THESIS FOR THE DEGREE OF LICENTIATE OF ENGINEERING

Physics-informed neural networks with hard and soft boundary conditions for problems in fluid dynamics

Mohammad Sheikholeslami

Department of Mechanics and Maritime Sciences
Chalmers University of Technology
Gothenburg, Sweden, 2025

Physics-informed neural networks with hard and soft boundary conditions for problems in fluid dynamics

Mohammad Sheikholeslami

Department of Mechanics and Maritime Sciences
Divisions of Marine Technology - Fluid Dynamics
Chalmers University of Technology
SE-412 96 Göteborg,
Sweden
Phone: +46(0)31 772 6067

Printed by Chalmers Digitaltryck,
Gothenburg, Sweden 2025.

Physics-informed neural networks with hard and soft boundary conditions for problems in fluid dynamics

Mohammad Sheikholeslami

Department of Mechanics and Maritime Sciences
Chalmers University of Technology

Abstract

This thesis investigates the use of Physics-Informed Neural Networks (PINNs) for solving forward problems in fluid dynamics through two classical cases: linear waves and lid-driven cavity problem. The linear wave case, based on potential flow theory, involves periodic waves over a flat bed and serves to examine how PINNs handle periodic boundary conditions and smooth solutions. The lid-driven cavity problem, a benchmark for viscous incompressible flow, introduces sharp gradients and corner singularities, challenging the network's ability to handle discontinuous boundary conditions. Together, these cases evaluate PINNs across both smooth and complex flow regimes.

For the linear wave problem, both hard and soft strategies were used to enforce periodic boundary conditions (PBCs). The vanilla PINN predicted the velocity field with 2.31% error. A periodic input achieved a 0.16% velocity error, while soft constraints performed slightly better (0.10%) but did not fully enforce periodicity. Hard enforcement of the kinematic bottom boundary condition (KBBC) using trial functions satisfied the constraint at machine precision, though at the cost of increased velocity error (2.36%). The network also predicted the angular frequency ω with 0.03% error and maintained velocity errors below 0.2% without labeled data.

In the cavity flow case, a baseline PINN with soft boundary enforcement performed well across most of the domain but struggled near the top corners due to singularities. To improve accuracy, two strategies were tested: trial functions for hard boundary conditions and spatially varying loss weights. Hard Constraints 1, which left the top boundary soft, led to a marginal improvement, reducing MAE(v) by about 8%, while Hard Constraints 2, which enforced the top boundary with a trial function, worsened the results. Weighting strategies were more effective: applying ω_{mw} reduced MAE(u) by 63% and MAE(v) by 68%, and using both ω_F and ω_{mw} yielded the best accuracy, with reductions of 81% in MAE(u) and 78% in MAE(v). These findings highlight the challenge of handling singularities.

Overall, the study provides practical insights and quantitative benchmarks to guide the design and training of PINNs for fluid dynamics problems.

Keywords

Physics-informed neural network, PINN, linear waves, lid-driven cavity, periodic boundary condition, soft constraints, hard constraints

List of appended papers

In each of the four attached papers, the author contributed substantially, providing the main ideas and refining them in collaboration with the co-authors. Each manuscript was written by the author, with final reviews and editing completed in collaboration with the co-authors.

[Paper I] **M. Sheikholeslami**, S. Salehi, W. Mao, A. Eslamdoost, and H. Nilsson. *Physics-Informed Neural Networks for Modeling Linear Waves. In proceedings of the 43rd International Conference on Ocean, Offshore and Arctic Engineering*, Singapore, 2024. DOI: `https://doi.org/10.1115/OMAE2024-125048`

[Paper II] **M. Sheikholeslami**, S. Salehi, W. Mao, A. Eslamdoost, and H. Nilsson. *Physics-informed neural networks with hard and soft boundary conditions for linear free surface waves. Manuscript submitted for Journal Publication.*

[Paper III] **M. Sheikholeslami**, S. Salehi, W. Mao, A. Eslamdoost, and H. Nilsson. *Comparative Evaluation of Periodic Boundary Condition Approaches in PINNs. Accepted in the 1st International Symposium AI and Fluid Mechanics*, Greece, 2025.

[Paper IV] **M. Sheikholeslami**, S. Salehi, W. Mao, A. Eslamdoost, and H. Nilsson. *Addressing Corner Singularities in Physics-Informed Neural Network Solutions of the Lid-Driven Cavity Problem. Manuscript to be submitted for Journal Publication.*

CRediT Authorship Contribution Statement

Mohammad Sheikholeslami (All papers): Conceptualisation, Validation, Methodology, Investigation, Visualisation, Writing–Original draft.

Saeed Salehi (All papers): Conceptualisation, Validation, Methodology, Visualisation, Funding acquisition, Supervision, Project administration, Writing-review and editing.

Wengang Mao (All papers): Conceptualisation, Validation, Methodology, Visualisation, Funding acquisition, Supervision, Project administration, Writing-review and editing.

Arash Eslamdoost (All papers): Conceptualisation, Validation, Methodology, Visualisation, Funding acquisition, Supervision, Project administration, Writing-review and editing.

Håkan Nilsson (All papers): Conceptualisation, Validation, Methodology, Visualisation, Funding acquisition, Supervision, Project administration, Writing-review and editing.

Acknowledgment

This project was carried out within the Divisions of Marine Technology and Fluid Dynamics at the Department of Mechanics and Maritime Sciences, Chalmers University of Technology. It was financially supported by the department and conducted as part of the Swedish Centre for Sustainable Hydropower (SVC). SVC is a national research program initiated by the Swedish Energy Agency, Energiforsk, and Svenska kraftnät, in collaboration with Luleå University of Technology, Uppsala University, KTH Royal Institute of Technology, Chalmers University of Technology, Karlstad University, Umeå University, and Lund University.

First and foremost, I would like to express my sincere gratitude to my main supervisor, Prof. Wengang Mao, for his constant support and guidance throughout this work. His encouragement and advice have been invaluable at every stage of this journey. I am also deeply thankful to Prof. Håkan Nilsson, my co-supervisor and examiner, for his generous investment of time, insightful ideas, and constructive feedback, which helped improve both the content and structure of my work. My appreciation extends to Dr. Saeed Salehi, whose deep understanding of technical details and prompt feedback significantly accelerated my progress and contributed to the efficiency of this thesis. Finally, I would like to thank Prof. Arash Eslamdoost for his consistent support and thoughtful comments, which meaningfully shaped the development of this thesis. I'm grateful to Prof. Jonas Ringsberg for being such a supportive and approachable manager throughout this period. His encouragement and regular updates were always appreciated and made a real difference.

I am fortunate to have had wonderful colleagues who made my time at the department both productive and memorable. Rui, Daniel, Stephan, Qais, Mohsen, Malik, and Azim, who also formed our informal swimming club, have been great colleagues and teammates in and out of the pool. Thank you for making the swimming sessions more about laughs than laps and for always keeping the spirits high, even when the water was cold and the deadlines were near. I am also grateful to my colleagues and friends Mehmet, Xinyuan, Xiao, Chengqian, Yuhan, Negin, Heng, Oweis, Chi, Seoyun, Lucile, Mohammad, Erik, Henrik, Martina, Kourosh, Sucheth, Jaseung, Ali (senior), Matheus, Ali (junior), Alexander, and Amin for creating a supportive and enjoyable environment. I would also like to thank my former officemate Jiabing, and I wish him all the best in his future endeavors.

Contents

Abstract i

List of Publications iii

Acknowledgement v

Nomenclature ix

1 Introduction 1
- 1.1 Background and motivation . . . 1
- 1.2 A literature review on PINNs for fluid dynamics . . . 3
  - 1.2.1 Laminar flows . . . 4
  - 1.2.2 High-speed and turbulent flows . . . 4
  - 1.2.3 Complex flow phenomena and advanced applications . . . 5
  - 1.2.4 Free surface flows . . . 5
- 1.3 Objectives, goals and contributions . . . 6

2 Mathematical description of the linear wave and lid-driven cavity problem 9
- 2.1 Linear wave theory: modeling free surface dynamics . . . 9
  - 2.1.1 Governing equation and assumptions . . . 9
  - 2.1.2 Boundary conditions in their original form . . . 10
  - 2.1.3 Linearization of boundary conditions . . . 11
  - 2.1.4 Analytical solution . . . 11
- 2.2 Lid-driven cavity problem . . . 12
  - 2.2.1 Governing equation . . . 12
  - 2.2.2 Boundary conditions . . . 13

3 Methodology 15
- 3.1 Physics-informed neural network . . . 15
  - 3.1.1 Solving direct and inverse problems . . . 17
  - 3.1.2 Regularizing and refinement of the data . . . 17
  - 3.1.3 Data provision with lower error . . . 17
  - 3.1.4 Integrating data and physics in modeling . . . 18
- 3.2 PINN for the linear wave problem . . . 18
  - 3.2.1 Periodic boundary condition . . . 19

3.2.2 Trial functions 21
3.2.3 Finding an unknown in the physics 24
3.3 PINN formulation for the lid-driven cavity flow 24
3.3.1 Hard constraints through trial functions 25
3.3.2 Spatially varying weights 25

4 Summary of appended papers 27
4.1 Paper I 28
4.1.1 Objectives and contributions 28
4.1.2 Methods 28
4.1.3 Results and discussion 28
4.2 Paper II 31
4.2.1 Objectives and contributions 31
4.2.2 Methods 31
4.2.3 Results and discussion 31
4.3 Paper III 35
4.3.1 Objectives and contributions 35
4.3.2 Methods 35
4.3.3 Results and discussion 37
4.4 Paper IV 38
4.4.1 Objectives and contributions 38
4.4.2 Methods 38
4.4.3 Results and discussion 39

Bibliography 41

Nomenclature

Linear waves

x	Spatial coordinate in x-direction [m]
z	Spatial coordinate in z-direction [m]
t	Time [s]
$\phi(x, z, t)$	Velocity potential [m^2/s]
u	Velocity in x-direction [m/s]
$\mathbf{u}$	Velocity field [m/s]
w	Velocity in z-direction [m/s]
η	Surface elevation [m]
A	Wave amplitude [m]
h	Water depth [m]
L	Domain length or wavelength [m]
k	Wave number [rad/m]
ω	Wave angular frequency [rad/s]
g	Gravitational acceleration [m/s^2]
λ	Wavelength [m]
Ω	Computational domain
$\partial\Omega$	Boundary of computational domain
θ	Trainable parameters of the neural network
AD	Automatic differentiation
$\mathcal{N}_\tau[\cdot]$	Governing differential operator
$\mathcal{B}_\tau[\cdot]$	Boundary operator
L_T	Total loss function
L_{GE}	Loss for governing equation
L_{KFSBC}	Loss for kinematic free surface boundary condition
L_{DFSBC}	Loss for dynamic free surface boundary condition
L_{KBBC}	Loss for kinematic bottom boundary condition
$L_{\phi_{x\text{-}per}}$	Periodicity loss in x-direction
Adam	Adaptive Moment Estimation optimizer
L-BFGS	Limited-memory BFGS optimizer
MSE	Mean Squared Error
BC	Boundary Condition
PBC	Periodic Boundary Condition

Lid-driven cavity problem

x	Spatial coordinate in x-direction [m]
y	Spatial coordinate in y-direction [m]
z	Collocation coordinate pair $z = (x, y)$
u	Velocity in x-direction [m/s]
v	Velocity in y-direction [m/s]
p	Pressure [Pa]
U	Top wall velocity [m/s]
H	Cavity height [m]
ν	Kinematic viscosity [m^2/s]
Re	Reynolds number $= UH/\nu$
$\mathcal{F}$	PDE residual operator
$\hat{\mathcal{P}}_\theta(z_i)$	PINN output at collocation point parameterized by θ
N_C	Number of collocation points in the domain
$N_{B,mw}$	Number of boundary points on the moving wall
$N_{B,nmw}$	Number of boundary points on non-moving walls
$\mathcal{B}_{Ref}$	Reference boundary values for velocity [m/s]
$\mathcal{B}_{mw}$	Moving wall boundary operator
$\mathcal{B}_{nmw}$	Non-moving wall boundary operator
$\mathcal{L}_F$	Loss for governing equations
$\mathcal{L}_B$	Loss for boundary conditions
$\mathcal{L}_{Bmw}$	Loss for the moving wall boundary
$\mathcal{L}_{Bnmw}$	Loss for the non-moving wall boundaries
$\mathcal{L}_{momX}$	Loss for x-momentum equation
$\mathcal{L}_{momY}$	Loss for y-momentum equation
$\mathcal{L}_{cont}$	Loss for continuity equation
$\mathcal{L}_{top\ u}$	Loss for u at the top wall
$\mathcal{L}_{wall\ u}$	Loss for u on the side and bottom walls
$\mathcal{L}_{wall\ v}$	Loss for v on all walls
ω_F	Spatial weight for PDE residual loss
ω_{mw}	Spatial weight for moving wall loss
MAE	Mean Absolute Error
MSE	Mean Squared Error
RMSE	Root Mean Squared Error

Chapter 1

Introduction

1.1 Background and motivation

Fluid dynamics is a fundamental branch of classical physics that describes the motion of liquids and gases. It plays a critical role in understanding and predicting a wide range of physical phenomena in both nature and technology. In engineering, it is essential for the design of ships, offshore structures, airplanes, wind turbines, and cooling systems. In natural systems, fluid dynamics governs the behavior of ocean currents, atmospheric circulation, and river flows.

Accurately modeling such systems is of practical importance for improving performance, safety, and efficiency. However, due to the complexity of fluid behavior, especially in three dimensions and over time, the development of reliable computational tools remains a central challenge in modern engineering.

The physics of fluid dynamics is mathematically described using partial differential equations (PDEs), which express fundamental conservation laws [1]. These include the conservation of mass, momentum, and in some cases energy. Depending on the assumptions made about the fluid properties and the nature of the flow, the mathematical formulation of these laws can vary significantly in complexity.

A common starting point is potential flow theory, which assumes that the fluid is inviscid, incompressible, and irrotational. Under these conditions, the velocity field can be expressed as the gradient of a scalar potential function ϕ, and the governing equation becomes the Laplace equation, given by

$$\nabla^2 \phi = 0. \tag{1.1}$$

Potential theory offers a significant simplification and is widely used for problems such as wave propagation in deep water and aerodynamics around slender bodies. Although it captures important features of flow fields, it neglects viscous effects and vorticity, making it inadequate for describing boundary layers, turbulence, and many engineering flows.

To capture these effects, the model must be extended to include viscosity and nonlinear interactions. This leads to the general formulation of fluid motion

through the incompressible Navier Stokes equations. These equations consist of the continuity equation:

$$\nabla \cdot \mathbf{u} = 0, \tag{1.2}$$

which ensures mass conservation, and the momentum equation:

$$\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} = -\nabla p + \nu \nabla^2 \mathbf{u}, \tag{1.3}$$

which represents the balance of inertial, pressure, and viscous forces in the fluid. Here, $\mathbf{u}$ is the velocity field, p is the pressure, and ν is the kinematic viscosity.

These equations are nonlinear and coupled, making them difficult to solve analytically in most practical cases. As a result, various numerical techniques have been developed to approximate their solutions [2].

The transition from potential flow to the full Navier Stokes equations reflects a hierarchy of fluid models. At one end of this spectrum are inviscid and linearized models, such as linear wave theory, which are tractable and often sufficient for understanding global flow patterns in simple settings. At the other end are the fully nonlinear and viscous models required for capturing complex behaviors like vortex shedding, turbulence, and flow separation.

In linear wave theory, the governing equations consist of the Laplace equation for the velocity potential, accompanied by linearized boundary conditions at the free surface and solid boundaries. These include a kinematic boundary condition (representing conservation of mass at the free surface) and a dynamic condition derived from the unsteady Bernoulli equation (representing momentum balance). Together, these equations form a complete linear model for small-amplitude surface waves.

In contrast, for internal recirculating flows such as that in a lid-driven cavity, the inviscid and irrotational assumptions break down. The full incompressible Navier Stokes equations must be solved to capture the primary and secondary vortices that arise due to the interaction between the lid motion and the fluid viscosity.

This thesis investigates both regimes: the linear wave problem as a simplified, inviscid, irrotational model, and the lid-driven cavity as a viscous, nonlinear flow described by the general form of the Navier Stokes equations.

Since analytical solutions are rarely available, numerical methods have been developed to solve PDEs that arise in fluid mechanics. The most widely used methods include:

- **Finite Difference Methods (FDM):** These methods approximate derivatives using differences between function values at discrete points on a structured grid. They are simple to implement and computationally efficient for regular domains.

- **Finite Volume Methods (FVM):** FVMs apply the conservation laws directly to control volumes in the computational domain. They are widely used in engineering applications due to their ability to handle complex geometries and ensure local conservation.

- **Finite Element Methods (FEM):** FEMs use variational formulations and piecewise-defined basis functions, making them well-suited for unstructured meshes and problems with complex boundaries.
- **Spectral Methods:** These use global basis functions, such as Fourier or Chebyshev polynomials, to represent the solution. They offer very high accuracy for smooth problems but are less suitable for localized phenomena.

Each method has its strengths and limitations, and their selection depends on the nature of the problem, domain geometry, and required accuracy [3], [4].

In recent years, Physics-Informed Neural Networks (PINNs), introduced by Raissi et al. [5], have emerged as a promising framework that integrates physical laws into neural network training. Rather than relying exclusively on data, PINNs incorporate governing differential equations and boundary conditions into the loss function, enabling the model to produce solutions that adhere to the underlying physics. This framework provides a powerful alternative to traditional numerical solvers. PINNs are mesh-free, which allows them to handle irregular geometries and scattered data. They are also capable of solving both forward and inverse problems, often within a single framework. Because the physics is encoded in the loss function, they can make use of sparse or incomplete data, offering a promising route for simulation and data assimilation in complex systems.

Despite the growing interest in PINNs, their success in fluid dynamics modeling remains problem-dependent. The following literature review focuses on recent applications of PINNs in fluid dynamics, highlighting the types of problems addressed, the modeling strategies adopted, and the key findings from those studies.

1.2 A literature review on PINNs for fluid dynamics

One of the most prominent areas of application for PINNs is fluid dynamics, where they have been used to solve the Navier Stokes equations governing incompressible and compressible flows. PINNs have shown success in modeling laminar flows and are increasingly being adapted to simulate turbulent flow regimes. The foundational work by Raissi et al. [5] introduced PINNs as a framework capable of solving forward and inverse problems involving nonlinear PDEs, with applications demonstrated in fluid dynamics contexts. Their study showcased the ability of PINNs to learn solutions to the Navier Stokes equations, highlighting their potential in fluid mechanics.

Cai et al. [6] provided a comprehensive overview of how PINNs have been applied and adapted for a wide range of fluid mechanics problems, including vortex dynamics, flow around obstacles, and high-speed aerodynamics. Similarly, the recent review by Toscano et al. [7] discusses the progression from classical PINNs to more advanced variants like PIKANs, offering insights into their evolving capabilities in fluid dynamics applications.

It is important to note that the classification of PINN applications into categories such as laminar, turbulent, or free surface flows is not mutually exclusive. Many studies combine aspects of multiple flow regimes or incorporate hybrid modeling strategies depending on the specific physical and computational requirements of the problem.

1.2.1 Laminar flows

The applications of PINNs to laminar flows demonstrated their ability to infer hidden variables from limited data. Vortex-induced vibrations were modeled using sparse velocity field measurements and no pressure data, showing that PINNs can reconstruct the entire flow field under severe data constraints [8]. Similarly, Lou et al. [9] used a Boltzmann-based PINN formulation to solve forward and inverse cavity flows, while in another study, a conservative PINN (cPINN) was introduced using domain decomposition to improve the enforcement of conservation laws in laminar cavity problems [10]. Zhang et al. [11] derived flow fields from temperature data in laminar settings like channel flow and cavity flow. Jin et al. [12] developed NSFnets to solve benchmark problems by optimizing network architecture and loss weighting strategies.

Alternative approaches have also been proposed. Karnakov, Litvinov and Koumoutsakos [13] introduced a discrete loss optimization framework (ODIL), which avoids neural networks altogether and demonstrated superior performance over PINNs in both forward and inverse cavity flow simulations. Wang et al. [14] highlighted issues of solution multiplicity and instability in PINNs when modeling higher Reynolds number flows.

1.2.2 High-speed and turbulent flows

Modeling high Reynolds number flows is essential for capturing turbulence, which is inherently multiscale and chaotic. While traditional approaches like RANS and LES have dominated this domain, PINNs offer an alternative by embedding physical laws directly into the learning process.

Several studies have extended PINNs to turbulent flows. Eivazi et al. [15] applied PINNs to the RANS equations, enabling high Reynolds modeling with turbulence effects. Similarly, Yang et al. [16] incorporated physics-based constraints to allow extrapolation beyond the training Reynolds number, recovering the law of the wall and improving LES modeling. Kag et al. [17] observed that basic PINNs captured large-scale statistics but not small-scale turbulence, and proposed modifications to resolve energy spectra more accurately with less reliance on training data. Raissi et al. [18] developed turbulence models from sparse, noisy experimental data and validated them against analytical solutions. Leoni et al. [19] reconstructed Rayleigh Bénard convection using temperature data, showing PINNs outperform Nudging [20] at high Rayleigh numbers when data coverage is dense. PINNs have also been employed as a super-resolution tool for PIV/PTV data [21], enabling velocity and pressure field reconstruction with limited measurements.

In high-speed compressible flows, Mao et al. [22] used PINNs with the

Euler equations and sparse data to capture shock waves and steep gradients. Although promising, they noted that PINNs still lag behind conventional solvers in forward problems. The XPINN framework [23], further developed by Jagtap et al. [24], extended PINNs to supersonic flows using domain decomposition and was successfully applied to inverse problems involving shocks and expansions. These works demonstrate the adaptability of PINNs to high Reynolds and turbulent flows, especially when combined with hybrid models, physical priors, or domain decomposition strategies.

1.2.3 Complex flow phenomena and advanced applications

Beyond traditional Newtonian fluid dynamics, PINNs have been extended to more complex and non-Newtonian flow problems. Reyes et al. [25] used PINNs to infer viscosity models of polymer melts and particle suspensions from sparse velocity data, enabling solution of the momentum equations for non-Newtonian fluids. Similarly, Thakur et al. [26] introduced ViscoelasticNet for stress discovery and model selection in viscoelastic flows using velocity fields and partial stress data.

PINNs have also been used in rarefied flow regimes. Lou et al. [9] applied PINNs to flows governed by the Boltzmann-BGK model in both continuum and rarefied regimes, demonstrating success on benchmark problems such as Kovasznay and cavity flow. De Florio et al. [27] approximated creep flow in a plane channel using a constrained PINN formulation combining free functions and functionals.

In bio-inspired and experimental contexts, Calicchia et al. [28] used PIV data to reconstruct pressure fields sensed by fish, showing that PINNs are robust even with low-resolution inputs. Cai et al. [29] reconstructed 3D unsteady velocity and pressure fields over an espresso cup from Schlieren imaging, validating their results against PIV measurements.

1.2.4 Free surface flows

Chen et al. [30] applied a PINN to estimate flow properties beneath nonlinear periodic waves in inviscid rotational flows. Huang et al. [31] proposed a PINN approach that treats both initial and boundary conditions as hard constraints and tested it on free surface problems. Jagtap et al. [32] addressed ill-posed nonlinear wave dynamics governed by the Serre–Green–Naghdi equations by integrating labeled data with physical laws in a PINN framework. Duong et al. [33] used PINNs to reconstruct free surface profiles in wave-in-deck loading scenarios using measured data and Euler's equations. Wang et al. [34] demonstrated the potential of PINNs to recover nearshore wave fields by combining energy balance laws, dispersion relations, and observational data.

1.3 Objectives, goals and contributions

While PINNs have shown significant promise, several challenges remain, particularly in solving forward problems, enforcing boundary conditions, and choosing appropriate modeling strategies. This study investigates the application of PINNs to two representative problems in fluid dynamics that span a spectrum of physical assumptions and mathematical complexity.

The first case is the classical linear wave model which describes inviscid, incompressible, and irrotational flow with a free surface. The governing Laplace equation, together with linearized dynamic and kinematic boundary conditions, offers an analytically tractable system. In particular, our aim is to improve the enforcement of boundary conditions within the PINN framework, focusing on the periodic boundary condition in the x-direction and the kinematic free surface boundary condition. These are addressed through the addition of specialized layers to the network architecture and the use of tailored trial functions. Accurate implementation of these conditions is essential for capturing the correct physics of wave propagation and is crucial for the reliability of PINN based solvers in this context. In addition, we explore the estimation of unknown physical parameters appearing in the loss function without relying on any field data.

The second case is the lid-driven cavity flow, a benchmark problem governed by the incompressible Navier Stokes equations. Unlike the linear wave model this system features viscous effects and nonlinear momentum transport. Its bounded geometry and strong wall interactions pose additional challenges for PINN based solvers. Our aim is to improve the results that are affected by the singularity points in the domain. We use hard constraints and spatial varying weights to assess how they can diminish the negative effects of singularities on the solution quality and network training.

Although the two problems differ in complexity, there is a theoretical connection between them. The linear wave problem can be derived from the Navier Stokes equations under the assumptions of inviscid and irrotational flow. This relationship enables us to assess how well PINNs generalize across simplified and full formulations of fluid motion.

Figure 1.1 outlines the typical stages involved in the design of a PINN, from formulating the problem and representing the model to embedding knowledge and optimizing the network. While these stages often involve both data and physical laws, this thesis focuses solely on fully physics-informed forward problems, without incorporating any observational data. Within this framework, the emphasis is placed on the formulation of physics-based constraints, specifically, how the governing equations and boundary conditions are translated into loss functions and integrated into the PINN architecture.

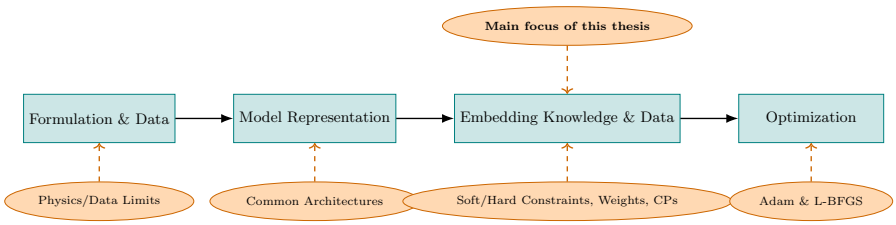


Figure 1.1: Stages in the design of a PINN and the focus areas of this thesis.

Chapter 2

Mathematical description of the linear wave and lid-driven cavity problem

This chapter presents the theoretical foundation for the two fluid dynamics problems explored using PINNs. The first problem is based on linear wave theory, which models an inviscid and irrotational flow with a time-evolving free surface. The second is the lid-driven cavity flow, which describes a viscous and nonlinear flow confined within a closed domain.

2.1 Linear wave theory: modeling free surface dynamics

Free surface flows are central to many problems in fluid dynamics, from oceanography and coastal engineering to ship hydrodynamics and offshore design. In such systems, the upper boundary of the fluid domain is not fixed but deforms in response to the fluid motion, creating a dynamic interface between the liquid and the air. Capturing the behavior of this interface is crucial for predicting wave propagation, energy transmission, and pressure loading on structures.

One of the simplest and most widely studied models for free surface flows is linear wave theory, which describes the evolution of small-amplitude surface waves on an inviscid, incompressible, and irrotational fluid.

2.1.1 Governing equation and assumptions

We consider a two-dimensional fluid domain bounded below by a rigid, impermeable bottom and above by a free surface, as shown in Fig. 2.1. We assume that the fluid is incompressible and irrotational.

Under these assumptions, the velocity field $\mathbf{u} = (u, w)$ can be expressed in

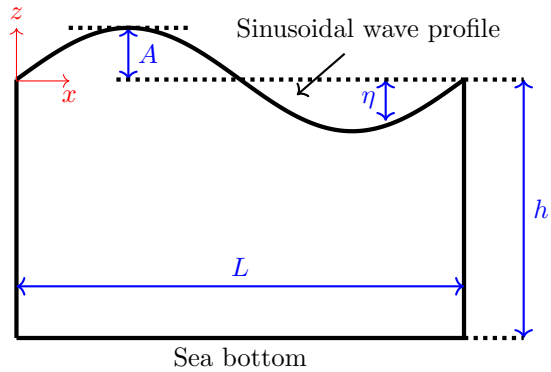


Figure 2.1: Schematic representation of the spatial domain governed by linear wave theory.

terms of a scalar potential function $\phi(x, z, t)$, such that

$$u = \frac{\partial \phi}{\partial x}, \qquad w = \frac{\partial \phi}{\partial z}.$$

The incompressibility condition,

$$\nabla \cdot \mathbf{u} = \frac{\partial u}{\partial x} + \frac{\partial w}{\partial z} = 0,$$

leads to the Laplace equation for the velocity potential, given by

$$\frac{\partial^2 \phi}{\partial x^2} + \frac{\partial^2 \phi}{\partial z^2} = 0. \tag{2.1}$$

2.1.2 Boundary conditions in their original form

To solve the Laplace equation, we must impose boundary conditions at the bottom and the free surface. At the rigid bottom $z = -h$, the vertical velocity must vanish. This gives the kinematic bottom boundary condition (KBBC) as

$$\left.\frac{\partial \phi}{\partial z}\right|_{z=-h} = 0. \tag{2.2}$$

The free surface moves with the fluid. Letting $\eta(x, t)$ represent the surface elevation, a fluid particle initially on the surface remains there if

$$\frac{\partial \eta}{\partial t} + u\frac{\partial \eta}{\partial x} = w \quad \text{at } z = \eta(x, t). \tag{2.3}$$

This condition expresses that the surface moves with the fluid particles and is called kinematic free surface boundary condition (KFSBC). At the free surface, the pressure must match the atmospheric pressure, which we take as constant. Applying Bernoulli's equation for unsteady, irrotational flow gives

$$\frac{p}{\rho} + \frac{\partial \phi}{\partial t} + \frac{1}{2}(u^2 + w^2) + g\eta = C(t), \tag{2.4}$$

and setting $C(t) = \frac{p_{\text{atm}}}{\rho}$ and choosing $p = p_{\text{atm}}$ at the surface yields

$$\frac{\partial \phi}{\partial t} + \frac{1}{2}(u^2 + w^2) + g\eta = 0 \quad \text{at } z = \eta(x,t). \tag{2.5}$$

This boundary condition is called the dynamic free surface boundary condition (DFSBC).

2.1.3 Linearization of boundary conditions

The boundary conditions above are nonlinear. To obtain a tractable linear model, we consider small-amplitude waves where $\eta \ll h$ and apply dimensional analysis to simplify the expressions.

From the original kinematic condition (2.3), we note that

$$\frac{\partial \eta}{\partial t} + u\frac{\partial \eta}{\partial x} = w.$$

The second term $u\frac{\partial \eta}{\partial x}$ is of higher order in the wave amplitude and is therefore negligible compared to the first term. Thus, the linearized kinematic boundary condition becomes

$$\frac{\partial \eta}{\partial t} = w(x,0,t) = \left.\frac{\partial \phi}{\partial z}\right|_{z=0}. \tag{2.6}$$

In the original DFSBC, Eq. (2.5), the nonlinear velocity terms $u^2 + w^2$ are small for small-amplitude waves and can be neglected. Evaluating the potential at the undisturbed surface $z = 0$, the linearized DFSBC becomes

$$\frac{\partial \phi}{\partial t} = -g\eta \quad \text{at } z = 0. \tag{2.7}$$

2.1.4 Analytical solution

The free surface elevation is assumed to be sinusoidal, according to

$$\eta(x,t) = A\cos(kx - \omega t), \tag{2.8}$$

where A is the amplitude, k is the wave number, and ω is the angular frequency.

The solution to this problem yields the velocity potential

$$\phi(x,z,t) = \frac{Ag}{\omega}\frac{\cosh(k(h+z))}{\cosh(kh)}\sin(kx - \omega t). \tag{2.9}$$

From the dynamic condition, the dispersion relation is obtained as

$$\omega^2 = gk\tanh(kh), \tag{2.10}$$

which connects the wave frequency and the wave number and is a fundamental result in surface wave theory.

This problem exhibits several important features from a computational perspective. First, it includes a free surface with two boundary conditions.

Second, it involves periodic solutions, making it an ideal candidate to test how PINNs handle periodicity. Finally, the angular frequency ω is not a prescribed input but must satisfy the dispersion relation, which introduces the possibility of estimating physical parameters as part of an inverse problem. These characteristics make the linear wave theory a valuable test case for validating the fidelity and flexibility of PINN models.

2.2 Lid-driven cavity problem

In contrast to the linear wave model, the lid-driven cavity problem involves viscous, nonlinear, and bounded internal flows. It is a classical benchmark in computational fluid dynamics (CFD), frequently used to assess the accuracy and robustness of numerical solvers, especially those solving the incompressible Navier Stokes equations.

As shown in Fig. 2.2, the setup consists of a square cavity completely enclosed by solid walls. The vertical and bottom walls remain stationary, while the top wall (the lid) moves tangentially at a constant velocity. Despite its geometric simplicity, this configuration gives rise to rich and physically significant fluid behavior. The motion of the lid induces a strong shear layer near the top boundary, which propagates into the interior, generating complex recirculating patterns. A large primary vortex develops in the center, accompanied by secondary corner vortices whose size and structure depend on the Reynolds number.

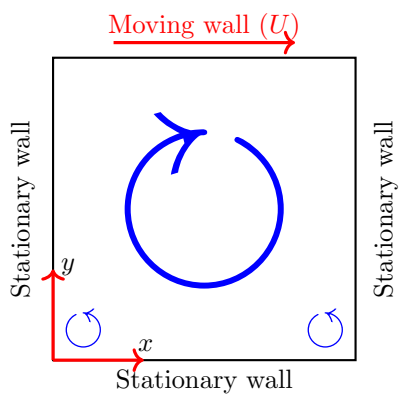


Figure 2.2: Schematic representation of the lid-driven cavity problem, with coordinate system, square geometry, boundary conditions, as well as primary and secondary vortices.

2.2.1 Governing equation

The mathematical formulation of this problem is governed by the two-dimensional steady-state incompressible Navier Stokes equations, which are written as

$$\nabla \cdot \mathbf{u} = 0, \quad \text{(Continuity equation)}, \tag{2.11}$$

$$\mathbf{u} \cdot \nabla \mathbf{u} = -\nabla p + \nu \nabla^2 \mathbf{u}, \quad \text{(Momentum equation)}. \tag{2.12}$$

In these equations, $\mathbf{u} = (u, v)$ denotes the velocity field in the x- and y-directions, p represents the pressure field, and ν is the kinematic viscosity. The first equation enforces incompressibility by requiring the velocity field to be divergence-free. The second describes the momentum balance, incorporating inertial, pressure, and viscous forces.

2.2.2 Boundary conditions

The no-slip boundary condition is imposed on all walls:

1. $\mathbf{u} = (U, 0)$ on the top wall (with U being a constant velocity),
2. $\mathbf{u} = (0, 0)$ on the left, right, and bottom walls.

These conditions create a discontinuity in tangential velocity at the top corners, where the moving wall meets the stationary side walls. This results in steep velocity gradients and strong localized vorticity, which are particularly challenging to resolve accurately in numerical simulations.

In the context of PINNs, this problem offers a rigorous test case to evaluate the framework's ability to solve nonlinear PDEs, handle complex boundary conditions, and reproduce multi-scale flow features. The lid-driven cavity also serves as a bridge to more realistic internal flow problems, such as flow in enclosures and heat transfer in electronics cooling. Success in modeling this case with PINNs would indicate their potential to tackle a wider class of viscous, wall-bounded flows.

Chapter 3

Methodology

3.1 Physics-informed neural network

Solving PDEs using neural networks begins by approximating the solution function $u(x)$ with a trainable neural network $u_\theta(x)$, where θ denotes the network parameters (weights and biases). This approach transforms the PDE problem into an optimization problem, where the goal is to minimize the residuals of the governing equations and boundary conditions evaluated at selected points in the domain.

A general form of a PDE defined on a domain $\mathbf{\Omega} \subset \mathbb{R}^d$ with appropriate boundary conditions on $\partial\mathbf{\Omega}$ can be shown as

$$\mathcal{N}_\tau[u](x) = s(x), \qquad x \in \mathbf{\Omega}, \tag{3.1a}$$

$$\mathcal{B}_\tau[u](x) = g(x), \qquad x \in \partial\mathbf{\Omega}. \tag{3.1b}$$

Here, x denotes spatial or spatio-temporal coordinates, $u(x)$ is the unknown function to be learned, $\mathcal{N}_\tau$ and $\mathcal{B}_\tau$ are differential operators associated with the governing physics and boundary conditions, and τ is a set of physical parameters. The source term $s(x)$ represents external influences or forcing functions acting within the domain, while the function $g(x)$ defines the prescribed values or behavior on the boundary, such as Dirichlet or Neumann boundary conditions.

To approximate $u(x)$, a fully connected feed-forward neural network is used:

$$\hat{u}(x) \approx u_\theta(x), \qquad x \in \mathbf{\Omega} \cup \partial\mathbf{\Omega}, \tag{3.2}$$

where θ denotes the trainable parameters (weights and biases). The network maps the input $x \in \mathbb{R}^d$ to the output via a sequence of layers defined recursively as:

$$h^{(0)} = x, \tag{3.3a}$$

$$h^{(\ell)} = f^{(\ell)}\left(W^{(\ell)}h^{(\ell-1)} + b^{(\ell)}\right), \quad \text{for } \ell = 1, 2, \ldots, L-1, \tag{3.3b}$$

$$u_\theta(x) = W^{(L)}h^{(L-1)} + b^{(L)}, \tag{3.3c}$$

where $f^{(\ell)}$ denotes the activation function in layer ℓ, and $W^{(\ell)}$, $b^{(\ell)}$ are the weight matrix and bias vector at layer ℓ. The final layer typically uses a linear activation function.

PINNs enhance the standard neural network approach by embedding the physics of the problem directly into the training process. Instead of relying solely on data or conventional numerical discretization, PINNs combine the governing equations and, when available, observational data within a unified loss function. This loss function generally includes the residuals of the governing PDEs evaluated at interior collocation points, the residuals of the boundary conditions on the domain boundaries $\partial\Omega$, and, optionally, terms that penalize the mismatch between the network's predictions and given data points. The required derivatives are evaluated using automatic differentiation, which eliminates truncation errors [35]. The network is trained by minimizing the total loss across a set of collocation points distributed in the computational domain.

Overall, PINNs offer a flexible and mesh-free framework for solving PDEs, especially in cases involving complex geometries, limited data availability, or mixed boundary conditions. The framework introduced by Karniadakis et al. [36] further categorizes the sources of bias in PINNs into three types. Observational bias is introduced through the data, inductive bias is a result of the network architecture, and learning bias arises from the structure of the loss function. These categories are illustrated in Fig. 3.1. In this thesis, particular attention is given to the influence of soft and hard constraint formulations on the convergence behavior of PINNs.

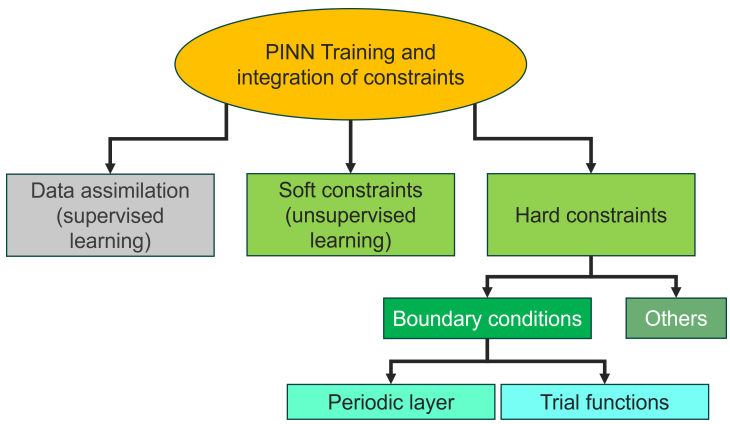


Figure 3.1: Different training approaches for PINNs

Before presenting the specific PINN architectures developed for the linear wave and lid-driven cavity problems, we briefly review advantages of PINNs from different perspectives. These advantages help motivate their use in fluid dynamics and highlight their potential in handling complex physical systems without relying exclusively on dense data or traditional discretization schemes.

3.1.1 Solving direct and inverse problems

Problems addressed by PINNs can be broadly classified as direct or inverse. Direct problems have unique solutions and include well-posed boundary value problems. While PINNs can solve these cases when provided with sufficient collocation points and network expressiveness [5], they often show lower accuracy and higher cost compared to traditional methods [10].

Inverse problems involve unknown parameters and partial data. For example, one may measure velocity fields while the viscosity remains unknown. PINNs can infer such parameters by combining known data and physics. These problems are generally more complex, but PINNs have shown more promise in handling them [37], [38]. Unlike direct problems, inverse setups may include trainable physical parameters within the loss function. When both the solution and some physical parameters are unknown, the setup is referred to as a mixed problem.

3.1.2 Regularizing and refinement of the data

Providing data to train a classic neural network usually requires prior data analysis to exclude biased data and outliers. This process can be time consuming and sometimes not much reliable, and causes removing parts of the physics. PINNs offer a novel way to alleviate noises and outliers in the data. In many cases, the physics of the problem should satisfy some principal laws. Conservation laws are good examples of these laws that for example fluid flow-related problems should conform with. These kinds of laws can be incorporated in the loss function in order to regularize the data points fed to the neural network. Therefore, even in the cases that we have enough data points, PINNs that take advantage of basic knowledge about the problems' physics can outperform classic neural networks. This kind of data refinement based on the basic physical principles will also improves the intuitive interpretability of the results compared to statistical data refinement methods.

3.1.3 Data provision with lower error

Traditional neural networks are sometimes trained using data points generated by other solvers, such as CFD or FEM solvers. As illustrated in Fig. 3.2, employing a separate solver to generate data point for the neural network introduces errors at two stages: during the numerical solution used to generate the data points and during the training of the neural network. In contrast, PINNs circumvent this process (Fig. 3.2). By incorporating the governing equations directly into the neural network, PINNs enable the network to generate data based on the fundamental physics of the problem. If the problem is correctly formulated as a closed boundary-value problem within the PINN framework, the network can access precise data at the specified collocation points, thereby reducing errors introduced by separate solvers used for data generation. However, integrating the problem's physics into a PINN may present additional challenges, which studies like the present one aim to identify and address.

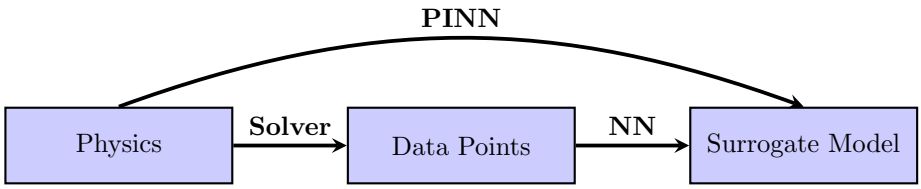


Figure 3.2: The difference between the PINNs and classic neural networks (NN) in dealing with data points

3.1.4 Integrating data and physics in modeling

Unlike traditional neural networks that rely solely on data, and conventional solvers that depend on problem formulation, PINNs combine both. This hybrid approach enables PINNs to incorporate available data and known physics, potentially improving accuracy and convergence.

3.2 PINN for the linear wave problem

The theoretical formulation of the linear wave problem, including its governing equation and boundary conditions, was presented in Chapter 2. In the current subsection, the developed PINN to solve that problem is introduced. This framework enforces the Laplace equation and boundary conditions through physics-informed loss terms and hard constraints.

The network receives spatial coordinates x and z, along with time t, as input. In PINNs, time can be treated as an additional spatial variable, as noted by Raissi et al. [5]. The output is the velocity potential $\phi(x, z, t)$. Automatic Differentiation is used to compute the first and second derivatives of ϕ with respect to its inputs to evaluate the residuals of the governing equation and boundary conditions.

As illustrated in Fig. 3.3, the total loss function L_T is composed of the governing equation loss L_{GE}, the kinematic bottom boundary condition loss L_{KBBC}, and the kinematic free-surface boundary condition loss L_{KFSBC}. These terms are computed assuming the wave number k and angular frequency ω are known and fixed. Once defined, the total loss is minimized using a combination of Adam and L-BFGS optimization algorithms.

Three key challenges are addressed in this modeling framework. First, enforcing the periodic boundary condition in the x-direction is handled through both soft and hard constraints. This topic is explored in the first and second appended papers, and the third paper further develops the periodic layer and loss formulation. Second, the kinematic bottom boundary condition is implemented through trial functions. Two distinct trial functions were designed and evaluated for this purpose, which is discussed in the second appended paper. Lastly, the network is also used to estimate the unknown angular frequency ω, casting the task as an inverse problem. This extension and its results are presented in the second appended paper.

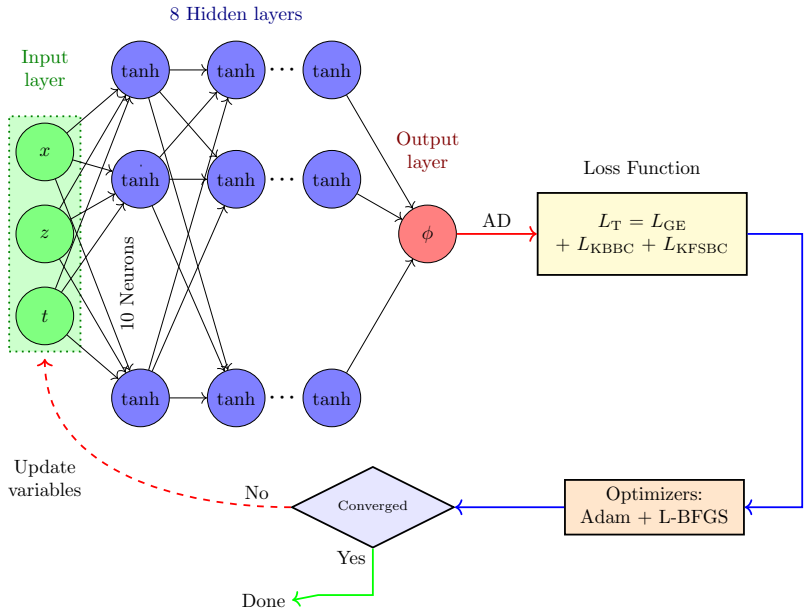


Figure 3.3: Schematic of the PINN used for the linear wave problem, when wave number k and angular frequency ω are known (From Paper II).

3.2.1 Periodic boundary condition

Soft periodic boundary condition To impose periodic boundary conditions in the x-direction, loss terms that ensure the periodicity of the solution and its derivatives are introduced and incorporated into the total loss function. This method is called the soft method, since the periodicity of the solution and its derivatives will not necessarily be satisfied perfectly by dedicating some specified loss terms to them, and the periodicity error depends on how successful the PINN is in the minimization of the respected loss terms. To account for the periodicity of the solution ϕ and its spatial derivatives u and v in the x-direction, three periodicity loss terms are defined. These losses enforce consistency between the values at the two ends of the domain in the x-direction, where λ denotes the spatial wavelength of the wave. The periodicity loss for ϕ is defined as:

$$L_{\phi_{x\text{-per}}} = |\phi_{x=0} - \phi_{x=\lambda}|\,, \tag{3.4}$$

$$L_{u_{x-per}} = |u_{x=0} - u_{x=\lambda}|, \text{and} \tag{3.5}$$

$$L_{v_{x-per}} = |v_{x=0} - v_{x=\lambda}|. \tag{3.6}$$

Another definition for the loss terms has also been studied in Paper III in the appendix. Since that loss term did not show any advantages over this form, that is not presented here.

Hard periodic boundary condition Although adding periodicity-related loss terms has led the average periodicity error to decrease considerably, the success of that method in reducing the periodicity error depends on the performance of the optimization process in the training of the PINN. Therefore, there is no guarantee that having a soft periodicity constraint can decrease the periodicity error up to a certain level. In contrast, Dong and Ni [39] introduced another method for implementing the periodic boundary condition, known as the periodic layer. This layer is inserted between the input and hidden layers in the PINN, as depicted in Fig. 3.4.

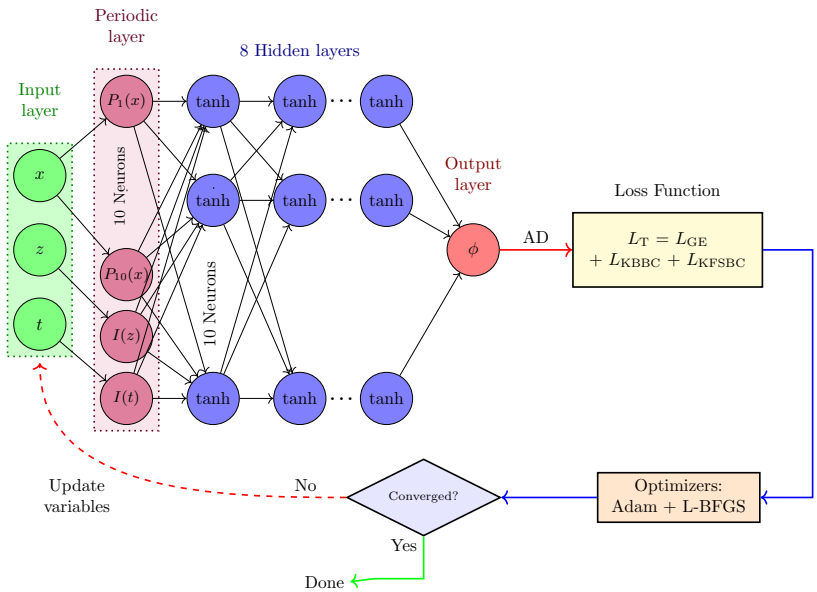


Figure 3.4: Schematic of the PINN used for the linear wave problem, having a periodic layer to meet the periodic boundary condition (BC) of the problem (From Paper II).

The periodic layer enforces the periodic dependence of the PINN output, ϕ, and its spatial derivatives on the input dimension of interest, x. Here, x is transformed into a weighted sum of several periodic functions of x within the periodic layer. This transformation ensures that the PINN's final output remains periodic with respect to x. The important point is that all these periodic functions, should have the same periodicity, α. The detailed mathematical proof for this method is provided in the work by Dong and Ni [39]. The periodic layer is composed of several neurons, each represent a periodic function of the input of interest, x, as

$$P_i = \tanh(A \sin(\alpha x + B) + C), \tag{3.7}$$

where P_i is the ith neuron in the periodic layer connected to the x neuron from the input layer, as shown in Fig. 3.4. The function amplitude, A, phase angle,

B, and the offset, C, are trainable, and α is not trainable and is the same for all P_i functions. The parameter α is the periodicity of the solution with respect to x, which is equal to the periodicity of the free surface profile with respect to x, i.e., $\alpha = 2\pi/\lambda = 1$. It must be noted that tanh operator in Eq. (3.7) is the activation function and has not to be only tanh. We have used this activation function to be consistent with those in other layers of the PINN.

This method is called the hard constraint method as it strictly imposes periodicity, so that the periodic boundary condition will be satisfied to the level of machine precision. Unlike the soft method explained in section 3.2.1, the success of the hard periodicity method does not depend on the optimization performance and convergence of the loss function of the PINN, and the results of the PINN will always be periodic.

By applying Eq. (3.7) on the input of interest in the periodic layer, each neuron adds three trainable variable, i.e., A, B, and C, to the PINN. Having more trainable variable demands more computational cost to converge to the solution. Lu et al. [40] applied the hard periodic boundary conditions in their study, without considering A, B, and C in Eq. (3.7). They also set constant values for α and argued that the input of interest must be decomposed into a weighted summation of the basis functions of the Fourier series, to impose a periodic relationship between that intput and the output of the PINN. Inspiring from the formula proposed by Dong and Ni [39], i.e., Eq. (3.7), and the Fourier basis function, suggested by Lu et al. [40], we evaluate the performance of the PINN, imposing a simplified version of Eq. (3.7), as

$$P_i = \tanh(\sin(\alpha x + B)), \tag{3.8}$$

in the neurons of its periodic layer. The phase angle B in Eq. (3.8) is kept as a trainable variable to offer this freedom to the PINN to find the best phase angles to match the periodical behavior of the solution. Each neuron of the periodic layer imposing Eq. (3.8) adds one trainable variable to the PINN, instead of three with Eq. (3.7).

3.2.2 Trial functions

Lagaris et al. [41] introduced a method for incorporation of boundary conditions in the neural networks designed to solve Ordinary Differential Equations (ODEs) and PDEs. This method has also been improved and extended by Lagari et al. [42]. This method can be categorized as a hard method, since the method strictly imposes the boundary conditions of interest, and no loss term needs to be explicitly dedicated to them. The satisfaction of the boundary conditions treated with this method is independent of the training process and is always guaranteed, even if the PINN does not converge to the correct solution, just like what was observed about the hard implementation of the periodic boundary condition in section 3.2.1. In this method, a function of the PINN's output should be set as the target function, which is being used in different loss terms of the loss function. This function is called the trial function, which its location in the PINN can be seen in Fig. 3.5. The design of the trial function needs to be conducted in a way satisfying the boundary condition(s) of interest

automatically. For example, if we are solving a boundary value problem with PINN to find a target function of $\psi(x)$, having a Dirichlet boundary condition of $\psi(0) = A$, the output of the PINN, $N(x)$, can be put in a trial function like $\psi_t(x) = A + xN(x)$. By using ψ_t in our loss terms as the target function, the Dirichlet boundary condition will always be strictly satisfied.

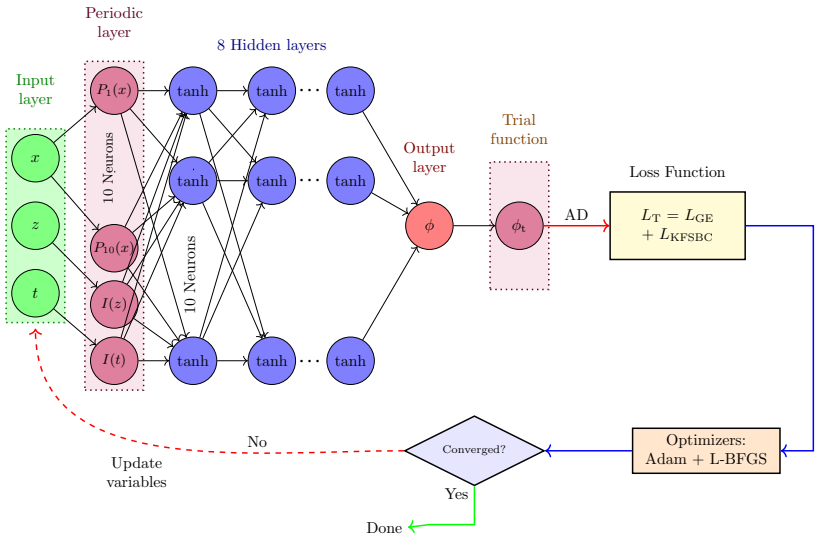


Figure 3.5: Schematic of the PINN used for the linear wave problem, having a periodic layer to meet the periodic BC of the problem, and a trial function to ensure that the neural network output adheres to the required physical constraints (From Paper II).

A proper trial function should fulfill two requirements.

`Requirement #1-` On the boundaries of interest, the trial function must be devoid of the output of the PINN or its derivatives. At these locations, the trial function or its derivatives of interest should be equal to the boundary conditions of the problem.

`Requirement #2-` On the rest of the domain, the trial function should include the output of the PINN or its derivatives.

Since a trial function tailored for implicit implementation of a single homogeneous Neumann boundary condition, i.e., KBBC, has not been included among trial function provided by Lagaris et al. [41] and Lagari et al. [42], we need to develop suitable trial functions based on the requirements mentioned before. We begin with the suggested formula by Lagaris et al. [41] for Poisson boundary value problem, i.e.,

$$\frac{\partial^2}{\partial x^2}\psi(x,y) + \frac{\partial^2}{\partial y^2}\psi(x,y) = f(x,y) \tag{3.9}$$

with mixed boundary conditions of $\psi(0,y) = f_0(y)$, $\psi(1,y) = f_1(y)$, $\psi(x,0) = g_0(x)$, and $\frac{\partial}{\partial y}\psi(x,1) = g_1(x)$. The suggested trial function by Lagaris et al. [41] for imposing these four boundary conditions implicitly is

$$\Psi_t(x,y) = B(x,y) + x(1-x)y[N(x,y,p) - N(x,1,p) - \frac{\partial}{\partial y}N(x,1,p)], \quad (3.10)$$

$$\begin{aligned} B(x,y) = (1-x)f_0(y) + xf_1(y) + g_0(x) - [(1-x)g_0(0) + xg_0(1)] \\ + y\{g_1(x) - [(1-x)g_1(0) + xg_1(1)]\}, \end{aligned} \quad (3.11)$$

where N is the output of the neural network and p is the weights and biases of the neural network. In the suggested trial function for the Poisson boundary value problem, Eq. (3.10), the multipliers of x, $(1-x)$, and y in relate respectively to three Dirichlet conditions of $\psi(0,y) = f_0(y)$, $\psi(1,y) = f_1(y)$, and $\psi(x,0) = g_0(x)$. Since this study intends to use the implicit method only for the KBBC, and KBBC is a homogeneous Neumann boundary condition, we do not need those three multipliers. The term $B(x,y)$ in Eq. (3.10) is also responsible to hold the value of the Poisson function or its derivatives at the boundaries, which is zero in the case of KBBC. Therefore, Eq. (3.10) will be updated accordingly and turns to

$$\phi_t(x,z,t) = N(x,z,t) - N(x,-h,t) - \frac{\partial}{\partial z}N(x,-h,t), \quad (3.12)$$

where $\Psi_t(x,y)$ in Eq. (3.10) has turned to $\phi_t(x,z,t)$ to conform with the linear wave problem terminology, and p is not mentioned, since it is apparent that the output of the PINN is also a function of weights and biases, and we don't mention it. The derivative of Eq. (3.12) with respect to z on $z = -h$ is

$$\frac{\partial}{\partial z}\phi_t(x,z,t)\bigg|_{z=-h} = \frac{\partial}{\partial z}N(x,z,t)\bigg|_{z=-h}, \quad (3.13)$$

which includes a derivation of the PINN output and is not necessarily zero, which violates `requirement #1` of a proper trial function for this problem. One way to resolve this problem is multiplying the bracket via a function of z, which its derivatives with respects to z equals the original function. The only function that its derivatives equals the original function is the exponential function [43]. The exponential function was also used in the trial functions introduced by Lagari et al. [42]. The trial function, Eq. (3.12), turns to

$$\phi_t(x,z,t) = e^z[N(x,z,t) - N(x,-h,t) - \frac{\partial}{\partial z}N(x,-h,t)], \quad (3.14)$$

The derivative of Eq. (3.14) with respect to z on $z = -h$ is

$$\frac{\partial}{\partial z}\phi_t(x,z,t)\bigg|_{z=-h} = 0, \quad (3.15)$$

which satisfies KBBC. Since the exponential function never becomes zero, the PINN output always exists in the function, and no unwanted Dirichlet boundary condition is being imposed on the problem. Therefore, the trial

function in Eq. (3.14) meets both requirements to be a trial function for implicitly imposing KBBC on liner wave problem in the PINN.

For the second candidate for strictly imposing KBBC on the linear wave problem, the trial function for an initial value problem, with Cauchy initial condition introduced by Lagaris et al. [41] has been chosen as a starting point. By adapting the trial function developed by Lagaris et al. [41] for implicitly implementation of KBBC, we can end up with

$$\phi_t(x, z, t) = (z + h)^2 \, N(x, z, t) \tag{3.16}$$

It can be shown that the derivative of this trial function on $z = -h$ equals zero, which satisfies KBBC. However, one drawback of this trial function is becoming zero on $z = -h$, which is the bottom of the computational domain. However, since the values of ϕ and its derivatives, i.e., u and v, are quite small in the bottom, and since the interesting part of the domain for ocean engineering problems is near the free surface, this trial function can be acceptable, as long as it does not ruin the general solution.

3.2.3 Finding an unknown in the physics

If the wave angular frequency, ω, or the wave number, k is considered as an unknown, the dynamic free surface boundary condition (DFSBC) should be used. DFSBC is introduced in Eq. (2.7). Finding ω by the PINN can be interpreted as solving the dispersion relationship, i.e., Eq. (2.10) via the PINN. For this purpose, the DFSBC is added to the loss function of the PINN in Fig. 3.4. The trainable variables in this PINN include the weights and biases of the neural network, phase angles of the neurons in the periodic layer, and ω in KFSBC and DFSBC.

3.3 PINN formulation for the lid-driven cavity flow

In this part of the work, the modeling of the lid-driven cavity problem via a PINN at Reynolds number $Re = 100$ is discussed. The PINN is built using a fully connected feed-forward neural network just like the one used for the linear wave problem. The total loss function used to train the PINN model is composed of multiple components, including residuals of the governing equations and terms that enforce boundary conditions. The total loss is written as:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{cont}} + \mathcal{L}_{\text{mom}X} + \mathcal{L}_{\text{mom}Y} + \mathcal{L}_{\text{top}\,u} + \mathcal{L}_{\text{wall}\,u} + \mathcal{L}_{\text{wall}\,v}. \tag{3.17}$$

Each component is defined as follows:

- $\mathcal{L}_{\text{cont}}$: It enforces the incompressibility condition $\nabla \cdot \mathbf{u} = 0$.
- $\mathcal{L}_{\text{mom}X}$ and $\mathcal{L}_{\text{mom}Y}$: They enforce the the x and y components of the momentum equations, respectively.

- $\mathcal{L}_{\text{top}\,u}$: It enforces the moving wall boundary condition where the horizontal velocity $u = 1$.
- $\mathcal{L}_{\text{wall}\,u}$ and $\mathcal{L}_{\text{wall}\,v}$: They enforce the no-slip condition ($u = v = 0$) on the left, right, and bottom walls.

The following subsections outline two strategies to improve the model's behavior near singularities at the top corners.

3.3.1 Hard constraints through trial functions

Two types of trial functions are used to impose hard boundary conditions in the lid-driven cavity problem. The formulation for the horizontal velocity differs between the two implementations. In the first case, which enforces the boundary condition on the left, right, and bottom walls, the horizontal velocity is defined as

$$u(x, y) = x(1 - x)y\,\hat{u}(x, y), \tag{3.18}$$

which satisfies $u = 0$ on those boundaries while the top wall condition is softly enforced. $\hat{u}(x, y)$ is the network output. In the second case, where hard constraints are imposed at the top and bottom walls, the horizontal velocity is defined as

$$u(x, y) = y(1 - y)\hat{u}(x, y) + y, \tag{3.19}$$

which satisfies $u = 0$ at the bottom and $u = 1$ at the top, while the side wall conditions are softly enforced. For both types, the vertical velocity component is defined as

$$v(x, y) = x(1 - x)y(1 - y)\hat{v}(x, y), \tag{3.20}$$

which ensures that $v = 0$ on all boundaries. $\hat{v}(x, y)$ is the network output.

3.3.2 Spatially varying weights

To mitigate the impact of corner singularities without altering the solution structure, spatially varying weights can be incorporated into the loss function. These weights reduce the influence of collocation and boundary points near high-gradient regions. Two specific weighting functions are introduced, according to

$$\begin{aligned}\omega_F(x, y) = 1 &- 0.99 \exp\left(-20(x - 0.0)^2 - 20(y - 1.0)^2\right) \\ &- 0.99 \exp\left(-20(x - 1.0)^2 - 20(y - 1.0)^2\right), \end{aligned} \tag{3.21}$$

$$\omega_{\text{mw}}(x) = 1 - 4x^2. \tag{3.22}$$

The function $\omega_F(x, y)$ is applied to the residual loss of the Navier Stokes equations to reduce the contribution of points near the top corners, where velocity discontinuities lead to steep gradients. The function $\omega_{\text{mw}}(x)$ is applied to the boundary loss of the horizontal velocity component on the moving wall, down-weighting the ends of the boundary to minimize the influence of corner singularities.

The weighted loss terms are defined as

$$\mathcal{L}_F = \frac{1}{N_F} \sum_{i=1}^{N_F} \omega_F(x_i, y_i) \left| F(u, v, p) \right|^2 , \tag{3.23}$$

$$\mathcal{L}_B = \frac{1}{N_B} \sum_{i=1}^{N_B} \omega_{\mathrm{mw}}(x_i) \left| u(x_i, y_i) - 1 \right|^2 , \tag{3.24}$$

where $\mathcal{L}_F$ is the residual loss of the governing equations, $\mathcal{L}_B$ is the penalty on the top boundary condition, the function F represents the residuals of the continuity equation and the x- and y-momentum equations. The variable p denotes the pressure field, which is also learned by the neural network along with the velocity components u and v. Here, N_F and N_B are the number of the entire collocation points and those at the respected boundary, respectively.

To analyze the individual and combined effects of these weighting strategies, four configurations are considered: (i) baseline with no weights, $\omega_F = \omega_{\mathrm{mw}} = 1$; (ii) using only ω_F; (iii) using only ω_{mw}; and (iv) applying both weights simultaneously. This formulation enables a controlled investigation into how each weighting strategy affects training stability and predictive accuracy.

Chapter 4

Summary of appended papers

This chapter presents a summary of the main findings from each appended paper. For each study, the objectives and contributions are first introduced, followed by a brief description of the methods used, and a summary of the results and discussion.

4.1 Paper I

4.1.1 Objectives and contributions

This study systematically investigates the use of PINNs for modeling linear water waves. The objective is to predict the velocity potential field beneath a periodic free surface wave, using a PINN trained on the governing equations and boundary conditions of linear wave theory. Linear waves are chosen due to their foundational role in wave mechanics and the availability of analytical solutions, which enable rigorous evaluation of PINN accuracy. Both the wave number and angular frequency are treated as known quantities, allowing the dispersion relation to be satisfied during training. The study contributes a detailed assessment of network behavior under periodic and kinematic boundary conditions and highlights the strengths and limitations of PINNs in this canonical wave modeling problem.

4.1.2 Methods

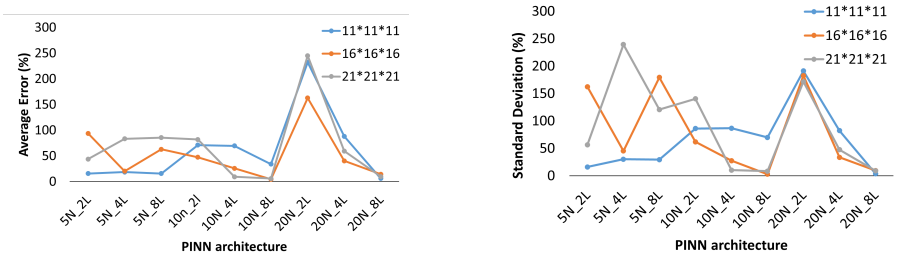
A fully connected PINN architecture was developed. The free surface was modeled as a sinusoidal wave, allowing periodic boundary conditions to be defined in the x-direction. The loss function was composed of six terms derived from the governing equation and the boundary conditions: the Laplace equation in the domain, the kinematic condition at the bottom, the kinematic condition at the free surface, and three periodic boundary conditions in the x-direction for ϕ, $\partial\phi/\partial x$, and $\partial\phi/\partial z$.

The PINN was trained using a combination of the Adam optimizer and the Limited-memory Broyden–Fletcher–Goldfarb–Shanno (L-BFGS) algorithm. A structured mesh of collocation points was used for training, and the network's inputs were the spatial and temporal coordinates (x, z, t), with outputs being $\phi(x, z, t)$. A sensitivity study was conducted to determine the optimal number of layers, neurons per layer, and collocation points. The evaluation metrics included the average error and standard deviation of the error across ten independent training runs.

4.1.3 Results and discussion

Figure 4.1 shows the results of a sensitivity study conducted to investigate the impact of network architecture and collocation density. This study revealed that deeper networks with a moderate number of neurons generally perform better and more consistently. Notably, increasing the number of neurons from 10 to 20 did not always improve accuracy. The chosen architecture with 8 layers, 10 neurons per layer, and a grid of $16 \times 16 \times 16$ collocation points yielded an average error of 4.34% and a standard deviation of 2.79% over ten independent training runs.

Figure 4.2 compares the predicted velocity potential $\phi(x, z, 0)$ by the PINN and the analytical solution at different times. It shows that the PINN accurately predicted the velocity potential field beneath a sinusoidal free surface wave, especially in regions near the free surface where the velocity gradients are most

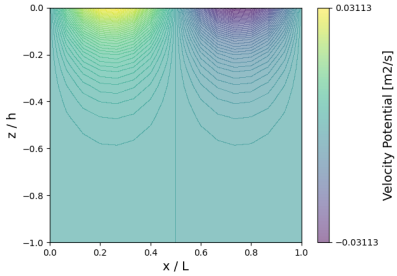


(a) Average error across 10 runs.

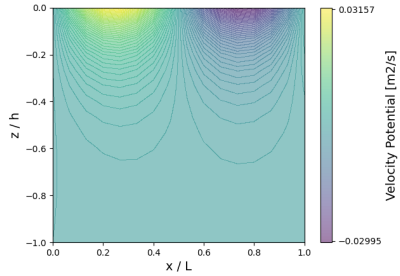
(b) Standard deviation of error across 10 runs.

Figure 4.1: Average error and standard deviation of the error in $\phi(x, z, t)$ prediction across 10 training runs for various network architectures and collocation densities.

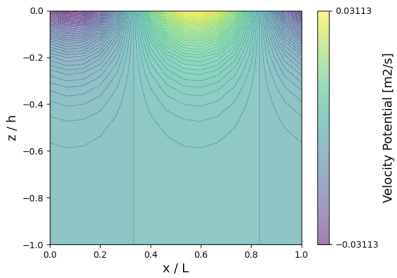
significant. However, some deviations were observed in the lower regions of the domain. The observed prediction error can be attributed to the incomplete enforcement of periodic boundary conditions in the loss function. Although periodic boundary constraints for ϕ, ϕ_x, and ϕ_z in the x-direction were included in the loss formulation, they may not have been sufficiently satisfied during training.



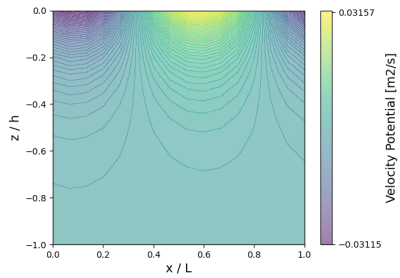
(a) Exact $\phi(x, z, 0)$



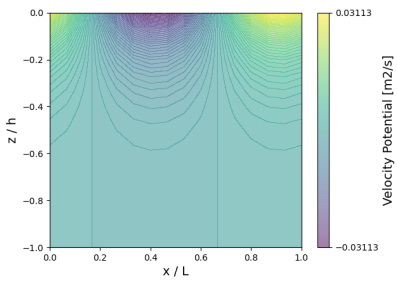
(b) Predicted $\phi(x, z, 0)$



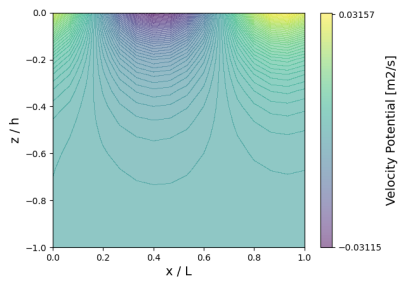
(c) Exact $\phi(x, z, T/3)$



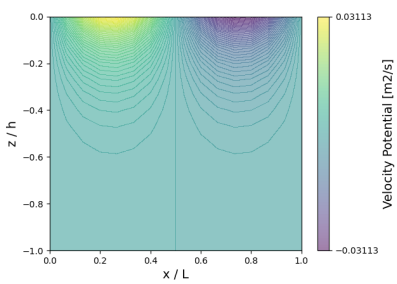
(d) Predicted $\phi(x, z, T/3)$



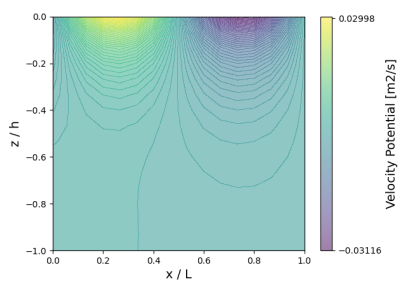
(e) Exact $\phi(x, z, 2T/3)$



(f) Predicted $\phi(x, z, 2T/3)$



(g) Exact $\phi(x, z, T)$



(h) Predicted $\phi(x, z, T)$

Figure 4.2: Comparison of exact and predicted velocity potential distributions $\phi(x, z, t)$ at various time instances.

4.2 Paper II

4.2.1 Objectives and contributions

This study extends prior work on applying PINNs to model linear water waves governed by potential flow theory. While the previous paper focused on assessing PINN performance under periodic and kinematic boundary conditions with known wave characteristics, this work advances the methodology in several key directions.

First, the study systematically evaluates the enforcement of boundary conditions using both soft and hard approaches. In particular, it explores the impact of implementing periodic boundary conditions (PBCs) as hard constraints through architectural modifications, as opposed to soft enforcement via loss terms. Second, trial functions are developed to impose the kinematic bottom boundary condition (KBBC) as a hard constraint. Third, the study investigates the ability of PINNs to infer the wave angular frequency ω as an unknown parameter, using only physics-based loss terms without any labeled data.

Together, these contributions aim to improve the accuracy and interpretability of PINN-based solvers in fluid dynamics problems and provide practical guidance for incorporating boundary conditions and learning unknown parameters in PINN frameworks.

4.2.2 Methods

A fully connected PINN architecture was employed, with inputs consisting of spatial and temporal coordinates (x, z, t) and output being the velocity potential $\phi(x, z, t)$. The model was trained using a hybrid optimization strategy, beginning with Adam optimizer, followed by the L-BFGS algorithm for fine-tuning. A structured mesh of collocation points was uniformly distributed throughout the three-dimensional space–time domain. The loss function was constructed from the Laplace equation in the domain and the boundary conditions derived from linear wave theory.

Hard enforcement of boundary conditions was implemented through architectural modifications to the network. Specifically, the PBC in the x-direction was imposed via a periodic input transformation, effectively acting as a middle layer immediately after the input layer. For the KBBC, custom trial functions were used that embed the constraint directly into the output of the network, as an additional final layer applied to the network output. In the inverse modeling case, the wave angular frequency ω was treated as an additional trainable parameter, incorporated in the same manner as network weights and biases, and optimized jointly with the rest of the model.

4.2.3 Results and discussion

Table 4.1 summarizes a sensitivity study performed to evaluate how the number of layers, neurons per layer, and collocation points affect PINN performance for linear wave modeling. The standard deviations (Std) indicate variability

across 100 independent runs. The best accuracy was achieved with a network of 16 layers, 10 neurons per layer, and 41^3 collocation points.

Table 4.1: PINN Performance for Linear Wave Problem Under Different Configurations

Layers	Neurons	N_{cp}	u Total Error (Std) [%]	w Total Error (Std) [%]	u Period. Error (Std) [%]	w Period. Error (Std) [%]	Time (Std) [s]
4	10	21^3	18.22 (28.04)	0.81 (0.57)	2.31 (1.53)	2.93 (2.17)	40.91 (5.11)
8	5	21^3	24.95 (90.93)	0.54 (0.40)	1.71 (1.31)	1.69 (1.54)	48.08 (6.03)
8	10	21^3	6.64 (12.19)	0.44 (0.38)	1.30 (1.05)	1.52 (1.42)	61.05 (5.82)
8	20	21^3	17.04 (10.28)	5.09 (2.28)	16.46 (6.11)	14.63 (4.47)	34.07 (1.03)
16	10	11^3	3.18 (5.20)	0.57 (1.36)	1.09 (0.95)	1.05 (0.92)	88.76 (8.75)
16	10	21^3	2.68 (6.18)	0.33 (0.46)	1.05 (1.47)	1.00 (1.14)	145.85 (19.02)
16	10	41^3	2.31 (3.37)	0.27 (0.19)	0.84 (0.56)	0.84 (0.63)	250.69 (35.90)
16	10	61^3	2.97 (10.85)	0.26 (0.20)	0.82 (0.57)	0.84 (0.73)	579.76 (94.62)
16	5	21^3	14.22 (31.17)	0.90 (1.35)	2.53 (2.73)	2.51 (2.97)	84.29 (10.10)
16	20	21^3	12.64 (10.62)	4.28 (1.64)	14.69 (4.69)	13.40 (4.77)	60.84 (1.57)
32	10	21^3	6.11 (18.40)	0.44 (0.45)	1.42 (1.46)	1.27 (1.26)	166.98 (16.69)

Figure 4.3 compares the the results of the vanilla PINN (the PINN without PBC) with those of the PINNs with soft hard enforcement of PBC. It can be seen that PBC enforcement decreases the average error and its variation noticeably. Hard PBC with the function introduced in Eq. (3.7) ensures machine-level periodicity and led to the average error of 0.16% in the velocity components u and w. In comparison, soft PBC reduced periodicity errors to 0.10% for both u and w, with slightly lower computational cost. Advantages and limitations of each implementation approach of PBC are summarized in Fig. 4.4.

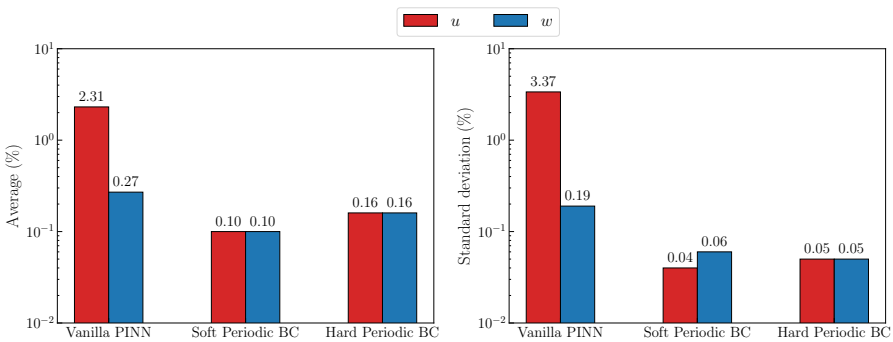


Figure 4.3: Comparison of average and standard deviation of the total errors for different methods of dealing with periodic BCs.

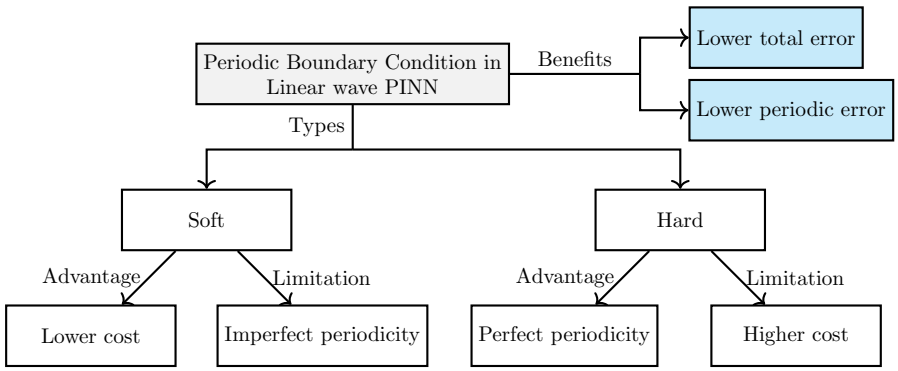


Figure 4.4: Benefits of using PBCs for modeling linear waves, and the advantages and limitations of each implementation approach.

Figure 4.5 compares the the results of the PINN with hard PBC with those of the PINNs with the trial functions in Eq. (3.14) (Hard KBBC 1) and in Eq. (3.16) (Hard KBBC 2). While both succeeded in strictly satisfying the boundary, they unintentionally restricted the solution space, resulting in higher total errors, up to 2.36% for u and 2.10% for w. Figure 4.6 summarizes the limitations and requirements of using trial functions for boundary enforcement.

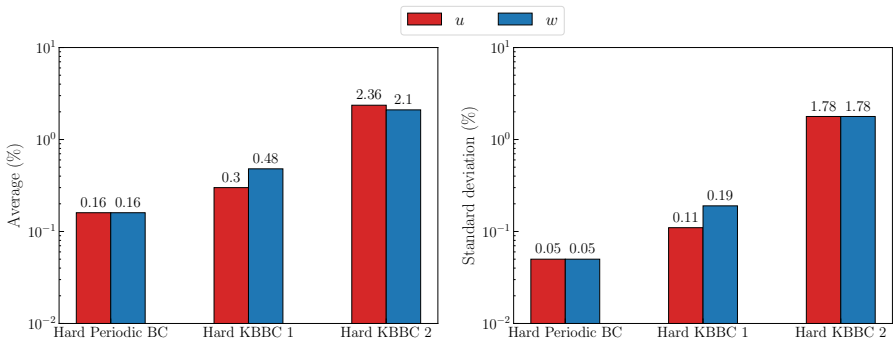


Figure 4.5: Comparison of average and standard deviations of total errors for different methods of dealing with the KBBC.

Additionally, when ω was treated as an unknown, the PINN was able to estimate it with an average error of 0.03%, while maintaining velocity prediction errors below 0.16%. This demonstrates the model's ability to solve inverse-like problems using only physical laws, without requiring labeled data.

Compared to the previous work presented in paper I:

- The current paper introduces and evaluates both soft and hard approaches, while the earlier study used only soft constraints for boundary conditions.
- This work tackles more complex settings, including unknown parameters like ω, thus extending applicability to inverse problems.

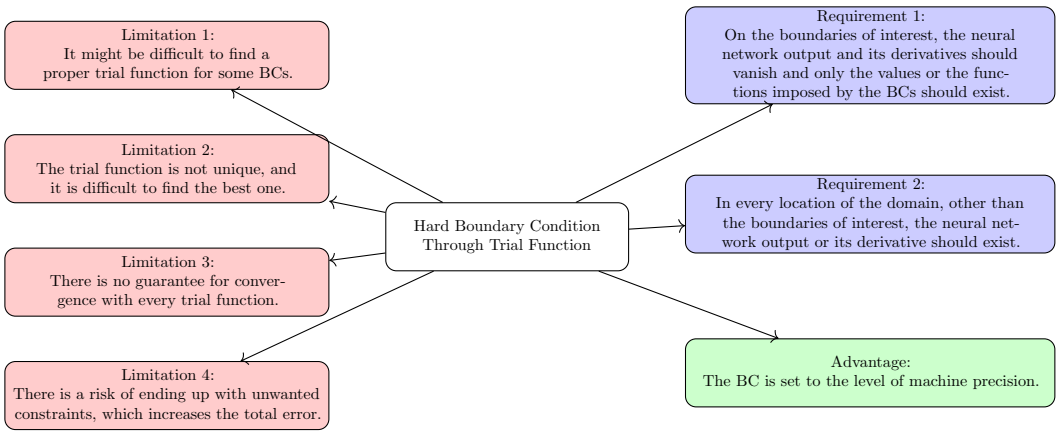


Figure 4.6: Requirements and limitations of hard boundary enforcement using trial functions.

- Practical guidance is provided for implementing hard constraints using trial functions and periodic layers.

4.3 Paper III

4.3.1 Objectives and contributions

This study focuses on the implementation of PBCs in PINNs for modeling linear wave dynamics. Unlike prior work that compared soft and hard enforcement, this research provides a detailed evaluation of two different strategies for imposing PBCs as soft constraints through the loss function. The effectiveness of each strategy is assessed in terms of accuracy and consistency with the periodic nature of the wave field.

In addition, the study investigates hard enforcement of PBCs via periodic layers. Several formulations proposed in the literature are reviewed and analyzed, highlighting their theoretical requirements and practical implications. A simplified variant of a known periodic function is proposed, and its performance is tested to determine the minimum number of neurons needed to accurately capture periodic behavior.

4.3.2 Methods

A fully connected PINN architecture was employed, consisting of eight hidden layers with ten neurons each and using the tanh activation function. The model was trained using a hybrid optimization strategy, beginning with Adam optimizer, followed by the L-BFGS algorithm for fine-tuning. The overall formulation, including the governing equations and case setup, follows the framework established in Papers I and II.

Two soft constraint settings explored in this paper are

1. **One-Period Approach**, where the loss penalizes mismatches at domain boundaries, as shown in Fig. 4.7. The periodic loss function is given by

$$L_{\text{periodic}} = \|u(0) - u(L)\|_2^2. \tag{4.1}$$

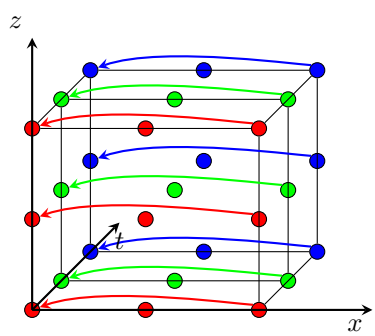


Figure 4.7: One-Period approach for setting soft periodic boundary condition.

2. **Three-Period Approach**, where periodicity is enforced by aligning

points over three consecutive periods, as shown in Fig. 4.8. The periodic loss function is defined as given by

$$L_{\text{periodic}} = \|u(x+L) - u(x)\|_2^2 + \|u(x-L) - u(x)\|_2^2. \tag{4.2}$$

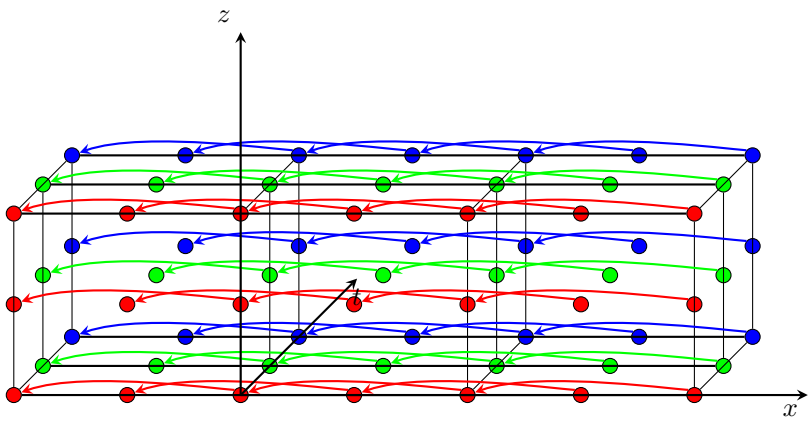


Figure 4.8: Three-Period approach for setting soft periodic boundary condition.

For hard constraint strategies, four periodic layer functions are analyzed, which are based on functions introduced by Dong and Ni [39] and Lu et al. [40], as shown in Fig. 4.9. The study also examines the impact of neuron count in the periodic layer.

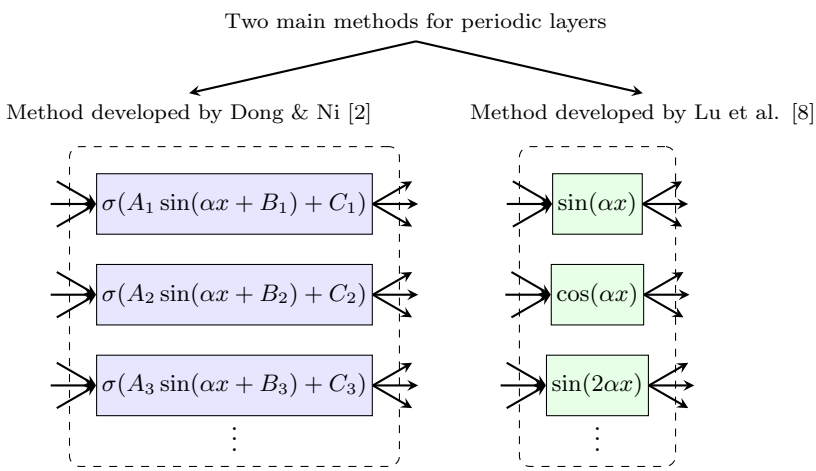
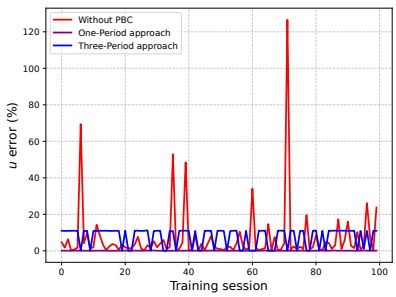


Figure 4.9: Methods developed for hard PBC.

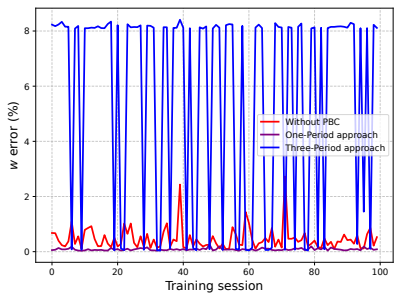
4.3.3 Results and discussion

The performance of different soft and hard PBC enforcement strategies is evaluated using a benchmark linear wave problem. The results, obtained from 100 training runs for each approach, highlight the relative advantages and limitations of each approach:

Soft constraints: As shown in Fig.4.10, the One-Period approach outperformed the Three-Period approach in reducing the total error in u and v. The One-Period approach reduces the average periodicity errors of u and w from 1.34% and 1.58% to 0.10% and 0.07%, respectively, across 100 training runs. In comparison, the Three-Period approach achieves periodicity errors of 0.17% for u and 0.29% for w. These errors are computed by evaluating the differences between predicted values at the periodic domain boundaries and normalizing them by the maximum exact value of the respective variable in the domain. The Three-Period approach often led to convergence to suboptimal solutions (69% of runs).



(a) Average total error values in u



(b) Average total error values in w

Figure 4.10: Errors in u and w without PBC and with PBC using One-Period and Three-Period approaches.

Hard constraints: The best results are achieved using the simplified function $\tanh(\sin(\alpha x + B))$, which combines periodicity and nonlinear transformation. This configuration yields the lowest average errors (0.18% for u, 0.17% for w). Functions with additional parameters (like A and C in $A\sin(\alpha x + B) + C$) do not offer additional modeling benefits but increased training complexity.

Neuron count: Analytical and empirical findings confirm that *two neurons* in the periodic layer are necessary for flexible and accurate approximation of general periodic functions. Using only one neuron limits the optimizer due to mathematical dependency between sine and cosine components.

4.4 Paper IV

4.4.1 Objectives and contributions

This study investigates the application of PINNs to solve the classical lid-driven cavity problem at Reynolds number $Re = 100$, a benchmark case known for its smooth interior solution and singular behavior at the top corners.

Following the linear wave case, which is governed by a potential flow model, the lid-driven cavity problem is considered to examine PINN performance on a more complex system. Unlike the inviscid and irrotational assumptions in the wave problem, the lid-driven cavity is governed by the incompressible Navier Stokes equations and features strong nonlinear and viscous effects. Studying this problem allows us to evaluate how PINNs perform when extended to more realistic fluid dynamics settings.

The study first evaluates a baseline PINN using soft boundary condition enforcement to assess its ability to resolve the general flow features and capture the effects of corner singularities. To examine how different strategies affect the solution accuracy near the singularities, trial functions are introduced for hard boundary condition enforcement. Additionally, the use of spatially varying weights in the loss function is explored to reduce the optimizer's sensitivity to corner effects and improve prediction accuracy.

4.4.2 Methods

The incompressible Navier Stokes equations serve as the governing equations. A fully connected neural network with 8 hidden layers and 10 neurons per layer was trained using a two-phase strategy: an initial 400,000 epochs of Adam optimization followed by the L-BFGS optimizer. The baseline model uses loss terms to enforce all boundary conditions.

In the cases with hard boundary conditions, two configurations of Hard Constraints 1 and Hard Constraints 2 are tested. As shown in Table 4.2, Hard Constraints 1 enforces u hard on the bottom and side walls via the trial function in Eq. (3.18). Hard Constraints 2 enforces u hard on the top and bottom walls using the trial function in Eq. (3.19). In both cases, v is enforced hard on all walls using the trial function in Eq. (3.20).

Table 4.2: Boundary condition enforcement strategies for u and v in Hard Constraints 1 and 2.

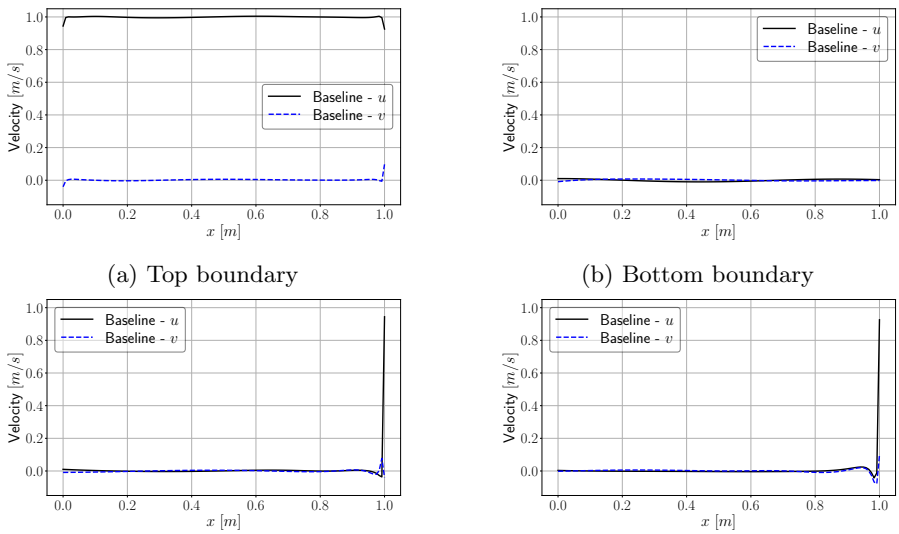
Velocity	**Boundary**	**Hard Constraints 1**	**Hard Constraints 2**
u	Top	Soft (loss term)	Hard (trial function)
	Bottom	Hard (trial function)	Hard (trial function)
	Left	Hard (trial function)	Soft (loss term)
	Right	Hard (trial function)	Soft (loss term)
v	All boundaries	Hard (trial function)	Hard (trial function)

In the cases that spatial varying weights are used, two weight functions, introduced in Eqs. (3.21) and (3.22) are applied independently and in combination, one for the PDE residual loss and one for the boundary loss on the moving wall.

4.4.3 Results and discussion

For analyzing different models' accuracy, their results are compared to CFD results obtained by OpenFOAM. The CFD simulation employed the simpleFoam solver, and a structured mesh with uniform cell distribution was generated. Convective terms in the momentum equations were discretized using the second-order linearUpwind scheme, while the diffusive terms were treated with central differencing linear. Since the case is steady, temporal terms were excluded. Pressure–velocity coupling was handled via the SIMPLE algorithm, and convergence was ensured by reducing the residuals of both momentum and continuity equations below 10^{-7}. These CFD results are validated by comparing to the benchmark results of Ghia et al. [44].

The baseline PINN with all boundary conditions enforced softly produces reasonably accurate results across most of the domain, but fails to capture the correct solution near the top corners, as illustrated in Fig. 4.11.



(a) Top boundary

(b) Bottom boundary

(c) Left boundary

(d) Right boundary

Figure 4.11: baseline PINN solution at boundaries

Hard constraints: The use of trial functions improves adherence to the prescribed boundary conditions in the regions where hard enforcement is applied. However, the presence of discontinuities limits their effectiveness at the top corners. As shown in Fig. 4.12, Hard Constraints 1, with hard constraints

on the bottom and sides, outperforms Hard Constraints 2 in terms of error metrics.

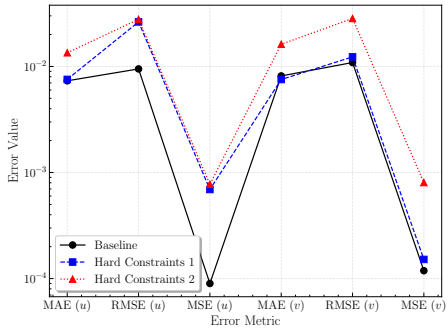


Figure 4.12: Comparison of error metrics for the velocity components u and v, between CFD results and three PINN configurations: the baseline PINN with soft boundary conditions, PINN with Hard Constraints 1, and PINN with Hard Constraints 2. Log scale used for clarity.

Spatially Varying weights: The weighting strategy provides another way to mitigate the impact of corner singularities. Figure 4.13 shows that using the ω_{mw} weight alone yields modest improvements, while using only ω_F weight deteriorates accuracy. The combined use of both weights results in the best balance, specifically in predicting v velocity component.

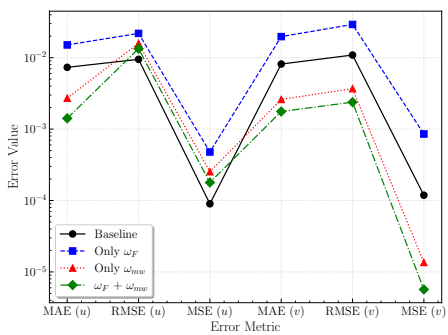


Figure 4.13: Comparison of error metrics for the velocity components u and v, between CFD results and four PINN configurations: the baseline PINN with no weights, PINN with only ω_F, PINN with only ω_{mw}, and PINN with ω_F and ω_{mw}. Log scale used for clarity.

Bibliography

[1] G. K. Batchelor, *An Introduction to Fluid Dynamics*. Cambridge University Press, 1967. DOI: 10.1017/CBO9780511800955. [Online]. Available: https://www.cambridge.org/core/books/an-introduction-to-fluid-dynamics/18AA1576B9C579CE25621E80F9266993 (cit. on p. 1).

[2] H. K. Versteeg and W. Malalasekera, *An introduction to computational fluid dynamics*. Harlow: Pearson Prentice Hall, 1995, ISBN: 0582218845 (cit. on p. 2).

[3] J. H. Ferziger and M. Perić, *Computational Methods for Fluid Dynamics*, Third. Berlin: Springer, 2002, ISBN: 978-3-540-42074-3. DOI: 10.1007/978-3-642-56026-2. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-56026-2 (cit. on p. 3).

[4] J. S. Hesthaven, S. Gottlieb and D. Gottlieb, *Spectral Methods for Time-Dependent Problems* (Cambridge Monographs on Applied and Computational Mathematics). Cambridge University Press, 2007 (cit. on p. 3).

[5] 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, ISSN: 0021-9991. DOI: https://doi.org/10.1016/j.jcp.2018.10.045 (cit. on pp. 3, 17, 18).

[6] S. Cai, Z. Mao, Z. Wang, M. Yin and G. E. Karniadakis, *Physics-informed neural networks (pinns) for fluid mechanics: A review*, 2021. arXiv: 2105.09506 [physics.flu-dyn]. [Online]. Available: https://arxiv.org/abs/2105.09506 (cit. on p. 3).

[7] J. D. Toscano, V. Oommen, A. J. Varghese *et al.*, *From pinns to pikans: Recent advances in physics-informed machine learning*, 2024. arXiv: 2410.13228 [cs.LG]. [Online]. Available: https://arxiv.org/abs/2410.13228 (cit. on p. 3).

[8] M. Raissi, Z. Wang, M. S. Triantafyllou and G. E. Karniadakis, "Deep learning of vortex-induced vibrations," *Journal of Fluid Mechanics*, vol. 861, 119–137, 2019. DOI: 10.1017/jfm.2018.872 (cit. on p. 4).

[9] Q. Lou, X. Meng and G. E. Karniadakis, "Physics-informed neural networks for solving forward and inverse flow problems via the boltzmann-bgk formulation," *Journal of Computational Physics*, vol. 447, p. 110 676, 2021, ISSN: 0021-9991. DOI: https://doi.org/10.1016/j.jcp.2021.110676 (cit. on pp. 4, 5).

[10] A. D. Jagtap, E. Kharazmi and G. E. Karniadakis, "Conservative physics-informed neural networks on discrete domains for conservation laws: Applications to forward and inverse problems," *Computer Methods in Applied Mechanics and Engineering*, vol. 365, p. 113 028, 2020, ISSN: 0045-7825. DOI: https://doi.org/10.1016/j.cma.2020.113028 (cit. on pp. 4, 17).

[11] C. Zhang, C. Li, X. Li, M. Ye and Z. Liu, "A general physics-informed neural network approach for deriving fluid flow fields from temperature distribution," *Chemical Engineering Science*, vol. 302, p. 120 950, 2025, ISSN: 0009-2509. DOI: https://doi.org/10.1016/j.ces.2024.120950 (cit. on p. 4).

[12] X. Jin, S. Cai, H. Li and G. E. Karniadakis, "Nsfnets (navier-stokes flow nets): Physics-informed neural networks for the incompressible navier-stokes equations," *Journal of Computational Physics*, vol. 426, p. 109 951, 2021, ISSN: 0021-9991. DOI: https://doi.org/10.1016/j.jcp.2020.109951 (cit. on p. 4).

[13] P. Karnakov, S. Litvinov and P. Koumoutsakos, "Solving inverse problems in physics by optimizing a discrete loss: Fast and accurate learning without neural networks," *PNAS Nexus*, vol. 3, no. 1, pgae005, 2024. DOI: 10.1093/pnasnexus/pgae005 (cit. on p. 4).

[14] Z. Wang, X. Meng, X. Jiang, H. Xiang and G. E. Karniadakis, *Solution multiplicity and effects of data and eddy viscosity on navier-stokes solutions inferred by physics-informed neural networks*, 2023. arXiv: 2309.06010 [physics.flu-dyn]. [Online]. Available: https://arxiv.org/abs/2309.06010 (cit. on p. 4).

[15] H. Eivazi, M. Tahani, P. Schlatter and R. Vinuesa, "Physics-informed neural networks for solving reynolds-averaged navier–stokes equations," *Physics of Fluids*, vol. 34, no. 7, p. 075 117, Jul. 2022, ISSN: 1070-6631. DOI: 10.1063/5.0095270 (cit. on p. 4).

[16] X. I. A. Yang, S. Zafar, J.-X. Wang and H. Xiao, "Predictive large-eddy-simulation wall modeling via physics-informed neural networks," *Phys. Rev. Fluids*, vol. 4, p. 034 602, 3 2019. DOI: 10.1103/PhysRevFluids.4.034602. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevFluids.4.034602 (cit. on p. 4).

[17] V. Kag, K. Seshasayanan and V. Gopinath, "Physics-informed data based neural networks for two-dimensional turbulence," *Physics of Fluids*, vol. 34, no. 5, p. 055 130, May 2022, ISSN: 1070-6631. DOI: 10.1063/5.0090050 (cit. on p. 4).

[18] M. Raissi, H. Babaee and P. Givi, *Deep learning of turbulent scalar mixing*, 2018. arXiv: 1811.07095 [physics.flu-dyn]. [Online]. Available: https://arxiv.org/abs/1811.07095 (cit. on p. 4).

[19] P. C. D. Leoni, L. Agasthya, M. Buzzicotti and L. Biferale, *Reconstructing rayleigh-benard flows out of temperature-only measurements using physics-informed neural networks*, 2023. arXiv: 2301.07769 [physics.flu-dyn]. [Online]. Available: https://arxiv.org/abs/2301.07769 (cit. on p. 4).

[20] L. Agasthya, P. Clark Di Leoni and L. Biferale, "Reconstructing rayleigh–bénard flows out of temperature-only measurements using nudging," *Physics of Fluids*, vol. 34, no. 1, Jan. 2022, ISSN: 1089-7666. DOI: 10.1063/5.0079625. [Online]. Available: http://dx.doi.org/10.1063/5.0079625 (cit. on p. 4).

[21] S. Cai, C. Gray and G. E. Karniadakis, "Physics-informed neural networks enhanced particle tracking velocimetry: An example for turbulent jet flow," *IEEE Transactions on Instrumentation and Measurement*, vol. 73, pp. 1–9, 2024. DOI: 10.1109/TIM.2024.3398068 (cit. on p. 4).

[22] Z. Mao, A. D. Jagtap and G. E. Karniadakis, "Physics-informed neural networks for high-speed flows," *Computer Methods in Applied Mechanics and Engineering*, vol. 360, p. 112 789, 2020, ISSN: 0045-7825. DOI: https://doi.org/10.1016/j.cma.2019.112789. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0045782519306814 (cit. on p. 4).

[23] A. D. Jagtap and G. E. Karniadakis, "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*, vol. 28, no. 5, pp. 2002–2041, 2020. DOI: https://doi.org/10.4208/cicp.OA-2020-0164 (cit. on p. 5).

[24] A. D. Jagtap, Z. Mao, N. Adams and G. E. Karniadakis, "Physics-informed neural networks for inverse problems in supersonic flows," *Journal of Computational Physics*, vol. 466, p. 111 402, 2022, ISSN: 0021-9991. DOI: https://doi.org/10.1016/j.jcp.2022.111402 (cit. on p. 5).

[25] B. Reyes, A. A. Howard, P. Perdikaris and A. M. Tartakovsky, "Learning unknown physics of non-newtonian fluids," *Physical Review Fluids*, vol. 6, no. 7, Jul. 2021, ISSN: 2469-990X. DOI: 10.1103/physrevfluids.6.073301. [Online]. Available: http://dx.doi.org/10.1103/PhysRevFluids.6.073301 (cit. on p. 5).

[26] S. Thakur, M. Raissi and A. M. Ardekani, "Viscoelasticnet: A physics informed neural network framework for stress discovery and model selection," *Journal of Non-Newtonian Fluid Mechanics*, vol. 330, p. 105 265, 2024, ISSN: 0377-0257. DOI: https://doi.org/10.1016/j.jnnfm.2024.105265 (cit. on p. 5).

[27] M. De Florio, E. Schiassi, B. D. Ganapol and R. Furfaro, "Physics-informed neural networks for rarefied-gas dynamics: Thermal creep flow in the bhatnagar–gross–krook approximation," *Physics of Fluids*, vol. 33, no. 4, p. 047 110, Apr. 2021, ISSN: 1070-6631. DOI: 10.1063/5.0046181 (cit. on p. 5).

[28] M. A. Calicchia, R. Mittal, J.-H. Seo and R. Ni, "Reconstructing the pressure field around swimming fish using a physics-informed neural network," *Journal of Experimental Biology*, vol. 226, no. 8, jeb244983, Apr. 2023, ISSN: 0022-0949. DOI: 10.1242/jeb.244983. [Online]. Available: https://doi.org/10.1242/jeb.244983 (cit. on p. 5).

[29] S. Cai, Z. Wang, F. Fuest, Y. J. Jeon, C. Gray and G. E. Karniadakis, "Flow over an espresso cup: Inferring 3-d velocity and pressure fields from tomographic background oriented schlieren via physics-informed neural networks," *Journal of Fluid Mechanics*, vol. 915, A102, 2021. DOI: 10.1017/jfm.2021.135 (cit. on p. 5).

[30] L. Chen, B. Li, C. Luo and X. Lei, "Wavenets: Physics-informed neural networks for full-field recovery of rotational flow beneath large-amplitude periodic water waves," *Engineering with Computers*, 2024 (cit. on p. 5).

[31] Y. H. Huang, Z. Xu, C. Qian and L. Liu, "Solving free-surface problems for non-shallow water using boundary and initial conditions-free physics-informed neural network (bif-pinn)," *Journal of Computational Physics*, vol. 479, p. 112 003, 2023, ISSN: 0021-9991. DOI: https://doi.org/10.1016/j.jcp.2023.112003 (cit. on p. 5).

[32] A. D. Jagtap, D. Mitsotakis and G. E. Karniadakis, "Deep learning of inverse water waves problems using multi-fidelity data: Application to serre–green–naghdi equations," *Ocean Engineering*, vol. 248, p. 110 775, 2022, ISSN: 0029-8018 (cit. on p. 5).

[33] T. T. Duong, K. H. Jung, G. N. Lee and S. B. Suh, "Physics-informed neural network for the reconstruction of velocity and pressure of wave-in-deck loading from particle image velocimetry data," *Applied Ocean Research*, vol. 153, p. 104 193, 2024, ISSN: 0141-1187. DOI: https://doi.org/10.1016/j.apor.2024.104193 (cit. on p. 5).

[34] N. Wang, Q. Chen and Z. Chen, "Reconstruction of nearshore wave fields based on physics-informed neural networks," *Coastal Engineering*, vol. 176, p. 104 167, 2022, ISSN: 0378-3839. DOI: https://doi.org/10.1016/j.coastaleng.2022.104167 (cit. on p. 5).

[35] L. Lu, X. Meng, Z. Mao and G. E. Karniadakis, "Deepxde: A deep learning library for solving differential equations," *SIAM Review*, vol. 63, no. 1, pp. 208–228, 2021. DOI: 10.1137/19M1274067 (cit. on p. 16).

[36] G. E. Karniadakis, I. G. Kevrekidis, L. Lu, P. Perdikaris, S. Wang and L. Yang, "Physics-informed machine learning," *Nature Reviews Physics*, vol. 3, no. 6, pp. 422–440, 2021, ISSN: 25225820. DOI: 10.1038/s42254-021-00314-5. [Online]. Available: https://doi.org/10.1038/s42254-021-00314-5 (cit. on p. 16).

[37] H. Chen, G. E. C. Flores and C. Li, "Physics-informed neural networks with hard linear equality constraints," *Computers Chemical Engineering*, vol. 189, p. 108 764, 2024, ISSN: 0098-1354. DOI: https://doi.org/10.1016/j.compchemeng.2024.108764 (cit. on p. 17).

[38] H. Baty, *A hands-on introduction to physics-informed neural networks for solving partial differential equations with benchmark tests taken from astrophysics and plasma physics*, 2024. arXiv: 2403.00599 [physics.comp-ph]. [Online]. Available: https://arxiv.org/abs/2403.00599 (cit. on p. 17).

[39] S. Dong and N. Ni, "A method for representing periodic functions and enforcing exactly periodic boundary conditions with deep neural networks," *Journal of Computational Physics*, vol. 435, p. 110 242, 2021, ISSN: 0021-9991. DOI: https://doi.org/10.1016/j.jcp.2021.110242 (cit. on pp. 20, 21, 36).

[40] L. Lu, R. Pestourie, W. Yao, Z. Wang, F. Verdugo and S. G. Johnson, "Physics-informed neural networks with hard constraints for inverse design," *SIAM Journal on Scientific Computing*, vol. 43, no. 6, B1105–B1132, 2021. DOI: 10.1137/21M1397908 (cit. on pp. 21, 36).

[41] I. Lagaris, A. Likas and D. Fotiadis, "Artificial neural networks for solving ordinary and partial differential equations," *IEEE Transactions on Neural Networks*, vol. 9, no. 5, 987–1000, 1998, ISSN: 1045-9227. DOI: 10.1109/72.712178. [Online]. Available: http://dx.doi.org/10.1109/72.712178 (cit. on pp. 21, 22, 24).

[42] P. L. Lagari, L. H. Tsoukalas, S. Safarkhani and I. E. Lagaris, "Systematic construction of neural forms for solving partial differential equations inside rectangular domains, subject to initial, boundary and interface conditions," *International Journal on Artificial Intelligence Tools*, vol. 29, no. 05, p. 2 050 009, 2020. DOI: 10.1142/S0218213020500098 (cit. on pp. 21–23).

[43] J. Fehribach, *Sequences and Series in Calculus* (De Gruyter Textbook). De Gruyter, 2023, ISBN: 9783110768466 (cit. on p. 23).

[44] U Ghia, K. Ghia and C. Shin, "High-re solutions for incompressible flow using the navier-stokes equations and a multigrid method," *Journal of Computational Physics*, vol. 48, no. 3, pp. 387–411, 1982, ISSN: 0021-9991. DOI: https://doi.org/10.1016/0021-9991(82)90058-4. [Online]. Available: https://www.sciencedirect.com/science/article/pii/0021999182900584 (cit. on p. 39).