QUANTIFYING IMPLICIT AND EXPLICIT CONSTRAINTS ON PHYSICS-INFORMED NEURAL PROCESSES

by

Haoyang Zheng

A Dissertation

Submitted to the Faculty of Purdue University

In Partial Fulfillment of the Requirements for the degree of

Master of Science

School of Mechanical Engineering

West Lafayette, Indiana

May 2021

THE PURDUE UNIVERSITY GRADUATE SCHOOL
STATEMENT OF COMMITTEE APPROVAL

Dr. Guang Lin, Chair

School of Mechanical Engineering / Departments of Mathematics

Dr. Wen-Wen Tung

Department of Earth, Atmospheric, and Planetary Sciences

Dr. Ming Qu

School of Civil Engineering

Approved by:

Dr. Nicole Key

This is dedicated to my mentors, my friends and my parents.

ACKNOWLEDGMENTS

This work would not be possible without the tremendous support and helpful guidance from my advisor Professor Guang Lin. I would like to thank Professor Lin for giving me the opportunity to join his lab, and his support and encouragement on my research. I would also like to express my gratitude to my committee members, Professor Tung and Professor Qu for their help and suggestions, which help me to improve this work. I would also like to thank my fellow students and friends, Ziyang Huang, and Yixuan Sun for their contribution to this research. I am grateful to all of those whom I have had pleasure to work with during my master's program. This work would not be possible if I had not had their support and inspiration.

I want to thank my parents for the emotional and financial support. Their generous encouragement inspires me, which allows me to work toward my master's degree. My friends Zichang He, Yixin Xu, Tian Bian, Kun Tang, Mingyue Jin, and my undergraduate advisor Professor Yong Deng, have been such an important part of the pursuit of this research. I would like to thank them for sharing their experiences, and their support and caring.

PREFACE

This research is the master's thesis as a conclusion of my master's program at School of Mechanical Engineering, Purdue University. The basis of this research stemmed from my interest in the applications of deep learning techniques in real-world engineering problems. With the suggestion from my advisor, Professor Guang Lin, and my learning of flow dynamics, the idea of applying neural network for regression to simulate the dynamics of multiphase flows came to my mind.

Although researches on multiphase flows benefit a wide-range of fields, the mechanism is so complex that it is still difficult to build a complete mathematical model to describe different kinds of multiphase flows, as well as to solve those models efficiently and accurately. In this research, a predictive model based on deep learning has been developed to help on the multiphase flows simulation. With a consideration of implicit and explicit constraints, the model can simulate multiphase flows governed by physical laws.

TABLE OF CONTENTS

LIST OF TABLES 8

LIST OF FIGURES 9

LIST OF SYMBOLS 11

ABBREVIATIONS 13

NOMENCLATURE 14

GLOSSARY 15

ABSTRACT 16

1 INTRODUCTION 17

1.1 Our Contribution: Novelties and Outline 19

2 PROBLEM DEFINITION 22

2.1 Multiphase flows 22

2.2 Time series prediction 23

3 THE PROPOSED PHYSICS-INFORMED NEURAL PROCESSES 26

3.1 The neural processes to predict the order parameters 26

3.2 The multiphase consistent and conservative boundedness mapping algorithm . 30

3.3 Loss function with physics-informed constraints 31

4 EXPERIMENTS 36

4.1 Horizontal shear layer simulation 36

4.2 Dam break simulation 42

5 CONCLUSION 46

REFERENCES 47

LIST OF TABLES

3.1 An algorithm to update strategy using Adam optimization. This algorithm first update parameter space $\boldsymbol{\theta}_1$ with Adam optimizer. Then the MCBOM algorithm is applied to obtain the modified order parameters. The modified current order parameters and flow velocities in the last time step are applied to update parameter space $\boldsymbol{\theta}_2$ with the use of Adam optimizer and implicit physical constraints in the loss function. 35

4.1 A general summary of data set. The number of phases, number of available time steps, total simulation time, and their grid sizes are provided in this table. 36

LIST OF FIGURES

3.1 A general framework of CNP-LSTM to predict order parameters. The order parameters (ϕ_{i}, i = 1, 2, $\cdots$, m) are processed by LSTM cells and get r_{i}. Then a commutative operation transform a combination of r_1, r_2, $\cdots$, r_1 to $R_{\mathrm{i}} \in \mathcal{R}$. Then R_{i} and ϕ_{j} will be used to predict $\phi_{\mathrm{j+1}}$ (j = m, $m+1$, $\cdots$, $n-1$). The red diagram in the middle (LSTM cell) represents mathematical operations in a LSTM cell. 26

3.2 A general framework of PINP model. The observations $\{\phi_1, \phi_2, \cdots, \phi_m\}$ are encoded in the encoder in CNP-LSTM to update R_{i} and then the CNP-LSTM approximates a prediction of $\phi'_{\mathrm{i+1}}$ at first. This prediction will be modified by MCBOM algorithm and get $\phi_{\mathrm{i+1}}$. The modified $\phi_{\mathrm{i+1}}$ will constrain the update of $v_{\mathrm{i+1}}$ as a form of regularization term in the loss function of the second neural network. $v_{\mathrm{i+1}}$ will be predicted by the PICNP-LSTM. 34

4.1 Phase values indication of the three-phase shear layer simulation. For example, in the top left figure, the left blue one, middle red one, and the right blue one represent three different phase flow separately. The red one represents one of the phases (its phase is close to 1), and the remaining blue ones are the rest of other two phases (their values are close to -1). Therefore, three side-by-side pictures indicate the position information of multiphase flows at a certain time. From top to bottom, from left to right, the results show the change of shear layer over time t from 0.8s to 2.0s with 0.3s increment. 37

4.2 a) Validation of mass conservation: the sum of order parameters of a specific phase should be consistent over time ($\sum_{n_C} [\phi_p]_{n_C}$). p is the indices of phase and we show results of the first phase here; b) Validation of phase range constraints: all phase values should fall between [-1, 1] ($\max|\phi_p| \leq 1$); c) Number of values exceed the physical boundary with respect to time t (number of $\max|\phi_p| > 1$); d) Validation of summation for the order parameters: sum of order parameters is restricted to -1 ($\max|\sum_{p=1}^{N} \phi_p - (2-N)|$). 39

4.3 (a) Comparison of loss values over time between PINPs and traditional model; (b) Comparison of loss values over epochs between PINPs and traditional model. 40

4.4 (a) Validation of momentum conservation in x direction (in the use of PICNP-LSTM): $\sum_{n_C} [\rho u]_{n_C}$ vs. t; (b) Validation of momentum conservation in y direction: $\sum_{n_C} [\rho v \Delta\Omega]_{n_C}$ vs. t. 41

4.5 Phase values indication of the water, oil and air three-phase flows during the dam break procedure. From left to right, the read ones (phase values close to 1) in figures show the interface of water, oil and air. The blue one (values close to -1) indicates two other phases. From top to bottom, t is from 10s to 20s with 2s increment. 43

4.6 a) Validation of mass conservation: the sum of order paramaters of a specific phase should be consistent over time ($\sum_{n_C} [\phi_p]_{n_C}$). p is the indices of phase and we show results of the first phase here; b) Validation of phase range constraints: all phase values should fall between [-1, 1] ($\max|\phi_p| \leq 1$); c) Number of values exceed the physical boundary with respect to time t (number of $\max|\phi_p| > 1$); d) Validation of summation for the order parameters: sum of order parameters is restricted to -1 ($\max|\sum_{p=1}^{N} \phi_p - (2 - N)|$). 44

4.7 Validation of momentum conservation in x direction: $\sum_{n_C} [\rho u]_{n_C}$ vs. t. 45

LIST OF SYMBOLS

N	number of phases
ρ	flow density
μ	phase viscosity
ϕ	order parameter of phase
p	phase index
Ω	domain of interest
$\boldsymbol{n}$	outward normal at the domain of interest
x_{i}	observed sequence at time i
x_{j}	unobserved sequence at time j
$\boldsymbol{x}$	observed sequence set
$\boldsymbol{y}$	unobserved sequence set
$\boldsymbol{\varepsilon}$	bias vector
ε_{i}	bias at time i
σ_{i}^2	variance of ε_{i}
$\boldsymbol{\sigma}^2$	covariance matrix of $\boldsymbol{x}$
$\boldsymbol{\theta}$	parameter space of the model
$R_{\mathcal{L}}(\cdot)$	risk function
$\mathcal{L}(\cdot,\cdot)$	loss function
$\widehat{\boldsymbol{\theta}}$	estimated parameter space
$\widehat{\boldsymbol{\sigma}}^2$	estimated covariance matrix
$\widehat{f}(\cdot)$	estimated mapping for $\boldsymbol{x}$
$\boldsymbol{\phi}$	order parameter set
$\boldsymbol{v}$	flow velocity set
$S(\cdot)$	sigmoid function
b_f	bias of forget gate
U_f	input weight of forget gate

W_f	cycle weight of forget gate
b_{i}	bias of input gate
U_{i}	input weight of input gate
W_{i}	cycle weight of input gate
$\tanh(\cdot)$	hypertangent activation function
$b_{\widetilde{C}}$	bias of candidate value
$U_{\widetilde{C}}$	input weight of candidate value
$W_{\widetilde{C}}$	cycle weight of candidate value
b_o	bias of output gate
U_o	input weight of output gate
W_o	cycle weight of output gate
$H_{\mathrm{i}}(\cdot,\cdot,\cdot)$	affine transformation
Q_θ	conditional stochastic processes
$W^{p,q}$	weight function
λ	Lagrange multiplier
$\mathcal{M}_{\mathrm{j}}$	total momentum at time j

ABBREVIATIONS

MPF	multiphase flow
PINN	physics-informed neural network
OP	order parameter
CNP	conditional neural process
LSTM	long short-term memory
CNP-LSTM	conditional neural process and long short-term memory
PICNP-LSTM	physics-informed conditional neural processes and long short-term memory
MCBOM	multiphase consistent and conservative boundedness mapping
PINP	physics-informed neural process
HSL	horizontal shear layer
DB	dam break
NN	neural network
PIC	physics-informed constraint
PDE	partial differential equation
RNN	recurrent neural network
GP	Gaussian process

NOMENCLATURE

sigmoid	sigmoid function
tanh	hyperbolic tangent function
ReLU	rectified linear unit
Adam	adaptive moment estimation

GLOSSARY

affine transformation	a transformation to change distances and angles
gate	a point-wise function to regulate the flow of information
cell	a unit in LSTM to remember values over arbitrary time intervals

ABSTRACT

Due to strong interactions among various phases and among the phases and fluid motions, multiphase flows (MPFs) are so complex that lots of efforts have to be paid to predict its sequential patterns of phases and motions. The present paper applies the physical constraints inherent in MPFs and enforces them to a physics-informed neural network (PINN) model either explicitly or implicitly, depending on the type of constraints. To predict the unobserved order parameters (OPs) (which locate the phases) in the future steps, the conditional neural processes (CNPs) with long short-term memory (LSTM, combined as CNP-LSTM) are applied to quickly infer the dynamics of the phases after encoding only a few observations. After that, the multiphase consistent and conservative boundedness mapping (MCBOM) algorithm is implemented the correction the predicted OPs from CNP-LSTM so that the mass conservation, the summation of the volume fractions of the phases being unity, the consistency of reduction, and the boundedness of the OPs are strictly satisfied. Next, the density of the fluid mixture is computed from the corrected OPs. The observed velocity and density of the fluid mixture then encode in a physics-informed conditional neural processes and long short-term memory (PICNP-LSTM) where the constraint of momentum conservation is included in the loss function. Finally, the unobserved velocity in future steps is predicted from PICNP-LSTM. The proposed physics-informed neural processes (PINPs) model (CNP-LSTM-MCBOM-PICNP-LSTM) for MPFs avoids unphysical behaviors of the OPs, accelerates the convergence, and requires fewer data. The proposed model successfully predicts several canonical MPF problems, i.e., the horizontal shear layer (HSL) and dam break (DB) problems, and its performances are validated.

1. INTRODUCTION

Multiphase flows (MPFs) include the motions of various phases, each of which forms moving and deforming interfaces. The motion of each phase and the interactions among them are all non-linear and complex. Although it is possible to predict MPFs by collecting their historical information, such as the flow rate, temperature, pressure, and position of the phases, the mechanism is so complex that it is still difficult to build a complete mathematical model to describe different kinds of MPFs, as well as to solve those models efficiently and accurately. Researches on MPFs benefit a wide-range of fields, e.g., the chemical industry [1], [2], environmental protection [3], life sciences [4]. Therefore, the simulation of MPFs becomes a popular topic and many researchers have focused on it.

In recent years, the development of neural network (NN) technology promotes time series modeling, which also contributes to predicting MPFs. With the use of self-organizing NN, Mi *et al.* [5] simulated the vertical bubbly, slug, churn, and annular flows for the impedance of the multiphase admixture, and the flow patterns were successfully predicted. Based on the given experimental and theoretical velocity data, Valero and Bung [6] identified the characteristic shapes of the fluid phases and explored the behaviors of the air-water flow. Rashid *et al.* [7] developed a radical basis function NN to predict MPFs under critical conditions and discovered negative relationships between the upstream pressure and gas-liquid ratio, and between the choke bin size and liquid flow rate. Serra *et al.* [8] proposed a randomized hough transform with a NN to predict the void fraction measurement given image samples. Their experiments indicated a high consistency with increasing the void fraction of the proposed model. More works on MPFs simulation with the help of NNs was summarized in Yan′s review [9].

To simulate MPFs, the parameters inside the NN will be updated until the similar and even the same flow characteristics can be predicted by the network. Some popular optimization algorithms, such as stochastic gradient descent [10], [11], RMSprop [12], and adaptive moment estimation [13], will be applied to update the parameters. However, when applying those algorithms to find a suitable parameter space that can predict dynamics of MPFs, previous

works did not consider the physical laws that should be strictly satisfied by the prediction. As a result, the NN can easily produce unphysical results, especially when the number of samples is insufficient to train a complex NN with a large parameter space. Thus, recent studies are considering including PIC to improve the performance of NNs so that not only available data are used more effectively but also the burden of data acquisition is alleviated. The physics-informed constraints (PICs) can be enforced either implicitly and explicitly. For implicit constraints, physics-informed penalties are added in the loss function [14]. For explicit constraints, the prediction from the NN is modified to satisfy the physical laws [15].

Physics-informed neural networks (PINNs) are universal function approximators that aim to enforce underlying PIC governed by data sets [14] in the loss function. The regularization enforced by PICs can reduce the hypothesis space of parameters and thus can learn the model with fewer observations. Moreover, a NN model with desired physical properties is more acceptable for epxerts in this area when it applies to physical science research because it can be more generalizable to out-of-sample scenarios [16]. Pang *et al.* [17] developed fractional physics-informed neural networks (fPINNs) to encode fractional-order partial differential equations (PDEs), which can be specified as time, space and space-time fractional advection-diffusion equations. Yang and Perdikaris [18] proposed probabilistic PINNs to approximate arbitrary conditional probability densities, which is capable of generating samples similar to the one given by PDEs. Goswami et al. [19] proposed a PINN algorithm to solve brittle fracture problems and optimize its loss by decreasing the variational energy of the system. The above and other related works [20], [21] indicate the performance of modeling data sets governed by physical information. In the present study, we also consider physical insights from the scientific system [14], [22] and characterize loss function with PIC. Besides, many works indicated that a NN with PICs contributes to improving prediction accuracy [23]–[25], discovering PDEs and governing equations [22], [26], modeling inverse problems [27], [28], and solving uncertainty quantification [18], [29]. Some works also focus on considering general deep learning framework to learn diverse continuous nonlinear operators [30]–[32]. Therefore,

adding physics-informed penalization in loss function becomes more and more acceptable and interpretable when considering the research of both NNs and physical science.

1.1 Our Contribution: Novelties and Outline

We consider the simulation of MPFs, where the location of each phase is indicated by a set of order parameters (OP) related to the volume fractions of the phases. For MPFs, there are several PIC that the OPs should follow [15]. The first one is called the summation constraint. The OPs are not independent of each other because the summation of the volume fractions needs to be unity [33]. The second one is called the conservation constraint. The mass (or volume) of each phase is conserved without considering any sources of the phases either in or at the boundary of the domain [15]. The third one is called the consistency of reduction [34]. If a phase is absent at the beginning, it should remain absent and behaviors of the other phases should not influenced by the absent phase. The last one is called the boundedness constraint. The OPs should produce the volume fractions in between zero and one. When applying NNs to simulation MPFs, they usually do not commit physical law, which results in unphysical solutions. Violating those constraints results in the presence of fictitious phases, local voids or overfilling, mass loss, or negative density or viscosity. The inconsistent errors occurred from unphysical solutions are positively related to the density contrast of the fluids. The physical behaviors of the results will be propagated into the training process, and gradually accumulated, leading to large errors in the prediction, as simulation time goes on. Considering the consistent and conservative form of the inertial term, a scheme for flow motion regulation is obtained. This is a practical tool to correct potential errors in NN predictions. Therefore, we encode these PICs and modify the results after each model training to enforce the predictions exactly and explicitly.

For the NN structure, we consider the conditional neural processes and long short-term memory (CNP-LSTM), to improve the estimation of MPFs. Long short-term memory (LSTM) [35] is a specific form of recurrent neural networks (RNNs) that aims to solve exploding or vanishing gradient problems in the process of model training [36], [37]. It is able

to memorize previous information selectively via changing the status of the forget gates, to apply the memorized information to the output calculation of the current neuron through memory cells, and to continuously update with the latest inputs. As one of the most famous modified RNN model, LSTM has been widely known in solving natural language processing [38]–[40], weather prediction [41], and human activity recognition [42], etc. A large scale study on variants of the LSTM architecture was presented by Greff *et al.* [40] and the conclusion that vanilla LSTM [43] performs well on various data was finally drwan. Cornia *et al.* [44] designed attentive convolutional LSTM to improve saliency predictions successively and applied in predicting human eye fixations on natural images. Al-Shedivat *et al.* [45] proposed Gaussian processes and long short-term memory (GP-LSTM) model that fully encapsulates the inductive biases of recurrent networks while keeping possession of the non-parametric probabilistic properties of Gaussian processes (GPs). The GP-LSTM model indeed performs well in time series prediction, yet the computational efficiency of the model is limited by the highly intensive kernel functions, thus CNPs are applied here to improve the predictive performance of traditional GPs. The conditional neural processes (CNPs) perform well when dealing with independent and identically distributed random variables [46]. Also, it avoids training from scratch by extracting prior knowledge and thus can make predictions after observing only a small number of datasets. It is possible to predict MPFs with a small number of samples thanks to CNPs.

The OPs are first predicted from the CNP-LSTM model. The time-dependent dynamics of MPFs are encoded by LSTM. Combined with the characteristics of CNPs and LSTM, a sequential prediction of MPFs can be made with high efficiency. The predicted multiphase then modified by the multiphase consistent and conservative boundedness mapping (MCBOM) algorithm [47] to strictly satisfy the four PICs. The density and velocity are then predicted from the physics-informed CNP-LSTM model (PICNP-LSTM), which considers a PIC of momentum conservation in the loss function. For velocities and densities are strictly governed by momentum conservation laws, a physics-informed penalty term can shrink the available regression coefficient space towards initial momentum in respect to the maximum

likelihood estimates. With the constraints, PICNP-LSTM can simulate the phase-field flow trend while do not deviate from the actual settings. We simplify this complex CNP-LSTM-MCBOM-PICNP-LSTM structure as physics-informed neural processes (PINPs).

The remaining sections is structured as follows to introduce PINPs. In section 2, the research problem is specified. Section 3 introduces the PINP model for predicting MPFs, including the CNP-LSTM, the MCBOM algorithm, and PICs with momentum conservation. Then in section 4, we verify the feasibility and practicability of PINPs through two applications. Finally, conclusions are drawn in section 5.

2. PROBLEM DEFINITION

In this section, a mathematical description of multiphase flows with physical constraints and time series analysis are defined here.

2.1 Multiphase flows

In the present study, the incompressible MPFs are considered, where there are N ($N \geq 1$) phases, each phase has a fixed density and viscosity, denoted by ρ^p and μ^p, respectively, for Phase p, and each two phases has a surface tension, denoted as $\sigma^{p,q}$ for Phases p and q. The locations of the phases are determined from the volume fraction contrast $\{\phi^p\}_{p=1}^N$, such that $\phi^p = 1$ represents pure Phase p while Phase p is absent at $\phi^p = -1$.

Four physical constraints need to be strictly satisfied by the OPs. The first one is called the summation constraint, i.e.,

$$\sum_{p=1}^{N} \frac{1+\phi^p}{2} = 1, \quad \text{or} \quad \sum_{p=1}^{N} \phi^p = 2 - N. \tag{2.1}$$

The summation constraint requires that the summation of the volume fractions of all the phases is one so that local voids or overfilling is not produced.

The second constraint is the mass (or volume) conservation, i.e.,

$$\frac{d}{dt}\int_{\Omega} \phi^p d\Omega + \int_{\partial\Omega} (\mathbf{n}\cdot\mathbf{u})\phi^p d\Gamma = 0, \quad 1 \leq p \leq N, \tag{2.2}$$

where Ω is the domain of interest, $\partial\Omega$ is the domain boundary, $\mathbf{n}$ is the outward normal at the domain boundary, and $\mathbf{u}$ is the flow velocity. If there is no normal velocity at the domain boundary, the integral of ϕ^p ($1 \leq p \leq N$) over the domain of interest should not be changed. As a result, the volume of each phase, i.e., $\int_{\Omega} \frac{1+\phi^p}{2} d\Omega$, $1 \leq p \leq N$, does not change with time either.

The third constraint is the consistency of reduction, whose definition is in [48], [49], and its necessary condition is

$$\frac{\partial \phi^p}{\partial t}|_{\phi^p \equiv -1} = 0, \tag{2.3}$$

so that Phase p remains absent if it was absent at the beginning. The consistency of reduction eliminates the production of fictitious phases.

The fourth constraint is the boundedness constraint, which is

$$|\phi^p| \leq 1, \forall t > 0, \quad 1 \leq p \leq N. \tag{2.4}$$

One can physically explain $\{\phi^p\}_{p=1}^N$ as volume fraction contrasts only when they are in $[-1, 1]$. This corresponds to the volume fractions $\{\frac{1+\phi^p}{2}\}_{p=1}^N$ in $[0, 1]$.

The density and viscosity of the fluid mixture are

$$\rho = \sum_{p=1}^{N} \rho^p \frac{1+\phi^p}{2}, \tag{2.5}$$

$$\mu = \sum_{p=1}^{N} \mu^p \frac{1+\phi^p}{2}. \tag{2.6}$$

Therefore, if ϕ^p is outside $[-1, 1]$, both the density and viscosity of the fluid mixture can become negative.

The flow velocity is divergence-free, i.e.,

$$\nabla \cdot \mathbf{u} = 0. \tag{2.7}$$

2.2 Time series prediction

Time series is a sequence of data set arranged in chronological order [50], [51]. The time series are analyzed to predict future behaviors of the observation object, so as to propose better decisions. Except fluid simulation [52], [53], time series prediction has already been

widely applied in many different areas, such as economics [54], meteorology [55], [56], and industry [57].

Given the time series of the observed data $\{x_{\mathrm{i}}\}_{\mathrm{i}=1}^{m}$, the goal is to predict the unobserved data in the future, i.e., $\{x_{\mathrm{j}}\}_{\mathrm{j}=m+1}^{n}$. Here, m and n are the numbers of the observed and total (including both observed and unobserved) data, respectively. The subscripts i and j are timestamp indexes for the data.

To achieve the goal, we specify a projection from $\mathbf{x}$ to $\mathbf{y}$ with a Borel measurable function f such that:

$$\mathbf{y} = f(\mathbf{x}) + \boldsymbol{\varepsilon}, \tag{2.8}$$

where $\mathbf{x} = \{x_k\}_{k=\mathrm{i}}^{t+\mathrm{i}}$, $\mathbf{y} = \{x_{k+1}\}_{k=\mathrm{i}}^{t+\mathrm{i}}$, and $\boldsymbol{\varepsilon} = \{\varepsilon_k\}_{k=\mathrm{i}}^{t+\mathrm{i}}$ $(1 \leq \mathrm{i} \leq n-t-1)$ is the bias vector. Here t $(1 \leq t \leq m)$ is the sequence length. We first assume that the elements in $\boldsymbol{\varepsilon}$ are independent and identically distributed, and follow the Gaussian distribution:

$$\varepsilon_{\mathrm{i}} \sim \mathcal{N}(0, \sigma_{\mathrm{i}}^2), \tag{2.9}$$

where σ_k^2 is the variance of ε_k. We further assume that f is linear with respect to $\mathbf{x}$, i.e.,

$$f(\mathbf{x}) = \mathbf{x}^T\boldsymbol{\theta}, \tag{2.10}$$

where $\boldsymbol{\theta}$ is the parameter space of the proposed model. Thanks to the above two assumptions, we can derive the following likelihood for $\mathbf{y}$:

$$p(\mathbf{y}|\mathbf{x}, \boldsymbol{\theta}) = \mathcal{N}(f(\mathbf{x}), \boldsymbol{\sigma}^2\mathbf{I}) = \mathcal{N}(\mathbf{x}^T\boldsymbol{\theta}, \boldsymbol{\sigma}^2\mathbf{I}), \tag{2.11}$$

where $\boldsymbol{\sigma}^2$ is the variance vector that represents $\{\sigma_k^2\}_{k=\mathrm{i}}^{t+\mathrm{i}}$.

To determine $\boldsymbol{\theta}$ and $\boldsymbol{\sigma}$, we minimize the risk, denoted by $R_{\mathcal{L}}(f)$, and its definition is

$$R_{\mathcal{L}}(f) = \int \mathcal{L}(\mathbf{y}, f(\mathbf{x}))d\mu(\mathbf{x}, \mathbf{y}). \tag{2.12}$$

Here $\mathcal{L}(\mathbf{y}, f(\mathbf{x}))$ is a loss function to measure the difference between $\mathbf{y}$ and $f(\mathbf{x})$, and $\mu(\mathbf{x}, \mathbf{y})$ is a probability measure given by observations $\mathbf{x}$ and $\mathbf{y}$. With the given $\mathbf{x}$ and $\mathbf{y}$, we apply Kaiming initialization strategy [58] to initialize $\boldsymbol{\theta}$, and then update the parameters $\boldsymbol{\theta}$ and variances $\boldsymbol{\sigma}^2$ during the training, until we find $\widehat{\boldsymbol{\theta}}$ and $\widehat{\boldsymbol{\sigma}}^2$ that minimize the risk in Eq.(2.12). Correspondingly, we obtain $\widehat{f}(\mathbf{x})$ from Eq.(2.10) with $\widehat{\boldsymbol{\theta}}$, and the prediction $\widehat{\mathbf{y}}$ from $\widehat{f}(\mathbf{x})$ has the following distribution:

$$\widehat{\mathbf{y}} = \widehat{f}(\mathbf{x}) \sim \mathcal{N}(\mathbf{x}^T\widehat{\boldsymbol{\theta}}, \widehat{\boldsymbol{\sigma}}^2\mathbf{I}), \tag{2.13}$$

from Eq.(2.11) with $\widehat{\boldsymbol{\sigma}}$. It should be noted that both $\widehat{\boldsymbol{\theta}}$ and $\widehat{\boldsymbol{\sigma}}$ are obtained from a finite number of observation data. Suppose the exact solution of minimizing $R_{\mathcal{L}}(f)$ in Eq.(2.12) with infinite observation data is $f^*_{\mathcal{L}}(\mathbf{x})$, we assume that $\widehat{f}(\mathbf{x})$ is consistent with $f^*_{\mathcal{L}}(\mathbf{x})$, i.e.,

$$R_{\mathcal{L}}(\widehat{f}) \to R_{\mathcal{L}}(f^*_{\mathcal{L}}), \quad \text{when } m \to \infty, \tag{2.14}$$

When the model is trained well, i.e., $\widehat{f}(\mathbf{x}) \approx f^*_{\mathcal{L}}(\mathbf{x})$, the expectation of the unobserved data, i.e., $\mathbf{x}' = \{x_k\}_{k=\mathrm{i}}^{t+\mathrm{i}}$ $(m+2 \le \mathrm{i} \le n-t)$, will be predicted from $\widehat{f}(\mathbf{x}')$, and its distribution follows

$$\widehat{f}(\mathbf{x}')|\mathbf{x}', \mathbf{x}, \mathbf{y} \sim \mathcal{N}((\mathbf{x}')^T\widehat{\boldsymbol{\theta}}, \widehat{\boldsymbol{\sigma}'}^2\mathbf{I}), \tag{2.15}$$

where $\widehat{\boldsymbol{\sigma}'}^2 = \{\sigma_k^2\}_{k=\mathrm{i}}^{t+\mathrm{i}}$.

In practice, the training stopped when either some stop criteria are met or the number of training times reaches the specified one. In sections 3.1 and 3.3, we will elaborate the method we used to find the distribution $\widehat{f}(\mathbf{x}')$ in Eq.(2.15). It should be noted that more specific notations will be used in section 3 for predicting the OPs $\boldsymbol{\phi} = \{\phi^p\}_{p=1}^N$ and the flow velocity $\mathbf{v} = \{v^p\}_{p=1}^N$. The sequence length t is set to 1 in the proposed neural network, unless otherwise specified.

3. THE PROPOSED PHYSICS-INFORMED NEURAL PROCESSES

This section will elaborate on the model that we developed for multiphase flow prediction, including the CNP-LSTM model to predict the OPs in Section 3.1, the boundedness mapping (MCBOM) from [15] to correct the order parameters in Section 3.2, and the PICNP-LSTM model to predict the flow velocity in Section 3.3.

3.1 The neural processes to predict the order parameters

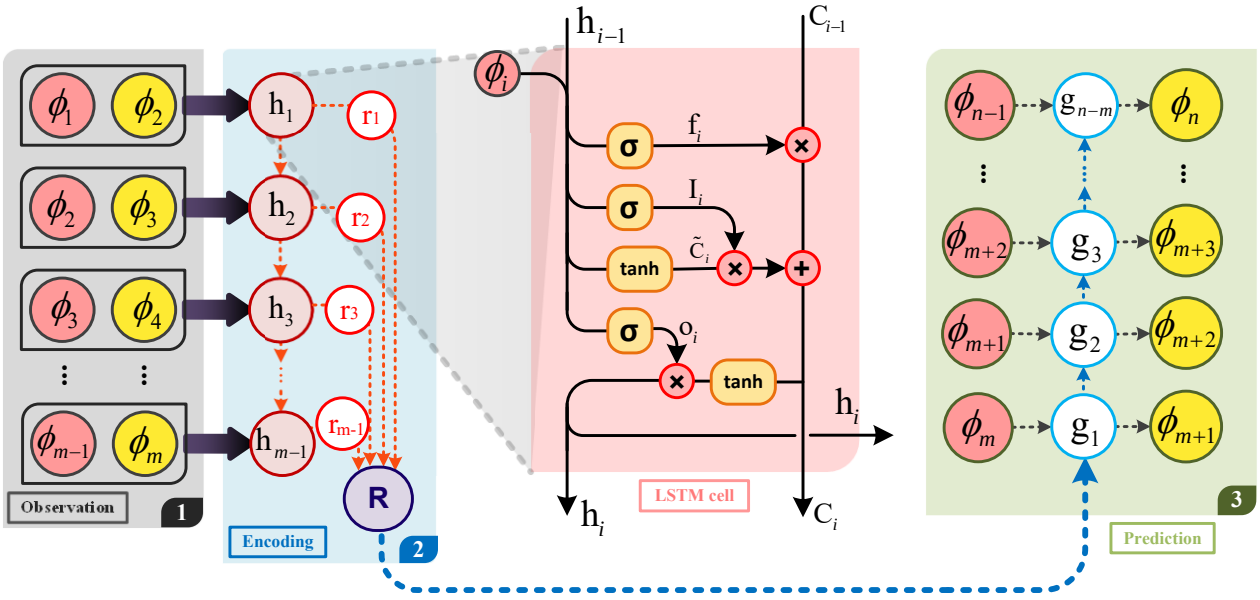

Figure 3.1. A general framework of CNP-LSTM to predict order parameters. The order parameters (ϕ_{i}, i $= 1, 2, \cdots, m$) are processed by LSTM cells and get r_{i}. Then a commutative operation transform a combination of r_1, r_2, $\cdots$, r_1 to $R_{\mathrm{i}} \in \mathcal{R}$. Then R_{i} and ϕ_{j} will be used to predict $\phi_{\mathrm{j+1}}$ (j $= m, m+1, \cdots, n-1$). The red diagram in the middle (LSTM cell) represents mathematical operations in a LSTM cell.

The following operations is applied to ϕ^p $(1 \leq p \leq N)$ independently. Therefore, we skip the superscript for convenience. Given a set of OPs $\{\phi_{\mathrm{i}}\}_{\mathrm{i}=1}^{m}$, a CNP-LSTM is a NPs [59], [60] that developed to predict the OPs $\{\phi_{\mathrm{j}}\}_{\mathrm{j}=m+1}^{n}$. The workflow of LSTM is shown in Figure 3.1.

Each LSTM cell maps inputs ϕ_{i} and $\phi_{\mathrm{i}+1}$ to hidden state h_{i} and cell state C_{i} through several mathematical operations. LSTM cells are then connected by the hidden states and cell states to form a complete LSTM network.

LSTM is effective to solve long-term dependent problems because it introduces the forget gate f_{i} in each LSTM cell, see Figure 3.1, to help controlling the circulation and loss of features of the data. For each LSTM cell, the input of a LSTM cell includes the output of its previous cell, and all LSTM cells share the same structure and parameters. In the following content, we will demonstrate the structure of a LSTM cell and then explain how LSTM cells connect with each other to construct a CNP-LSTM network. Inside LSTM cell i ($1 \leq \mathrm{i} \leq m-1$), a forget gate $f_{\mathrm{i}} \in [0,1]$ is defined as a sigmoid unit [35],

$$f_{\mathrm{i}} = S(b_f + U_f \cdot \phi_{\mathrm{i}} + W_f \cdot h_{\mathrm{i}-1}), \tag{3.1}$$

where $S(\cdot)$ is the sigmoid function, $h_{\mathrm{i}-1}$ is the hidden state in the previous LSTM cell. Here, b, U, and W are the bias, input weight, and cycle weight, respectively, and their subscript f represents the forget gate. The same format is used in the rest of the equations. Then an input gate I_{i} decides which information needs to be updated from

$$I_{\mathrm{i}} = S(b_{\mathrm{i}} + U_{\mathrm{i}} \cdot \phi_{\mathrm{i}} + W_{\mathrm{i}} \cdot h_{\mathrm{i}-1}), \tag{3.2}$$

To obtain the candidate value $\widetilde{C}_{\mathrm{i}}$, a hypertangent activation function is applied,

$$\widetilde{C}_{\mathrm{i}} = \tanh(b_{\widetilde{C}} + U_{\widetilde{C}} \cdot \phi_{\mathrm{i}} + W_{\widetilde{C}} \cdot h_{\mathrm{i}-1}). \tag{3.3}$$

Then, we obtain the current cell state C_{i}, incorporating the previous cell state $C_{\mathrm{i}-1}$ and the candidate value $\widetilde{C}_{\mathrm{i}}$ in Eq.(3.3),

$$C_{\mathrm{i}} = f_{\mathrm{i}} \cdot C_{\mathrm{i}-1} + I_{\mathrm{i}} \cdot \widetilde{C}_{\mathrm{i}}. \tag{3.4}$$

The cell states can be understood as the memory of LSTM, which connects the previous, current, and future LSTM cells. Next, we move to an output gate o_{i}, which decides the general state of the next hidden state, Here we use a sigmoid function to activate previous hidden state $h_{\mathrm{i}-1}$, and project ϕ_{i} to the output gate o_{i} by

$$o_{\mathrm{i}} = S(b_o + U_o \cdot \phi_{\mathrm{i}} + W_o \cdot h_{\mathrm{i}-1}). \tag{3.5}$$

Finally, hidden state h_{i} is computed from the current cell state C_{i} in Eq.(3.4) and the output gate o_{i} in Eq.(3.6) with.

$$h_{\mathrm{i}} = \tanh(C_{\mathrm{i}}) \cdot o_{\mathrm{i}}. \tag{3.6}$$

So far, we have completed the operations in a LSTM cell. Once the hidden state h_{i} is computed, we encode it with input via a modified CNP model. Traditional CNP uses multiple dense layers to encode inputs. Here the dense layers are replaced with LSTM to reflect the dependencies among time series data. Different from the dense layers in traditional CNP, the present hidden state h_{i} depends on not only the observations but also the previous hidden state $h_{\mathrm{i}-1}$. Same as the procedure of CNPs, the present neural networks define the representation r_{i} as a projection of hidden state h_{i} $(1 \leq \mathrm{i} \leq m-1)$, OPs ϕ_{i} and $\phi_{\mathrm{i}+1}$:

$$r_{\mathrm{i}} = H_1\left(\phi_{\mathrm{i}}, \phi_{\mathrm{i}+1}, h_{\mathrm{i}}\right), \tag{3.7}$$

where $H(\cdot)$ is an affine transformation followed by a non-linear activation (usually this is a sigmoid function):

$$H_1\left(\phi_{\mathrm{i}}, \phi_{\mathrm{i}+1}, h_{\mathrm{i}}\right) = S\left(\boldsymbol{A}_1 \cdot [\phi_{\mathrm{i}}\ \phi_{\mathrm{i}+1}\ h_{\mathrm{i}}] + \boldsymbol{b}_1\right). \tag{3.8}$$

Then a commutative operation maps all the representations $\{r_{\mathrm{i}}\}_{\mathrm{i}=1}^{m-1}$ into a one-dimensional element R_{i} with the following rules:

$$R_{\mathrm{i}} = r_1 \oplus r_2 \oplus \cdots \oplus r_{\mathrm{i}}, \tag{3.9}$$

where $\oplus$ here is the mean operation. Then we use another affine transformation with a non-linear activation to map the elements R_{i} and the unobserved data $\phi_{\mathrm{i}+1}$ into $\phi_{\mathrm{i}+2}$:

$$\phi_{\mathrm{i}+2} = H_2\left(\phi_{\mathrm{i}+1}, R_{\mathrm{i}}\right) = S\left(\boldsymbol{A}_2 \cdot \left[\phi_{\mathrm{i}+1}\ R_{\mathrm{i}}\right] + \boldsymbol{b}_2\right), \tag{3.10}$$

where $\phi_{\mathrm{i}+2}$ can be expressed as a Gaussian distribution $\widehat{f}(\phi_{\mathrm{i}+1})$, which stores both the mean of $\widehat{m}(\phi_{\mathrm{i}+1})$ and the variance of $\widehat{\sigma}^2_{\mathrm{i}+2}$:

$$\phi_{\mathrm{i}+2} = \widehat{f}(\phi_{\mathrm{i}+1}) \sim \mathcal{N}(\widehat{m}(\phi_{\mathrm{i}+1}), \widehat{\sigma}^2_{\mathrm{i}+2}). \tag{3.11}$$

It should be noted here that since $\widehat{m}(\cdot)$ and $\widehat{\sigma}^2_{\mathrm{i}+2}$ are predicted by separating the outputs of neural networks into two parts, the variance $\widehat{\sigma}^2_{\mathrm{i}+2}$ here is a prediction given by the neural network rather than the one in statistics. Therefore, comparing $\hat{\sigma}$ is not a good way to judge the predictive accuracy.

To update parameters in the CNP-LSTM network, we defined a loss function to calculate the difference between the predicted ϕ and their observations here. Given the observed time series $\boldsymbol{\phi} = \{\phi_{\mathrm{i}}\}_{\mathrm{i}=1}^{m}$ and the distribution of $\phi_{\mathrm{j}+1}$ $(= \widehat{f}(\phi_{\mathrm{j}}),\ m \leq \mathrm{j} \leq n-1)$, a conditional stochastic processes:

$$Q_\theta(\hat{f}(\boldsymbol{\phi}')|\boldsymbol{\phi}, \boldsymbol{\phi}') = \prod_{\mathrm{j}} \hat{f}_\theta(\phi_{\mathrm{j}}|\boldsymbol{\phi}, \boldsymbol{\phi}') \tag{3.12}$$

defines distributions over $\{\widehat{f}(\phi_{\mathrm{j}})\}_{\mathrm{j}=m}^{n-1}$. Here, $\boldsymbol{\phi}' = \{\phi_{\mathrm{j}}\}_{\mathrm{j}=m}^{n-1}$. Given Q_θ, the loss value $\mathcal{L}(\theta)$ in Eq.(2.12) is expressed as the expected value of the sum of the log likelihood function of Q_θ:

$$\mathcal{L}_\phi(\theta) = -E_{\boldsymbol{\phi},\boldsymbol{\phi}'}\left\{E_{\hat{f}}\left[\textstyle\sum_{\boldsymbol{\phi}'} \log Q_\theta(\hat{f}(\boldsymbol{\phi}')|\boldsymbol{\phi}, \boldsymbol{\phi}')\right]\right\}. \tag{3.13}$$

In practice, we minimize $\mathcal{L}_\phi(\theta)$ by Adam optimizer [13].

3.2 The multiphase consistent and conservative boundedness mapping algorithm

After predicting $\{\phi_k^p\}_{p=1}^N$ $(m+1 \leq k \leq n)$ from all the previous data before timestamp k by CNP-LSTM in Section 3.1, there is no guarantee that $\{\phi_k^p\}_{p=1}^N$ satisfy the summation constraint, the mass conservation, and the boundedness constraint, which are the physical constraints on the OPs, as discussed in Section 2. Even though those errors are small after performing one time step, it will accumulate and become significant as the computation proceeds. The MCBOM algorithm proposed in [15] is developed to correct those small errors in each time step. The resulting $\{\phi_b^p\}_{p=1}^N$ not only satisfy the summation constraint, the mass conservation, and the boundedness constraint, but also are reduction-consistent with the given data $\{\phi_k^p\}_{p=1}^N$. Then, we replace $\{\phi_k^p\}_{p=1}^N$ with $\{\phi_b^p\}_{p=1}^N$, and continue the rest of the process. MCBOM works for every timestamp, and therefore it is skipped in the following descriptions in this section.

Given a set of volume fraction contrasts $\{\phi^p\}_{p=1}^N$ and a set of scalars $\{\Phi^p\}_{p=1}^N$ that represents total amounts of $\{\phi^p\}_{p=1}^N$ inside the domain of interest, the boundedness mapping includes three steps. The first one is the clipping step:

$$\begin{cases} 1, & \phi^p \geq 1, \\ -1, & \phi^p \leq -1, \\ \phi^p, & \text{else.} \end{cases} \tag{3.14}$$

The second one is the re-scaling step

$$C_{b*}^p = \frac{1+\phi_{b*}^p}{2}, \quad C_{b**}^p = \frac{C_{b*}^p}{\sum_{q=1}^N C_{b*}^q}, \quad \phi_{b**}^p = 2C_{b**}^p - 1. \tag{3.15}$$

The last step is the conservation step

$$\phi_b^p = \phi_{b**}^p + \sum_{q=1}^N W_{b**}^{p,q} B^q, \tag{3.16}$$

where $W^{p,q}$ is the weight function

$$W^{p,q} = \begin{cases} -(1+\phi^p)(1+\phi^q), & p \neq q, \\ (1+\phi^p)(1-\phi^p), & p = q, \end{cases} \tag{3.17}$$

and $\{B^p\}_{p=1}^N$ are obtained from solving the linear system

$$\left[\int_\Omega W_{b**}^{p,q} d\Omega\right][B^p] = \left[\Phi^p - \int_\Omega \phi^p d\Omega\right]. \tag{3.18}$$

In practice, the integral is approximated by the mid-point rule. The resulting $\{\phi_b^p\}_{p=1}^N$ has the following properties

$$\sum_{q=1}^{N} \frac{1+\phi_b^q}{2} = 1, \quad |\phi_b^p| \leq 1, \quad \int_\Omega \phi_b^p d\Omega = \Phi^p, \quad \phi_b^p|_{\phi^p \leq -1}, \quad 1 \leq p \leq N. \tag{3.19}$$

MCBOM modifies the prediction given by CNP-LSTM so that the properties of the OPs are strictly satisfied by the final prediction.

3.3 Loss function with physics-informed constraints

Once the OPs are predicted from CNP-LSTM in Section 3.1 and corrected by MCBOM in Section 3.2, the density and viscosity of the mixture are computed from Eq.(2.5) and Eq.(2.6), respectively. Thanks to satisfying the boundedness constraint of the OPs, the density and viscosity of the fluid mixture will stay in their physical interval as well, no matter how large the density/viscosity ratio of the problem could be. The remaining task is to predict the flow velocity $\{v_{\mathrm{j}}\}_{\mathrm{j}=m+1}^n$ given the observations $\{v_{\mathrm{i}}\}_{\mathrm{i}=1}^m$, which is finished by the proposed PICNP-

LSTM. The structure of CNP-LSTM network here is the same as the one in section 3.1, but the input becomes the flow velocity rather than the OPs:

$$\begin{cases} f'_{\mathrm{i}} = S(b'_f + U'_f \cdot v_{\mathrm{i}} + W'_f \cdot h'_{\mathrm{i}-1}), \\ I'_{\mathrm{i}} = S(b'_{\mathrm{i}} + U'_{\mathrm{i}} \cdot v_{\mathrm{i}} + W'_{\mathrm{i}} \cdot h'_{\mathrm{i}-1}), \\ \widetilde{C}'_{\mathrm{i}} = \tanh(b'_{\widetilde{C}} + U'_{\widetilde{C}} \cdot v_{\mathrm{i}} + W'_{\widetilde{C}} \cdot h'_{\mathrm{i}-1}), \\ C'_{\mathrm{i}} = f'_{\mathrm{i}} \cdot C'_{\mathrm{i}-1} + I'_{\mathrm{i}} \cdot \widetilde{C}'_{\mathrm{i}}, \\ o'_{\mathrm{i}} = S(b'_o + U'_o \cdot v_{\mathrm{i}} + W'_o \cdot h'_{\mathrm{i}-1}), \\ h'_{\mathrm{i}} = \tanh(C'_{\mathrm{i}}) \cdot o'_{\mathrm{i}}, \\ r'_{\mathrm{i}} = H'_1\left(v_{\mathrm{i}}, v_{\mathrm{i}+1}, h'_{\mathrm{i}}\right), \\ R'_{\mathrm{i}} = r'_1 \oplus r'_2 \oplus \cdots \oplus r'_{\mathrm{i}}, \\ v_{\mathrm{i}+2} = H'_2\left(v_{\mathrm{i}+1}, R'_{\mathrm{i}}\right), \end{cases} \tag{3.20}$$

and our prediction $v_{\mathrm{i}+2}$ is expressed as a Gaussian distribution with mean $\widehat{m'}(v_{\mathrm{i}+1})$ and variance $\widehat{\sigma'}^2_{\mathrm{i}+2}$:

$$v_{\mathrm{i}+2} = \widehat{f'}(v_{\mathrm{i}+1}) \sim \mathcal{N}(\widehat{m'}(v_{\mathrm{i}+1}), \widehat{\sigma'}^2_{\mathrm{i}+2}). \tag{3.21}$$

Here we use the prime symbol ($'$) to distinguish the symbols in In Eqs.(3.20)-(3.21) from the ones in Section 3.1 because the parameter spaces of neural networks are not the same. Also, the loss function are different from Section 3.1, and an additional penalty is considered to account for the momentum conservation. The momentum of the multiphase flow is

$$\mathcal{M}_{\mathrm{j}} = \sum_{n_C} \left[\rho_{\mathrm{j}} v_{\mathrm{j}} \Delta\Omega\right]_{n_C}, \tag{3.22}$$

where $\mathcal{M}_{\mathrm{j}}$ is the total momentum of the multiphase flow at timestamp j, and $\mathrm{j} = m+1, m+2, \cdots, n$. We can get the flow density ρ_{j} from the predicted OPs:

$$\rho_{\mathrm{j}} = \sum_{p=1}^{N} \frac{\phi^p_{\mathrm{j}} + (N-1)}{2} \cdot \rho^p_{\mathrm{j}}, \tag{3.23}$$

where ρ_{j}^{p} is the density for the phase flow p at time step j. The loss function for predicting the flow velocity is

$$\begin{aligned}\mathcal{L}_v(\theta_v) = &-E_{\boldsymbol{v},\boldsymbol{v}'}\left\{E_{\widehat{f}'}\left[\textstyle\sum_{\boldsymbol{v}'}\log Q_v(\widehat{f}'(\boldsymbol{v}')|\boldsymbol{v},\boldsymbol{v}')\right]\right\}\\ &+\lambda\left|\textstyle\sum_{n_C}\left[\sum_{\boldsymbol{\rho}'\boldsymbol{v}'}\boldsymbol{\rho}'\widehat{f}'(\boldsymbol{v}')\Delta\Omega\right]_{n_C}-\sum_{\mathrm{j}}\mathcal{M}_{\mathrm{j}}\right|, \qquad \mathrm{j}=m+1,m+2,\cdots,n,\end{aligned} \tag{3.24}$$

where $\boldsymbol{v}' = \{v_{\mathrm{i}}\}_{\mathrm{i}=1}^{m}$, $\boldsymbol{v} = \{v_{\mathrm{j}}\}_{\mathrm{j}=m}^{n-1}$, $\boldsymbol{\rho}' = \{\rho_k\}_{k=m+1}^{n}$, and λ is the Lagrange multiplier. Here, $\boldsymbol{\rho}'$ and $\boldsymbol{v}'$ form a PIC and penalize an unphysical solution. Right now, the loss function in Eq.(3.24) not only includes the part to maximize the log likelihood (the one in Eq.(3.13)), but also has the penalty due to violating the momentum conservation. We call this model physics-informed conditional neural processes and long short-term memory (PICNP-LSTM).

A flowchart of the whole process in sections 3.1-3.3 is shown in Figure 3.2. The methods to update neural network parameters and predictions are elaborated in Algorithm ??. The consideration of PICs can alleviate errors of the predictions deviated from the actual situation to some extent, and the effects will then be shown in the next section.

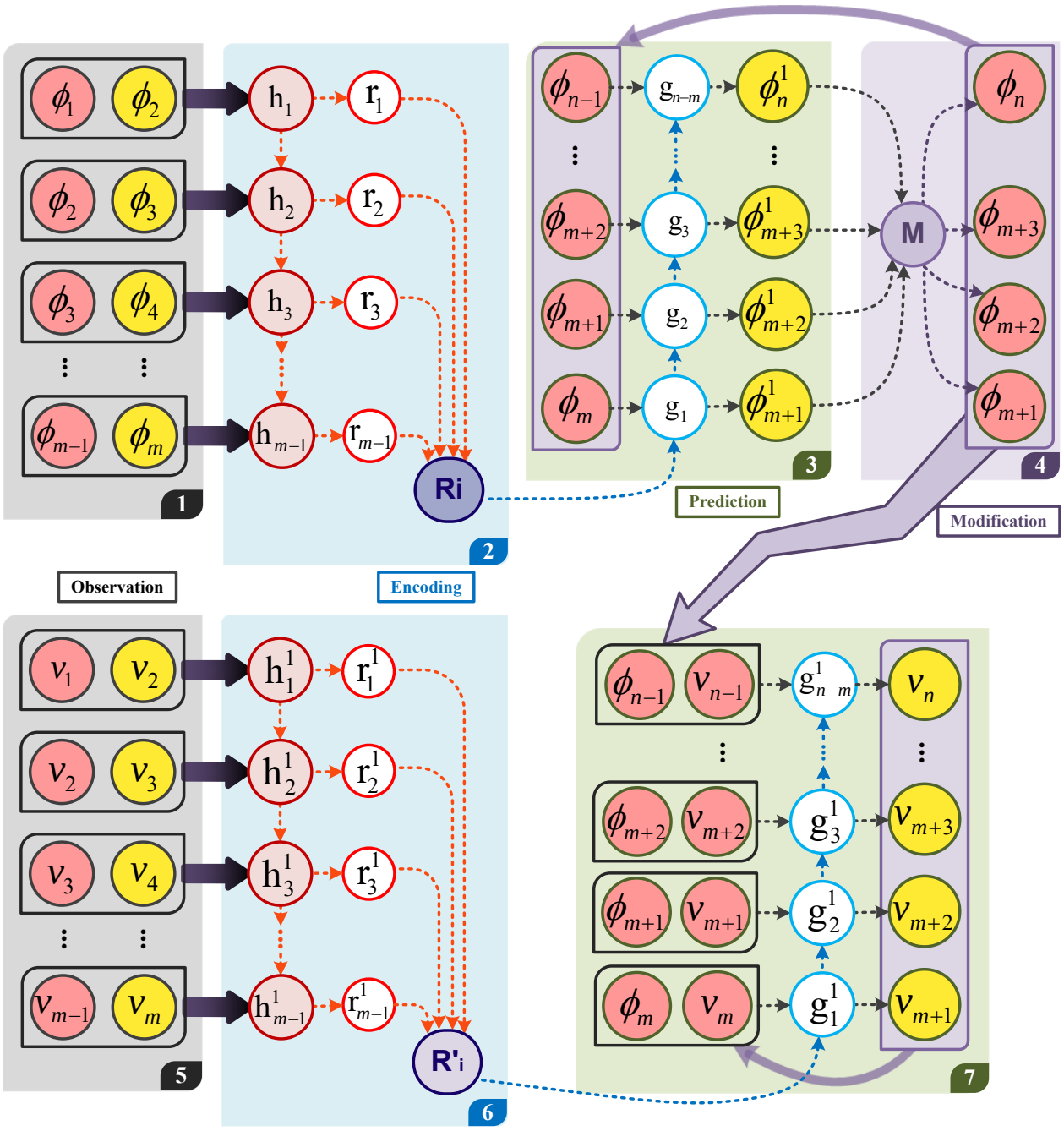

Figure 3.2. A general framework of PINP model. The observations $\{\phi_1, \phi_2, \cdots, \phi_m\}$ are encoded in the encoder in CNP-LSTM to update R_i and then the CNP-LSTM approximates a prediction of ϕ'_{i+1} at first. This prediction will be modified by MCBOM algorithm and get ϕ_{i+1}. The modified ϕ_{i+1} will constrain the update of v_{i+1} as a form of regularization term in the loss function of the second neural network. v_{i+1} will be predicted by the PICNP-LSTM.

Table 3.1. An algorithm to update strategy using Adam optimization. This algorithm first update parameter space $\boldsymbol{\theta}_1$ with Adam optimizer. Then the MCBOM algorithm is applied to obtain the modified order parameters. The modified current order parameters and flow velocities in the last time step are applied to update parameter space $\boldsymbol{\theta}_2$ with the use of Adam optimizer and implicit physical constraints in the loss function.

Require: Observations: order parameters $\{\boldsymbol{\phi}_1, \boldsymbol{\phi}_2, \cdots, \boldsymbol{\phi}_m\}$ and velocities $\{\boldsymbol{v}_1, \boldsymbol{v}_2, \cdots, \boldsymbol{v}_m\}$.
Require: Learning rate for order parameters and velocities: η, η'.
Require: Exponential decay rates: β_1, β_2.
Require: Small constant for stabilization: ϵ.

Initialize time stamp: $\mathrm{i} = 1$.
Initialize the first and second moment: $\boldsymbol{s} = 0$, $\boldsymbol{r} = 0$, $\boldsymbol{s}' = 0$, and $\boldsymbol{r}' = 0$.
while stopping criterion not met **do**
  while $\mathrm{i} \neq n - 1$ **do**
    procedure UPDATE $\widehat{f}(\boldsymbol{\phi}_\mathrm{i}, \boldsymbol{\theta}_1)$:
      Compute gradient: $\boldsymbol{g} \leftarrow \frac{1}{m}\Delta_{\boldsymbol{\theta}_1}\Sigma_\mathrm{i}\mathcal{L}(\hat{f}(\boldsymbol{\phi}_\mathrm{i}, \boldsymbol{\theta}_1))$.
      Update the first moment estimate: $\boldsymbol{s} \leftarrow \beta_1 \cdot \boldsymbol{s} + (1 - \beta_1)\boldsymbol{g}$.
      Update the second moment estimate: $\boldsymbol{r} \leftarrow \beta_2 \cdot \boldsymbol{r} + (1 - \beta_2)\boldsymbol{g} \odot \boldsymbol{g}$.
                                                ▷ $\odot$ is the Hadamard product [61].
      Correct the first moment estimation: $\hat{\boldsymbol{s}}$ $\frac{\boldsymbol{s}}{1-\beta_1}$.
      Correct the second moment estimation: $\hat{\boldsymbol{r}}$ $\frac{\boldsymbol{r}}{1-\beta_2}$.
      Update parameters: $\boldsymbol{\theta}_1 \leftarrow \boldsymbol{\theta}_1 - \eta\frac{\hat{\boldsymbol{s}}}{\sqrt{\hat{\boldsymbol{r}}}+\epsilon}$.
      Update prediction: $\boldsymbol{\phi}'_{\mathrm{i}+1} \leftarrow \boldsymbol{\phi}_\mathrm{i}^T\boldsymbol{\theta}_1$.
    Modification: $\boldsymbol{\phi}_{\mathrm{i}+1} \leftarrow \boldsymbol{\phi}'_{\mathrm{i}+1}$. ▷ Eqs.(3.14)-(3.19).
    Transformation: $\boldsymbol{\rho}_{\mathrm{i}+1} \leftarrow \boldsymbol{\phi}_{\mathrm{i}+1}$. ▷ Eq.(3.23).
    procedure UPDATE $\widehat{f}'(\boldsymbol{v}_\mathrm{i}, \theta'_2)$:
      Compute gradient: $\boldsymbol{g}'$ $\frac{1}{m}(\Delta_{\boldsymbol{\theta}_2}\Sigma_\mathrm{i}\mathcal{L}(f(\boldsymbol{v}_\mathrm{i}, \boldsymbol{\theta}'_2)) + \lambda|\sum_{n_C}[\sum_{\boldsymbol{\rho v}}\boldsymbol{\rho}_{\mathrm{i}+1}\widehat{f}'(\boldsymbol{v}_\mathrm{i})\Delta\Omega]_{n_C} - \sum_\mathrm{i}\mathcal{M}_\mathrm{i}|$.
      Update the first moment estimate: $\boldsymbol{s}' \leftarrow \beta_1 \cdot \boldsymbol{s}' + (1 - \beta_1)\boldsymbol{g}'$.
      Update the second moment estimate: $\boldsymbol{r}' \leftarrow \beta_2 \cdot \boldsymbol{r}' + (1 - \beta_2)\boldsymbol{g}' \odot \boldsymbol{g}'$.
      Correct the first moment estimation: $\hat{\boldsymbol{s}}' \leftarrow \frac{\boldsymbol{s}'}{1-\beta_1}$.
      Correct the second moment estimation: $\hat{\boldsymbol{r}}'$ $\frac{\boldsymbol{r}'}{1-\beta_2}$.
      Update parameters: $\boldsymbol{\theta}'_2$ $\boldsymbol{\theta}'_2 - \eta'\frac{\hat{\boldsymbol{s}}'}{\sqrt{\hat{\boldsymbol{r}}'}+\epsilon}$.
      Update prediction: $\boldsymbol{v}_{\mathrm{i}+1}$ $\boldsymbol{v}_\mathrm{i}^T\boldsymbol{\theta}'_2$.
      Update time step: i i + 1.
return $\{\boldsymbol{\phi}_{m+1}, \boldsymbol{\phi}_{m+2}, \cdots, \boldsymbol{\phi}_n\}$ and $\{\boldsymbol{v}_{m+1}, \boldsymbol{v}_{m+2}, \cdots, \boldsymbol{v}_n\}$.

---

4. EXPERIMENTS

We will elaborate on two experiments here to prove the efficiency and practicability of PINPs. The first experiment is the horizontal shear layer problem and the other one is the dam break (DB) problem. A general summary of the data set is given in Table 4.1, including the number of phases, simulation duration, number of time steps, number of grid of the simulation.

Table 4.1. A general summary of data set. The number of phases, number of available time steps, total simulation time, and their grid sizes are provided in this table.

Dataset	# of phase	# time steps available	Total time	Grid size
HSL	3	160	2.0	$[128 \times 128]$
DB	3	200	10.0	$[512 \times 128]$

4.1 Horizontal shear layer simulation

Horizontal shear layers (HSLs) with small-scale turbulence happen under a variety of contexts [62]: e.g., along coastal fronts, at the edges of energetic ocean currents, in oceans and atmospheric currents around the terrain. The study of the diffusion patterns of horizontal shear flows and the prediction of their propagation trends can be widely applied to hemodynamic analysis [63], ocean circulation [64], etc. However, due to its external disturbance, it may develop into turbulence in mixed layers over large horizontal distances, which makes the prediction tasks more difficult.

In our task, the numerical domain of the problem is in a $[1 \times 1]$ box with the periodic boundary condition, and the cell size is $\frac{1}{128}$ in both horizontal and vertical directions. The time step is set to be $\Delta t = 0.0125$. The viscosity and density of each flow here are: $\mu_1 = 0.01$, $\mu_2 = 0.10$, $\mu_3 = 0.05$, $\rho_1 = 50.0$, $\rho_2 = 10.0$ and $\rho_3 = 1.0$. The surface tensions are $\sigma_{1,2} = 0.05$, $\sigma_{1,3} = 0.01$, and $\sigma_{2,3} = 0.1$. We set $\eta = 0.01$ and $M_0 = 10^{-7}$. When t=0, phase 1 occupies the region $y \in (y_0,\ y_2]$ and keeps stationary. Phase 2 is between $y \in (y_1,\ y_0]$ and moving

to the right with a unity speed. Phase 3 is at the rest of the domain ($y \in [0, y_1] \cup (y_2, 1.0]$) and is moving to the left with a unity speed. Here we have y_0=0.5, y_1=0.25, and y_2=0.75. A sinusoidal vertical velocity is applied here to model the perturbation with an amplitude of 0.05 and wavelength of 2π.

For the deep neural networks, the LSTM consists of a single feed-forward layer followed by a recurrent LSTM layers with 64 LSTM cells, a combination of two hidden layers with 32 neurons each, and ReLU as their activation functions to construct the CNPs. The learning rate is set to 10^{-5}, and weights are initialized from a uniform distribution over the range [-0.05, 0.05]. The dataset within the first half time is set as the training set, the rest half of the dataset as the validation set. The predictive results is shown in Fig. 4.1.

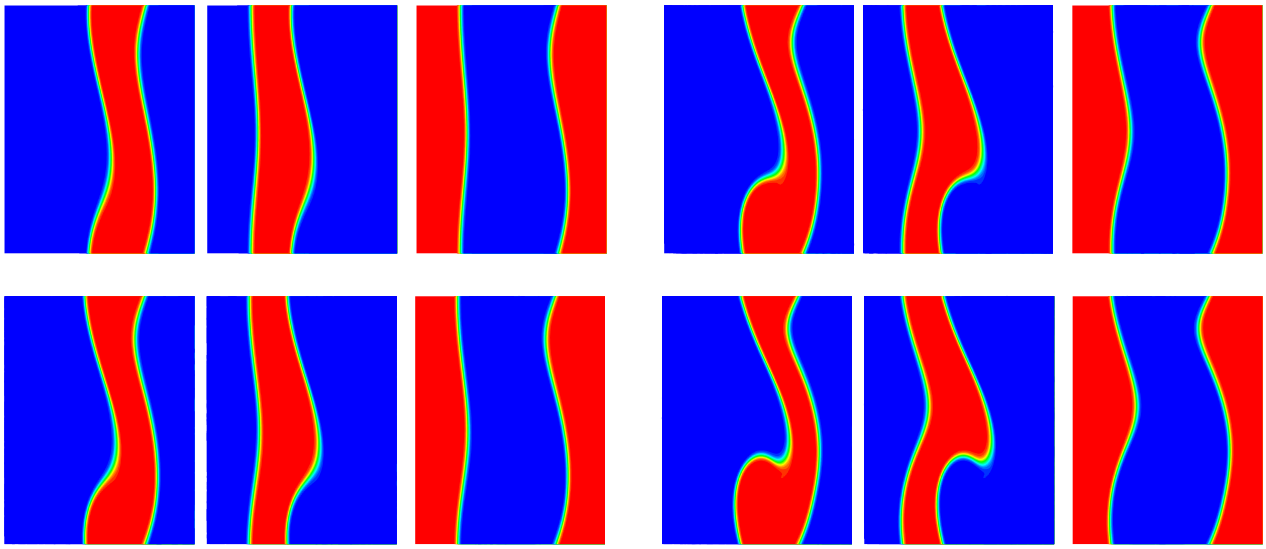

Figure 4.1. Phase values indication of the three-phase shear layer simulation. For example, in the top left figure, the left blue one, middle red one, and the right blue one represent three different phase flow separately. The red one represents one of the phases (its phase is close to 1), and the remaining blue ones are the rest of other two phases (their values are close to -1). Therefore, three side-by-side pictures indicate the position information of multiphase flows at a certain time. From top to bottom, from left to right, the results show the change of shear layer over time t from 0.8s to 2.0s with 0.3s increment.

We first illustrate the necessity of MCBOM in Section 3.2. For a clear presentation, we only show the results of the first phase in Fig. 4.2. We run the CNP-LSTM model in Section 3.1 until its training loss converges, which, in our experiment, is about 400 epochs. Fig. 4.2(a) shows the mass change of the phase in the entire domain versus time. It is clear that the mass of the phase is time-independent, after applying MCBOM, while the mass is fluctuating from the output of CNP-LSTM in Section 3.1 only. In other words, mass gain or loss appears without using MCBOM, which violates the principle of mass conservation. This issue is successfully addressed with the help of MCBOM algorithm. Fig. 4.2(b) indicates the time history of the maximum absolute value of the phase. The OPs is limited between -1 and 1, as mentioned in Section 2, so that the density and viscosity of the fluid mixture computed from the OPs are also in their physical intervals, respectively. Without performing MCBOM, the prediction from CNP-LSTM exceeds the interval $[-1.1]$ even after a long time of training. This out-of-bound issue does not appear in the results of the proposed model using MCBOM. Further, we count the number of locations where the OP of the phase is beyond $[-1, 1]$, or $\max|\phi_p| > 1$, and the results are shown in Fig. 4.2(c). As time goes on, more and more locations have an out-of-bound OP in the prediction from CNP-LSTM, and the number finally reach around 34,000. With MCBOM, out-of-bound OPs are not observed, as expected. Finally, we check the summation of the OPs, which related to the production of local viod or overfilling, and the results are shown in Fig. 4.2(d). As mentioned in Section 2, the sum of the OPs should always be $2 - N$, which is -1 in the present case. However, the prediction from CNP-LSTM dose not have this property because it is not enforced in CNP-LSTM model. The summation of the OPs predicted from CNP-LSTM is observed oscillating between 0 and 0.04. On the orther hand, the summation property of the OPs is strictly obeyed with the help of MCBOM.

Figs. 4.3(a) and 4.3(b) compares the mean square error (MSE) between the prediction from CNP-LSTM or CNP-LSTM-MCBOM and the validation data. It seems that the prediction from CNP-LSTM has a smaller error, because minimizing MSE is the only training goal in CNP-LSTM. However, this process does not considering any physical constraints im-

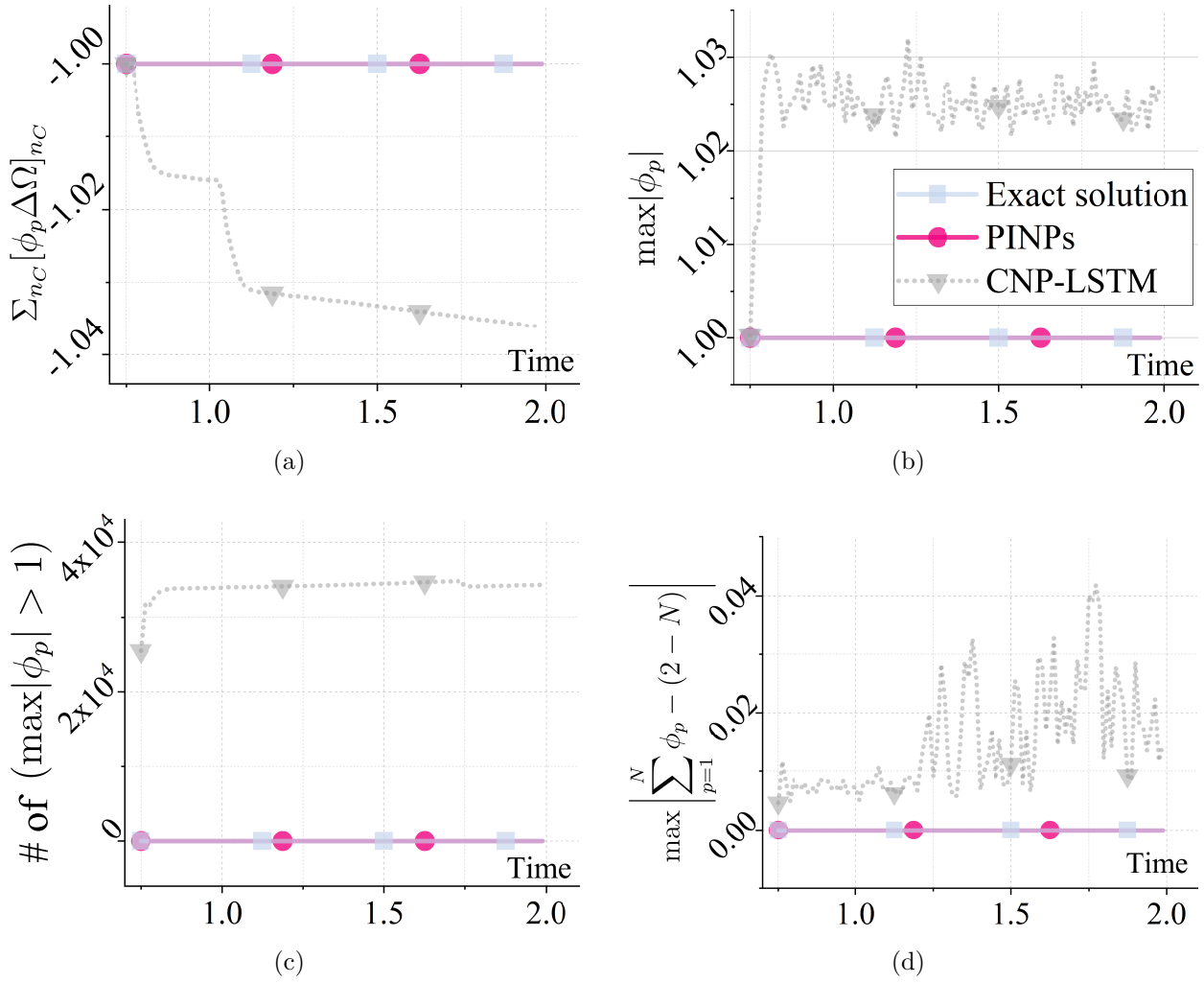

Figure 4.2. a) Validation of mass conservation: the sum of order parameters of a specific phase should be consistent over time ($\sum_{n_C}[\phi_p]_{n_C}$). p is the indices of phase and we show results of the first phase here; b) Validation of phase range constraints: all phase values should fall between [-1, 1] ($\max|\phi_p| \leq 1$); c) Number of values exceed the physical boundary with respect to time t (number of $\max|\phi_p| > 1$); d) Validation of summation for the order parameters: sum of order parameters is restricted to -1 ($\max|\sum_{p=1}^{N}\phi_p - (2-N)|$).

plied in the training data. These physical constraints are explicitly and exactly enforced by MCBOM, which modifies the prediction having minimum MSE. As a result, MSE from the prediction including MCBOM is enlarged to some extent.

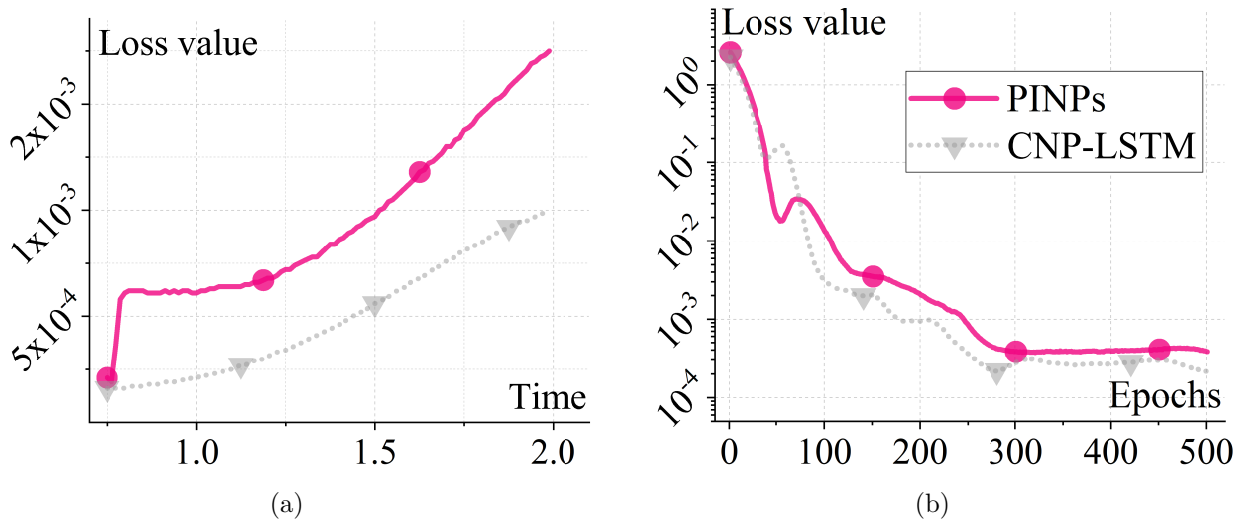

Figure 4.3. (a) Comparison of loss values over time between PINPs and traditional model; (b) Comparison of loss values over epochs between PINPs and traditional model.

Figs. 4.4(a) and 4.4(b) shows the momentum in X and Y directions predicted by PINPs and the tradictional CNP-LSTM. The prediction from PICNP-LSTM is closer to the validation data, and almost not changing with time. On the other hand, the momentum from CNP-LSTM gives a very different behavior from the validation data. We can conclude that PICNP-LSTM can provide a more physically plausible prediction.

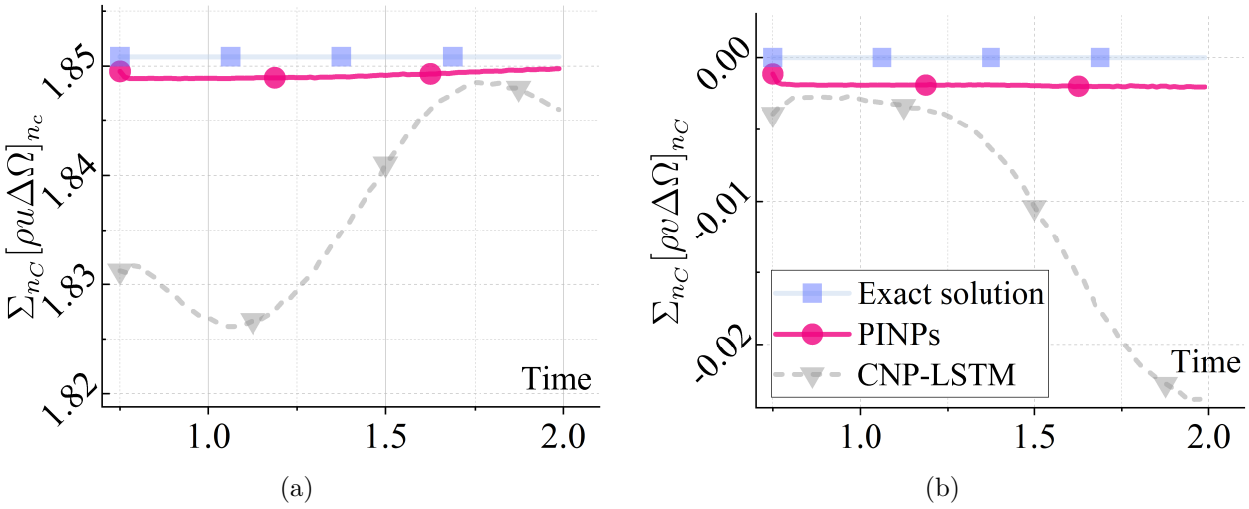

Figure 4.4. (a) Validation of momentum conservation in x direction (in the use of PICNP-LSTM): $\sum_{n_C}[\rho u]_{n_C}$ vs. t; (b) Validation of momentum conservation in y direction: $\sum_{n_C}[\rho v \Delta\Omega]_{n_C}$ vs. t.

4.2 Dam break simulation

Extreme weather associated with rainfall often threatens dam safety. DB problem, as a low-frequency but high-hazard catastrophic factors that may cause significant losses, has received increasing attention and is widely studied. In recent years, the development of computer simulation techniques makes possible modeling the dynamics of DB [65]. Also, DB simulation is a good case of large-density-ratio problem when considering water, oil, and air. The first phase is considered as water, whose density is 998.207kg/m^3 and viscosity is 1.002×10^{-3} Pa $\cdot$ s. The second phase is oil with the density of 557kg/m^3 and viscosity of 9.15×10^{-2} Pa $\cdot$ s. The third phase is considered as air, whose density is set to be 1.204 and viscosity to be 1.75×10^{-5}Pa $\cdot$ s. Initially, a water square is at the left most of the domain, while an oil square is at the right most. The problem is non-dimensionalized by the length of the water square, a density scale 1.204kg/m^3, and an acceleration scale 1 m/s^2. After the non-dimensionalization, the numerical domain of the problem is in a $[4 \times 1]$ box no-slip boundary conditions. The cell size is $\frac{1}{128}$ in depth H and length L, and then a [512×128] cells are created here. The time step is set to be $\Delta t = 5\times\ 10^{-2}$ s. η and M_0 here are set to 0.01 and 10^{-7} here. The flows are set to be stationary when $t = 0$.

The setup for the prediction model is the same as the one in the HSL, except the LSTM layer are set to 128 LSTM cells. The predictive results are shown in Figs. **??**-4.5.

Fig. 4.6(a) shows the mass of the first phase versus time. According to the conservation of mass, total amount of ϕ_p in the entire domain should not change. This is true in the prediction given by CNP-LSTM-MCBOM, but the prediction given by traditional CNP-LSTM model (the grey dashed line with pentagons) violates this principle.

Fig. 4.6(b) shows the maximum absolute value of the OP of the first phase given by the proposed CNP-LSTM-MCBOM model and the traditional CNP-LSTM model. It is obvious that the OP predicted by the traditional model exceeds 1, which violates the boundedness constraints for the OPs. This further indicates the necessity of applying MCBOM, as shown by the red solid line.

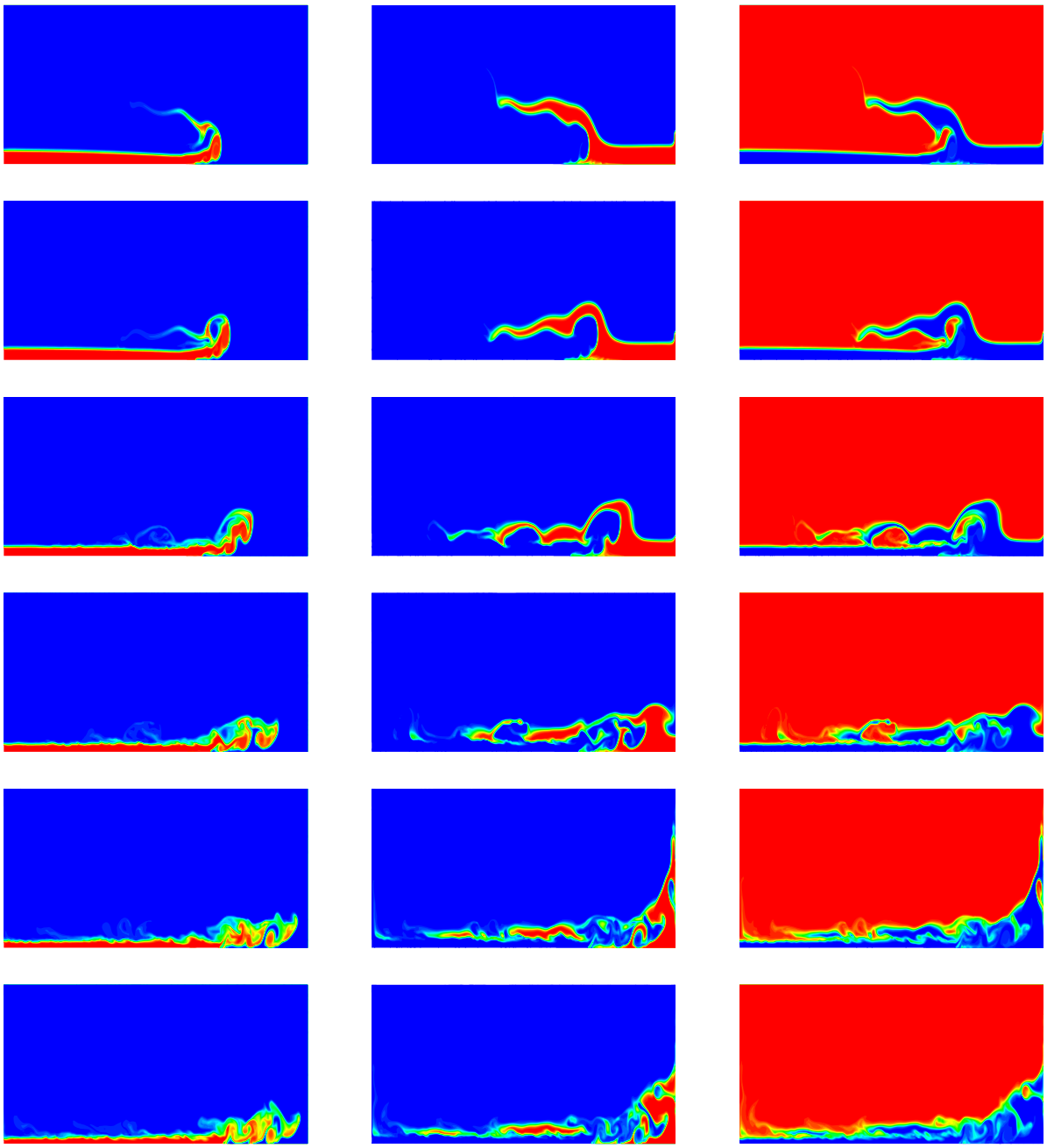

Figure 4.5. Phase values indication of the water, oil and air three-phase flows during the dam break procedure. From left to right, the read ones (phase values close to 1) in figures show the interface of water, oil and air. The blue one (values close to -1) indicates two other phases. From top to bottom, t is from 10s to 20s with 2s increment.

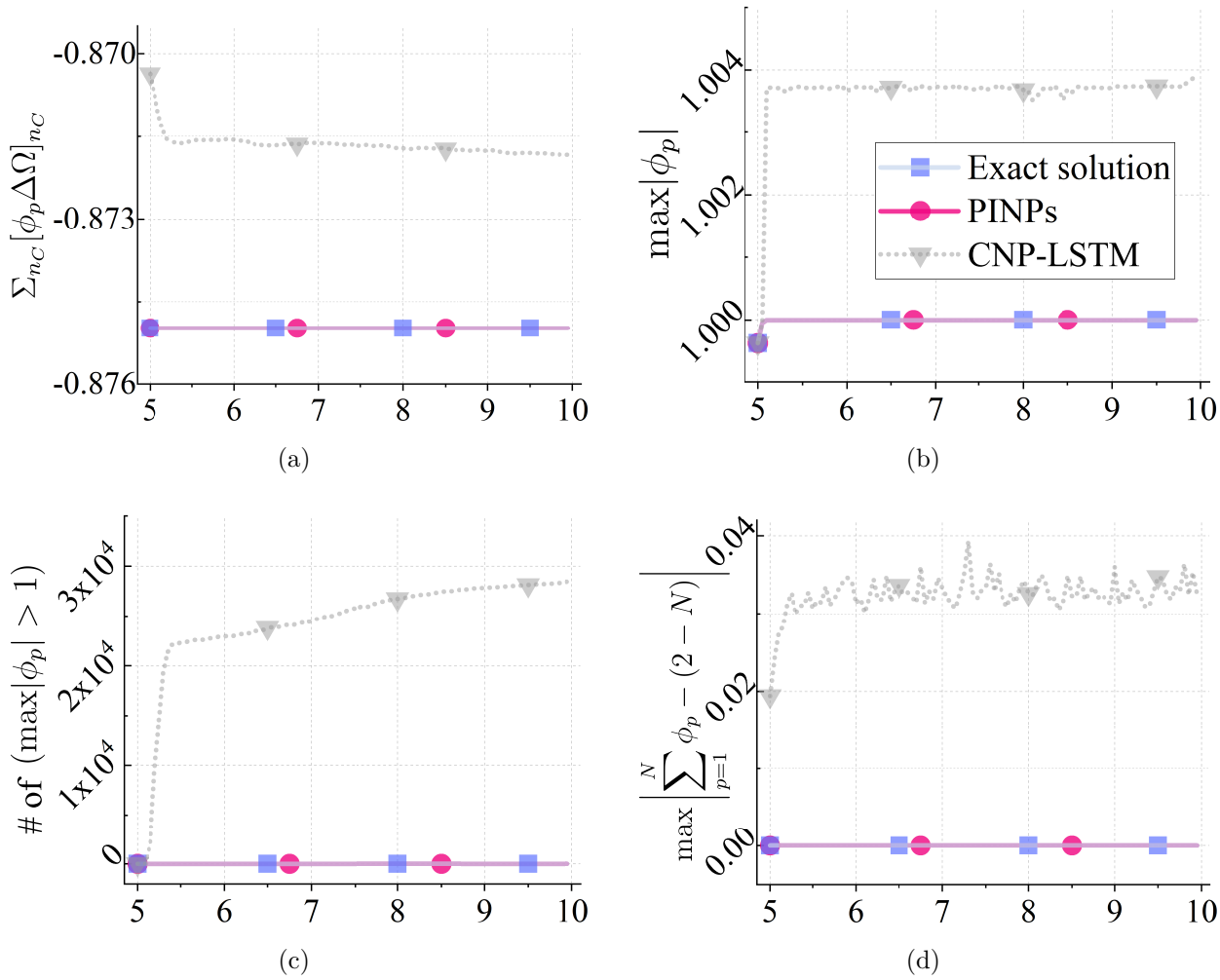

Figure 4.6. a) Validation of mass conservation: the sum of order paramaters of a specific phase should be consistent over time ($\sum_{n_C}[\phi_p]_{n_C}$). p is the indices of phase and we show results of the first phase here; b) Validation of phase range constraints: all phase values should fall between [-1, 1] ($\max|\phi_p| \leq 1$); c) Number of values exceed the physical boundary with respect to time t (number of $\max|\phi_p| > 1$); d) Validation of summation for the order parameters: sum of order parameters is restricted to -1 ($\max|\sum_{p=1}^{N}\phi_p - (2-N)|$).

As shown in Fig. 4.6(c), there are about 28,000 locations having an out-of-bound OP if only CNP-LSTM is used for prediction. Fig. 4.6(d) shows the summation of the OPs. PINPs strictly follow the summation constraint for the OPs, while this is not the case without the help of MCBOM. The results given by CNP-LSTM oscillate between 0.02 and 0.04. Figs. 4.7 shows the prediction of the momentum in the x direction. Results predicted by CNP-LSTM will drop from -0.01 to -0.07. On the other hand, the prediction of the PICNP-LSTM, including a physical-informed penalization of momentum conservation, is much closer to the validation data.

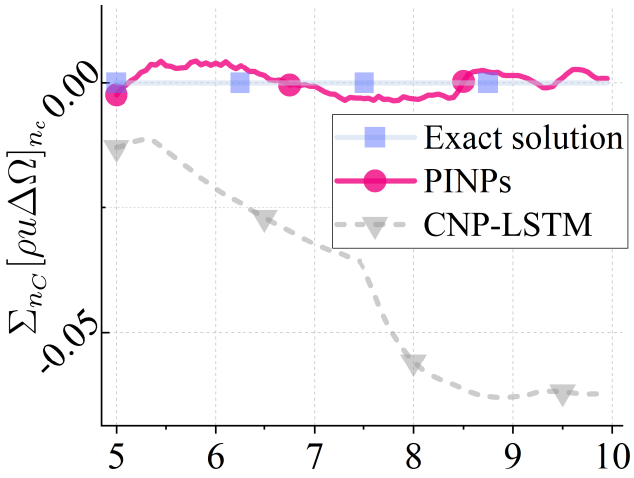

Figure 4.7. Validation of momentum conservation in x direction: $\sum_{nC}[\rho u]_{nC}$ vs. t.

5. CONCLUSION

In the present work, PINPs are developed to predict MPFs. It consists of CNP-LSTM model to predict the OPs, MCBOM to correct the OPs and then obtain the density and viscosity of the fluid mixture, and PICNP-LSTM to predict the flow velocity.

Several important physical constraints in MPFs are considered and enforced in different ways. The physical constraints for the OPs, which are the summation constraint, conservation constraint, boundedness constraint, and the consistency of reduction, are explicitly and exactly enforced by the MCBOM algorithm [66]. MCBOM corrects the OPs, which are the output of CNP-LSTM. We demonstrate that the prediction of the OPs using only CNP-LSTM violates all the physical constraints. As a result, unphysical behaviors, such as mass loss, local voids, or overfilling, appear. However, this issues is successfully addressed after combining CNP-LSTM and MCBOM.

To predict the flow velocity, another NN model is developed, while the momentum conservation is included as a penalization in the loss function of the NNs. This modeled is called PICNP-LSTM. Although the momentum conservation is not strictly satisfied by the prediction from PICNP-LSTM, the behavior of the prediction is much more close to the validation data than the one from the traditional CNP-LSTM without the penalization.

In summary, we consider CNP-LSTM model as the basic structure of the model, and enforce the physical constraints either implicitly by penalization or explicitly by MCBOM. This model is applied to two realistic MPFs and successfully predicts the dynamics. Therefore, the proposed PINPs is a powerful tool for MPF simulation.

REFERENCES

[1] M. Abolhasani and K. F. Jensen, "Oscillatory multiphase flow strategy for chemistry and biology," *Lab on a Chip*, vol. 16, no. 15, pp. 2775–2784, 2016.

[2] J. Yue, "Multiphase flow processing in microreactors combined with heterogeneous catalysis for efficient and sustainable chemical synthesis," *Catalysis Today*, vol. 308, pp. 3–19, 2018.

[3] N. Behari, M. Z. Sheriff, M. A. Rahman, M. Nounou, I. Hassan, and H. Nounou, "Chronic leak detection for single and multiphase flow: A critical review on onshore and offshore subsea and arctic conditions," *Journal of Natural Gas Science and Engineering*, p. 103 460, 2020.

[4] D. Gidaspow and M. S. Bacelos, "Kinetic theory based multiphase flow with experimental verification," *Reviews in Chemical Engineering*, vol. 34, no. 3, pp. 299–318, 2018.

[5] Y. Mi, M. Ishii, and L. Tsoukalas, "Flow regime identification methodology with neural networks and two-phase flow models," *Nuclear engineering and design*, vol. 204, no. 1-3, pp. 87–100, 2001.

[6] D. Valero and D. B. Bung, "Artificial neural networks and pattern recognition for air-water flow velocity estimation using a single-tip optical fibre probe," *Journal of Hydro-Environment Research*, vol. 19, pp. 150–159, 2018.

[7] S. Rashid, A. Ghamartale, J. Abbasi, H. Darvish, and A. Tatar, "Prediction of critical multiphase flow through chokes by using a rigorous artificial neural network method," *Flow Measurement and Instrumentation*, vol. 69, p. 101 579, 2019.

[8] P. L. Serra, P. H. Masotti, M. S. Rocha, D. A. de Andrade, W. M. Torres, and R. N. de Mesquita, "Two-phase flow void fraction estimation based on bubble image segmentation using randomized hough transform with neural network (rhtn)," *Progress in Nuclear Energy*, vol. 118, p. 103 133, 2020.

[9] Y. Yan, L. Wang, T. Wang, X. Wang, Y. Hu, and Q. Duan, "Application of soft computing techniques to multiphase flow measurement: A review," *Flow Measurement and Instrumentation*, vol. 60, pp. 30–43, 2018.

[10] T. A. AL-Qutami, R. Ibrahim, I. Ismail, and M. A. Ishak, "Virtual multiphase flow metering using diverse neural network ensemble and adaptive simulated annealing," *Expert Systems with Applications*, vol. 93, pp. 72–85, 2018.

[11] S. Mo, Y. Zhu, N. Zabaras, X. Shi, and J. Wu, "Deep convolutional encoder-decoder networks for uncertainty quantification of dynamic multiphase flow in heterogeneous media," *Water Resources Research*, vol. 55, no. 1, pp. 703–728, 2019.

[12] J. N. Kani and A. H. Elsheikh, "Reduced-order modeling of subsurface multi-phase flow models using deep residual recurrent neural networks," *Transport in Porous Media*, vol. 126, no. 3, pp. 713–741, 2019.

[13] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

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

[15] Z. Huang, G. Lin, and A. M. Ardekani, "A consistent and conservative volume distribution algorithm and its applications to multiphase flows using phase-field models," *arXiv preprint arXiv:2010.01738*, 2020.

[16] J. Willard, X. Jia, S. Xu, M. Steinbach, and V. Kumar, "Integrating physics-based modeling with machine learning: A survey," *arXiv:2003.04919*, 2020.

[17] G. Pang, L. Lu, and G. E. Karniadakis, "Fpinns: Fractional physics-informed neural networks," *SIAM Journal on Scientific Computing*, vol. 41, no. 4, A2603–A2626, 2019.

[18] Y. Yang and P. Perdikaris, "Adversarial uncertainty quantification in physics-informed neural networks," *Journal of Computational Physics*, vol. 394, pp. 136–152, 2019.

[19] S. Goswami, C. Anitescu, S. Chakraborty, and T. Rabczuk, "Transfer learning enhanced physics informed neural network for phase-field modeling of fracture," *Theoretical and Applied Fracture Mechanics*, vol. 106, p. 102 447, 2020.

[20] A. M. Tartakovsky, C. O. Marrero, P. Perdikaris, G. D. Tartakovsky, and D. Barajas-Solano, "Learning parameters and constitutive relationships with physics informed deep neural networks," *arXiv preprint arXiv:1808.03398*, 2018.

[21] X. Meng, Z. Li, D. Zhang, and G. E. Karniadakis, "Ppinn: Parareal physics-informed neural network for time-dependent pdes," *Computer Methods in Applied Mechanics and Engineering*, vol. 370, p. 113 250, 2020.

[22] Y. Zhu, N. Zabaras, P.-S. Koutsourelakis, and P. Perdikaris, "Physics-constrained deep learning for high-dimensional surrogate modeling and uncertainty quantification without labeled data," *Journal of Computational Physics*, vol. 394, pp. 56–81, 2019, ISSN: 0021-9991.

[23] A. Karpatne, W. Watkins, J. Read, and V. Kumar, "Physics-guided neural networks (pgnn): An application in lake temperature modeling," *arXiv:1710.11431*, 2017.

[24] X. Jia, J. Willard, A. Karpatne, J. Read, J. Zwart, M. Steinbach, and V. Kumar, "Physics guided rnns for modeling dynamical systems: A case study in simulating lake temperature profiles," in *Proceedings of the 2019 SIAM International Conference on Data Mining*, SIAM, 2019, pp. 558–566.

[25] J. S. Read, X. Jia, J. Willard, A. P. Appling, J. A. Zwart, S. K. Oliver, A. Karpatne, G. J. Hansen, P. C. Hanson, W. Watkins, *et al.*, "Process-guided deep learning predictions of lake water temperature," *Water Resources Research*, vol. 55, no. 11, pp. 9173–9190, 2019.

[26] N. Geneva and N. Zabaras, "Modeling the dynamics of pde systems with physics-constrained deep auto-regressive networks," *Journal of Computational Physics*, vol. 403, p. 109 056, 2020.

[27] M. Raissi, Z. Wang, M. S. Triantafyllou, and G. E. Karniadakis, "Deep learning of vortex-induced vibrations," *Journal of Fluid Mechanics*, vol. 861, pp. 119–137, 2019.

[28] A. Kahana, E. Turkel, S. Dekel, and D. Givoli, "Obstacle segmentation based on the wave equation and deep learning," *Journal of Computational Physics*, p. 109 458, 2020.

[29] L. Yang, S. Treichler, T. Kurth, K. Fischer, D. Barajas-Solano, J. Romero, V. Churavy, A. Tartakovsky, M. Houston, M. Prabhat, *et al.*, "Highly-ccalable, physics-informed gans for learning solutions of stochastic pdes," in *2019 IEEE/ACM Third Workshop on Deep Learning on Supercomputers (DLS)*, IEEE, 2019, pp. 1–11.

[30] L. Lu, P. Jin, G. Pang, Z. Zhang, and G. E. Karniadakis, "Learning nonlinear operators via deeponet based on the universal approximation theorem of operators," *Nature Machine Intelligence*, vol. 3, no. 3, pp. 218–229, 2021.

[31] Z. Li, N. Kovachki, K. Azizzadenesheli, B. Liu, K. Bhattacharya, A. Stuart, and A. Anandkumar, "Neural operator: Graph kernel network for partial differential equations," 2020. arXiv: 2003.03485 [cs.LG].

[32] Z. Li, N. Kovachki, K. Azizzadenesheli, B. Liu, K. Bhattacharya, A. Stuart, and A. Anandkumar, "Fourier neural operator for parametric partial differential equations," 2020. arXiv: 2010.08895 [cs.LG].

[33] Z. Huang, G. Lin, and A. M. Ardekani, "A consistent and conservative model and its scheme for n-phase-m-component incompressible flows," *arXiv preprint arXiv:2101.04252*, 2021.

[34] S. Dong, "Multiphase flows of n immiscible incompressible fluids: A reduction-consistent and thermodynamically-consistent formulation and associated algorithm," *Journal of Computational Physics*, vol. 361, pp. 1–49, 2018.

[35] C. Olah, "Understanding lstm networks," 2015.

[36] S. Hochreiter, "The vanishing gradient problem during learning recurrent neural nets and problem solutions," *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 6, no. 02, pp. 107–116, 1998.

[37] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[38] M. Sundermeyer, H. Ney, and R. Schl**ü**ter, "From feedforward to recurrent lstm neural networks for language modeling," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 23, no. 3, pp. 517–529, 2015.

[39] Z. Huang, W. Xu, and K. Yu, "Bidirectional lstm-crf models for sequence tagging," *arXiv preprint arXiv:1508.01991*, 2015.

[40] K. Greff, R. K. Srivastava, J. Koutník, B. R. Steunebrink, and J. Schmidhuber, "Lstm: A search space odyssey," *IEEE transactions on neural networks and learning systems*, vol. 28, no. 10, pp. 2222–2232, 2016.

[41] S. Xingjian, Z. Chen, H. Wang, D.-Y. Yeung, W.-K. Wong, and W.-c. Woo, "Convolutional lstm network: A machine learning approach for precipitation nowcasting," in *Advances in neural information processing systems*, 2015, pp. 802–810.

[42] F. J. Ordóñez and D. Roggen, "Deep convolutional and lstm recurrent neural networks for multimodal wearable activity recognition," *Sensors*, vol. 16, no. 1, p. 115, 2016.

[43] A. Graves and J. Schmidhuber, "Framewise phoneme classification with bidirectional lstm and other neural network architectures," *Neural networks*, vol. 18, no. 5-6, pp. 602–610, 2005.

[44] M. Cornia, L. Baraldi, G. Serra, and R. Cucchiara, "Predicting Human Eye Fixations via an LSTM-based Saliency Attentive Model," *IEEE Transactions on Image Processing*, vol. 27, no. 10, pp. 5142–5154, 2018.

[45] M. Al-Shedivat, A. G. Wilson, Y. Saatchi, Z. Hu, and E. P. Xing, "Learning scalable deep kernels with recurrent structure," *Journal of Machine Learning Research*, 2017.

[46] M. Garnelo, D. Rosenbaum, C. J. Maddison, T. Ramalho, D. Saxton, M. Shanahan, Y. W. Teh, D. J. Rezende, and S. Eslami, "Conditional neural processes," *arXiv preprint arXiv:1807.01613*, 2018.

[47] Z. Huang, G. Lin, and A. M. Ardekani, "Consistent and conservative scheme for incompressible two-phase flows using the conservative allen-cahn model," *Journal of Computational Physics*, vol. 420, p. 109 718, 2020.

[48] F. Boyer and S. Minjeaud, "Hierarchy of consistent n-component cahn–hilliard systems," *Mathematical Models and Methods in Applied Sciences*, vol. 24, no. 14, pp. 2885–2928, 2014.

[49] Z. Huang, G. Lin, and A. M. Ardekani, "Consistent and conservative scheme for incompressible two-phase flows using the conservative allen-cahn model," *Journal of Computational Physics*, vol. 420, p. 109 718, 2020.

[50] J. D. Hamilton, "Time series analysis," vol. 2, 1994.

[51] M. Weber, M. Alexa, and W. Müller, "Visualizing time-series on spirals.," in *Infovis*, vol. 1, 2001, pp. 7–14.

[52] N. H. Packard, J. P. Crutchfield, J. D. Farmer, and R. S. Shaw, "Geometry from a time series," *Physical review letters*, vol. 45, no. 9, p. 712, 1980.

[53] J. D. Farmer and J. J. Sidorowich, "Predicting chaotic time series," *Physical review letters*, vol. 59, no. 8, p. 845, 1987.

[54] T. C. Mills and T. C. Mills, "Time series techniques for economists," 1991.

[55] J. T. Connor, R. D. Martin, and L. E. Atlas, "Recurrent neural networks and robust time series prediction," *IEEE transactions on neural networks*, vol. 5, no. 2, pp. 240–254, 1994.

[56] D. M. Smith, S. Cusack, A. W. Colman, C. K. Folland, G. R. Harris, and J. M. Murphy, "Improved surface temperature prediction for the coming decade from a global climate model," *science*, vol. 317, no. 5839, pp. 796–799, 2007.

[57] J. Yang, M. N. Nguyen, P. P. San, X. L. Li, and S. Krishnaswamy, "Deep convolutional neural networks on multichannel time series for human activity recognition," in *Twenty-Fourth International Joint Conference on Artificial Intelligence*, 2015.

[58] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1026–1034.

[59] M. Garnelo, J. Schwarz, D. Rosenbaum, F. Viola, D. J. Rezende, S. Eslami, and Y. W. Teh, "Neural processes," *arXiv preprint arXiv:1807.01622*, 2018.

[60] H. Kim, A. Mnih, J. Schwarz, M. Garnelo, A. Eslami, D. Rosenbaum, O. Vinyals, and Y. W. Teh, "Attentive neural processes," *arXiv preprint arXiv:1901.05761*, 2019.

[61] R. A. Horn and C. R. Johnson, *Matrix analysis*. Cambridge university press, 2012.

[62] F. S. Pereira, L. Eça, G. Vaz, and S. S. Girimaji, "Challenges in scale-resolving simulations of turbulent wake flows with coherent structures," *Journal of Computational Physics*, vol. 363, pp. 98–115, 2018.

[63] S. Xu, Z. Xu, O. V. Kim, R. I. Litvinov, J. W. Weisel, and M. Alber, "Model predictions of deformation, embolization and permeability of partially obstructive blood clots under variable shear flow," *Journal of the Royal Society Interface*, vol. 14, no. 136, p. 20 170 441, 2017.

[64] K. H. Christensen, Ø. Breivik, K.-F. Dagestad, J. Röhrs, and B. Ward, "Short-term predictions of oceanic drift," *Oceanography*, vol. 31, no. 3, pp. 59–67, 2018.

[65] Z. Huang, G. Lin, and A. M. Ardekani, "Consistent, essentially conservative and balanced-force phase-field method to model incompressible two-phase flows," *Journal of Computational Physics*, vol. 406, p. 109 192, 2020.

[66] Z. Huang, G. Lin, and A. M. Ardekani, "A consistent and conservative phase-field model for thermo-gas-liquid-solid flows including liquid-solid phase change," *arXiv preprint arXiv:2102.06863*, 2021.