [7], post processing [8], in-situ feedstock materials blending [9,10], nanoscale nucleants [11], microalloying with grain refiners [12], and others have major influences on the final solidified microstructures. Major microstructural characteristics such as internal defects (e.g., gas pore, lack of fusion), surface roughness, grain size distribution, grain shapes (e.g., cellular, columnar, equiaxed), phase distribution, composition distribution, precipitation, and others affect the mechanical, thermal, and electrochemical properties of final parts [13–16]. The lack of complete and reliable P-S-P relationships for metal AM is still the bottleneck to produce defect-free, structurally sound, and reliable AM parts [14].

1.1 Challenges of AM Process Design and Optimization

There are two major technical challenges in establishing P-S-P relationships for AM process design and optimization. The first challenge is the lack of fundamental understanding of the rapid solidification process during which microstructures are formed and the properties of solid parts are determined. During the complex process of rapid solidification, interactions between solute transport, heat transfer, fluid dynamics, and

phase transition have significant effects on the formation of the microstructure. The final solidified microstructure determines the mechanical strength, thermal conductivity, corrosion resistance, and other properties of the final parts. Currently, the capabilities of in-situ observation for rapid solidification at the nano- and micro-scales are very limited. Compared to experimental studies, simulation is more cost-effective to reveal the cause-effect relationships. Therefore, multiscale multiphysics simulation models are needed to enable a deeper understanding of rapid solidification. Traditional macroscale or mesoscale simulations [17,18] can predict multiphysics of heat transfer and melt flow [19–24], as well as powder dynamics [25,26], residual stress and distortion [27–29]. However, they do not provide the details of microstructure formation. Although empirical and semi-empirical approaches [30–32] have been developed to predict texture evolutions very efficiently, they do not provide fine-grained details of dendritic growth and composition distribution, which are important for property predictions. Therefore, mesoscale simulations such as phase-field method and cellular automaton are needed to predict microstructure evolution during rapid solidification.

The second challenge is the curse of dimensionality in the process and structure design spaces, which leads to the lack of data to construct reliable P-S-P relationships. To systematically optimize process parameters, a comprehensive P-S-P relationship is necessary. Given a large number of design parameters for processes and structures therefore high complexity of P-S-P relationships, surrogates of P-S-P relationships need to be constructed to systematically guide the design optimization. Constructing surrogates of high-dimensional P-S-P relationships with data-driven approaches such as machine learning (ML) requires a significant amount of training data, which are obtained from

experiments or simulations. As ML tools need to capture more detailed patterns or sensitive features, more complex modeling structures need to be introduced with more parameters and degrees of freedom. As a result, training algorithms need to explore and exploit in a very high-dimensional parameter space to search for the optimal parameters. When the dimension increases, the volume of the parameter space increases exponentially, so does the required amount of training data to cover the space, to ensure the convergence of training. Only relying on expensive simulations or physical experiments to generate a large volume of training data will be very expensive. When the size of the training data set is small, overfitting can occur. That is, the training results in a spurious relationship that looks deceptively good but has low generality outside the labeled data range. Therefore, new data-driven machine learning frameworks are needed to construct reliable P-S-P surrogates even with a small amount of training data.

1.2 An Overview of the Developed Process Design Framework

The objective of this research is to investigate the feasibility of a hybrid physics-based data-driven process optimization approach to establish reliable surrogates of process-structure-property relationships for metal additive manufacturing aided by mesoscale multiphysics simulation and physics-constrained machine learning.

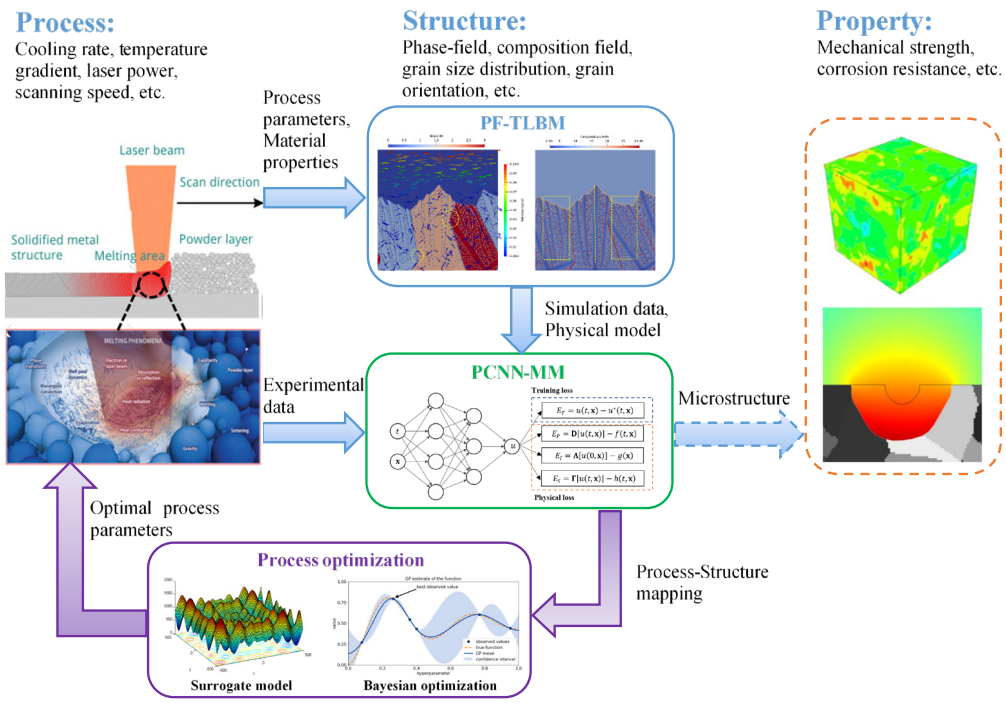


Figure 1.1. Process optimization based on multiphysics simulation and physics-constrained machine learning.

The proposed generic process design framework is shown in Figure 1.1. Mesoscale multiphysics simulation models help reveal the details of rapid solidification with predictions of grain texture and composition distributions in microstructures. A phase-field and thermal lattice Boltzmann method (PF-TLBM) is developed with simultaneous considerations of heterogeneous nucleation, solute transport, heat transfer, fluid dynamics, and phase transition. To reduce the computational costs in establishing the process-structure relationships, a physics-constrained machine learning approach is taken to construct the surrogate to predict dendritic morphology and alloy compositions from process parameters. A new scheme of multi-fidelity physics-constrained neural network (MF-PCNN) is developed to improve the efficiency of training in neural networks by reducing the required amount of training data and incorporating physical knowledge as

constraints. A new physics-constrained neural network with the minimax architecture (PCNN-MM) is also developed, where the training of PCNN-MM is formulated as a minimax problem. The training can be based on either simulation or experimental data. After training, this new machine learning tool can predict microstructures from process parameters efficiently as a surrogate without relying on expensive simulations or experiments. The process-structure surrogate relationship can be applied to perform process optimization. Global optimization tools such as Bayesian optimization can be used to search the optimal process parameters so that the desired microstructures such as dendritic area and composition distribution with target material properties can be achieved. Continuum simulations such as finite element analysis (FEA) can be used to predict properties (e.g. mechanical strength, corrosion resistance) from microstructures. The predicted structure-property mappings can also be utilized in the optimization.

The process design framework in Figure 1.1 is generic. In this dissertation, we only focus on its three core components, which are the proposed phase-field and thermal lattice Boltzmann method for multiphysics simulation, physics-constrained neural network with minimax architecture, and Bayesian optimization for process design.

1.2.1 Phase-Field and Thermal Lattice Boltzmann Method

To better understand the rapid solidification during AM processes, we developed a mesoscale multiphysics simulation model, called phase-field and thermal lattice Boltzmann method (PF-TLBM), with simultaneous considerations of heterogeneous nucleation, solute transport, heat transfer, fluid dynamics, and phase transition. In this model, the phase-field method simulates the dendrite growth of alloys, whereas the thermal lattice Boltzmann method models heat transfer and fluid flow. The phase-field method and

the thermal lattice Boltzmann method are tightly coupled. A nucleation model is introduced in the PF-TLBM model to simulate heterogeneous nucleation at the boundary of the melt pool in SLM. The PF-TLBM model is also extended to predict the multi-layer epitaxial grain growth in the complex heating and cooling environment in SLM. A new marching cell simulation scheme is developed to reduce the computational cost.

1.2.2 Multi-Fidelity Physics-Constrained Neural Network

To meet the lack-of-data challenge in constructing P-S-P relationships, a new scheme of multi-fidelity physics-constrained neural network (MF-PCNN) is developed to improve the efficiency of training in neural networks by reducing the required amount of training data and incorporating physical knowledge as constraints. Neural networks with two levels of fidelities are combined to improve prediction accuracy. A low-cost low-fidelity physics-constrained neural network (LF-PCNN) is used as the baseline model, whereas a limited amount of data from a high-fidelity physics-constrained neural network (HF-PCNN) is used to train a second neural network to predict the difference between the two models.

1.2.3 Physics-Constrained Neural Network with the Minimax Architecture

To systematically adjust the weights of different losses for data and physical constraints in a physics-constrained neural network (PCNN), a new physics-constrained neural network with the minimax architecture (PCNN-MM) is developed. The training of the PCNN-MM is searching the high-order saddle points of the objective or loss function. A novel saddle point search algorithm called the Dual-Dimer algorithm is developed, where only first derivatives need to be calculated. It is demonstrated that the Dual-Dimer method is computationally more efficient than the gradient descent ascent (GDA) method

for nonconvex-nonconcave functions and provides additional eigenvalue information to verify the search results. To further reduce the computational cost, a multi-fidelity PCNN-MM (MF-PCNN-MM) is developed to integrate the low-fidelity (LF) and high-fidelity (HF) data. In the MF-PCNN-MM, LF and HF data are integrated together to make the tradeoff between efficiency and accuracy for multi-fidelity metamodeling. The MF-PCNN-MM is composed of two artificial neural networks (ANNs). The first ANN is used to approximate the LF data, whereas the second ANN is adopted to approximate the mapping function between the LF and HF data.

To accelerate the convergence of PCNN-MMs during training in solving multiphysics problems, a new sequential training scheme is developed. For each of the coupled physical fields, one neural network is constructed. The trainings of them are done sequentially. The sequential training scheme helps improve the convergence speed and the prediction accuracy from the original PCNN-MM. A new saddle point search algorithm called Dual-Dimer with compressive sampling (DD-CS) is developed with iterations of three stages in the DD-CS algorithm. In the first stage, the PCNN-MM is trained in the complete parameter space with all weights of neurons. In the second stage, the PCNN-MM is trained in the reduced parameter space by taking advantage of sparsity. In the third stage, the compressive sampling approach is used to recover the original weights of the neural network.

1.2.4 Process Design Framework

A new generic process design framework is developed. The dendritic growth under different processing parameters (initial temperature and cooling rate) are simulated by PF-TLBM. The simulation outputs include phase field, temperature field, and composition

field. Partial simulation results serve as the training data for the training of the PCNN-MM. Once the training of PCNN-MM is completed, the predicted microstructures from the PCNN-MM are characterized with dendritic area and microsegregation. In this way, a surrogate model of the process-structure relationship is built for additive manufacturing. Based on the surrogate model, multi-objective Bayesian optimization (BO) is utilized to search the optimal initial temperature and cooling rate to obtain the desired dendritic area and microsegregation level.

1.3 Technical Contributions

The novel contributions of the dissertation are highlighted as follows.

- A new mesoscale multiphysics simulation model called PF-TLBM is developed to predict microstructure evolution of alloys in rapid solidification by concurrently coupling heterogeneous nucleation, solute transport, heat transfer, latent heat, fluid dynamics, and phase transition. A new method is developed to compute heat fluxes for a two-dimensional small melt pool to approximate the actual non-isothermal temperature field in metal AM processes. To reduce the computational cost, a new marching cell simulation scheme is introduced in the PF-TLBM model to predict the multi-layer epitaxial grain growth in metal AM processes. The developed PF-TLBM model enables a deeper understanding of the rapid solidification process in metal AM and makes it possible to simulate the microstructure evolution at the scale of the whole building part.
- A novel physics-based machine learning model called MF-PCNN is proposed to reduce the required amount of training data, and multi-fidelity networks

are constructed to improve the training efficiency. A new PCNN-MM is proposed to adjust the weights of different losses systematically. A novel and general saddle point search algorithm called Dual-Dimer method is developed to train the PCNN-MM. To reduce the computational cost, a new MF-PCNN-MM is developed to integrate the LF and HF data for the concurrent training of LF and HF neural networks. A new sequential training scheme is developed to improve the convergence and prediction accuracy of PCNN-MMs in solving multiphysics problems. A new saddle point search algorithm called DD-CS is also developed to alleviate the curse of dimensionality in searching high-order saddle points during the training. The developed physics-constrained machine learning models and training algorithms are promising approaches to alleviate the curse of dimensionality in general ML applications.

- A new generic process design framework is developed for process optimization. A surrogate model of process-structure relationships is constructed based on the PCNN-MM trained by the simulation data from PF-TLBM. The constructed surrogate model is used in multi-objective BO to search the optimal process parameters so that the desired dendritic area and composition distribution can be achieved.

1.4 Dissertation Organization

In the remainder of this dissertation, CHAPTER 2 provides some background knowledge related to the topics of phase-field method, nucleation models, multiphysics simulation of rapid solidification coupled with phase field or cellular automaton

simulations, methods of incorporating prior knowledge in machine learning, saddle point search methods, and physics-constrained neural networks for solving multiphysics problems. The PF-TLBM model is developed to simulate rapid solidification of Ti-6Al-4V alloy in CHAPTER 3. In CHAPTER 4, the PF-TLBM model is extended to simulate the nucleation and dendritic growth of AlSi10Mg alloy in single scanning pass and multiple scanning passes. The MF-PCNN is developed and demonstrated by three examples of materials modeling in CHAPTER 5. The formulation and demonstration of the PCNN-MM and the MF-PCNN-MM are described in CHAPTER 6. The formulation, convergence analysis, and evaluation of the Dual-Dimer algorithm are also included. In CHAPTER 7, the PCNN-MM is extended to solve multiphysics problems. A generic process design framework is developed for rapid solidification process optimization for metal AM processes in CHAPTER 8.

CHAPTER 2. BACKGROUND

In this chapter, the background of phase-field method and nucleation models are introduced. The literature review on multiphysics simulation of rapid solidification coupled with phase-field method or cellular automaton is included. Existing methods of incorporating prior knowledge in machine learning are described. The existing saddle point search methods are reviewed. The background of physics-constrained neural networks for solving multiphysics problems is also given.

2.1 Phase-Field Method

Various simulation models have been developed to understand and predict microstructure evolution during rapid solidification. These include sharp interface tracking, enthalpy, level-set, cellular automaton (CA), and phase-field method (PFM) [33]. Particularly, PFM and CA are the most used methods. PFM simulates a much longer time scale than what molecular dynamics is able to, and provides more physical details of material phases than what kinetic Monte Carlo simulation can. CA [34–39] is computationally much more efficient than PFM since it explicitly tracks the liquid-solid front so that coarser mesh can be used. CA can be applied to simulate much larger systems than PFM. PFM method [40,41] implicitly models the interface and can reveal more details of interface geometry and side branch emission. The major technical issue of CA is its predictions of anisotropic grain structures are sensitive to the choice of mesh shapes and sizes due to coarse representation [42]. Improvements of CA have been made to calculate the local interface curvature and velocity from solid fractions of cells [43–49] in order to mitigate the mesh-dependent anisotropy issue. Yet these improvements require much

denser meshes than regular CA. As a result, CA's major advantage of computational efficiency quickly diminishes and computational cost can become comparable to PFM [50].

In PFM, a continuous variable named phase field indicates the solid phase fraction in the simulated domain. The microstructure evolution is modeled with a partial differential equation driven by the interfacial and chemical free energies. In parallel, the composition field is also evolved as a diffusion process coupled with the phase field. PFM has been widely used to simulate dendritic growth in the solidification processes of casting and welding [51–53]. Comprehensive models such as multi-component multiphase field model coupled with thermodynamic databases can simulate alloy solidification accurately and reveal details [54]. Recently, PFM was employed to simulate the grain growth of Ti-6Al-4V alloy in the EBM process [55,56]. It has been revealed that increases in temperature gradient and beam scanning speed reduce the primary arm spacing of columnar dendrites. However, in the studies with PFM only, the multiphysics effects of melt flow and local temperature due to latent heat were not considered. Those effects need to be studied to better understand rapid solidification.

2.2 Nucleation Models

Nucleation affects the accuracy of simulated microstructures in metal AM, but it has only been considered in few studies of PFM simulation. Shimono et al. [57] simulated the columnar-to-equiaxed (CET) transition of Ti-6Al-4V alloy during the AM process by coupling PFM with the calculation of phase diagrams (CALPHAD). A continuous Gaussian nucleation distribution was used to describe the grain density increase with the increase in undercooling. The empirical nucleation parameters, such as maximum

nucleation density and mean undercooling, were calibrated based on experimental results. However, this empirical model missed some important physics of nucleation compared to classical nucleation theory (CNT). Gránásy et al. [58,59] described two methods to include homogeneous nucleation into PFM simulations. In the first method, Langevin noise terms were introduced in PFM as a nucleation force. In the second one, the nucleation energy barrier was determined by solving the Euler–Lagrange equations of the phase-field and composition field. PFM was also used to determine the nucleation energy barrier for heterogeneous nucleation, where appropriate boundary conditions were introduced at the foreign wall to realize the required contact angle [60]. Pusztai et al. [61] introduced Langevin noise terms in PFM to simulate homogeneous and heterogeneous nucleation in polycrystalline. However, by introducing Langevin noises, nucleation could occur anywhere in the simulation domain rather than the solid-liquid interface because of the nature of stochastic partial differential equations. The model works well for large melt pools such as in casting, but is not accurate in SLM with small melt pools.

In powder-based SLM, the size of the melt pool is usually less than 100 μm. Furthermore, it is known that nucleation and growth occur at different time scales, the observation of nucleation would require an impractically large number of sample frequencies and integration cycles. Therefore, Simmons et al. [62] replaced the Langevin noise terms in PFM with a Poisson seeding algorithm, where viable nuclei were introduced at a time-dependent nucleation rate. However, the developed model is used for homogeneous nucleation rather than heterogeneous nucleation. In the work of Li et al. [63], the nucleation kinetics of binary melts is calculated based on CNT. The model was originally used to simulate polycrystalline solidification of NiCu alloy in casting, where

the clear CET transition was clearly shown. However, the heterogeneous nucleation in the model occurred in the melt pool rather than the boundary of the melt pool. In SLM, heterogeneous nucleation tends to occur at the solid-liquid interface at the bottom of the small melt pool as experimentally observed. An accurate and reliable nucleation model is necessary to simulate microstructure evolution within the melt pool in metal AM processes.

2.3 Multiphysics Simulation of Rapid Solidification Coupled with PFM or CA

The rapid solidification in AM is a highly complex process, where it is not appropriate to make equilibrium assumptions about the thermal field and melt flow. A multiphysics simulation approach is necessary to understand the coupling effects among temperature, velocity, composition, and phase fields. Fluid dynamics can be simulated with finite volume method (FVM), lattice Boltzmann method (LBM), and finite element method (FEM), whereas heat transfer is usually simulated with FEM. Integrated models include CA-FVM [64], CA-LBM [65–68], PFM-FEA [69–73], and PFM-FVM [74] have been developed. However, in those multiphysics models, simple one-way coupling was adopted, which cannot reveal the complex coupling effects between different physics. Some efforts have been made to combine PFM and LBM to simulate the dendritic growth in the solidification of pure metals and alloys [75–79], allowing for the interplay between grain growth and melt flow. However, in these integrated models, either the isothermal condition or a one-dimensional temperature field was assumed [80], which still oversimplifies the physical processes. The temperature field during rapid solidification can be much more complex than that of solidification under the equilibrium thermal condition because melt flow and the release of latent heat will constantly change the temperature distribution. Therefore, the effects of latent heat and melt flow on phase transition should be

simultaneously considered in tightly coupled multiphysics modeling of solidification for more accurate predictions.

In this dissertation, a new mesoscale multiphysics simulation approach is proposed, which couples PFM and thermal lattice Boltzmann method (TLBM) to simulate microstructure evolution during the rapid solidification process. Compared to traditional finite volume methods to simulate fluid flow, the lattice Boltzmann method (LBM) has computational advantages for systems with complex boundaries [81–83]. LBM is capable of simulating single-phase and multiphase flow with complex boundary conditions and multiphase interfaces. To incorporate thermal effects into fluid dynamics, the thermal lattice Boltzmann method (TLBM) [84–90] has also been developed. Unlike LBM, which uses a single particle distribution function for fluids, TLBM uses two distinct particle distribution functions for fluid dynamics and heat transfer respectively. TLBM has been recently adopted to simulate the evolution of temperature and velocity fields in the EBM process [91]. However, the simulation using TLBM alone lacks fine-grained phase information, because it cannot simulate the evolution of dendrite structures. Our new integrated phase-field and thermal lattice Boltzmann method (PFM-TLBM) provides multiple physics information of phase field, composition, fluid velocity, and temperature simultaneously for the first time. Besides, a nucleation model is introduced into the PF-TLBM for simulating the microstructure evolution of alloys in SLM.

2.4 Methods of Incorporating Prior Knowledge in Machine Learning

Our goal of employing ML tools is to help establish P-S-P relationships. In various engineering and scientific applications, the cost of obtaining a large amount of data from experiments or high-fidelity simulations can be prohibitive. Data sparsity is the bottleneck

for applying the state-of-the-art ML techniques in the domains of engineering, where establishing high-dimensional P-S-P relationships for either product or process design is the essential task. Training ML tools based on prior knowledge of physics can help navigate the high-dimensional parameter space with a smaller amount of training data [92].

Incorporating physical meanings and physical knowledge in artificial neural networks (ANNs) has been studied from two major perspectives [93,94]. The first approach is to customize ANNs and incorporate physical meanings in the architecture [95,96]. For instance, prior knowledge can be applied as preprocessing tools to filter training data [97,98], or embedded as some analytical input-output functions in additional layers of ANNs [99,100], to improve the training efficiency. Prior knowledge can also be expressed as rules and interpreted with weights and basis functions in the ANN architecture, which could be further refined using training data [101–104]. The major challenge of incorporating physical meanings into the ANN architecture is the complexity of customized networks.

The second approach to incorporate physical knowledge is treating prior knowledge as constraints so that it can guide the training process. For instance, prior knowledge can be embedded in ANNs as architectural constraints and connection weight constraints to improve training efficiency [105]. In addition to functional values, the information of derivatives has also been incorporated as prior knowledge for support vector regression [106]. ANNs have been used to approximate the solutions of partial differential equations (PDEs). The prior knowledge of model forms and boundary conditions can be embedded as regularization terms into the objective function during the training process [107–111]. A regularization parameter has been introduced to control the trade-off between data fitting

and knowledge-based regularization [112]. Instead of regularization, information about boundary conditions can be explicitly used as equality constraints between the weights in ANNs such that a constrained backpropagation training can be taken [113–115]. The effectiveness of regularization during the ML training has been demonstrated in the above work. However, the training efficiency is still limited in high-dimensional problems, where the sampling of PDE solutions can be costly. Furthermore, the choices of weights to incorporate constraints as regularization in the cost functions are largely empirical and require case-by-case sensitivity studies.

Recently, physics-constrained machine learning emerged as a promising approach to alleviate the issue of data sparsity. In this approach, prior knowledge in science and engineering is incorporated as constraints to guide the training of ML models. In the training of physics-constrained neural networks (PCNNs) [112,116–121], physical models serve as the constraints and regularize the training loss. It has been shown that the required amount of training data can be reduced by adding physical constraints as the regularization terms. However, the training efficiency is sensitively dependent on the weights associated with the different losses with respect to data and physical constraints. In existing PCNNs, the weights were either fixed or adjusted empirically. Systematic approaches for weight adjustment are needed.

Although PCNNs can significantly reduce the required amount of training data, a large set of high-fidelity training data is still needed for PCNNs to learn the unknown weights of neural networks. On the other hand, the concept of multi-fidelity has been widely explored in metamodeling, particularly Gaussian process (GP) or co-kriging. In the scheme of multi-fidelity metamodeling, it is assumed that accurate high-fidelity (HF) data

are sparse and expensive, whereas less accurate low-fidelity (LF) data are sufficient and cheaper. The tradeoff between efficiency and accuracy for metamodeling can be made by integrating LF and HF data [122–124]. Multi-fidelity Gaussian process modeling has been widely adopted to search optimal design choices for various engineering problems [125–128]. To further reduce the cost of data acquisition, multi-fidelity physics-informed neural networks [129] were developed to solve PDEs. However, the weights of different losses are fixed in this work.

Different from the above work, a multi-fidelity physics-constrained neural network (MF-PCNN) is proposed in this dissertation to further enhance the efficiency of training by considering the different costs of low- and high-fidelity samples. In addition, a new physics-constrained neural network with minimax architecture (PCNN-MM) is proposed, where the training of the PCNN is formulated as a minimax problem. In this way, the robustness of training is improved by systematically and adaptively adjusting the weights during the training. Furthermore, a multi-fidelity physics-constrained neural network with minimax architecture (MF-PCNN-MM) is also proposed. In the MF-PCNN-MM, LF and HF data are integrated together to make the tradeoff between efficiency and accuracy for multi-fidelity metamodeling.

2.5 Saddle Point Search Methods

The training of our new PCNN-MM is searching high-order saddle points. Various saddle point search algorithms have been developed [130,131]. These include surface walking algorithm [132], DHS method [133], partitioned rational function optimization method [134], activation-relaxation technique [135], dimer method [136–138], nudged elastic band [139,140], curve swarm method [141,142], and Kriging metamodels [143–

145]. However, these methods can only identify first-order saddle points instead of high-order ones.

The well-known gradient descent ascent (GDA) algorithm has been widely used to search saddle points. In the past decade, the GDA algorithm has been applied to solve the nonconvex-nonconcave minimax problems, which arise from game theory [146], generative adversarial networks [147], and robust optimization [148]. However, it has difficulty converging to the saddle points of the nonconvex-nonconcave functions [149]. Some GDA extensions are also available. For instance, a proximally guided stochastic subgradient method [150] was proposed to solve a class of weakly-convex-concave minimax problems. A multi-step GDA algorithm [151] and a proximal dual implicit accelerated gradient algorithm [152] were developed to solve the nonconvex but concave minimax problems. Two-time-scale GDA [153] was shown to converge to stationary local Nash equilibria under certain strong conditions. Symplectic gradient adjustment (SGA) algorithm [154] was proposed to search stable fixed points in general games, including potential games and Hamiltonian games. Hessian-based algorithms [155] were developed to search local saddle points in the nonconvex-nonconcave settings. However, the computation of the Hessian matrix is expensive for high-dimensional problems. A novel saddle point search method called Dual-Dimer is developed in this dissertation to search the high-order saddle points for training our proposed PCNN-MM.

2.6 Physics-Constrained Neural Networks for Solving Multiphysics Problems

Training PCNNs to solve multiphysics problems is similar to a multi-objective optimization or multi-task learning problem. The weighted-sum objective function is commonly used in multi-objective optimization. Different losses in the total objective

function could be in conflict and the gradients of different losses could be unbalanced, both of which can lead to the failure of convergence in training PCNNs. There are three major challenges to solve multiphysics problems using PCNNs. First, the weights of different losses from data and physical constraints need to be adjusted systematically to balance their respective gradients. Second, different physics are tightly coupled with each other in multiphysics problems, which could cause instability during the training. To mitigate the instability, the weights of losses for different physics need to be adjusted properly or the training of different physics need to be partially decoupled. Third, the high dimensionality of the parameter space needs to be reduced to some degree so that the training of PCNNs can escape local minima and converge to a better one.

PCNNs have been used to solve some single-physics problems, such as Navier-Stokes equations [156–158], heat transfer [121], and phase transition [121]. There is only limited work on multiphysics problems, including dendritic growth [121], electroconvection [159], alloy solidification [160], Stefan problem [161], and thermal convection [162,163]. An AI-driven multiphysics simulation framework called SimNet [164] was developed by NVIDIA to accelerate simulations in science and engineering. However, the weights of different losses from data and physical constraints are either fixed or adjusted empirically in PCNNs in the above work.

Some efforts have been made to improve the training of PCNNs to solve multiphysics problems. PCNNs were first applied to dendritic growth with simultaneous consideration of thermal distribution and phase field [121]. By formulating the training of PCNNs as a minimax problem, a novel training algorithm called Dual-Dimer was proposed in our previous work to systematically adjust the weights of different losses for reliable training

[165]. New architectures of PCNNs were also proposed to solve multiphysics problems. New architectures that employ spatio-temporal and multi-scale random Fourier features were proposed to mitigate the issue of unbalanced gradients of different losses [166]. Two subnetworks in a PCNN were trained sequentially to mitigate instability in traditional training methods [167]. However, the high dimensionality is still the bottleneck of training PCNNs to solve multiphysics problems. New architectures of neural networks and novel training algorithms are needed to alleviate the curse of dimensionality in training PCNNs to solve multiphysics problems or other general ML applications.

CHAPTER 3. MULTIPHYSICS SIMULATION OF RAPID SOLIDIFICATION OF TI-6AL-4V ALLOY

3.1 Introduction

Powder bed fusion is a recently developed AM technique for alloys, which builds parts by selectively melting metallic powders with a high-energy laser or electron beam. Different alloys have been used in metal AM processes. Particularly, titanium alloy Ti–6Al–4V has a wide range of applications from aerospace to biomedical devices. As a high strength ($\alpha + \beta$) titanium alloy, Ti–6Al–4V's microstructure mainly depends on its chemical composition, processing condition, and heat treatment history. Nevertheless, there is still a lack of fundamental understanding of the rapid solidification process for better quality control.

To simulate the microstructure evolution of alloys during the rapid solidification, in this chapter, PF-TLBM is proposed to simulate rapid solidification of Ti-6Al-4V alloy by concurrently coupling solute transport, heat transfer, latent heat, fluid dynamics, and phase transition. In this model, the phase-field method simulates the dendrite growth of alloys, whereas the thermal lattice Boltzmann method models heat transfer and fluid flow. The phase-field method and the thermal lattice Boltzmann method are tightly coupled.

In the remainder of this chapter, the formulation of the proposed PF-TLBM model is described in Section 3.2. The simulation results and effects of latent heat and melt flow on the dendrite growth are shown in Section 3.3, which also contains experimental comparison, sensitivity analysis of mesh sizes, as well as quantitative analyses of the temperature gradient, growth velocity, and their combinations.

3.2 Methodology

In the PF-TLBM model, phase formation is described with partial differential equations of phase field and composition variables, whereas fluid flow and thermal effects are modeled with convection-diffusion equations of velocity and temperature fields, respectively. Information exchange between the phase, temperature, and velocity fields is achieved by updating the variables in each iteration of the simulation. The latent heat effect is also incorporated in the simulation of heat transfer. The PF-TLBM model proposed here is an extension of our previous work [168–174]. In the extension, a local non-equilibrium partition coefficient is considered for rapid solidification, and a variable grid computational scheme is developed to simulate the phase field and the temperature field. A coarser grid is used in TLBM to improve simulation efficiency and accuracy because the thermal diffusivity and solute diffusivity differ by three orders of magnitude.

3.2.1 Phase-Field Method

The multi-phase multi-component phase-field method is a generic formulation for phase transitions of alloys. In this work, the multi-phase field method [77] is adopted. The essential component of PFM is a free energy functional that describes the kinetics of phase transition. The free energy functional

$$F = \int_{\Omega} (f^{GB} + f^{CH}) dV \tag{3.1}$$

is defined with an interfacial free energy density f^{GB} and a chemical free energy density f^{CH} in a domain Ω.

A continuous variable called phased field ϕ indicates the fraction of the solid phase in the simulation domain during the solidification process, and the fraction of the liquid phase is $\phi_l = 1 - \phi$. The interfacial free energy density is defined as

$$f^{GB} = \frac{4\sigma^*(\mathbf{n})}{\eta}\left\{|\nabla\phi|^2 + \frac{\pi^2}{\eta^2}\phi(1-\phi)\right\} \tag{3.2}$$

where $\sigma^*(\mathbf{n})$ is the anisotropic interfacial energy stiffness, η is the interfacial width, and $\mathbf{n} = \nabla\phi/|\nabla\phi| = (n_x, n_y)$ is the local normal direction of the interface. The anisotropic interfacial energy stiffness is defined as

$$\sigma^*(\mathbf{n}) = \sigma + \frac{\partial^2\sigma}{\partial\psi^2} = \sigma_0^*\left[1 - 3\delta + 4\delta\left(n_x^4 + n_y^4\right)\right] \tag{3.3}$$

where σ is the interface energy, $\psi = \arctan(n_y/n_x)$ indicates the grain orientation, σ_0^* is the prefactor of interface energy stiffness, and δ is the anisotropy strength of interface energy stiffness, which models the difference between the primary and secondary growth directions of dendrites.

The chemical free energy is the combination of bulk free energies of individual phases

$$f^{CH} = h(\phi)f_s(C_s) + h(\phi_l)f_l(C_l) + \mu[C - (\phi C_s + \phi_l C_l)] \tag{3.4}$$

where C_s and C_l are the weight percentages (wt%) of solute in the solid or liquid phase, respectively. C is the overall composition of a solution in the simulation domain. $f_s(C_s)$ and $f_l(C_l)$ are the chemical bulk free energy densities of solid and liquid phases, respectively. μ is the generalized chemical potential of solute introduced as a Lagrange multiplier to conserve the solute mass balance $C = \phi C_s + \phi_l C_l$. The weight function

$$h(\phi) = \frac{1}{4}\left[(2\phi - 1)\sqrt{\phi(1-\phi)} + \frac{1}{2}arcsin(2\phi - 1)\right] \tag{3.5}$$

provides the coefficients associated with solid and liquid bulk energies.

The evolution of the phase field is described by

$$\frac{\partial \phi}{\partial t} = M_\phi \left\{ \sigma^*(\mathbf{n}) \left[\nabla^2 \phi + \frac{\pi^2}{\eta^2} \left(\phi - \frac{1}{2} \right) \right] + \frac{\pi}{\eta} \sqrt{\phi(1-\phi)} \Delta G_V \right\} \tag{3.6}$$

where M_ϕ is the coefficient of interface mobility, and the driving force is given by

$$\Delta G_V = \Delta S (T_m - T + m_l C_l) \tag{3.7}$$

where ΔS is the entropy difference between solid and liquid phases, T_m is the melting temperature of a pure substance, T is the temperature field, and m_l is the slope of liquidus. Existing studies of interface mobility are restricted to pure metal or one-component systems. For the complex ternary alloy Ti-6Al-4V, there is a lack of information to reveal the dependency of interface mobility on temperature. For simplification, the interface mobility is assumed to be constant in this work.

The evolution of the composition is modeled by

$$\frac{\partial C}{\partial t} + \mathbf{u}_l \cdot \nabla [(1-\phi) C_l] = \nabla \cdot [D_l (1-\phi) \nabla C_l] + \nabla \cdot \mathbf{j}_{at} \tag{3.8}$$

where $\mathbf{u}_l$ is the velocity of the liquid phase. During rapid solidification, the assumption of local composition equilibrium is not reasonable. Therefore, the local non-equilibrium partition coefficient k is computed based on Aziz's model [175,176]

$$k = \frac{C_s}{C_l} = \frac{k_e + V_I \lambda / D_l}{1 + V_I \lambda / D_l} \tag{3.9}$$

where k_e is the equilibrium partition coefficient, λ is the actual interface width in atomic dimensions, and $V_I = \dot{\phi} / |\nabla \phi|$ is the local velocity of the interface. D_l is the diffusion coefficient of liquid, which is assumed to follow an Arrhenius form based on [177]

$$D_l = D_0 \exp\left(\frac{-\Delta E}{RT}\right) \tag{3.10}$$

where D_0 is the prefactor of the diffusion coefficient of the liquid phase, ΔE is the activation energy, and R is gas constant. Furthermore, $\mathbf{j}_{at}$ is the anti-trapping current and defined as

$$\mathbf{j}_{at} = \frac{\eta}{\pi}\sqrt{\phi(1-\phi)}(C_l - C_s)\frac{\partial \phi}{\partial t}\frac{\nabla \phi}{|\nabla \phi|} \tag{3.11}$$

which is used to eliminate the unphysical solute trapping during the interface diffusion process. It removes the anomalous chemical potential jump [40,178] so that simulations can be done more efficiently with the simulated interface width exceeding that of the physical one.

Eqs. (3.6) and (3.8) are the main equations to solve during the phase field simulation. The anti-trapping current was originally introduced for the quasi-equilibrium condition. For simplification, it is still used here under the non-equilibrium condition for rapid solidification, since here the simulated domain size of $90 \times 90\ \mu m^2$ is small and the simulation time of 1.4 ms is short. The upwind scheme of the finite difference method is applied to solve Eqs. (3.6) and (3.8).

3.2.2 Thermal Lattice Boltzmann Method

The conservation equations of mass, momentum, and energy are given by

$$\nabla \cdot (\phi_l \mathbf{u}_l) = 0 \tag{3.12}$$

$$\frac{\partial}{\partial t}(\phi_l \mathbf{u}_l) + \nabla \cdot (\phi_l \mathbf{u}_l \otimes \mathbf{u}_l) = -\frac{\phi_l}{\rho}\nabla P + \nabla \cdot [\nu \nabla (\phi_l \mathbf{u}_l)] + \frac{\mathbf{F}_d}{\rho} \tag{3.13}$$

$$\frac{\partial T}{\partial t} + \nabla \cdot (\phi_l \mathbf{u}_l T) = \nabla \cdot (\alpha \nabla T) + \dot{q} \tag{3.14}$$

respectively, where $\mathbf{u}_l$ is the velocity of liquid with density ρ, P is the pressure, ν is the coefficient of kinematic viscosity, α is the thermal diffusivity, and

$$\mathbf{F}_d = -h^*(1-\phi)\rho\nu\frac{\phi^2}{\eta^2}\mathbf{u}_l \tag{3.15}$$

is the dissipative force caused by the interaction between solid and liquid phases, where $h^* = 147$ is a coefficient fitted from the calculation of Poiseille flow in a channel with diffuse walls [77]. Furthermore,

$$\dot{q} = \frac{L_H}{c_p}\frac{\partial\phi}{\partial t} \tag{3.16}$$

is the released latent heat during solidification, where L_H is the latent heat of fusion, and c_p is the specific heat capacity.

Instead of solving Eqs. (3.12)-(3.14) directly, particle distribution functions for density $f_i(\mathbf{x}, t)$ and temperature $g_i(\mathbf{x}, t)$ are used to capture the dynamics of the system in TLBM. The macroscopic properties of velocity $\mathbf{u}_l$ and temperature T can be calculated based on the density and temperature distribution functions. In the TLBM model, the spatial domain is discretized as a lattice. Particles move dynamically between neighboring lattice nodes. In a two-dimensional D2Q9 model, each node has eight neighbors. The velocity vector

$$\mathbf{e}_i = \begin{cases} (0,0), & i = 0 \\ (\pm c, 0), (0, \pm c), & i = 1, \cdots, 4 \\ (\pm c, \pm c), & i = 5, \cdots, 8 \end{cases} \tag{3.17}$$

represents the velocity along the *i*-th direction in the lattice with respect to a reference node, where $c = \Delta x/\Delta t$ is the lattice velocity with spatial resolution Δx and time step Δt, *i*=0 is the reference lattice node, and *i*=1 to 4 indicate the right, top, left, and down directions,

whereas *i*=5 to 8 indicate the top-right, top-left, down-left, and down-right directions, respectively.

The evolution of particle distribution for density f_i is modeled by

$$f_i(\mathbf{x}+\mathbf{e}_i\Delta t, t+\Delta t) = f_i(\mathbf{x},t) + \frac{1}{\tau_f}\left[f_i^{eq}(\mathbf{x},t) - f_i(\mathbf{x},t)\right] + F_i(\mathbf{x},t)\Delta t \quad (3.18)$$

where $\tau_f = \nu/(c_s^2\Delta t) + 0.5$ is a dimensionless relaxation time parameter with the speed of the sound $c_s^2 = c^2/3$,

$$f_i^{eq}(\mathbf{x},t) = \omega_i\rho\left[1 + \frac{\mathbf{e}_i\cdot\mathbf{u}_l}{c_s^2} + \frac{(\mathbf{e}_i\cdot\mathbf{u}_l)^2}{2c_s^4} - \frac{\mathbf{u}_l^2}{2c_s^2}\right] \quad (3.19)$$

is the equilibrium distribution, and

$$F_i = \left(1 - \frac{1}{2\tau_f}\right)\omega_i\left(\frac{\mathbf{e}_i - \mathbf{u}_l}{c_s^2} + \frac{\mathbf{e}_i\cdot\mathbf{u}_l}{c_s^4}\mathbf{e}_i\right)\cdot\mathbf{F}_d \quad (3.20)$$

is the force source [89,179]. In the D2Q9 scheme, the weights ω_i 's associated with direction *i*'s are

$$\omega_i = \begin{cases} 4/9, & i = 0 \\ 1/9, & i = 1,\cdots,4 \\ 1/36, & i = 5,\cdots,8 \end{cases} \quad (3.21)$$

The evolution of particle distribution for temperature g_i is modeled in parallel by

$$g_i(\mathbf{x}+\mathbf{e}_i\Delta t, t+\Delta t) = g_i(\mathbf{x},t) + \frac{1}{\tau_g}\left[g_i^{eq}(\mathbf{x},t) - g_i(\mathbf{x},t)\right] + Q_i(\mathbf{x},t)\Delta t \quad (3.22)$$

where $\tau_g = \alpha/(c_s^2\Delta t) + 0.5$ is similarly a dimensionless relaxation time,

$$g_i^{eq}(\mathbf{x},t) = \omega_i T\left[1 + \frac{\mathbf{e}_i\cdot\mathbf{u}_l}{c_s^2} + \frac{(\mathbf{e}_i\cdot\mathbf{u}_l)^2}{2c_s^4} - \frac{\mathbf{u}_l^2}{2c_s^2}\right] \quad (3.23)$$

is the equilibrium distribution, and

$$Q_i = \left(1 - \frac{1}{2\tau_g}\right)\omega_i\dot{q} \quad (3.24)$$

is the heat source.

Eqs. (3.18) and (3.23) are the main equations to be solved in TLBM, based on which density and temperature distributions are updated at each time step. During a simulation, the macroscopic quantities of density, velocity, and temperature can be calculated from f_i's and g's as

$$\rho = \sum_i f_i \tag{3.25}$$

$$\rho \mathbf{u}_l = \sum_i \mathbf{e}_i f_i + \frac{\Delta t}{2} \mathbf{F}_d \tag{3.26}$$

$$T = \sum_i g_i + \frac{\Delta t}{2} \dot{q} \tag{3.27}$$

respectively. At each iteration, the properties are calculated, and Eqs. (3.18) and (3.23) are updated accordingly.

In rapid solidification, heat transfer is much faster than solute diffusion, where thermal diffusivity can be three orders of magnitude larger than solute diffusivity. In this work, to reduce the computational cost and improve accuracy, a fine grid spacing dx is used for the PFM simulation, whereas a coarse grid spacing $\Delta x = 30\ dx$ is used for the TLBM simulation. The same time step Δt is used for both simulations. The results of PFM are averaged and transferred to the TLBM model, while the results of TLBM are linearly interpolated as the input for the PFM model. To satisfy the no-slip boundary condition, a bounce-back scheme is used at the solid-liquid interface. The density distribution function at the boundary node $f_{\bar{\imath}}(\mathbf{x}_b, t + \Delta t)$ with the direction $\bar{\imath}$ such that $\mathbf{e}_{\bar{\imath}} = -\mathbf{e}_i$ is determined by

$$f_{\bar{i}}(\mathbf{x}_b, t+\Delta t) = f_i(\mathbf{x}_b, t) + \frac{1}{\tau_f}\left[f_i^{eq}(\mathbf{x}_b, t) - f_i(\mathbf{x}_b, t)\right] - 6\omega_i\rho_w\frac{\mathbf{e}_i \cdot \mathbf{u}_w}{c^2} \quad (3.28)$$

where $\mathbf{u}_w$ is the velocity of the moving wall at the location $\mathbf{x}_w = \mathbf{x}_b + 0.5\mathbf{e}_i\Delta t$ and ρ_w is the density at the wall. For the thermal boundary condition, an anti-bounceback scheme [180–182] is used. At the boundary, the temperature distribution function $g_{\bar{i}}(\mathbf{x}_b, t+\Delta t)$ is given by

$$\begin{aligned} g_{\bar{i}}(\mathbf{x}_b, t+\Delta t) = -g_i(\mathbf{x}_b, t) - \frac{1}{\tau_g}\left[g_i^{eq}(\mathbf{x}_b, t) - g_i(\mathbf{x}_b, t)\right] \\ + 2\omega_i T_w\left[1 + 4.5\frac{(\mathbf{e}_i \cdot \mathbf{u}_w)^2}{c^2} - 1.5\frac{|\mathbf{u}_w|^2}{c^2}\right] \end{aligned} \quad (3.29)$$

The temperature of the wall T_w is given by

$$T_w = T_b - \frac{q_H \Delta x}{2\kappa} \quad (3.30)$$

where T_b is the temperature at the boundary node, q_H is the outward heat flux at the boundary, and κ is the thermal conductivity of the material.

3.2.3 PF-TLBM Algorithm Implementation

In the multi-physics PF-TLBM simulation, different variables are tightly coupled, including liquid velocity $\mathbf{u}_l$, composition C, temperature T, and phase field ϕ and its time derivative $\dot{\phi}$. Figure 3.1 illustrates the algorithm of PF-TLBM. The composition is first calculated based on the initial temperature and phase field by solving Eq. (3.8) with the finite difference method. Then phase field is updated based on Eq. (3.6) with the updated composition values. The dissipative force in Eq. (3.15) is updated with the latest values of the phase field. The total force applied in LBM as in Eq. (3.20) is then updated. Temperature and liquid velocity field are coupled in TLBM as in Eq. (3.23). The updated

velocity values are passed to update the composition by solving the advection equations. The updated temperature and fluid velocity in TLBM are then used in PFM for the next iteration. The proposed PF-TLBM algorithm is implemented and integrated with the open-source phase field simulation toolkit OpenPhase [183].

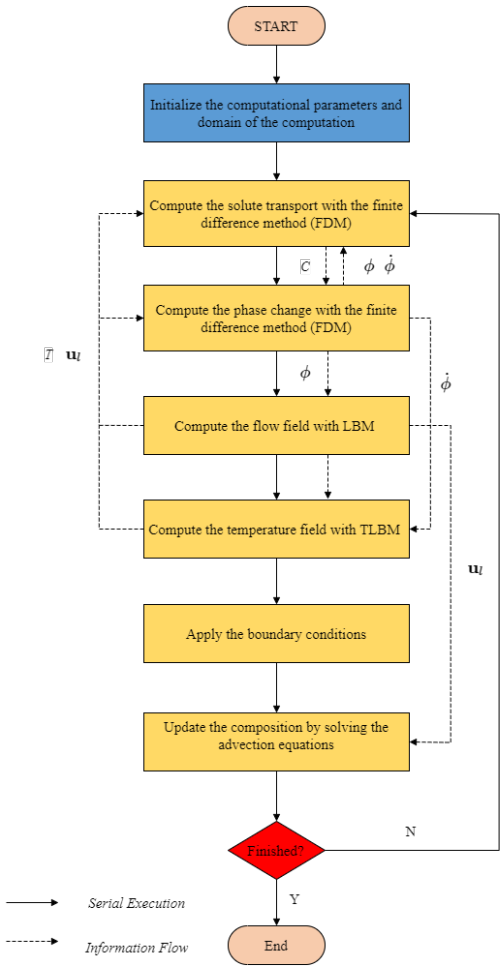


Figure 3.1. The flow chart of the PF-TLBM simulation algorithm

3.3 Simulation Results and Discussion

Here, Ti-6Al-4V alloy is used to demonstrate the PF-TLBM simulation scheme. In this model, the ternary Ti-6Al-4V alloy is treated as a binary alloy, and the solute is the combination of Al and V. This pseudo-binary approach is similar to the existing work

[55,184], which was shown to be an effective replacement of the multi-component approach for modeling solidification kinetics of Ti-6Al-4V alloy. The physical properties of Ti-6Al-4V alloy are given in Table 3.1 [77].

To reduce or eliminate the effect of numerical solute trapping, the fine grid spacing dx should be smaller than the solute diffusion length D_l/V, where D_l is solute diffusivity and V is interface velocity. The maximum dendrite growth velocity is assumed to be $V_{max} = 50$ mm/s. Therefore, a fine grid spacing $dx = 0.1$ μm and a coarse grid spacing $\Delta x = 30\ dx = 3$ μm are adopted. Based on the von Neumann stability analysis or Fourier stability analysis, the upper limit of the time step is $\Delta t \leq min\{dx^2/(4D_l), \Delta x^2/(4v), \Delta x^2/(4\alpha)\}$. Therefore, the time step $\Delta t = 0.1$ μs is applied in all simulation runs. The initial temperature is $T = 1920$ K, which means that the undercooling is 8 K given the initial composition. The length of the simulated domain is $L_x = 900\ dx$ in the *x*-direction and the width is $L_y = 900\ dx$ in the *y*-direction. The initial radius of the nucleus is $D = 9\ dx$ and the interface width is $\eta = 5\ dx$, which means that there are 6 nodes on the interface or boundary layer. The initial composition of the solute is set as $C_0 = 10$ wt% for the whole simulation domain. The setup of boundary conditions for all simulations is schematically illustrated in Figure 3.2. Zero Neumann conditions are set at the bottom $y = 0$ and top $y = L_y$ boundaries for the phase field ϕ and composition C. Although the change of temperature gradient within the melt pool will affect the grain structure and grain size distribution [185], the change of temperature gradient can be assumed to be small given the fact that the small simulation domain is small compared with the whole melt pool. A fixed heat flux $q_H = \rho c_p L_y \dot{T}$ [186] is set at the bottom boundary given the constant cooling rate $\dot{T} = 5 \times 10^4$ K/s, whereas an adiabatic boundary condition is set at the top

boundary. When the dendrite grows in a forced flow, a constant flow velocity $|\mathbf{u}_w| = 0.1\ \mathrm{m/s}$ is imposed at the top boundary of the domain. Periodic boundary conditions are set at the left $x = 0$ and right $y = L_x$ boundaries for the phase field ϕ, composition C, temperature T, and flow $\mathbf{u}_l$. The nuclei are located at the bottom cold wall with constant heat flux to simulate the directional dendrite growth in selective laser melting. The locations of the three nuclei are $x = 10$ μm, 45 μm, and 80 μm, respectively. To compare the simulation results with the experiments done by Simonelli et al. [187], the orientation of the three nuclei is set to be almost the same as the orientation of reconstructed β grains based on the electron backscatter diffraction (EBSD) data.

Table 3.1. Physical properties of Ti-6Al-4V alloy

Physical properties	Value
Melting point of pure Ti, T_m [K]	1941
Liquidus temperature, T_l [K]	1928
Solidus temperature, T_s [K]	1878
Liquidus slope, m_l [K/wt%]	-1.3
Equilibrium partition coefficient, k_e	0.206
Prefactor of interfacial energy stiffness, σ_0^* [J/m^2]	0.5
Interfacial energy stiffness anisotropy, ε^*	0.35
Interface mobility, M_ϕ [m^4/(J·s)]	1.2×10^{-8}
Entropy difference, ΔS [J/(m^3·K)]	1×10^{6}
Physical interface width, λ [m]	3×10^{-9}
Prefactor of diffusion coefficient of liquid phase, D_0 [m^2/s]	5.93×10^{-2}
Activation energy, ΔE [J/mol]	2.5×10^{5}
Kinematic viscosity, ν [m^2/s]	6.11×10^{-7}
Thermal diffusivity, α [m^2/s]	8.1×10^{-6}
Thermal conductivity, κ [W/(m·K)]	28.25
Latent heat of fusion, L_H [J/kg]	2.90×10^{5}
Specific heat capacity, c_p [J/(kg·K)]	872
Density, ρ [kg/m^3]	4000

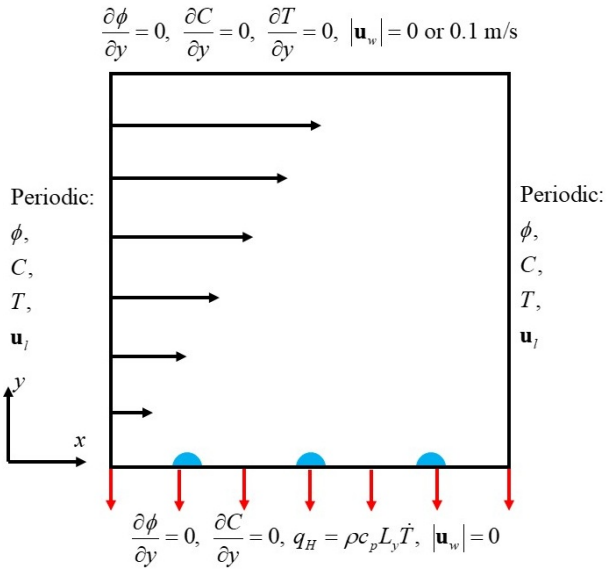


Figure 3.2. Setup of boundary conditions.

3.3.1 Dendrite Growth without Latent Heat

For comparison, dendrite growth is first simulated without the release of latent heat. Figure 3.3 shows the simulation results. The grain identification (ID) 0 represents the liquid phase, while other grain IDs represent solid phases with different orientations. Using the temperature gradient $G = |\nabla T|$ and growth rate V, a solidification map is constructed based on the values of the local cooling rate GV and the ratio G/V [188]. The solidified microstructure can be equiaxed dendritic, columnar dendritic, cellular, or planar as the ratio G/V increases. When the ratio G/V is small at the beginning of the simulation, the columnar dendritic growth pattern can be easily recognized at the time of 0.35 ms, as shown in Figure 3.3(a). The primary arms and secondary arms can be differentiated easily. It is observed that the primary arms of the dendrite grow faster than the secondary arms, as a result of the anisotropy of the interface energy. Without the release of latent heat, the secondary arms grow so fast that they quickly merge with each other as shown in Figure

3.3(b-d). It is also seen in Figure 3.3(d) that growth competition between grains of different orientations exists. Vertices or corners occur during dendrite growth, as highlighted by circles. The segregation of solute occurs at the solid-liquid interface because the solid phase has a lower composition than the liquid phase. High segregation of solute can be observed at the grain boundaries between secondary arms inside the grains, as shown in Figure 3.3(e).

In this model, the effect of latent heat is not considered. As a result, the temperature is reduced monotonically from the top to the bottom of the simulation domain. At the same time, the detailed morphology of secondary arms cannot be observed, and there is no gap between grains. With the limitation of in-situ experimental methods, there is still no direct observation of dendrite growth under rapid solidification. For a slow solidification process, in-situ X-ray microscopy experiments [189] showed much slower growth of secondary arms and that gaps between grains sustain for a long period during dendrite growth. Therefore, it is reasonable to suspect that the simulation without latent heat overestimates the solidification speed.

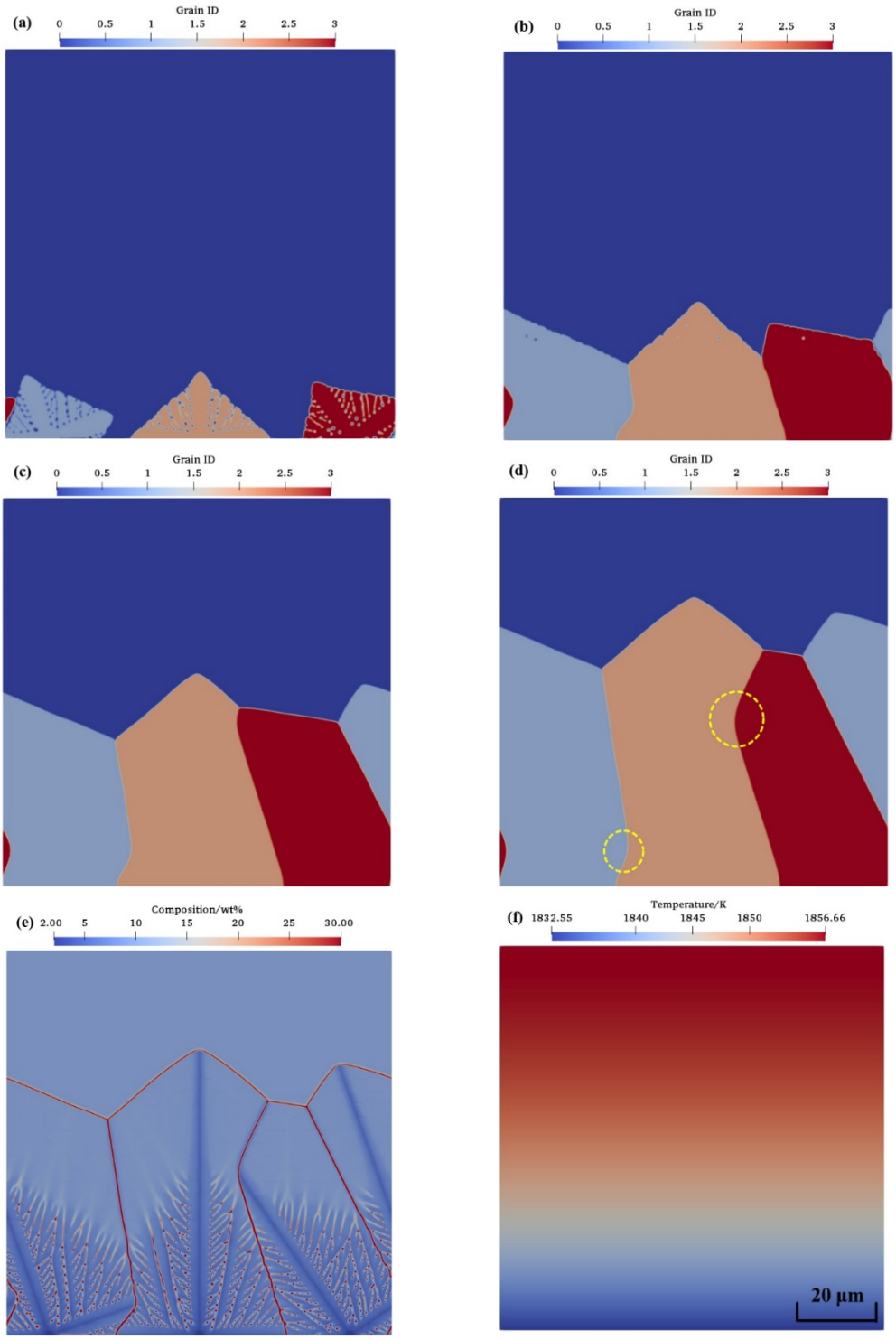


Figure 3.3. Dendrite growth without latent heat. Phase field at (a) 0.35 ms, (b) 0.7 ms, (c) 1.05 ms, (d) 1.4 ms, (e) composition field at 1.4 ms, and (f) temperature field at 1.4 ms.

3.3.2 Non-Isothermal Dendrite Growth with Latent Heat

In the second model, non-isothermal dendrite growth with the release of latent heat during the phase transition is considered. Figure 3.4 shows the simulation results. The temperature field, composition distribution, and the morphology of the dendrite are quite different from the case of dendrite growth without latent heat in Section 3.3.1. The columnar dendritic growth pattern is shown in Figure 3.4(a-d). Because of the release of latent heat, the temperature gradient G is smaller than that in the case without latent heat, which results in a lower ratio G/V. At the initial stage of growth, the columnar dendrites grow with the four-fold symmetry that is similar to equiaxed dendrites. Because of the high temperature gradient along the vertical direction, the vertical secondary arms become dominant, while the growth of horizontal secondary arms is suppressed. In Figure 3.4(e), high segregation of solute can be observed at the grain boundaries and between secondary arms inside the grains, where some small portions of liquid are trapped and surrounded by the solid phase. The composition of the trapped liquid phase increases as the liquid phase shrinks. The small pocket of the liquid phase may remain liquid for a long period until solid diffusion takes away the remaining solute supersaturation before it is completely solidified. The degree of solute segregation at the solid-liquid phase decreases from the bottom to the top of the grains.

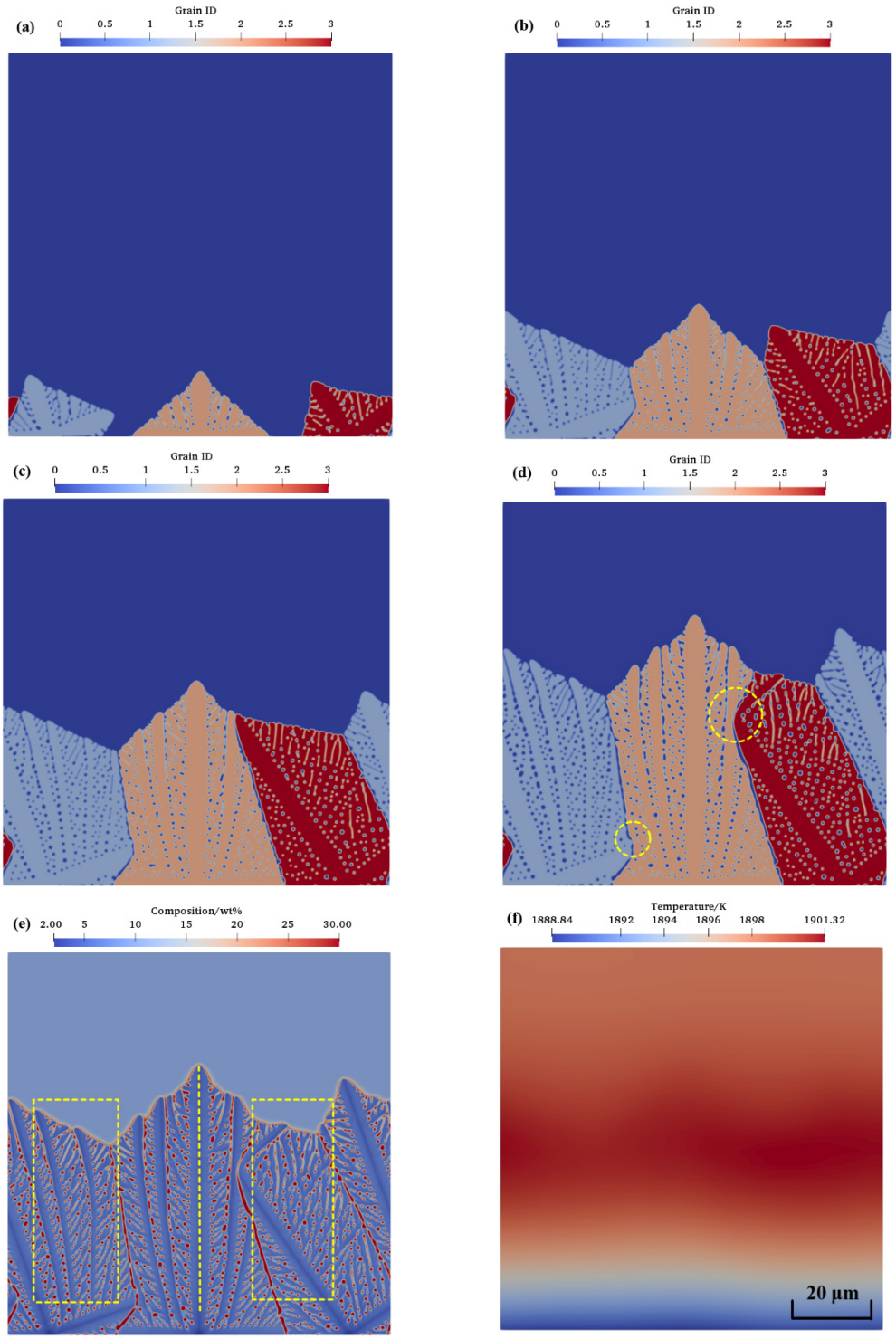


Figure 3.4. Non-isothermal dendrite growth with latent heat. Phase field at (a) 0.35 ms, (b) 0.7 ms, (c) 1.05 ms, (d) 1.4 ms, (e) composition field at 1.4 ms, and (f) temperature field at 1.4 ms.

The simulated solute trapping is verified as follows. Based on the simulation results, the partition coefficient at the tips of dendrites is estimated as $k = C_s/C_l \approx 0.223$. With $V_I = 0.043$ m/s and $D_l = 7.9 \times 10^{-9}$ m/s^2, the partition coefficient, according to Aziz's model, is calculated as

$$k = \frac{k_e + V_I \lambda / D_l}{1 + V_I \lambda / D_l} \approx 0.219,$$

which is close to the above simulation result. The average temperature in the whole simulation domain is higher than that in the case without latent heat. The temperature of the solid phase is higher than that of the liquid phase, as shown in Figure 3.4(f), which decreases the undercooling and the driving force of growth. The release of latent heat prevents the secondary arms from merging with each other quickly, which explains the columnar dendritic growth to some extent. The simulation results suggest that it is important to consider heat transfer, especially latent heat, during the solidification process, which provides detailed composition, temperature, and grain growth pattern information.

3.3.3 Non-Isothermal Dendrite Growth with Latent Heat in a Forced Flow

A further refinement of the model is to incorporate fluid flow. A constant flow velocity $|\mathbf{u}_w| = 0.1$ m/s is imposed at the top boundary of the domain along the positive *x*-direction. Simulation results are shown in Figure 3.5. Note that the magnitude of the velocity field is represented by the colors of the arrows rather than their sizes. The velocities corresponding to the arrows appearing in the solid phase region are near zero.

It is observed that the columnar dendrite morphology is slightly different from that in non-isothermal dendrite growth without flow. Compared to Figure 3.4(e), the growth of some horizontal secondary arms in Figure 3.5(e) is enhanced under the effect of flow,

which is shown in the regions highlighted with rectangles. In addition, the primary dendrite is inclined slightly under the forced flow, as the vertical dashed line in Figure 3.5(e) indicates. When the flow encounters the continually growing dendrites, the local velocity field is disturbed. Some vortexes are observed in Figure 3.5(a). The flow changes the dendrite morphology by affecting both the composition and the temperature field. The flow can accelerate grain growth by enhancing solute diffusion and increasing undercooling, which results in a higher driving force. It is also observed in Figure 3.5(f) that the temperature and temperature gradient rise slightly in a forced flow. This is because the flow enhances the growth of some horizontal secondary arms and increases the release of latent heat. The simulation results suggest that the melt flow has some effect on dendrite growth. However, our sensitivity study shows that rapid solidification can suppress the flow effect if velocity is relatively small.

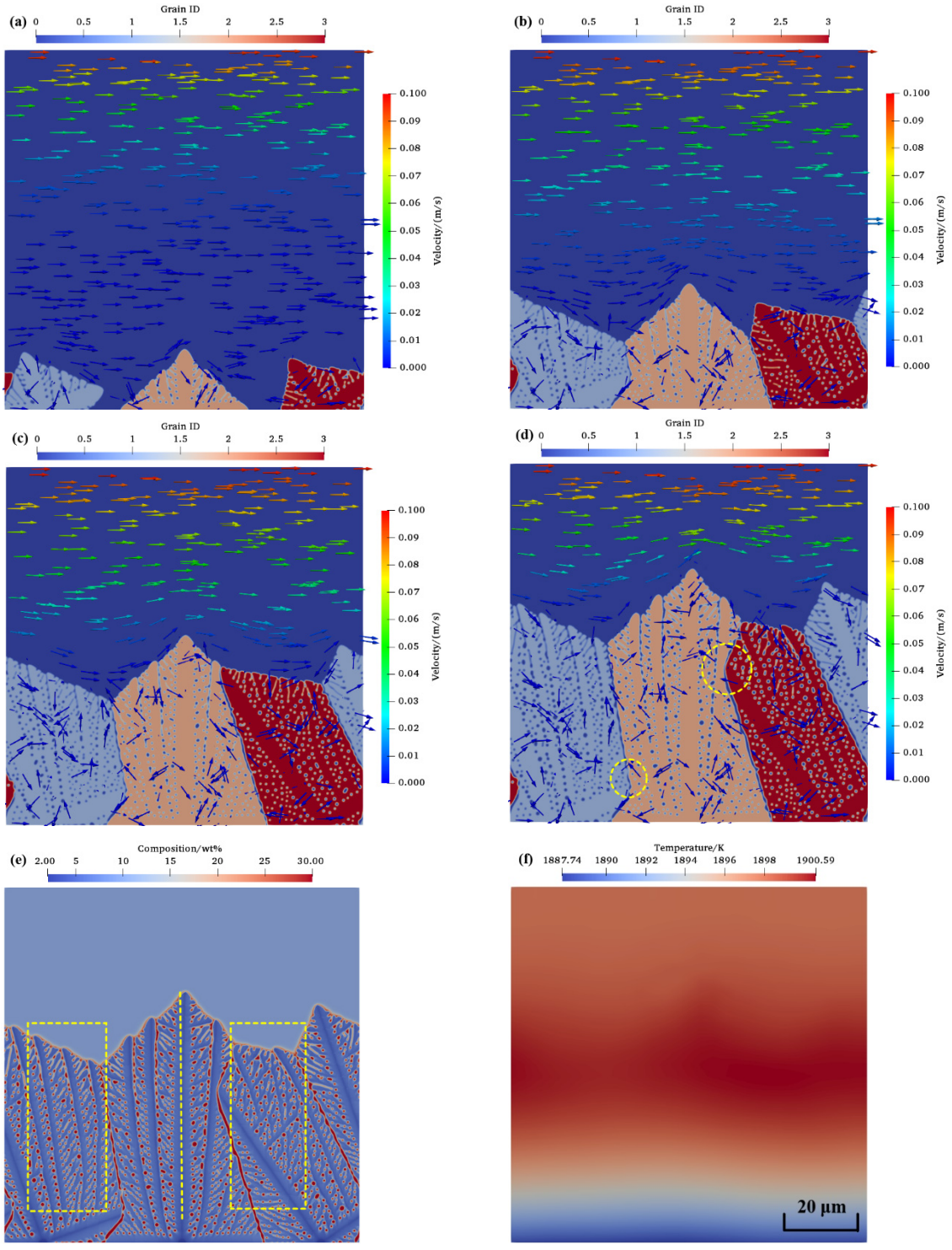


Figure 3.5. Non-isothermal dendrite growth with latent heat in a forced flow. Phase field and flow field at (a) 0.35 ms, (b) 0.7 ms, (c) 1.05 ms, (d) 1.4 ms, (e) composition field at 1.4 ms, and (f) temperature field at 1.4 ms.

3.3.4 Experimental Comparison

Solidification of Ti-6Al-4V has several pathways, including suppression of the reaction and primary beta phase formation, monovariant reactions, and invariant reactions for the residue alloy melt [190]. Our model simulates the rapid solidification process of Ti-6Al-4V with emphasis on primary beta phase formation. During the SLM process of Ti-6Al-4V alloy, the β phase is formed from the liquid. Then the prior β phase transforms to the acicular α' martensite phase. This solid-state phase transition is described by the Burgers orientation relationship. However, the solid-state phase transition is not considered in our solidification simulation. Given that in-situ experimental observation of dendrite evolution during the rapid solidification process is challenging, it is difficult to compare simulated dendrite morphology and growth with experimental observation directly.

Nevertheless, EBSD images of acicular α' martensite phases, which originate from the parent β grains, are available. Here, the simulated dendrite morphology is compared with the reconstructed prior β phase orientation map from an EBSD image [187], as shown in Figure 3.6. It is observed that acicular α' martensite phases are formed in the prior columnar β grains. Usually, prior columnar β grains have a high aspect ratio because of the high temperature gradient along the building direction. The simulated dendrite morphology in Figure 3.5(d) matches qualitatively with the prior columnar β grains, such as the bottom-right corner with a size of 90×90 μm^2 in Figure 3.6. The primary arm spacing is 35 μm. Because of the growth competition between grains of different orientations, curved grain boundaries, highlighted by circles, are observed when two dendrites encounter each other, which was also predicted by simulations. Furthermore, the secondary arm spacing of the simulated microstructure is $\lambda_2 = 1.2$ μm, which is close to

the calculated value $\lambda_2 = 1.5$ μm based on an analytical model proposed by Bouchard and Kirkaldy [191], as

$$\lambda_2 = 12\pi \left[\frac{4\sigma}{C_0(1-k)^2 \rho L_H} \left(\frac{D_l}{V_I} \right)^2 \right]^{\frac{1}{3}} \tag{3.31}$$

The difference between the predicted and observed secondary arm spacing is possibly caused by parameter uncertainty and model form uncertainty. The parameter uncertainty can be associated with the interface energy σ, latent heat L_H, solute diffusivity D_l, and local velocity of the interface V_I.

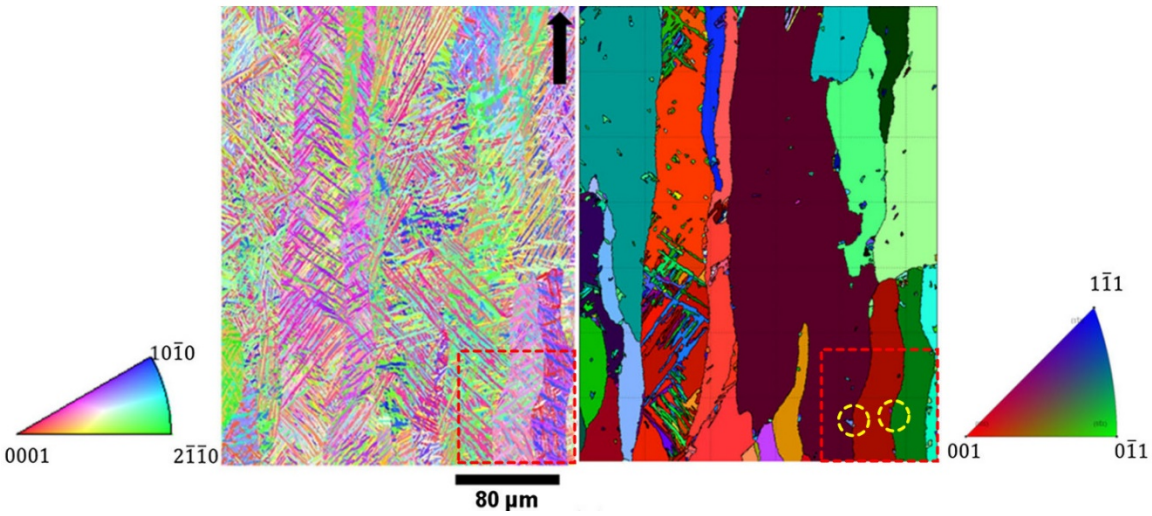


Figure 3.6. α' and corresponding reconstructed β orientation maps from EBSD data. Courtesy of Simonelli et al. [187]

Though microsegregation occurs in the SLM of AlSi10Mg [192] and IN718 [193] alloys, the microsegregation effect for the main alloy elements Al and V is weak in metal AM of Ti-6Al-4V alloy [194,195]. The weak microsegregation effect of Ti-6Al-4V alloy is mainly because the partition coefficients of Al and V are close to unity [196,197]. It has been shown that when trace element Fe with a theoretical partition coefficient of 0.38 is used to refine the grains and reduce the microstructural anisotropy of Ti-6Al-4V alloy, a

strong microsegregation effect can be observed during the rapid solidification[198]. Therefore, it is necessary to simulate solute transport for the accurate prediction of microstructure evolution in metal AM. The difference in compositions between our simulation and experimental results is likely caused by the inaccurate equilibrium partition coefficient in the pseudo binary approach. In future work, a multi-component multi-phase field model is needed to predict the composition distribution more accurately.

3.3.5 Convergence Study with Finer Mesh

To assess the sensitivity of mesh size on the simulation results, a finer mesh $dx = 0.03$ μm is used in the convergence study. Other simulation setups are kept the same. Figure 3.7 shows the simulation results of dendrite growth without latent heat and non-isothermal dendrite growth with latent heat in a forced flow at 0.7 ms. After the mesh refinement, the difference in dendrite growth speed with and without latent heat becomes more obvious. Without latent heat, as shown in Figure 3.7(a), some detailed morphology of secondary arms can now be observed around the dendrite tips, but not at the bottom of dendrites. In contrast, with latent heat, as shown in Figure 3.7(c), the morphology has clear patterns of secondary arms that are similar to the ones in Figure 3.5(b). The growth speed of dendrites using the fine mesh $dx = 0.03$ μm is almost the same as that in the coarse mesh $dx = 0.1$ μm. The dendrite growth slows down when latent heat is considered. The solute distribution with the fine mesh is also similar to that of the coarse mesh. The results further confirm that considering latent heat is necessary to reveal the details of secondary arms and provide more realistic kinetics of dendrite growth. Compared to the fine mesh, the simulation with the coarse mesh $dx = 0.1$ μm reveals enough details of dendrite growth and reduces the computational cost.

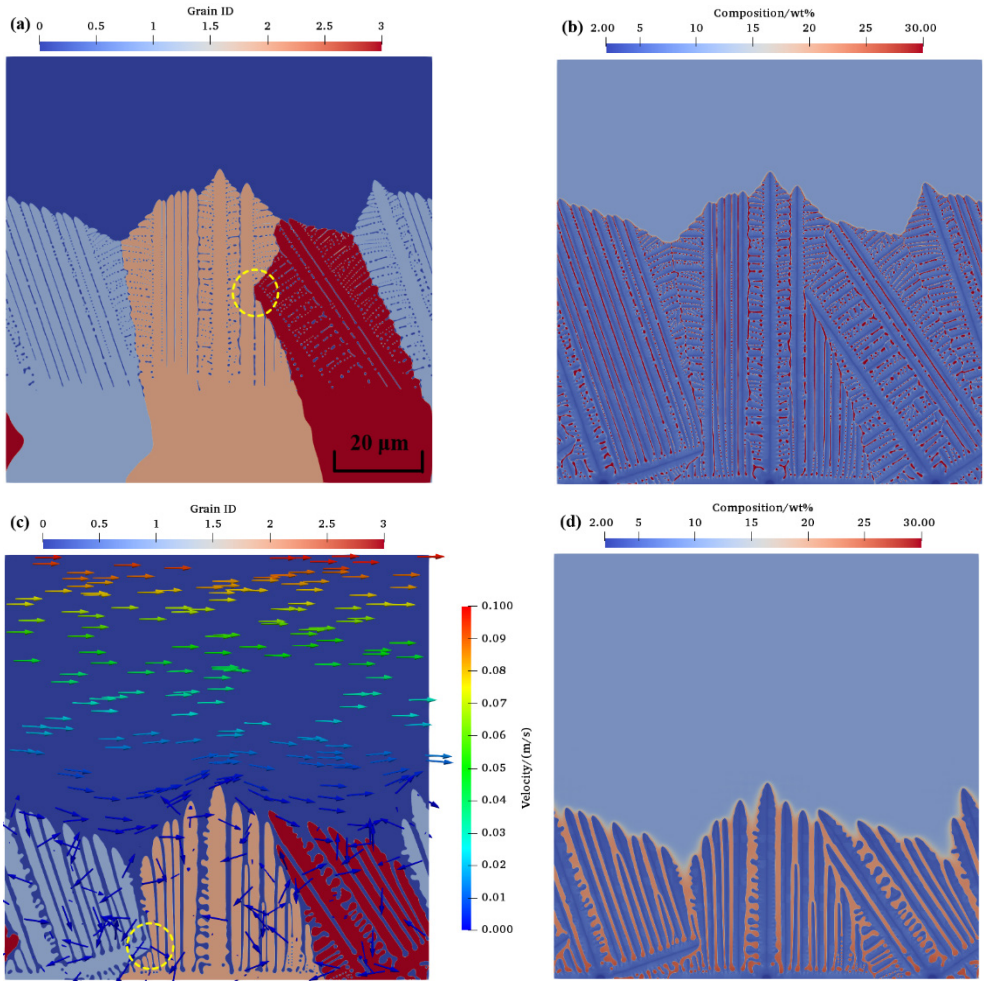


Figure 3.7. With fine mesh, (a) phase field and (b) composition field in dendrite growth without latent heat at 0.7 ms; (c) phase field and flow field, and (d) composition field in non-isothermal dendrite growth with latent heat in a forced flow at 0.7 ms.

3.3.6 Quantitative Analysis

To compare the effects on temperature quantitatively, the thermal histories in different simulation scenarios are plotted in Figure 3.8, where the three curves are the temperatures observed at the location of $x = 45$ μm and $y = 0$ μm for the cases without latent heat, with latent heat, and with latent heat and flow, respectively. There is little difference in the thermal histories with and without considering melt flow, whereas

considering latent heat gives a significantly different prediction. At the beginning of solidification ($0 \leq t < 175$ μs), the effect of latent heat is not obvious because the fraction of phase transition is small. The temperature drops at a similar rate for all three cases. When $t \geq 175$ μs, the temperature without latent heat decreases linearly. In contrast, the temperature with latent heat decreases slowly and starts to increase at $t = 875$ μs because of the continuous release of latent heat. The phenomenon is commonly known as recalescence during the solidification of metals, similarly observed in the simulation results of Ref. [186].

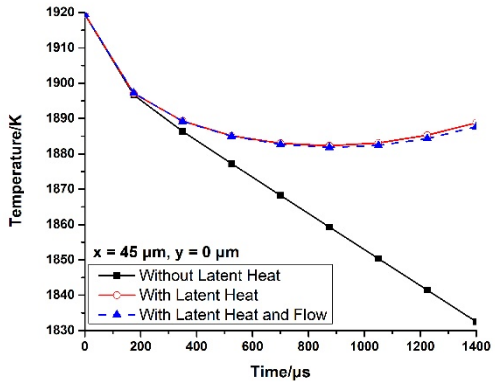


Figure 3.8. Thermal histories at the location of $x = 45$ μm and $y = 0$ μm under different conditions.

Figure 3.9 shows the temperature distribution of non-isothermal dendrite growth along the y-direction at $x = 45$ μm. It is observed that the forced flow reduces the temperature values but increases the temperature gradients only slightly.

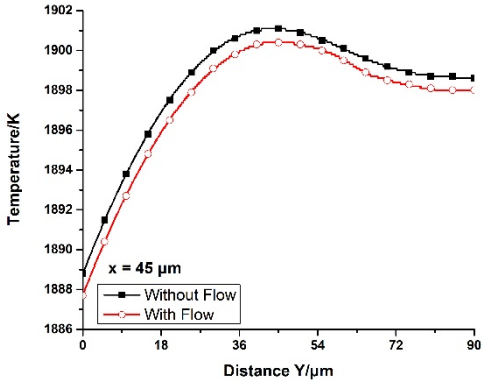


Figure 3.9. The temperature distribution of non-isothermal dendrite growth along the vertical line at $x = 45$ μm.

Table 3.2 summarizes the dendrite tip temperature gradient G, dendrite tip growth velocity V, and their combinations for the three cases of simulations. When the release of latent heat is considered, the dendrite tip temperature gradient G, dendrite tip growth velocity V, and average growth velocity V_{ave} are smaller than those without latent heat. The local cooling rate GV and the ratio G/V are also lower. When a forced flow is imposed, the dendrite tip temperature gradient G and dendrite tip growth velocity V slightly increase. This suggests that the forced flow can accelerate dendrite growth, resulting in further release of latent heat and a higher temperature gradient. The average growth velocities V_{ave} with the flow and without the flow are almost the same, which means that the release of latent heat can stabilize the dendrite growth. The local cooling rate GV and the ratio G/V increase slightly with the forced flow. The effect of melt flow on dendrite growth is suppressed by rapid solidification.

Table 3.2. Quantitative analysis of simulation results

	Without latent heat	With latent heat	With latent heat and flow
Dendrite tip temperature gradient G at 1.4 ms [K/mm]	130	80	100
Dendrite tip growth velocity V at 1.4 ms [mm/s]	49	43	44
Average growth velocity V_{ave} [mm/s]	46.9	44.7	45
Local cooling rate GV [K/s]	6370	3440	4400
Ratio G/V [K·s/mm^2]	2.65	1.86	2.27

3.4 Conclusions

In this work, a mesoscale multi-physics model is developed to simulate the rapid solidification of Ti-6Al-4V alloy by integrating the phase-field method and the thermal lattice Boltzmann method. This model simulates the rapid solidification process of Ti-6Al-4V with emphasis on primary β phase formation. The model concurrently predicts solute transport, phase transition, heat transfer, latent heat, and melt flow. The local non-equilibrium partition coefficient is calculated based on Aziz's model to compute the solute distribution during rapid solidification. The diffusivity of the liquid is temperature-dependent, while other physical properties of Ti-6Al-4V are assumed to be constant. By considering the release of latent heat, the model can predict the composition distribution, temperature field, grain growth, and dendrite morphology with more detail than models without latent heat. The results show that considering latent heat is important for modeling thermal effects on dendrite growth. The average growth rate V_{ave} is lower with latent heat than without it. The local cooling rate GV and the ratio G/V are lower as well. The recalescence occurs during the non-isothermal dendrite growth.

The effect of fluid flow on dendrite growth is small under rapid solidification. The advection changes the distributions of temperature and composition. The flow can accelerate the grain growth by enhancing solute diffusion and increasing undercooling, which results in a higher driving force. The forced flow enhances the growth of horizontal secondary arms and increases the temperature gradient slightly.

The simultaneous considerations of solute transport, kinetics of phase transition, and thermal effects are necessary to understand rapid solidification. The multi-physics modeling approach can elucidate the complex physical processes with more details. The challenge of coupling multiple physical effects is the highly varied time scales used in these simulated processes. Because of the high cooling rate in rapid solidification, the time step needs to be small enough for numerical stability. However, the dimensionless relaxation time in TLBM should be greater than 0.5 and not much larger than 1 because of truncation errors [199]. The trade-offs mostly rely on sensitivity studies. In our model, a variable grid approach is taken to treat PFM and TLBM separately to alleviate this problem.

There are several approximations in our model that may affect the accuracy of predictions. First, the interface mobility is assumed to be constant. However, it depends on the temperature in reality. Since there is a lack of experimental results, molecular dynamics simulations can be applied to estimate mobility and assess the influential factors such as temperature. Interatomic potentials for Ti-6Al-4V alloy for molecular dynamics nevertheless need to be developed. Second, the pseudo-binary approach is adopted to model the ternary alloy. However, Al and V will not be trapped in the same way during rapid solidification. A multi-component multi-phase field model is needed to reveal further details. Third, to simulate the complete process of SLM, nucleation and solid-state phase

transition should also be considered to predict the final microstructure. This allows for direct quantitative comparison and model validation based on existing experimental capabilities. Some emerging in-situ characterization techniques for rapid solidification such as dynamic transmission electron microscopy [200] can help calibrate and validate models. Fourth, the current model is only applied to the 2D domain. Future work will include the extension to 3D domains. 3D models will be much more computationally demanding. Parallelization is a viable solution to this issue. Both the phase-field method and the thermal lattice Boltzmann method can be easily adapted for parallel computation.

CHAPTER 4. MULTIPHYSICS SIMULATION OF NUCLEATION AND GRAIN GROWTH IN SELECTIVE LASER MELTING OF ALLOYS

4.1 Introduction

In this chapter, a nucleation model is introduced in our previously developed PF-TLBM [168–174] model to simulate heterogeneous nucleation at the boundary of the melt pool in SLM. AlSi10Mg alloy is used to demonstrate the simulation framework. AlSi10Mg alloy, with good weldability, hardenability and high dynamic properties, has been widely applied in the automotive and aerospace industries. The main contribution of this work is the simulation of nucleation and dendritic growth of alloys in the small melt pool of SLM, where heterogeneous nucleation tends to occur at the boundary. A new method to calculate heat fluxes out of the small melt pool for PF-TLBM is also developed, given a constant cooling rate. The effects of latent heat and cooling rate on dendritic morphology and solute distribution are studied. The PF-TLBM model is also extended to predict the multi-layer epitaxial grain growth in the complex heating and cooling environment in SLM. The re-melting and solidification process in multiple scanning passes are simulated. A marching cell simulation scheme is proposed to further reduce the computational complexity.

In the melt pool model, the nucleus with random orientations are planted at the bottom of a 2D simulation domain in advance to simulate the columnar dendrite growth during the SLM process. An accurate and reliable thermal model is critical for predicting microstructure evolution. To save the computational cost, however, the well-known

Rosenthal equation rather than the thermal lattice Boltzmann method is used to predict the thermal history for simulating multi-layer epitaxial grain growth.

In this chapter, the melt is assumed to be static in the small melt pool for simplification. The solid phase transition and recrystallization are not simulated. The effect of Marangoni flow on dendritic growth will be considered in future work. In the remainder of this chapter, the formulation of PF-TLBM with zero fluid velocity, the new nucleation model, and the new method for calculation of heat fluxes out of the melt pool are described in Section 4.2. The thermal model and the marching cell simulation scheme for simulating multi-layer epitaxial grain growth are also shown in Section 4.2. The simulation settings and simulation results of single-layer and multi-layer dendritic growth of AlSi10Mg alloy are shown in Section 4.3. It also includes the effects of latent heat and cooling rate on dendritic growth. The quantitative analyses of thermal history, the time evolution of solid-phase fraction, and composition distribution are also provided.

4.2 Methodology

In this study, the simplified formulation of PF-TLBM with zero velocity in the static melt is used, which is different from the original formulation of PF-TLBM in Section 3.2. The phase-field and composition are calculated by solving the Allen-Cahn equation and diffusion equation. Since the melt is static, the kinetic equation for the composition field is modified to

$$\frac{\partial C}{\partial t} = \nabla \cdot [D_l(1-\phi)\nabla C_l] + \nabla \cdot \mathbf{j}_{at} \tag{4.1}$$

TLBM [82,86] is used to calculate the temperature field only for simulating single-layer dendritic growth in this study. TLBM is used to simulate the temperature evolution in the

melt pool and consider the effects of the release of latent heat at the solid-liquid interface. PFM and TLBM are tightly coupled by updating and exchanging the information of phase, composition, and temperature fields in each iteration of the simulation. More details about TLBM can be found in Section 3.2.2.

In the remainder of this section, the thermal lattice Boltzmann method for static melt is introduced in Section 4.2.1. In Section 4.2.2, the new nucleation model is described, which is used to simulate the heterogeneous nucleation in the mushy zone of the melt pool in the SLM process. In Section 4.2.3, a new method to calculate heat fluxes out of the SLM melt pool model given a constant cooling rate is described. The thermal model and the marching cell simulation scheme for simulating multi-layer epitaxial grain growth are shown in Section 4.2.4.

4.2.1 Thermal Lattice Boltzmann Method for Static Melt

The heat conduction equation is given by

$$\frac{\partial T}{\partial t} = \nabla \cdot (\alpha \nabla T) + \dot{q} \tag{4.2}$$

where α is the thermal diffusivity. The released latent heat $\dot{q}$ during solidification is given by Eq. (3.16). Instead of solving Eq. (4.2) directly, a particle distribution function of temperature $g_i(\mathbf{x}, t)$ is utilized to capture the dynamics of the system in TLBM. The equilibrium particle distribution of temperature is given by

$$g_i^{eq}(\mathbf{x}, t) = \omega_i T \tag{4.3}$$

To improve the computational efficiency, a fine grid spacing dx is used for the PFM simulation, whereas a coarse grid spacing $\Delta x = 50\ dx$ is used for the TLBM simulation. The same time step Δt is used for both simulations. The results of TLBM are linearly

interpolated as the input for the PFM model, whereas the results of PFM are averaged and transferred to the TLBM model in each iteration. The anti-bounceback scheme [181,182] is used for the thermal boundary condition. The particle distribution of temperature at the boundary node $g_{\bar{\imath}}(\mathbf{x}_b, t+\Delta t)$, for direction $\bar{\imath}$ such that $\mathbf{e}_{\bar{\imath}} = -\mathbf{e}_i$, is determined by

$$g_{\bar{\imath}}(\mathbf{x}_b, t+\Delta t) = -g_i(\mathbf{x}_b, t) - \frac{1}{\tau_g}\left[g_i^{eq}(\mathbf{x}_b, t) - g_i(\mathbf{x}_b, t)\right] + 2\omega_i T_w \qquad (4.4)$$

4.2.2 Nucleation Model

During the SLM process, columnar dendrites grow from the bottom of the melt pool upwards, as usually observed in experiments. Heterogeneous nucleation usually has a much lower energy barrier than homogeneous nucleation. Therefore, it is reasonable to assume that heterogeneous nucleation dominates and nuclei concentrate at the solid-liquid interface. To simulate the heterogeneous nucleation process, a Poisson seeding algorithm [62,63] is adopted. Nucleation can be treated as fully localized events and can be modeled as a Poisson process. The major assumption is the spatial and temporal independence between events with the memoryless property. The nucleation probability is given by

$$P_n = 1 - exp(-Iv\Delta t) \qquad (4.5)$$

where I is the nucleation rate, v is the cell spacing, and Δt is a sufficiently small time interval. Based on the CNT, the nucleation rate can be calculated by

$$I = I_0 exp\left[-\frac{16\pi\sigma^3 f(\bar{\theta})}{3k_B T(\Delta G_V)^2}\right] \qquad (4.6)$$

where $I_0 \approx 1\times10^{16}\ \mathrm{m}^{-2}s^{-1}$ is the prefactor of the nucleation rate determined by the jump frequency across the interface, σ is the interface energy, $f(\bar{\theta}) = (2 - 3cos\bar{\theta} + cos^3\bar{\theta})/4 = 1\times10^{-5}$ with $\bar{\theta}$ as the contact angle, k_B is the Boltzmann constant, ΔG_V is

the driving force in Eq. (3.7). The prefactor of the nucleation rate for AlSi10Mg alloy is calibrated based on the average β grain size observed in the SLM experiment [201], which is 5 μm. During each time step, the nucleation probability P_n is calculated at each liquid cell at the boundary of the melt pool during the simulation. At the same time, a random number with the standard uniform distribution between 0 and 1 will be generated and compared with the nucleation probability P_n. If the random number is less than the nucleation probability P_n, then the nucleus is planted.

4.2.3 Calculation of Heat Fluxes out of the Melt Pool

The setup of boundary conditions for simulating single-layer dendritic growth in SLM is schematically illustrated in Figure 4.1. The rectangular region stands for the cross-section of the melt pool, which is perpendicular to the scanning direction. The curve indicates the boundary of the melt pool, where nuclei with random distributions are generated. Zero Neumann conditions are set at all boundaries for the phase-field ϕ and composition C.

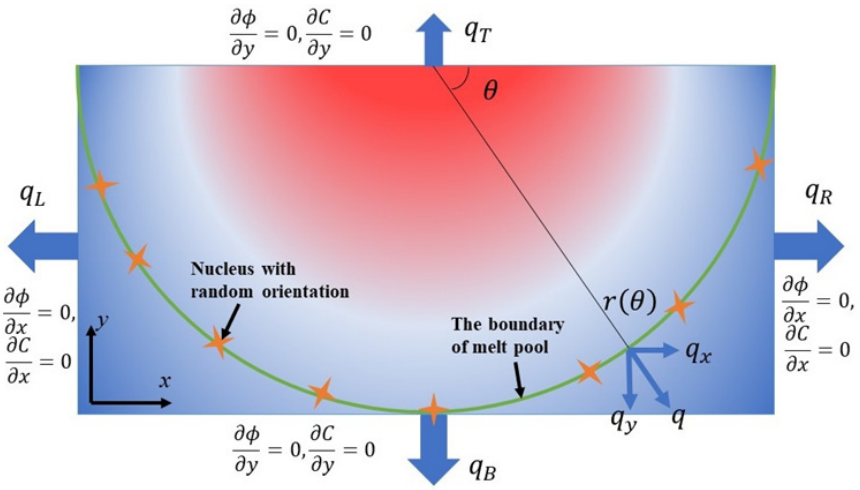


Figure 4.1. Schematic diagram of the setup of boundary conditions.

Constant cooling rates are applied indirectly. It is noted that the nonisothermal solidification in the melt pool is simulated by applying heat fluxes at boundaries rather than cooling the whole simulation domain at a constant cooling rate. Firstly, four constant heat fluxes q_T, q_B, q_L, and q_R are calculated based on a constant cooling rate. Then these four heat fluxes q_T, q_B, q_L, and q_R are applied at the top, bottom, left, and right boundaries, respectively. The relationship between a constant cooling rate and heat fluxes at the boundaries needs to be established so that the solidification in the melt pool is nonisothermal to reflect the real situation. Three heat fluxes q_B, q_L, and q_R are estimated based on their geometric relation to the heat flux $\mathbf{q}$ which is normal to the boundary of the melt pool. More specifically, $\mathbf{q}$ is decomposed into three heat fluxes q_B, q_L, and q_R. It is assumed that the heat flux $\mathbf{q}$ has a constant magnitude. It is necessary to determine the relationship between heat fluxes q_B, q_L, and q_R so that the magnitude of heat fluxes at boundaries can be calculated. To make it more general, it is assumed that the shape of the melt pool is a semi-ellipse, which is defined as

$$r(\theta) = (a\cos\theta, b\sin\theta) \tag{4.7}$$

where a is the major axis, b is the minor axis, and θ is an angular parameter that defines the position. The heat flux $\mathbf{q}$ normal to the boundary of the melt pool has a constant magnitude and is given by

$$\mathbf{q} = q\mathbf{N} = \frac{q}{\sqrt{a^2\sin^2\theta + b^2\cos^2\theta}}(b\cos\theta, a\sin\theta) \tag{4.8}$$

where $\mathbf{N}$ is the unit normal vector perpendicular to the boundary of the melt pool. The heat flux $\mathbf{q}$ can be decomposed to $\mathbf{q_x} = q/\sqrt{a^2\sin^2\theta + b^2\cos^2\theta}(b\cos\theta, 0)$ and $\mathbf{q_y} = q/\sqrt{a^2\sin^2\theta + b^2\cos^2\theta}(0, a\sin\theta)$. Because the semi-ellipse is symmetric with respect to the

y-axis, let us consider the case when $-2/\pi \leq \theta \leq 0$ first. By using vector calculus, the rate of heat flow caused by the horizontal heat flux $\mathbf{q_x}$ can be calculated by

$$\begin{aligned} \dot{Q}_x &= \int_{-\frac{\pi}{2}}^{0} \mathbf{q_x} \cdot \mathbf{N}\, ds = \int_{-\frac{\pi}{2}}^{0} \mathbf{q_x} \cdot \mathbf{N}\, |r'(\theta)| d\theta \\ &= \int_{-\frac{\pi}{2}}^{0} \frac{qb^2 cos^2\theta}{\sqrt{a^2 sin^2\theta + b^2 cos^2\theta}} d\theta \end{aligned} \tag{4.9}$$

Similarly, the rate of heat flow caused by the vertical heat flux $\mathbf{q_y}$ is given by

$$\begin{aligned} \dot{Q}_y &= \int_{-\frac{\pi}{2}}^{0} \mathbf{q_y} \cdot \mathbf{N}\, ds = \int_{-\frac{\pi}{2}}^{0} \mathbf{q_y} \cdot \mathbf{N}\, |r'(\theta)| d\theta \\ &= \int_{-\frac{\pi}{2}}^{0} \frac{qa^2 sin^2\theta}{\sqrt{a^2 sin^2\theta + b^2 cos^2\theta}} d\theta \end{aligned} \tag{4.10}$$

Both $\dot{Q}_x$ and $\dot{Q}_y$ can be calculated by numerical integration. On the other hand, from the definition of rate of heat flow, we have

$$\frac{\dot{Q}_x}{\dot{Q}_y} = \frac{q_R L_y}{q_B L_x/2} \tag{4.11}$$

Because of the symmetry of the melt pool, the rates of heat flow at the left and right boundaries are the same, as

$$q_L L_y = q_R L_y \tag{4.12}$$

For all simulations in this work, the length and width of the simulation domain are the same. Therefore, the ratio between the rates of heat flow along the x-direction and y-direction can be computed by numerical integration of Eqs. (4.9) and (4.10) as

$$\frac{\dot{Q}_x}{\dot{Q}_y} = \frac{2q_R L_y}{q_B L_x} = \frac{2q_R}{q_B} \approx 2.84 \tag{4.13}$$

Based on Eqs. (4.12) and (4.18), a relationship can be derived as

$$q_L = q_R = 1.42 q_B \tag{4.14}$$

The heat flux at the top boundary caused by the convection and radiation heat transfer is defined as [202]

$$q_T = h(T_l - T_0) + \sigma_{SB}\varepsilon(T_l^4 - T_0^4) \tag{4.15}$$

where σ_{SB} is Stefan–Boltzmann constant and $T_0 = 298$ K is room temperature. Given a constant cooling rate $\dot{T}$, the other three heat fluxes q_B, q_L, and q_R can be calculated based on the energy balance equation [186]

$$L_x L_y \rho c_p \dot{T} = q_T L_x + q_B L_x + q_L L_y + q_R L_y \tag{4.16}$$

After four heat fluxes are obtained, the temperature of the wall T_w can be updated in each iteration based on Eq. (3.30).

4.2.4 Multi-Layer Epitaxial Grain Growth

To simulate multi-layer epitaxial grain growth in SLM, an analytical thermal model is combined with a new marching cell simulation scheme to save the computational cost. The re-melting and solidification processes in multiple scanning passes are simulated. To start the simulation, nuclei with random orientations are planted in a two-dimensional (2D) simulation domain that represents the substrate layer. The initial grains in the substrate layer are first simulated by growing from some random nuclei. After the initial grains are formed, in the first laser scan, the moving laser from left to right partially re-melts the substrate layer to enable cross-layer dendritic growth. Afterwards, feedstock materials are added layer-by-layer and the laser scans through the domain in the SLM process. That is, the nuclei implanted in the substrate domain serve as the initialization of grain growth. To accelerate the process of nuclei implant in the substrate, a larger PFM grid spacing is

applied since the detailed dendritic morphology in the substrate is not the focus here. During the layer-by-layer process, the stochastic nucleation model is applied and nuclei are introduced as the result of impurity and defects. An analytical thermal model is applied to model laser heating.

4.2.4.1 Thermal Model

To save the computational cost, the thermal history for multi-layer grain growth can be approximated by an analytical model. The analytical thermal model is based upon Rosenthal equation [203] for the temperature distribution resulting from a point heat source traversing the surface with a constant scanning speed $V_b = 1.4\ \mathrm{m/s}$. The temperature distribution is assumed to be quasi-stationary and a moving coordinate system centered is used at the origin of the beam. With the beam traversing in the x-direction, the coordinate transformation from the fixed to the moving coordinate system is $d_x = x - V_b t$. The temperature of any given point is calculated as

$$T = T_0 + \frac{AP_b}{2\pi\kappa d} exp\left(-\frac{V_b(d + d_x)}{2\alpha}\right) \tag{4.17}$$

where $T_0 = 300$ K is the start temperature, $A = 0.32$ is the powder absorptivity, $P_b = 200$ W is the laser beam power, κ is the thermal conductivity of the alloy, α is the thermal diffusivity of the alloy, d is the distance from the given point to the heat source.

4.2.4.2 Marching Cell Simulation Scheme

One major challenge of PFM for larger domain simulation is the computational complexity. The building tank in the SLM process has a volume of several cubic decimeters, and the cross-layer melting and heat-affected zone (HAZ) is in the scale of millimeters. However, the actual solidification only occurs inside the melt pool on the scale

of 100 micrometers. In order to improve the computational efficiency for multi-layer grain growth simulation, here we propose a marching-cell simulation scheme, as illustrated in Figure 4.2. The actively simulated 2D domain only has a depth of heat-affected zone in the thermal model instead of modeling the complete building tank. The solidified workpiece below the heat-affected zone is not actively simulated during the layer-by-layer process. The formed grain structures from previous simulation steps are just stored in the computer memory. The PFM model only focuses on a cell that is large enough to include the melt pool, where the actual solidification occurs. After one layer of cells is finished, all cells corresponding to the same one-layer thickness at the bottom of the thermal model domain are stored, the cells above are shifted downwards and an equal number of new cells are initialized at the top. By repeating this procedure, an arbitrary build height with the constant computational effort per layer is possible. When the simulation is completed, the stored cells are merged to show the whole build volume.

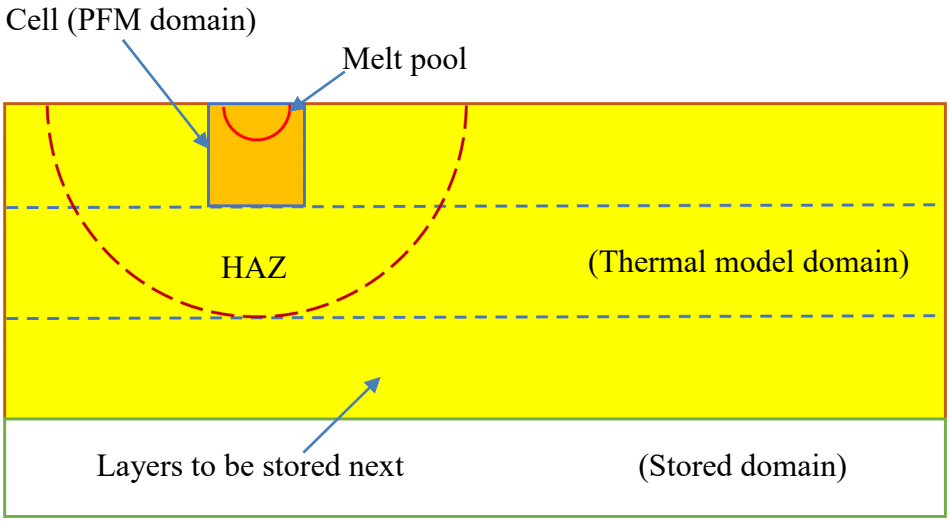


Figure 4.2. The schematic illustration of the proposed marching-cell simulation scheme

4.3 Results and Discussion

In this section, the simulation setup and the simulation results are described. The effects of latent heat and cooling rate on the dendritic growth of AlSi10Mg alloy in the melt pool are studied. The quantitative analyses of thermal history, the time evolution of solid-phase fraction, and composition distribution are also provided.

4.3.1 Computational Setup

4.3.1.1 Single-Layer Dendritic Growth of AlSi10Mg Alloy

The PF-TLBM framework is used to simulate nucleation and dendritic growth of AlSi10Mg alloy in the melt pool during the single-layer SLM process. In AlSi10Mg alloy, the composition of Si is high (9~11 wt%) and the composition of Mg is low (0.2~0.45 wt%). Therefore, it is reasonable to assume the main solute of AlSi10Mg alloy is Si. By using the pseudo-binary approach, the ternary AlSi10Mg alloy is treated as a binary alloy, and the solute is the combination of Si and Mg. The physical properties of AlSi10Mg alloy are listed in Table 4.1 [202,204–209]. For simplification, most properties of AlSi10Mg except the diffusivity of the liquid phase are assumed to be temperature independent during solidification. The dependence of physical properties on temperature needs to be considered to further improve prediction accuracy in future work. The algorithm is implemented in C++ programming language and integrated with the open-source software OpenPhase [183]. PFM and LBM have been implemented in the original OpenPhase. The OpenMP shared-memory parallel programming framework is used to accelerate the computation. There are three main contributions and new features in our implementation. First, LBM has been extended as TLBM so that heat transfer can be simulated. Second, a

probabilistic nucleation model has been introduced in the framework. Finally, a double-mesh scheme has been implemented to improve the computational efficiency of the multiphysics model.

Table 4.1. Physical properties of AlSi10Mg alloy

Physical properties	Value
The melting point of pure Al, T_m [K]	933 [204]
Liquidus temperature, T_l [K]	867 [205]
Solidus temperature, T_s [K]	831 [205]
Liquidus slope, m_l [K/wt%]	−6.6 [204]
Equilibrium partition coefficient, k_e	0.104 [204]
Prefactor of interface energy stiffness, σ_0^* [J/m^2]	0.169 [206]
Interfacial energy stiffness anisotropy, δ	0.27 [206]
Interface mobility, M_ϕ [m^4/(J·s)]	1×10^{-8} [207]
Entropy difference, ΔS [J/(m^3·K)]	1.3×10^{6}
Physical interface width, λ [m]	3×10^{-9} [208]
Prefactor of diffusion coefficient of liquid phase, D_0 [m^2/s]	1.34×10^{-7} [209]
Activation energy, ΔE [J/mol]	3×10^{4} [209]
Kinematic viscosity, ν [m^2/s]	4.87×10^{-7} [205]
Thermal diffusivity, α [m^2/s]	4.5×10^{-5} [205]
Thermal conductivity, κ [W/(m·K)]	110 [205]
Latent heat of fusion, L_H [J/kg]	4.23×10^{5} [205]
Specific heat capacity, c_p [J/(kg · K)]	915 [205]
Density, ρ [kg/m^3]	2670 [205]
Heat transfer coefficient, h [W/(m^2·K)]	82 [202]
Emissivity, ε	0.4 [202]

Since thermal diffusivity is three to four orders of magnitude larger than solute diffusivity, a double-mesh scheme is adopted in simulations to reduce computational cost. A fine grid spacing $dx = 0.2$ μm is used for the PFM simulation, whereas a coarse grid spacing $\Delta x = 50\ dx = 10$ μm is used for the TLBM simulation. Based on the stability analysis, the upper limit of the time step should be $\Delta t \leq min\{dx^2/(4D_l), \Delta x^2/(4\nu), \Delta x^2/(4\alpha)\}$. As a result, the time step $\Delta t = 0.2$ μs is applied. The experimental results show that

the width and depth of the melt pool in SLM of AlSi10Mg are 100 μm [201]. Therefore, the length and width of the two-dimensional simulation domain as the cross-section of the melt pool are chosen to be $L_x = L_y = 500\ dx = 100$ μm. The interface width is $\eta = 5\ dx$, meaning that there are 6 nodes on the interface or boundary layer. The initial composition of the solute is set as $C_0 = 10$ wt% for the whole simulation domain. The initial temperature is $T_0 = 867$ K for the whole simulation domain.

4.3.1.2 Multi-Layer Epitaxial Grain Growth of AlSi10Mg Alloy

To save the computational cost, a coarse grid spacing $dx = 1$ μm and a large time step $\Delta t = 1$ μs are used for the PFM to speed up the simulation of the multi-layer epitaxial grain growth of AlSi10Mg alloy in SLM. Since a coarse mesh is used, the accuracy of the predicted composition field reduces. It is noted that the grain morphology is the focus of multi-layer epitaxial grain growth simulation. The physical properties of AlSi10Mg alloy are shown in Table 4.1. To reduce the side effect caused by the coarse mesh, the interface mobility $M_\phi = 1\times10^{-7}$ m^4/(J·s) of AlSi10Mg alloy is calibrated based on the growth speed reported in the experiment [201] so that the dynamics of grain growth is correct.

The length and width of the two-dimensional simulation domain are chosen to be $L_x = 300\ dx = 300$ μm and $L_y = 140\ dx = 140$ μm, respectively. The interface width is $\eta = 5\ dx$. The initial composition of the solute is set as $C_0 = 10$ wt% for the whole simulation domain. Zero Neumann conditions are set at all boundaries for the phase-field ϕ and composition C. The simulation time is 20 ms. At the beginning of the simulation, 30 spherical nuclei with a radius of 5 μm and random orientations are planted at the bottom of the simulation domain. The grown grains based on the nuclei serve as the base plate of the powder bed. The powder layer thickness is 30 μm. When the top layer of the simulation

domain is fully solidified, all cells corresponding to one-layer thickness at the bottom are stored. The cells above are shifted downwards and an equal number of new cells are initialized on top. The unidirectional scanning strategy is adopted for simplification. Once the laser beam reaches the scanning length of 4 mm, it will move back to the origin of the scanning path.

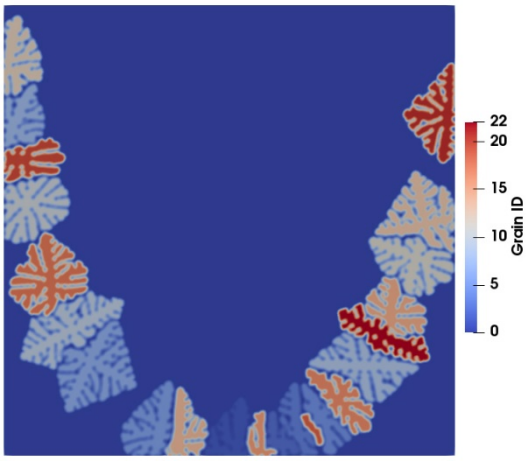
4.3.2 Simulation Results of Single-Layer Dendritic Growth

4.3.2.1 Dendritic Growth without Latent Heat

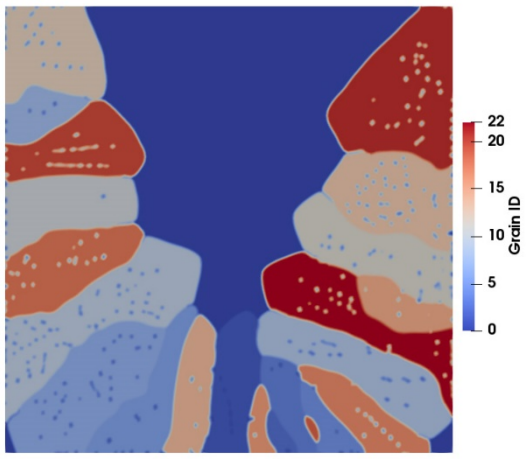
The dendritic growth of AlSi10Mg alloy is first simulated without the release of latent heat for comparison. A constant cooling rate $\dot{T} = 5 \times 10^4$ K/s is used. The simulation results are shown in Figure 4.3. The grain identification (ID) 0 represents the liquid phase, whereas other grain IDs represent solid phases with different orientations. During the rapid solidification process, the columnar dendritic growth dominates in the melt pool, as shown in Figure 4.3.

At the time of 2.8 ms, the columnar dendritic growth pattern is observed, as shown in Figure 4.3(a). The primary arms and secondary arms still can be differentiated. As a result of the anisotropy of interface energy, the primary arms grow faster than secondary arms. Since the release of latent heat is ignored, the secondary arms grow so fast that they quickly merge with each other as shown in Figure 4.3(b). At the time of 11.2 ms, the melt has been completely solidified as shown in Figure 4.3(d). The composition field at 11.2 ms is shown in Figure 4.3(e), where primary arms and secondary arms can be differentiated easily. The microsegregation occurs at the grain boundaries and the small pockets between secondary arms. In Figure 4.3(f), the temperature at the upper center of the melt pool is the highest, which is caused by the setup of heat fluxes at the boundaries. Since the primary

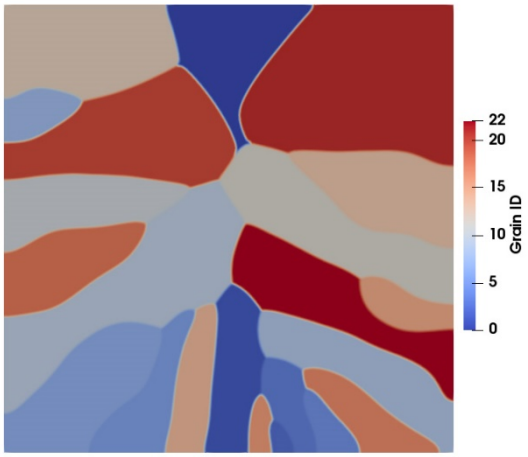
arms aligned with the temperature gradient grow faster than those that do not, this results in the radial distribution pattern of columnar dendrites in the melt pool, as shown in Figure 4.3(d). Since the latent heat is ignored, the temperature decreases so fast that it approaches room temperature at 11.2 ms as shown in Figure 4.3(f). This observation does not agree well with the experimental evidence, which also indicates the significance of considering the release of latent heat.



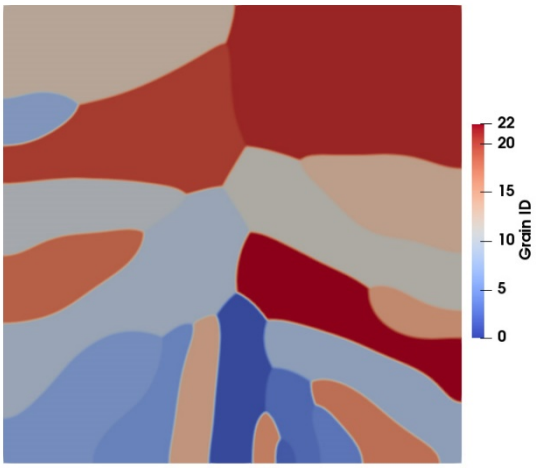
(a) Phase field at 2.8 ms



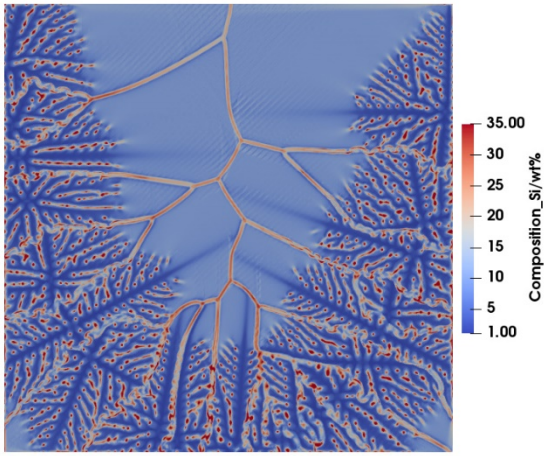
(b) Phase field at 5.6 ms



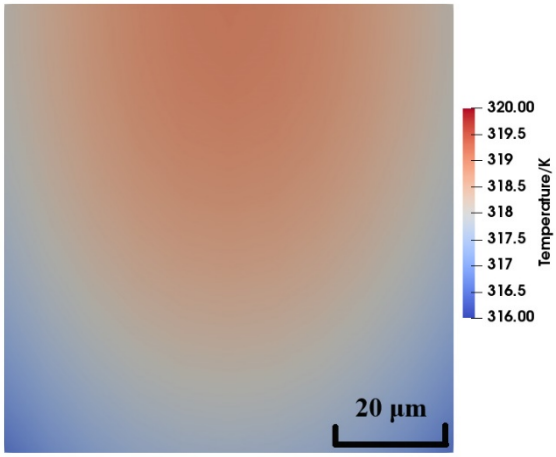
(c) Phase field at 8.4 ms



(d) Phase field at 11.2 ms



(e) Composition field at 11.2 ms

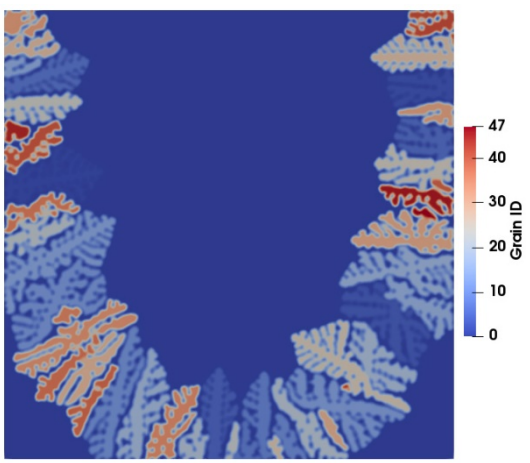


(f) Temperature field at 11.2 ms

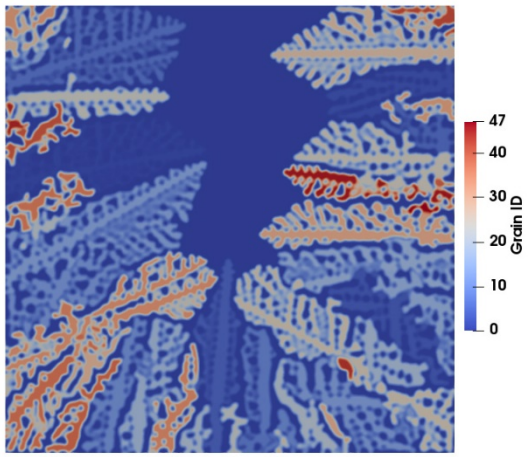
Figure 4.3. Dendritic growth without latent heat.

4.3.2.2 Dendritic Growth with Latent Heat

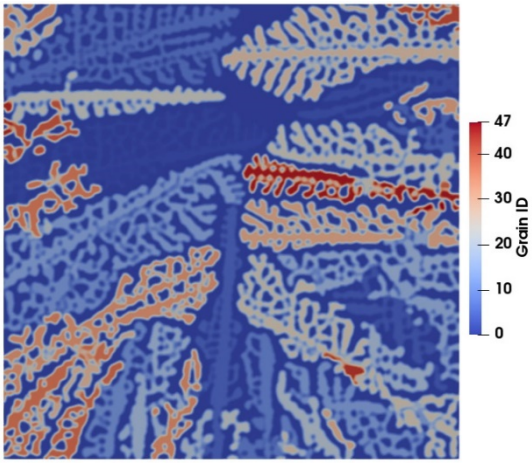
In the second case, the dendritic growth with the release of latent heat is simulated. The cooling rate is also kept as $\dot{T} = 5 \times 10^4$ K/s. Figure 4.4 shows the simulation results. At the time of 4 ms, a clear dendritic growth pattern is shown in Figure 4.4(a), where primary arms and secondary arms can be differentiated easily. When the dendrites continue to grow, the primary arms aligned with the temperature gradient grow faster than those that do not as shown in Figure 4.4(b-d). However, there is still some residual melt between grains. The melt is not completely solidified even at the time of 16 ms, as shown in Figure 4.4(d). The composition field at 16 ms is shown in Figure 4.4(e), where secondary arms can still be observed clearly. The small pockets of the liquid phase at grain boundaries may remain liquid for a long period of time until solid diffusion takes away the remaining solute supersaturation before it is completely solidified. The microsegregation at grain boundaries in the case with latent heat is lower than that in the case without latent heat. Figure 4.4(f) shows the temperature field at 16 ms. The temperature in the case with latent heat is higher than that in the case without latent heat. The maximum temperature (742.5 K) is lower than the temperature of solidus (831 K) in equilibrium. However, the melt is not completely solidified. This is because the actual solidus temperature during nonequilibrium solidification is lower than that in the equilibrium case. This observation agrees with the CALPHAD results in the work of Marola et al. [204]. Based on the above comparison, the inclusion of latent heat is very necessary because it reveals the details of the formation of secondary arms.



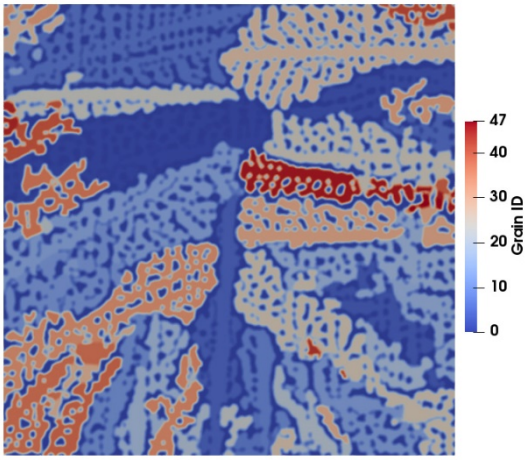
(a) Phase field at 4 ms



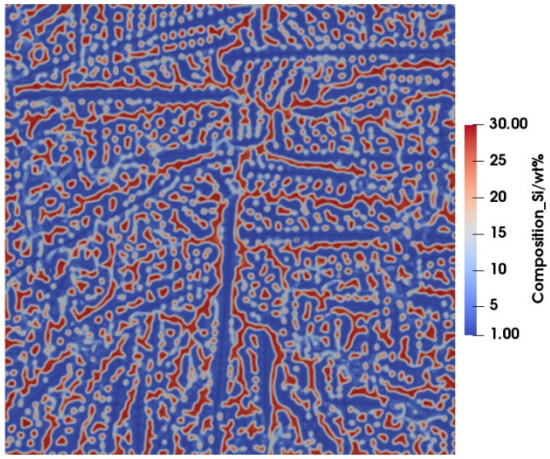
(b) Phase field at 8 ms



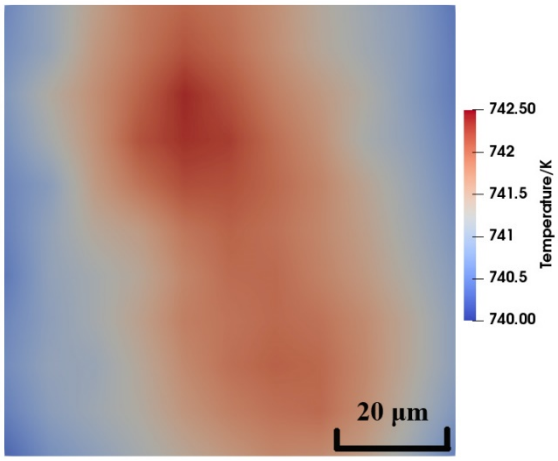
(c) Phase field at 12 ms



(d) Phase field at 16 ms



(e) Composition field at 16 ms



(f) Temperature field at 16 ms

Figure 4.4. Dendritic growth with latent heat.

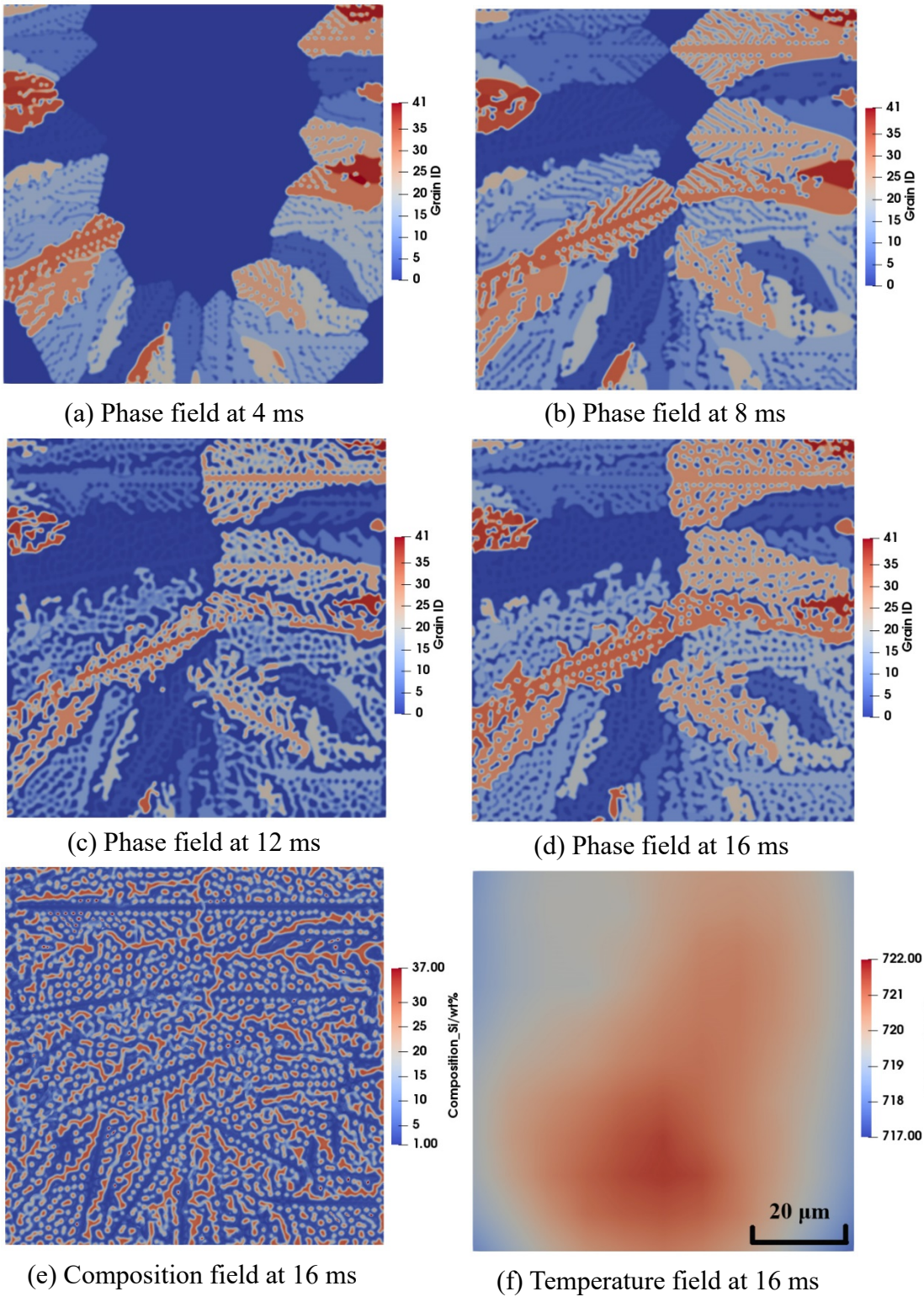
The simulated grain structure in Figure 4.4(d) qualitatively matches the experimental observation by electron backscatter diffraction (EBSD) [201] in Figure 4.5. After one scanning pass, the dendrites at the curved boundary of the melt pool will grow and result in a radial distribution pattern. The cross-section of the AlSi10Mg sample by SLM in the top layer is highlighted with a dashed rectangle. The secondary arm spacing of the simulated dendrite is $\lambda_2 = 1.1$ μm, which is close to the calculated value $\lambda_2 = 0.6$ μm based on the analytical model developed by Bouchard and Kirkaldy [191], as shown in Eq. (3.31). The difference between the predicted and observed secondary arm spacing could be caused by parameter uncertainty and model form uncertainty. The parameter uncertainty can be associated with the interface energy σ, latent heat L_H, solute diffusivity D_l, and local velocity of the interface V_I.

Figure 4.5. Experimental EBSD result of the grain texture in the cross-section of the AlSi10Mg sample produced by SLM (Courtesy of Thijs et al. [201]).

4.3.2.3 The Effect of Cooling Rate

In order to investigate the effect of cooling rate on dendritic morphology and composition distribution, a higher cooling rate $\dot{T} = 1 \times 10^5$ K/s is used. The release of

latent heat is included in this case. The simulation results are presented in Figure 4.6. It is observed that the growth velocity of dendrites increases with the cooling rate. As a result, the secondary arms merge with each other and disappear. The melt is almost completely solidified at the time of 16 ms as shown in Figure 4.6(d). Because of the competitive growth of different grains, a small grain is merged with its neighbor grain, as shown in Figure 4.6. The final grain structure in Figure 4.6(d) is different from those in Figure 4.3(d) and Figure 4.4(d) because the increased cooling rate influences the competitive growth of dendrites. The rising cooling rate also increases the microsegregation at grain boundaries, as shown in Figure 4.6(e). Figure 4.6(f) shows that the temperature is lower than that in the case of the cooling rate $\dot{T} = 5 \times 10^4$ K/s in Figure 4.4(f).



(a) Phase field at 4 ms

(b) Phase field at 8 ms

(c) Phase field at 12 ms

(d) Phase field at 16 ms

(e) Composition field at 16 ms

(f) Temperature field at 16 ms

Figure 4.6. Dendritic growth with latent heat and a high cooling rate.

4.3.2.4 Quantitative Analysis

In this section, a quantitative analysis is conducted to compare the effects of latent heat and cooling rate on temperature field, dendritic morphology, and composition field. Figure 4.7 shows the thermal histories at the location of $x = 50$ μm, $y = 50$ μm for the simulated three situations. When the release of latent heat is not considered, the temperature decreases linearly. When the release of latent heat is considered and the nominal cooling rate is $\dot{T} = 5 \times 10^4$ K/s, the temperature drops quasi-linearly at the beginning of solidification ($0 \leq t < 4$ ms). Since the fraction of phase transition is small at the beginning, the effect of latent heat is not obvious. When $t \geq 4$ ms, the temperature starts to increase until 10 ms and then decreases again. This phenomenon is widely known as recalescence during the solidification of alloys. When the cooling rate is increased to $\dot{T} = 1 \times 10^5$ K/s, the effect of latent heat on the temperature field is reduced. the temperature drops quasi-linearly at the beginning of solidification ($0 \leq t < 4$ ms) and it is lower than that in the case of $\dot{T} = 5 \times 10^4$ K/s. When $t \geq 4$ ms, the temperature starts to increase until 13 ms and then decreases again.

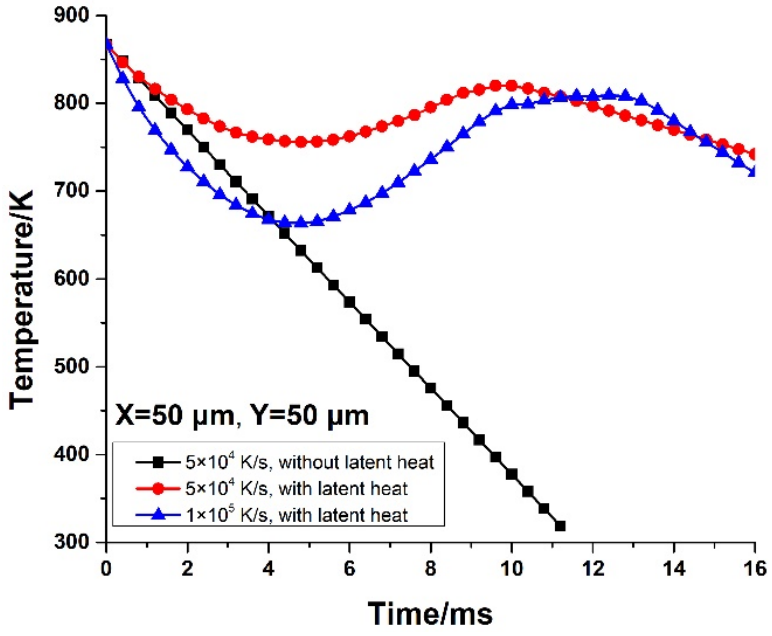


Figure 4.7. Thermal histories at the location of $x = 50$ μm, $y = 50$ μm under different conditions.

The histories of the solid-phase fractions for different cases are shown in Figure 4.8. Here, the solid phase fraction means the total fraction of solid phases in the simulation domain. When solid phase fraction equals to one, the melt is completely solidified. When the latent heat is ignored, the history curve of solid-phase fraction looks like an "S"-shaped logistic sigmoid function, which increases slowly at the beginning, then increases rapidly and reaches plateaus near the end. The solid phase fraction reaches 1.0 at the time of 11 ms, meaning that the liquid-solid phase transition is finished. When the latent heat is considered and the nominal cooling rate is $\dot{T} = 5 \times 10^4$ K/s, the solid fraction increases at the beginning, then decreases and increases again. This means the remelting happens during rapid solidification because of the release of latent heat. The solid phase fraction is 0.72 at the time of 16 ms. It will take some additional time to finish the solidification process because of the release of latent heat and microsegregation in small pockets. When

the cooling rate increases to $\dot{T} = 1 \times 10^5$ K/s, the speed of phase transition increases, and the solid fraction is 0.76 at the time of 16 ms.

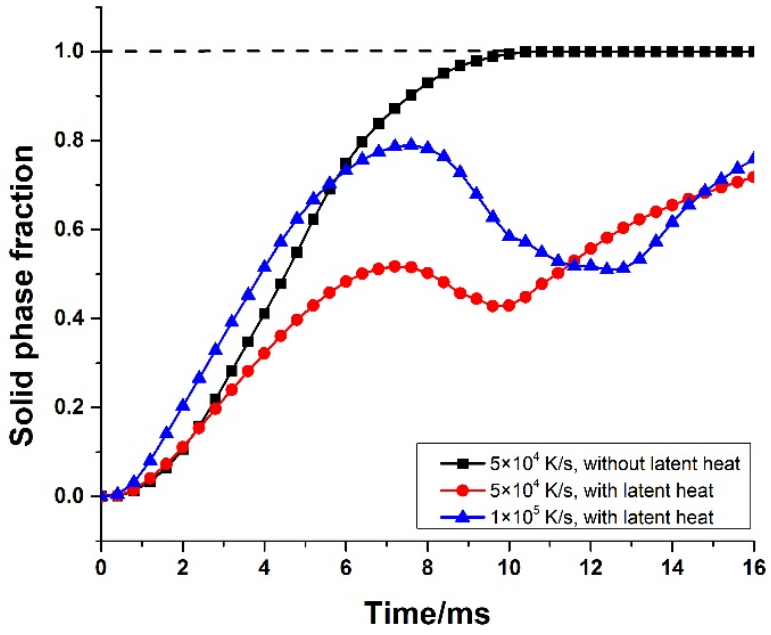


Figure 4.8. Histories of solid-phase fractions under different conditions.

The composition distributions at the location of $y = 50$ μm at the time of 11.2 ms without latent heat and 16 ms with latent heat are shown in Figure 4.9. It is observed that the locations where microsegregation occurs are mostly the same for different cases. Microsegregation can be defined as

$$\chi = \frac{C_{max}}{C_{min}} \tag{4.18}$$

where C_{max} is the maximum of composition, and C_{min} is the minimum of composition. When the latent heat is ignored, the microsegregation is overestimated, which is $\chi = 45.44/1.27 = 35.78$. When the latent heat is considered and the nominal cooling rate is $\dot{T} = 5 \times 10^4$ K/s, there are more secondary arms and peaks of microsegregation. The microsegregation is $\chi = 29.5/1.1 = 26.82$. Therefore, the microsegregation without the

latent heat is overestimated by at least 33% compared to that with the latent heat. When the cooling increases to $\dot{T} = 1 \times 10^5$ K/s, the microsegregation is $\chi = 37.14/1.2 = 30.95$.

Based on the above quantitative analysis, the inclusion of latent heat is important because it provides more realistic kinetics of dendritic growth and reduces overestimated microsegregation. The increased cooling rate increases the speed of phase transition and microsegregation.

In this chapter, all simulations were run using 8 processors with Intel Xeon Processor E5-2680 (2.50 GHz) and memories of 16 GB. It took 41 hours and 31 minutes for simulating 11.2 ms of the case in Section 4.3.2.1, 48 hours and 22 minutes for 16 ms of the case in Section 4.3.2.2, and 48 hours and 42 minutes for 16 ms of the case in Section 4.3.2.3.

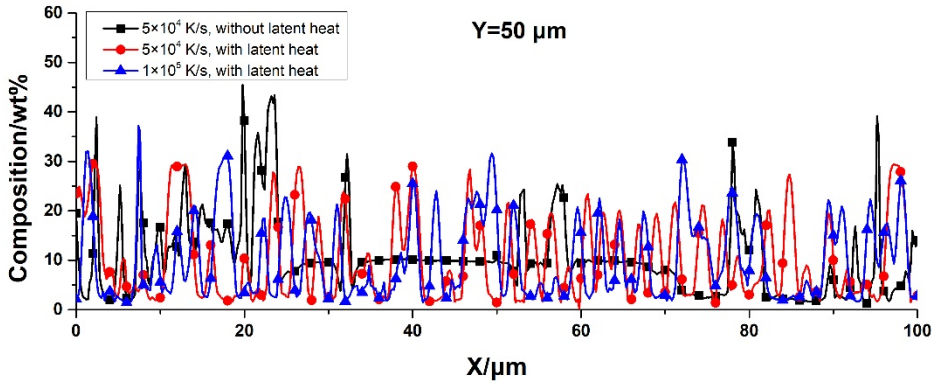
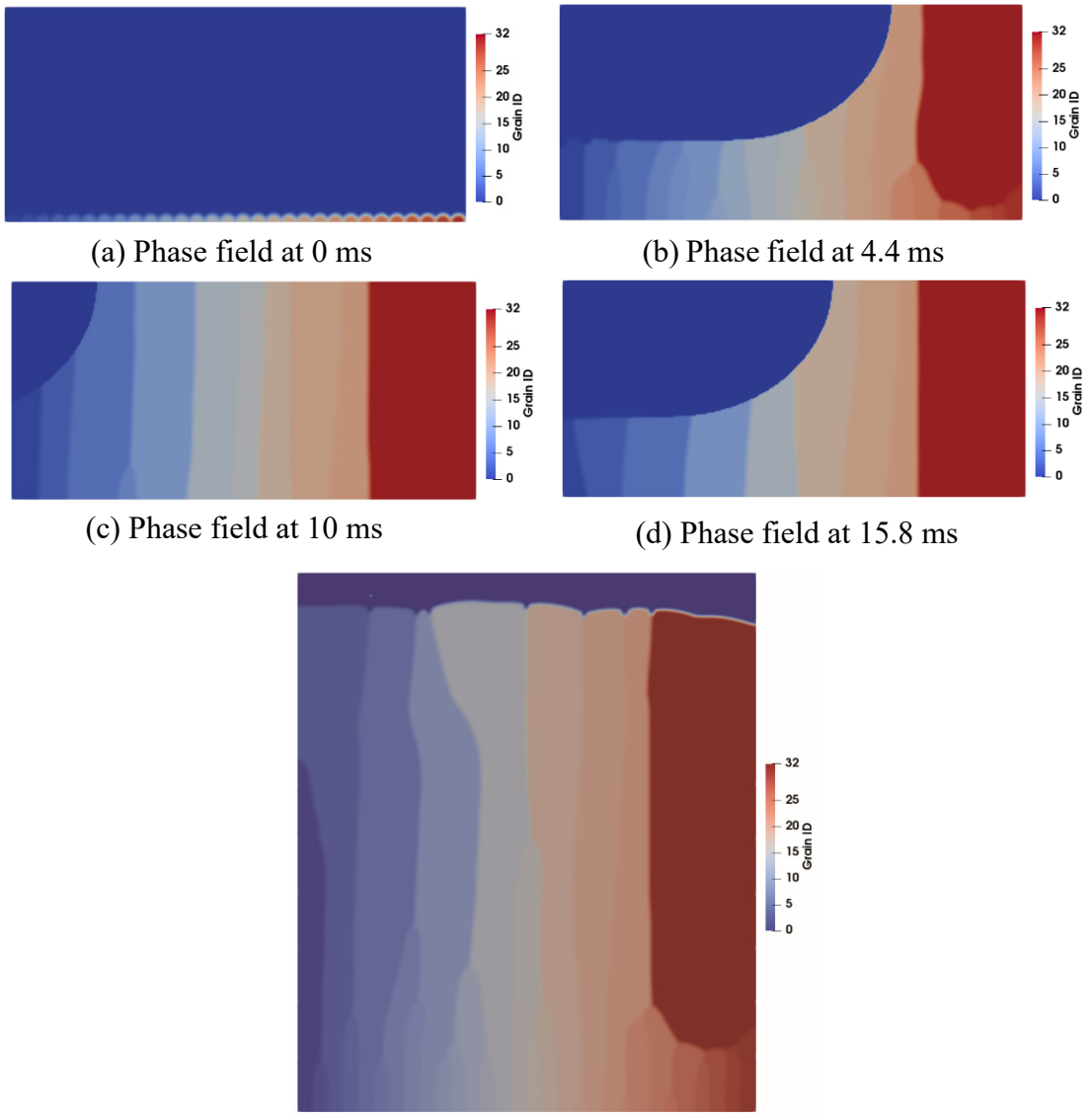


Figure 4.9. Composition distributions at the location of $y = 50$ μm at the time of 11.2 ms without latent heat and 16 ms with latent heat.

4.3.3 Simulation Results of Multi-Layer Epitaxial Grain Growth

The simulation results of multi-layer epitaxial grain growth are shown in Figure 4.10. The melt pool depth is about 88 μm. The powder layer thickness is 30 μm, whereas the

depth of remelting layer is about 58 μm. The simulated layers of the heat-affected zone at different time periods are shown in Figure 4.10(b)-(d), It is observed that the epitaxial columnar dendritic growth dominates during the multi-layer printing processes. When the melt pool front boundary reaches the solidified grains, those grains are remelted and their grain IDs are reset to be zero. When a small nucleus is generated during the multi-layer printing processes based on the nucleation model, the nucleus grows quickly and competes with other columnar dendrites. At the end of the simulated period of 20 ms, the stored cells are merged as the whole build volume, as shown in Figure 4.10(e). The size of the simulation domain in Figure 4.10(e) is 300 μm × 350 μm. It is seen that the primary arm spacings of the grains at the upper portion of the simulation domain are larger than those at the lower portion. This phenomenon is caused by the competitive growth of columnar dendrites. The grains with the preferential <100> growth direction are better aligned with the temperature gradient and outgrow other grains. This result agrees with the observation in Ref. [210].



(a) Phase field at 0 ms

(b) Phase field at 4.4 ms

(c) Phase field at 10 ms

(d) Phase field at 15.8 ms

(e) Merged phase field at 20 ms

Figure 4.10. Multi-layer epitaxial grain growth in SLM.

4.4 Conclusions

In this chapter, a nucleation model is introduced into the recently developed PF-TLBM framework to consider the heterogeneous nucleation process at the solid-liquid interface. This mesoscale multiphysics model is used to simulate the nucleation and dendritic growth of AlSi10Mg alloy in the SLM melt pool. A new method is proposed to

compute heat fluxes for a 2D small melt pool in order to approximate the actual non-isothermal temperature field in SLM. The simultaneous considerations of solute transport, heat transfer, nucleation, and dendritic growth are necessary to understand complex rapid solidification in SLM. By considering the release of latent heat, the model can predict the temperature field, composition distribution, and dendritic morphology with more details than models without latent heat. The recalescence occurs when the latent heat is considered. The qualitative and quantitative analyses show that the inclusion of latent heat is necessary because it reveals the details of the formation of secondary arms, reduces the overestimation of microsegregation, and provides more realistic kinetics of dendritic growth. A higher cooling rate results in faster liquid-solid phase transition and higher microsegregation at grain boundaries.

The PF-TLBM model is also extended to predict the multi-layer epitaxial grain growth in the complex heating and cooling environment in SLM. To save the computational cost, the Rosenthal equation method is used to predict the thermal history of the melt pool. A marching cell simulation scheme is used to further reduce the computational cost. It is demonstrated that the PF-TLBM model is capable of simulating multi-layer printing processes in SLM. In future work, the PF-TLBM model will be used to simulate the grain growth within the whole workpiece with the size of decimeters during the SLM process. A moving Gaussian heat source will be introduced in the thermal lattice Boltzmann method to predict a more accurate thermal history.

Further work is also needed to improve the fidelity, accuracy, and efficiency of the PF-TLBM model. For instance, the surface tension source term could be introduced into the TLBM so that the effect of Marangoni flow on dendritic growth can be investigated.

The motion of grains can be enabled as well. Furthermore, the empirical nucleation parameters need to be determined and calibrated based on experimental measurements, first-principles calculations, or atomistic simulations. The determination of the nucleation energy barrier or nucleation rate can help to predict a more realistic microstructure. The dependence of physical properties of AlSi10Mg alloy on temperature needs to be considered to improve prediction accuracy, which can be found in the work of Wei et al. [202]. The model form and parameter uncertainties associated with the developed model should be quantified to provide more confidence in the prediction. A parallelized 3D PF-TLBM is needed to simulate the dendritic growth in the melt pool with more details. Both the PFM and the TLBM can be modified for parallel computation without much difficulty.

The proposed mesoscale multi-physics PF-TLBM model is a key component in a multiscale simulation framework for SLM processes, which involves multiple and complex physical phenomena. It predicts the microstructure evolution in the SLM process at a reasonable time scale. The predicted microstructure is the central hinge of the P-S-P relationship, which needs to be investigated for process design and optimization. Classical continuum simulation methods cannot provide fine-grained material phase and composition distribution, whereas atomistic models cannot simulate the time scales which are long enough for manufacturing processes.

CHAPTER 5. MULTI-FIDELITY PHYSICS-CONSTRAINED NEURAL NETWORK AND ITS APPLICATION IN MATERIALS MODELING

5.1 Introduction

Training machine learning tools such as neural networks requires the availability of sizable data, which can be difficult for engineering and scientific applications where experiments or simulations are expensive. In this chapter, multi-fidelity physics-constrained neural networks (MF-PCNN) is proposed to reduce the required amount of training data, where physical knowledge is applied to constrain neural networks, and multi-fidelity networks are constructed to improve training efficiency.

Physics-constrained neural networks (PCNNs) can be constructed to approximate the solutions of PDEs to predict the dynamic properties of systems. Some solutions from the simulations serve as the training data. The prior knowledge of PDEs, including the initial and boundary conditions, are applied to guide the training process of PCNNs with reduced searching space. The multi-fidelity concept is introduced here to further reduce the cost to obtain training data. By combining a low-fidelity physics-constrained neural network (LF-PCNN) and a high-fidelity physics-constrained neural network (HF-PCNN), a multi-fidelity physics-constrained neural network (MF-PCNN) can be created with a lower training cost and higher prediction accuracy than traditional ANNs. The LF-PCNN is trained with low-fidelity simulation results, whereas the HF-PCNN is trained from high-fidelity simulations. Then another ANN called discrepancy artificial neural network (DANN) is trained based on the difference between the LF-PCNN and HF-PCNN

predictions. The MF-PCNN is constructed by combining the predictions from the LF-PCNN and DANN. The advantage of the MF-PCNN is that the overall computational cost to obtain training data can be reduced by using the data with different fidelities. In this chapter, three examples are used to demonstrate the MF-PCNN framework. The first example is the prediction of the temperature field in a heat transfer problem. The second example is the prediction of the phase field in a phase transition process. The third example is to predict the dendritic growth during solidification based on multiphysics simulations. It is shown that the MF-PCNN can be constructed with a limited amount of simulation data but achieve a good accuracy of prediction.

In the remainder of this chapter, the training of PCNNs, the construction of MF-PCNNs, and the setup of the computational scheme are described in Section 5.2. The computational results of the examples are shown in Section 5.3.

5.2 Methodology

In MF-PCNNs, the training data for LF-PCNNs and HF-PCNNs can be obtained from the analytical or numerical solutions of PDEs, e.g. from the finite-element method (FEM). During the training, the prior knowledge about the form of PDEs or boundary values is added as the regularization terms in the loss function. The knowledge constraints guide the searching direction for optimization. The MF-PCNN is constructed based on the information from the LF-PCNN as well as the additional information that the HF-PCNN provides. The cost of obtaining high-fidelity information is higher than that of low-fidelity one. Therefore, the allocation of computational resources between high- and low-fidelity simulations can help reduce the overall training cost.

5.2.1 Training of PCNNs

Generally, a wide range of physical phenomena and dynamics can be described by PDEs, including heat transfer, advection-diffusion process, fluid dynamics, and others. Let us consider a time-dependent parametrized PDE with the general form

$$P\left(u, \frac{\partial u}{\partial t}, \frac{\partial u}{\partial \mathbf{x}}, \frac{\partial^2 u}{\partial t^2}, \frac{\partial^2 u}{\partial \mathbf{x}^2}, \ldots\right) = f(t, \mathbf{x}), \qquad t \in [0, T], \qquad \mathbf{x} \in \Omega \tag{5.1}$$

where $u(t, \mathbf{x})$ is the hidden solution to be found, $f(t, \mathbf{x})$ is a source or sink term, t is the time, $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ is the spatial vector, and $\Omega \in \mathbb{R}^n$ denotes the definition domain. This general PDE is subject to initial conditions (ICs)

$$I\left(u, \frac{\partial u}{\partial t}, \frac{\partial^2 u}{\partial t^2}, \ldots\right) = g(\mathbf{x}), \qquad t = 0, \qquad \mathbf{x} \in \Omega \tag{5.2}$$

and boundary conditions (BCs)

$$S\left(u, \frac{\partial u}{\partial t}, \frac{\partial u}{\partial \mathbf{x}}, \frac{\partial^2 u}{\partial t^2}, \frac{\partial^2 u}{\partial \mathbf{x}^2}, \ldots\right) = h(t, \mathbf{x}), \qquad t \in [0, T], \qquad \mathbf{x} \in \partial\Omega \tag{5.3}$$

where $\partial\Omega$ is the boundary of the definition domain. A more compact form of the above initial-boundary value problem can be written as

$$\mathbf{D}[u(t, \mathbf{x})] = f(t, \mathbf{x}), \qquad t \in [0, T], \qquad \mathbf{x} \in \Omega \tag{5.4}$$

$$\mathbf{\Lambda}[u(0, \mathbf{x})] = g(\mathbf{x}), \qquad t = 0, \qquad \mathbf{x} \in \Omega \tag{5.5}$$

$$\mathbf{\Gamma}[u(t, \mathbf{x})] = h(t, \mathbf{x}), \qquad t \in [0, T], \qquad \mathbf{x} \in \partial\Omega \tag{5.6}$$

where $\mathbf{D}[\cdot]$, $\mathbf{\Lambda}[\cdot]$, and $\mathbf{\Gamma}[\cdot]$ are differential operators. For example, the three-dimensional (3D) heat equation without the source term corresponds to $\mathbf{D}[u(t, \mathbf{x})] = u_t - \alpha\left(u_{xx} + u_{yy} + u_{zz}\right) = 0$, where α is the thermal diffusivity, and the subscripts represent the partial derivative with respect to either time or space.

In this chapter, the MLP architecture is used as a demonstration, which includes one input layer $(t, \mathbf{x})$, multiple hidden layers, and one output layer $U(t, \mathbf{x})$ to approximate the true solution $u(t, \mathbf{x})$. The neurons are connected with those in the neighbor layers, and the weights represent the strength of connections. The output from the hidden layer to the following layer is calculated as

$$y_i = \varphi\left(\sum w_{ij}\theta_j + b_i\right) \tag{5.7}$$

where w_{ij} is the weight of the connection between neuron j in the previous layer and neuron i in the current layer, θ_j is the j-th input value from the previous layer, and b_i is the bias for the neuron i in the current layer. φ is a nonlinear activation function, which can be sigmoid, tanh, rectified linear unit, or others.

The weights of a PCNN can be learned by minimizing the mean squared loss or total cost function

$$E = \lambda_T E_T + \lambda_P E_P + \lambda_I E_I + \lambda_s E_s \tag{5.8}$$

where $E_T = \frac{1}{N_T}\sum_{i=1}^{N_T}\left|U(t_i^T, \mathbf{x}_i^T) - T(t_i^T, \mathbf{x}_i^T)\right|^2$ is the loss caused by the discrepancy between the training data $T(\cdot)$ and the PCNN model prediction $U(\cdot)$, $\left\{t_i^{(\cdot)}, \mathbf{x}_i^{(\cdot)}\right\}$ denotes the sampling points in the defined domain, and $N_{(\cdot)}$ denotes the number of sampling points. Similarly, $E_P = \frac{1}{N_P}\sum_{i=1}^{N_P}\left|\mathbf{D}[U(t_i^P, \mathbf{x}_i^P)] - f(t_i^P, \mathbf{x}_i^P)\right|^2$, $E_I = \frac{1}{N_I}\sum_{i=1}^{N_I}\left|\mathbf{\Lambda}[U(t_i^I, \mathbf{x}_i^I)] - g(\mathbf{x}_i^I)\right|^2$, and $E_S = \frac{1}{N_S}\sum_{i=1}^{N_S}\left|\mathbf{\Gamma}\left[U\left(t_i^S, \mathbf{x}_i^S\right)\right] - h\left(t_i^S, \mathbf{x}_i^S\right)\right|^2$ are the losses caused by the violations of the model, initial conditions, and boundary conditions as the physical constraints from Eqs. (5.4)-(5.6). The constraint on the weights of different losses is given as

$$\lambda_T + \lambda_P + \lambda_I + \lambda_s = 1 \tag{5.9}$$

The relative importance of prior knowledge can be adjusted by changing the weights of physical constraints λ_P, λ_I and λ_S. If the total loss function only includes the training loss E_T, then this is the traditional pure data-driven ANN to solve the initial-boundary value problem. It will be shown in Section 5.3.1 that assigning different weights will affect the speed of training. An adaptive scheme to assign the weights is proposed here so that the overall loss is calculated as

$$E = \frac{E_T^2 + E_P^2 + E_I^2 + E_S^2}{E_T + E_P + E_I + E_s} \tag{5.10}$$

for each iteration during the training process. That is, the weights are proportional to individual losses from data and physical constraints respectively. By adding physical losses E_P, E_I and E_S as the regularization terms, the prior physical knowledge can help to reduce the size of searching space and provide guidance for the searching directions in training.

5.2.2 Construction of MF-PCNNs

The LF-PCNN and HF-PCNN must be trained first before the MF-PCNN is constructed. In this work, the fidelities are determined by the resolutions of FEM simulations given the same density of physical constraints. To be more specific, low-fidelity simulations are used to construct the LF-PCNN during a long time period $t \in [0, T]$, whereas high-resolution simulations are applied for the HF-PCNN during a short time period $t \in [0, T_0]$ $(T_0 < T)$.

After the LF-PCNN and HF-PCNN are trained, the difference between the predictions of the LF-PCNN $U_L(t, \mathbf{x})$ and HF-PCNN $U_H(t, \mathbf{x})$ is calculated as

$$\delta(t,\mathbf{x}) = U_H(t,\mathbf{x}) - U_L(t,\mathbf{x}), \qquad t \in [0,T_0], \qquad \mathbf{x} \in \Omega \tag{5.11}$$

Then the DANN is constructed to predict the discrepancy between the LF-PCNN and HF-PCNN, denoted as $U_\delta(t,\mathbf{x})$, during a longer time period $t \in [0,T]$. The weights of the DANN can be learned by using the observed discrepancy $\delta(t,\mathbf{x})$ as the training data to minimize the mean squared error loss

$$E_\delta = \frac{1}{N_\delta}\sum_{i=1}^{N_\delta} |U_\delta(t_i,\mathbf{x}_i) - \delta(t_i,\mathbf{x}_i)|^2, \qquad t \in [0,T_0], \qquad \mathbf{x} \in \Omega \tag{5.12}$$

where N_δ is the number of sampling points for the DANN. It is assumed that the evolution of the difference between the LF-PCNN and HF-PCNN during a longer time period $t \in [0,T]$ can be predicted by the DANN using the observed discrepancy $\delta(t,\mathbf{x})$ as the training data during the short time period $t \in [0,T_0]$. Then the MF-PCNN is a combination of the LF-PCNN and DANN. The prediction from the MF-PCNN during the time period $t \in [0,T]$ is given by

$$U_M(t,\mathbf{x}) = U_L(t,\mathbf{x}) + U_\delta(t,\mathbf{x}), \qquad t \in [0,T], \qquad \mathbf{x} \in \Omega \tag{5.13}$$

5.2.3 Experimental Setup of the Proposed MF-PCNN

The construction and training of the MF-PCNN are accomplished by using Tensorflow [211], which is an open-source Python library for machine learning. The partial derivatives of the ANNs are calculated based on the chain rules using the automatic differentiation [212]. Automatic differentiation is different from numerical differentiation such as the method of finite difference. By applying the chain rules repeatedly, the derivatives of arbitrary order can be computed automatically, and accurately to working precision.

Three examples are applied to demonstrate the proposed MF-PCNN framework. The first example is a heat transfer problem where the evolution of the two-dimensional (2D) temperature distribution is modeled with the heat equation. The heat transfer example is used to demonstrate the effectiveness of the proposed adaptive weighting schemes of the total loss function. The second example is the phase transition problem where the evolution of the 2D phase field is modeled with the Allen-Cahn equation. The phase transition example is utilized to demonstrate the MF-PCNN framework. The third example is the dendritic growth during solidification where heat transfer and phase transition are tightly coupled. The purpose is to demonstrate the applicability of the proposed MF-PCNN framework for complex multiphysics problems in materials design.

The details of the computational setup for different ML models in the heat transfer, phase transition, and dendritic growth example are listed in Table 5.1, Table 5.2, and Table 5.3, respectively. The ANNs, LF-PCNNs, and HF-PCNNs have the same structure of 30-20-30-20. That is, each of the networks has 4 layers. There are 30 neurons in the first and third layer, and 20 neurons in the second and last layer. The neural network architecture was identified by conducting some simple sensitivity studies. Finding the optimal architecture requires some systematic searching and sampling procedures, which can be done in future work. The structures of the tested DANNs are 5-5-5-5 and 10-10-10-10, which are simpler in order to avoid overfitting. For comparison purposes, two GP surrogate models with the RBF kernel are also constructed to predict the difference between the LF-PCNN and HF-PCNN. Only one run of the optimizer is performed from the RBF kernel's initial parameters. The noise level of the RBF kernel is set to be *alpha* = 0.1 to prevent overfitting. The hyperbolic tangent (tanh) function is used as the activation function. All

of the loss functions in neural networks are minimized by using a gradient-based optimization algorithm called Adam [213].

The training data for the ANNs, LF-PCNNs, and HF-PCNNs come from the FEM solutions of *COMSOL*, whereas the training data for the DANNs and GPs come from the observed discrepancy between the predictions of the LF-PCNNs and HF-PCNNs during the short time period $t \in [0, T_0]$. All FEM simulations are finished in less than one minute for these 2D problems. The training data and physical constraints for the first two examples are sampled uniformly in both temporal and spatial dimensions. Random sampling is used to obtain the LF and HF training data for the dendritic growth example.

Notice that in a multi-fidelity modeling framework, the LF data can come from LF models with lower resolutions, reduced-order models, models with simplified geometry, and others where the computational cost is lower than HF models. In this work, LF data were taken from the FEM simulations with low resolutions. The proposed MF-PCNN does not require a fixed hierarchy of fidelities over the whole range of input parameters. That is, the LF and HF data do not form a nested hierarchy for both spatial and time domains.

5.2.3.1 Example 1: Heat Transfer

The evolution of temperature distributions can be modeled by parabolic PDEs. The heat equation describes the diffusion process of energy, which is important in modeling microstructure evolution during phase transition. The 2D heat equation with zero Neumann boundary conditions used in this example is

$$\begin{cases} u_t - 0.01\left(u_{xx} + u_{yy}\right) = 0, \;\; t, x, y \in [0,1], \\ u(0, x, y) = 0.5[sin(4\pi x) + sin(4\pi y)], \\ u_x(t, 0, y) = 0, \\ u_x(t, 1, y) = 0, \\ u_y(t, x, 0) = 0, \\ u_y(t, x, 1) = 0. \end{cases} \tag{5.14}$$

where u is the 2D temperature field.

The goal of training a neural network is to ensure the prediction $U(t, x, y)$ from the neural network can approximate the true solution $u(t, x, y)$ from FEM simulations with the desired accuracy. In the total loss function defined by Eq. (5.8), the training loss here is given by

$$E_T = \frac{1}{N_T}\sum_{i=1}^{N_T}\left|U(t_i^T, x_i^T, y_i^T) - T(t_i^T, x_i^T, y_i^T)\right|^2 \tag{5.15}$$

The physical loss is

$$E_P = \frac{1}{N_P}\sum_{i=1}^{N_P}\left|U_t(t_i^P, x_i^P, y_i^P) - 0.01\begin{bmatrix} U_{xx}(t_i^P, x_i^P, y_i^P) \\ +U_{yy}(t_i^P, x_i^P, y_i^P) \end{bmatrix}\right|^2 \tag{5.16}$$

The initial loss is given by

$$E_I = \frac{1}{N_I}\sum_{i=1}^{N_I}\left|U(0, x_i^I, y_i^I) - 0.5[sin(4\pi x_i^I) + sin(4\pi y_i^I)]\right|^2 \tag{5.17}$$

The boundary loss is

$$E_S = \frac{1}{N_S}\sum_{i=1}^{N_S}\begin{bmatrix} \left|U_x\left(t_i^S, 0, y_i^S\right)\right|^2 + \left|U_x\left(t_i^S, 1, y_i^S\right)\right|^2 \\ +\left|U_y\left(t_i^S, x_i^S, 0\right)\right|^2 + \left|U_y\left(t_i^S, x_i^S, 1\right)\right|^2 \end{bmatrix} \tag{5.18}$$

As shown in Table 5.1, the amount of training data for the heat transfer example is $N_T = 21 \times 6 \times 6$, which means that there are 21 sampling points in the temporal dimension or time period, 6 sampling points in the x-direction, and 6 sampling points in

the y-direction of the spatial domain. The simulation domain is $x, y \in [0,1]$ and time period is $t \in [0,1]$. The training data for the heat transfer example are from the FEM simulation where the grid spacing is $\Delta x = 0.2$ and the time step is $\Delta t = 0.05$. For the PCNNs in the heat transfer example, the number of physical constraints is $41 \times 11 \times 11 = 4961$. The grid spacing is $\Delta x = 0.1$ and the time step is $\Delta t = 0.025$ for physical constraints. The numbers of sampling points corresponding to the physical loss, initial loss, and boundary loss are $N_P = 3240$, $N_I = 121$ and $N_S = 1600$ respectively, which sum up to 4961. In the heat transfer example, three different weighting schemes (PCNN1, PCNN2, and PCNN3) are compared. The training of ANN and PCNNs stops when the total loss E is lower than a threshold value of 0.01.

Table 5.1. The setup for different ML models in the heat transfer example

ML model	Structure	Amount of training data ($t \times x \times y$)	Number of physical constraints ($t \times x \times y$)	Time period/s
ANN	30-20-30-20	21×6×6	0	[0, 1]
PCNN1, PCNN2, PCNN3	30-20-30-20	21×6×6	41×11×11	[0, 1]

5.2.3.2 Example 2: Phase Transition

The second example is the Allen-Cahn equation, which is a nonlinear reaction-diffusion equation that describes the process of phase transition such as grain growth and spinodal decomposition. It has become the foundational model for interface diffusion in the phase-field method, which is developed to study phase transitions and interfacial

dynamics in materials science. The Allen-Cahn equation with periodic boundary conditions in this example is

$$\begin{cases} u_t - 0.001\left(u_{xx} + u_{yy}\right) = u - u^3, \ \ t, x, y \in [0,1], \\ u(0, x, y) = 0.5[sin(4\pi x) + sin(4\pi y)], \\ u(t, 0, y) = u(t, 1, y), \\ u_x(t, 0, y) = u_x(t, 1, y), \\ u(t, x, 0) = u(t, x, 1), \\ u_y(t, x, 0) = u_y(t, x, 1). \end{cases} \tag{5.19}$$

where a non-conserved variable u is the order parameter or phase field.

Based on the results of the previous example, the weights of the physical constraints are adaptively adjusted as in Eq. (5.10). The training loss is given by

$$E_T = \frac{1}{N_T}\sum_{i=1}^{N_T}\left|U(t_i^T, x_i^T, y_i^T) - T(t_i^T, x_i^T, y_i^T)\right|^2 \tag{5.20}$$

The physical loss is given by

$$E_P = \frac{1}{N_P}\sum_{i=1}^{N_P}\left|\begin{matrix} U_t(t_i^P, x_i^P, y_i^P) - 0.001\left[U_{xx}(t_i^P, x_i^P, y_i^P) + U_{yy}(t_i^P, x_i^P, y_i^P)\right] \\ -U(t_i^P, x_i^P, y_i^P) + U^3(t_i^P, x_i^P, y_i^P) \end{matrix}\right|^2 \tag{5.21}$$

The initial loss is given by

$$E_I = \frac{1}{N_I}\sum_{i=1}^{N_I}\left|U(0, x_i^I, y_i^I) - 0.5[sin(4\pi x_i^I) + sin(4\pi y_i^I)]\right|^2 \tag{5.22}$$

The boundary loss is given by

$$E_S = \frac{1}{N_S}\sum_{i=1}^{N_S}\left[\begin{matrix} \left|U\left(t_i^S, 0, y_i^S\right) - U\left(t_i^S, 1, y_i^S\right)\right|^2 + \left|U_x\left(t_i^S, 0, y_i^S\right) - U_x\left(t_i^S, 1, y_i^S\right)\right|^2 \\ +\left|U\left(t_i^S, x_i^S, 0\right) - U\left(t_i^S, x_i^S, 1\right)\right|^2 + \left|U_y\left(t_i^S, x_i^S, 0\right) - U_y\left(t_i^S, x_i^S, 1\right)\right|^2 \end{matrix}\right] \tag{5.23}$$

In the phase transition example, two HF-PCNNs (HF-PCNN1 and HF-PCNN2) are trained as shown in Table 5.2. The simulation domain is $x, y \in [0,1]$ and time period is $t \in [0,1]$. The training data for the ANN and LF-PCNN are from the LF simulation where the grid spacing is $\Delta x = 0.2$ and the time step is $\Delta t = 0.05$. The training data for the HF-PCNNs are from the HF simulation where the grid spacing is $\Delta x = 0.05$ and the time step is $\Delta t = 0.025$. Therefore, the training data for the HF-PCNNs is more accurate and expensive than for the ANN and LF-PCNN. HF-PCNN1 is trained during the time period $t \in [0, 0.2]$, whereas the HF-PCNN2 is trained during two time periods, $t \in [0, 0.2]$ and $t \in [0.8, 1]$. Therefore, the amount of training data and the number of physical constraints for the HF-PCNN2 are twice of those for the HF-PCNN1. The observed discrepancy between the predictions of the LF-PCNN and HF-PCNN1 serves as the training data for the DANN1, DANN2, and GP1. Here, the amount of training data for the DANN1, DANN2, and GP1 is $9 \times 26 \times 26$, which means that the grid spacing is $\Delta x = 0.04$ and the time step is $\Delta t = 0.025$. The difference between the HF and LF simulation data is not used as the training data for the discrepancy function because they may not be measured at the same location or time step. That is, since the data are not in a nested hierarchy, the observed discrepancy is obtained from the neural network predictions. Similarly, the observed discrepancy between the predictions of the LF-PCNN and HF-PCNN2 serves as the training data for the DANN3, DANN4, and GP2. In this work, the difference between the HF simulation data and the prediction of LF-PCNN is not used as the training data for the discrepancy function. This is because more accurate predictions can be obtained by adding physical constraints into the training of HF-PCNNs. Besides, the trained HF-PCNNs can provide more training data to the DANNs or GPs so that the constructed MF-PCNNs are

more general and have better prediction accuracy. For ANNs, LF-PCNNs, and HF-PCNNs, the training of a neural network stops when the total loss E is lower than a threshold value of 0.01. Similarly, the training of a DANN stops when the loss function E_δ is below 0.01.

Table 5.2. The setup for different ML models in the phase transition example

ML model	Structure	Amount of training data ($t \times x \times y$)	Number of physical constraints ($t \times x \times y$)	Time period/s
ANN	30-20-30-20	21×6×6	0	[0, 1]
LF-PCNN	30-20-30-20	21×6×6	21×11×11	[0, 1]
HF-PCNN1	30-20-30-20	9×21×21	5×11×11	[0, 0.2]
HF-PCNN2	30-20-30-20	18×21×21	10×11×11	[0, 0.2], [0.8, 1]
DANN1	5-5-5-5	9×26×26	0	[0, 0.2]
DANN2	10-10-10-10	9×26×26	0	[0, 0.2]
DANN3	5-5-5-5	18×26×26	0	[0, 0.2], [0.8, 1]
DANN4	10-10-10-10	18×26×26	0	[0, 0.2], [0.8, 1]
GP1	RBF kernel	9×26×26	0	[0, 0.2]
GP2	RBF kernel	18×26×26	0	[0, 0.2], [0.8, 1]

5.2.3.3 Example 3: Dendritic Growth

The third example is dendritic growth during solidification, where heat transfer and phase transition are coupled with each other. In this multiphysics problem, the heat equation and the Allen-Cahn equation need to be solved simultaneously to predict the evolution of dendritic growth. The coupled PDEs and corresponding boundary conditions for the dendritic growth example are

$$\begin{cases} 0.001p_t - 0.0001\left(p_{xx} + p_{yy}\right) = p(1-p)\left[p - 0.5 + \frac{0.9}{\pi}\tan^{-1}(10q_e - 10q)\right] \\ p(0,x,y) = exp\left(-\frac{x^2+y^2}{0.04}\right) \\ p_x(t,-2.5,y) = p_x(t,2.5,y) = p_y(t,x,-2.5) = p_y(t,x,2.5) = 0 \\ 0.001\left(q_t - q_{xx} - q_{yy}\right) = 0.001Kp_t \\ q(0,x,y) = 0 \\ q_x(t,-2.5,y) = q_x(t,2.5,y) = q_y(t,x,-2.5) = q_y(t,x,2.5) = 0 \\ t \in [0,1], \qquad x,y \in [-2.5,2.5] \end{cases} \tag{5.24}$$

where p is the phase field and q is the temperature field. The liquidus temperature q_e and latent heat K are materials dependent and are the design variables in materials design. That is, we need to find the best material compositions corresponding to the optimal values of these two variables so that the desirable dendritic growth behavior can be obtained. The evolutions of dendrites are sensitive to the liquidus temperature and latent heat of the materials [214]. The dendritic growth (which can be quantified by the growth speed, symmetry, secondary arm spacing, and other descriptors) affects the final mechanical, thermal, and other properties of solid crystals. Therefore, efficient ML predictions of dendritic growths help establish the process-structure-property relationship for materials design. In this simplified example, all variables in the coupled PDEs are dimensionless. A scaling factor of 0.001 is multiplied at both sides of the heat equation so that the magnitudes of the Allen-Cahn equation and the heat equation are on the same scale. The normalization procedure ensures the fast convergence of PCNNs.

In order to apply the proposed MF-PCNN framework to design optimization, the design variables q_e and K need to be included in the inputs of the PCNN. To be more specific, the input for the PCNN is (t,x,y,q_e,K). The training data for the phase field $P_T(t,x,y,q_e,K)$ and temperature field $Q_T(t,x,y,q_e,K)$ come from FEM simulations. Then the outputs of the PCNN are $P(t,x,y,q_e,K)$ and $Q(t,x,y,q_e,K)$, which

approximate the true phase field p and temperature field q respectively. The training of the PCNN for design optimization will require multiple sets of design variables and FEM simulation runs. In this example, we only demonstrate the feasibility of multiphysics prediction. Only two sets of design variables and FEM simulation data corresponding to two samples of dendritic growth are used for the training of MF-PCNNs. For each dendritic growth sample, the design variables q_e and K have constant values only, and two MF-PCNNs with different number of physical constraints are tested. The complete framework of materials design based on the MF-PCNN will be demonstrated in future work.

Similar to the previous two examples, loss functions are formulated based on the PDEs as well as the initial and boundary conditions. The adaptive weighting scheme in Eq. (5.10) is used to maximize the overall reduction speed of the total loss. The prediction of the MF-PCNN is a combination of the LF-PCNN prediction and the DANN as in Eq. (5.13). The LF-PCNN is trained with coarse simulation data during the time period $t \in [0, 1]$, and the HF-PCNN is trained with the denser simulation data during the time period $t \in [0, 0.2]$. The observed discrepancy between the predictions of the LF-PCNN and HF-PCNN during the time period $t \in [0, 0.2]$ serves as the training data for the DANN.

In the dendritic growth example, the simulation domain is $x, y \in [-2.5,2.5]$ and time period is $t \in [0,1]$. As listed in Table 5.3, the training data for the LF-PCNNs are sampled randomly from the FEM simulation, where the grid spacing is $\Delta x = 0.01$ and the time step is $\Delta t = 0.1$. The training data for the HF-PCNNs are sampled randomly from the FEM simulation, where the grid spacing is $\Delta x = 0.01$ and the time step is $\Delta t = 0.05$. More training data are used for the training of the HF-PCNNs to increase the prediction accuracy. Compared to the LF-PCNN1 and HF-PCNN1, the LF-PCNN2 and HF-PCNN2 have more

physical constraints involved. The training of the LF-PCNNs and HF-PCNNs stops when the total loss E is lower than a threshold value of 0.0001. Similarly, the training of the DANNs stops when the loss function E_δ is below 0.005. The threshold values are lower than those in the previous examples because more accurate predictions are needed to observe the complex dendritic shape.

Table 5.3. The setup for different ML models in the dendritic growth example

ML model	Structure	Amount of training data ($t \times x \times y$)	Number of physical constraints ($t \times x \times y$)	Time period/s
LF-PCNN1	30-20-30-20	2861 (random, $\Delta t = 0.1$)	21×21×21	[0, 1]
HF-PCNN1	30-20-30-20	3901 (random, $\Delta t = 0.05$)	5×21×21	[0, 0.2]
DANN1	5-5-5-5	9×51×51	0	[0, 0.2]
LF-PCNN2	30-20-30-20	2861 (random, $\Delta t = 0.1$)	11×41×41	[0, 1]
HF-PCNN2	30-20-30-20	3901 (random, $\Delta t = 0.05$)	3×41×41	[0, 0.2]
DANN2	5-5-5-5	9×51×51	0	[0, 0.2]

5.3 Computational Results

In this section, the results for the heat transfer, phase transition, and dendritic growth examples are shown. The heat transfer example is used to demonstrate the effectiveness of the proposed adaptive weighting scheme for the total loss function. Convergence analysis for the ANN and the PCNN is also conducted. The phase transition example is to demonstrate the performance of the MF-PCNN framework. The dendritic growth example is used to demonstrate the applicability of the proposed MF-PCNN framework for complex multiphysics problems in materials design.

5.3.1 Heat Transfer Example

To assess the sensitivity of weights, three weighting schemes of the total loss function are tested and compared with each other. In the PCNN1 listed in Table 5.1, the weights are equal and fixed in the total loss function

$$E = 0.25(E_T + E_P + E_I + E_s) \tag{5.25}$$

In the PCNN2, the weights are unequal and fixed in the total loss function

$$E = 0.125(E_T + 2E_P + 4E_I + E_s) \tag{5.26}$$

In the PCNN3, the weights are adaptive during the training as shown in Eq. (5.10). Assigning larger weights to the physical constraints indicates that prior knowledge will be more influential in the training process. When the training data are sparse, increasing the number of physical constraints can help improve the training efficiency. In addition, the weights of physical constraints need to be large enough in order to ensure training efficiency and prediction accuracy. When the weights of physical constraints are assigned, it is also necessary to consider the balance among different losses so that the reduction speeds of the four errors are comparable. The ideal case is that the four losses are reduced at the same speed so that the overall reduction speed of the total loss is maximized.

Here, the training data come from the FEM solutions. Figure 5.1 shows the original FEM solution of the temperature field, as well as the predictions by the traditional ANN, the equally-weighted PCNN1, the unequally-weighted PCNN2, and the adaptively-weighted PCNN3 at t = 1, respectively. The errors of the predicted temperature fields compared with the original FEM solution for different neural networks at t = 1 are shown in Figure 5.2, respectively. Here, the prediction error is the absolute difference between the prediction from the neural network and the FEM solution. The dots in the figures indicate

the sampling positions in the 2D domain, where a total of 26×26 samples are taken. It is seen that the prediction from the ANN is less accurate than the three PCNNs, because of the small training data set. The error is especially large in the area around saddle points. Notice that the training data for the ANN and PCNNs come from the same LF simulations. With the physical constraints added as regularization terms, the prediction errors of the PCNNs are reduced significantly.

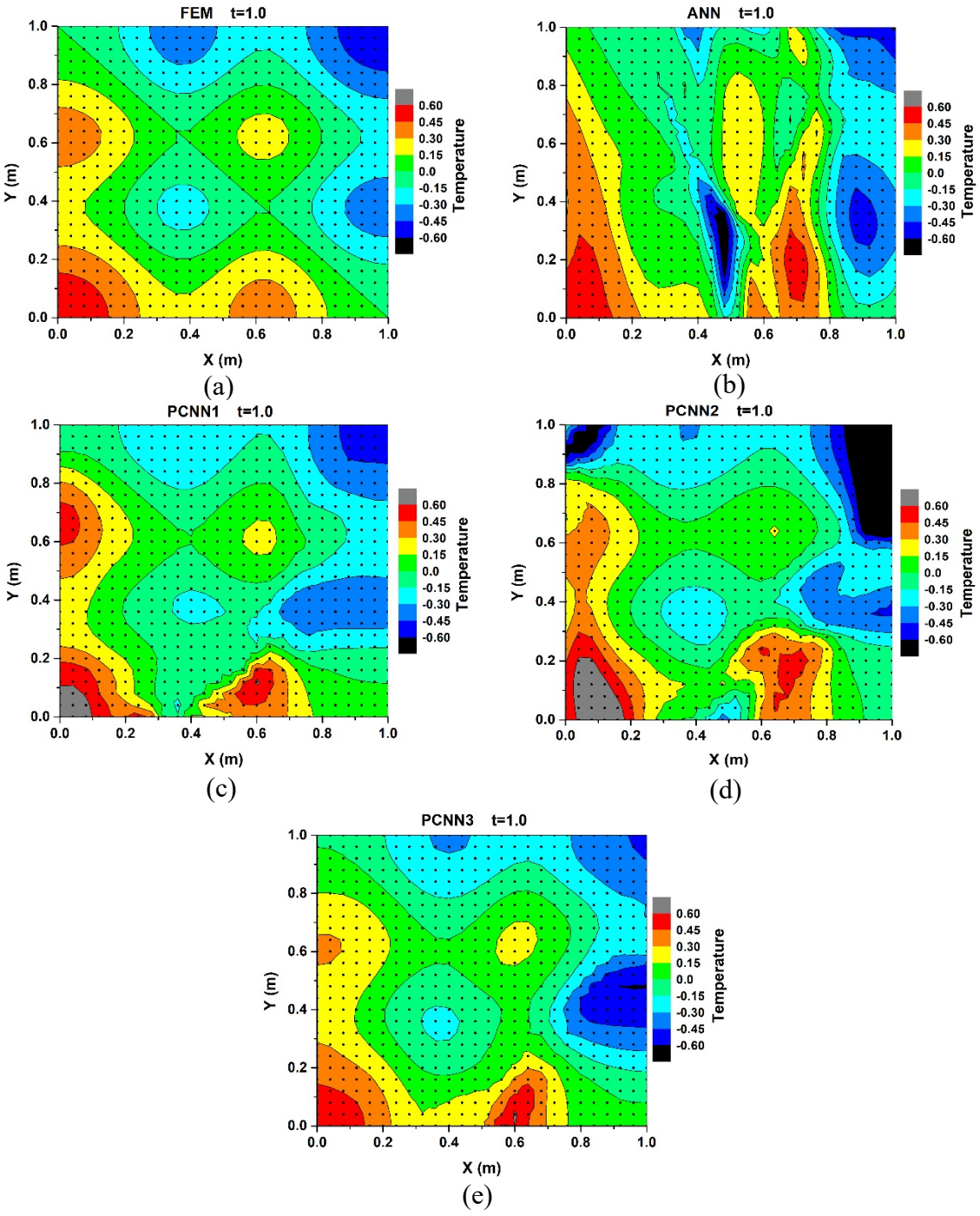


Figure 5.1. The predicted temperature fields from different models at t = 1: (a) original FEM solution, (b) traditional ANN, (c) equally-weighted PCNN1, (d) unequally-weighted PCNN2, and (e) adaptively-weighted PCNN3.

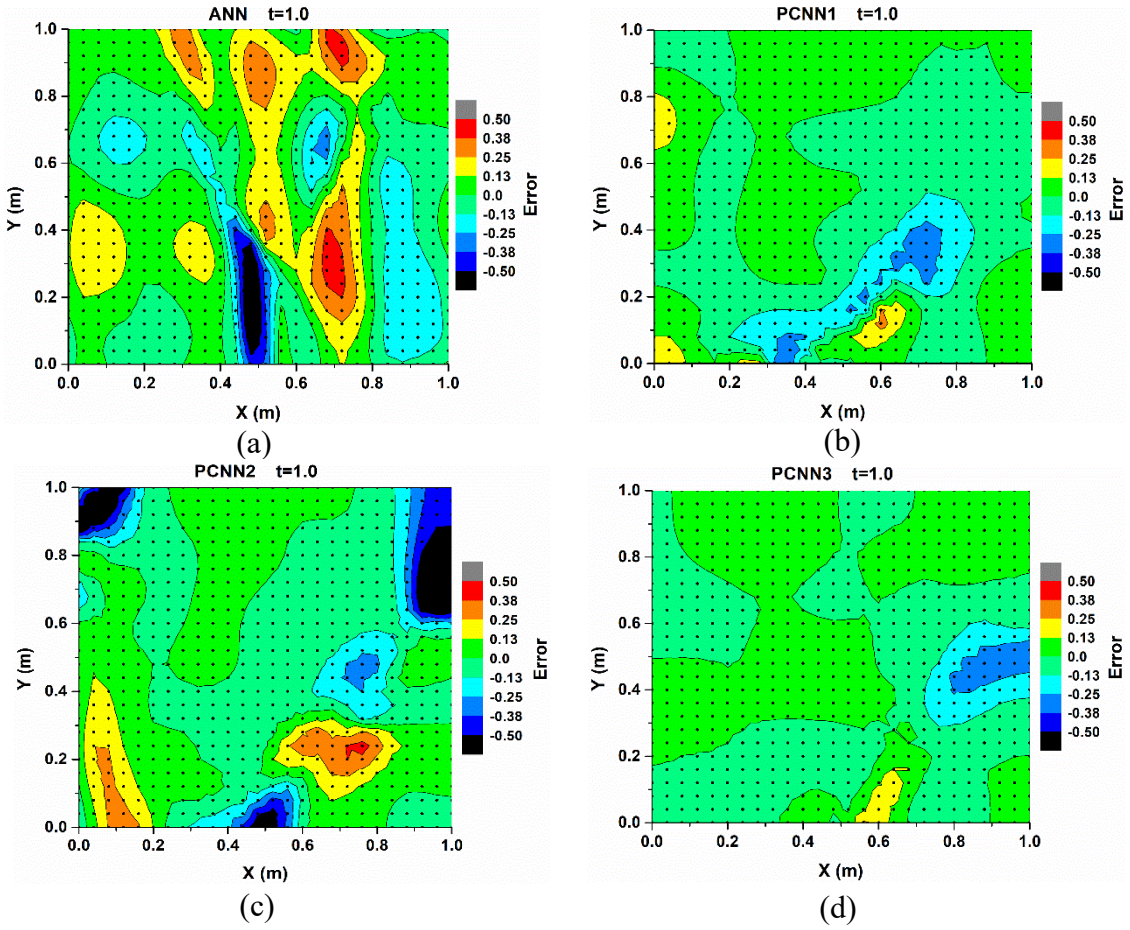


Figure 5.2. The errors of the predicted temperature fields compared to the FEM solution at t = 1: (a) traditional ANN, (b) equally-weighted PCNN1, (c) unequally-weighted PCNN2, and (d) adaptively-weighted PCNN3.

The learning curves for different PCNNs are shown in Figure 5.3. For the three different PCNNs, all losses monotonically decrease during the training. However, the difference between the convergence speeds of individual losses varies with the different weighting schemes. For the equally-weighted PCNN1, as shown in Figure 5.3(a), the initial loss is one order of magnitude larger than the boundary loss, meaning that the difference between the convergence speeds of individual losses is large. Therefore, it takes a longer time for the PCNN1 to converge. For the unequally-weighted PCNN2, the weights of physical constraints are higher in order to increase the influence of prior knowledge. As a

result, the different losses are within the same order of magnitude, as shown in Figure 5.3(b). As for the adaptively-weighted PCNN3, the weights are dynamically adjusted based on the percentages of individual losses in the total loss function. As shown in Figure 5.3(c), the different losses converged at the same speed and are well-balanced. The training time is the shortest among the three cases.

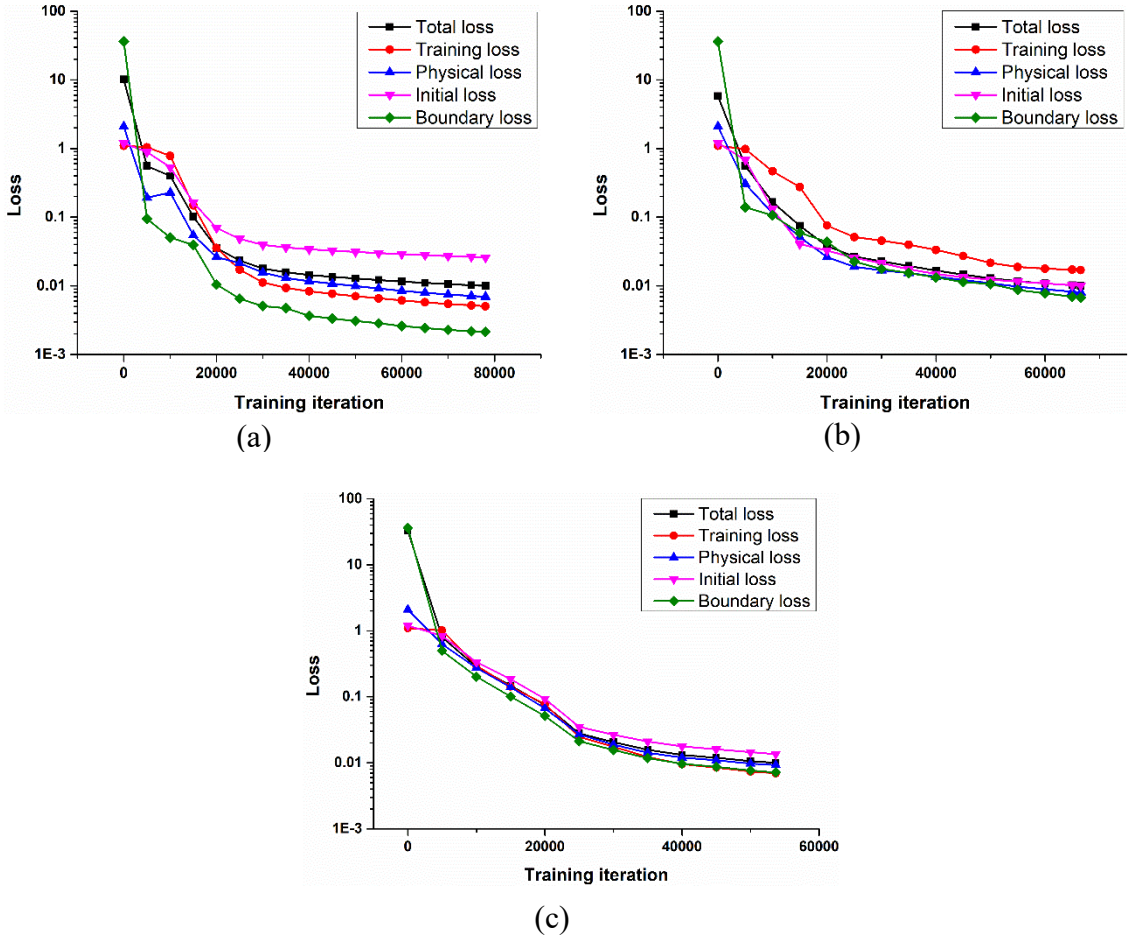


Figure 5.3. The learning curves for different PCNNs: (a) the equally-weighted PCNN1, (b) the unequally-weighted PCNN2, and (c) the adaptively-weighted PCNN3.

The quantitative comparisons of training time and the mean squared error (MSE) of prediction for the four neural networks are listed in Table 5.4. All MSEs of prediction for the PCNN1 and PCNN3 are almost one order of magnitude lower than that for the ANN.

As a result of stronger enforcement for the physical constraints, the prediction accuracy of the PCNN2 is higher than that of the PCNN1 at $t = 0$. However, the MSE of prediction at $t = 1$ for the PCNN2 is larger than that of the PCNN1. This could be caused by the in-balance between different losses in the PCNN2. As shown in Figure 5.3(b), the training loss is still larger than the threshold value of 0.01 when the training is finished, although the total loss as the weighted average has reached the threshold. The adaptively-weighted PCNN3 has all individual losses well-balanced and has the highest prediction accuracy. The PCNN3 also has the least training time among the three PCNNs. Notice that the computational time for training the PCNNs is much longer than that for the ANN because additional information from physical knowledge is used in the training.

Table 5.4. Quantitative comparison for different neural networks to solve the heat equation

Neural network	Training time (second)	MSE of prediction at $t = 0$	MSE of prediction at $t = 1$
ANN	8.66	0.1998	0.0293
PCNN1	1475.40	0.0225	0.0079
PCNN2	1259.91	0.0125	0.0350
PCNN3	1019.07	0.0139	0.0055

The convergence speeds of the ANN and the adaptively-weighted PCNN3 with respect to the amount of training data are compared in Figure 5.4. It is shown that the required amount of training data to reach a certain accuracy level of prediction at time $t = 1$ can be reduced by adding the physical constraints. Here, the number of physical constraints of the PCNN3 is $21 \times 6 \times 6 = 756$. The prediction MSEs at $t = 1$ of both ANN and PCNN decrease when the training data size increases. The advantage of PCNN over ANN is obvious when the training data size is small. When the training data size is less than 400,

the prediction accuracy can have nearly one order of magnitude difference. To reach the same accuracy level of 0.01, the ANN requires about 900 training data points, whereas the PCNN only needs about 300 training data points. As the training data size increases, the difference in prediction accuracy between the ANN and PCNN gradually reduces.

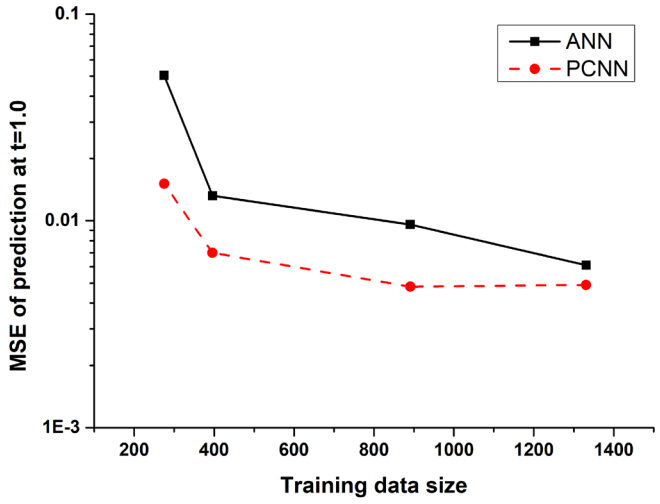


Figure 5.4. Convergence analysis for the ANN and the PCNN3.

5.3.2 Phase Transition Example

As shown in Eq. (5.13), the prediction of a MF-PCNN is a combination of the LF-PCNN prediction and the discrepancy predicted by a ML model (DANN or GP). First, a low-cost LF-PCNN is trained during the time period $t \in [0, 1]$ and then used as the baseline model. In addition, two high-cost HF-PCNNs (HF-PCNN1 and HF-PCNN2) are constructed. As shown in Table 5.2, the HF-PCNN1 is trained with data for the time period $t \in [0, 0.2]$, whereas the HF-PCNN2 is trained with the data for two time periods, $t \in [0, 0.2]$ and $t \in [0.8, 1]$. Then DANNs and GPs are trained to predict the discrepancy between the LF-PCNN and HP-PCNN predictions during the time period $t \in [0, 1]$. The observed discrepancy between the predictions of the LF-PCNN and HF-PCNN1 during the

time period $t \in [0, 0.2]$ serves as the training data for the DANN1, DANN2, and GP1. The network structure of DANN2 is more complex than DANN1. Similarly, the observed discrepancy between the predictions of the LF-PCNN and HF-PCNN2 for two time periods, $t \in [0, 0.2]$ and $t \in [0.8, 1]$, serves as the training data for the DANN3, DANN4, and GP2. Finally, the prediction of the MF-PCNN is the sum of the LF-PCNN prediction and the predicted discrepancy by DANNs or GPs. In this example, the mean square of the difference between the LF simulation and HF simulation is 0.0001 during the time period $t \in [0, 0.2]$. However, since a coarser mesh and larger time step is used in the LF simulation, errors are accumulated over time. Then, the mean square of the difference between the LF simulation and HF simulation becomes 0.0029 during the time period $t \in [0.8, 1.0]$. Therefore, LF simulations are less accurate than HF simulations in the later stage. It is useful to adopt the MF-PCNN framework to fully utilize the training data with different fidelity.

The predictions of the phase field from different models, including FEM solution, traditional ANN, and LF-PCNN at time $t = 0.5$ are shown in Figure 5.5. It is seen that the traditional ANN has larger prediction errors than PCNNs, especially at the saddle points where the true values are zeros. Adding physical constraints can significantly reduce the prediction errors, as in the LF-PCNN. At some saddle points, the phase field predicted by the LF-PCNN is still larger than zero, as shown in Figure 5.5(c). The predictions of the phase field from multi-fidelity models (combinations of LF-PCNN and DANNs, as well as LF-PCNN and GPs) at time $t = 0.5$ are shown in Figure 5.6. Compared to the LF-PCNN, the prediction errors of MF-PCNNs can be further reduced by adding the discrepancy prediction from DANNs or GPs. The phase field predicted by the MF-PCNNs is almost

zero at all saddle points. The difference between the predictions of the LF-PCNN and HF-PCNN can be captured by DANNs or GPs very well.

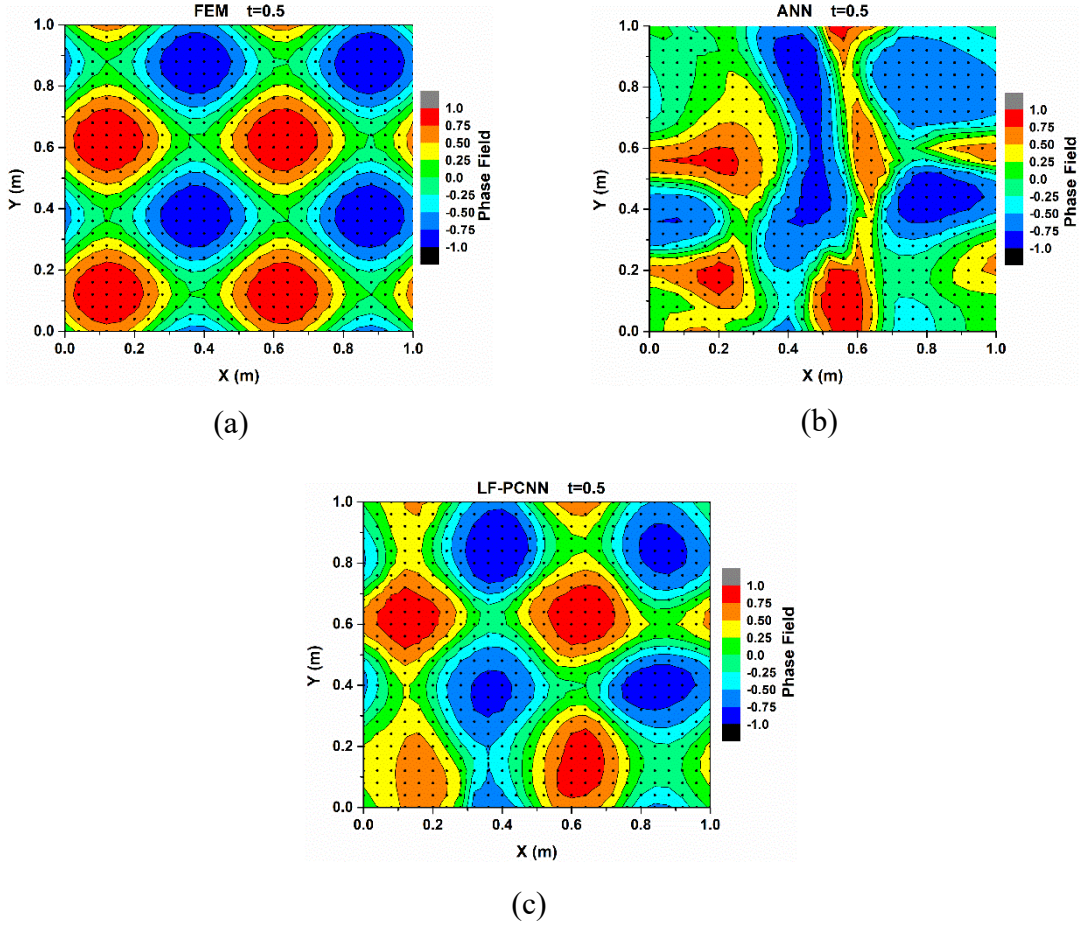


Figure 5.5. The predicted phase fields from different models at t = 0.5: (a) FEM solution, (b) ANN, and (c) LF-PCNN.

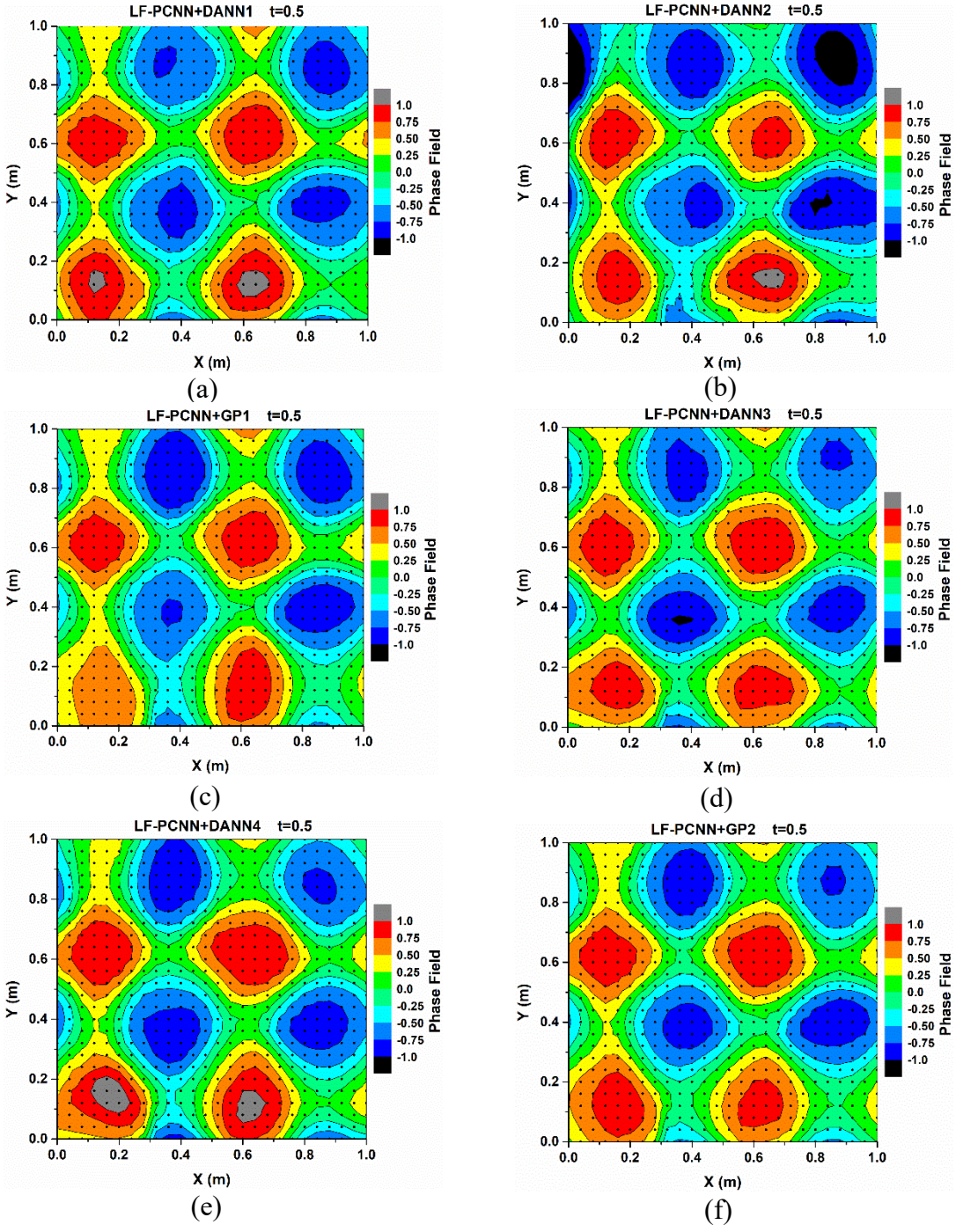


Figure 5.6. The predicted phase fields from multi-fidelity models at t = 0.5: (a) LF-PCNN+DANN1, (b) LF-PCNN+DANN2, (c) LF-PCNN+GP1, (d) LF-PCNN+DANN3, (e) LF-PCNN+DANN4, and (f) LF-PCNN+GP2.

The quantitative comparisons of training time and the MSEs of prediction for different ML models to solve the Allen-Cahn equation are listed in Table 5.5, where a MF-PCNN is composed of a LF-PCNN and a ML model to predict the discrepancy. For example, MF-PCNN1 = LF-PCNN+DANN1 means that the MF-PCNN1 is a combination of the LF-PCNN and DANN1. The total training time of a MF-PCNN is the sum of training times for the LF-PCNN, HF-PCNN, and the discrepancy model (DANN or GP). The total training time of MF-PCNN1 is 774.32+324.37+79.52 = 1178.21 s, where the training times of the LF-PCNN, HF-PCNN1, and DANN1 are 774.32, 324.37, and 79.52 s, respectively. It is noted that the prediction of the HF-PCNN1 is used for the training of the MF-PCNN1, MF-PCNN2, and MF-PCNN3, whereas the prediction of the HF-PCNN2 is used for the training of the rest of the MF-PCNNs. Therefore, the training times of the MF-PCNN4, MF-PCNN5, and MF-PCNN6 are longer because of more training data and physical constraints. It is noted that the training time of the GP1 is comparable to the DANNs, whereas the training time of the GP2 is one magnitude longer than the DANNs. The standard GP is computationally more expensive because the computation of inverse covariance matrices is involved. Therefore, the standard GP has a cubic time complexity $O(n^3)$, whereas the standard ANN has a linear time complexity $O(n)$, where n is the number of training data. Nevertheless, the GPs predict not only mean values but also the associated variances. Therefore, they are valuable in uncertainty quantification and robust optimization [126,127,215].

Table 5.5. Quantitative comparison between different ML models in the phase transition example

ML model	Training time (second)	MSE of prediction at $t = 0.5$	MSE of prediction at $t = 1.5$
ANN	7.93	0.2215	0.8866
LF-PCNN	774.32	0.0258	0.0684
MF-PCNN1=LF-PCNN+DANN1	774.32+324.37+79.52=1178.21	0.0133	0.0521
MF-PCNN2=LF-PCNN+DANN2	774.32+324.37+25.19=1123.88	0.0753	0.8508
MF-PCNN3=LF-PCNN+GP1	774.32+324.37+62.58=1161.27	0.0218	0.0587
MF-PCNN4=LF-PCNN+DANN3	774.32+3095.68+100.38=3970.38	0.0114	0.0399
MF-PCNN5=LF-PCNN+DANN4	774.32+3095.68+58.01=3928.01	0.0173	0.1926
MF-PCNN6=LF-PCNN+GP2	774.32+3095.68+1498.41=5368.41	0.0129	0.0648

The MSEs of predictions at different time periods for different ML models are compared in Figure 5.7. In general, the MSE of the prediction increases over time for different ML models. Since the prediction of the phase field at the current time step relies on the predictions from previous steps, the error will be accumulated over time. It is also noted that the time period $t \in [1, 2]$ is outside the time range $t \in [0, 1]$ of LF training data for the LF-PCNN. Therefore, the error for extrapolation is larger, which is a common issue for most ML models. Nevertheless, the MSEs of extrapolation for the LF-PCNN, MF-PCNN1, MF-PCNN3, MF-PCNN4, and MF-PCNN6 are one order of magnitude lower than that of the ANN.

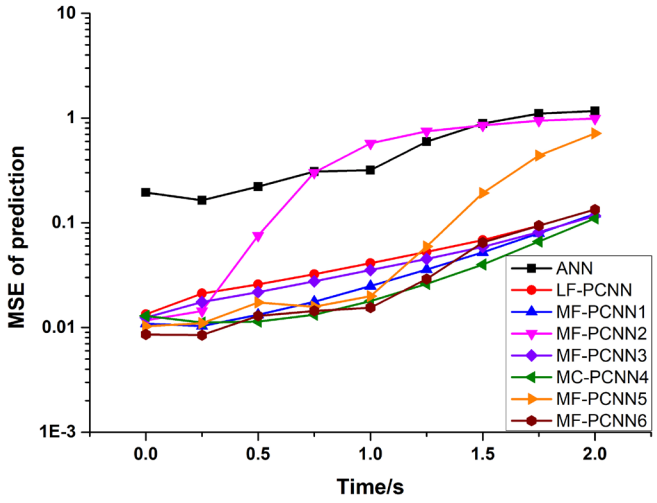


Figure 5.7. The change of MSE of prediction for different ML models.

The MSE of prediction from the MF-PCNN1 is significantly lower than that of the LF-PCNN for $t \in [0, 1]$. The difference between the MSEs however decreases for $t \in [1, 2]$. Furthermore, the MSE of prediction at $t = 0.5$ for the MF-PCNN1 is decreased by about 50%, compared to that of the LF-PCNN. As for the MF-PCNN2, its MSE of prediction is higher than that of LF-PCNN when $t > 0.5$. The MSE of prediction for the MF-PCNN2 is almost the same as that of the ANN when $t > 0.75$. The increased MSE for the MF-PCNN2 is most likely caused by overfitting since the DANN2 has more neurons than the DANN1. The MSE of prediction for the MF-PCNN3 is slightly lower than that of the LF-PCNN but higher than that of MF-PCNN1 for $t \in [0, 2]$. Notice that $t = 0.5$ is outside the time range $t \in [0, 0.2]$ of the HF training data for the HF-PCNN1, and the prediction is based on extrapolation. The errors indicate that DANN1 is better than GP1 for extrapolation.

With more training data and physical constraints, the HF-PCNN2 has two sampling spaces in the temporal dimension, which are [0, 0.2] and [0.8, 1]. The observed discrepancy between the predictions of the LF-PCNN and HF-PCNN2 is served as the training data for

the DANN3, DANN4, and GP2. Therefore, the prediction of the discrepancy between the LF-PCNN and HF-PCNN at t = 0.5 has become an interpolation problem. Compared to the MF-PCNN1, the MSE of prediction for the MF-PCNN4 is the lowest among all ML models for most of the time. With more training data, the MSE of prediction for the MF-PCNN5 is lower than that of the MF-PCNN2. However, the MSE of prediction for the MF-PCNN5 becomes higher than that of the MF-PCNN4 when $t > 0.25$ because of the overfitting. Compared to the MF-PCNN3, the MSE of prediction for the MF-PCNN6 is reduced with more training data when $t \in [0.0, 1.5]$. Therefore, the MF-PCNN6 has a lower MSE of prediction than that of MF-PCNN3 at the cost of a longer training time. However, the MSE of prediction for the MF-PCNN6 becomes the same as that of the LF-PCNN when $t > 1.75$, indicating the failure of prediction by GP2.

Among all ML models in this example, the MF-PCNN1 has the best performance since it has a relatively low training time and very good accuracy. The good performance of the MF-PCNN1 is mostly due to the simpler neural network structure of the DANN1.

5.3.3 Dendritic Growth Example

Here, instead of showing the complete design optimization procedure for iterative predictions and searching, we only show the evolutions of dendritic growth predicted by MF-PCNNs with two different sets of design variables. For the first design, the liquidus temperature is $q_e = 1$ and latent heat is $K = 2$. The predicted phase fields and temperature fields from FEM and the MF-PCNNs at t = 1.0 are shown in Figure 5.8. The settings for the two MF-PCNNs are compared in Table 5.3. There are 51×51 sampling points in the 2D domain. For the second design, the liquidus temperature is $q_e = 1.4$ and latent heat is $K = 2.8$. The predicted phase fields and temperature fields from FEM and the MF-PCNNs at t

= 1.0 are shown in Figure 5.9. For both cases, the predicted shapes of the primary arms from the MF-PCNNs are similar to the FEM simulation result, whereas the predicted secondary arms deviate from the simulation. Since secondary arms contain very thin and delicate features, more training data or physical constraints are needed to predict the secondary arms more accurately. The number of neurons also needs to be increased. For both dendrites, the predicted temperature fields from the MF-PCNNs correspond to the FEM simulation result very well, given that gradients in the temperature field are smaller than the phase field. The MSEs of prediction from the MF-PCNN for dendritic growth at t = 1.0 are shown in Table 5.6. It is shown that the MSE of prediction for the phase field is at least one order of magnitude larger than that for the temperature field in both cases. Compared to the MF-PCNN1, the MF-PCNN2 has more physical constraints. Therefore, the MSE of the prediction for the phase field from the MF-PCNN2 is lower than that from the MF-PCNN1. In future work, the architecture of MF-PCNNs needs to be optimized for such multiphysics examples, which will be largely determined by the resolutions of predicted fields.

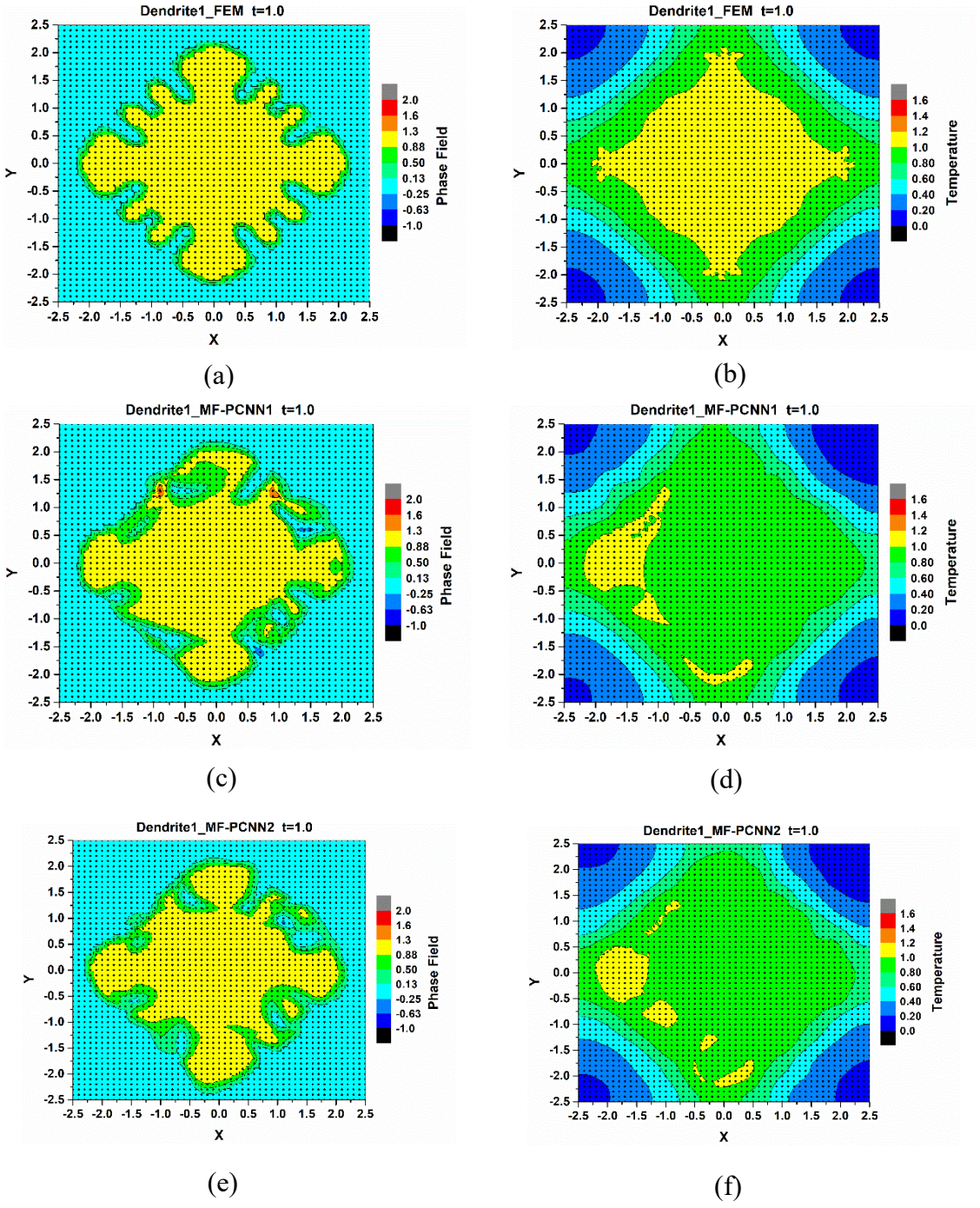


Figure 5.8. The predicted phase fields and temperature fields from different models for the first material option ($q_e = 1$; $K = 2$) at t = 1.0. Phase fields are shown in (a), (c), and (e). Temperature fields are shown in (b), (d), and (f).

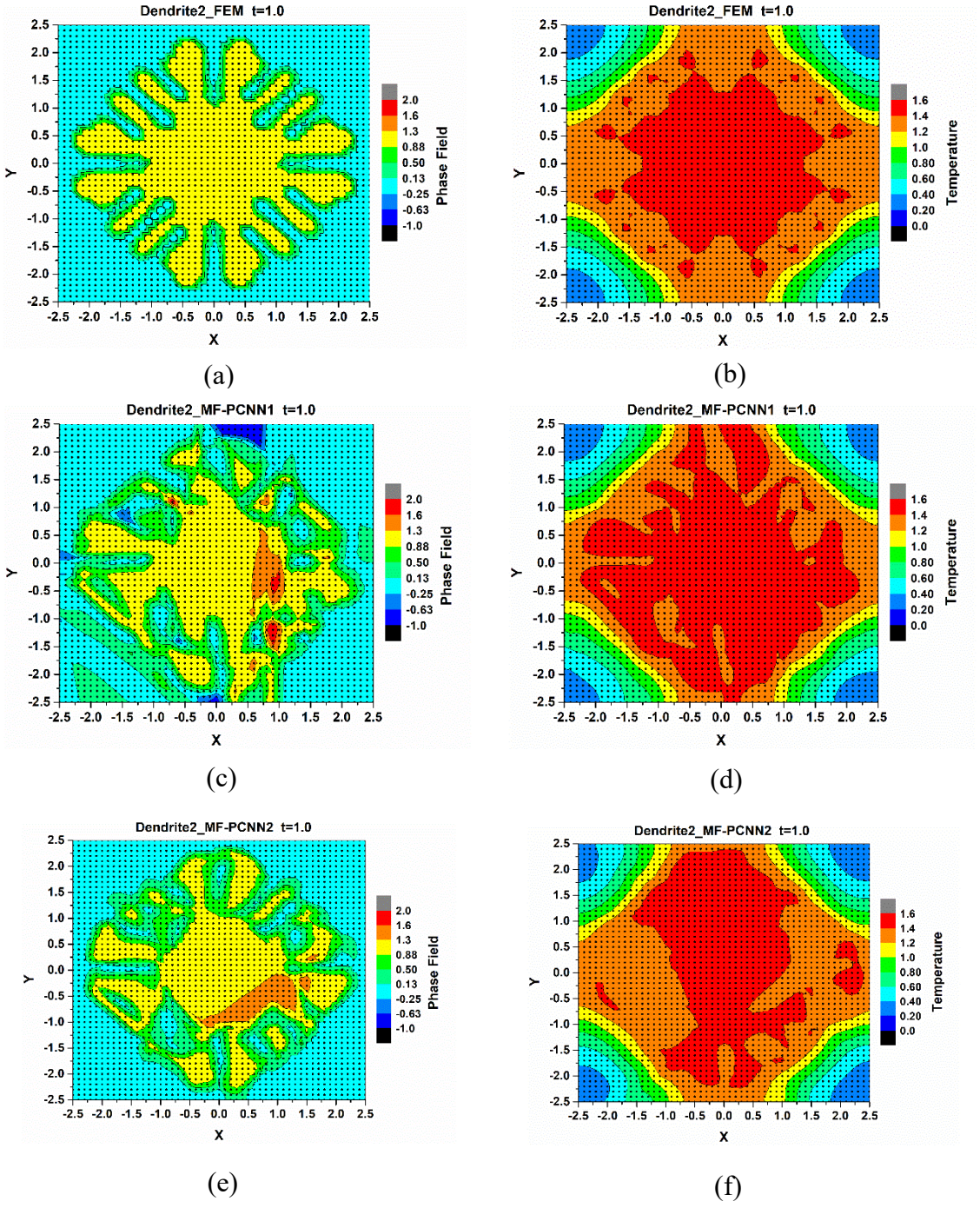


Figure 5.9. The predicted phase fields and temperature fields from different models for the second material option ($q_e = 1.4$; $K = 2.8$) at $t = 1.0$. Phase fields are shown in (a), (c), and (e). Temperature fields are shown in (b), (d), and (f).

Table 5.6. MSEs of prediction from the MF-PCNN for dendritic growth at $t = 1.0$

		Training time (second)	MSE of prediction for phase field	MSE of prediction for temperature field
Dendrite 1	MF-PCNN1	4320.18	0.0356	0.0047
	MF-PCNN2	8738.51	0.0346	0.0049
Dendrite 2	MF-PCNN1	37836.12	0.1127	0.0010
	MF-PCNN2	154791.97	0.0713	0.00384

5.4 Discussion and Conclusions

In this chapter, a new scheme of multi-fidelity physics-constrained neural networks is proposed to improve the efficiency of training in neural networks by reducing the required amount of training data and incorporating physical knowledge as constraints. Neural networks with two (or more) levels of fidelities are combined to improve the prediction accuracy. Low-fidelity networks predict the general trend, whereas high-fidelity networks model local details and fluctuations. For the concern of training cost, low-fidelity networks can be trained with low-fidelity data, and the prediction accuracy can be further improved with supplementary high-fidelity data. Thus, the training efficiency is improved from two aspects. The first one is the guidance from the physical knowledge, and the second one is a more cost-effective data collection and sampling strategy.

In engineering and physical sciences, the knowledge about the system behaviors is typically described by PDEs as well as the associated initial and boundary condition information. The physical knowledge can be easily added as the regularization terms into the total loss functions in neural networks. The physical constraints then can help reduce the searching space and guide the searching direction during the training. When new knowledge about the system is obtained, more physical constraints can be added to PCNNs

to improve the prediction accuracy. Thus the LF-PCNN and HF-PCNN to construct a MF-PCNN can be trained with different sets of constraints. It is reasonable to assume that the information of constraints can be obtained efficiently by the evaluation of analytical functions, which is less costly than obtaining simulation data for the training. The proposed formulation is generic and can be extended to other machine learning approaches, where regularization can be similarly applied.

The proposed scheme is demonstrated with three examples of materials modeling. The first example is the heat equation with zero Neumann boundary conditions, which is a linear PDE. The second example is the Allen-Cahn equation with periodic boundary conditions, which is a nonlinear PDE. The PCNN is effective for these two different types of PDEs with different boundary conditions. The third example is the dendritic growth during solidification where heat transfer and phase transition are coupled. The classical ANN with small training data sets tends to have large prediction errors. By adding physical constraints, the prediction accuracy of the PCNN can be one order of magnitude higher than the one from the classical ANN. Even with limited training data, the prediction of the PCNN is comparable with the original FEM solution. The weights associated with physical constraints can be adjusted to reflect the importance of prior knowledge. They also affect the prediction accuracy. It is demonstrated that the adaptive weighting scheme results in higher prediction accuracy and shorter training time because the different losses in the total cost function are well balanced and have a similar convergence speed. The convergence analysis shows that the required amount of training data can be reduced by adding more physical constraints. Based on the computational results, DANNs are more robust than GPs for extrapolation to predict the discrepancy between the LF-PCNN and HF-PCNN.

The results in this chapter have demonstrated the effectiveness of the proposed MF-PCNN framework for simulation prediction. The purpose of the proposed MF-PCNN framework is to reduce the required amount of training data by taking a gray-box approach. Although the offline training of MF-PCNN is slow, which takes up to several hours, its online evaluation or forward prediction can be done in a few seconds, once the training is finished. It was also shown that the training efficiency can be improved if the training data are from numerical simulations with different fidelities. The training data however are not limited to numerical simulation results only. They can also come from experimental measurements. The costs of experimental measurements can also be incorporated into the multi-fidelity scheme, where cost-effective sampling strategies can be taken.

To enhance the capability of the developed MF-PCNN, some extensions have been made. The weights of different losses are systematically adjusted by the new training algorithm called Dual-Dimer method as shown in CHAPTER 6. The architecture of MF-PCNNs can be further optimized for efficiency improvement by training the LF-PCNN and DANN together, which is shown in CHAPTER 6. Besides, to construct the process-structure-property relationship using ML tools, the design variables should be directly incorporated as the input vector. The PCNN is extended to include design variables for rapid solidification process optimization in metal AM in CHAPTER 8.

Future work will include several extensions. First, although classical ANN is a good baseline model to demonstrate the proposed MF-PCNN framework, the ANN could be replaced by the recurrent neural networks, such as long short-term memory neural networks, which may be more appropriate to solve time-dependent problems. Second, to further improve the training efficiency, a sequential sampling strategy can be adopted to

obtain an optimal combination of the HF and LF sample points for a given computational budget [126]. Finally, a more rigorous and comprehensive comparison of the scalability between the proposed MF-PCNN and multi-fidelity GP models is needed. Theoretically, the standard GP has a cubic time complexity, whereas the standard ANN has linear time complexity. However, incorporating the computational cost of data sampling in the multi-fidelity simulation scenario will provide an overall scalability picture in terms of training and prediction time.

The proposed scheme should not be regarded as the replacement of classical numerical simulation methods (e.g. finite element and spectral methods) for solving partial differential equations. Rather, it enhances the efficiency of engineering design when high-fidelity simulations need to be run repetitively to obtain samples for design optimization. The required number of samples for optimization for high-dimensional problems usually is very large. The cost of training machine learning tools therefore can only be justified for complex problems with a high-dimensional searching space. For high-dimensional problems, the physical constraints can still be applied in the same scheme. However, as the number of constraints increases, they may not be treated as equally important, and those with trivial weights will be removed. In principle, as the dimension of the problem increases, the advantage of PCNNs will become more prominent because the required amount of training data can be reduced more significantly. The proposed scheme has the potential of making machine learning useful for real-world engineering applications where data sparsity is a common issue.

CHAPTER 6. A DUAL-DIMER METHOD FOR TRAINING PHYSICS-CONSTRAINED NEURAL NETWORKS WITH MINIMAX ARCHITECTURE

6.1 Introduction

Physics-constrained neural networks (PCNNs) can reduce the required amount of training data for neural networks in predicting physical phenomena. However, the weights of different losses from data and physical constraints are adjusted empirically in PCNNs. In this chapter, we propose a new formulation of PCNN to systematically search the optimal weights of different losses. The training of the PCNN is formulated as a minimax problem instead of minimization. The PCNN with the minimax architecture is called PCNN-MM. The training of the PCNN-MM is searching the high-order saddle points of the objective function. The order of saddle points indicates the number of negative eigenvalues of the Hessian matrix. Most of the existing saddle point search algorithms only find first-order saddle points. The traditional gradient descent ascent (GDA) algorithm for high-order saddle points has the convergence issue for nonconvex-nonconcave functions, where the functions are neither convex in the subspace for minimization nor concave in the subspace for maximization. A novel saddle point search algorithm called the Dual-Dimer algorithm is proposed to search high-order saddle points during the training of the PCNN-MM.

A multi-fidelity physics-constrained neural network with minimax architecture (MF-PCNN-MM) is also proposed. In the MF-PCNN-MM, LF and HF data are integrated together to make the tradeoff between efficiency and accuracy for multi-fidelity

metamodeling. The MF-PCNN-MM is composed of two ANNs. The first ANN is used to approximate the LF data, whereas the second ANN is adopted to approximate the mapping function between the LF and HF data.

In the remainder of this chapter, the proposed PCNN-MM formulation and the Dual-Dimer algorithm will be described in Section 6.2. The local convergence analysis of the Dual-Dimer algorithm is also included. The formulation of the MF-PCNN-MM is also described in Section 6.2. In Section 6.3, the proposed Dual-Dimer algorithm is evaluated using three nonconvex-nonconcave analytical functions, including a four-dimensional (4D) Rastrigin function, a 4D Ackley function, and a 20D Styblinski–Tang function. In Section 6.4, a heat transfer problem is used to demonstrate the effectiveness of the Dual-Dimer algorithm. The performance of the PCNN-MM trained by the Dual-Dimer method is compared with the PCNN with the adaptive weighting scheme and the PCNN-MM trained by the GDA method. The convergence speed and stability of different models are also tested. In Section 6.5, the developed MF-PCNN-MM is demonstrated by three 2D examples of heat transfer, phase transition, and dendritic growth.

6.2 Methodology

Here, we propose a new generic formulation of physics-constrained neural networks with the minimax architecture. The adjustment of weights associated with physical constraints can be done systematically during the training process. A new high-order saddle point search method is also developed to train the new PCNNs with nonconvex-nonconcave objective functions. The formulation of the PCNN-MM is described in Section 6.2.1. The generic Dual-Dimer saddle point search method is introduced in Section 6.2.2.

The local convergence analysis of the Dual-Dimer algorithm is included in Section 6.2.3. The formulation of the MF-PCNN-MM is shown in 6.2.4.

6.2.1 Physics-Constrained Neural Network with Minimax Architecture

The training of the PCNN-MM is to solve the minimax problem

$$\min_{\mathbf{w}} \max_{\boldsymbol{\alpha}} E(\mathbf{w}, \boldsymbol{\alpha}) = \lambda_T(\boldsymbol{\alpha}) E_T(\mathbf{w}) + \lambda_P(\boldsymbol{\alpha}) E_P(\mathbf{w}) + \lambda_I(\boldsymbol{\alpha}) E_I(\mathbf{w}) + \lambda_s(\boldsymbol{\alpha}) E_s(\mathbf{w}) \tag{6.1}$$

where the weights of different losses λ_T, λ_P, λ_I, and λ_S are now functions of parameters $\boldsymbol{\alpha} = (\alpha_T, \alpha_P, \alpha_I, \alpha_S)$. Training is to minimize the possible loss for a worst-case (maximum loss) scenario. That is, we perform the maximization of the total loss $E(\mathbf{w}, \boldsymbol{\alpha})$ over the parameter subspace of $\boldsymbol{\alpha}$ and the minimization of the total loss over the parameter subspace of $\mathbf{w}$. During the training of the PCNN-MM, the weights of different losses λ's will be adjusted to maximize the total loss $E(\mathbf{w}, \boldsymbol{\alpha})$ in $\boldsymbol{\alpha}$ subspace, whereas the weights of the neural network $\mathbf{w}$'s will be tuned to minimize the total loss $E(\mathbf{w}, \boldsymbol{\alpha})$. When one of the losses is larger than the other ones, its corresponding weight tends to increase to emphasize the importance of that particular loss so that the total loss is maximized. To counteract, the weights of the neural network will be adjusted to minimize the total loss so that the total loss can be reduced faster. That is how the weights of different losses are systematically adjusted. In this work, the weights of different losses are defined as the softmax functions as

$$\lambda_i(\boldsymbol{\alpha}) = \frac{\exp(\alpha_i)}{\sum_i \exp(\alpha_i)}, i \in \{T, P, I, S\} \tag{6.2}$$

After applying softmax functions, the range of the weights of different losses λ_i will be in the interval $[0,1]$, and they will add up to one.

Let $\boldsymbol{\theta} = (\mathbf{w}, \boldsymbol{\alpha})$ denote the optimization parameters for objective function E. The training of the PCNN-MM is to find a minimax point or saddle point on a high-dimensional energy landscape E. The training of the PCNN-MM, which is to solve the minimax problem in Eq. (6.1), is equivalent to finding a saddle point $\boldsymbol{\theta}^* = (\mathbf{w}^*, \boldsymbol{\alpha}^*)$ such that

$$E(\mathbf{w}^*, \boldsymbol{\alpha}) \leq E(\mathbf{w}^*, \boldsymbol{\alpha}^*) \leq E(\mathbf{w}, \boldsymbol{\alpha}^*) \ \ (\forall \mathbf{w} \in \mathbb{R}^D, \forall \boldsymbol{\alpha} \in \mathbb{R}^4) \tag{6.3}$$

That is, the saddle point is the minimum in $\mathbf{w}$ subspace and maximum in $\boldsymbol{\alpha}$ subspace. The sufficient conditions for $\boldsymbol{\theta}^* = (\mathbf{w}^*, \boldsymbol{\alpha}^*)$ to be the desired saddle point are: (1) the gradients of the objective function with respect to $(\mathbf{w}, \boldsymbol{\alpha})$ are zeros, i.e., $\nabla_{\mathbf{w}} E(\boldsymbol{\theta}^*) = \mathbf{0}$ and $\nabla_{\boldsymbol{\alpha}} E(\boldsymbol{\theta}^*) = \mathbf{0}$; (2) the second derivatives $\nabla^2_{\mathbf{w}} E(\boldsymbol{\theta}^*)$ in the $\mathbf{w}$ subspace are positive semi-definite; and (3) the second derivatives $\nabla^2_{\boldsymbol{\alpha}} E(\boldsymbol{\theta}^*)$ in the $\boldsymbol{\alpha}$ subspace are negative semi-definite. It is noted that the training of the PCNN-MM is different from training traditional neural networks where saddle points need to be avoided and overcome. The goal of training PCNN-MM is to search the desired saddle point which is the minimum in $\mathbf{w}$ subspace and maximum in $\boldsymbol{\alpha}$ subspace.

6.2.2 The Dual-Dimer Method

It is known that the steepest step $\Delta\boldsymbol{\theta}$ to reach a stationary point (local minimum, local maximum, or saddle point) can be obtained by Newton's method

$$\Delta\boldsymbol{\theta} = \mathbf{H}^{-1}\mathbf{f} = \sum_i \frac{(\mathbf{v}_i \cdot \mathbf{f})\mathbf{v}_i}{\beta_i} \tag{6.4}$$

where $\mathbf{f} = -\nabla E$ is the force, $\mathbf{H}$ is the Hessian matrix, $\mathbf{v}_i$ is the eigenvector, and β_i is the corresponding eigenvalue. The drawback of the gradient descent method is not the search direction but the size of the step along each eigenvector direction. Therefore, a small step should be taken in the direction $\mathbf{v}_i$ when the corresponding eigenvalue β_i is small. By

rescaling the gradients in each direction with the inverse of the corresponding eigenvalue, Newton's method in Eq. (6.4) can accelerate the convergence. However, in high-dimensional problems, the computations of all eigenvectors and eigenvalues are very expensive.

The Dual-Dimer method is designed to improve the computational efficiency for high-dimensional problems. Let β_s denotes the minimum eigenvalue of $\nabla^2_{\mathbf{w}} E(\boldsymbol{\theta})$ with its corresponding eigenvector $\mathbf{v}_s$, and β_l denotes the maximum eigenvalue of $\nabla^2_{\boldsymbol{\alpha}} E(\boldsymbol{\theta})$ with its corresponding eigenvector $\mathbf{v}_l$. By augmenting the gradient descent ascent with the rescaled projections of the force along the extreme eigenvectors $(\mathbf{v}_s, \mathbf{v}_l)$, the step to reach the desired high-order saddle point in the Dual-Dimer method is given by

$$\begin{aligned} \Delta\boldsymbol{\theta} &= (\Delta\boldsymbol{\theta}_{\mathbf{w}}, \Delta\boldsymbol{\theta}_{\boldsymbol{\alpha}}) + (\Delta\boldsymbol{\theta}_s, \Delta\boldsymbol{\theta}_l) \\ &= \eta\big(-\nabla_{\mathbf{w}} E(\boldsymbol{\theta}), \nabla_{\boldsymbol{\alpha}} E(\boldsymbol{\theta})\big) + \left(-\frac{\big(\mathbf{v}_s \cdot \nabla_{\mathbf{w}} E(\boldsymbol{\theta})\big)\mathbf{v}_s}{|\beta_s|}, \frac{\big(\mathbf{v}_l \cdot \nabla_{\boldsymbol{\alpha}} E(\boldsymbol{\theta})\big)\mathbf{v}_l}{|\beta_l|}\right) \end{aligned} \tag{6.5}$$

where $\Delta\boldsymbol{\theta}_{\mathbf{w}}$ is the gradient descent sub-step given by the first-order gradient-based optimization method [213] in the $\mathbf{w}$ subspace, and $\Delta\boldsymbol{\theta}_{\boldsymbol{\alpha}}$ is the gradient ascent sub-step in the $\boldsymbol{\alpha}$ subspace. η is the learning rate for the gradient descent ascent sub-steps. $\Delta\boldsymbol{\theta}_s$ is the projection of the force along the $\mathbf{v}_s$ direction, and $\Delta\boldsymbol{\theta}_l$ is the projection of the force along the $\mathbf{v}_l$ direction. With augmented sub-steps $\Delta\boldsymbol{\theta}_s$ and $\Delta\boldsymbol{\theta}_l$, it is expected that at the end of the training $\nabla^2_{\mathbf{w}} E(\boldsymbol{\theta}^*)$ does not have negative eigenvalues in $\mathbf{w}$ and $\nabla^2_{\boldsymbol{\alpha}} E(\boldsymbol{\theta}^*)$ does not have positive eigenvalues in $\boldsymbol{\alpha}$. Therefore, the use of the extreme eigenvalues and eigenvectors in the Dual-Dimer method is to make sure that the high-order saddle points are found.

In the original dimer method [136–138], a dimer is rotated to find the minimum curvature direction and then translated to a first-order saddle point. The minimum curvature direction corresponds to the extreme eigenvector in the minimum subspace for the first-

order saddle point. In the proposed Dual-Dimer method, the way to calculate extreme eigenvalues and eigenvectors for first-order saddle points in the original dimer method is adopted and extended to calculate the extreme values in both the minimum and maximum subspaces for high-order saddle points. The proposed Dual-Dimer method is also different from the dimer method by rescaling the step sizes along the extreme eigenvectors with the inverse of the extreme eigenvalues. The extreme eigenvalues (β_s, β_l) and eigenvectors $(\mathbf{v}_s, \mathbf{v}_l)$ are computed by rotating two dimers in the subspaces of $\mathbf{w}$ and $\boldsymbol{\alpha}$ without expensive calculations of the Hessian matrix $\mathbf{H}$. The first dimer in the $\mathbf{w}$ subspace is composed of two endpoints $\boldsymbol{\theta}_1$ and $\boldsymbol{\theta}_2$, which are slightly displaced by the fixed dimer length $2\Delta R$. The locations of the endpoints $\boldsymbol{\theta}_1$ and $\boldsymbol{\theta}_2$ are given by

$$\begin{cases} \boldsymbol{\theta}_1 = \boldsymbol{\theta}_0 + \Delta R\mathbf{n} \\ \boldsymbol{\theta}_2 = \boldsymbol{\theta}_0 - \Delta R\mathbf{n} \end{cases} \tag{6.6}$$

where $\mathbf{n}$ is the unit vector along the dimer axis and $\boldsymbol{\theta}_0$ is the midpoint of the dimer. Here, the components of $\mathbf{n}$ in the $\mathbf{w}$ subspace are nonzero, whereas the components of $\mathbf{n}$ in the $\boldsymbol{\alpha}$ subspace are always zero. Therefore, the rotation of the first dimer is confined in the $\mathbf{w}$ subspace. The dimer axis $\mathbf{n}$ is rotated into the smallest curvature direction of the potential energy $C(\mathbf{n})$ at the dimer midpoint $\boldsymbol{\theta}_0$, which is to solve the minimization problem

$$\min_{\mathbf{n}} C(\mathbf{n}) = \mathbf{n}^T \mathbf{H} \mathbf{n} \approx \frac{(\mathbf{f}_2 - \mathbf{f}_1) \cdot \mathbf{n}}{2\Delta R} \tag{6.7}$$

where $\mathbf{H}$ is the Hessian matrix at the dimer midpoint $\boldsymbol{\theta}_0$. $\mathbf{f}_1 = -\nabla E(\boldsymbol{\theta}_1)$ and $\mathbf{f}_2 = -\nabla E(\boldsymbol{\theta}_2)$ are the forces at the locations $\boldsymbol{\theta}_1$ and $\boldsymbol{\theta}_2$, respectively. It is noted that only first derivatives are required to estimate curvatures in Eq. (6.7). This is the reason that the Dual-Dimer method is computationally efficient. Furthermore, the curvature $C(\mathbf{n})$ becomes the eigenvalue if $\mathbf{n}$ is the eigenvector of the Hessian matrix. Once the smallest curvature $C(\mathbf{n})$

is found, the minimum eigenvalue β_s in the $\mathbf{w}$ subspace is equal to $C(\mathbf{n})$ and the components of $\mathbf{n}$ in the $\mathbf{w}$ subspace becomes the extreme eigenvector $\mathbf{v}_s$. The minimization problem in Eq. (6.7) is numerically solved by rotating the dimer. The details can be found in the original dimer method [136–138].

Similarly, the second dimer in the $\boldsymbol{\alpha}$ subspace is composed of two endpoints $\boldsymbol{\theta}_3$ and $\boldsymbol{\theta}_4$ with their locations given by

$$\begin{cases} \boldsymbol{\theta}_3 = \boldsymbol{\theta}_0 + \Delta R\mathbf{m} \\ \boldsymbol{\theta}_4 = \boldsymbol{\theta}_0 - \Delta R\mathbf{m} \end{cases} \tag{6.8}$$

where $\mathbf{m}$ is the unit vector along the dimer axis. Here, the components of $\mathbf{m}$ in the $\boldsymbol{\alpha}$ subspace are nonzero, whereas the components of $\mathbf{m}$ in the $\mathbf{w}$ subspace are always zero. Therefore, the rotation of the second dimer is confined in the $\boldsymbol{\alpha}$ subspace. The dimer axis $\mathbf{m}$ is rotated into the largest curvature direction of the potential energy, which is to solve the maximization problem

$$\max_{\mathbf{m}} C(\mathbf{m}) = \mathbf{m}^T\mathbf{H}\mathbf{m} \approx \frac{(\mathbf{f}_4 - \mathbf{f}_3)\cdot\mathbf{m}}{2\Delta R} \tag{6.9}$$

where $\mathbf{f}_3 = -\nabla E(\boldsymbol{\theta}_3)$ and $\mathbf{f}_4 = -\nabla E(\boldsymbol{\theta}_4)$ are the forces at the locations $\boldsymbol{\theta}_3$ and $\boldsymbol{\theta}_4$, respectively. Once the largest curvature $C(\mathbf{m})$ is found, the maximum eigenvalue β_l in the $\boldsymbol{\alpha}$ subspace is equal to $C(\mathbf{m})$ and the components of $\mathbf{m}$ in the $\boldsymbol{\alpha}$ subspace become the extreme eigenvector $\mathbf{v}_l$.

The details of the Dual-Dimer algorithm are shown in Table 6.1. Iteratively, the sub-steps $\Delta\boldsymbol{\theta}_{\mathbf{w}}$, $\Delta\boldsymbol{\theta}_{\boldsymbol{\alpha}}$, $\Delta\boldsymbol{\theta}_s$, and $\Delta\boldsymbol{\theta}_l$ are calculated and the estimated saddle point location is updated. There are five hyperparameters (p, δ, γ, η, ε) that need to be tuned in the Dual-Dimer method. Parameter p represents the frequency of updating extreme eigenvalues and eigenvectors. If p is small, the overall computational cost will be high. If p is large, the

estimations of current extreme eigenvalues and eigenvectors are not accurate. Parameter δ is introduced in the algorithm to avoid the zero-division error. When the eigenvalue is close to zero, it means that the curvature is very small and the saddle point degenerates. Parameter γ means the maximum step length of $\Delta\boldsymbol{\theta}_s$ and $\Delta\boldsymbol{\theta}_l$ to make sure that the training is converged. Parameter η is the learning rate for the gradient descent ascent sub-steps. If η is small, the training will be slow. If η is large, the training may be unstable. When the objective function E or the norm of the force $\|\mathbf{f}\|_2$ is less than the threshold ε, the search for the saddle points stops. Trade-offs need to be made between the computational accuracy and efficiency for these hyperparameters to improve the overall performance of the Dual-Dimer method. Sensitivity studies were done in this work to tune them. A more systematic method to find the optimal hyperparameters is needed in future work.

Table 6.1. The Dual-Dimer algorithm

Input:	initial optimization parameters $\boldsymbol{\theta}_0 = (\mathbf{w}_0, \boldsymbol{\alpha}_0)$, objective function E, hyperparameters p, δ, γ, η, ε.
Output:	desired saddle point $\boldsymbol{\theta}^*$
Procedure:	1. Initialize the iteration $t = 0$, $\boldsymbol{\theta}_t = \boldsymbol{\theta}_0$ 2. Evaluate energy $E(\boldsymbol{\theta}_t)$ and force $\mathbf{f} = -\nabla E$ 3. When $t \; mod \; p = 0$, compute the extreme eigenvalues (β_s, β_l) and eigenvectors $(\mathbf{v}_s, \mathbf{v}_l)$ by rotating two dimers in the subspaces of $\mathbf{w}$ and $\boldsymbol{\alpha}$ 4. Calculate $\Delta\boldsymbol{\theta}_{\mathbf{w}} = -\eta\nabla_{\mathbf{w}}E(\boldsymbol{\theta})$ and $\Delta\boldsymbol{\theta}_{\boldsymbol{\alpha}} = \eta\nabla_{\boldsymbol{\alpha}}E(\boldsymbol{\theta})$ 5. If $\|\beta_s\| > \delta$, $\Delta\boldsymbol{\theta}_s = -\frac{(\mathbf{v}_s\cdot\nabla_{\mathbf{w}}E(\boldsymbol{\theta}))\mathbf{v}_s}{\|\beta_s\|}$; otherwise, $\Delta\boldsymbol{\theta}_s = \mathbf{0}$; If $\|\beta_l\| > \delta$, $\Delta\boldsymbol{\theta}_l = \frac{(\mathbf{v}_l\cdot\nabla_{\boldsymbol{\alpha}}E(\boldsymbol{\theta}))\mathbf{v}_l}{\|\beta_l\|}$; otherwise, $\Delta\boldsymbol{\theta}_l = \mathbf{0}$ 6. If $\|\Delta\boldsymbol{\theta}_s\|_2 > \gamma$, $\Delta\boldsymbol{\theta}_s = \gamma\frac{\Delta\boldsymbol{\theta}_s}{\|\Delta\boldsymbol{\theta}_s\|_2}$; If $\|\Delta\boldsymbol{\theta}_l\|_2 > \gamma$, $\Delta\boldsymbol{\theta}_l = \gamma\frac{\Delta\boldsymbol{\theta}_l}{\|\Delta\boldsymbol{\theta}_l\|_2}$ 7. $t = t + 1$ 8. Update optimization parameters by calculating $\Delta\boldsymbol{\theta} = (\Delta\boldsymbol{\theta}_{\mathbf{w}}, \Delta\boldsymbol{\theta}_{\boldsymbol{\alpha}}) + (\Delta\boldsymbol{\theta}_s, \Delta\boldsymbol{\theta}_l)$ and $\boldsymbol{\theta}_t = \boldsymbol{\theta}_{t-1} + \Delta\boldsymbol{\theta}$ 9. Return to step 2 until $\|\mathbf{f}\|_2 < \varepsilon$ or $E < \varepsilon$ 10. Output $\boldsymbol{\theta}^* = \boldsymbol{\theta}_t$

6.2.3 Local Convergence

The local convergence of the Dual-Dimer method is analyzed here. Let us define a fixed-point function

$$F(\boldsymbol{\theta}) = \boldsymbol{\theta} + \eta\big(-\nabla_{\mathbf{w}}E(\boldsymbol{\theta}), \nabla_{\boldsymbol{\alpha}}E(\boldsymbol{\theta})\big) + \left(-\frac{\big(\mathbf{v}_s \cdot \nabla_{\mathbf{w}}E(\boldsymbol{\theta})\big)\mathbf{v}_s}{|\beta_s|}, \frac{\big(\mathbf{v}_l \cdot \nabla_{\boldsymbol{\alpha}}E(\boldsymbol{\theta})\big)\mathbf{v}_l}{|\beta_l|}\right) \tag{6.10}$$

and assume that $F(\boldsymbol{\theta})$ is differentiable. The desired saddle point $\boldsymbol{\theta}^*$ can be found by iteratively applying the fixed-point function $F(\boldsymbol{\theta})$. If $\beta_s = 0$ and $\beta_l = 0$, as shown in Table 6.1, then the fixed-point iteration becomes the GDA method, which is locally stable according to [216,217]. If $\beta_s \neq 0$ and $\beta_l \neq 0$, we have the following lemmas and theorem.

Lemma 1. The Jacobian of the loss function at the desired saddle point $\boldsymbol{\theta}^* = (\mathbf{w}^*, \boldsymbol{\alpha}^*)$ is

$$\nabla F(\boldsymbol{\theta}^*) = \mathbf{I} + \eta\begin{pmatrix} -\nabla^2_{\mathbf{w}}E(\boldsymbol{\theta}^*) & -\nabla^2_{\mathbf{w},\boldsymbol{\alpha}}E(\boldsymbol{\theta}^*) \\ \nabla^2_{\boldsymbol{\alpha},\mathbf{w}}E(\boldsymbol{\theta}^*) & \nabla^2_{\boldsymbol{\alpha}}E(\boldsymbol{\theta}^*) \end{pmatrix} + \begin{pmatrix} -\frac{1}{\beta_s}\mathbf{v}_s\mathbf{v}_s^T\nabla^2_{\mathbf{w}}E(\boldsymbol{\theta}^*) & -\frac{1}{\beta_s}\mathbf{v}_s\mathbf{v}_s^T\nabla^2_{\mathbf{w},\boldsymbol{\alpha}}E(\boldsymbol{\theta}^*) \\ -\frac{1}{\beta_l}\mathbf{v}_l\mathbf{v}_l^T\nabla^2_{\boldsymbol{\alpha},\mathbf{w}}E(\boldsymbol{\theta}^*) & -\frac{1}{\beta_l}\mathbf{v}_l\mathbf{v}_l^T\nabla^2_{\boldsymbol{\alpha}}E(\boldsymbol{\theta}^*) \end{pmatrix} \tag{6.11}$$

where $\mathbf{I}$ is the real-valued identity matrix. If there exists an η $(\eta > 0)$ such that the absolute values of all the eigenvalues of $\nabla F(\boldsymbol{\theta}^*)$ are less than 1, then there is an open neighborhood K of $\boldsymbol{\theta}^*$ so that for all $\boldsymbol{\theta} \in K$, the fixed-point iterations of $F(\boldsymbol{\theta})$ in Eq. (6.10) are stable in K. The rate of convergence is at least linear.

Proof. Since $\boldsymbol{\theta}^*$ is the desired saddle point, we have $\nabla_{\mathbf{w}}E(\boldsymbol{\theta}^*) = \mathbf{0}$, $\nabla_{\boldsymbol{\alpha}}E(\boldsymbol{\theta}^*) = \mathbf{0}$, $\beta_s \geq 0$, and $\beta_l \leq 0$. Furthermore $F(\boldsymbol{\theta}^*) = \boldsymbol{\theta}^*$. The Jacobian $\nabla F(\boldsymbol{\theta}^*)$ is given by

$$\begin{aligned}\nabla F(\boldsymbol{\theta}^*) &= \mathbf{I} + \eta \begin{pmatrix} -\nabla^2_{\mathbf{w}} E(\boldsymbol{\theta}^*) & -\nabla^2_{\mathbf{w},\boldsymbol{\alpha}} E(\boldsymbol{\theta}^*) \\ \nabla^2_{\boldsymbol{\alpha},\mathbf{w}} E(\boldsymbol{\theta}^*) & \nabla^2_{\boldsymbol{\alpha}} E(\boldsymbol{\theta}^*) \end{pmatrix} \\ &\quad + \nabla \left(-\frac{\left(\mathbf{v}_s \cdot \nabla_{\mathbf{w}} E(\boldsymbol{\theta}^*)\right)\mathbf{v}_s}{|\beta_s|}, \frac{\left(\mathbf{v}_l \cdot \nabla_{\boldsymbol{\alpha}} E(\boldsymbol{\theta}^*)\right)\mathbf{v}_l}{|\beta_l|} \right)\end{aligned} \tag{6.12}$$

If $\beta_s > 0$, then we have

$$\begin{aligned}&\nabla \left(-\frac{\left(\mathbf{v}_s \cdot \nabla_{\mathbf{w}} E(\boldsymbol{\theta}^*)\right)\mathbf{v}_s}{|\beta_s|} \right) = \nabla \left(-\frac{\left(\mathbf{v}_s \cdot \nabla_{\mathbf{w}} E(\boldsymbol{\theta}^*)\right)\mathbf{v}_s}{\beta_s} \right) \\ &= -\mathbf{v}_s \otimes \nabla \left(\frac{\mathbf{v}_s \cdot \nabla_{\mathbf{w}} E(\boldsymbol{\theta}^*)}{\beta_s} \right) - \left(\frac{\mathbf{v}_s \cdot \overbrace{\nabla_{\mathbf{w}} E(\boldsymbol{\theta}^*)}^{\mathbf{0}}}{\beta_s} \right) \nabla \mathbf{v}_s \\ &= -\mathbf{v}_s \otimes \left[\frac{1}{\beta_s} \nabla \left(\mathbf{v}_s \cdot \nabla_{\mathbf{w}} E(\boldsymbol{\theta}^*) \right) + \left(\mathbf{v}_s \cdot \overbrace{\nabla_{\mathbf{w}} E(\boldsymbol{\theta}^*)}^{\mathbf{0}} \right) \nabla \frac{1}{\beta_s} \right] \\ &= -\frac{1}{\beta_s} \mathbf{v}_s \otimes \left[\nabla^T \mathbf{v}_s \overbrace{\nabla_{\mathbf{w}} E(\boldsymbol{\theta}^*)}^{\mathbf{0}} + \nabla^T \left(\nabla_{\mathbf{w}} E(\boldsymbol{\theta}^*) \right) \mathbf{v}_s \right] \\ &= -\frac{1}{\beta_s} \mathbf{v}_s \mathbf{v}_s^T \nabla \left(\nabla_{\mathbf{w}} E(\boldsymbol{\theta}^*) \right) \\ &= \left(-\frac{1}{\beta_s} \mathbf{v}_s \mathbf{v}_s^T \nabla^2_{\mathbf{w}} E(\boldsymbol{\theta}^*) \quad -\frac{1}{\beta_s} \mathbf{v}_s \mathbf{v}_s^T \nabla^2_{\mathbf{w},\boldsymbol{\alpha}} E(\boldsymbol{\theta}^*) \right)\end{aligned} \tag{6.13}$$

Similarly, if $\beta_l < 0$, we have

$$\nabla \left(\frac{\left(\mathbf{v}_l \cdot \nabla_{\boldsymbol{\alpha}} E(\boldsymbol{\theta}^*)\right)\mathbf{v}_l}{|\beta_l|} \right) = \left(-\frac{1}{\beta_l} \mathbf{v}_l \mathbf{v}_l^T \nabla^2_{\boldsymbol{\alpha},\mathbf{w}} E(\boldsymbol{\theta}^*) \quad -\frac{1}{\beta_l} \mathbf{v}_l \mathbf{v}_l^T \nabla^2_{\boldsymbol{\alpha}} E(\boldsymbol{\theta}^*) \right) \tag{6.14}$$

Therefore, we have the Jacobian

$$\begin{aligned}\nabla F(\boldsymbol{\theta}^*) &= \mathbf{I} + \eta \begin{pmatrix} -\nabla^2_{\mathbf{w}} E(\boldsymbol{\theta}^*) & -\nabla^2_{\mathbf{w},\boldsymbol{\alpha}} E(\boldsymbol{\theta}^*) \\ \nabla^2_{\boldsymbol{\alpha},\mathbf{w}} E(\boldsymbol{\theta}^*) & \nabla^2_{\boldsymbol{\alpha}} E(\boldsymbol{\theta}^*) \end{pmatrix} \\ &\quad + \begin{pmatrix} -\frac{1}{\beta_s} \mathbf{v}_s \mathbf{v}_s^T \nabla^2_{\mathbf{w}} E(\boldsymbol{\theta}^*) & -\frac{1}{\beta_s} \mathbf{v}_s \mathbf{v}_s^T \nabla^2_{\mathbf{w},\boldsymbol{\alpha}} E(\boldsymbol{\theta}^*) \\ -\frac{1}{\beta_l} \mathbf{v}_l \mathbf{v}_l^T \nabla^2_{\boldsymbol{\alpha},\mathbf{w}} E(\boldsymbol{\theta}^*) & -\frac{1}{\beta_l} \mathbf{v}_l \mathbf{v}_l^T \nabla^2_{\boldsymbol{\alpha}} E(\boldsymbol{\theta}^*) \end{pmatrix}.\end{aligned}$$

According to the fixed point theorem [216], if there is an $\eta > 0$ such that the absolute values of the eigenvalues of the Jacobian $\nabla F(\boldsymbol{\theta}^*)$ are all smaller than 1, then there is an open neighborhood K of $\boldsymbol{\theta}^*$ so that for all $\boldsymbol{\theta} \in K$, the iterates of $F(\boldsymbol{\theta})$ in Eq. (6.10) are stable. The rate of convergence is at least linear.

Lemma 2. Let $\beta_A = a + bi$ be the eigenvalues of the matrix $\mathbf{A}$, $\beta_B = c + di$ be the eigenvalues of the matrix $\mathbf{B}$, where $i = \sqrt{-1}$. The eigenvalues of the matrix $\mathbf{I} + \eta\mathbf{A} + \mathbf{B}$, where $\eta > 0$, lie in the unit ball if

$$\Delta = [2(a + ac + bd)]^2 - 4(a^2 + b^2)(c^2 + 2c + d^2) > 0 \tag{6.15}$$

and

$$\begin{cases} 0 < \eta < \dfrac{-2(a + ac + bd) + \sqrt{\Delta}}{2(a^2 + b^2)}, \\ if\ a + ac + bd \geq 0\ and\ c^2 + 2c + d^2 > 0\ ; \\ max\left\{0, \dfrac{-2(a + ac + bd) - \sqrt{\Delta}}{2(a^2 + b^2)}\right\} < \eta < \dfrac{-2(a + ac + bd) + \sqrt{\Delta}}{2(a^2 + b^2)}, \\ if\ a + ac + bd < 0; \end{cases} \tag{6.16}$$

for all eigenvalues β_A of $\mathbf{A}$ and β_B of $\mathbf{B}$.

Proof. If the eigenvalues of the matrix $\mathbf{I} + \eta\mathbf{A} + \mathbf{B}$ lie in the unit ball, then $|1 + \eta\beta_A + \beta_B|^2 < 1$. That is, $(1 + \eta a + b)^2 + (\eta c + d)^2 < 1$, which leads to

$$(a^2 + b^2)\eta^2 + 2(a + ac + bd)\eta + c^2 + 2c + d^2 < 0 \tag{6.17}$$

To find the real solutions of η, we need to make sure the discriminant is larger than zero, as

$$\Delta = [2(a + ac + bd)]^2 - 4(a^2 + b^2)(c^2 + 2c + d^2) > 0 \tag{6.18}$$

Two real roots $\eta_1 = \frac{-2(a+ac+bd)-\sqrt{\Delta}}{2(a^2+b^2)}$ and $\eta_2 = \frac{-2(a+ac+bd)+\sqrt{\Delta}}{2(a^2+b^2)}$ can be obtained. Since $\eta > 0$, we have

$$\eta_2 = \frac{-2(a+ac+bd)+\sqrt{\Delta}}{2(a^2+b^2)} > 0 \text{ or } \sqrt{\Delta} > 2(a+ac+bd) \tag{6.19}$$

If $a + ac + bd \geq 0$, then $\Delta > [2(a + ac + bd)]^2$. Therefore,

$$c^2 + 2c + d^2 > 0 \tag{6.20}$$

Meanwhile, it is obvious that $\eta_1 = \frac{-2(a+ac+bd)-\sqrt{\Delta}}{2(a^2+b^2)} < 0$. Therefore, the range of η should be $0 < \eta < \eta_2$ in order to satisfy Eq. (6.17). If $a + ac + bd < 0$, automatically $\eta_2 = \frac{-2(a+ac+bd)+\sqrt{\Delta}}{2(a^2+b^2)} > 0$ without any further conditions. The range of η should be $max\{0, \eta_1\} < \eta < \eta_2$ to satisfy Eq. (6.17).

Theorem 1. Let $\boldsymbol{\theta}^* = (\mathbf{w}^*, \boldsymbol{\alpha}^*)$ be the desired saddle point, $\beta_A = a + bi$ be the eigenvalues of $\mathbf{A} = \begin{pmatrix} -\nabla^2_{\mathbf{w}} E(\boldsymbol{\theta}^*) & -\nabla^2_{\mathbf{w},\boldsymbol{\alpha}} E(\boldsymbol{\theta}^*) \\ \nabla^2_{\boldsymbol{\alpha},\mathbf{w}} E(\boldsymbol{\theta}^*) & \nabla^2_{\boldsymbol{\alpha}} E(\boldsymbol{\theta}^*) \end{pmatrix}$, $\beta_B = c + di$ be the eigenvalues of $\mathbf{B} = \begin{pmatrix} -\frac{1}{\beta_s}\mathbf{v}_s\mathbf{v}_s^T\nabla^2_{\mathbf{w}} E(\boldsymbol{\theta}^*) & -\frac{1}{\beta_s}\mathbf{v}_s\mathbf{v}_s^T\nabla^2_{\mathbf{w},\boldsymbol{\alpha}} E(\boldsymbol{\theta}^*) \\ -\frac{1}{\beta_l}\mathbf{v}_l\mathbf{v}_l^T\nabla^2_{\boldsymbol{\alpha},\mathbf{w}} E(\boldsymbol{\theta}^*) & -\frac{1}{\beta_l}\mathbf{v}_l\mathbf{v}_l^T\nabla^2_{\boldsymbol{\alpha}} E(\boldsymbol{\theta}^*) \end{pmatrix}$, and $\eta > 0$. The fixed-point iterations of $F(\boldsymbol{\theta})$ in Eq. (6.10) are locally stable if

$$\Delta = [2(a + ac + bd)]^2 - 4(a^2 + b^2)(c^2 + 2c + d^2) > 0 \tag{6.21}$$

and

$$\begin{cases} 0 < \eta < \dfrac{-2(a+ac+bd)+\sqrt{\Delta}}{2(a^2+b^2)}, \\ if\ a + ac + bd \geq 0\ and\ c^2 + 2c + d^2 > 0\ ; \\ max\left\{0, \dfrac{-2(a+ac+bd)-\sqrt{\Delta}}{2(a^2+b^2)}\right\} < \eta < \dfrac{-2(a+ac+bd)+\sqrt{\Delta}}{2(a^2+b^2)}, \\ if\ a + ac + bd < 0; \end{cases} \tag{6.22}$$

for all eigenvalues β_A of $\mathbf{A}$ and β_B of $\mathbf{B}$.

6.2.4 Multi-Fidelity Physics-Constrained Neural Network with Minimax Architecture

In the MF-PCNN-MM, LF and HF data are integrated together to make the tradeoff between efficiency and accuracy for multi-fidelity metamodeling. The optimal weights of training losses with different fidelities and other losses associated with physical constraints can be systematically searched during the training process.

The key to multi-fidelity metamodeling is to discover the relationship between the LF and HF data. A comprehensive correlation function widely used in multi-fidelity metamodeling [122] is given by

$$u_H(\mathbf{x}, u_L) = \rho(\mathbf{x})u_L + \delta(\mathbf{x}) \tag{6.23}$$

where $\mathbf{x}$ is the input vector, u_L is the LF data, u_H is the HF data, $\rho(\mathbf{x})$ is the multiplicative correlation surrogate, and $\delta(\mathbf{x})$ is the additive correlation surrogate. To make it more general, the correlation function in Eq. (6.23) can be rewritten as

$$u_H(\mathbf{x}, u_L) = F(\mathbf{x}, u_L) \tag{6.24}$$

where F represents the mapping function between the LF and HF data. The output of the MF-PCNN-MM $U_H(\mathbf{x}, U_L)$ is used to approximate the true solution $u(\mathbf{x})$ of a general time-dependent PDE in Eqs. (5.4)-(5.6).

The training of the MF-PCNN-MM is conducted by solving the minimax problem

$$\begin{aligned} \min_{\mathbf{w}} \max_{\boldsymbol{\alpha}} E(\mathbf{w}, \boldsymbol{\alpha}) = {} & \lambda_L(\boldsymbol{\alpha})E_L(\mathbf{w}) + \lambda_H(\boldsymbol{\alpha})E_H(\mathbf{w}) \\ & + \lambda_P(\boldsymbol{\alpha})E_P(\mathbf{w}) + \lambda_I(\boldsymbol{\alpha})E_I(\mathbf{w}) + \lambda_S(\boldsymbol{\alpha})E_S(\mathbf{w}) \end{aligned} \tag{6.25}$$

where $\mathbf{w}$ are the weights of two ANNs in the MF-PCNN-MM. E_L, E_H, E_P, E_I, and E_S correspond to the losses caused by low-fidelity data, high-fidelity data, the physical model, initial conditions, and boundary conditions, respectively. The weights of different losses λ_L, λ_H, λ_P, λ_I, and λ_S are functions of parameters $\boldsymbol{\alpha} = (\alpha_L, \alpha_H, \alpha_P, \alpha_I, \alpha_S)$, which are defined by the softmax function as

$$\lambda_i(\boldsymbol{\alpha}) = \frac{\exp(\alpha_i)}{\sum_i \exp(\alpha_i)}, i \in \{L, H, P, I, S\} \tag{6.26}$$

Here,

$$E_L = \frac{1}{N_L}\sum_{i=1}^{N_L}\left|U_L(\mathbf{x}_i^L) - u_L(\mathbf{x}_i^L)\right|^2 \tag{6.27}$$

is the LF training loss caused by the discrepancy between the LF prediction $U_L(\cdot)$ and the LF training data $u_L(\cdot)$, $\mathbf{x}_i^{(\cdot)}$ denotes the sampling points in the defined domain, and $N_{(\cdot)}$ denotes the number of sampling points. Similarly,

$$E_H = \frac{1}{N_H}\sum_{i=1}^{N_H}\left|U_H\left(\mathbf{x}_i^H, U_L(\mathbf{x}_i^H)\right) - u_H(\mathbf{x}_i^H)\right|^2 \tag{6.28}$$

is the HF training loss caused by the discrepancy between the HF prediction $U_H(\cdot)$ and the HF training data $u_H(\cdot)$. Usually, the amount of the LF training data is larger than that of the HF training data. It is noted that the LF prediction $U_L(\mathbf{x}^H)$ and the HF prediction $U_H\left(\mathbf{x}^H, U_L(\mathbf{x}^H)\right)$ share the same input $\mathbf{x}^H$ when the HF data is used during the training process. In addition,

$$E_P = \frac{1}{N_P}\sum_{i=1}^{N_P}\left|\mathbf{D}\left[U_H\left(\mathbf{x}_i^P, U_L(\mathbf{x}_i^P)\right)\right] - f(\mathbf{x}_i^P)\right|^2 \tag{6.29}$$

$$E_I = \frac{1}{N_I}\sum_{i=1}^{N_I}\left|\boldsymbol{\Lambda}\left[U_H\left(\mathbf{x}_i^I, U_L(\mathbf{x}_i^I)\right)\right] - g(\mathbf{x}_i^I)\right|^2 \tag{6.30}$$

and

$$E_S = \frac{1}{N_S}\sum_{i=1}^{N_S}\left|\boldsymbol{\Gamma}\left[U_H\left(\mathbf{x}_i^S, U_L(\mathbf{x}_i^S)\right)\right] - h(\mathbf{x}_i^S)\right|^2 \tag{6.31}$$

are the losses caused by the violations of the model, the initial condition, and boundary conditions as the physical constraints from Eqs. (5.4)-(5.6), respectively.

The architecture of the proposed MF-PCNN-MM is schematically illustrated in Figure 6.1, which is composed of two ANNs. The first ANN $U_L(\mathbf{x})$ is used to approximate the LF data, whereas the second ANN $U_H(\mathbf{x}, U_L)$ is adopted to approximate the mapping function between the LF and HF data. If the total loss in Eq. (6.25) only includes the training losses associated with the LF and HF training data, then the MF-PCNN-MM will become the MF-NN-MM. If the total loss in Eq. (6.25) includes the training loss associated with the LF data and losses caused by the physical constraints, then the MF-PCNN-MM will become the LF-PCNN-MM by using the low-fidelity ANN only.

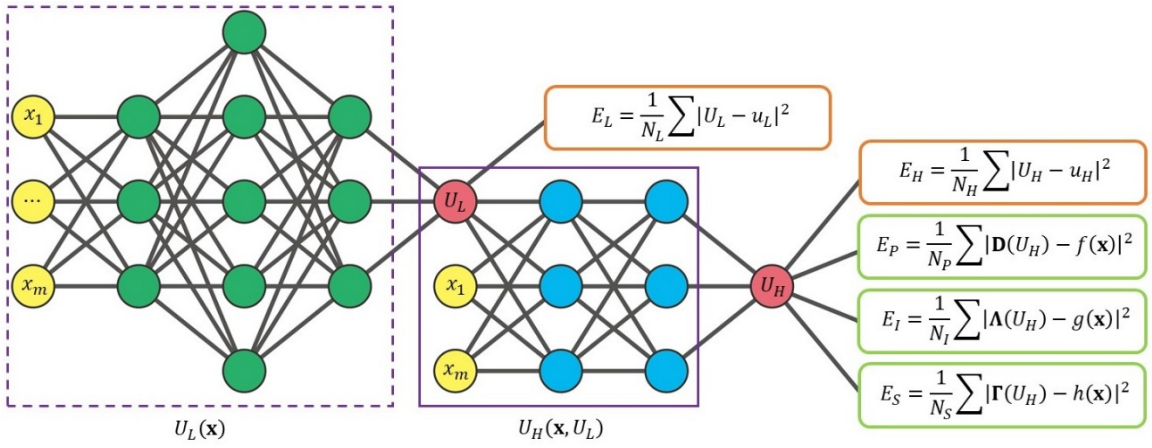


Figure 6.1. Schematic diagram of MF-PCNN-MM. The purple dash box (green nodes) represents the low-fidelity ANN connected to the purple solid box (blue nodes) representing the high-fidelity ANN. The orange rounded boxes represent the training losses caused by data discrepancy, whereas the green rounded boxes represent the losses associated with physical constraints. (For interpretation of the colors in the figure, the reader is referred to the web version.)

6.3 Evaluation of the Dual-Dimer Algorithm

The proposed Dual-Dimer algorithm is evaluated with three analytical nonconvex-nonconcave functions. They are a 4D Rastrigin function, a 4D Ackley function, and a 20D Styblinski–Tang function. The first saddle point problem of the 4D Rastrigin function is given by

$$\min_{x_1,x_2}\max_{x_3,x_4} E(\mathbf{x}) = \sum_{i=1}^{4}[x_i^2 - 10\cos(2\pi x_i) + 10] \tag{6.32}$$

The second problem of the 4D non-separable Ackley function is given by

$$\min_{x_1,x_2}\max_{x_3,x_4} E(\mathbf{x}) = -20exp\left(-0.2\sqrt{\frac{1}{4}\sum_{i=1}^{4}x_i^2}\right) - exp\left(\frac{1}{4}\sum_{i=1}^{4}\cos(2\pi x_i)\right) \tag{6.33}$$

$$+20 + e$$

The third one of the 20D Styblinski–Tang function is given by

$$\min_{x_1\sim x_{10}}\max_{x_{11}\sim x_{20}} E(\mathbf{x}) = \frac{1}{2}\sum_{i=1}^{20}[x_i^4 - 16x_i^2 + 5x_i] \tag{6.34}$$

There are multiple stationary points on the surfaces of these analytical functions, which makes it difficult to find high-order saddle points. Since the objective functions are analytical, the gradients and Hessian matrices of the objective functions can be computed easily. Therefore, the high-order saddle points can be easily verified.

Both GDA and Dual-Dimer methods are used to search a second-order saddle point of the 4D Rastrigin function, a second-order saddle point of the 4D Ackley function, and a tenth-order saddle point of the 20D Styblinski–Tang function. The gradient descent ascent steps in the GDA and Dual-Dimer method are given by the Adam algorithm with the learning rate of 5×10^{-4}. The dimer distance is $2\Delta R = 2 \times 10^{-4}$. The hyperparameters

of the Dual-Dimer method in examples of analytical functions are listed in Table 6.2. The search stops when the norm of the force is less than the threshold ($\|\mathbf{f}\|_2 < \varepsilon$).

Table 6.2. Hyperparameters of the Dual-Dimer method in examples of analytical functions

Hyperparameters	Value
Frequency of updating extreme eigenvalues and eigenvectors, p	40
The parameter to avoid the zero-division error, δ	1×10^{-3}
Maximum step length of $\Delta\boldsymbol{\theta}_s$ and $\Delta\boldsymbol{\theta}_l$, γ	0.1
Learning rate for the gradient descent ascent sub-steps, η	5×10^{-4}
The threshold for stopping search ($\|\mathbf{f}\|_2 < \varepsilon$), ε	1×10^{-4}

The high-order saddle points found by the GDA and Dual-Dimer methods are listed in Table 6.3. In the examples of Rastrigin and Ackley functions, the second-order saddle points $\mathbf{x}^*$ found by the GDA and Dual-Dimer methods are the same. In the example of Styblinski–Tang function, two different tenth-order saddle points were found by the GDA and Dual methods. By changing the random seed, different second-order saddle points can be found by the GDA and the Dual-Dimer method. Since variables in the Rastrigin and Styblinski–Tang functions are separable, all off-diagonal elements of their Hessian matrices are zeros. Therefore, the diagonal elements of their Hessian matrices are eigenvalues. On the contrary, since variables in Ackley function are non-separable, some off-diagonal elements of its Hessian matrix are nonzero. It is shown in Table 6.3 that the extreme eigenvalues (β_s, β_l) calculated by the Dual-Dimer method agree well with the true extreme eigenvalues (β_s^*, β_l^*). It is noted that the GDA method does not provide additional eigenvalue information, whereas the Dual-Dimer method provides. It is easy to verify that the norms of the gradient $\|\nabla E(\mathbf{x}^*)\|_2$ at all identified saddle points are less than 1×10^{-4}. The minimum eigenvalue β_s in the minimum subspace at the saddle point $\mathbf{x}^*$ is positive,

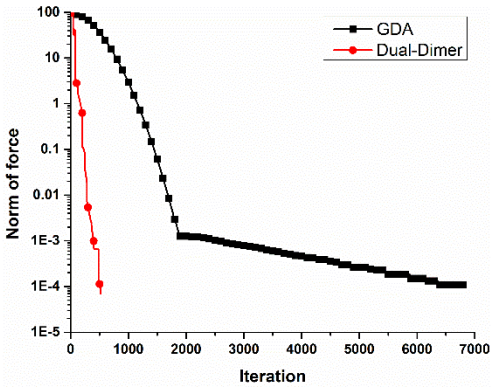
whereas the maximum eigenvalue β_l in the maximum subspace at the saddle point $\mathbf{x}^*$ is negative. It is demonstrated that the high-order saddle points of these nonconvex-nonconcave analytical functions can be found by the Dual-Dimer method.

Table 6.3. High-order saddle points found by the GDA and Dual-Dimer method

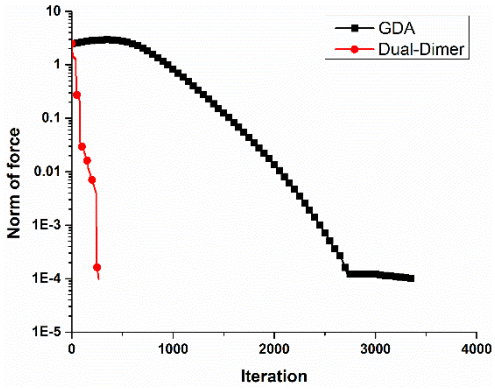
	4D Rastrigin function	4D Ackley function	20D Styblinski–Tang function
Saddle point $\mathbf{x}^*$	$\begin{pmatrix} -0.9950 \\ -0.9950 \\ 0.5025 \\ 0.5025 \end{pmatrix}$	$\begin{pmatrix} 0.9532 \\ 0 \\ -2.6489 \\ 0.5255 \end{pmatrix}$	$x_i = \begin{cases} -2.9035 & i = 1,2,3,4,6,7,10 \\ 2.7468 & i = 5,8,9 \\ 0.1567 & i = 11\sim20 \end{cases}$ (GDA) $x_i = \begin{cases} -2.9035 & i = 1,2,5,6 \\ 2.7468 & i = 2,3,7,8,9,10 \\ 0.1567 & i = 11\sim20 \end{cases}$ (Dual-Dimer)
True minimum eigenvalue β_s^* in the minimum subspace	$\nabla^2_{x_i=-0.9950}E(\mathbf{x}^*) = 396.53$	10.64	$\nabla^2_{x_i=2.7468}E(\mathbf{x}^*) = 29.30$
True maximum eigenvalue β_l^* in the maximum subspace	$\nabla^2_{x_i=0.5025}E(\mathbf{x}^*) = -392.62$	−8.18	$\nabla^2_{x_i=0.1567}E(\mathbf{x}^*) = -15.85$
Calculated minimum eigenvalue β_s in the minimum subspace by Dual-Dimer	396.53	10.83	29.30
Calculated maximum eigenvalue β_l in the maximum subspace by Dual-Dimer	−392.62	−8.13	−15.85

In addition, Figure 6.2 shows the changes in the forces or gradients for the two methods during the search for saddle points of the three analytical functions. It is seen that the force for the Dual-Dimer method decreases faster than the GDA method. The results

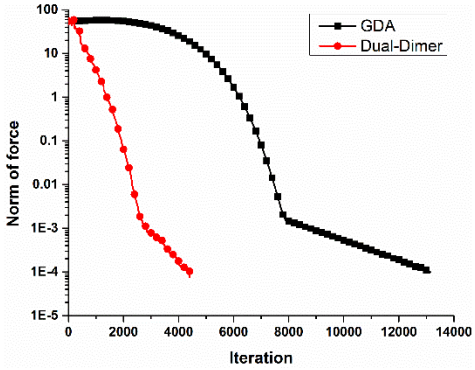
show that the Dual-Dimer method is computationally more efficient than the GDA method to find these high-order saddle points. Table 6.4 shows the quantitative comparison of the convergence between the GDA and Dual-Dimer methods. The convergence speeds of the Dual-Dimer method are about 10 times, 9 times, and 2 times faster than those of the GDA method for the Rastrigin, Ackley, and Styblinski–Tang functions, respectively.



(a)



(b)



(c)

Figure 6.2. The change in the force during the search for saddle points of (a) a 4D Rastrigin function, (b) a 4D Ackley function, and (c) a 20D Styblinski–Tang function.

Table 6.4. Comparison of convergence speeds of the GDA and Dual-Dimer methods

Methods	4D Rastrigin function		4D Ackley function		20D Styblinski–Tang function	
	Training iteration	Training time (second)	Training iteration	Training time (second)	Training iteration	Training time (second)
GDA	6840	6.56	3366	3.23	13136	44.70
Dual-Dimer	**522**	**0.58**	**265**	**0.31**	**4403**	**16.55**

6.4 Demonstration of PCNN-MM

In this section, a heat transfer example is used to demonstrate the increased computational efficiency of PCNNs by adopting the new minimax architecture. The problem description of the heat transfer example is shown in Section 5.2.3.1. In the heat transfer problem, the evolution of the 2D temperature distribution is predicted by a PCNN with the adaptive weighting scheme, a PCNN-MM trained by the GDA method, and a PCNN-MM trained by the Dual-Dimer method. The PCNN setup is described in Section 6.4.1. The computational results and a quantitative comparison for different models are provided in Section 6.4.2. The convergence speed and stability of different models are also investigated.

6.4.1 Computational Setup

The construction of the PCNN and PCNN-MMs is accomplished by using PyTorch [218], which is an open-source Python library for machine learning. The PCNN and PCNN-MMs have the same structure of 30-20-30-20, where each network has 4 layers and the numbers of neurons in these layers are 30, 20, 30, and 20 respectively. The neural

network architecture was identified by conducting some simple sensitivity studies. The hyperbolic tangent (tanh) function is used as the activation function.

The training data for the heat transfer example come from the FEM solutions by *COMSOL*. The simulation domain is $x, y \in [0,1]$ and the time period is $t \in [0,1]$. The training data and physical constraints are sampled uniformly in both temporal and spatial dimensions. The amount of training data is $N_T = 21 \times 6 \times 6 = 756$, which means that there are 21 sampling points in the temporal dimension, 6 sampling points in the *x*-direction, and 6 in the *y*-direction of the spatial domain. In other words, the grid spacing is $\Delta x = 0.2$ and the time step is $\Delta t = 0.05$ in the FEM solution. The number of physical constraints is $21 \times 11 \times 11 = 2541$, where the grid spacing is $\Delta x = 0.1$ and the time step is $\Delta t = 0.05$ for physical constraints. The numbers of sampling points corresponding to the physical loss, initial loss, and boundary loss are $N_P = 1620$, $N_I = 121$, and $N_S = 800$ respectively, which sum up to 2541. Once the training is finished, the temperature at $t = 1$ will be predicted from different models with a grid spacing of $\Delta x = 0.04$, which is finer than the grid spacings of the training data and physical constraints.

Both GDA and Dual-Dimer methods are used to search high-order saddle points for the PCNN-MM formulation. The gradient descent ascent steps in the GDA and Dual-Dimer method are given by the Adam algorithm with the learning rate of 5×10^{-4}. The same Adam algorithm with the learning rate of 5×10^{-4} is used to minimize the total loss during the training of a PCNN. The dimer distance is $2\Delta R = 2 \times 10^{-4}$. The hyperparameters for the Dual-Dimer method are listed in Table 6.5. In the heat transfer example, the search for a saddle point stops when the total loss is less than the threshold ($E < \varepsilon$). This is because that the total loss could still be large when the norm of the force is small in the heat transfer

example. In the heat transfer example, if the true solution u is found, then the total loss E becomes zero. That is the reason that $E < \varepsilon$ is used as the criteria to determine whether a good prediction to approximate the true solution is found.

Table 6.5. Hyperparameters of the Dual-Dimer method in the heat transfer example

Hyperparameters	Value
Frequency of updating extreme eigenvalues and eigenvectors, p	40
The parameter to avoid the zero-division error, δ	1×10^{-3}
Maximum step length of $\Delta\boldsymbol{\theta}_s$ and $\Delta\boldsymbol{\theta}_l$, γ	1×10^{-5}
Learning rate for the gradient descent ascent sub-steps, η	5×10^{-4}
The threshold for stopping search ($E < \varepsilon$), ε	1×10^{-3}

6.4.2 Computational Results

The predicted temperature fields from different models at $t = 1$ are shown in Figure 6.3. The dots in the figures represent the evaluation positions of the temperature field in the 2D domain, where a total of 26×26 samples are taken. It is observed that the predicted temperature fields from the PCNN and PCNN-MMs are close to the FEM solution.

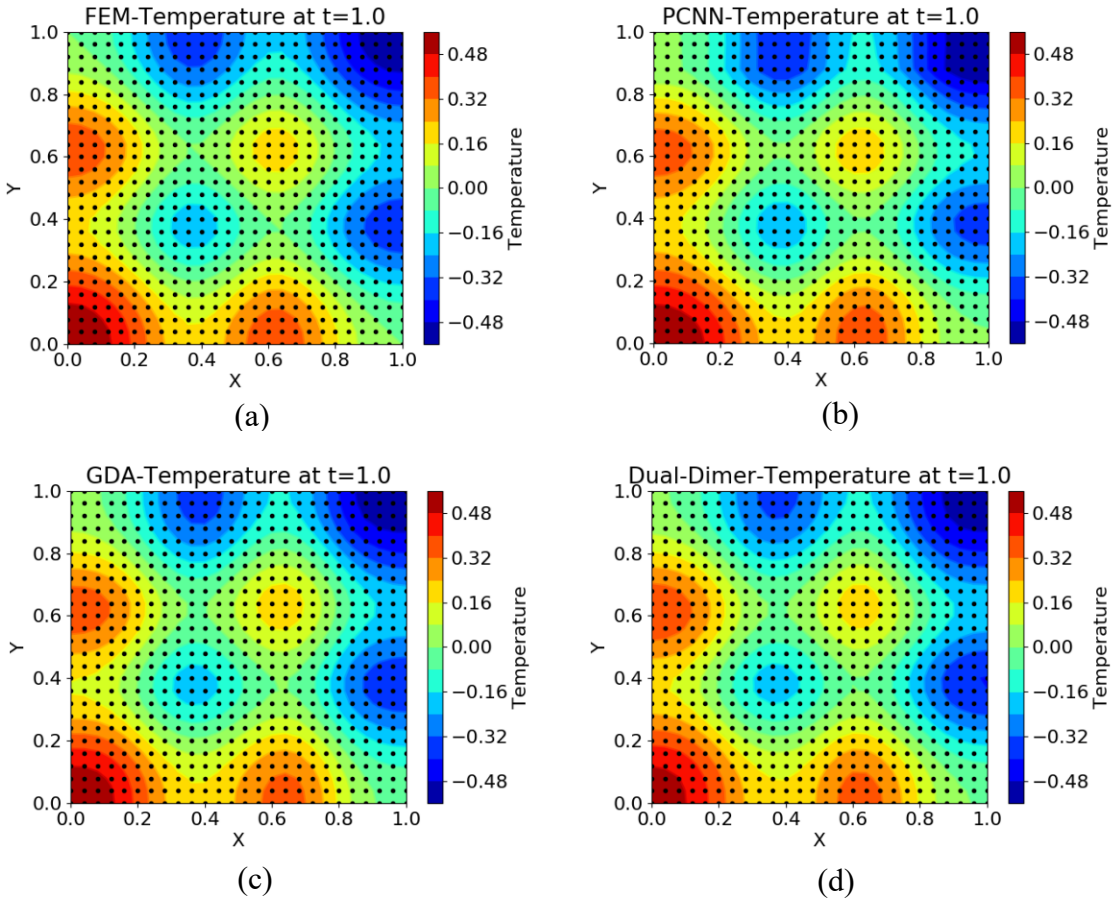


Figure 6.3. The predicted temperature fields from different models at $t = 1$: (a) the original FEM solution, (b) the PCNN with the adaptive weighting scheme, (c) the PCNN-MM trained by the GDA method, and (d) the PCNN-MM trained by the Dual-Dimer method.

The changes in losses and weights for different models during the training process are shown in Figure 6.4. In general, most losses for different models monotonically decrease during the training. The total loss is less than the desired threshold at the end of the training. However, the convergence speeds of PCNN-MMs are greater than that of the PCNN because the problem formulations are different. The training of the PCNN is to solve the minimization problem, whereas the training of the PCNN-MM is to solve the minimax problem. Note that in the training of the PCNN and PCNN-MMs, the relative

importance of the training data and prior knowledge in the total loss function is adjusted dynamically by changing the weights of different losses. As shown in Figure 6.4(c), the weights of the PCNN are adjusted dynamically based on the percentages of individual losses in the total loss function. Therefore, a larger weight will be assigned to a larger loss term. As shown in Figure 6.4(a), different losses converge at the same speed in the later training stage of the PCNN when different losses have the same magnitude. In the training of PCNN-MMs, the weights of different losses are adjusted dynamically to maximize the total loss. Similarly, a larger weight is assigned to a larger loss term. As shown in Figure 6.4(b) and Figure 6.4(d), the initial loss is high, whereas the physical loss is low in the early training stage of the PCNN-MM. Therefore, the weight of the initial loss increases, whereas the weight of the physical loss decreases. By minimizing the possible maximum total loss, the convergence speed of the PCNN-MM increases. The changes in losses and weights for different PCNN-MMs are similar because the maximum step lengths of $\boldsymbol{\Delta\theta}_s$ and $\boldsymbol{\Delta\theta}_l$ are small to avoid divergence. By using the information of extreme eigenvalues, the convergence speed of the PCNN-MM trained by the Dual-Dimer method is slightly higher than that of the PCNN-MM trained by the GDA method. Note that the purpose of using the extreme eigenvalues and eigenvectors in the Dual-Dimer method is not to accelerate the convergence, but to make sure that the high-order saddle points are found.

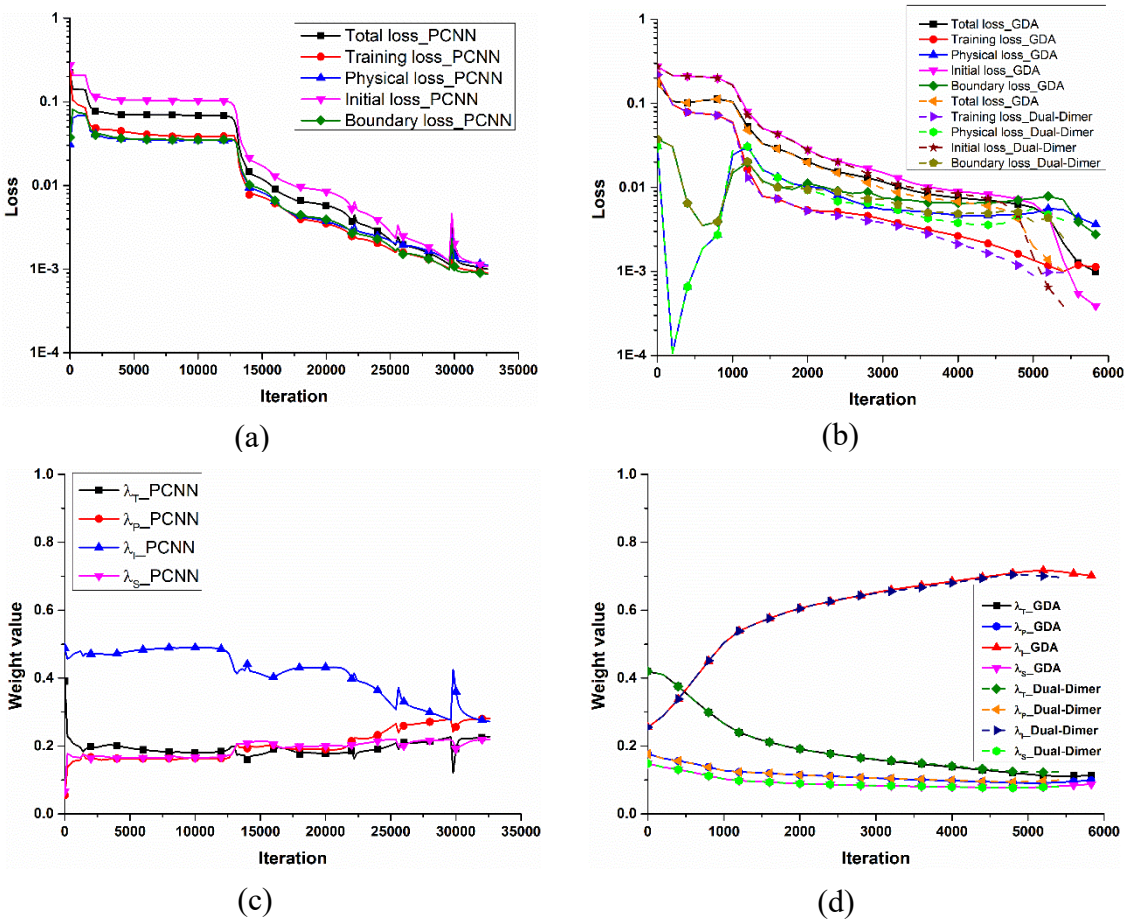


Figure 6.4. The changes in losses and weights for different models during the training process: (a) losses of the PCNN, (b) losses of PCNN-MMs, (c) weights of the PCNN, and (d) weights of PCNN-MMs.

The changes in the forces and eigenvalues during the training of PCNN-MMs are shown in Figure 6.5. As is shown in Figure 6.5(a), the total loss can still be large when the norm of the force is small during the training process. That is the reason that $E < \varepsilon$ is used as the criteria to determine whether a good prediction is found. At the end of the training, the forces for both PCNN-MMs are close to zero, meaning that a critical point is found. Note that eigenvalues are not directly provided by the GDA method. The eigenvalues shown in Figure 6.5(b) and Figure 6.5(c) are recalculated by the Dual-Dimer method. At the end of the training, the minimum eigenvalue β_s in the $\mathbf{w}$ subspace is positive and

maximum eigenvalue β_l in the $\boldsymbol{\alpha}$ subspace is close to zero. This means that the desired high-order saddle point is found. The results demonstrate the effectiveness of the proposed Dual-Dimer method.

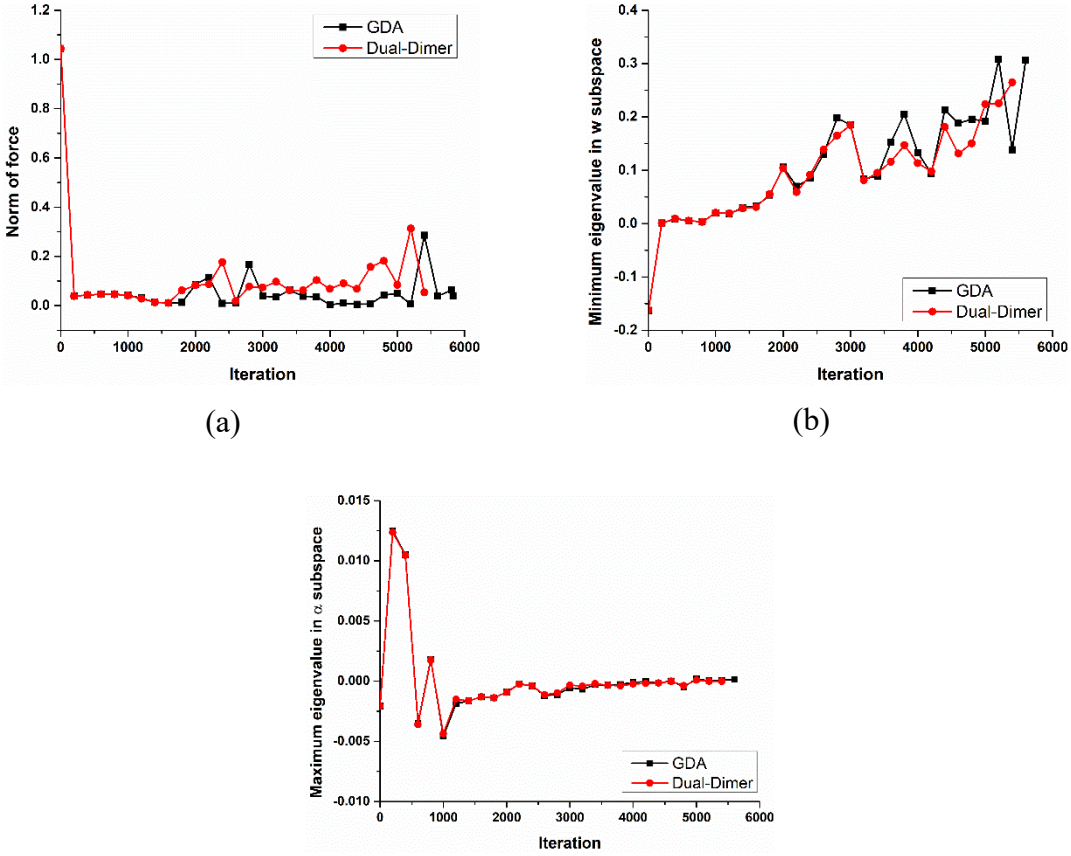


Figure 6.5. Forces and eigenvalues during the training of PCNN-MMs: (a) norm of force, (b) minimum eigenvalue in the $\mathbf{w}$ subspace, and (c) maximum eigenvalue in the $\boldsymbol{\alpha}$ subspace.

To test the convergence speed and stability of different models, the PCNN and PCNN-MMs were run 20 times with random initial weights of neural networks. The mean values of training iterations, training time, and mean squared error (MSE) for different models are shown in Table 6.6, where their standard deviations are also shown in

parentheses. Figure 6.6 shows that the convergence speeds of PCNN-MMs are about 3 times faster than that of the PCNN, whereas the MSEs of predictions by PCNN-MMs at $t = 1$ are slightly larger than that by the PCNN. The MSEs of predictions by the PCNN and PCNN-MMs are all less than the error threshold $\varepsilon = 1 \times 10^{-3}$ with negligible differences. The results show the increased computational efficiency of PCNNs by adopting the new minimax architecture. The standard deviations of the training iterations and training time by PCNN-MMs are less than those by the PCNN, whereas the standard deviations of the MSEs of prediction by PCNN-MMs at $t = 1$ are slightly larger than that by the PCNN. The results also indicate the stability of the proposed PCNN-MMs. The training times of the PCNN-MMs by the GDA method and the Dual-Dimer method are similar. However, the Dual-Dimer method can provide additional eigenvalue information to make sure that the desired high-order saddle points are found at the end of the training.

The above computational results demonstrate that PCNN-MMs are computationally more efficient in training than the original PCNN with the adaptive weighting scheme. The proposed minimax architecture has the advantage of systematically adjusting the weights of different losses. The results also show that the local convergence of PCNN-MMs is stable. In addition, with the similar accuracy and efficiency of the GDA method, the Dual-Dimer method can provide additional eigenvalue information to make sure that the desired saddle points are found at the end of the training.

Table 6.6. Quantitative comparison for different models to solve the heat transfer problem

Models	Training iteration	Training time (second)	MSE of prediction at $t = 1$	Minimum eigenvalue β_s in the $\mathbf{w}$ subspace at the end of the training	Maximum eigenvalue β_l in the $\boldsymbol{\alpha}$ subspace at the end of the training
PCNN with the adaptive weighting scheme	58497 (24878)	2259.46 (930.81)	3.24×10^{-4} (1.62×10^{-4})	N/A	N/A
PCNN-MM with the GDA method	15322 (7023)	614.72 (247.48)	4.22×10^{-4} (3.72×10^{-4})	0.95 (0.78)	5.84×10^{-5} (8.19×10^{-5})
PCNN-MM with the Dual-Dimer method	13376 (6035)	560.85 (246.08)	5.56×10^{-4} (4.13×10^{-4})	0.71 (0.53)	-6.91×10^{-5} (1.77×10^{-4})

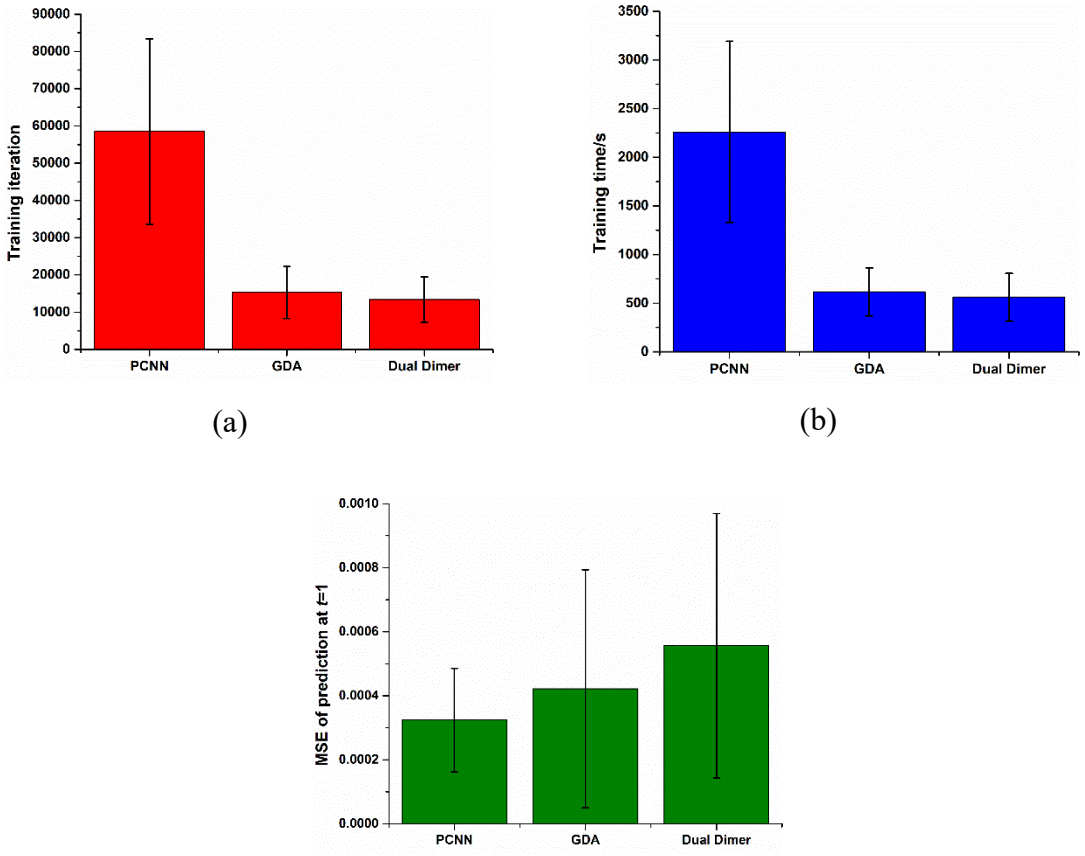


Figure 6.6. Quantitative comparison for different models in (a) training iteration, (b) training time, and (c) MSE of prediction at $t = 1$.

6.5 Demonstration of MF-PCNN-MM

Three examples are used to demonstrate the proposed MF-PCNN-MM framework. The first example is a heat transfer problem described in Section 5.2.3.1. The heat transfer example is used to investigate the effects of the amount of HF training data and the number of physical constraints on prediction accuracy. The second example is a phase transition problem described in Section 5.2.3.2. The phase transition example is used to illustrate the effectiveness of the MF-PCNN-MM framework. The third example is a dendritic growth

problem described in Section 5.2.3.3. In the dendritic growth example, the liquidus temperature is $q_e = 1$ and latent heat is $K = 2$. The goal is to demonstrate the applicability of the proposed MF-PCNN-MM framework for complex multiphysics problems.

6.5.1 Computational Setup

The construction of different models, including the LF-PCNN-MM, MF-NN-MM, and MF-PCNN-MM, is implemented by using PyTorch [218]. The numbers of input and output variables vary with different examples. However, the structures of the hidden layers of different models are consistent in three examples. The structure of the LF-PCNN-MM is 30-20-30-20. The MF-NN-MM and MF-PCNN-MM have the same architecture, which is composed of a low-fidelity ANN and a high-fidelity ANN. The structure of the low-fidelity ANN is 30-20-30-20, whereas the structure of the high-fidelity ANN is 20-20. The neural network architectures were identified by running some simple sensitivity studies. The tanh function is used as the activation function. The learning rate is 5×10^{-4}.

The training data come from the FEM solutions by *COMSOL*. In this work, the LF training data were taken from the FEM results with low resolutions, whereas the HF training data were taken from the FEM results with high resolutions. Notice that the LF and HF data do not necessarily form a nested hierarchy for both spatial and time domains. The physical constraints for all examples are sampled uniformly in both temporal and spatial dimensions. All LF data are used for training for the heat transfer and phase transition example. Random sampling is used to obtain the HF training data for the heat transfer and phase transition example, whereas random sampling is used to obtain the LF and HF training data for the dendritic growth example.

As shown in Table 6.7, the amount of LF training data in the heat transfer example is $N_L = 396$. The LF training data come from the FEM simulation where the time step is $\Delta t = 0.1$ and the grid spacing is $\Delta x = 0.2$. The HF training data are randomly sampled from the FEM simulation with a time step $\Delta t = 0.1$ and finer grid spacing $\Delta x = 0.1$. For the LF-PCNN-MM and MF-PCNN-MM, physical constraints are added with the finest grid spacing $\Delta x = 0.05$. The training of all models stops when the mean loss

$$E_{mean} = \frac{1}{n}\sum_{i} E_i \, , i \in \{L, H, P, I, S\} \tag{6.35}$$

is lower than a threshold value 10^{-4}. During the training process, the weights of different losses $\lambda_i(\boldsymbol{\alpha})$ are not the same. By using $E_{mean} < 10^{-4}$ as the stopping criterion, the performance of different models can be fairly compared.

Table 6.7. The setup for different ML models in the heat transfer example

ML model	Amount of LF training data	Amount of HF training data	Number of physical constraints
MF-NN-MM	396 ($\Delta t = 0.1$, $\Delta x = 0.2$)	67, 133, 199, 266, 332 (random, $\Delta t = 0.1$, $\Delta x = 0.1$)	N/A
LF-PCNN-MM	396 ($\Delta t = 0.1$, $\Delta x = 0.2$)	N/A	4851 ($\Delta t = 0.1$, $\Delta x = 0.05$)
MF-PCNN-MM	396 ($\Delta t = 0.1$, $\Delta x = 0.2$)	67, 133, 199, 266, 332 (random, $\Delta t = 0.1$, $\Delta x = 0.1$)	216 ($\Delta t = 0.2$, $\Delta x = 0.2$), 726 ($\Delta t = 0.2$, $\Delta x = 0.1$), 1331 ($\Delta t = 0.1$, $\Delta x = 0.1$), 4851 ($\Delta t = 0.1$, $\Delta x = 0.05$)

Table 6.8 shows the setup for different ML models in the phase transition example. Similarly, the training of all models stops when the mean loss E_{mean} is lower than a threshold value 10^{-4}.

Table 6.8. The setup for different ML models in the phase transition example

ML model	Amount of LF training data	Amount of HF training data	Number of physical constraints
MF-NN-MM	396 ($\Delta t = 0.1, \Delta x = 0.2$)	332 (random, $\Delta t = 0.1, \Delta x = 0.1$)	N/A
LF-PCNN-MM	396 ($\Delta t = 0.1, \Delta x = 0.2$)	N/A	4851 ($\Delta t = 0.1, \Delta x = 0.05$)
MF-PCNN-MM	396 ($\Delta t = 0.1, \Delta x = 0.2$)	332 (random, $\Delta t = 0.1, \Delta x = 0.1$)	4851 ($\Delta t = 0.1, \Delta x = 0.05$)

In the dendritic growth example, as shown in Table 6.9, the training data for the LF-PCNNs are sampled randomly from the FEM simulation, where the time step is $\Delta t = 0.2$ and the grid spacing is $\Delta x = 0.125$. The training data for the HF-PCNNs are sampled randomly from the FEM simulation, where the time step is $\Delta t = 0.1$ and the grid spacing is $\Delta x = 0.1$. the training of all models stops when the mean loss E_{mean} is lower than a threshold value 5×10^{-4}.

Table 6.9. The setup for different ML models in the dendritic growth example

ML model	Amount of LF training data	Amount of HF training data	Number of physical constraints
MF-NN-MM	1512 (random, $\Delta t = 0.2, \Delta x = 0.125$)	1144 (random, $\Delta t = 0.1, \Delta x = 0.1$)	N/A
LF-PCNN-MM	1512 (random, $\Delta t = 0.2, \Delta x = 0.125$)	N/A	28611 ($\Delta t = 0.1, \Delta x = 0.1$)
MF-PCNN-MM	1512 (random, $\Delta t = 0.2, \Delta x = 0.125$)	1144 (random, $\Delta t = 0.1, \Delta x = 0.1$)	28611 ($\Delta t = 0.1, \Delta x = 0.1$)

6.5.2 Computational Results

6.5.2.1 Heat Transfer Example

In the heat transfer example, a sensitivity study is conducted to investigate the impact of amounts of training data and physical constraints on the prediction accuracy. The predicted temperature fields from different models at t = 1 are shown in Figure 6.7. It is seen that the predicted temperature field from the MF-PCNN-MM agrees best with the FEM solution. As shown in Figure 6.8, the MF-NN-MM and the MF-PCNN-MM are compared with various amounts of high-fidelity training data ranging from 5% to 25% at t=0 and t=1. Meanwhile, the MF-PCNN-MM also varies with different amounts of physical constraint (the values of $t \times x \times y$ range from $6 \times 6 \times 6$ to $11 \times 21 \times 21$). It is observed that the MSEs of temperature fields decrease when the amount of HF training data or physical constraint increases. For all ML models except for the MF-PCNN-MM with the physical constraint of $6 \times 6 \times 6$, the MSEs of temperature fields reach a plateau when the amount of HF data is more than 10%. When the amount of physical constraint is more than $6 \times 11 \times 11$, the MSEs of temperature fields from MF-PCNN-MMs are lower than that of the MF-NN-MM. It is noted that the effect of increasing the amount of HF data to improve the prediction accuracy decreases when the number of physical constraints increases.

As shown in Figure 6.9, the LF-PCNN-MM and the MF-PCNN-MM are compared with various amounts of physical constraint at t=0 and t=1. The MF-PCNN-MM also varies with the amount of high-fidelity data from 5% to 25%. Notice again as the amount of HF data increases, the increase in physical constraints starts to play a smaller role in improving the prediction accuracy. Figure 6.8 and Figure 6.9 suggest that the effects of increasing physical constraint and high-fidelity data to improve the prediction accuracy decrease when

the amounts surpass a certain value. A sensitivity study can be used to find a good combination of the amount of HF data and physical constraints to find the tradeoff between computational efficiency and accuracy. In this example, when the amount of HF data is 10% and the amount of physical constraint is $6 \times 11 \times 11$, it reaches the balance between computational efficiency and accuracy.

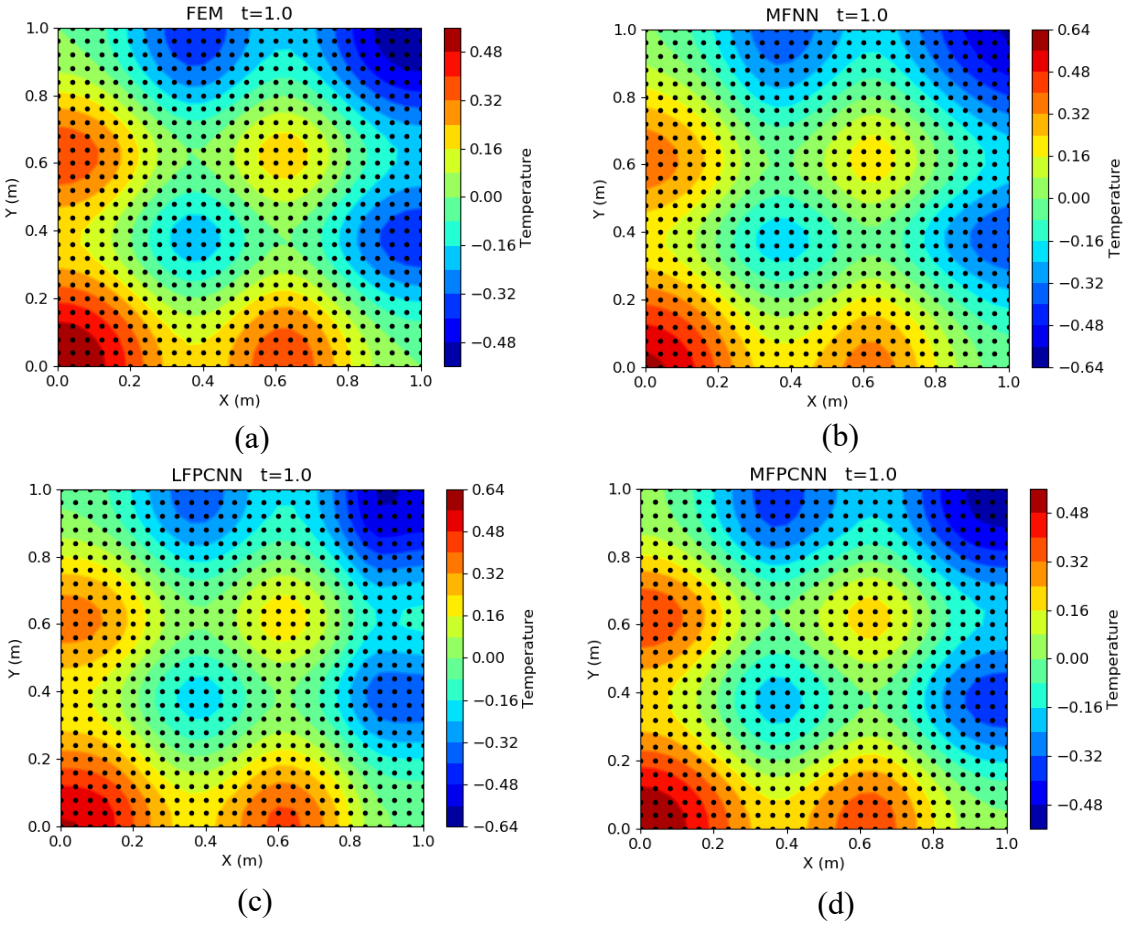
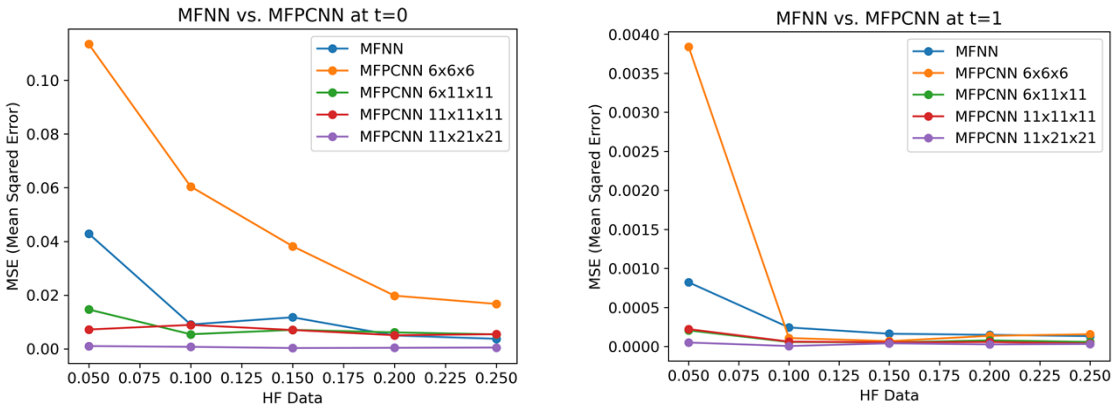
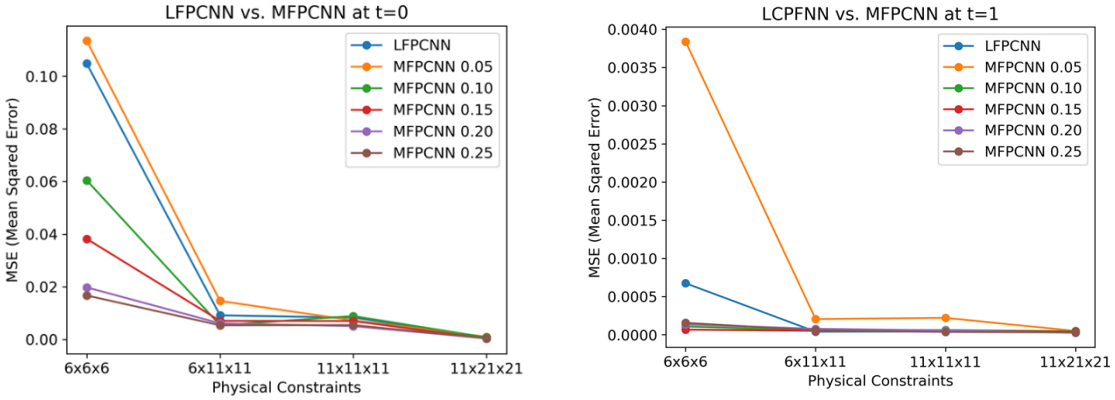


Figure 6.7. The predicted temperature fields from different models at t = 1: (a) original FEM solution, (b) MF-NN-MM, (c) LF-PCNN-MM, and (d) MF-PCNN-MM.



(a) (b)

Figure 6.8. The MSEs of predicted temperature fields from different models with various amounts of HF training data (a) at t = 0 and (b) t = 1.



(a) (b)

Figure 6.9. The MSEs of predicted temperature fields from different models with various amounts of physical constraints (a) at t = 0 and (b) t = 1.

6.5.2.2 Phase Transition Example

In the phase transition example, the predicted phase fields from different models at t = 1 are shown in Figure 6.10. For this nonlinear PDE problem, the predictions from the LF-PCNN-MM and the MF-PCNN-MM are more accurate than that from the MF-NN-MM with the guidance of physical constraints. It is observed that the predicted phase field from

the MF-PCNN-MM agrees best with the FEM solution. The quantitative comparison between different ML models in the phase transition example is shown in Table 6.10. At t = 0, the MSE of prediction from the LF-PCNN-MM is two magnitudes lower than that from the MF-NN-MM, whereas the MSE of prediction from the MF-PCNN-MM is lower than that from the LF-NN-MM. At t = 1, the MSE of prediction from the LF-PCNN-MM is one magnitude lower than that from the MF-NN-MM, whereas the MSE of prediction from the MF-PCNN-MM is one magnitude lower than that from the LF-NN-MM. This demonstrates the effectiveness of the developed MF-PCNN-MM model. The physical constraint and HF data can further improve the prediction accuracy. The learning curves for different models are shown in Figure 6.11. The training loss of HF data is higher than the training loss of LF data in the MF-NN-MM, whereas the training loss of HF data fluctuates in the LF-PCNN-MM and the MF-PCNN-MM. This suggests that it is difficult to minimize the training loss of HF data for approximating accurately the mapping relationship between the LF data and HF data.

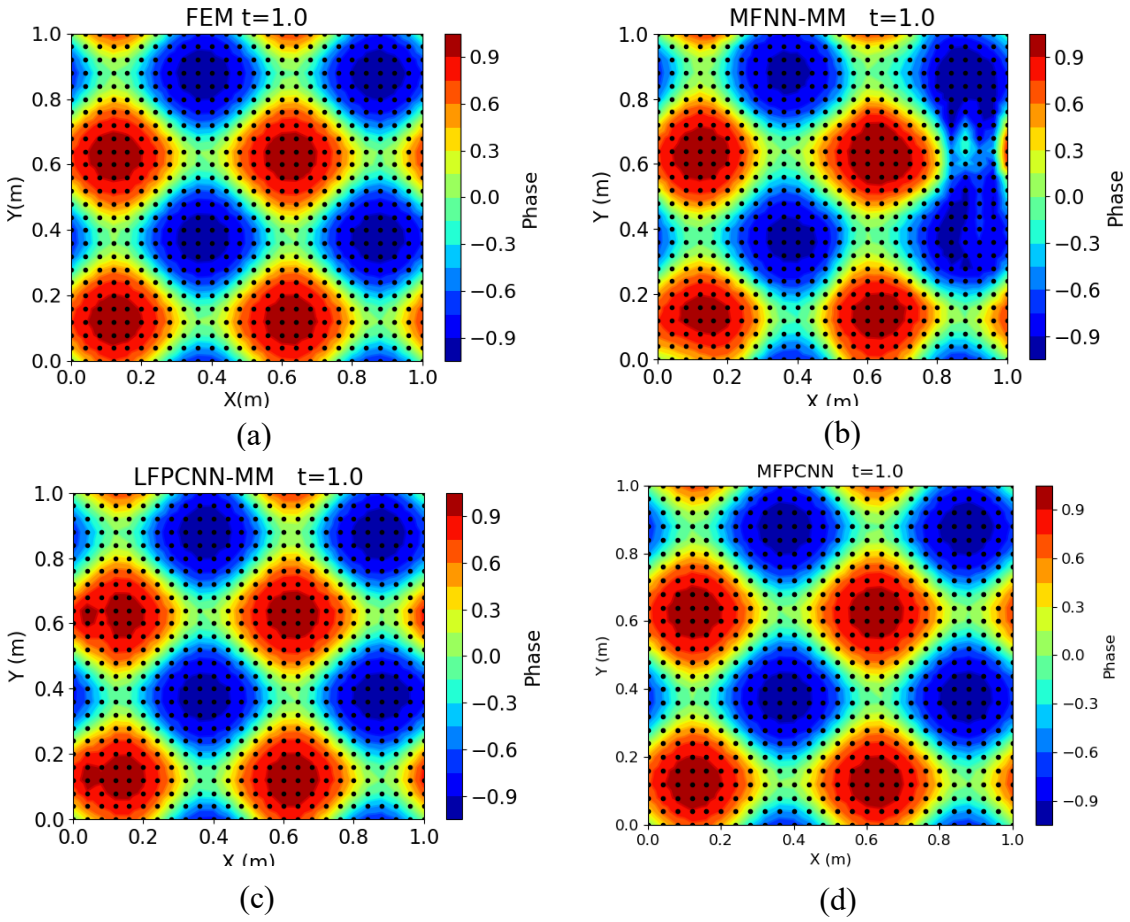


Figure 6.10. The predicted phase fields from different models at t = 1: (a) original FEM solution, (b) MF-NN-MM, (c) LF-PCNN-MM, and (d) MF-PCNN-MM.

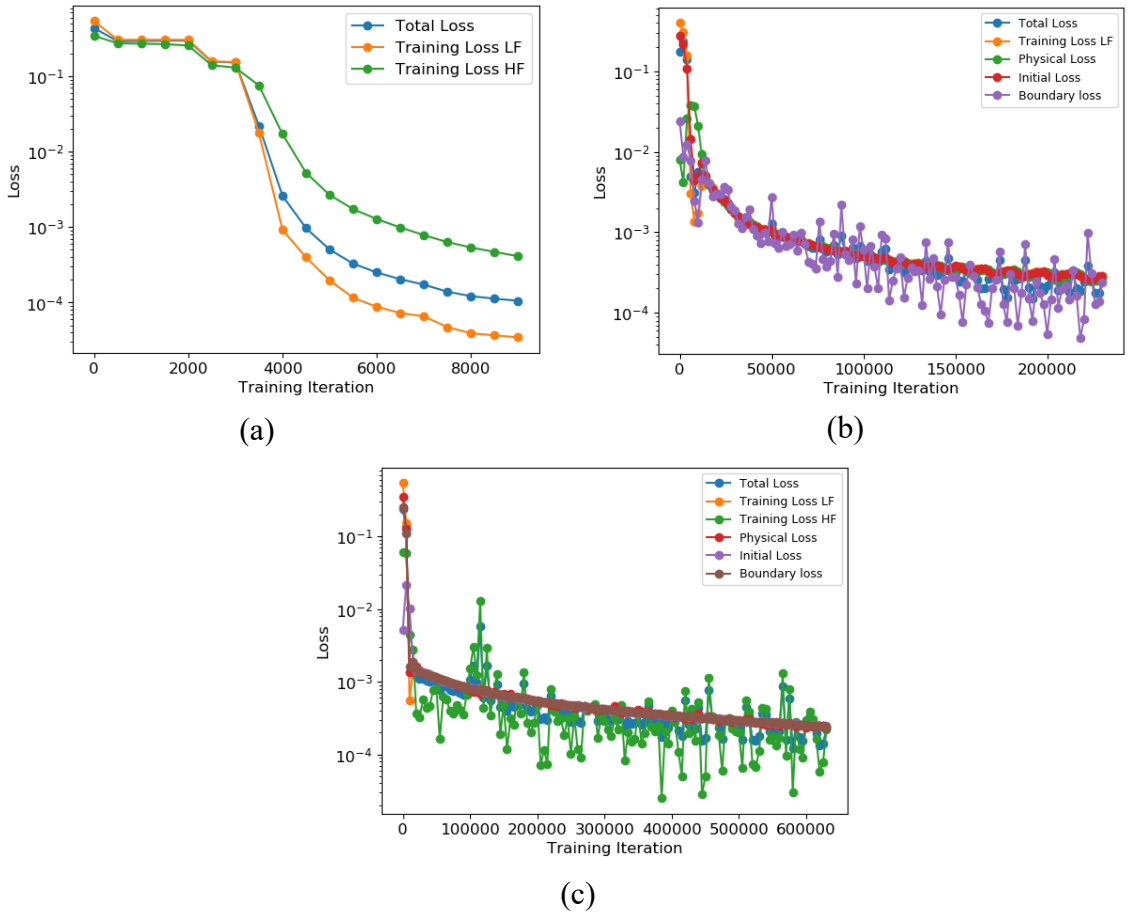


Figure 6.11. The learning curves for different models: (a) MF-NN-MM, (b) LF-PCNN-MM, and (c) MF-PCNN-MM.

Table 6.10. The quantitative comparison between different ML models in the phase transition example

ML model	Training time/s	MSE of prediction at t = 0	MSE of prediction at t = 1
MF-NN-MM	100.60	0.021062	0.018291
LF-PCNN-MM	646335.14	0.000153	0.001725
MF-PCNN-MM	739397.99	0.000125	0.000217

6.5.2.3 Dendritic Growth Example

In the dendritic growth example, the predicted phase fields and temperature fields from different models at $t = 1$ are shown in Figure 6.12. The quantitative comparison between different ML models is shown in Table 6.11. All ML models can predict the primary arms of the dendrite very well, but they cannot reveal the detailed secondary arms. The predicted phase fields and temperature fields from the LF-PCNN-MM and MF-PCNN-MM are slightly more accurate than those from the MF-NN-MM. The predicted phase fields and temperature fields from the LF-PCNN-MM are similar to those from the MF-NN-MM. Since multiple physics are coupled with each other, the training of the ML model is difficult to converge. Different losses in the total objective function could be in conflict and the gradients of different losses could be unbalanced, both of which can lead to the failure of convergence. This implies that it is still a challenge to solve multiphysics problems, such as dendritic growth, using existing PCNNs and MF-PCNN-MMs. New training algorithms or neural network architectures are needed to solve multiphysics problems with PCNNs.

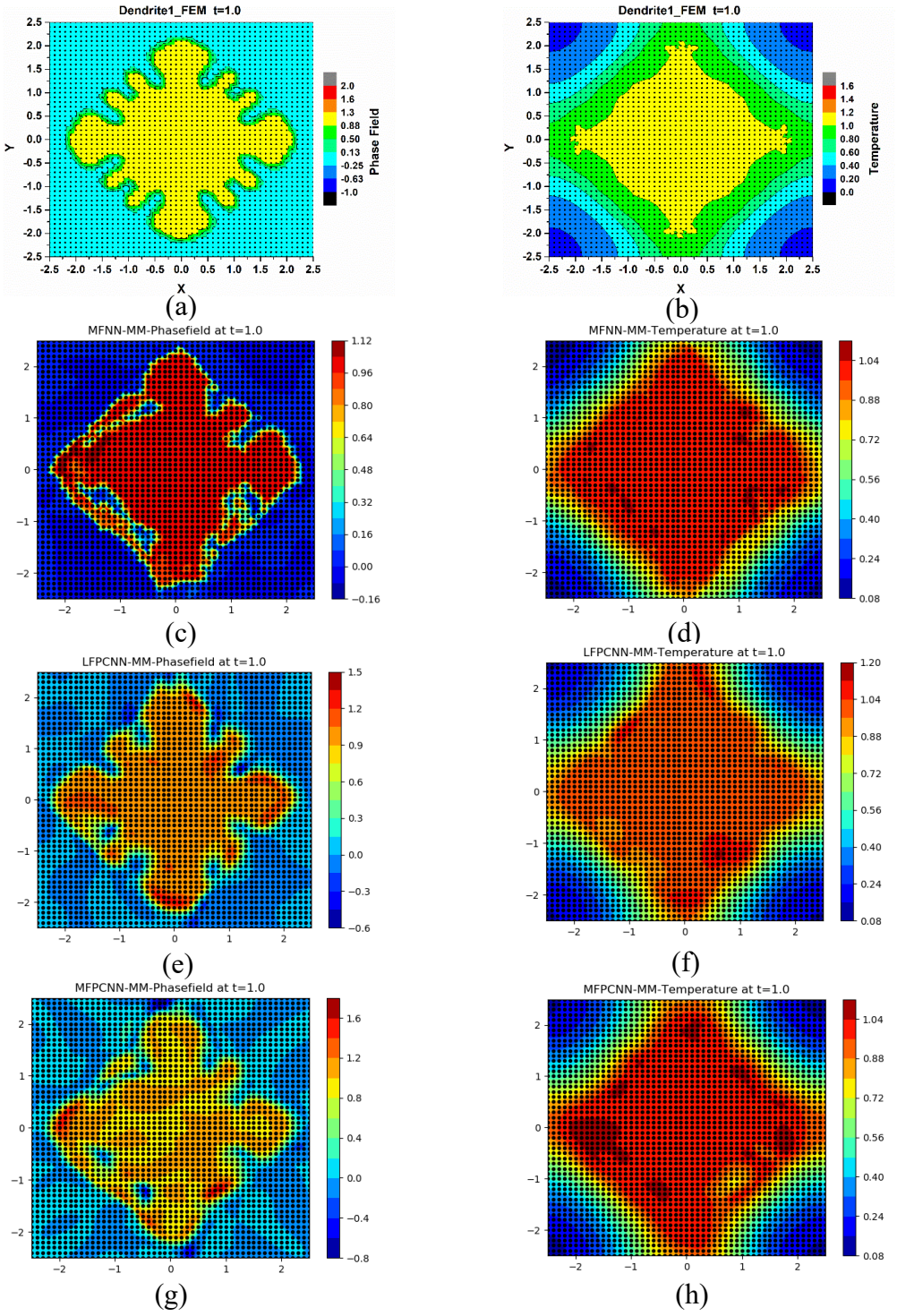


Figure 6.12. The predicted phase fields and temperature fields from different models at t = 1. Phase fields are shown in (a), (c), (e), and (g). Temperature fields are shown in (b), (d), (f), and (h).

Table 6.11. The quantitative comparison between different ML models in the dendritic growth example

ML model	Training time/s	MSE of phase field at t = 1	MSE of temperature at t = 1
MF-NN-MM	490.45	0.062626	0.001322
LF-PCNN-MM	24807.45	0.058481	0.001267
MF-PCNN-MM	136118.80	0.059111	0.001233

6.6 Discussions and Conclusions

In this chapter, a new physics-constrained neural network with the minimax architecture is proposed to adjust the weights of different losses systematically. The training of the PCNN-MM is to solve a minimax problem and search for the high-order saddle points of the nonconvex-nonconcave loss function. To address the challenges of searching high-order saddle points, a novel saddle point search algorithm called Dual-Dimer method is proposed, where only first derivatives need to be calculated. The local convergence of the Dual-Dimer method is analyzed. The performance of the Dual-Dimer method is evaluated with three analytical nonconvex-nonconcave loss functions. It was shown that the Dual-Dimer method is computationally more efficient than the GDA method to find high-order saddle points in these analytical functions. The Dual-Dimer method also provides additional eigenvalue information to make sure that the desired high-order saddle points are found at the end of the training. A heat transfer example is used to demonstrate the effectiveness of the PCNN-MM, where its convergence is faster than that of the original PCNN with the adaptive weighting scheme.

The adjustment of hyperparameters in this study is based on sensitivity studies. In future work, a more systematic method to find the optimal hyperparameters will be developed so that the computational efficiency of the Dual-Dimer method can be further improved. In addition, using more eigenvalues and eigenvectors in the Dual-Dimer method can potentially accelerate the saddle point search. Further investigation is needed. In this chapter, the softmax function is used as the form of the weights of different losses. Though theoretically parameter $\boldsymbol{\alpha}$ can be infinite in the training of the PCNN-MM, the bounds of λ's are $[0,1]$ because of the softmax weighting function. However, the minimax problem could be nonconcave in the $\boldsymbol{\alpha}$ subspace since the softmax function is neither convex nor concave. This increases the difficulty in searching the desired saddle point. The effects of various forms of weighting functions on the training dynamics of the PCNN-MM will be investigated more thoroughly in the future. The training of the PCNN-MM could be more efficient and robust by making the minimax problem to be strongly concave in the subspace of λ's. For instance, the minimax problem will be concave in λ's by using a linear weighting function. The parameters λ's can be directly optimized without using the parameter $\boldsymbol{\alpha}$. The sum of λ's can be set to unity, and a bound of $[0,1]$ can be added for λ's in the training. The minimax problem in the PCNN-MM can even be strongly concave in λ's by adding an entropy regularizer on λ's. In future work, the generic Dual-Dimer method can be applied to solve other minimax problems, which arise from game theory, generative adversarial networks, and robust optimization.

To further reduce the computational cost, the MF-PCNN-MM is developed to integrate the LF and HF data. The MF-PCNN-MM is demonstrated by three examples. The computational results demonstrate the effectiveness of the MF-PCNN-MM. However, the

MF-PCNN-MM cannot solve the multiphysics dendritic growth problem very well. New training algorithms or neural network architectures are needed to solve multiphysics problems with PCNNs. The PCNN-MMs with the new training schemes will be introduced to solve multiphysics problems in CHAPTER 7.

CHAPTER 7. PHYSICS-CONSTRAINED NEURAL NETWORKS WITH MINIMAX ARCHITECTURE FOR MULTIPHYSICS PROBLEMS

7.1 Introduction

Data sparsity is still the main challenge to apply ML models to solve complex scientific and engineering problems. The root cause is the "curse of dimensionality" in training these models. Training algorithms need to explore and exploit in a very high dimensional parameter space to search the optimal parameters for complex models. Although PCNNs [116,118,140–144] have been used to tackle the issue of data sparsity recently, it is still a challenge to solve multiphysics problems using existing PCNNs. Our previous work in Section 5.3.3 and Section 6.5.2.3 also shows that it is difficult to solve multiphysics dendritic growth problems using existing PCNNs.

In this chapter, the PCNN-MM in CHAPTER 6 is extended to solve multiphysics problems. To accelerate the convergence of the training of PCNN-MMs, a new sequential training scheme is proposed. The bottleneck of the training of the PCNN-MM is the high dimensionality of the $\mathbf{w}$ subspace. To further alleviate the curse of dimensionality, the Dual-Dimer algorithm in Section 6.2.2 is extended with compressive sampling to train the PCNN-MM for finding a better local saddle point. This new Dual-Dimer with compressive sampling (DD-CS) algorithm is developed to train the PCNN-MM for solving multiphysics problems.

In the remainder of this chapter, the formulation of the sequential training scheme, compressive sampling, and DD-CS algorithm is shown in Section 7.2. In Section 7.3, the

developed PCNN-MM is demonstrated by a thermal dendritic growth example and a thermo-solutal dendritic growth example. In each example, an ANN is trained using the sequential training scheme. And PCNN-MMs are trained with three training schemes: concurrent training with the Dual-Dimer algorithm, sequential training with the Dual-Dimer algorithm, and sequential training with the DD-CS algorithm. The computational setups and computational results are included.

7.2 Methodology

7.2.1 Sequential Training of PCNN-MMs

Consider a general multiphysics problem involving two physics fields u and v. Our previous work in Section 5.3.3 and Section 6.5.2.3 has shown that it is difficult to train a single neural network to predict multiple outputs u and v since two outputs share the same weights of a single network. Therefore, it will be more flexible to train two neural networks without sharing degrees of freedom to predict u and v, respectively. It is straightforward to train these two neural networks by using the concurrent training scheme. By adopting the formulation of the PCNN-MM, these two neural networks for u and v can be trained concurrently as

$$\mathbf{w}_u^*, \mathbf{w}_v^*, \boldsymbol{\alpha}_u^*, \boldsymbol{\alpha}_v^* = arg \min_{\mathbf{w}_u, \mathbf{w}_v} \max_{\boldsymbol{\alpha}_u, \boldsymbol{\alpha}_v} E(\mathbf{x}, u, v; \mathbf{w}_u, \mathbf{w}_v, \boldsymbol{\alpha}_u, \boldsymbol{\alpha}_v) \tag{7.1}$$

where $\mathbf{x}$ is the input vector, $\mathbf{w}_u$ and $\mathbf{w}_v$ are the respective weights of two neural networks, $\boldsymbol{\alpha}_u$ and $\boldsymbol{\alpha}_v$ are the parameters to adjust the weights of different losses for u and v, respectively. The total loss including the training loss, physical loss, initial loss, and boundary loss are shown in Eq. (6.1). By using the Dual-Dimer algorithm to search high-

order saddle points, the parameters of two neural networks can be updated as $\mathbf{w}_u^*$, $\mathbf{w}_v^*$, $\boldsymbol{\alpha}_u^*$, and $\boldsymbol{\alpha}_v^*$.

Different from the concurrent training scheme, a new sequential training scheme is proposed to training PCNN-MMs in this work. By using the sequential training scheme, two neural networks for u and v in the *i*-th iteration can be trained as

$$\begin{aligned} \mathbf{w}_u^{*\,i+1}, \boldsymbol{\alpha}_u^{*\,i+1} &= arg \min_{\mathbf{w}_u} \max_{\boldsymbol{\alpha}_u} E_u\left(\mathbf{x}, u^i, v^i; \mathbf{w}_u, \boldsymbol{\alpha}_u\right) \\ \mathbf{w}_v^{*\,i+1}, \boldsymbol{\alpha}_v^{*\,i+1} &= arg \min_{\mathbf{w}_v} \max_{\boldsymbol{\alpha}_v} E_v\left(\mathbf{x}, u^{i+1}, v^i; \mathbf{w}_v, \boldsymbol{\alpha}_v\right) \end{aligned} \tag{7.2}$$

where E_u and E_v are the total loss of two neural networks for u and v, respectively. A schematic illustration of the proposed network and the sequential training scheme is shown in Figure 7.1. In the *i*-th iteration, the parameters $\mathbf{w}_u$ and $\boldsymbol{\alpha}_u$ of the neural network for u are optimized first based on the current input $\mathbf{x}$ and outputs u^i, v^i with the parameters $\mathbf{w}_v$ and $\boldsymbol{\alpha}_v$ fixed. Then the parameters $\mathbf{w}_v$ and $\boldsymbol{\alpha}_v$ of the neural network for v are optimized based on the current input $\mathbf{x}$ and outputs u^{i+1}, v^i with the parameters $\mathbf{w}_u$ and $\boldsymbol{\alpha}_u$ fixed. The process is iterated until the convergence of the training. It is noted that the updated output u^{i+1} is used to optimize the parameters of the neural network for v in the *i*-th iteration.

Both concurrent and sequential training schemes are general and can be applied in training PCNN-MMs for solving multiphysics problem which involves more than two physics easily. In the concurrent training scheme, multiple neural networks are trained to predict multiple physical fields simultaneously. In the sequential training scheme, multiple neural networks are trained sequentially to predict multiple physical fields respectively. By using the sequential training scheme, different physics fields are one-way coupled rather than fully coupled, which contributes to the convergence of PCNN-MMs.

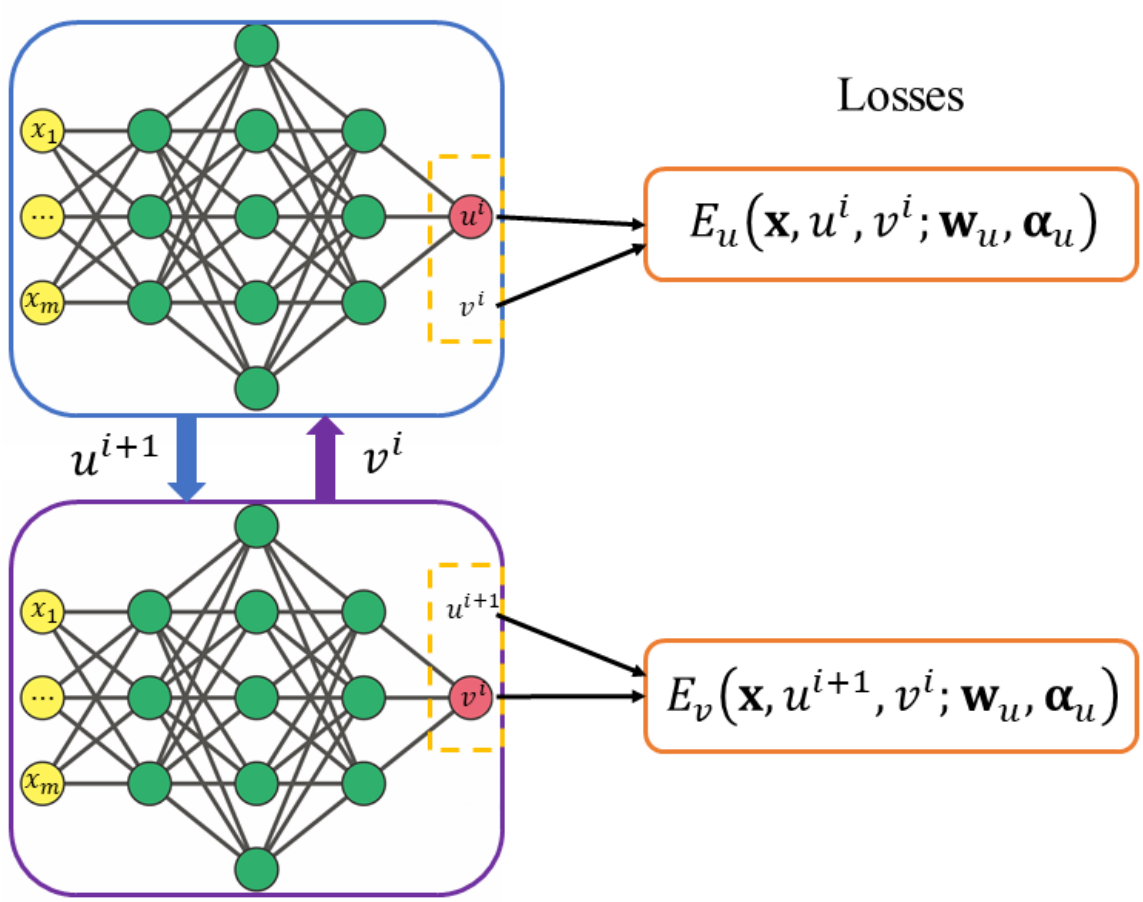


Figure 7.1. Schematic illustration of the proposed network and the sequential training scheme.

7.2.2 Compressive Sampling

Compressive sampling or compressed sensing [219] was initially developed to solve the inverse problem of information recovery purely based on statistical characteristics of signals. It has been widely applied in signal processing, image processing, and others. Let vector $\mathbf{s} \in \mathbb{R}^n$ represent the original signal. The signal can be represented in the reciprocal space via transformation as

$$\mathbf{s} = \boldsymbol{\Psi}\mathbf{z} \tag{7.3}$$

where $\mathbf{\Psi} \in \mathbb{R}^{n\times n}$ is the transformation matrix or basis matrix, $\mathbf{z} \in \mathbb{R}^n$ is the vector of coefficients in the reciprocal space. If k $(k < n)$ elements of $\mathbf{z}$ are nonzero, then $\mathbf{z}$ is k-sparse. The measurement of the signal is done by projecting it to a reduced space as

$$\mathbf{y} = \mathbf{\Phi s} = \mathbf{\Phi\Psi z} = \mathbf{Az} \tag{7.4}$$

where $\mathbf{\Phi} \in \mathbb{R}^{m\times n}$ $(m < n)$ is the projection matrix or measurement matrix. To obtain the sparse solution of $\mathbf{z}$ and recover the original signal from the measurement, a L1 norm regularization term can be introduced in solving the minimization problem

$$\min_{\mathbf{z}} \|\mathbf{y} - \mathbf{Az}\|_2^2 + \lambda\|\mathbf{z}\|_1 \tag{7.5}$$

where λ is the regularization coefficient. Using the L1 norm for regularization is also known as the least absolute shrinkage and selection operator (LASSO). The solution to the problem in Eq. (7.5) can be obtained by using the proximal gradient descent methods, such as the iterative soft thresholding algorithm (ISTA) [220]

$$\mathbf{z}_{t+1} = S_\lambda[\mathbf{z}_t + \mathbf{A}^T(\mathbf{y} - \mathbf{Az})] \tag{7.6}$$

where $S_\lambda(\mathbf{z})$ is the soft thresholding operator as

$$[S_\lambda(\mathbf{z})]_i = \begin{cases} z_i - \lambda, & z_i > \lambda \\ 0, & -\lambda \le z_i \le \lambda \\ z_i + \lambda, & z_i < -\lambda \end{cases} \tag{7.7}$$

7.2.3 The DD-CS Algorithm

As is shown in Section 6.2.1, the training of the PCNN-MM is to solve the minimax problem. The training of the PCNN-MM is divided into three stages in the proposed DD-CS algorithm as shown in Table 7.1. The flowchart of the DD-CS algorithm is shown in Figure 7.2.

In Stage 1, the PCNN-MM is trained in the complete $\mathbf{w}$ subspace and the complete $\boldsymbol{\alpha}$ subspace to reach the desired high-order saddle point by the Dual-Dimer method as in Eq. (6.5). If $\|\Delta\boldsymbol{\theta}_1\|_2 = \|\Delta\boldsymbol{\theta}_{\mathbf{w}} + \Delta\boldsymbol{\theta}_s\|_2 < c_1\rho^{\frac{t}{d}}$, which means that the neighborhood of the local minimum in the $\mathbf{w}$ subspace has been reached, then the training of the PCNN-MM will switch to Stage 2. Here, $c_1\rho^{\frac{t}{d}}$ is an adaptive threshold during the training, where c_1, ρ , and d are hyperparameters, and t is the iteration number.

In Stage 2, the PCNN-MM is trained in the subspace $\mathbf{y} = \boldsymbol{\Phi}\mathbf{w} \in \mathbb{R}^m$ and the complete $\boldsymbol{\alpha}$ subspace, where $\boldsymbol{\Phi} \in \mathbb{R}^{m\times n}$ is the measurement matrix. By reducing the dimension of the $\mathbf{w}$ subspace, there is a higher chance for the search to escape the current local minimum and reach a lower local minimum in the $\mathbf{w}$ subspace. If $\|\Delta\boldsymbol{\theta}_1\|_2 < c_2\rho^{\frac{t}{d}}$, which means the neighborhood of the local minimum in the $\mathbf{y}$ subspace has been reached, then the training of the PCNN-MM will switch to Stage 3.

In Stage 3, the compressive sampling technique in Section 7.2.2 is used to recover the original weight $\mathbf{w}$ from the reduced weight $\mathbf{y}$ can be represented in a reciprocal space as

$$\mathbf{y} = \boldsymbol{\Phi}\mathbf{w} = \boldsymbol{\Phi}\boldsymbol{\Psi}\mathbf{z} = \mathbf{A}\mathbf{z} \tag{7.8}$$

where $\boldsymbol{\Psi} \in \mathbb{R}^{n\times n}$ is the discrete cosine transform matrix, $\mathbf{z} \in \mathbb{R}^n$ is the vector of coefficients in the reciprocal space. To obtain the sparse solution of $\mathbf{z}$, a L1 norm regularization term is introduced to recover the original weight $\mathbf{w}$ by solving the minimization problem in Eq. (7.5). By using the solution in Eq. (7.6), the proximal gradient descent step to update the weight $\mathbf{w}$ is

$$\Delta\boldsymbol{\theta}_4 = \boldsymbol{\Psi}S_{\boldsymbol{\lambda}}[\boldsymbol{\Psi}^{-1}\mathbf{w} + \mathbf{A}^T(\boldsymbol{y} - \boldsymbol{\Phi}\mathbf{w})] - \mathbf{w} \tag{7.9}$$

If $\|\Delta\boldsymbol{\theta}_4\|_2 < \zeta$, where ζ is a threshold, then the recovery of the original $\mathbf{w}$ is completed and the training of the PCNN-MM will switch back to Stage 1. Therefore, the training iterations with the three-stage cycle will continue. Since the adaptive thresholds $c_1\rho^{\frac{t}{d}}$ and $c_2\rho^{\frac{t}{d}}$ gradually decrease as the iterations continue, the chance to switch to Stages 2 and 3 is getting lower. Then the PCNN-MM will be eventually trained in the original $\mathbf{w}$ subspace and the search will gradually converge to the desired saddle point.

There are thirteen hyperparameters (p, δ, γ, η, ρ, c_1, c_2, d, m_0, t_b, λ, ζ, t_m) that need to be tuned in the DD-CS algorithm. When the iteration number is larger than the maximum iteration number t_m, the training of the PCNN-MM stops. c_1 and c_2 are the prefactors of the adaptive thresholds $c_1\rho^{\frac{t}{d}}$ and $c_2\rho^{\frac{t}{d}}$, respectively. ρ and d control the exponential decay rate of the adaptive thresholds. If the adaptive thresholds are too large, then the training of the PCNN-MM will switch to Stage 2 and 3 more frequently, which may cause instability in the training process. If the adaptive thresholds are too small, then the training of the PCNN-MM may never switch to Stage 2 and 3, which cannot help to find a better local saddle point. ζ is a threshold to determine when the training of the PCNN-MM will switch from Stage 3 to Stage 1. If ζ is too large, then the recovery error of the original $\mathbf{w}$ will be large. If ζ is too small, then the training of the PCNN-MM may never switch from Stage 3 to Stage 1. The regularization parameter λ controls data fitting and the sparsity of the solution. m_0 is the minimum of the reduced dimension m, whereas t_b determines the linear growth speed of the reduced dimension m. It is expected that the PCNN-MM is trained in a small reduced space in Stage 2 at the beginning when m is small. Therefore, more fluctuations can be introduced in the training so that there is a higher chance for the search to escape the current local minimum and reach a lower local

minimum in the $\mathbf{w}$ subspace. In the later stage of the training, the reduced dimension m is closer to the complete dimension of the $\mathbf{w}$ subspace. Therefore, the training of the PCNN-MM will be more stable in order to converge to a local minimum in the $\mathbf{w}$ subspace. Trade-offs need to be made between the computational accuracy and efficiency for these hyperparameters to improve the overall performance of the DD-CS algorithm. A sensitivity study has been conducted to investigate the effects of the hyperparameters of the DD-CS algorithm on the training of the PCNN-MM. A more systematic method to find the optimal hyperparameters is needed in future work.

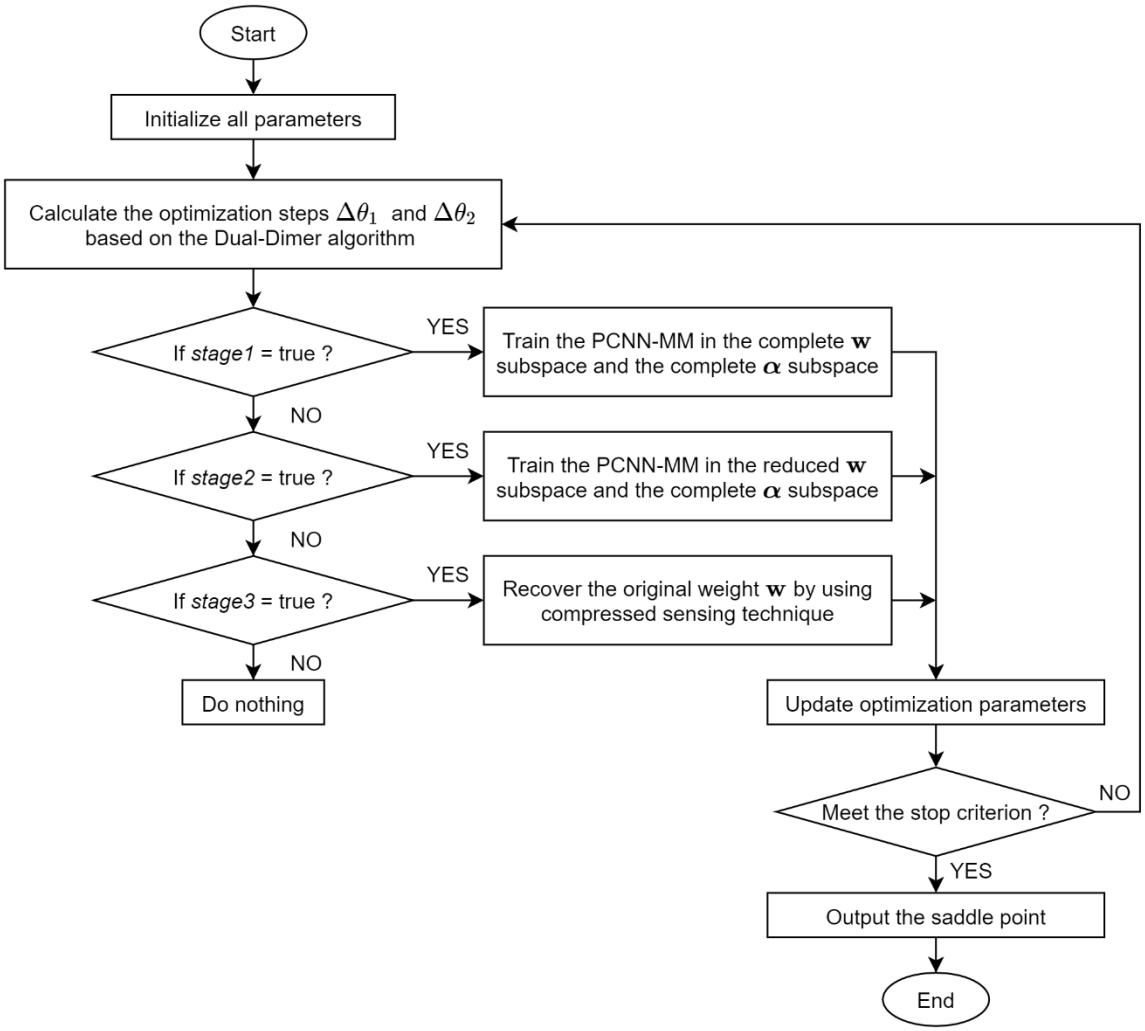


Figure 7.2. The computational flowchart of the DD-CS algorithm.

Table 7.1. The DD-CS algorithm

Input:	initial optimization parameters $\boldsymbol{\theta}_0 = (\mathbf{w}_0, \boldsymbol{\alpha}_0)$, objective function E, hyperparameters p, δ, γ, η, ρ, c_1, c_2, d, m_0, t_b, λ, ζ, t_m
Output:	desired saddle point $\boldsymbol{\theta}^*$

Procedure:	1. Initialize the parameters $t = 0$, $\boldsymbol{\theta}_t = \boldsymbol{\theta}_0$, *stage1* = true, *stage2* = false, *stage3* = false, $\boldsymbol{\Psi}$
	2. Evaluate energy $E(\boldsymbol{\theta}_t)$ and force $\mathbf{f} = -\nabla E$
	3. When $t \bmod p = 0$, compute the extreme eigenvalues (β_s, β_l) and eigenvectors $(\mathbf{v}_s, \mathbf{v}_l)$ by rotating two dimers in the subspaces of $\mathbf{w}$ and $\boldsymbol{\alpha}$
	4. Calculate $\Delta\boldsymbol{\theta}_{\mathbf{w}} = -\eta\nabla_{\mathbf{w}}E(\boldsymbol{\theta})$ and $\Delta\boldsymbol{\theta}_{\boldsymbol{\alpha}} = \eta\nabla_{\boldsymbol{\alpha}}E(\boldsymbol{\theta})$
	5. If $\lvert\beta_s\rvert > \delta$, $\Delta\boldsymbol{\theta}_s = -\frac{(\mathbf{v}_s\cdot\nabla_{\mathbf{w}}E(\boldsymbol{\theta}))\mathbf{v}_s}{\lvert\beta_s\rvert}$; otherwise, $\Delta\boldsymbol{\theta}_s = \mathbf{0}$;
	If $\lvert\beta_l\rvert > \delta$, $\Delta\boldsymbol{\theta}_l = \frac{(\mathbf{v}_l\cdot\nabla_{\boldsymbol{\alpha}}E(\boldsymbol{\theta}))\mathbf{v}_l}{\lvert\beta_l\rvert}$; otherwise, $\Delta\boldsymbol{\theta}_l = \mathbf{0}$
	6. If $\lVert\Delta\boldsymbol{\theta}_s\rVert_2 > \gamma$, $\Delta\boldsymbol{\theta}_s = \gamma\frac{\Delta\boldsymbol{\theta}_s}{\lVert\Delta\boldsymbol{\theta}_s\rVert_2}$; If $\lVert\Delta\boldsymbol{\theta}_l\rVert_2 > \gamma$, $\Delta\boldsymbol{\theta}_l = \gamma\frac{\Delta\boldsymbol{\theta}_l}{\lVert\Delta\boldsymbol{\theta}_l\rVert_2}$
	7. $t = t + 1$
	8. Calculate $\Delta\boldsymbol{\theta}_1 = \Delta\boldsymbol{\theta}_{\mathbf{w}} + \Delta\boldsymbol{\theta}_s$ and $\Delta\boldsymbol{\theta}_2 = \Delta\boldsymbol{\theta}_{\boldsymbol{\alpha}} + \Delta\boldsymbol{\theta}_l$
	9. If *stage1* = true and $\lVert\Delta\boldsymbol{\theta}_1\rVert_2 < c_1\rho^{\frac{t}{d}}$:
	stage1 = false, *stage2* = true;
	get the reduced dimension $m = \left\lfloor m_0 + \frac{t}{t_b}\right\rfloor$;
	get the measurement matrix $\boldsymbol{\Phi}$ and the measurement index $\mathbf{k}$
	10. If *stage2* = true:
	$\Delta\boldsymbol{\theta}_3 = \mathbf{0} \in \mathbb{R}^n$, $\Delta\boldsymbol{\theta}_3[\mathbf{k}] = \Delta\boldsymbol{\theta}_1[\mathbf{k}]$, $\Delta\boldsymbol{\theta}_1 = \Delta\boldsymbol{\theta}_3$;
	if $\lVert\Delta\boldsymbol{\theta}_1\rVert_2 < c_2\rho^{\frac{t}{d}}$:
	stage2 = false, *stage3* = true;
	get the reduced weight $\mathbf{y} = \boldsymbol{\Phi}\mathbf{w}$
	11. If *stage3* = true:
	$\Delta\boldsymbol{\theta}_4 = \boldsymbol{\Psi}S_{\lambda}[\boldsymbol{\Psi}^{-1}\mathbf{w} + \mathbf{A}^T(\boldsymbol{y} - \boldsymbol{\Phi}\mathbf{w})] - \mathbf{w}$;
	$\Delta\boldsymbol{\theta}_1 = \Delta\boldsymbol{\theta}_1 + \Delta\boldsymbol{\theta}_4$;
	if $\lVert\Delta\boldsymbol{\theta}_4\rVert_2 < \zeta$:
	stage3 = false, *stage1* = true
	12. Update optimization parameters by $\boldsymbol{\theta}_t = \boldsymbol{\theta}_{t-1} + \Delta\boldsymbol{\theta} = \boldsymbol{\theta}_{t-1} + (\Delta\boldsymbol{\theta}_1, \Delta\boldsymbol{\theta}_2)$
	13. Return to step 2 until $t \geq t_m$
	14. Output $\boldsymbol{\theta}^* = \boldsymbol{\theta}_t$

7.3 Demonstration

In this section, the developed PCNN-MM and the DD-CS algorithm are demonstrated with two examples: a thermal dendritic growth example and a thermo-solutal dendritic growth example. In each example, an ANN will be trained using the sequential training scheme. And PCNN-MMs are trained with three training schemes: concurrent training with the Dual-Dimer algorithm, sequential training with the Dual-Dimer algorithm, and sequential training with the DD-CS algorithm.

7.3.1 Thermal Dendritic Growth Example

7.3.1.1 Computational Setup

The first example is the thermal dendritic growth during solidification, where heat transfer and phase transition are coupled with each other. In this multiphysics problem, the heat equation and the Allen-Cahn equation need to be solved simultaneously to predict the evolution of dendritic growth. The coupled PDEs and corresponding boundary conditions for the dendritic growth example are

$$\begin{cases} 0.001p_t - 0.0001\left(p_{xx} + p_{yy}\right) = p(1-p)\left[p - 0.5 + \dfrac{0.9}{\pi}\tan^{-1}(10 - 10q)\right] \\ p(0,x,y) = \dfrac{1}{2}[1 - sgn(x^2 + y^2 - 0.04)] \\ p_x(t,-2.5,y) = p_x(t,2.5,y) = p_y(t,x,-2.5) = p_y(t,x,2.5) = 0 \\ 0.01\left(q_t - q_{xx} - q_{yy}\right) = 0.02p_t \\ q(0,x,y) = 0 \\ q_x(t,-2.5,y) = q_x(t,2.5,y) = q_y(t,x,-2.5) = q_y(t,x,2.5) = 0 \\ t \in [0,1], \qquad x,y \in [-2.5,2.5] \end{cases} \tag{7.10}$$

where p is the phase field and q is the temperature field.

In this dendritic growth example, the simulation domain is $x, y \in [-2.5,2.5]$ and time period is $t \in [0,1]$. The training data are sampled randomly from the FEM simulation

with a sampling percentage of 20%, where the grid spacing is $\Delta x = 0.1$ and the time step is $\Delta t = 0.1$. The amount of training data is $N_T = 5722$. The physical constraints are sampled uniformly in both temporal and spatial dimensions. The number of physical constraints is $11 \times 51 \times 51 = 28611$, which means that there are 11 sampling points in the temporal dimension, 51 sampling points in the *x*-direction, and 51 in the *y*-direction of the spatial domain. The grid spacing is $\Delta x = 0.1$ and the time step is $\Delta t = 0.1$ for physical constraints.

The ANN and PCNN-MMs have the same hidden layer structure of 30-30-30-30, where each network has 4 layers and the number of neurons in each layer is 30 The hyperbolic tangent (tanh) function is used as the activation function. The hyperparameters of the Dual-Dimer algorithm and the DD-CS algorithm in the thermal dendritic growth example are listed in Table 7.2. The training of all ML models stops when the maximum number of iterations t_m is reached.

Table 7.2. Hyperparameters of the Dual-Dimer algorithm and the DD-CS algorithm in the thermal dendritic growth example

Parameter	p	δ	γ	η	ρ	c_1	c_2
Value	40	0.001	1×10^{-5}	5×10^{-4}	0.9	0.003	0.0015
Parameter	d	m_0	t_b	t_m	λ	ζ	
Value	1000	0.2	1×10^5	2×10^4	0.02	0.05	

7.3.1.2 Computational Results

The predicted phase fields and temperature fields from different models at t = 1.0 are shown in Figure 7.3 and Figure 7.4, respectively. The quantitative comparison for different models to solve the thermal dendritic growth problem is shown in Table 7.3. It is seen that the predicted phase fields from PCNN-MMs are more accurate that that from the ANN

since the physical constraints are added to guide the training process. It demonstrates the effectiveness of the PCNN-MM. The predicted phase fields from PCNN-MMs with the sequential training scheme are more consistent than that from the PCNN-MM with the concurrent training scheme. It suggests that the sequential training scheme is better than the concurrent sequential training scheme since it helps the convergence of the PCNN-MM. The predicted phase field from the PCNN-MM with the DD-CS algorithm is slightly worse than that from the PCNN-MM with the Dual-Dimer algorithm. Future work is needed to adjust the hyperparameters of the DD-CS algorithm or improve the DD-CS algorithm. The predicted temperature fields from different models are similar.

The learning curves from different models are shown in Figure 7.5. Though the training loss of the phase field is about 10^{-5} at the end of the training for the ANN, the MSE of the predicted phase field is higher than those from the PCNN-MMs. This suggests that the ANN tends to be overfitted without the physical constraints. For the PCNN-MM with the concurrent training scheme, different losses decrease first and then reach the same magnitude during the training. For the PCNN-MM with the sequential training scheme and the Dual-Dimer algorithm, different losses are not at the same magnitude in the later stage of the training, which may accelerate the convergence of the training. As shown in Figure 7.5(d), multiple spikes are introduced in the learning curve for the PCNN-MM with the DD-CS algorithm. When the iteration increases, fewer spikes are observed. Those spikes are designed to help the training of the PCNN-MM to escape the local saddle point so that the neural network can converge to a better saddle point.

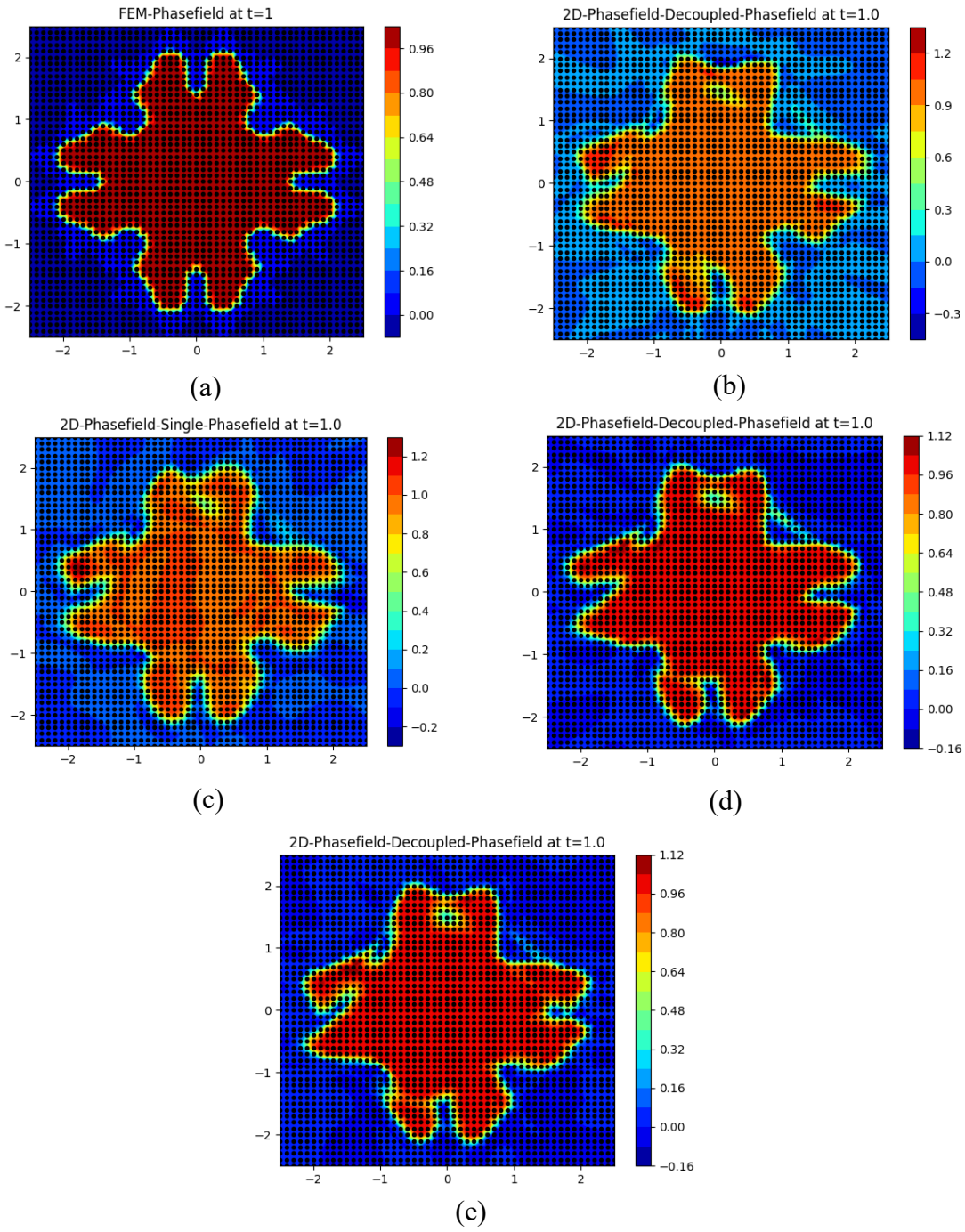


Figure 7.3. The predicted phase fields from different models at t = 1.0: (a) the original FEM solution, (b) the ANN with sequential training, (c) the PCNN-MM with concurrent training and Dual-Dimer algorithm, (d) the PCNN-MM with sequential training and Dual-Dimer algorithm, and (e) the PCNN-MM with sequential training and DD-CS algorithm.

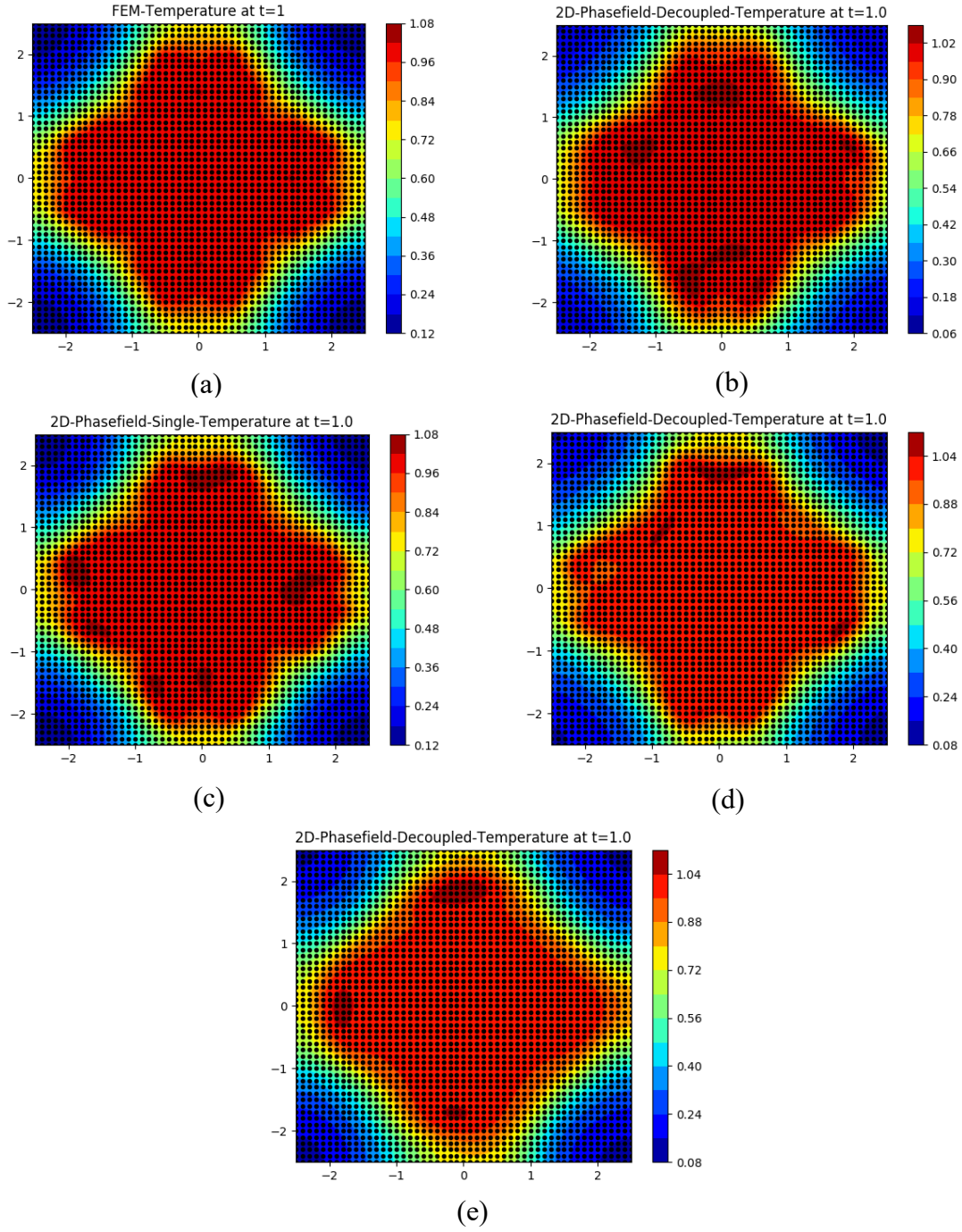


Figure 7.4. The predicted temperature fields from different models at t = 1.0: (a) the original FEM solution, (b) the ANN with sequential training, (c) the PCNN-MM with concurrent training and Dual-Dimer algorithm, (d) the PCNN-MM with sequential training and Dual-Dimer algorithm, and (e) the PCNN-MM with sequential training and DD-CS algorithm.

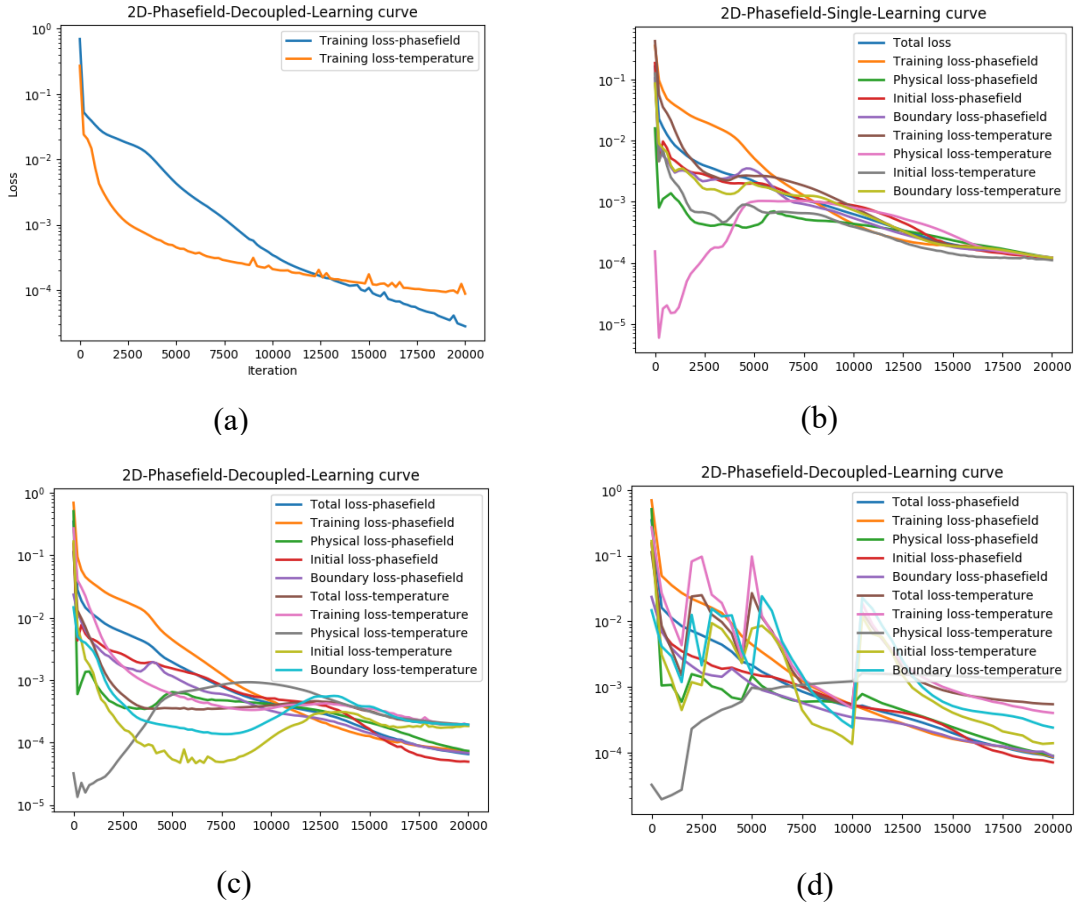


Figure 7.5. The learning curves from different models: (a) the ANN with sequential training, (b) the PCNN-MM with concurrent training and Dual-Dimer algorithm, (c) the PCNN-MM with sequential training and Dual-Dimer algorithm, and (d) the PCNN-MM with sequential training and DD-CS algorithm.

Table 7.3. Quantitative comparison for different models to solve the thermal dendritic growth problem

	The ANN with sequential training	The PCNN-MM with concurrent training and Dual-Dimer algorithm	The PCNN-MM with sequential training and Dual-Dimer algorithm	The PCNN-MM with sequential training and DD-CS algorithm
MSE of phase field	0.0248	0.0189	0.0195	0.0221
MSE of temperature field	0.0004	0.0003	0.0005	0.0008

7.3.2 Thermo-Solutal Dendritic Growth Example

7.3.2.1 Computational Setup

The second example is the thermo-solutal dendritic growth during the rapid solidification process, where heat transfer, solute transport, and phase transition are coupled with each other. The kinetic equation for the phase field ϕ is described by Eq. (3.6), where $M_\phi = M_0\left[1 - 1.5\varepsilon + 2.5\varepsilon\left(n_x^4 + n_y^4\right)\right]$ is the effective interface mobility, $\sigma^* = \sigma_0^*\left[1 + 1.5\varepsilon - 2.5\varepsilon\left(n_x^4 + n_y^4\right)\right]$ is anisotropic interface energy stiffness. The kinetic equation for the composition field is given by Eq. (4.1). The heat conduction equation is described by Eq. (4.2). The cooling rate is $\dot{T} = -1 \times 10^4$ K/s. The physical properties of Ti-6Al-4V alloy are shown in Table 3.1. The training data comes from the PF-TLBM simulation as described in Section 3.2. In the PF-TLBM simulation, the fine grid spacing is $dx = 0.5$ μm and the time step is $dt = 1$ μs. The length and width of the 2D simulation domain are $L_x = L_y = 200\ dx = 100$ μm. The interface width is $\eta = 5\ dx$.

At the beginning of the simulation, a circular nucleus is planted at the center of the simulation domain. The initial condition for the phase field is

$$\phi = \begin{cases} 1 & (R < r - \frac{\eta}{2}) \\ \frac{1}{2} - \frac{1}{2} sin\left[\frac{\pi(R-r)}{\eta}\right] & (r - \frac{\eta}{2} \leq R \leq r + \frac{\eta}{2}) \\ 0 & (R > r + \frac{\eta}{2}) \end{cases} \tag{7.11}$$

where R is the distance between a location (x, y) and the center of the nucleus in the domain, and $r = 8\ dx$ is the radius of the nucleus. The initial composition of the solute $C_0 = 10$ wt% and the initial temperature $T_0 = 1928$ K are set for the whole simulation domain. Zero Neumann conditions are set at all boundaries for the phase field ϕ, composition field C, and temperature field T. The simulation time is $t_{max} = 0.01$ s and the simulation results are stored every 0.5 ms.

Since the values of different physical fields can differ from each other by several orders of magnitude, they need to be scaled to make the training of the PCNN-MM easier to converge. The scaled variables are defined as $x^* = x/L_x$, $y^* = y/L_y$, $t^* = t/t_{max}$, $T^* = (T - T_s)/(T_l - T_s)$, and $C_l^* = k_e(C_l - C_0)/[(1 - k_e)C_0]$, respectively. After scaling, all the variables have values with magnitudes ranging from zero to unity. Phase field ϕ does not need to be scaled since it is already in the range of [0,1]. By plugging the scaled variables into the kinetic equations, initial conditions, and boundary conditions for all physical variables, the scaled form of the multiphysics model for thermo-solutal dendritic growth is used.

Partial PF-TLBM simulation results were used to train the ML models, where 5% of the simulation data along different time frames were randomly selected as the training data. Specifically, the number of training data points is $N_T = 42421$. Knowledge of the

multiphysics model as the supplement is used to guide the training of the PCNN-MM. That is, the physical constraints are evaluated uniformly in both temporal and spatial domains. The number of physical constraints is $t^* \times x^* \times y^* = 11 \times 51 \times 51 = 28611$, where the time step is $\Delta t^* = 0.1$ and the grid spacing is $\Delta x^* = \Delta y^* = 0.02$ for physical constraints. The numbers of sampling points corresponding to the physical loss, initial loss, and boundary loss are $N_P = 24010$, $N_I = 2601$, and $N_S = 2000$, respectively, which add up to 28611. Both physical constraints and training data are employed in the training. Once the training is finished, all physical variables at $t^* = 1$ are predicted from the PCNN-MM with a grid spacing of $\Delta x^* = \Delta y^* = 0.005$, which is finer than the grid spacing of the physical constraints. The predicted values of temperature, phase field, and concentration are converted back to the original scale.

The ANN and PCNN-MMs have the same hidden layer structure of 30-30-30-30-30, where each network has 5 layers and the number of neurons in each layer is 30 The hyperbolic tangent (tanh) function is used as the activation function. The hyperparameters of the Dual-Dimer algorithm and the DD-CS algorithm in the thermo-solutal dendritic growth example are listed in Table 7.4.

Table 7.4. Hyperparameters of the Dual-Dimer algorithm and the DD-CS algorithm in the thermo-solutal dendritic growth example

Parameter	p	δ	γ	η	ρ	c_1	c_2
Value	40	0.001	1×10^{-5}	5×10^{-4}	0.96	0.0004	0.0002
Parameter	d	m_0	t_b	t_m	λ	ζ	
Value	1000	0.2	2×10^5	2×10^4	0.02	0.05	

7.3.2.2 Computational Results

The predicted phase fields, temperature fields, and composition fields from different models at t = 1.0 for the thermo-solutal dendritic growth example are shown in Figure 7.6, Figure 7.7, and Figure 7.8, respectively. The quantitative comparison for different models to solve the thermo-solutal dendritic growth problem is shown in Figure 7.9. It shows that the predicted phase field and composition field from the PCNN-MM with the concurrent training scheme are slightly worse than those from the ANN. The predicted phase field and composition field from the PCNN-MM with the sequential training scheme and the Dual-Dimer algorithm are more accurate than those from other ML models. It demonstrates the effectiveness of the PCNN-MM with the sequential training scheme and the Dual-Dimer algorithm. It also suggests that the sequential training scheme is better than the concurrent training scheme since it helps the convergence of the PCNN-MM. The PCNN-MM with the DD-CS algorithm has the highest prediction errors among all ML models in this example, which could be caused by improper sets of hyperparameters. Future work is needed to adjust the hyperparameters of the DD-CS algorithm or improve the DD-CS algorithm. The predicted temperature fields from PCNN-MMs are not as accurate as of that from the ANN, which could be caused by the simulation error. Theoretically speaking, the temperature field should be four-fold symmetric, similar to what PCNN-MMs predict. PCNN-MMs were trained based on both simulation data and the original physical models. However, the simulated temperature field in the TLBM is not four-fold symmetric. This is likely caused by the coarse mesh used in the TLBM model. In the PF-TLBM, the mesh size in TLBM is coarser than the one in the phase field. Further investigation is needed to provide more accurate simulations and neural network predictions.

The learning curves from different models are shown in Figure 7.9. For the PCNN-MM with the concurrent training scheme, different losses decrease first and then reach about the same magnitude during the training. For the PCNN-MM with the sequential training scheme and the Dual-Dimer algorithm, different losses are not at the same magnitude in the later stage of the training, which may help the convergence of the training. Some spikes are observed in the later stage of the training for the PCNN-MM with the sequential training scheme and the Dual-Dimer algorithm, which may contribute to the convergence of the training. As shown in Figure 7.9(d), multiple spikes are introduced in the learning curve at the beginning of the training for the PCNN-MM with the DD-CS algorithm. When the iteration increases, fewer spikes are observed. Those spikes are designed to help the training of the PCNN-MM to escape the local saddle point so that the neural network may converge to a better saddle point.

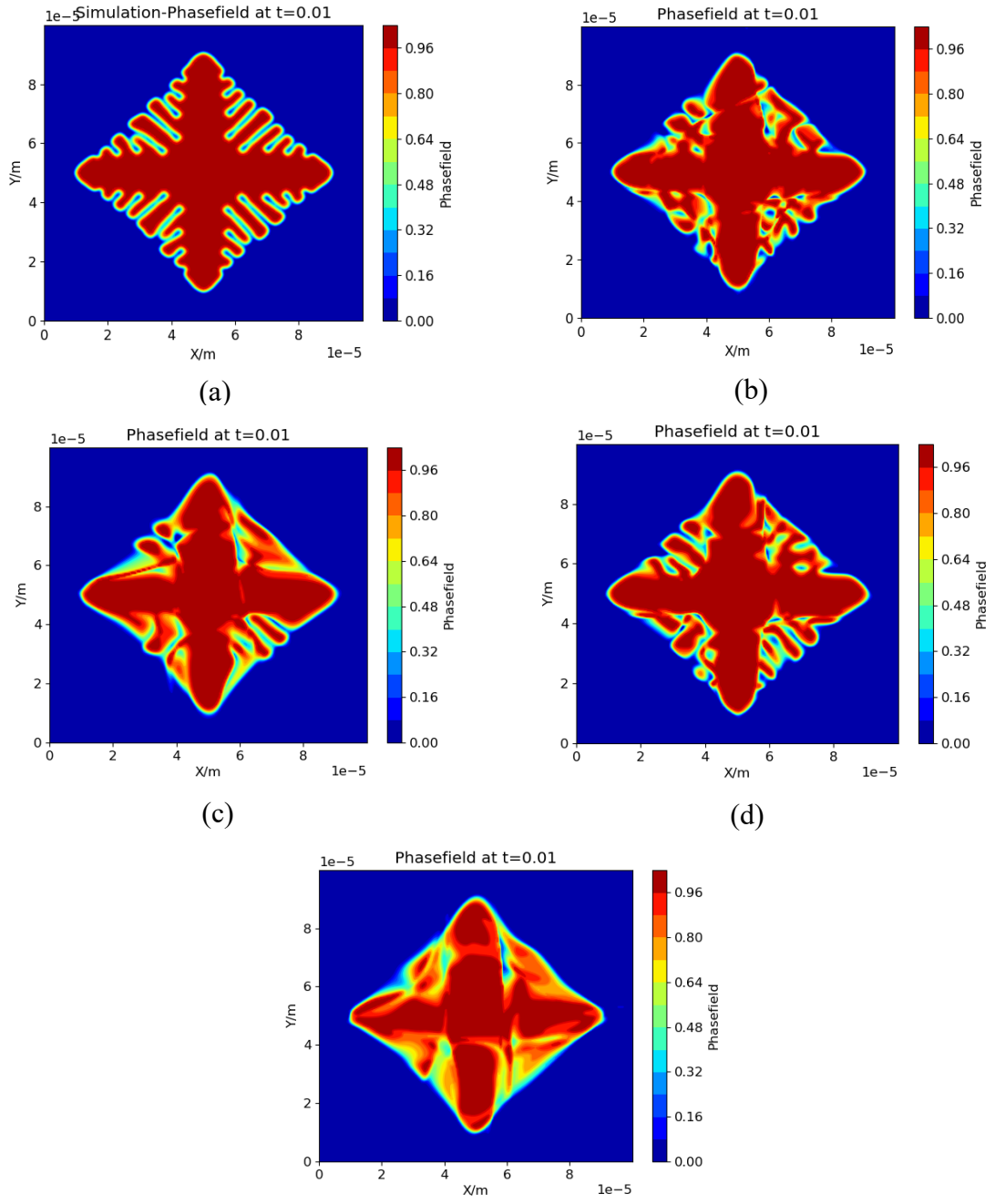


Figure 7.6. The predicted phase fields from different models at t = 0.01: (a) the original PF-TLBM solution, (b) the ANN with sequential training, (c) the PCNN-MM with concurrent training and Dual-Dimer algorithm, (d) the PCNN-MM with sequential training and Dual-Dimer algorithm, and (e) the PCNN-MM with sequential training and DD-CS algorithm.

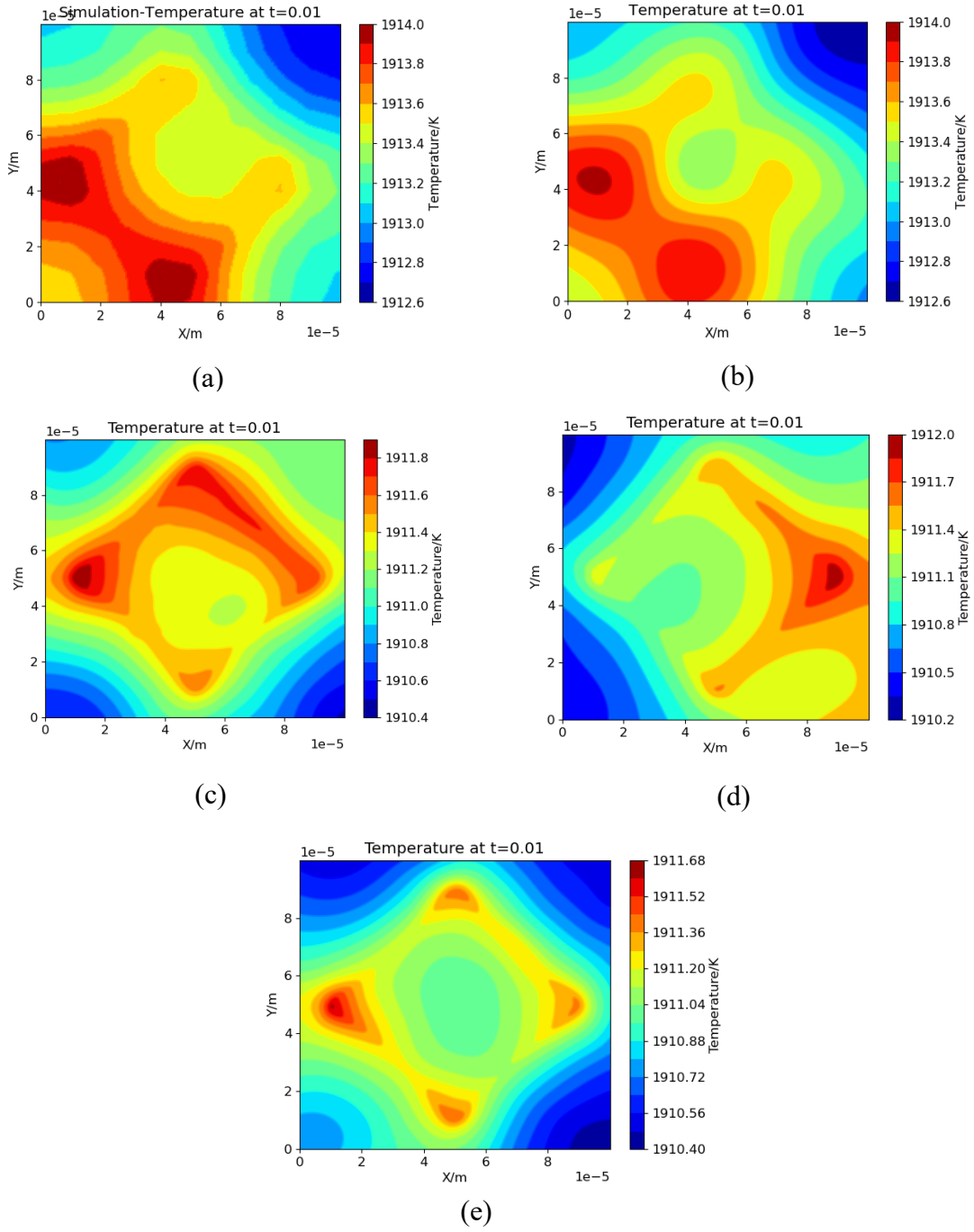


Figure 7.7. The predicted temperature fields from different models at t = 0.01: (a) the original PF-TLBM solution, (b) the ANN with sequential training, (c) the PCNN-MM with concurrent training and Dual-Dimer algorithm, (d) the PCNN-MM with sequential training and Dual-Dimer algorithm, and (e) the PCNN-MM with sequential training and DD-CS algorithm.

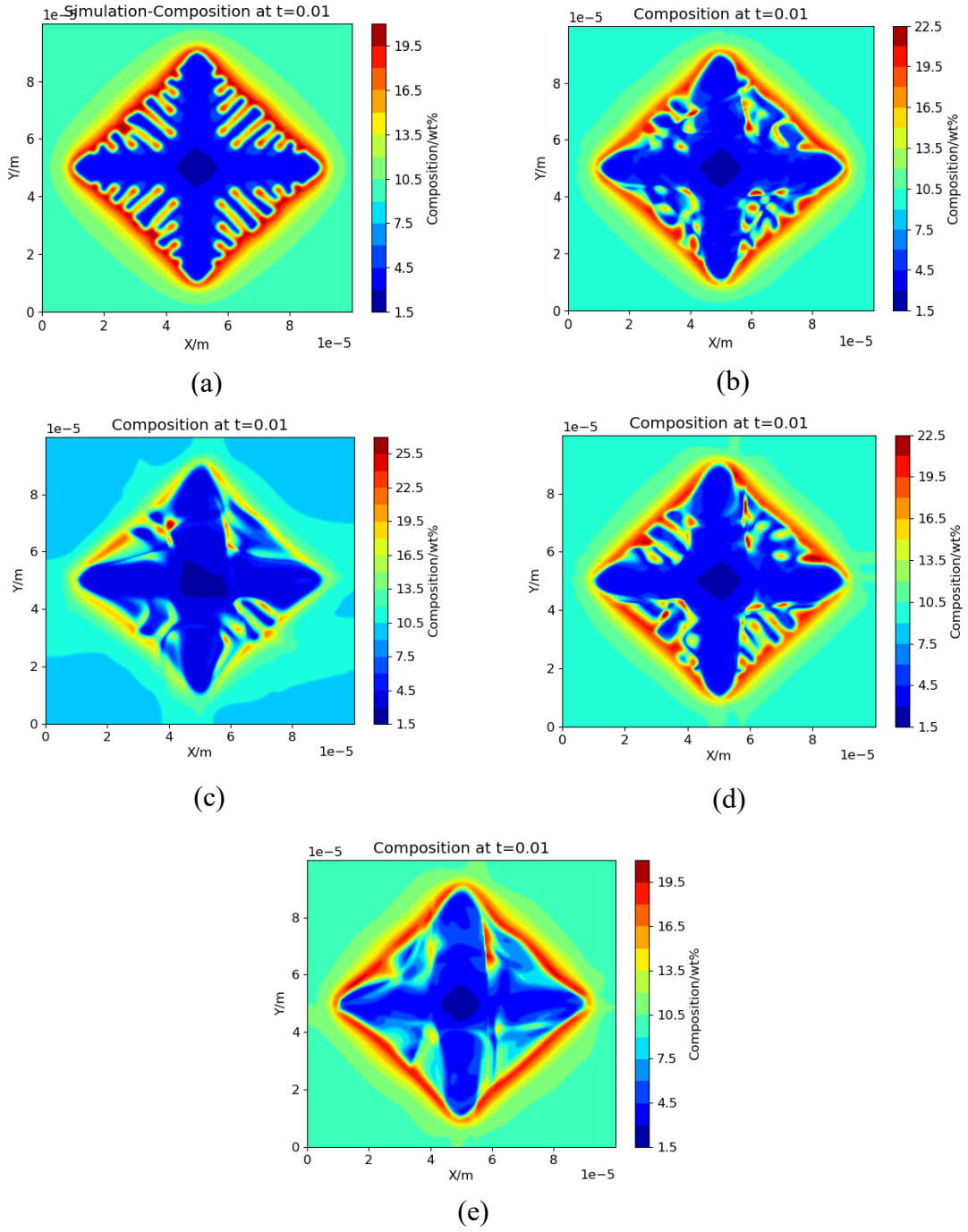


Figure 7.8. The predicted composition fields from different models at t = 0.01: (a) the original PF-TLBM solution, (b) the ANN with sequential training, (c) the PCNN-MM with concurrent training and Dual-Dimer algorithm, (d) the PCNN-MM with sequential training and Dual-Dimer algorithm, and (e) the PCNN-MM with sequential training and DD-CS algorithm.

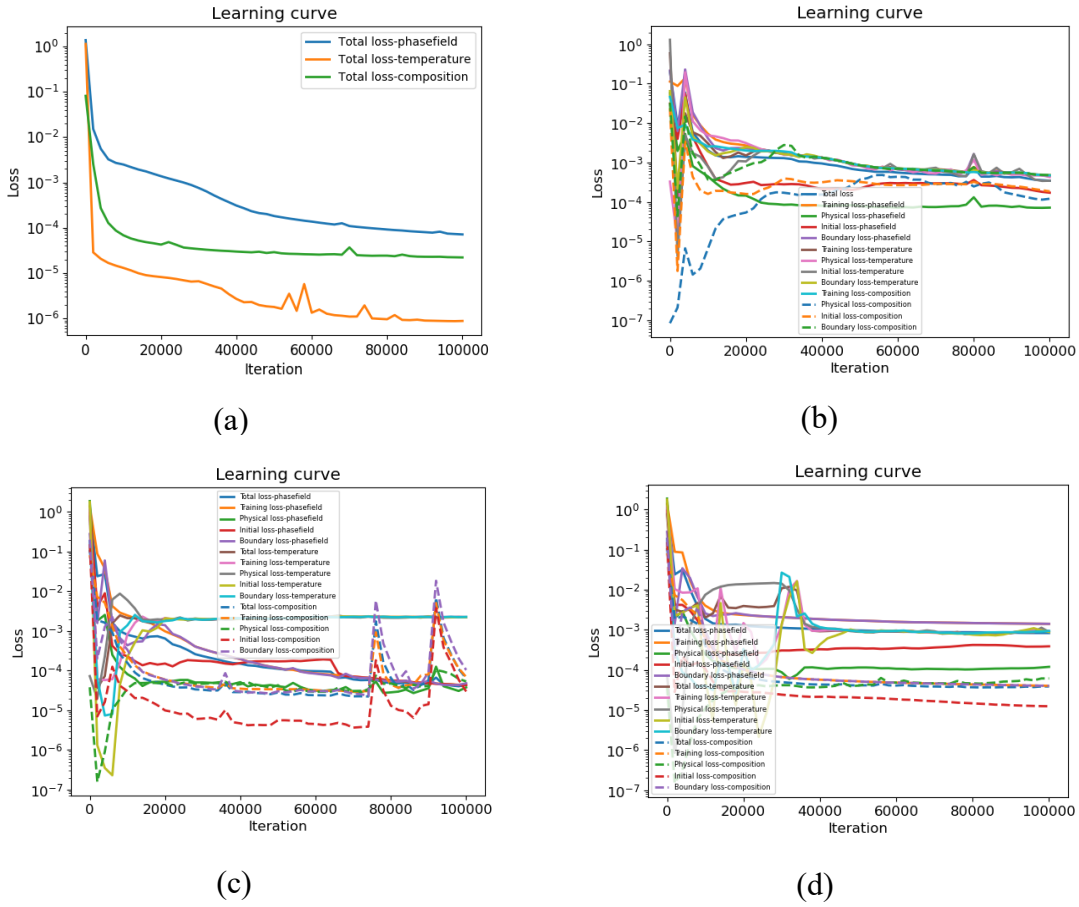


Figure 7.9. The learning curves from different models: (a) the ANN with sequential training, (b) the PCNN-MM with concurrent training and Dual-Dimer algorithm, (c) the PCNN-MM with sequential training and Dual-Dimer algorithm, and (d) the PCNN-MM with sequential training and DD-CS algorithm.

Table 7.5. Quantitative comparison for different models to solve the thermo-solutal dendritic growth problem

	The ANN with sequential training	The PCNN-MM with concurrent training and Dual-Dimer algorithm	The PCNN-MM with sequential training and Dual-Dimer algorithm	The PCNN-MM with sequential training and DD-CS algorithm
MSE of phase field	0.0112	0.0118	0.0096	0.0142
MSE of temperature field	0.0017	4.6777	5.3205	5.9412
MSE of composition field	3.2124	4.0891	2.8251	4.0629

7.4 Discussions and Conclusions

In this chapter, a new sequential training scheme is developed to aid the convergence of PCNN-MMs for solving multiphysics problems. A new saddle point search algorithm called the DD-CS algorithm is also developed to alleviate the curse of dimensionality in searching high-order saddle points during the training. The developed PCNN-MM and the DD-CS algorithm are demonstrated by two examples: thermal dendritic growth example and thermo-solutal dendritic growth example. In each example, different ML models with various training schemes are compared with each other.

The computational results suggest that the sequential training scheme is better than the concurrent sequential training scheme since it facilitates the convergence of the PCNN-MM. In the sequential training scheme, multiple neural networks are trained sequentially to predict multiple physical fields respectively. By using the sequential training scheme, different physics fields are one-way coupled rather than fully coupled, which accelerates

the convergence of PCNN-MMs. In this work, each neural network is trained only once within one iteration in the sequential training scheme. In future work, sensitivity studies need to be conducted to investigate the effects of the training frequency of different neural networks on prediction accuracy and training efficiency of PCNN-MMs.

The predicted phase field from the PCNN-MM with the DD-CS algorithm is slightly worse than that from the PCNN-MM with the Dual-Dimer algorithm. The spikes in the learning curve are designed to help the training of the PCNN-MM to escape the local saddle point so that the neural network may converge to a better saddle point. Future work is needed to adjust the hyperparameters of the DD-CS algorithm or improve the DD-CS algorithm. The training dynamics of the PCNN-MM with the DD-CS algorithm needs to be analyzed by mathematical analysis or computational experiments. The training dynamics of the PCNN-MM around the spikes needs more attention to improve the training efficiency and robustness. Rather than choosing adaptive thresholds by trial and error, a more rigorous and systematic approach is needed to search optimal hyper parameters. For instance, the information on gradients and extreme eigenvalues may be used to choose the optimal adaptive thresholds. The DD-CS algorithm is a promising training algorithm to find the high-order saddle points, which may be applied to alleviate the curse of dimensionality in other general ML models and applications.

As a deterministic surrogate, the constructed PCNN-MM model can provide fast predictions from some given inputs. However, it does not provide the uncertainty information about the predictions. In future work, the uncertainty of the PCNN-MM model should be quantified to provide more confidence in the predictions. Uncertainty quantification has been widely applied in multiscale modeling and simulation [221–225].

Uncertainty can be generally classified into two main categories, epistemic uncertainty and aleatory uncertainty [226,227]. Epistemic uncertainty is caused by the lack of data or knowledge, which is reducible, whereas aleatory uncertainty is the variability that is due to inherently random effects and is irreducible. In a model, the major components of epistemic uncertainty are model-form uncertainty and parameter uncertainty. Model-form uncertainty results from simplification, approximation error, and subjectivity of model choice. The model-form uncertainty of the PCNN-MM model comes from the subjective choice of neural network structure, including architecture (fully-connected neural network, convolutional neural network, recurrent neural network, etc.), number of hidden units and layers, activation functions, etc. Parameter uncertainty is related to model calibration. The parameters of models need to be calibrated properly to reduce parameter uncertainty. The parameter uncertainty of the PCNN-MM model comes from the training process when training data are inaccurate or the prior physical knowledge used in the PCNN is not perfect. The model-form and parameter uncertainties of one model can propagate to another when the former is used as the baseline to calibrate the latter, and the errors can remain in simulation data. Experimental data can contain measurement errors caused by instruments or human operators. The parameter uncertainty of the PCNN-MM model can also come from the formulation of the training algorithm and the choices of hyperparameters of the training algorithm. In the Dual-Dimer algorithm, the extreme eigenvalues and eigenvectors are approximated by rotating two dimers, which could cause parameter uncertainty. In the third stage of the DD-CS algorithm, the reconstruction error of the weights for the neural network can be introduced by the compressive sampling methods, which can lead to parameter uncertainty. Some uncertainty quantification

methods have been applied in deep learning, which include Bayesian techniques (Monte Carlo dropout, Markov chain Monte Carlo, variational inference, Bayesian active learning, Bayes by backprop, variational autoencoders, Laplacian approximations) and ensemble techniques [228]. Most Bayesian techniques can be used to quantify parameter uncertainty, whereas ensemble techniques can be used to quantify both model-form and parameter uncertainty.

CHAPTER 8. RAPID SOLIDIFICATION PROCESS OPTIMIZATION FOR ADDITIVE MANUFACTURING

8.1 Introduction

In simulation-based design optimization, the objective functions are usually expensive to be evaluated. Because of the high-dimensionality of P-S-P relationships, it requires a significant amount of simulations or training of ML models to conduct the process optimization. In order to reduce the number of function evaluations during the optimization, surrogate models [124] are commonly used to assist design optimization. In this chapter, a new generic process design framework is developed. It is demonstrated with the optimization of the initial temperature and cooling rate in the SLM process to achieve the desirable microstructures. The dendritic growth of Ti-6Al-4V alloy is simulated using the developed PF-TLBM model under various initial temperatures and cooling rates. A surrogate model of process-structure relationships is constructed based on the PCNN-MM in CHAPTER 7 trained by the simulation data from PF-TLBM in CHAPTER 3. The constructed surrogate model is used in single-objective and multi-objective BO to search the optimal process parameters so that the desired dendritic area and composition distributions can be achieved.

In the remainder of this chapter, multi-objective Bayesian optimization is introduced in Section 8.2. A generic process design framework and optimization problems are described in Section 8.3. In Section 8.4, the proposed process design framework is demonstrated by optimizing the initial temperature and cooling rate for the rapid solidification of Ti-6Al-4V alloy with single-objective and multi-objective BO.

8.2 Multi-Objective Bayesian Optimization

In general, multi-objective optimization problems are formulated as

$$\begin{aligned} &\min_{\mathbf{x}} \mathbf{F}(\mathbf{x}) = \{f_1(\mathbf{x}), f_2(\mathbf{x}), \dots, f_i(\mathbf{x}), \dots, f_n(\mathbf{x})\} \\ &s.t. \quad g_j(\mathbf{x}) \le 0, \;\; j = 1,2, \dots, m \\ &\mathbf{x}_{min} \le \mathbf{x} \le \mathbf{x}_{max} \end{aligned} \tag{8.1}$$

where $\mathbf{F}(\mathbf{x})$ is the total objective function including n objective functions among which at least two objective functions are conflicting with each other, $\mathbf{g}(\mathbf{x})$ are the constraints, $\mathbf{x}$ is the design variable, and $\mathbf{x}_{min}$ and $\mathbf{x}_{max}$ are the lower bounds and upper bounds, respectively. Since there are trade-offs among those objective functions, the problem in Eq. (8.1) usually has a set of optimum solutions in the Pareto sense. Namely, there is no optimum that is superior to the other with respect to all objectives. These solutions form as Pareto set or Pareto front.

As a surrogate-based global optimization technique, BO [229] has been applied successfully in machine learning and design optimization. The uniqueness of BO is that the sequential sampling strategy is adopted to reach a balance between exploration (sampling in the most uncertain region) and exploitation (sampling in the region with the best predictions) for robust global optimization under uncertainty. Compared to other optimization techniques, the major advantages of BO include derivative-free, active learning, uncertainty quantification, and robustness in high-dimensional continuous design space. Based on a surrogate model, such as Gaussian process (GP), which approximates the objective function, BO performs sequential sampling to find the optimal solution. The sequential sampling strategy is to choose one solution that maximizes an acquisition function. The acquisition function needs to be constructed in the same design space as the

objective surrogate. Commonly used acquisition functions include expected improvement (EI), probability of improvement (PI), and lower confidence bound (LCB). The EI function is used here for single-objective BO.

Consider the optimization problem

$$\min f(\mathbf{x}) \tag{8.2}$$

Given some samples $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n\}$ and their responses $f(\mathbf{X}) = \{f(\mathbf{x}_1), f(\mathbf{x}_2), \dots, f(\mathbf{x}_n)\}$, the EI acquisition function is given by

$$a_{\mathrm{EI}}(\mathbf{x}) = s(\mathbf{x})\gamma(\mathbf{x})\Phi\big(\gamma(\mathbf{x})\big) + s(\mathbf{x})\phi\big(\gamma(\mathbf{x})\big) \tag{8.3}$$

where $s(\mathbf{x})$ is the posterior standard deviation, $\phi(\cdot)$ and $\Phi(\cdot)$ are the probability density function and cumulative distribution function, respectively. The deviation away from the best sample $\mathbf{x}_{\mathrm{best}}$ is given by

$$\gamma(\mathbf{x}) = \frac{\mu(\mathbf{x}) - f(\mathbf{x}_{\mathrm{best}})}{s(\mathbf{x})} \tag{8.4}$$

where $\mu(\mathbf{x})$ is the posterior mean.

The LCB acquisition function is given by

$$a_{\mathrm{LCB}}(\mathbf{x}) = \mu(\mathbf{x}) - ks(\mathbf{x}) \tag{8.5}$$

where k controls the exploitation/exploration ratio. A new sample point is selected by maximizing the EI function or negative LCB function. Then the GP model is updated with the new sample point. The iteration continues until a stop criterion is met. The advantage of using LCB as the acquisition function is that there is no need to calculate the integrals as in the EI and PI functions. The calculation of high-dimensional integrals is expensive for multi-objective problems.

To enhance the capability of the classical BO framework, many extensions, including multi-objective [230,231], multi-fidelity [232], constrained [233], mixed integer [234], and discrete [235], have been proposed. The multi-objective BO approach in Ref. [231] is used in this chapter. In this approach, a novel acquisition function is used to determine the next sample point, which helps improve the diversity and convergence of the Pareto solutions. The modified hyperarea difference (MHD) and modified overall spread (MOS) are used to measure the quality of Pareto fronts. The acquisition function is given by

$$a(\mathbf{x}) = \max\left(I_{\mathrm{MHD}}(\mathbf{x}), I_{\mathrm{MOS}}(\mathbf{x})\right) \tag{8.6}$$

where $I_{\mathrm{MHD}}(\mathbf{x})$ and $I_{\mathrm{MOS}}(\mathbf{x})$ are the relative improvements in MHD and MOS, respectively, as

$$\begin{aligned} I_{\mathrm{MHD}}(\mathbf{x}) &= \left|\frac{\mathrm{MHD}(\{\mathbf{D}_n, \mathbf{x}\}) - \mathrm{MHD}(\mathbf{D}_n)}{\mathrm{MHD}(\mathbf{D}_n)}\right| \\ I_{\mathrm{MOS}}(\mathbf{x}) &= \left|\frac{\mathrm{MOS}(\{\mathbf{D}_n, \mathbf{x}\}) - \mathrm{MOS}(\mathbf{D}_n)}{\mathrm{MOS}(\mathbf{D}_n)}\right| \\ \mathbf{x}_{min} &\le \mathbf{x} \le \mathbf{x}_{max} \end{aligned} \tag{8.7}$$

where $\mathbf{D}_n$ are existing n sample points, $\mathrm{MHD}(\mathbf{D}_n)$ and $\mathrm{MOS}(\mathbf{D}_n)$ represent the MHD and MOS based on $\mathbf{D}_n$, whereas $\mathrm{MHD}(\{\mathbf{D}_n, \mathbf{x}\})$ and $\mathrm{MOS}(\{\mathbf{D}_n, \mathbf{x}\})$ represent the updated MHD and MOS when $\mathbf{x}$ is added in the sample set. The LCB functions in Eq. (8.5) are used as the objectives to calculate $\mathrm{MHD}(\{\mathbf{D}_n, \mathbf{x}\})$ and $\mathrm{MOS}(\{\mathbf{D}_n, \mathbf{x}\})$. A new sample point will be selected to update the GP model by maximizing the acquisition function in Eq. (8.6). More details about the multi-objective BO can be found in Ref. [231].

8.3 Process Design Framework

In this chapter, the generic process design framework shown in Figure 1.1 is developed. A surrogate model of process-structure relationship is built for AM based on PF-TLBM in CHAPTER 3 and PCNN-MM in CHAPTER 7. The dendritic growth under different process parameters (initial temperature T_0 and cooling rate $\dot{T}$) are simulated by PF-TLBM. The simulation outputs include phase field, temperature field, and composition field. Partial simulation results serve as the training data for the training of the PCNN-MM and physical constraints will be added to reduce the required amount of training data. The normalized initial temperature $(T_0 - T_s)/(T_l - T_s)$ and cooling rate $\dot{T} * t_{max}/(T_l - T_s)$ are added into the PCNN-MM as the inputs so that the PCNN-MM can predicted the microstructure under different process parameters. The developed Dual-Dimer algorithm in Section 6.2.2 is used for the training of PCNN-MM. Once the training of the PCNN-MM is completed, the output microstructures from the PCNN-MM are characterized as dendritic area and microsegregation. In this way, a surrogate model of the process-structure relationship is built for SLM. After training, this surrogate model can predict microstructures from process parameters efficiently without relying on expensive simulations or experiments. Based on the surrogate model, BO is utilized to search the optimal initial temperature and cooling rate to obtain the desired dendritic area S_d^* and microsegregation χ^* level. The definition of the microsegregation is shown in Eq. (4.18).

When the dendritic area is larger during a fixed period of solidification time, it is quicker to solidify the melt and print the part under the current process parameters. When the microsegregation is smaller, the mechanical strength of the printed part will be improved. Therefore, the dendritic area should be as large as possible to reduce the printing

time. The microsegregation should be as small as possible to increase the mechanical strength. However, the increase of the dendritic area and the decrease of the microsegregation are usually in conflict, which makes multi-objective optimization necessary. A designer usually has a target performance. The design objectives are to find the optimal initial temperature and cooling rate to meet the target dendritic area and microsegregation level. Therefore, the two-objective optimization problem is given as

$$\begin{aligned} \min_{T_0,\dot{T}} \mathbf{F}(T_0,\dot{T}) &= \{f_1(T_0,\dot{T}), f_2(T_0,\dot{T})\} \\ &= \left\{\left[S_d(T_0,\dot{T}) - S_d^*\right]^2, \left[\chi(T_0,\dot{T}) - \chi^*\right]^2\right\} \\ T_{0,min} \le T_0 &\le T_{0,max} \\ \dot{T}_{min} \le \dot{T} &\le \dot{T}_{max} \end{aligned} \tag{8.8}$$

where $S_d(T_0,\dot{T})$ is the predicted dendritic area from the PCNN-MM, $\chi(T_0,\dot{T})$ is the predicted microsegregation. $T_{0,min}$ and $T_{0,max}$ are the lower and upper bound of the initial temperature, respectively. $\dot{T}_{min}$ and $\dot{T}_{max}$ are the lower and upper bound of the cooling rate, respectively.

Constructing the Pareto front usually is the goal of multi-objective optimization. However, it may be computationally infeasible if resources do not allow. A more efficient approach is combining multiple objectives into one as a weighted average, if the weights as the designer's preference can be predetermined. Then the problem is converted to single-objective optimization. Thus, the two-objective optimization problem in Eq. (8.8) can also be formulated as a single-objective optimization problem as

$$\min_{T_0,\dot{Q}} f(T_0,\dot{T}) = \alpha_1 \left[\frac{S_d(T_0,\dot{T}) - S_d^*}{S_{d,max}}\right]^2 + \alpha_2 \left[\frac{\chi(T_0,\dot{T}) - \chi^*}{\chi_{max}}\right]^2$$
$$T_{0,min} \le T_0 \le T_{0,max} \tag{8.9}$$
$$\dot{T}_{min} \le \dot{T} \le \dot{T}_{max}$$

where α_1 and α_2 are the weights of the objective functions of the dendritic area and the microsegregation, respectively. Since the scales of the dendritic area and the microsegregation are different, $S_{d,max}$ and χ_{max} are used to normalize the objectives of the dendritic area and the microsegregation.

8.4 Results and Discussion

The proposed process design framework is demonstrated by optimizing the initial temperature and cooling rate for the rapid solidification of Ti-6Al-4V alloy so that the desired dendritic area and microsegregation level can be achieved. The computational setup for simulating the dendritic growth in the rapid solidification of Ti-6Al-4V alloy is described in Section 7.3.2.1. The dendritic growth of Ti-6Al-4V alloy is simulated using the developed PF-TLBM model under various initial temperatures and cooling rates. The design of experiment and simulation outputs (dendritic area and microsegregation) are shown in Table 8.1. The range of the initial temperature is $[1918, 1928]$ K, whereas the range of the cooling rate is $[-10000, -5000]$ K/s. The range of the dendritic area is $[1475, 3436.74]$ μm^2, whereas the range of the microsegregation is $[8.79, 12.25]$.

Table 8.1. The design of experiment and simulation outputs

Case number	Initial temperature (K)	Cooling rate (K/s)	Dendritic area (μm^2)	Microsegregation
1	1918	-5000	1914.64	9.09
2	1918	-6250	2314.78	9.79
3	1918	-7500	2646.96	10.39
4	1918	-8750	3016.11	11.16
5	1918	-10000	3436.74	12.25
6	1920.5	-5000	1778.65	8.96
7	1920.5	-6250	2217.93	9.65
8	1920.5	-7500	2611.06	10.41
9	1920.5	-8750	2990.45	11.24
10	1920.5	-10000	3366.18	12.04
11	1923	-5000	1713.26	9.01
12	1923	-6250	2073.77	9.50
13	1923	-7500	2511.03	10.33
14	1923	-8750	2902.44	11.09
15	1923	-10000	3222.27	11.19
16	1925.5	-5000	1561.22	9.05
17	1925.5	-6250	1999.84	9.57
18	1925.5	-7500	2400.67	10.25
19	1925.5	-8750	2738.54	10.84
20	1925.5	-10000	3170.99	11.90
21	1928	-5000	1475.00	8.79
22	1928	-6250	1843.84	9.19
23	1928	-7500	2284.36	9.92
24	1928	-8750	2678.61	10.59
25	1928	-10000	2997.04	11.36

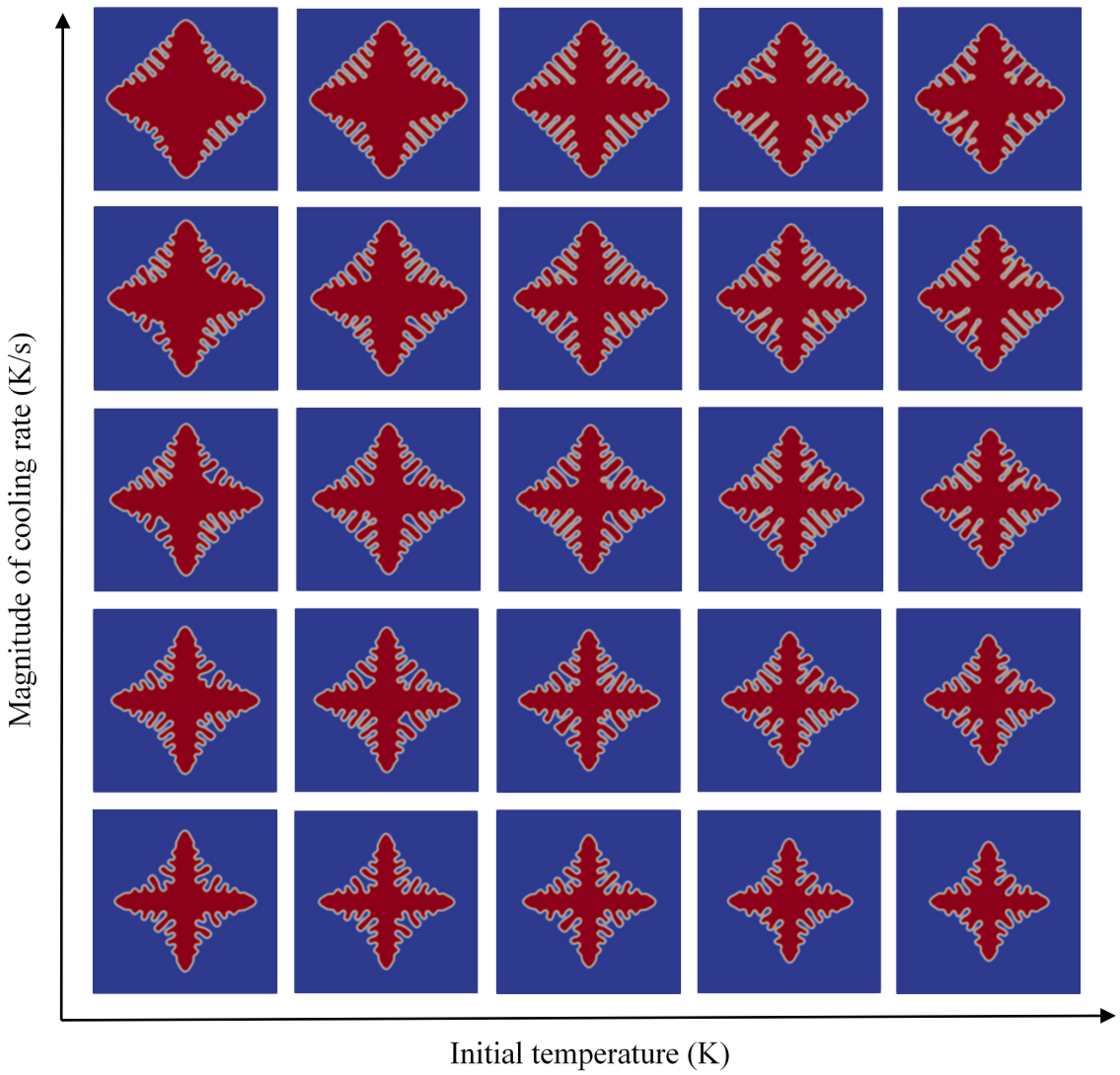
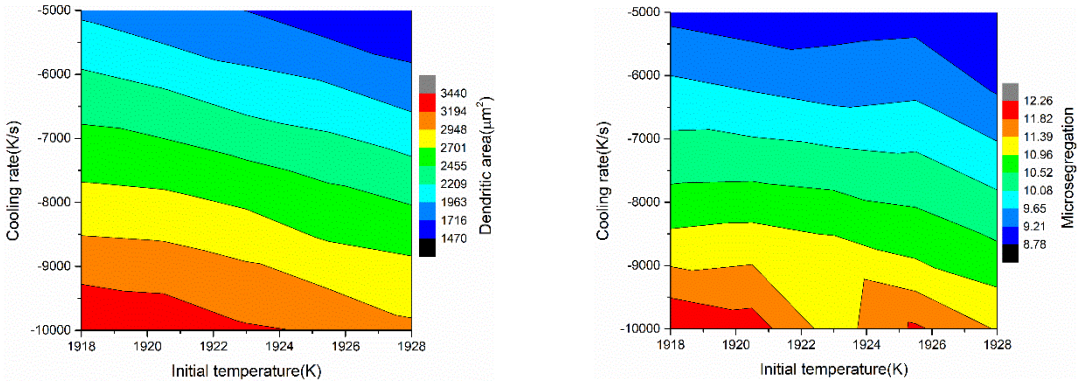


Figure 8.1. Dendritic morphology of Ti-6Al-4V alloy under various initial temperatures and cooling rates.



(a) (b)

Figure 8.2. The effects of the initial temperature and cooling rate on (a) dendritic area and (b) microsegregation.

The dendritic morphology of Ti-6Al-4V alloy at t = 10 ms under various initial temperatures and cooling rates is shown in Figure 8.1. The effects of the initial temperature and cooling rate on dendritic area and microsegregation are shown in Figure 8.2. When the initial temperature increases or the undercooling decreases, the dendritic area and the microsegregation decrease. The primary arm is thinner as the initial temperature increases. When the magnitude of the cooling rate increases, the dendritic area and the microsegregation increase. The primary arm is thicker as the cooling rate increases.

The partial PF-TLBM simulation results are used to train the PCNN-MM model, where 5% of the simulation data along different time frames are randomly selected as the training data. Since the training dataset is very large (about a million data points), the mini-batch training scheme is used to train the PCNN-MM. The learning curve of the training of the PCNN-MM is shown in Figure 8.3. It shows that the losses related to the temperature field are difficult to be minimized. After training, the PCNN-MM model can predict microstructures from process parameters efficiently without relying on expensive

simulations or experiments. For single forward prediction, the average evaluation time of the PF-TLBM model is 2880 seconds, whereas the average evaluation time of the PCNN-MM model is only 0.08 seconds. This means that the evaluation of the PCNN-MM model is 36000 times as fast as that of the PF-TLBM model. The high computational efficiency of the PCNN-MM model makes it to be a suitable surrogate model used in BO for process optimization so that the overall evaluation cost can be reduced. To find a feasible microstructure, the desired dendritic area is $S_d^* = 3000\ \mu m^2$ and microsegregation is $\chi^* = 10.0$. $T_{0,min} = 1918$ K and $T_{0,max} = 1928$ K are the lower and upper bound of the initial temperature, respectively. $\dot{T}_{min} = -10000$ K/s and $\dot{T}_{max} = -5000$ K/s are the lower and upper bound of the cooling rate, respectively.

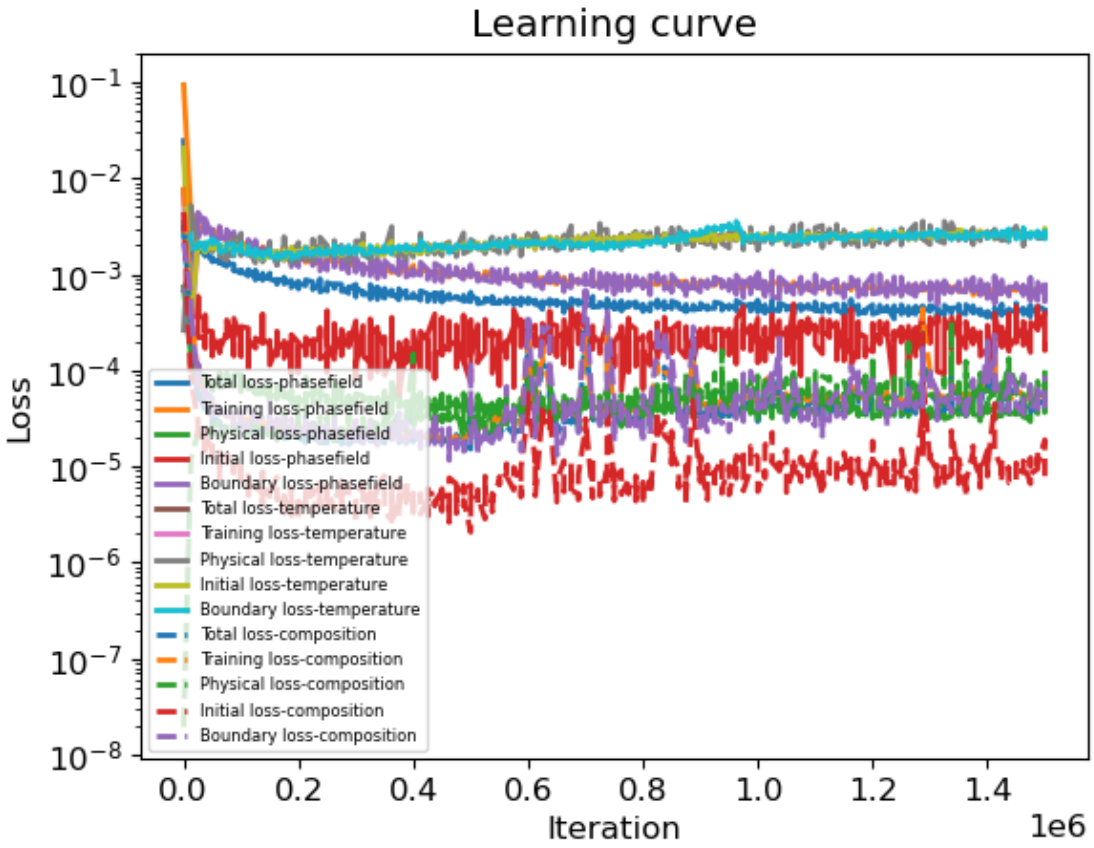


Figure 8.3. The learning curve during the training of the PCNN-MM.

8.4.1 Single-Objective Bayesian Optimization

The single-objective optimization problem described in Eq. (8.9) is first solved. The effects of the weights of individual objective functions on optimization results are investigated by sensitivity analysis. Since the scales of the dendritic area and the microsegregation are different, $S_{d,max} = 3500\ \mu m^2$ and $\chi_{max} = 14.0$ are used to normalize the objectives of the dendritic area and the microsegregation. This single-objective optimization problem is solved by single-objective BO from a python package called pyGPGO [236]. The EI acquisition function is used in single-objective BO. The BO stops once the maximum number of iterations of 100 is reached. The computation time for single iteration is about 0.54 seconds.

The simulation results from the PF-TLBM model and the predictions from the PCNN-MM at t = 10 ms under the optimal process parameters with different combinations of weights ($\alpha_1 = 0.25$ and $\alpha_2 = 0.75$; $\alpha_1 = 0.5$ and $\alpha_2 = 0.5$; $\alpha_1 = 0.75$ and $\alpha_2 = 0.25$) are shown in Figure 8.4, Figure 8.5, and Figure 8.6, respectively. The quantitative comparison between the PF-TLBM simulation result and PCNN-MM prediction during the Bayesian optimization is shown in Table 8.2. It is observed that the predicted primary arm from the PCNN-MM during the optimization agrees well with the simulation result, whereas the PCNN-MM cannot reveal the full details of secondary arms. Nevertheless, the predicted dendritic area and microsegregation agree well with the simulation results, which demonstrates the effectiveness of the PCNN-MM model to estimate the quantities of interest. When α_1 increases or α_2 decreases, the predicted dendritic areas are closer to the desired dendritic area, whereas the predicted microsegregations are further away from the desired microsegregation. This demonstrates that the weights of individual objective

functions can help the designer to produce desired microstructure. The relative errors of the dendritic area and the microsegregation for the PF-TLBM and the PCNN-MM are calculated based on the desired dendritic area and microsegregation. The relative errors of dendritic area for the PF-TLBM and the PCNN-MM are less than 5%, whereas the relative errors of microsegregation for both models are less than 12%. The relative errors of microsegregation are larger than those of dendritic area for both models because the desired microsegregation is 300 times as small as the desired dendritic area.

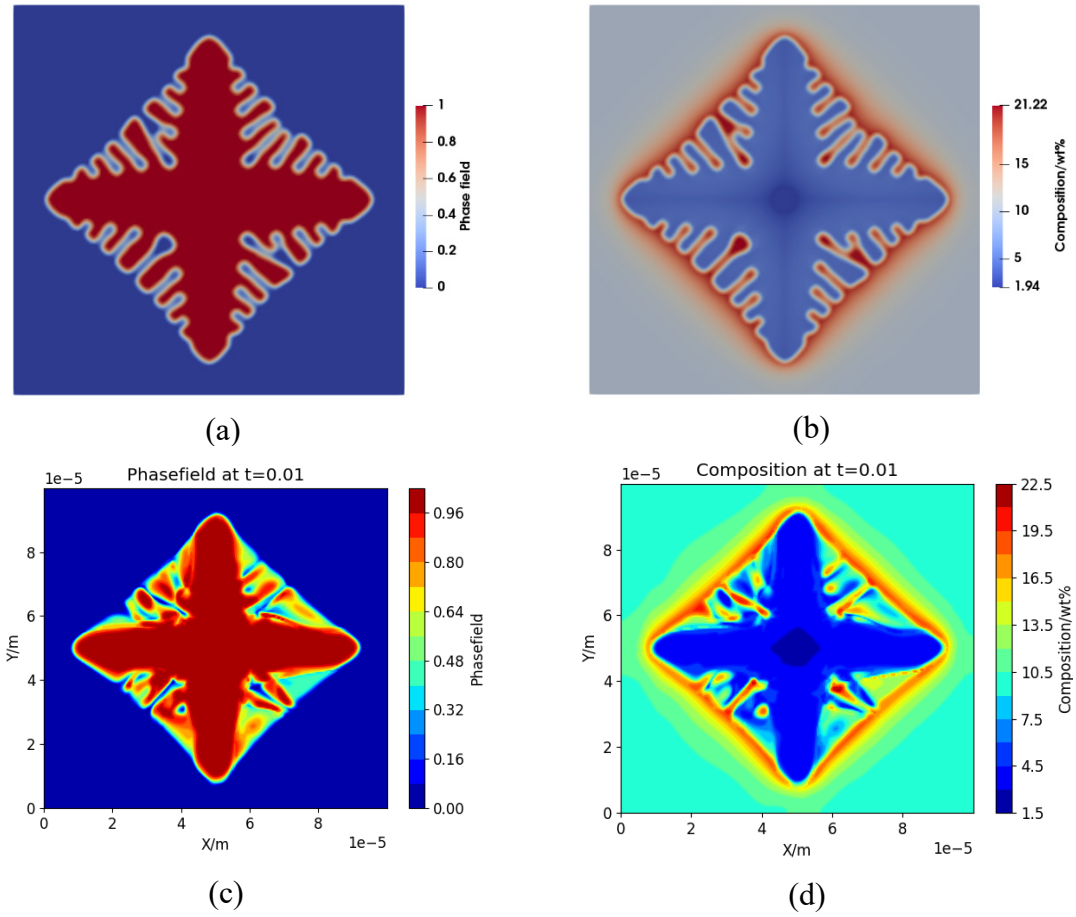


Figure 8.4. The simulation results from the PF-TLBM model and the predictions from the PCNN-MM at t = 10 ms under the optimal process parameters from single-objective BO when $\alpha_1 = 0.25$ and $\alpha_2 = 0.75$: (a) simulated phase field, (b) simulated composition field, (c) predicted phase field, and (d) predicted composition field.

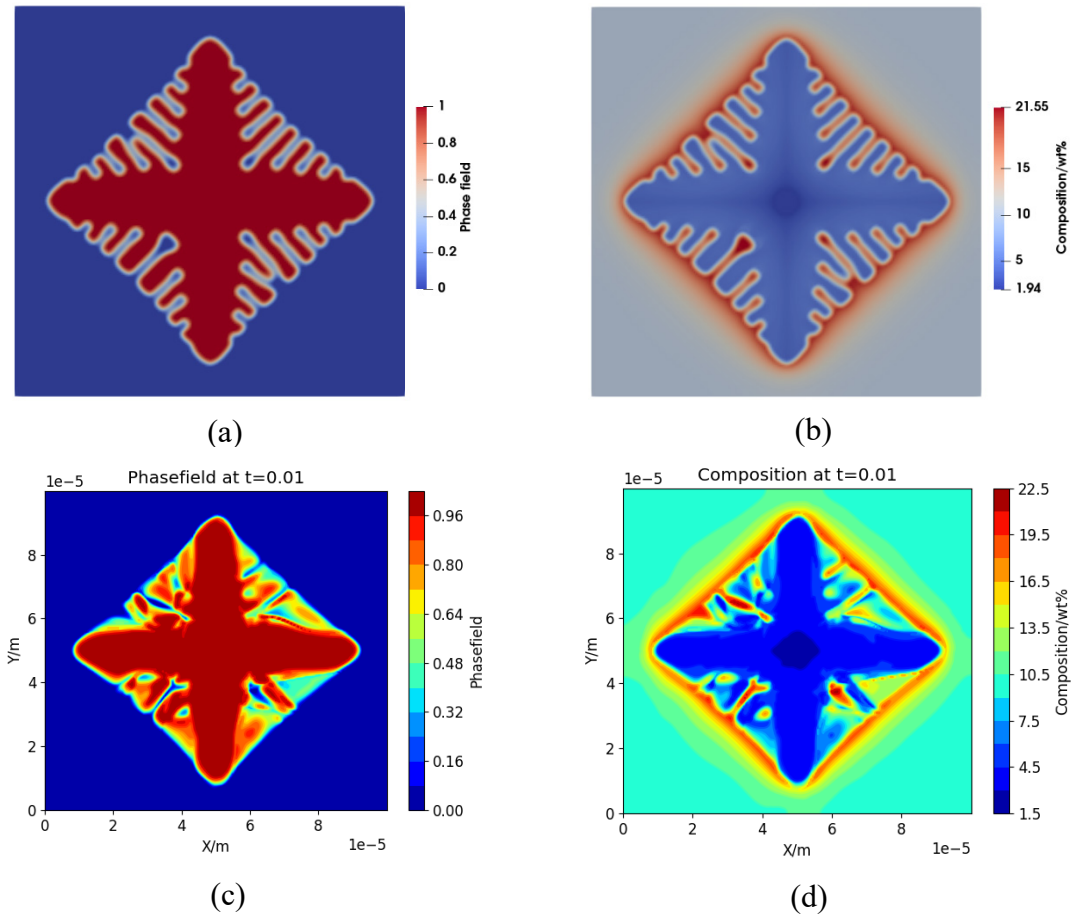


Figure 8.5. The simulation results from the PF-TLBM model and the predictions from the PCNN-MM at t = 10 ms under the optimal process parameters from single-objective BO when $\alpha_1 = 0.5$ and $\alpha_2 = 0.5$: (a) simulated phase field, (b) simulated composition field, (c) predicted phase field, and (d) predicted composition field.

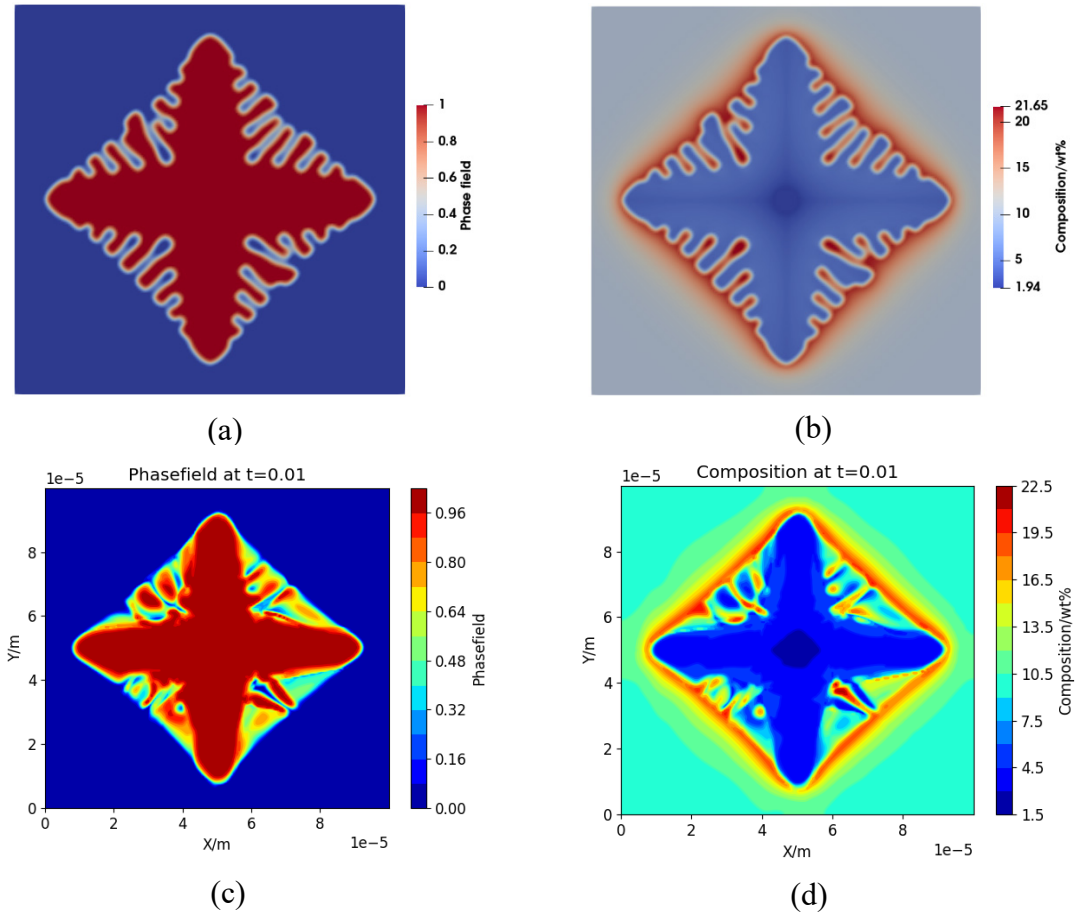


Figure 8.6. The simulation results from the PF-TLBM model and the predictions from the PCNN-MM at t = 10 ms under the optimal process parameters from single-objective BO when $\alpha_1 = 0.75$ and $\alpha_2 = 0.25$: (a) simulated phase field, (b) simulated composition field, (c) predicted phase field, and (d) predicted composition field.

Table 8.2. Quantitative analysis for single-objective Bayesian optimization

Weights of objectives	Optimal processing parameters	Model	S_d (μm^2)	χ	Relative error of S_d	Relative error of χ
$\alpha_1 = 0.25$ $\alpha_2 = 0.75$	$T_0^* = 1921.38$ K $\dot{T}^* = -8551.19$ K/s	PF-TLBM	2854.28	10.94	4.86%	9.40%
		PCNN-MM	2863.38	10.31	4.55%	3.10%
$\alpha_1 = 0.5$ $\alpha_2 = 0.5$	$T_0^* = 1922.80$ K $\dot{T}^* = -8907.11$ K/s	PF-TLBM	2906.16	11.11	3.13%	11.10%
		PCNN-MM	2919.20	10.44	2.69%	4.40%
$\alpha_1 = 0.75$ $\alpha_2 = 0.25$	$T_0^* = 1920.84$ K $\dot{T}^* = -8807.93$ K/s	PF-TLBM	2949.69	11.16	1.68%	11.60%
		PCNN-MM	2949.70	10.53	1.68%	5.30%

8.4.2 Multi-Objective Bayesian Optimization

The multi-objective optimization problem described in Eq. (8.8) is also applied. The multi-objective BO approach in Ref. [231] with the acquisition function in Eq. (8.6) defined by MHD and MOS is used to solve this problem. The multi-objective BO stops once the maximum number of iterations of 500 is reached. The computational time for single iteration is about 0.75 seconds. The Pareto front of the two-objective optimization problem of dendritic growth is shown in Figure 8.7. The original Pareto front based on 25 initial sampling points before the optimization is compared with the new Pareto front based on the updated sampling points after the optimization. The original process parameters of the initial sampling points at the original Pareto front before the optimization and the optimal process parameters at the new Pareto front after the optimization are shown in Figure 8.8.

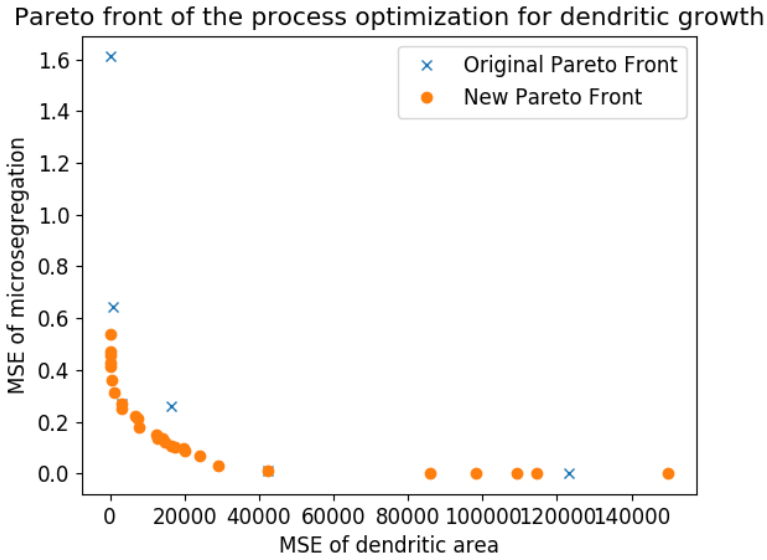


Figure 8.7. Pareto front of the two-objective optimization problem of dendritic growth.

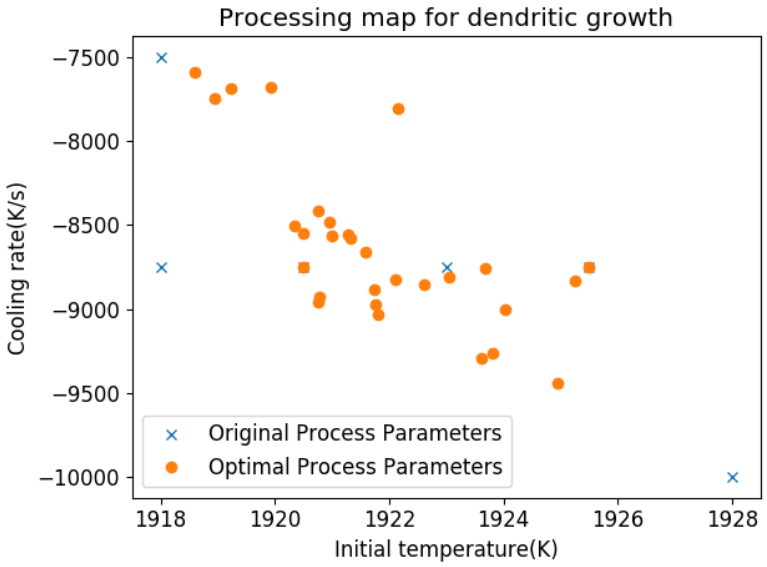


Figure 8.8. Processing map for dendritic growth.

One set of the optimal initial temperature $T_0^* = 1924.93$ K and cooling rate $\dot{T}^* = -9445.31$ K/s at the Pareto front are identified by multi-objective BO. The simulation results from the PF-TLBM model and the predictions from the PCNN-MM at t = 10 ms under the optimal process parameters are shown in Figure 8.9. The quantitative comparison

between the PF-TLBM simulation and the PCNN-MM prediction during the multi-objective Bayesian optimization is shown in Table 8.3. It is observed that the predicted dendritic area and microsegregation agree well with the simulation, which demonstrates the effectiveness of the PCNN-MM model to estimate quantities of interest. The relative errors of dendritic area for the PF-TLBM and the PCNN-MM model are less than 0.8%, whereas the relative errors of microsegregation for both models are less than 15%. Similarly, the relative errors of microsegregation for both models are large because the desired microsegregation is 300 times as small as the desired dendritic area. The optimization results demonstrate the feasibility and potential of the process design framework for process optimization. The advantage of using multi-objective BO is that the Pareto front and the processing map can be constructed after the optimization. There is no need to use a weighted-sum objective function, which can introduce prior bias by subjectively selecting the weights in the objective function. It is noted that the initial sampling points should cover the process window properly. This is because that the GP surrogate model used in BO is good at interpolation and bad at extrapolation. If the initial sampling points only occupy a small part of the process window, it may fail for BO to search optimal processing parameters.

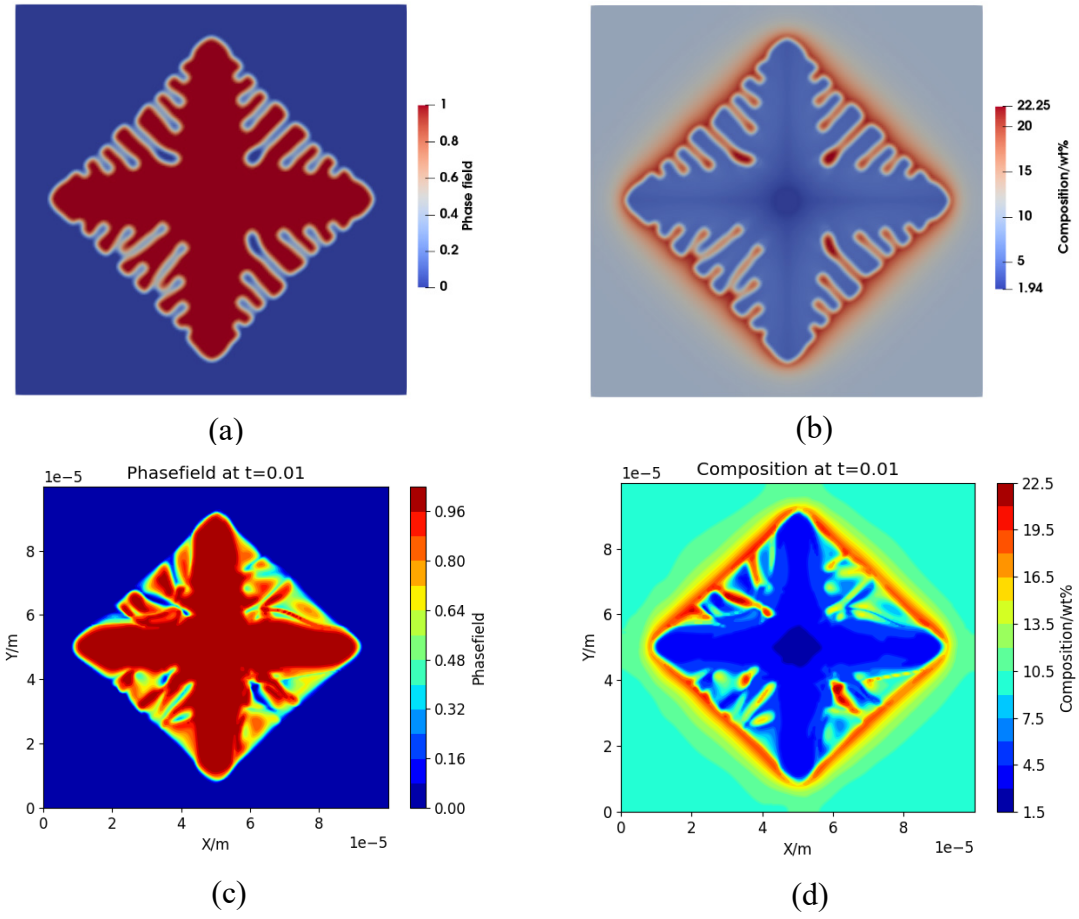


Figure 8.9. The simulation results from the PF-TLBM model and the predictions from the PCNN-MM at t = 10 ms under the optimal process parameters from multi-objective BO: (a) simulated phase field, (b) simulated composition field, (c) predicted phase field, and (d) predicted composition field.

Table 8.3. Quantitative analysis for multi-objective Bayesian optimization

Optimal processing parameters	Model	S_d (μm^2)	χ	Relative error of S_d	Relative error of χ
$T_0^* = 1924.93$ K $\dot{T}^* = -9445.31$ K/s	PF-TLBM	2977.40	11.47	0.75%	14.70%
	PCNN-MM	2998.85	10.73	0.04%	7.30%

8.4.3 Discussions

There are some limitations of the current constructed PCNN-MM surrogate model. First, since the PCNN-MM model is built for single material, the predictions from the PCNN-MM model may not be accurate when the material is changed. Second, the PCNN-MM model needs more training when the initial and boundary conditions are changed. Third, the PCNN-MM model only predicts the single dendritic growth rather than multiple dendritic growth. In future work, the simulation results of multiple dendritic growth for different materials and various initial and boundary conditions will be used to train the PCNN-MM model. The material properties, initial and boundary conditions can be added as input parameters of the PCNN-MM model to predict the process-structure relationship for different materials and various initial or boundary conditions. For instance, the multiple dendritic growth can be predicted by the PCNN-MM model by changing the number, locations, and orientations of nuclei in the initial condition of phase field. Domain decomposition [237,238] or coordinate mapping [239] can be used to handle complex boundary conditions in irregular geometric domains. It is known that nucleation affects the accuracy of simulated microstructures in metal AM. However, the heterogeneous nucleation is not predicted by the PCNN-MM model. As a stochastic process, nucleation can be modeled by adding Langevin noise terms in the kinetic equations of phase field and composition field. Therefore, the PCNN-MM model can potentially be used to predict nucleation and dendritic growth process by incorporating stochastic PDEs as physical constraints in the training process.

The constructed PCNN-MM model should not be regarded as the replacement of simulations. Rather, it provides an alternative to predict process-structure relationships

when high-fidelity simulations are computationally expensive. The advantages of physics-based simulations include high fidelity, good interpretability of results, and high accuracy if used as extrapolation, whereas their disadvantages include the reliance on prior physical knowledge and high computational cost for forward evaluation. The advantages of PCNN models include the low cost of forward evaluation and the capability of discovering unknown relationships. Yet they generally have some disadvantages for the fidelity and interpretability of results, as well as low accuracy in extrapolation, in comparison with simulations. The use of both simulations and PCNNs is expected in constructing high-dimensional P-S-P relationships at an affordable cost.

The PCNN-MM surrogate model can have three potential applications. First, the PCNN-MM model enhances the efficiency of process design when high-fidelity simulations need to be run repetitively to obtain samples for design optimization. Simulations can provide training data to train the PCNN-MM model, which is then used as the surrogate model for process design. Once optimal process parameters are identified, high-fidelity simulations can be used for verification. The required number of samples for high-dimensional optimization problems usually is very large. The cost of training PCNNs therefore can only be justified for complex problems with a high-dimensional searching space. Second, compared to high-fidelity simulations, the PCNN-MM model makes it possible to predict microstructure evolution in a larger domain with a lower computational cost. By using domain decomposition, the trained PCNN-MM model can be run multiple times to predict multiple dendritic growth in subdomains. The challenge is to take care of boundary conditions between subdomains. Third, the PCNN-MM model can be used in the digital twins [240] of AM processes to guide real-time process control, which has a high

demand for computational efficiency. In metal AM processes, the real-time thermal data can be collected by sensors and passed to the PCNN-MM model for predicting microstructure evolution. If the generated microstructure does not meet the requirement, then process parameters can be adjusted quickly.

8.5 Conclusions

In this chapter, a new generic process design framework is developed. The dendritic growth of Ti-6Al-4V alloy is simulated using the developed PF-TLBM model under various initial temperatures and cooling rates. A surrogate model of process-structure relationships is constructed based on the PCNN-MM trained by the simulation data from PF-TLBM. The evaluation of the PCNN-MM model after training is 36000 times as fast as that of the PF-TLBM model. The constructed surrogate model is used in single-objective and multi-objective BO to search the optimal process parameters so that the desired dendritic area and microsegregation can be achieved. For single-objective BO, the effects of the weights of individual objective functions on optimization results are investigated by sensitivity analysis. The computational results demonstrate that the weights of individual objective functions can help the designer to produce desired microstructure.

The Pareto front and processing map for dendritic growth are constructed by multi-objective BO. For multi-objective BO, the relative errors of dendritic area for the PF-TLBM and the PCNN-MM are less than 0.8%, whereas the relative errors of microsegregation for both models are less than 15%. The relative errors of microsegregation for both models are large because the desired microsegregation is 300 times as small as the desired dendritic area. The computational results demonstrate the feasibility and potential of the process design framework for process optimization. The

advantage of using multi-objective BO is that the Pareto front and the processing map can be constructed after the optimization. It is noted that the initial sampling points should cover the process window properly so that BO can search optimal processing parameters.

In future work, the generic process design framework will be extended for materials design so that the optimal process parameters can be searched to reach desired materials properties by using the constructed surrogate. To improve the efficiency of GP and BO for large sample sizes, reduced-rank sparse matrix approximation methods will be used to estimate the inverse of the covariance matrix (the main computational bottleneck of GP) where only a subset of samples (inducing points) will be selected to construct GP.

CHAPTER 9. CONCLUSIONS

9.1 Summary of the Work

The objective of this research is to investigate the feasibility of a hybrid physics-based data-driven process optimization approach to establish reliable surrogates of process-structure-property relationships for metal additive manufacturing aided by mesoscale multiphysics simulation and physics-constrained machine learning. A new mesoscale multiphysics simulation model called PF-TLBM is developed to reveal the details of rapid solidification with predictions of grain morphology and composition distributions in microstructures. To reduce the computational costs in establishing the process-structure relationships, a new physics-constrained machine learning model called PCNN-MM is developed to serve as the surrogate to predict grain morphology and alloy compositions from process parameters, after being trained with simulation data and experimental data. After training, PCNN-MM can predict microstructures from process parameters efficiently without relying on expensive simulations or experiments. The predicted process-structure mappings can be applied to construct the process-structure surrogate relationship and perform optimization. A multi-objective Bayesian optimization approach is used to search the optimal process parameters so that the desired grain morphology and composition distributions can be achieved.

In CHAPTER 3, PF-TLBM is developed to simulate rapid solidification of Ti-6Al-4V alloy by concurrently coupling solute transport, heat transfer, latent heat, fluid dynamics, and phase transition. In this model, the phase-field method simulates the dendrite growth of alloys, whereas the thermal lattice Boltzmann method models heat

transfer and fluid flow. The phase-field method and the thermal lattice Boltzmann method are tightly coupled. The effects of latent heat and melt flow on the dendrite growth are investigated. This work also contains experimental comparison, sensitivity analysis of mesh sizes, as well as quantitative analyses of the temperature gradient, growth velocity, and their combinations. The simulation results of Ti-6Al-4V show that the consideration of latent heat is necessary because it reveals the details of the formation of secondary arms and provides more realistic kinetics of dendrite growth. The effect of fluid flow on dendrite growth is small under rapid solidification. The proposed multi-physics simulation model provides new insights into the complex solidification process in AM.

In CHAPTER 4, a nucleation model is introduced in the PF-TLBM model to simulate heterogeneous nucleation at the boundary of the melt pool in SLM of AlSi10Mg alloy. A new method is proposed to compute heat fluxes for a 2D small melt pool in order to approximate the actual non-isothermal temperature field in SLM. The effects of latent heat and cooling rate on dendritic morphology and solute distribution are studied. The qualitative and quantitative analyses show that the inclusion of latent heat is necessary because it reveals the details of the formation of secondary arms, reduces the overestimation of microsegregation, and provides more realistic kinetics of dendritic growth. A higher cooling rate results in faster liquid-solid phase transition and higher microsegregation at grain boundaries. The PF-TLBM model is also extended to predict the multi-layer epitaxial grain growth in the complex heating and cooling environment in SLM. To save the computational cost, the Rosenthal equation method is used to predict the thermal history of the melt pool. A marching cell simulation scheme is used to further

reduce the computational cost. This work demonstrates the capability of the PF-TLBM model to simulate multi-layer printing processes in SLM.

In CHAPTER 5, a novel MF-PCNN is proposed to reduce the required amount of training data, where physical knowledge is applied to constrain neural networks, and multi-fidelity networks are constructed to improve training efficiency. A low-cost low-fidelity physics-constrained neural network is used as the baseline model, whereas a limited amount of data from a high-fidelity physics-constrained neural network is used to train a second neural network to predict the difference between the two models. The proposed framework is demonstrated with 2D heat transfer, phase transition, and dendritic growth problems, which are fundamental in materials modeling. Physics is described by PDEs. With the same set of training data, the prediction errors of PCNNs can be one order of magnitude lower than that of the classical ANN without physical constraints. The accuracy of the prediction is comparable to those from direct numerical solutions of equations.

In CHAPTER 6, a new PCNN-MM is proposed to adjust the weights of different losses systematically. The training of the PCNN-MM is to solve a minimax problem and search for the high-order saddle points of the nonconvex-nonconcave loss function. To address the challenges of searching high-order saddle points, a novel saddle point search algorithm called Dual-Dimer method is proposed, where only first derivatives need to be calculated. The local convergence of the Dual-Dimer method is analyzed. The performance of the Dual-Dimer method is evaluated with three analytical nonconvex-nonconcave loss functions. It was shown that the Dual-Dimer method is computationally more efficient than the GDA method to find high-order saddle points in these analytical functions. The Dual-Dimer method also provides additional eigenvalue information to make sure that the

desired high-order saddle points are found at the end of the training. A heat transfer example is used to demonstrate the effectiveness of the PCNN-MM, where its convergence is faster than that of the original PCNN with the adaptive weighting scheme. To further reduce the computational cost, an MF-PCNN-MM is developed to integrate the LF and HF data. The MF-PCNN-MM is demonstrated by three examples. The computational results demonstrate the effectiveness of the MF-PCNN-MM.

In CHAPTER 7, a new sequential training scheme is developed to improve the convergence and prediction accuracy of PCNN-MMs in solving multiphysics problems. A new saddle point search algorithm called DD-CS is also developed to alleviate the curse of dimensionality in searching high-order saddle points during the training. The developed PCNN-MM and the DD-CS algorithm are demonstrated by two examples: thermal dendritic growth example and thermo-solutal dendritic growth example. In each example, different ML models with various training schemes are compared with each other. The computational results suggest that the sequential training scheme is better than the concurrent sequential training scheme since it helps the convergence of the PCNN-MM. The predicted phase field from the PCNN-MM with the DD-CS algorithm is slightly worse than that from the PCNN-MM with the Dual-Dimer algorithm. The spikes in the learning curve are designed to help the training of the PCNN-MM to escape the local saddle point so that the neural network may converge to a better saddle point.

In CHAPTER 8, a new generic process design framework is developed. The dendritic growth of Ti-6Al-4V alloy is simulated using the developed PF-TLBM model under various initial temperatures and cooling rates. A surrogate model of process-structure relationships is constructed based on the PCNN-MM trained by the simulation data from

PF-TLBM. The constructed surrogate model is used in single-objective and multi-objective BO to search the optimal process parameters so that the desired dendritic area and microsegregation can be achieved. For single-objective BO, the effects of the weights of individual objective functions on optimization results are investigated through sensitivity analyses. The computational results demonstrate that the weights of individual objective functions can help the designer to produce desired microstructure. The Pareto front and processing map for dendritic growth are constructed by multi-objective BO. The computational results demonstrate the feasibility and potential of the process design framework for process optimization.

9.2 Contributions of the Dissertation

The novel contributions of the dissertation are highlighted as follows.

- A new mesoscale multiphysics simulation model called PF-TLBM is developed to predict microstructure evolution of alloys in rapid solidification by concurrently coupling heterogeneous nucleation, solute transport, heat transfer, latent heat, fluid dynamics, and phase transition. A new method is proposed to compute heat fluxes for a two-dimensional small melt pool in order to approximate the actual non-isothermal temperature field in metal AM processes. To reduce the computational cost, a new marching cell simulation scheme is developed and introduced in the PF-TLBM model to predict the multi-layer epitaxial grain growth in metal AM processes. The developed PF-TLBM model enables a deeper understanding of the rapid solidification process in metal AM and makes it possible to simulate the microstructure evolution within the whole building part.

- A novel physics-based machine learning model called MF-PCNN is proposed to reduce the required amount of training data, and multi-fidelity networks are constructed to improve training efficiency. A new PCNN-MM is proposed to adjust the weights of different losses systematically. A novel and general saddle point search algorithm called Dual-Dimer method is proposed to train the developed PCNN-MM. To reduce the computational cost, a new MF-PCNN-MM is developed to integrate the LF and HF data. A new sequential training scheme is developed to aid the convergence of PCNN-MMs for solving multiphysics problems. A new saddle point search algorithm called DD-CS is also developed to alleviate the curse of dimensionality in searching high-order saddle points during the training. The developed physics-constrained machine learning models and training algorithms are promising approaches to alleviate the curse of dimensionality in general ML applications.
- A new generic process design framework is developed for process optimization. A surrogate model of process-structure relationships is constructed based on the PCNN-MM trained by the simulation data from PF-TLBM. The constructed surrogate model is used in multi-objective BO to search the optimal process parameters so that the desired dendritic area and composition distribution can be achieved.

9.3 Future Work

In future work, the developed PF-TLBM model will be validated with ex-situ and in-situ experiments by comparing both phase field and composition field. Qualitative

comparison of grain morphology and texture is not enough to validate simulation models. A more comprehensive approach is needed to compare both phase field and composition field quantitatively, since the composition of the solidified workpiece also significantly affects its mechanical properties (e.g., precipitation strengthening) and electrochemical properties (e.g., corrosion resistance). Quantities such as primary arm spacing, composition distribution, grain size distribution, and grain orientation distribution will be compared in model validation.

The PF-TLBM model will be extended to simulate columnar-to-equiaxed transition, solid-state phase transformation, as well as multi-component PFM so that the more complete AM microstructure formation process can be modeled. Future extensions will also include 3D modeling, where efficient parallel algorithms will be developed. Note that there is still missing physics in the PF-TLBM simulation, such as Marangoni flow, defect/pore generation, and alloying element loss due to vaporization. The inclusion of those aspects in the future simulation framework will be necessary to model the variation of metal AM processes. The PF-TLBM model will be used to simulate the grain growth within the whole workpiece in decimeters during the SLM process in future work. A moving Gaussian heat source will be introduced in the thermal lattice Boltzmann method to predict the accurate thermal history. Furthermore, the empirical model parameters, such as interface energy, interface mobility, nucleation rate, and nucleation activation energy, need to be determined and calibrated based on experimental measurements, first-principles calculations, or atomistic simulations. The model form and parameter uncertainties associated with the developed model should be quantified to provide more confidence in the prediction.

In future work, the ANN architecture in the PCNN-MM could be replaced by recurrent neural networks, such as long short-term memory neural networks, which may be more appropriate to solve time-dependent problems. To further improve the training efficiency of the MF-PCNN and the MF-PCNN-MM, a sequential sampling strategy can be adopted to obtain an optimal combination of the HF and LF sample points for a given computational budget. The adjustment of hyperparameters for the Dual-Dimer algorithm and the DD-CS algorithm is based on sensitivity studies. In future work, a more systematic method to find the optimal hyperparameters will be developed so that the computational efficiency of both algorithms can be further improved. The developed MF-PCNN, PCNN-MM, and MF-PCNN-MM can be applied in other general applications such as battery life prediction, climate modeling, and others. The generic saddle point search method including the Dual-Dimer algorithm and the DD-CS algorithm will be applied to solve other minimax problems, which arise from game theory, generative adversarial networks, and robust optimization. A new method will be developed to quantify the uncertainties from the predictions of PCNNs.

In future work, other simulation models such as crystal plasticity can be adopted to predict mechanical properties of predicted microstructure from the PF-TLBM model. In this way, the surrogate model of complete process-structure-property relationships can be constructed using the PF-TLBM model, the PCNN-MM, and the crystal plasticity model. The generic process design framework will be extended for materials design so that the optimal process parameters can be searched to reach desired materials properties by using the constructed surrogate. To improve the efficiency of GP and BO for large sample sizes, reduced-rank sparse matrix approximation methods will be used to estimate the inverse of

the covariance matrix (the main computational bottleneck of GP) where only a subset of samples (inducing points) will be selected to construct GP.

REFERENCES

[1] Yang, L., Harrysson, O., West, H., and Cormier, D., 2012, "Compressive Properties of Ti-6Al-4V Auxetic Mesh Structures Made by Electron Beam Melting," Acta Materialia, **60**(8), pp. 3370–3379.

[2] Guo, C., Ge, W., and Lin, F., 2015, "Dual-Material Electron Beam Selective Melting: Hardware Development and Validation Studies," Engineering, **1**(1), pp. 124–130.

[3] Dehoff, R. R., Kirka, M., Sames, W. J., Bilheux, H., Tremsin, A. S., Lowe, L. E., and Babu, S. S., 2015, "Site Specific Control of Crystallographic Grain Orientation through Electron Beam Additive Manufacturing," Materials Science and Technology, **31**(8), pp. 931–938.

[4] Raghavan, N., Simunovic, S., Dehoff, R., Plotkowski, A., Turner, J., Kirka, M., and Babu, S., 2017, "Localized Melt-Scan Strategy for Site Specific Control of Grain Size and Primary Dendrite Arm Spacing in Electron Beam Additive Manufacturing," Acta Materialia, **140**, pp. 375–387.

[5] Hadadzadeh, A., Amirkhiz, B. S., Li, J., and Mohammadi, M., 2018, "Columnar to Equiaxed Transition during Direct Metal Laser Sintering of AlSi10Mg Alloy: Effect of Building Direction," Additive Manufacturing, **23**, pp. 121–131.

[6] Ladewig, A., Schlick, G., Fisser, M., Schulze, V., and Glatzel, U., 2016, "Influence of the Shielding Gas Flow on the Removal of Process By-Products in the Selective Laser Melting Process," Additive Manufacturing, **10**, pp. 1–9.

[7] Lu, B., Cui, X., Jiang, L., Liu, E., Zhang, D., Feng, X., Dong, M., and Jin, G., 2019, "Influence of Electromagnetic Stirring on Microstructure and Wear Resistance of Plasma Arc Deposited Shape Memory Alloy," Surface and Coatings Technology, **359**, pp. 125–131.

[8] Kirka, M. M., Nandwana, P., Lee, Y., and Dehoff, R. R., 2017, "Solidification and Solid-State Transformation Sciences in Metals Additive Manufacturing," Scripta Materialia, **135**, pp. 130–134.

[9] Schwendner, K. I., Banerjee, R., Collins, P. C., Brice, C. A., and Fraser, H. L., 2001,

"Direct Laser Deposition of Alloys from Elemental Powder Blends," Scripta Materialia, **45**(10), pp. 1123–1129.

[10] Yan, L., Chen, X., Li, W., Newkirk, J., and Liou, F., 2016, "Direct Laser Deposition of Ti-6Al-4V from Elemental Powder Blends," Rapid Prototyping Journal, **22**(5), pp. 810–816.

[11] Martin, J. H., Yahata, B. D., Hundley, J. M., Mayer, J. A., Schaedler, T. A., and Pollock, T. M., 2017, "3D Printing of High-Strength Aluminium Alloys," Nature, **549**(7672), pp. 365–369.

[12] Zhang, H., Zhu, H., Qi, T., Hu, Z., and Zeng, X., 2016, "Selective Laser Melting of High Strength Al–Cu–Mg Alloys: Processing, Microstructure and Mechanical Properties," Materials Science and Engineering: A, **656**, pp. 47–54.

[13] Liu, S., and Shin, Y. C., 2019, "Additive Manufacturing of Ti6Al4V Alloy: A Review," Materials and Design, **164**, p. 107552.

[14] DebRoy, T., Wei, H. L., Zuback, J. S., Mukherjee, T., Elmer, J. W., Milewski, J. O., Beese, A. M., Wilson-Heid, A., De, A., and Zhang, W., 2018, "Additive Manufacturing of Metallic Components – Process, Structure and Properties," Progress in Materials Science, **92**, pp. 112–224.

[15] Bourell, D., Kruth, J. P., Leu, M., Levy, G., Rosen, D., Beese, A. M., and Clare, A., 2017, "Materials for Additive Manufacturing," CIRP Annals, **66**(2), pp. 659–681.

[16] Smith, J., Xiong, W., Yan, W., Lin, S., Cheng, P., Kafka, O. L., Wagner, G. J., Cao, J., and Liu, W. K., 2016, "Linking Process, Structure, Property, and Performance for Metal-Based Additive Manufacturing: Computational Approaches with Experimental Support," Computational Mechanics, **57**(4), pp. 583–610.

[17] Liu, J., Jalalahmadi, B., Guo, Y. B., Sealy, M. P., and Bolander, N., 2018, "A Review of Computational Modeling in Powder-Based Additive Manufacturing for Metallic Part Qualification," Rapid Prototyping Journal, **24**(8), pp. 1245–1264.

[18] Markl, M., and Körner, C., 2016, "Multiscale Modeling of Powder Bed–Based Additive Manufacturing," Annual Review of Materials Research, **46**(1), pp. 93–123.

[19] Panwisawas, C., Qiu, C., Anderson, M. J., Sovani, Y., Turner, R. P., Attallah, M.

M., Brooks, J. W., and Basoalto, H. C., 2017, "Mesoscale Modelling of Selective Laser Melting: Thermal Fluid Dynamics and Microstructural Evolution," Computational Materials Science, **126**, pp. 479–490.

[20] Manvatkar, V., De, A., and DebRoy, T., 2014, "Heat Transfer and Material Flow during Laser Assisted Multi-Layer Additive Manufacturing," Journal of Applied Physics, **116**(12), p. 124905.

[21] Lee, Y. S., and Zhang, W., 2016, "Modeling of Heat Transfer, Fluid Flow and Solidification Microstructure of Nickel-Base Superalloy Fabricated by Laser Powder Bed Fusion," Additive Manufacturing, **12**, pp. 178–188.

[22] King, W., Anderson, A. T., Ferencz, R. M., Hodge, N. E., Kamath, C., and Khairallah, S. A., 2015, "Overview of Modelling and Simulation of Metal Powder Bed Fusion Process at Lawrence Livermore National Laboratory," Materials Science and Technology, **31**(8), pp. 957–968.

[23] Yan, W., Qian, Y., Ge, W., Lin, S., Liu, W. K., Lin, F., and Wagner, G. J., 2018, "Meso-Scale Modeling of Multiple-Layer Fabrication Process in Selective Electron Beam Melting: Inter-Layer/Track Voids Formation," Materials & Design, **141**, pp. 210–219.

[24] Chen, Q., Guillemot, G., Gandin, C.-A., and Bellet, M., 2017, "Three-Dimensional Finite Element Thermomechanical Modeling of Additive Manufacturing by Selective Laser Melting for Ceramic Materials," Additive Manufacturing, **16**, pp. 124–137.

[25] Pan, H., and Liou, F., 2005, "Numerical Simulation of Metallic Powder Flow in a Coaxial Nozzle for the Laser Aided Deposition Process," Journal of Materials Processing Technology, **168**(2), pp. 230–244.

[26] Wen, S. Y., Shin, Y. C., Murthy, J. Y., and Sojka, P. E., 2009, "Modeling of Coaxial Powder Flow for the Laser Direct Deposition Process," International Journal of Heat and Mass Transfer, **52**(25–26), pp. 5867–5877.

[27] Li, C., Fu, C. H., Guo, Y. B., and Fang, F. Z., 2016, "A Multiscale Modeling Approach for Fast Prediction of Part Distortion in Selective Laser Melting," Journal of Materials Processing Technology, **229**, pp. 703–712.

[28] Denlinger, E. R., Irwin, J., and Michaleris, P., 2014, "Thermomechanical Modeling

of Additive Manufacturing Large Parts," Journal of Manufacturing Science and Engineering, **136**(6).

[29] Hodge, N. E., Ferencz, R. M., and Solberg, J. M., 2014, "Implementation of a Thermomechanical Model for the Simulation of Selective Laser Melting," Computational Mechanics, **54**(1), pp. 33–51.

[30] Rodgers, T. M., Madison, J. D., and Tikare, V., 2017, "Simulation of Metal Additive Manufacturing Microstructures Using Kinetic Monte Carlo," Computational Materials Science, **135**, pp. 78–89.

[31] Liu, J., and To, A. C., 2017, "Additive Manufacturing Quantitative Texture Prediction of Epitaxial Columnar Grains in Additive Manufacturing Using Selective Laser Melting," Additive Manufacturing, **16**, pp. 58–64.

[32] Tabei, A., Mirkoohi, E., Garmestani, H., and Liang, S., 2019, "Modeling of Texture Development in Additive Manufacturing of Ni-Based Superalloys," The International Journal of Advanced Manufacturing Technology, **103**(1–4), pp. 1057–1066.

[33] Jaafar, M. A., Rousse, D. R., Gibout, S., and Bédécarrats, J. P., 2017, "A Review of Dendritic Growth during Solidification: Mathematical Modeling and Numerical Simulations," Renewable and Sustainable Energy Reviews, **74**, pp. 1064–1069.

[34] Tan, W., Bailey, N. S., and Shin, Y. C., 2011, "A Novel Integrated Model Combining Cellular Automata and Phase Field Methods for Microstructure Evolution during Solidification of Multi-Component and Multi-Phase Alloys," Computational Materials Science, **50**(9), pp. 2573–2585.

[35] Yin, H., and Felicelli, S. D., 2010, "Dendrite Growth Simulation during Solidification in the LENS Process," Acta Materialia, **58**(4), pp. 1455–1465.

[36] Liu, S., and Shin, Y. C., 2020, "Prediction of 3D Microstructure and Phase Distributions of Ti6Al4V Built by the Directed Energy Deposition Process via Combined Multi-Physics Models," Additive Manufacturing, **34**, p. 101234.

[37] Rolchigo, M. R., and LeSar, R., 2018, "Modeling of Binary Alloy Solidification under Conditions Representative of Additive Manufacturing," Computational Materials Science, **150**, pp. 535–545.

[38] Zhang, J., Liou, F., Seufzer, W., and Taminger, K., 2016, "A Coupled Finite Element Cellular Automaton Model to Predict Thermal History and Grain Morphology of Ti-6Al-4V during Direct Metal Deposition (DMD)," Additive Manufacturing, **11**, pp. 32–39.

[39] Li, X., and Tan, W., 2018, "Numerical Investigation of Effects of Nucleation Mechanisms on Grain Structure in Metal Additive Manufacturing," Computational Materials Science, **153**, pp. 159–169.

[40] Steinbach, I., 2009, "Phase-Field Models in Materials Science," Modelling and Simulation in Materials Science and Engineering, **17**(7), p. 073001.

[41] Steinbach, I., 2013, "Why Solidification? Why Phase-Field?," Jom, **65**(9), pp. 1096–1102.

[42] Zaeem, M. A., Yin, H., and Felicelli, S. D., 2012, "Comparison of Cellular Automaton and Phase Field Models to Simulate Dendrite Growth in Hexagonal Crystals," Journal of Materials Science and Technology, **28**(2), pp. 137–146.

[43] Choudhury, A., Reuther, K., Wesner, E., August, A., Nestler, B., and Rettenmayr, M., 2012, "Comparison of Phase-Field and Cellular Automaton Models for Dendritic Solidification in Al-Cu Alloy," Computational Materials Science, **55**, pp. 263–268.

[44] Chen, R., Xu, Q., and Liu, B., 2014, "A Modified Cellular Automaton Model for the Quantitative Prediction of Equiaxed and Columnar Dendritic Growth," Journal of Materials Science & Technology, **30**(12), pp. 1311–1320.

[45] Asle Zaeem, M., Yin, H., and Felicelli, S. D., 2013, "Modeling Dendritic Solidification of Al-3%Cu Using Cellular Automaton and Phase-Field Methods," Applied Mathematical Modelling, **37**(5), pp. 3495–3503.

[46] Pan, S., and Zhu, M., 2010, "A Three-Dimensional Sharp Interface Model for the Quantitative Simulation of Solutal Dendritic Growth," Acta Materialia, **58**(1), pp. 340–352.

[47] Wang, W., Lee, P. D., and McLean, M., 2003, "A Model of Solidification Microstructures in Nickel-Based Superalloys: Predicting Primary Dendrite Spacing Selection," Acta Materialia, **51**(10), pp. 2971–2987.

[48] Beltran-Sanchez, L., and Stefanescu, D. M., 2004, "A Quantitative Dendrite Growth Model and Analysis of Stability Concepts," Metallurgical and Materials Transactions A, **35**(8), pp. 2471–2485.

[49] Reuther, K., and Rettenmayr, M., 2014, "Perspectives for Cellular Automata for the Simulation of Dendritic Solidification - A Review," Computational Materials Science, **95**, pp. 213–220.

[50] Kurz, W., Rappaz, M., and Trivedi, R., 2020, "Progress in Modelling Solidification Microstructures in Metals and Alloys . Part II : Dendrites from 2001 to 2018," International Materials Reviews, **66**(1), pp. 30–76.

[51] Boettinger, W. J., Warren, J. A., Beckermann, C., and Karma, A., 2002, "Phase Field Simulation of Solidification," Annual Review Materials Research, **32**(1), pp. 63–94.

[52] Chen, L.-Q., 2002, "Phase -Field Models for Microstructure Evolution," Annual Review of Materials Research, **32**(1), pp. 113–140.

[53] Moelans, N., Blanpain, B., and Wollants, P., 2008, "An Introduction to Phase-Field Modeling of Microstructure Evolution," Calphad: Computer Coupling of Phase Diagrams and Thermochemistry, **32**(2), pp. 268–294.

[54] Böttger, B., Eiken, J., and Steinbach, I., 2006, "Phase Field Simulation of Equiaxed Solidification in Technical Alloys," Acta Materialia, **54**(10), pp. 2697–2704.

[55] Gong, X., and Chou, K., 2015, "Phase-Field Modeling of Microstructure Evolution in Electron Beam Additive Manufacturing," Jom, **67**(5), pp. 1176–1182.

[56] Sahoo, S., and Chou, K., 2016, "Phase-Field Simulation of Microstructure Evolution of Ti-6Al-4V in Electron Beam Additive Manufacturing Process," Additive Manufacturing, **9**, pp. 14–24.

[57] Shimono, Y., Oba, M., Nomoto, S., Koizumi, Y., and Chiba, A., 2017, "Numerical Simulation of Solidification in Additive Manufacturing of Ti Alloy by Multi-Phase Field Method," *Solid Freeform Fabrication Symposium*, pp. 1048–1057.

[58] Gránásy, L., Börzsönyi, T., and Pusztai, T., 2002, "Crystal Nucleation and Growth in Binary Phase-Field Theory," Journal of Crystal Growth, **237**–**239**(1–4), pp. 1813–1817.

[59] Gránásy, L., Börzsönyi, T., Börzsönyi, T., and Pusztai, T., 2002, "Nucleation and Bulk Crystallization in Binary Phase Field Theory," Physical Review Letters, **88**(20), p. 206105.

[60] Gránásy, L., Pusztai, T., Saylor, D., and Warren, J. A., 2007, "Phase Field Theory of Heterogeneous Crystal Nucleation," Physical Review Letters, **98**(3), p. 035703.

[61] Pusztai, T., Tegze, G., Tóth, G. I., Környei, Ĺ., Bansel, G., Fan, Z., and Gŕńsy, Ĺ., 2008, "Phase-Field Approach to Polycrystalline Solidification Including Heterogeneous and Homogeneous Nucleation," Journal of Physics Condensed Matter, **20**(40), p. 404205.

[62] Simmons, J. P., Wen, Y., Shen, C., and Wang, Y. Z., 2004, "Microstructural Development Involving Nucleation and Growth Phenomena Simulated with the Phase Field Method," Materials Science and Engineering A, **365**(1–2), pp. 136–143.

[63] Li, J., Wang, J., and Yang, G., 2007, "Phase-Field Simulation of Microstructure Development Involving Nucleation and Crystallographic Orientations in Alloy Solidification," Journal of Crystal Growth, **309**(1), pp. 65–69.

[64] Lian, Y., Gan, Z., Yu, C., Kats, D., Liu, W. K., and Wagner, G. J., 2019, "A Cellular Automaton Finite Volume Method for Microstructure Evolution during Additive Manufacturing," Materials and Design, **169**, p. 107672.

[65] Körner, C., Attar, E., and Heinl, P., 2011, "Mesoscopic Simulation of Selective Beam Melting Processes," Journal of Materials Processing Technology, **211**(6), pp. 978–987.

[66] Rai, A., Markl, M., and Körner, C., 2016, "A Coupled Cellular Automaton–Lattice Boltzmann Model for Grain Structure Simulation during Additive Manufacturing," Computational Materials Science, **124**, pp. 37–48.

[67] Jelinek, B., Eshraghi, M., Felicelli, S., and Peters, J. F., 2014, "Large-Scale Parallel Lattice Boltzmann-Cellular Automaton Model of Two-Dimensional Dendritic Growth," Computer Physics Communications, **185**(3), pp. 939–947.

[68] Nabavizadeh, S. A., Eshraghi, M., Felicelli, S. D., Tewari, S. N., and Grugel, R. N., 2019, "Effect of Bubble-Induced Marangoni Convection on Dendritic Solidification," International Journal of Multiphase Flow, **116**, pp. 137–152.

[69] Liu, P. W., Ji, Y. Z., Wang, Z., Qiu, C. L., Antonysamy, A. A., Chen, L. Q., Cui, X. Y., and Chen, L., 2018, "Investigation on Evolution Mechanisms of Site-Specific Grain Structures during Metal Additive Manufacturing," Journal of Materials Processing Technology, **257**, pp. 191–202.

[70] Wang, X., and Chou, K., 2019, "Microstructure Simulations of Inconel 718 during Selective Laser Melting Using a Phase Field Model," The International Journal of Advanced Manufacturing Technology, **100**(9–12), pp. 2147–2162.

[71] Keller, T., Lindwall, G., Ghosh, S., Ma, L., Lane, B. M., Zhang, F., Kattner, U. R., Lass, E. A., Heigel, J. C., Idell, Y., Williams, M. E., Allen, A. J., Guyer, J. E., and Levine, L. E., 2017, "Application of Finite Element, Phase-Field, and CALPHAD-Based Methods to Additive Manufacturing of Ni-Based Superalloys," Acta Materialia, **139**, pp. 244–253.

[72] Liu, P., Wang, Z., Xiao, Y., Horstemeyer, M. F., Cui, X., and Chen, L., 2019, "Insight into the Mechanisms of Columnar to Equiaxed Grain Transition during Metallic Additive Manufacturing," Additive Manufacturing, **26**, pp. 22–29.

[73] Yang, Y., Ragnvaldsen, O., Bai, Y., Yi, M., and Xu, B. X., 2019, "3D Non-Isothermal Phase-Field Simulation of Microstructure Evolution during Selective Laser Sintering," npj Computational Materials, **5**(1), pp. 1–12.

[74] Acharya, R., Sharon, J. A., and Staroselsky, A., 2017, "Prediction of Microstructure in Laser Powder Bed Fusion Process," Acta Materialia, **124**, pp. 360–371.

[75] Medvedev, D., and Kassner, K., 2005, "Lattice Boltzmann Scheme for Crystal Growth in External Flows," Physical Review E, **72**(5), p. 056703.

[76] Miller, W., Rasin, I., and Succi, S., 2006, "Lattice Boltzmann Phase-Field Modelling of Binary-Alloy Solidification," Physica A: Statistical Mechanics and its Applications, **362**(1), pp. 78–83.

[77] Medvedev, D., Varnik, F., and Steinbach, I., 2013, "Simulating Mobile Dendrites in a Flow," Procedia Computer Science, **18**, pp. 2512–2520.

[78] Rojas, R., Takaki, T., and Ohno, M., 2015, "A Phase-Field-Lattice Boltzmann Method for Modeling Motion and Growth of a Dendrite for Binary Alloy Solidification in the Presence of Melt Convection," Journal of Computational Physics, **298**, pp. 29–40.

[79] Takaki, T., Rojas, R., Ohno, M., Shimokawabe, T., and Aoki, T., 2015, "GPU Phase-Field Lattice Boltzmann Simulations of Growth and Motion of a Binary Alloy Dendrite," IOP Conference Series: Materials Science and Engineering, **84**, p. 012066.

[80] Schmitz, G. J., Böttger, B., and Apel, M., 2016, "On the Role of Solidification Modelling in Integrated Computational Materials Engineering 'ICME,'" IOP Conference Series: Materials Science and Engineering, **117**(1), p. 012041.

[81] Li, Q., Luo, K. H., Kang, Q. J., He, Y. L., Chen, Q., and Liu, Q., 2016, "Lattice Boltzmann Methods for Multiphase Flow and Phase-Change Heat Transfer," Progress in Energy and Combustion Science, **52**, pp. 62–105.

[82] Chen, S., and Doolen, G. D., 1998, "Lattice Boltzmann Method for Fluid Flows," Annual Review of Fluid Mechanics, **30**(1), pp. 329–364.

[83] Aidun, C. K., Clausen, J. R., and Woodruff, G. W., 2010, "Lattice-Boltzmann Method for Complex Flows," Annual Review of Fluid Mechanics, **42**, pp. 439–72.

[84] He, X., Chen, S., and Doolen, G. D., 1998, "A Novel Thermal Model for the Lattice Boltzmann Method in Incompressible Limit," Journal of Computational Physics, **146**(1), pp. 282–300.

[85] Chakraborty, S., and Chatterjee, D., 2007, "An Enthalpy-Based Hybrid Lattice-Boltzmann Method for Modelling Solid–Liquid Phase Transition in the Presence of Convective Transport," Journal of Fluid Mechanics, **592**, pp. 155–175.

[86] Guo, Z., Zheng, C., Shi, B., and Zhao, T. S., 2007, "Thermal Lattice Boltzmann Equation for Low Mach Number Flows: Decoupling Model," Physical Review E, **75**(3), p. 036704.

[87] Attar, E., and Körner, C., 2011, "Lattice Boltzmann Model for Thermal Free Surface Flows with Liquid-Solid Phase Transition," International Journal of Heat and Fluid Flow, **32**(1), pp. 156–163.

[88] Eshraghi, M., and Felicelli, S. D., 2012, "An Implicit Lattice Boltzmann Model for Heat Conduction with Phase Change," International Journal of Heat and Mass Transfer, **55**(9–10), pp. 2420–2428.

[89] Seta, T., 2013, "Implicit Temperature-Correction-Based Immersed-Boundary Thermal Lattice Boltzmann Method for the Simulation of Natural Convection," Physical Review E - Statistical, Nonlinear, and Soft Matter Physics, **87**(6), pp. 1–16.

[90] Perumal, D. A., and Dass, A. K., 2015, "A Review on the Development of Lattice Boltzmann Computation of Macro Fluid Flows and Heat Transfer," Alexandria Engineering Journal, **54**(4), pp. 955–971.

[91] Ammer, R., Markl, M., Ljungblad, U., Körner, C., and Rüde, U., 2014, "Simulating Fast Electron Beam Melting with a Parallel Thermal Free Surface Lattice Boltzmann Method," Computers and Mathematics with Applications, **67**(2), pp. 318–330.

[92] Karpatne, A., Atluri, G., Faghmous, J. H., Steinbach, M., Banerjee, A., Ganguly, A., Shekhar, S., Samatova, N., and Kumar, V., 2017, "Theory-Guided Data Science: A New Paradigm for Scientific Discovery from Data," IEEE Transactions on Knowledge and Data Engineering, **29**(10), pp. 2318–2331.

[93] Willard, J., Jia, X., Xu, S., Steinbach, M., and Kumar, V., 2020, "Integrating Physics-Based Modeling with Machine Learning: A Survey."

[94] Rai, R., and Sahu, C. K., 2020, "Driven by Data or Derived Through Physics? A Review of Hybrid Physics Guided Machine Learning Techniques With Cyber-Physical System (CPS) Focus," IEEE Access, **8**, pp. 71050–71073.

[95] Li-Zhi, L., and Hou-Duo, Q., 1999, "A Neural Network for the Linear Complementarity Problem," Mathematical and computer modelling, **29**(3), pp. 9–18.

[96] Xia, Y., Leung, H., and Wang, J., 2002, "A Projection Neural Network and Its Application to Constrained Optimization Problems," IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications, **49**(4), pp. 447–458.

[97] Thompson, M. L., and Kramer, M. A., 1994, "Modeling Chemical Processes Using Prior Knowledge and Neural Networks," AIChE Journal, **40**(8), pp. 1328–1340.

[98] Watson, P. M., Gupta, K. C., and Mahajan, R. L., 1998, "Development of Knowledge Based Artificial Neural Network Models for Microwave Components," 1998 IEEE MTT-S International Microwave Symposium Digest (Cat. No.98CH36192), **1**, pp. 9–12.

[99] Wang, F., 1997, “Knowledge-Based Neural Models for Microwave Design,” IEEE Transactions on Microwave Theory and Techniques, **45**(12), pp. 2333–2343.

[100] Nagarajan, H. P. N., Mokhtarian, H., Jafarian, H., Dimassi, S., Bakrani-Balani, S., Hamedi, A., Coatanéa, E., Gary Wang, G., and Haapala, K. R., 2018, “Knowledge-Based Design of Artificial Neural Network Topology for Additive Manufacturing Process Modeling: A New Approach and Case Study for Fused Deposition Modeling,” Journal of Mechanical Design, **141**(2), p. 21705.

[101] Tresp, V;Hollatz, J;Ahmad, S., 1993, “Network Structuring and Training Using Rule-Based Knowledge,” *Advances in Neural Information Processing Systems*, pp. 871–878.

[102] Towell, G. G., and Shavlik, J. W., 1994, “Knowledge-Based Artificial Neural Networks,” Artificial Intelligence, **70**(1–2), pp. 119–165.

[103] Ramuhalli, P., Udpa, L., and Udpa, S. S., 2005, “Finite-Element Neural Networks for Solving Differential Equations,” **16**(6), pp. 1381–1392.

[104] Xu, C., Wang, C., Ji, F., and Yuan, X., 2012, “Finite-Element Neural Network-Based Solving 3-D Differential Equations in Mfl,” IEEE Transactions on Magnetics, **48**(12), pp. 4747–4756.

[105] Han, F., and Huang, D. S., 2008, “A New Constrained Learning Algorithm for Function Approximation by Encoding a Priori Information into Feedforward Neural Networks,” Neural Computing and Applications, **17**(5–6), pp. 433–439.

[106] Lauer, F., and Bloch, G., 2008, “Incorporating Prior Knowledge in Support Vector Regression,” Machine Learning, **70**(1), pp. 89–118.

[107] Dissanayake, M. W. M. G., and Phan-Thien, N., 1994, “Neural-Network-Based Approximations for Solving Partial Differential Equations,” Communications in Numerical Methods in Engineering, **10**(3), pp. 195–201.

[108] Lagaris, I. E., Likas, A., and Fotiadis, D. I., 1998, “Artificial Neural Networks for Solving Ordinary and Partial Differential Equations,” IEEE Transactions on Neural Networks, **9**(5), pp. 987–1000.

[109] Lagaris, I. E., Likas, A. C., and Papageorgiou, D. G., 2000, “Neural-Network

Methods for Boundary Value Problems with Irregular Boundaries," IEEE Transactions on Neural Networks, **11**(5), pp. 1041–1049.

[110] Shekari Beidokhti, R., and Malek, A., 2009, "Solving Initial-Boundary Value Problems for Systems of Partial Differential Equations Using Neural Networks and Optimization Techniques," Journal of the Franklin Institute, **346**(9), pp. 898–913.

[111] Raissi, M., Perdikaris, P., and Karniadakis, G. E., 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**, pp. 686–707.

[112] Souza De Cursi, J. E., and Koscianski, A., 2007, "Physically Constrained Neural Network Models for Simulation," Advances and Innovations in Systems, Computing Sciences and Software Engineering, pp. 567–572.

[113] Ferrari, S., and Jensenius, M., 2008, "A Constrained Optimization Approach to Preserving Prior Knowledge during Incremental Training," IEEE Transactions on Neural Networks, **19**(6), pp. 996–1009.

[114] Di Muro, G., and Ferrari, S., 2008, "A Constrained-Optimization Approach to Training Neural Networks for Smooth Function Approximation and System Identification," Proceedings of the International Joint Conference on Neural Networks, pp. 2353–2359.

[115] Rudd, K., Muro, G. D., and Ferrari, S., 2014, "A Constrained Backpropagation Approach for the Adaptive Solution of Partial Differential Equations," IEEE Transactions on Neural Networks and Learning Systems, **25**(3), pp. 571–584.

[116] Dissanayake, M. W. M. G., and Phan-Thien, N., 1994, "Neural-network-based Approximations for Solving Partial Differential Equations," Communications in Numerical Methods in Engineering, **10**(3), pp. 195–201.

[117] Mai-Duy, N., and Tran-Cong, T., 2001, "Numerical Solution of Navier-Stokes Equations Using Multiquadric Radial Basis Function Networks," International Journal for Numerical Methods in Fluids, **37**(1), pp. 65–86.

[118] Jianyu, L., Siwei, L., Yingjian, Q., and Yaping, H., 2003, "Numerical Solution of Elliptic Partial Differential Equation Using Radial Basis Function Neural Networks," Neural Networks, **16**(5–6), pp. 729–734.

[119] Raissi, M., Perdikaris, P., and Karniadakis, G. E., 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**, pp. 686–707.

[120] Zhu, Y., Zabaras, N., Koutsourelakis, P. S., and Perdikaris, P., 2019, "Physics-Constrained Deep Learning for High-Dimensional Surrogate Modeling and Uncertainty Quantification without Labeled Data," Journal of Computational Physics, **394**, pp. 56–81.

[121] Liu, D., and Wang, Y., 2019, "Multi-Fidelity Physics-Constrained Neural Network and Its Application in Materials Modeling," Journal of Mechanical Design, **141**(12), pp. 1–13.

[122] Kennedy, M., and O'Hagan, A., 2000, "Predicting the Output from a Complex Computer Code When Fast Approximations Are Available," Biometrika, **87**(1), pp. 1–13.

[123] Fernández-Godino, M. G., Park, C., Kim, N.-H., and Haftka, R. T., 2016, "Review of Multi-Fidelity Models," arXiv:1609.07196.

[124] Peherstorfer, B., Willcox, K., and Gunzburger, M., 2018, "Survey of Multifidelity Methods in Uncertainty Propagation, Inference, and Optimization," SIAM Review, **60**(3), pp. 550–591.

[125] Xiong, Y., Chen, W., and Tsui, K. L., 2008, "A New Variable-Fidelity Optimization Framework Based on Model Fusion and Objective-Oriented Sequential Sampling," Journal of Mechanical Design, **130**(11), pp. 1114011–1114019.

[126] Zhou, Q., Wang, Y., Choi, S. K., Jiang, P., Shao, X., and Hu, J., 2017, "A Sequential Multi-Fidelity Metamodeling Approach for Data Regression," Knowledge-Based Systems, **134**, pp. 199–212.

[127] Zhou, Q., Wang, Y., Choi, S. K., Jiang, P., Shao, X., Hu, J., and Shu, L., 2018, "A Robust Optimization Approach Based on Multi-Fidelity Metamodel," Structural and Multidisciplinary Optimization, **57**(2), pp. 775–797.

[128] Shi, R., Liu, L., Long, T., Wu, Y., and Gary Wang, G., 2020, "Multi-Fidelity Modeling and Adaptive Co-Kriging-Based Optimization for All-Electric Geostationary Orbit Satellite Systems," Journal of Mechanical Design, **142**(2), pp.

1–13.

[129] Meng, X., and Karniadakis, G. E., 2020, "A Composite Neural Network That Learns from Multi-Fidelity Data: Application to Function Approximation and Inverse PDE Problems," Journal of Computational Physics, **401**, p. 109020.

[130] Alhat, D., Lasrado, V., and Wang, Y., 2008, "A Review of Recent Phase Transition Simulation Methods: Saddle Point Search," *2008 ASME International Design Engineering Technical Conferences & The Computer and Information in Engineering Conference (IDETC/CIE2008)*, ASME, New York City, NY, pp. DETC2008-49411.

[131] Lasrado, V., Alhat, D., and Wang, Y., 2008, "A Review of Recent Phase Transition Simulation Methods: Transition Path Search," *2008 ASME International Design Engineering Technical Conferences & The Computer and Information in Engineering Conference (IDETC/CIE2008)*, ASME, New York City, NY, pp. DETC2008-49410.

[132] Simons, J., Jørgensen, P., Taylor, H., and Ozment, J., 1983, "Walking on Potential Energy Surfaces," Journal of Physical Chemistry, **87**(15), pp. 2745–2753.

[133] Dewar, M. J. S., Healy, E. F., and Stewart, J. J. P., 1984, "Location of Transition States in Reaction Mechanisms," Journal of the Chemical Society, Faraday Transactions 2: Molecular and Chemical Physics, **80**(3), pp. 227–233.

[134] Banerjee, A., Adams, N., Simons, J., and Shepard, R., 1985, "Search for Stationary Points on Surfaces," Journal of Physical Chemistry, **89**(1), pp. 52–57.

[135] Mousseau, N., and Barkema, G. T., 1998, "Traveling through Potential Energy Landscapes of Disordered Materials: The Activation-Relaxation Technique," Physical Review E, **57**(2), pp. 2419–2424.

[136] Henkelman, G., and Jónsson, H., 1999, "A Dimer Method for Finding Saddle Points on High Dimensional Potential Surfaces Using Only First Derivatives," Journal of Chemical Physics, **111**(15), pp. 7010–7022.

[137] Heyden, A., Bell, A. T., and Keil, F. J., 2005, "Efficient Methods for Finding Transition States in Chemical Reactions: Comparison of Improved Dimer Method and Partitioned Rational Function Optimization Method," Journal of Chemical Physics, **123**(22), p. 224101.

[138] Kästner, J., and Sherwood, P., 2008, "Superlinearly Converging Dimer Method for Transition State Search," Journal of Chemical Physics, **128**(1), p. 014106.

[139] Henkelman, G., and Jónsson, H., 2000, "Improved Tangent Estimate in the Nudged Elastic Band Method for Finding Minimum Energy Paths and Saddle Points," Journal of Chemical Physics, **113**(22), pp. 9978–9985.

[140] Henkelman, G., Uberuaga, B. P., and Jónsson, H., 2000, "Climbing Image Nudged Elastic Band Method for Finding Saddle Points and Minimum Energy Paths," Journal of Chemical Physics, **113**(22), pp. 9901–9904.

[141] He, L., and Wang, Y., 2013, "A Concurrent Search Algorithm for Multiple Phase Transition Pathways," *2013 ASME International Design Engineering Technical Conferences & The Computer and Information in Engineering Conference (IDETC2013)*, American Society of Mechanical Engineers, Portland, Oregon, pp. DETC2013-12362.

[142] He, L., and Wang, Y., 2015, "A Curve Swarm Algorithm for Global Search of State Transition Paths," *Proceedings of the 3rd World Congress on Integrated Computational Materials Engineering (ICME 2015)*, Springer Cham, pp. 139–146.

[143] He, L., and Wang, Y., 2015, "An Efficient Saddle Point Search Method Using Kriging Metamodels," *Proceedings of 2015 ASME International Design Engineering Technical Conferences & Computers and Information in Engineering Conference (IDETC/CIE2015)*, American Society of Mechanical Engineers Digital Collection, Boston, Massachusetts, pp. DETC2015-47386.

[144] Tran, A., He, L., and Wang, Y., 2018, "An Efficient First-Principles Saddle Point Searching Method Based on Distributed Kriging Metamodels," ASCE-ASME Journal of Risk and Uncertainty in Engineering Systems, Part B: Mechanical Engineering, **4**(1), pp. 1–8.

[145] Tran, A., Liu, D., He-Bitoun, L., and Wang, Y., 2020, "Data-Driven Acceleration of First-Principles Saddle Point and Local Minimum Search Based on Scalable Gaussian Processes," *Uncertainty Quantification in Multiscale Materials Modeling*, Elsevier, pp. 119–168.

[146] Leyton-Brown, K., and Shoham, Y., 2008, *Essentials of Game Theory: A Concise Multidisciplinary Introduction*.

[147] Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y., 2014, "Generative Adversarial Nets," *Advances in Neural Information Processing Systems*, pp. 2672–2680.

[148] Beyer, H. G., and Sendhoff, B., 2007, "Robust Optimization - A Comprehensive Survey," Computer Methods in Applied Mechanics and Engineering, **196**(33–34), pp. 3190–3218.

[149] Daskalakis, C., and Panageas, I., 2018, "The Limit Points of (Optimistic) Gradient Descent in Min-Max Optimization," *Advances in Neural Information Processing Systems*, pp. 9236–9246.

[150] Rafique, H., Liu, M., Lin, Q., and Yang, T., 2021, "Weakly-Convex–Concave Min–Max Optimization: Provable Algorithms and Applications in Machine Learning," Optimization Methods and Software, pp. 1–35.

[151] Nouiehed, M., Sanjabi, M., Huang, T., Lee, J. D., and Razaviyayn, M., 2019, "Solving a Class of Non-Convex Min-Max Games Using Iterative First Order Methods," *Advances in Neural Information Processing Systems*, pp. 14905–14916.

[152] Thekumparampil, K. K., Jain, P., Netrapalli, P., and Oh, S., 2019, "Efficient Algorithms for Smooth Minimax Optimization," *Advances in Neural Information Processing Systems*, pp. 12659–12670.

[153] Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., and Hochreiter, S., 2017, "GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium," *Advances in Neural Information Processing Systems*, pp. 6627–6638.

[154] Balduzzi, D., Racaniere, S., Martens, J., Foerster, J., Karl, T., and Graepel, T., 2018, "The Mechanics of N-Player Differentiable Games," *International Conference on Machine Learning*, pp. 619–635.

[155] Adolphs, L., Daneshmand, H., Lucchi, A., and Hofmann, T., 2019, "Local Saddle Point Optimization: A Curvature Exploitation Approach," *The 22nd International Conference on Artificial Intelligence and Statistics*, PMLR, pp. 486–495.

[156] Raissi, M., and Karniadakis, G. E., 2018, "Hidden Physics Models: Machine Learning of Nonlinear Partial Differential Equations," Journal of Computational Physics, **357**, pp. 125–141.

[157] Sun, L., Gao, H., Pan, S., and Wang, J. X., 2020, "Surrogate Modeling for Fluid Flows Based on Physics-Constrained Deep Learning without Simulation Data," Computer Methods in Applied Mechanics and Engineering, **361**, p. 112732.

[158] Jin, X., Cai, S., Li, H., and Karniadakis, G. E., 2021, "NSFnets (Navier-Stokes Flow Nets): Physics-Informed Neural Networks for the Incompressible Navier-Stokes Equations," Journal of Computational Physics, **426**, p. 109951.

[159] Cai, S., Wang, Z., Lu, L., Zaki, T. A., and Karniadakis, G. E., 2021, "DeepM&Mnet: Inferring the Electroconvection Multiphysics Fields Based on Operator Approximation by Neural Networks," Journal of Computational Physics, **436**, pp. 1–17.

[160] Torabi Rad, M., Viardin, A., Schmitz, G. J., and Apel, M., 2020, "Theory-Training Deep Neural Networks for an Alloy Solidification Benchmark Problem," Computational Materials Science, **180**, p. 109687.

[161] Wang, S., and Perdikaris, P., 2021, "Deep Learning of Free Boundary and Stefan Problems," Journal of Computational Physics, **428**, p. 109914.

[162] Zhu, Q., Liu, Z., and Yan, J., 2021, "Machine Learning for Metal Additive Manufacturing: Predicting Temperature and Melt Pool Fluid Dynamics Using Physics-Informed Neural Networks," Computational Mechanics, **67**(2), pp. 619–635.

[163] Cai, S., Wang, Z., Wang, S., Perdikaris, P., and Karniadakis, G., 2021, "Physics-Informed Neural Networks (PINNs) for Heat Transfer Problems," Journal of Heat Transfer.

[164] Hennigh, O., Subramaniam, A., Narasimhan, S., Tangsali, K., Nabian, M. A., Rietmann, M., del Aguila Ferrandis, J., Byeon, W., Fang, Z., and Choudhry, S., 2021, "NVIDIA SimNet^{TM}: AN Ai-Accelerated Multi-Physics Simulation Framework," *International Conference on Computational Science*, Springer, Cham, pp. 447–461.

[165] Liu, D., and Wang, Y., 2021, "A Dual-Dimer Method for Training Physics-Constrained Neural Networks with Minimax Architecture," Neural Networks, **136**, pp. 112–125.

[166] Wang, S., Wang, H., and Perdikaris, P., 2021, "On the Eigenvector Bias of Fourier

Feature Networks: From Regression to Solving Multi-Scale PDEs with Physics-Informed Neural Networks," Computer Methods in Applied Mechanics and Engineering, **384**, p. 113938.

[167] Niaki, S. A., Haghighat, E., Li, X., Campbell, T., and Vaziri, R., 2020, "Physics-Informed Neural Network for Modelling the Thermochemical Curing Process of Composite-Tool Systems During Manufacture," Computer Methods in Applied Mechanics and Engineering, **384**, p. 113959.

[168] Liu, D., and Wang, Y., 2017, "Mesoscale Multi-Physics Simulation of Solidification in Selective Laser Melting Process Using a Phase Field and Thermal Lattice Boltzmann Model," *2017 ASME International Design Engineering Technical Conferences & The Computer and Information in Engineering Conference (IDETC/CIE2017)*, ASME, Cleveland, Ohio, pp. DETC2017-67633.

[169] Liu, D., and Wang, Y., 2019, "Mesoscale Multi-Physics Simulation of Rapid Solidification of Ti-6Al-4V Alloy," Additive Manufacturing, **25**, pp. 551–562.

[170] Tran, A., Liu, D., Tran, H., and Wang, Y., 2019, "Quantifying Uncertainty in the Process-Structure Relationship for Al–Cu Solidification," Modelling and Simulation in Materials Science and Engineering, **27**(6), p. 064005.

[171] Liu, D., and Wang, Y., 2019, "Simulation of Nucleation and Grain Growth in Selective Laser Melting of Ti-6Al-4V Alloy," *2019 ASME International Design Engineering Technical Conferences & The Computer and Information in Engineering Conference (IDETC/CIE2019)*, ASME, Anaheim, California, pp. DETC2019-97684.

[172] Cao, L., Liu, D., Jiang, P., Shao, X., Zhou, Q., and Wang, Y., 2019, "Multi-Physics Simulation of Dendritic Growth in Magnetic Field Assisted Solidification," International Journal of Heat and Mass Transfer, **144**, p. 118673.

[173] Liu, D., and Wang, Y., 2020, "Multiphysics Simulation of Nucleation and Grain Growth in Selective Laser Melting of Alloys," Journal of Computing and Information Science in Engineering, **20**(5).

[174] Sestito, J. M., Liu, D., Lu, Y., Song, J.-H., Tran, A. V., Kempner, M. J., Harris, T. A. L., Anh, S.-H., and Wang, Y., 2021, "Multiscale Process Modeling of Shape Memory Alloy Fabrication with Directed Energy Deposition," *Manufacturing in the Era of 4th Industrial Revolution - Vol. 1. Recent Advances in Additive Manufacturing*, H. Bruck, Y. Chen, and S.K. Gupta, eds., World Scientific, pp. 41–

76.

[175] Aziz, M. J., 1982, “Model for Solute Redistribution during Rapid Solidification,” Journal of Applied Physics, **53**(2), pp. 1158–1168.

[176] Ahmad, N. A., Wheeler, A. A., Boettinger, W. J., and McFadden, G. B., 1998, “Solute Trapping and Solute Drag in a Phase-Field Model of Rapid Solidification,” Physical Review E, **58**(3), pp. 3436–3450.

[177] Semiatin, S. L., Ivanchenko, V. G., and Ivasishin, O. M., 2004, “Diffusion Models for Evaporation Losses during Electron-Beam Melting of Alpha/Beta-Titanium Alloys,” Metallurgical and Materials Transactions B, **35**(2), pp. 235–245.

[178] Kim, S. G., 2007, “A Phase-Field Model with Antitrapping Current for Multicomponent Alloys with Arbitrary Thermodynamic Properties,” Acta Materialia, **55**(13), pp. 4391–4399.

[179] Guo, Z., Zheng, C., and Shi, B., 2002, “Discrete Lattice Effects on the Forcing Term in the Lattice Boltzmann Method,” Physical Review E - Statistical, Nonlinear, and Soft Matter Physics, **65**(4), pp. 1–6.

[180] Ginzburg, I., 2005, “Generic Boundary Conditions for Lattice Boltzmann Models and Their Application to Advection and Anisotropic Dispersion Equations,” Advances in Water Resources, **28**(11), pp. 1196–1216.

[181] Zhang, T., Shi, B., Guo, Z., Chai, Z., and Lu, J., 2012, “General Bounce-Back Scheme for Concentration Boundary Condition in the Lattice-Boltzmann Method,” Physical Review E - Statistical, Nonlinear, and Soft Matter Physics, **85**(1), p. 016701.

[182] Chen, Q., Zhang, X., and Zhang, J., 2013, “Improved Treatments for General Boundary Conditions in the Lattice Boltzmann Method for Convection-Diffusion and Heat Transfer Processes,” Physical Review E - Statistical, Nonlinear, and Soft Matter Physics, **88**(3), p. 033304.

[183] 2019, “OpenPhase” [Online]. Available: http://www.openphase.de/.

[184] Nastac, L., 2012, “Solute Redistribution, Liquid/Solid Interface Instability, and Initial Transient Regions during the Undirectional Solidification of Ti-6-4 and Ti-

17 Alloys," *CFD Modeling and Simulation in Materials Processing*, Wiley-TMS, New York, pp. 123–130.

[185] Helmer, H., Bauereiß, A., Singer, R. F., and Körner, C., 2016, "Grain Structure Evolution in Inconel 718 during Selective Electron Beam Melting," Materials Science and Engineering: A, **668**, pp. 180–187.

[186] Loginova, I., Amberg, G., and Ågren, J., 2001, "Phase-Field Simulations of Non-Isothermal Binary Alloy Solidification," Acta Materialia, **49**(4), pp. 573–581.

[187] Simonelli, M., Tse, Y. Y., and Tuck, C., 2014, "On the Texture Formation of Selective Laser Melted Ti-6Al-4V," Metallurgical and Materials Transactions A: Physical Metallurgy and Materials Science, **45**(6), pp. 2863–2872.

[188] Kou, S., 2003, *Welding Metallurgy*, John Wiley & Sons, Hoboken, NJ.

[189] Arnberg, L., and Mathiesen, R. H., 2007, "The Real-Time, High-Resolution x-Ray Video Microscopy of Solidification in Aluminum Alloys," JOM, **59**(8), pp. 20–26.

[190] Wei, S. L., Huang, L. J., Chang, J., Yang, S. J., Ma, Y. T., and Geng, L., 2016, "Containerless Rapid Solidification of Liquid Ti-Al-V Alloys Inside Drop Tube," Proceedings of the 13th World Conference on Titanium, pp. 397–404.

[191] Bouchard, D., and Kirkaldy, J. S., 1996, "Equations and Specification of Predictive Procedures," Metallurgical and Materials Transactions B, **28**(4), pp. 651–663.

[192] Van Cauwenbergh, P., Samaee, V., Thijs, L., Nejezchlebová, J., Sedlák, P., Iveković, A., Schryvers, D., Van Hooreweder, B., and Vanmeensel, K., 2021, "Unravelling the Multi-Scale Structure–Property Relationship of Laser Powder Bed Fusion Processed and Heat-Treated AlSi10Mg," Scientific Reports, **11**(1), pp. 1–15.

[193] Tao, P., Li, H., Huang, B., Hu, Q., Gong, S., and Xu, Q., 2019, "The Crystal Growth, Intercellular Spacing and Microsegregation of Selective Laser Melted Inconel 718 Superalloy," Vacuum, **159**, pp. 382–390.

[194] Ho, A., Zhao, H., Fellowes, J. W., Martina, F., Davis, A. E., and Prangnell, P. B., 2019, "On the Origin of Microstructural Banding in Ti-6Al4V Wire-Arc Based High Deposition Rate Additive Manufacturing," Acta Materialia, **166**, pp. 306–323.

[195] Neikter, M., Huang, A., and Wu, X., 2019, "Microstructural Characterization of Binary Microstructure Pattern in Selective Laser-Melted Ti-6Al-4V," International Journal of Advanced Manufacturing Technology, **104**(1–4), pp. 1381–1391.

[196] Bermingham, M. J., McDonald, S. D., Dargusch, M. S., and St.John, D. H., 2008, "Grain-Refinement Mechanisms in Titanium Alloys," Journal of Materials Research, **23**(1), pp. 97–104.

[197] Vrancken, B., Thijs, L., Kruth, J. P., and Van Humbeeck, J., 2014, "Microstructure and Mechanical Properties of a Novel β Titanium Metallic Composite by Selective Laser Melting," Acta Materialia, **68**, pp. 150–158.

[198] Simonelli, M., McCartney, D. G., Barriobero-Vila, P., Aboulkhair, N. T., Tse, Y. Y., Clare, A., and Hague, R., 2020, "The Influence of Iron in Minimizing the Microstructural Anisotropy of Ti-6Al-4V Produced by Laser Powder-Bed Fusion," Metallurgical and Materials Transactions A: Physical Metallurgy and Materials Science, **51**(5), pp. 2444–2459.

[199] Holdych, D. J., Noble, D. R., Georgiadis, J. G., and Buckius, R. O., 2004, "Truncation Error Analysis of Lattice Boltzmann Methods," Journal of Computational Physics, **193**(2), pp. 595–619.

[200] Majumder, S., and Liu, A. P., 2018, "Bottom-up Synthetic Biology: Modular Design for Making Artificial Platelets," Physical Biology, **15**(1), pp. 0–15.

[201] Thijs, L., Kempen, K., Kruth, J. P., and Van Humbeeck, J., 2013, "Fine-Structured Aluminium Products with Controllable Texture by Selective Laser Melting of Pre-Alloyed AlSi10Mg Powder," Acta Materialia, **61**(5), pp. 1809–1819.

[202] Pei, W., Zhengying, W., Zhen, C., Junfeng, L., Shuzhe, Z., and Jun, D., 2017, "Numerical Simulation and Parametric Analysis of Selective Laser Melting Process of AlSi10Mg Powder," Applied Physics A: Materials Science and Processing, **123**(8), pp. 1–15.

[203] Tang, M., Pistorius, P. C., and Beuth, J. L., 2017, "Prediction of Lack-of-Fusion Porosity for Powder Bed Fusion," Additive Manufacturing, **14**, pp. 39–48.

[204] Marola, S., Manfredi, D., Fiore, G., Poletti, M. G., Lombardi, M., Fino, P., and Battezzati, L., 2018, "A Comparison of Selective Laser Melting with Bulk Rapid Solidification of AlSi10Mg Alloy," Journal of Alloys and Compounds, **742**, pp.

271–279.

[205] Mukherjee, T., Wei, H. L., De, A., and DebRoy, T., 2018, "Heat and Fluid Flow in Additive Manufacturing – Part II: Powder Bed Fusion of Stainless Steel, and Titanium, Nickel and Aluminum Base Alloys," Computational Materials Science, **150**, pp. 369–380.

[206] Napolitano, R. E., Liu, S., and Trivedi, R., 2002, "Experimental Measurement of Anisotropy in Crystal-Melt Interfacial Energy," Interface Science, **10**(2–3), pp. 217–232.

[207] Steinbach, I., 2008, "Effect of Interface Anisotropy on Spacing Selection in Constrained Dendrite Growth," Acta Materialia, **56**(18), pp. 4965–4971.

[208] Hoyt, J. J., Asta, M., and Karma, A., 2003, "Atomistic and Continuum Modeling of Dendritic Solidification," Materials Science and Engineering R: Reports, **41**(6), pp. 121–164.

[209] Du, Y., Chang, Y. A., Huang, B., Gong, W., Jin, Z., Xu, H., Yuan, Z., Liu, Y., He, Y., and Xie, F. Y., 2003, "Diffusion Coefficients of Some Solutes in Fcc and Liquid Al: Critical Evaluation and Correlation," Materials Science and Engineering A, **363**(1–2), pp. 140–151.

[210] Koepf, J. A., Gotterbarm, M. R., Markl, M., and Körner, C., 2018, "3D Multi-Layer Grain Structure Simulation of Powder Bed Fusion Additive Manufacturing," Acta Materialia, **152**, pp. 119–126.

[211] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S., and Corrado, A. Davis, J. Dean, M. Devin, et al., 2016, "TensorFlow: A System for Large-Scale Machine Learning.," 12th USENIX Symposium on Operating Systems Design and Implementation.

[212] Güneş Baydin, A., Pearlmutter, B. A., Andreyevich Radul, A., and Mark Siskind, J., 2018, "Automatic Differentiation in Machine Learning: A Survey," Journal of Machine Learning Research, **18**, pp. 1–43.

[213] Kingma, D. P., and Ba, J. L., 2014, "Adam: A Method for Stochastic Optimization," *ArXiv Preprint ArXiv:1412.6980*.

[214] Tran, A., Liu, D., Tran, H., and Wang, Y., 2019, "Quantifying Uncertainty in the Process-Structure Relationship for Al-Cu Solidification," Modelling and Simulation in Materials Science and Engineering, **27**(6), p. 064005.

[215] Wang, X., Liu, Y., Sun, W., Song, X., and Zhang, J., 2018, "Multidisciplinary and Multifidelity Design Optimization of Electric Vehicle Battery Thermal Management System," Journal of Mechanical Design, Transactions of the ASME, **140**(9), p. 094501.

[216] Mescheder, L., Nowozin, S., and Geiger, A., 2017, "The Numerics of GANs," Advances in Neural Information Processing Systems, pp. 1826–1836.

[217] Nagarajan, V., and Kolter, J. Z., 2017, "Gradient Descent GAN Optimization Is Locally Stable," Advances in Neural Information Processing Systems, pp. 5586–5596.

[218] Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Köpf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S., 2019, "PyTorch: An Imperative Style, High-Performance Deep Learning Library," *Advances in Neural Information Processing Systems*, pp. 8024–8035.

[219] Donoho, D. L., 2006, "Compressed Sensing," IEEE Transactions on Information Theory, **52**(4), pp. 1289–1306.

[220] Beck, A., and Teboulle, M., 2009, "A Fast Iterative Shrinkage-Thresholding Algorithm for Linear Inverse Problems," SIAM Journal on Imaging Sciences, **2**(1), pp. 183–202.

[221] Wang, Y., 2011, "Multiscale Uncertainty Quantification Based on a Generalized Hidden Markov Model," Journal of Mechanical Design, **133**(3).

[222] Wang, Y., 2013, "Reliable Kinetic Monte Carlo Simulation Based on Random Set Sampling," Soft Computing 2013 17:8, **17**(8), pp. 1439–1451.

[223] Tallman, A. E., Blumer, J. D., Wang, Y., and McDowell, D. L., 2014, "Multiscale Model Validation Based on Generalized Interval Bayes' Rule and Its Application in Molecular Dynamics Simulation," *Proceedings of 2014 ASME International Design Engineering Technical Conferences & Computers and Information in Engineering Conference (IDETC/CIE2014)*, American Society of Mechanical Engineers Digital

Collection, Buffalo, New York, pp. DETC2014-35126.

[224] Tran, A. V., and Wang, Y., 2017, "Reliable Molecular Dynamics: Uncertainty Quantification Using Interval Analysis in Molecular Dynamics Simulation," Computational Materials Science, **127**, pp. 141–160.

[225] Tallman, A. E., Swiler, L. P., Wang, Y., and McDowell, D. L., 2017, "Reconciled Top-Down and Bottom-Up Hierarchical Multiscale Calibration of BCC Fe Crystal Plasticity," International Journal for Multiscale Computational Engineering, **15**(6), pp. 505–523.

[226] Wang, Y., 2015, "Uncertainty in Materials Modeling, Simulation, and Development for ICME," *Proceedings of 2015 Materials Science & Technology*, Columbus, Ohio, pp. 1295–1305.

[227] Wang, Y., and McDowell, D. L., 2020, "Uncertainty Quantification in Materials Modeling," Uncertainty Quantification in Multiscale Materials Modeling, pp. 1–40.

[228] Abdar, M., Pourpanah, F., Hussain, S., Rezazadegan, D., Liu, L., Ghavamzadeh, M., Fieguth, P., Cao, X., Khosravi, A., Acharya, U. R., Makarenkov, V., and Nahavandi, S., 2021, "A Review of Uncertainty Quantification in Deep Learning: Techniques, Applications and Challenges," Information Fusion, **76**, pp. 243–297.

[229] Shahriari, B., Swersky, K., Wang, Z., Adams, R. P., and De Freitas, N., 2016, "Taking the Human out of the Loop: A Review of Bayesian Optimization," Proceedings of the IEEE, **104**(1), pp. 148–175.

[230] Tran, A., Eldred, M., McCann, S., and Wang, Y., 2020, "SrMO-BO-3GP: A Sequential Regularized Multi-Objective Constrained Bayesian Optimization for Design Applications," *Proceedings of 2020 ASME International Design Engineering Technical Conferences & Computers and Information in Engineering Conference (IDETC/CIE2020)*, American Society of Mechanical Engineers (ASME), pp. DETC2020-22184.

[231] Shu, L., Jiang, P., Shao, X., and Wang, Y., 2020, "A New Multi-Objective Bayesian Optimization Formulation with the Acquisition Function for Convergence and Diversity," Journal of Mechanical Design, **142**(9), pp. 1–10.

[232] Shu, L., Jiang, P., and Wang, Y., 2021, "A Multi-Fidelity Bayesian Optimization Approach Based on the Expected Further Improvement," Structural and

Multidisciplinary Optimization, **63**(4), pp. 1709–1719.

[233] Tran, A., Sun, J., Furlan, J. M., Pagalthivarthi, K. V., Visintainer, R. J., and Wang, Y., 2019, "PBO-2GP-3B: A Batch Parallel Known/Unknown Constrained Bayesian Optimization with Feasibility Classification and Its Applications in Computational Fluid Dynamics," Computer Methods in Applied Mechanics and Engineering, **347**, pp. 827–852.

[234] Tran, A., Tran, M., and Wang, Y., 2019, "Constrained Mixed-Integer Gaussian Mixture Bayesian Optimization and Its Applications in Designing Fractal and Auxetic Metamaterials," Structural and Multidisciplinary Optimization, **59**(6), pp. 2131–2154.

[235] Wang, Y., 2021, "Design of Trustworthy Cyber–Physical–Social Systems With Discrete Bayesian Optimization," Journal of Mechanical Design, **143**(7).

[236] Jiménez, J., and Ginebra, J., 2017, "PyGPGO: Bayesian Optimization for Python," The Journal of Open Source Software, **2**(19), p. 431.

[237] 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), pp. 2002–2041.

[238] Kharazmi, E., Zhang, Z., and Karniadakis, G. E. M., 2021, "Hp-VPINNs: Variational Physics-Informed Neural Networks with Domain Decomposition," Computer Methods in Applied Mechanics and Engineering, **374**, p. 113547.

[239] Gao, H., Sun, L., and Wang, J. X., 2021, "PhyGeoNet: Physics-Informed Geometry-Adaptive Convolutional Neural Networks for Solving Parameterized Steady-State PDEs on Irregular Domain," Journal of Computational Physics, **428**, p. 110079.

[240] Lu, Y., Shevtshenko, E., and Wang, Y., 2021, "Physics-Based Compressive Sensing to Enable Digital Twins of Additive Manufacturing Processes," Journal of Computing and Information Science in Engineering, **21**(3).