Prediction of Flow Field Using Machine Learning Techniques

Submitted by

Timothy Toh Tze Han

Department of

Mechanical Engineering

In partial fulfilment of the

requirements for the Degree of

Bachelor of Engineering

National University of Singapore

Session 2020/2021

Summary

The present work compares two machine learning (ML) approaches to modelling a lid-driven cavity flow. The first is a data-driven multi-layer perceptron (MLP) model, which is a naive black-box function approximation of the true underlying process. The second is a physics-inferred neural network (PINN) [1], which incorporates the Navier–Stokes equations into the loss function for training.

We show that the PINN is able to produce a quantitatively and qualitatively superior prediction of the velocity field compared to the naive MLP when both are trained within a small-data regime. Qualitatively, the streamlines predicted by a PINN are much closer to those predicted by the data-driven MLP for the same number of training samples. Quantitatively, the mean squared error (MSE) of the PINN is lower than the MLP for most tested scenarios. This highlights the strength of a PINN as an alternative approach to neural network design for small-data scenarios.

We further demonstrate the ability of the PINN to learn in quasi-unsupervised learning scenarios. This is done through training the PINN without data from the pressure field within the lid-driven cavity, then attempting to predict the pressure. We show that the PINN can accurately predict the pressure field within the lid-driven cavity when trained with no observations of the pressure field, albeit with a constant error. We then demonstrate the possibility to remove this error by training the PINN using a single reference point in the pressure field. This underscores a potential use of the PINN for data generation in low-data regimes.

This project therefore adequately elucidates the usefulness of PINNs within the scientific computing community.

Acknowledgements

The author would like to extend his sincerest gratitude to Prof. Dr. Shu Chang for his guidance and suggestions throughout the course of the project. Prof. Dr. Shu's input was invaluable in ensuring the project remained ambitious yet grounded.

The author would also like to express his utmost appreciation for Mr. Jiang Qinghua, who greatly assisted in the preparation of this dissertation. Throughout the course of the project, Mr. Jiang offered indispensable advice and assistance in debugging the author's code, as well as acted as a sounding board for the author to bounce his ideas off.

Table of Contents

Summary i

Acknowledgements ii

List of Figures vi

List of Tables ix

List of Symbols x

1 Introduction 1

2 Literature Review 4

2.1 Summary 6

3 Theory 7

3.1 The Lid-Driven Cavity Flow 7

3.2 A Primer on Machine Learning 8

3.3 Data-Driven Multi-Layer Perceptron 11

3.4 Physics-Informed Neural Networks 11

4 Methodology 13

4.1 Multi-Layer Perceptron Model and Physics-Informed Neural Network Trained Within Small-Data Regime 13

4.2 Pressure recovery using Physics-Informed Neural Network 14

5 Experimental Procedure **18**

5.1 Computational Fluid Dynamics Data for the Lid-Driven Cavity 18

5.1.1 Validation of OpenFOAM Data 18

5.2 Construction of Both the Multi-Layer Perceptron Model and the Physics-Informed Neural Network 19

5.2.1 Optimising the Activation Function 22

5.2.2 Optimising the Number of Neurons 23

5.2.3 Optimising the Number of Layers 24

5.2.4 Final Construction of Both Neural Networks 24

5.3 Small-Data Regime Comparisons 25

5.3.1 Analysis of Small-Data Regime Training for Both Approaches 29

5.4 Pressure Recovery Using Physics-Informed Neural Network . . . 30

5.4.1 Analysis of Pressure Recovery 32

6 Conclusion and Future Work **33**

6.1 Conclusion 33

6.2 Weaknesses of Physics-Informed Neural Networks 34

6.3 Recommendations for Future Research 35

6.3.1 Relative Weighting of Mean Squared Error Components . 35

6.3.2 Boundary Conditions 35

6.3.3 Unsupervised Learning 36

6.3.4 Alternative Forms of the Navier–Stokes Equations 36

6.3.5 Applying Physics-Informed Neural Networks to Recovering Incomplete Data Sets 36

Bibliography **37**

A Derivations **40**

A.1 Derivation of MSE for PINN 40

B Additional Tables and Diagrams **42**

B.1 Small-Data Sampling Regime 42

B.2 Pressure Recovery 43

List of Figures

1.1 Characteristic steady-state streamlines of lid-driven cavity flow [2], at various Reynolds numbers. (a) $Re = 1000$. (b) $Re = 20\,000$. . . . 3

3.1 A visual representation of the architecture of a typical multi-layer perceptron model. This example model consists of 3 hidden internal layers with 8 neurons per layer. 10

4.1 Location of sampling points within the cavity for different numbers of training points. (a) 64 samples. (b) 128 samples. (c) 256 samples. (d) 512 samples. (e) 1024 samples. (f) 2048 samples. 15

5.1 Centre-line velocity profile comparison between OpenFOAM CFD and numerical solutions provided by Ghia et al. [3] and Erturk et al. [2], for multiple Reynolds numbers. (a), (b) $Re = 1000$. (c), (d) $Re = 5000$. (e), (f) $Re = 10\,000$. 20

5.2 Comparison between original streamline plots and ML-derived streamline plots for the lid-driven cavity at $Re = 1000$. The MSE of the velocity vector $\mathbf{u}$ at each spatial point is demarcated by the colour. (a) Original streamline plot. (b) Streamlines obtained after training PINN with 64 samples. (c) Streamlines obtained after training MLP with 64 samples. 26

5.3 Comparison between original streamline plots and ML-derived streamline plots for the lid-driven cavity at $Re = 8000$. The MSE of the velocity vector $\mathbf{u}$ at each spatial point is demarcated by the colour. (a) Original streamline plot. (b) PINN, 64 samples. (c) MLP, 64 samples. (d) PINN, 256 samples. (e) MLP, 256 samples. . 28

5.4 Comparison between exact pressure field and pressure field predicted by the PINN trained without referencing the pressure field, for a flow regime $Re = 2000$. Note the different scaling along the colour bars for both plots, indicating a constant error between both pressure fields. (a) Exact pressure distribution. (b) Pressure distribution recovered using a PINN. 30

5.5 Comparison between exact pressure field and pressure field predicted by the PINN trained with one reference point in the pressure field, for a flow regime $Re = 4000$. (a) Location of the reference point. (b) Exact pressure distribution. (c) Pressure distribution recovered using a PINN. 31

B.1 Comparison between original streamline plots and ML-derived streamline plots for the lid-driven cavity at $Re = 8000$. The MSE of the velocity vector $\mathbf{u}$ at each spatial point is demarcated by the colour. (a) Original streamline plot. (b) PINN, 1024 samples. (c) MLP, 1024 samples. 44

B.2 Comparison between exact pressure field and pressure field predicted by the PINN trained with one reference point in the pressure field, for a flow regime $Re = 2000$. (a) Location of the reference point. (b) Exact pressure distribution. (c) Pressure distribution recovered using a PINN. 45

B.3 Comparison between exact pressure field and pressure field predicted by the PINN trained without any reference in the pressure field, for a flow regime $Re = 4000$. (a) Location of the reference point. (b) Exact pressure distribution. (c) Pressure distribution recovered using a PINN. 45

List of Tables

5.1 Validation Mean Squared Error for Different Activation Functions . 22

5.2 Validation Mean Squared Error for Different Numbers of Neurons . 23

5.3 Validation Mean Squared Error for Different Numbers of Neurons . 24

5.4 Mean Squared Error for Different Numbers of Training Samples . . 27

B.1 Mean Squared Error for $Re = 8000$ 42

List of Symbols

b_j^n	bias term of the previous layer $n-1$ for neuron j in layer n
Fr	Froude number – $U/\sqrt{gL}$
f	function approximator for the neural network
g	acceleration due to gravity
$\hat{g}$	unit vector in the direction of gravity
h	activation function
i	index of training sample within the training set
j	neuron number
L	characteristic length—taken to be the length of the cavity
MSE_0	mean squared error between true and predicted outputs
$MSE_{0,\,new}$	mean squared error between true and predicted velocity $\mathbf{u}$
$MSE_{0,\,newer}$	mean squared error between true and predicted outputs, with pressure data removed save one reference point
MSE_c	mean squared error of the continuity equation
MSE_m	mean squared error of the conservation of momentum equation
MSE_{mx}	mean squared error of the conservation of linear momentum equation in the x-direction

MSE_{my}	mean squared error of the conservation of linear momentum equation in the y-direction
MSE_{resid}	residual mean squared error, reporting the adherence of the trained model to the Navier–Stokes equations
N	number of training points used
n	layer number
p	pressure
p_{ref}	pressure at a single reference point
$p_{ref,pred}$	predicted pressure at a single reference point
p_{pred}	predicted pressure
p_x	first derivative of pressure in x – $\partial p/\partial x$
p_y	first derivative of pressure in y – $\partial p/\partial y$
p^*	non-dimensionalised pressure – $p/\rho U^2$
Re	Reynolds number – UL/ν
t	time
t^*	non-dimensionalised time – $\mathrm{t}/(\mathrm{L}/\mathrm{U})$
U	driving velocity of the lid-driven cavity
$\mathbf{u}$	velocity vector – $[u, v]^T$
$\mathbf{u}^*$	non-dimensionalised velocity vector $\mathbf{u}/U$
u	horizontal velocity
u_x	first derivative of horizontal velocity u in x – $\partial u/\partial x$
u_{xx}	second derivative of horizontal velocity u in x – $\partial^2 u/\partial x^2$
u_y	first derivative of horizontal velocity u in y – $\partial u/\partial y$

u_{yy}	second derivative of horizontal velocity u in y – $\partial^2 u/\partial y^2$
u_{pred}	predicted horizontal velocity
v	vertical velocity
v_x	first derivative of horizontal velocity v in x – $\partial v/\partial x$
v_{xx}	second derivative of horizontal velocity v in x – $\partial^2 v/\partial x^2$
v_y	first derivative of horizontal velocity v in y – $\partial v/\partial y$
v_{yy}	second derivative of horizontal velocity v in y – $\partial^2 v/\partial y^2$
v_{pred}	predicted vertical velocity
$w_{i,j}^n$	weight of neuron i on the previous layer $(n-1)$ for neuron j in layer n
X	neural network training input
X_{unseen}	input to the neural network for spatial points not seen during training
x	horizontal co-ordinate within the cavity
Y	neural network training (true) output
Y_{unseen}	true output to the neural network for spatial points not seen during training
Y_{pred}	neural network predicted output
y	vertical co-ordinate within the cavity
Θ_j^n	output value of neuron j on layer n
Θ_j^{n-1}	output value of neuron j on previous layer $(n-1)$
μ	dynamic viscosity of the working fluid

ν	kinematic viscosity of the working fluid – μ/ρ
ρ	density of the working fluid
∇	del operator
∇^*	non-dimensionalised del operator – $L\nabla$

Chapter 1

Introduction

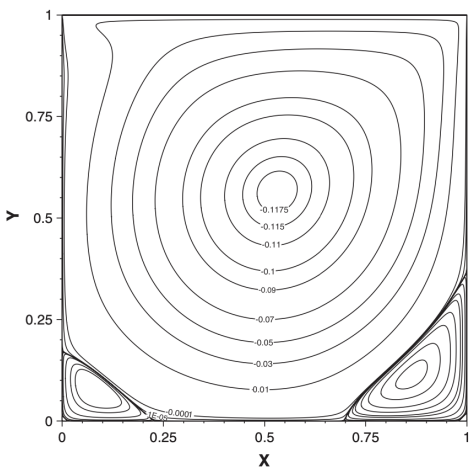
Fluid flow field analysis has traditionally been grounded in physical theory, based chiefly on the Navier–Stokes equations which are conservation equations of mass, momentum and energy [4]. Computational fluid dynamics (CFD) models rely upon these well-established differential equations as the basis upon which flow field characteristics can be found [5]. Practically, the primary challenge of CFD lies in the difficulty of analytically solving the Navier–Stokes differential equations, which can consist of both hyperbolic- and parabolic-dominant regimes within a single flow field [5].

Recent increases in commercially-available computing power have led to an explosion of interest in previously prohibitively computationally-expensive machine learning (ML). The use of ML approaches to solving CFD is by no means a novel concept [6], leveraging on the chief benefit of ML in not requiring explicit discretisation and solution of the underlying Navier–Stokes equations. The standard, data-driven ML approach is to first obtain densely-sampled data from the flow field [6]. This data can then be used to train a 'naive' ML model which can in turn be used to predict and model novel flow conditions [6]. In this way, the ML model acts as a 'good-enough' functional approximation of the flow field when the exact analytical solution is not required [6]. This is the case in many real-life experiments, which are chaotic in nature, and where stochastic effects inhibit modelling accuracy beyond a certain point.

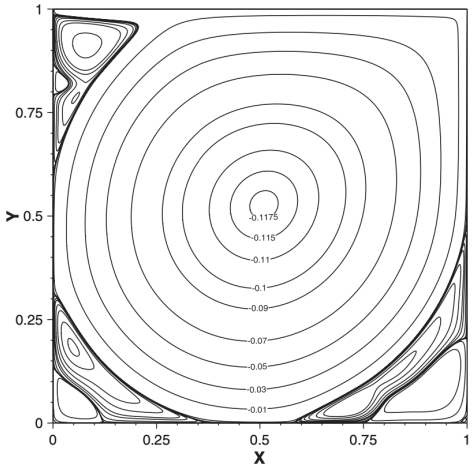
However, this 'strength' of the data-driven ML approach is also a potential drawback. The black-box nature of a standard ML model means that the true process parameters are never elucidated. Consequentially, the data-driven ML approach cannot derive a deeper understanding of the underlying process beyond a simple prediction of outputs based on given inputs [7]. There is also intuitive wisdom in that incorporating the known Navier–Stokes equations into the usual ML process could somehow improve on the data-driven model.

Raissi et al. introduce physics-informed neural networks (PINNs) [1] as a potential improvement over this naive ML approach. The PINN relies on using known physical relationships, such as the aforementioned Navier–Stokes equations, to guide the ML training process. This combines the benefits of both the known, infrangible underlying physical laws with the 'black-box' nature of ML. There are several real benefits of using a PINN over the data-driven ML approach. First, by punishing deviation from known physical equations, the training time and computational resources required to achieve the convergence can be minimised [1, 8–10]. Second, convergence can be achieved within a much smaller training data regime [1, 8–10]. This in turn has practical benefits—in a real-world scenario, the cost of obtaining higher-resolution data from any source is exponentially expensive but yields diminishing returns of convergence [1, 8].

The objective of this project is to study the efficacy, benefits and overall feasibility of using a PINN within the scope of modelling a steady-state lid-driven cavity flow (for which the characteristic streamlines are shown in Figure 1.1), as opposed to a naive data-driven ML approach. In Chapter 2, we explore existing literature on the current applications of PINNs, with a focus on current and potentially relevant applications to our specific scope. We formally introduce the underlying theories behind both the fluid dynamics underpinning the lid-driven cavity flow and the working principles of machine



(a) Streamlines of the lid-driven cavity flow at $Re = 1000$.



(b) Streamlines of the lid-driven cavity flow at $Re = 20\,000$.

Figure 1.1: Characteristic steady-state streamlines of lid-driven cavity flow [2], at various Reynolds numbers. (a) $Re = 1000$. (b) $Re = 20\,000$.

learning in Chapter 3. We then go into detail about our testing methodology in Chapter 4, before presenting our findings and providing a commentary on our results in Chapter 5. Finally, we conclude by highlighting key observations and pinpointing further areas for research in Chapter 6.

Chapter 2

Literature Review

Classical ML approaches are 'black-box' in nature; that is, they are simply an exercise in matching inputs to outputs with no application of *a priori* knowledge to the underlying processes. This approach has been successful in modelling and predicting relationships where the underlying process is either unknown or too complicated to accurately parameterise [1, 7, 9]. A fundamental flaw of the black-box approach to ML is the requirement for a large set of training data in order to best achieve convergence; in practice, it is both difficult and expensive to acquire high-fidelity and high-resolution data for any given process of interest [1, 8]. In addition, these purely data-driven models are not able to take advantage of known physical laws that govern many real-life processes [1, 7–9].

To address these issues, Psichogios and Ungar suggest a "hybrid neural network–first principles approach" to model a fedbatch bioreactor [7]. Such a reactor is governed by both known underlying differential equations and by an unknown process kinetics function. Psichogios and Ungar model the fedbatch bioreactor process using the neural network as an approximator of the unknown kinetics function [7]; the output of the neural network function approximator fed into the first-principles differential equations at each time-step to obtain the process parameters over time. This approach to solving the fedbatch bioreactor simulation is compared against a standard, unstructured, and purely

data-driven neural network. Psichogios and Ungar demonstrate that the hybrid neural network exhibits greater predictive accuracy compared to the 'black-box' neural network, especially in terms of both extrapolation and interpolation of data [7].

Raissi et al. introduce PINNs as a generalised solver for any given "parameterised and nonlinear partial differential equation" [1]. Building on the foundations laid by Psichogios and Ungar, Raissi et al. further apply automatic differentiation to solve for the derivative terms within a given partial differential equation [1]. Raissi et al. incorporate *a priori* knowledge of the underlying process into the neural network training process by defining a loss function which punishes deviation from infrangible underlying physical laws, hence directly constrain the model within the realm of potentially valid solutions [1]. Raissi et al. were able to show that the PINN approach can achieve good predictive accuracy even when the number of training data points is intentionally limited, and further that the final result is robust to effects of Gaussian noise [1]. Interestingly, Raissi et al. observe "qualitatively accurate" prediction of pressure within a flow field, even when the PINN is trained without pressure data [1]; Zhu et al. take this a step further by attempting to use unsupervised learning in conjunction with a PINN to create a process model without requiring any training data [11].

Other literature provide insights into possible ways to strengthen our PINN. Tipireddy et al. demonstrate how a PINN can achieve superior predictive accuracy compared to a naive neural network blind to the underlying physics [12], and in so doing acts as an excellent motivation for our own work. Raissi et al. provide more potential uses of PINNs, and elucidate different ways to apply PINNs to the Navier–Stokes equations, specifically by demonstrating the possibility of using different forms of the Navier–Stokes equation to formulate the loss function [8, 9, 13]. Jagtap et al. introduce an "adaptive activation

function" that promises to further optimise PINNs by accelerating convergence and improving accuracy [14]. Yang and Perdikaris provide a framework for the training of PINNs using partial, low-fidelity training data [15].

2.1 Summary

We are introduced to the idea of incorporating known physical equations to improve neural networks by Psichogios and Ungar [7]. Psichogios and Ungar [7] further provide an excellent methodology for testing and comparing the hybrid and data-driven approaches. Raissi et al. [1] provide us with a concrete framework which we can use to design a PINN to solve the lid-driven cavity flow. We are interested in building on the above works by comparing the capabilities of both traditional data-driven ML and physics-based PINNs through the prediction of a lid-driven cavity flow.

Chapter 3

Theory

3.1 The Lid-Driven Cavity Flow

The two-dimensional (2D) lid-driven cavity flow is a classical problem in fluid dynamics, and functions as an excellent "benchmark problem" for the testing and validation of novel CFD techniques [2]. This class of flow is governed by the incompressible Navier–Stokes equations. The full Navier–Stokes formulation can be cast in the following form:

$$\nabla^* \cdot \mathbf{u}^* = 0 \tag{3.1}$$

$$\frac{\partial \mathbf{u}^*}{\partial t^*} + \mathbf{u}^* \cdot \nabla^* \mathbf{u}^* = -\nabla^* p^* + \frac{1}{Re} \nabla^{*2} \mathbf{u}^* + \frac{1}{Fr^2} \hat{g} \tag{3.2}$$

Equation (3.1) is the non-dimensionalised incompressible continuity equation, and equation (3.2) is the non-dimensionalised momentum equation. $\mathbf{u}^*$ is the non-dimensionalised velocity vector ($\mathbf{u}^* = {}^{\mathbf{u}}/_{U}$); while $\mathbf{u}$ is $[u, v]^T$, where u is the horizontal velocity and v is the vertical velocity. t^* is non-dimensionalised time ($t^* = {}^{t}/_{(L/U)}$), p^* represents non-dimensionalised pressure ($p^* = {}^{p}/_{\rho U^2}$); $\hat{g}$ represents the unit vector in the direction of gravity; ∇^* is the non-dimensionalised del operator ($\nabla^* = L\nabla$); Re is the Reynolds number and Fr is the Froude number. U is the characteristic velocity (for the lid-

driven cavity flow, this is taken to be the driving velocity at the lid); L is the characteristic length (for this problem, this is the length of the cavity); ν is the dynamic viscosity of the working fluid.

Our analysis is limited to the steady-state regime of lid-driven cavity flow; in this condition, the time derivative term vanishes. Further, we consider a fluid in which inertial forces are of a much greater order than gravitational forces; hence the term dependent on the Froude number Fr vanishes. Hence, the steady state flow regime is governed only by the Reynolds number Re.

$$Re = \frac{UL}{\nu} \tag{3.3}$$

To simplify the model further, the governing equations are normalised by setting $U = 1$, $L = 1$ and $\rho = 1$. Equations (3.1) and (3.2) thus collapse into the following form:

$$\nabla \cdot \mathbf{u} = 0 \tag{3.4}$$

$$\mathbf{u} \cdot \nabla \mathbf{u} = -\nabla p + \frac{1}{Re} \nabla^2 \mathbf{u} \tag{3.5}$$

3.2 A Primer on Machine Learning

The present work relies on the Multi-Layer Perceptron (MLP) model trained using supervised learning as the underlying mechanism for the ML process. In supervised learning, a computer is supplied an input set together with a corresponding output set. With a sufficient number of input-output pairs, the computer is able to begin recognising hidden patterns within the

data and is then theoretically able to then predict the output when fed with a hitherto unseen input [16].

The MLP model can be considered a function approximator f that takes in an input X and returns a predicted output Y_{pred}:

$$Y_{pred} = f(X) \tag{3.6}$$

The actual structure of an MLP consists of an input layer and an output layer, with any number of 'hidden' internal layers comprising any number of neurons each. The input tuple X is passed through each internal layer sequentially in a process known as forward propagation through to the output layer Y_{pred}. A graphical representation of this process can be seen in Figure 3.1. Any single neuron j on hidden layer n can be represented by the value Θ_j^n; this value is related to all the neurons in the previous layer $n-1$. Specifically, it is related to the weighted sum of all the neurons in the preceding layer, plus a bias term.

$$\Theta_j^n = h\left(\sum_i w_{i,j}^n \Theta_i^{n-1} + b_j^n\right) \tag{3.7}$$

$w_{i,j}^n$ represents the weight of Θ_i^{n-1} in Θ_j^n; b_j^n represents the bias of the neuron; h represents the activation function. Function f is effectively parameterised by the entire set of weights and biases. The weights $w_{i,j}^n$ and biases b_j^n of each neuron are trainable parameters; the computer adjusts these parameters in order to reduce the Mean Squared Error (MSE) between the predicted output Y_{pred} against the 'true' output Y.

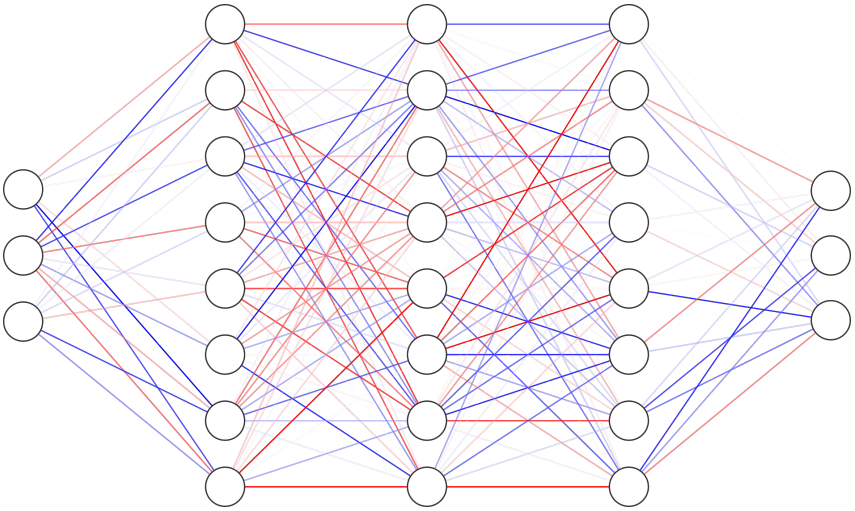


Figure 3.1: A visual representation of the architecture of a typical multi-layer perceptron model. This example model consists of 3 hidden internal layers with 8 neurons per layer.

$$MSE_0 = \frac{1}{3N}\sum_{i=1}^{N}\Big(f(X) - Y\Big)^2 \tag{3.8}$$

MSE_0 as defined in equation (3.8)[1] can be seen as the cost function for adopting a given set of weights and biases, and punishes deviation of the predicted output from the true value. The ML training process can therefore be seen as the iterative minimisation of cost through varying and optimising the trainable weights and biases.

Other parameters such as the number of layers, number of neurons and activation function h are preset hyperparameters which will affect the maximum achievable accuracy of the ML model but otherwise remain constant throughout the training process. These hyperparameters characterise the architecture of the neural network that underlies both the data-driven MLP approach and the PINN.

[1]Usually, the MSE is defined as $1/N\sum_{i=1}^{N}[Y - f(X)]$. The 3 in the denominator of equation (3.8) exists for consistency with the MSE as calculated by TensorFlow's Keras back-end [17, 18].

3.3 Data-Driven Multi-Layer Perceptron

The data-driven MLP represents the first approach to solving the lid-driven cavity flow. An MLP is constructed using TensorFlow's Keras back-end [17, 18]. The goal of this MLP is to minimise MSE as defined in equation (3.8)—this minimises the difference between the MLP output and the true known solution to the flow field. The optimisation process is done using the Adam optimiser [19].

For the 2D steady-state lid-driven cavity, the goal is predicting the flow field characteristics (u, v, p) from the flow parameters (x, y, Re):

$$X = (x,\, y,\, Re) \tag{3.9}$$

$$Y = (u,\, v,\, p) \tag{3.10}$$

$$Y_{pred} = (u_{pred},\, v_{pred},\, p_{pred}) \tag{3.11}$$

3.4 Physics-Informed Neural Networks

Raissi et al. incorporate physics equations into a PINN by defining a custom loss function that incorporates the governing partial differential equations [1]. The present work relies on similar underlying principles. Specifically, we define additional terms to include in the calculation of the overall loss which incorporates the Navier–Stokes partial differential equations.

$$MSE_{PINN} = MSE_0 + MSE_c + MSE_m \tag{3.12}$$

MSE_0 is as defined in equation (3.8). MSE_c encodes the continuity equation (3.4):

$$MSE_c = \frac{1}{N}\sum_{i=1}^{N}\Big(u_x - v_y\Big)^2 \tag{3.13}$$

MSE_m encodes the conservation of momentum equation (3.5). For better clarity, we can further break MSE_m down into horizontal (MSE_{mx}) and vertical (MSE_{my}) components:

$$MSE_m = MSE_{mx} + MSE_{my} \tag{3.14}$$

$$MSE_{mx} = \frac{1}{N}\sum_{i=1}^{N}\Big(u\cdot u_x + v\cdot u_y + p_x - \frac{1}{Re}(u_{xx}+u_{yy})\Big)^2 \tag{3.15}$$

$$MSE_{my} = \frac{1}{N}\sum_{i=1}^{N}\Big(u\cdot v_x + v\cdot v_y + p_y - \frac{1}{Re}(v_{xx}+v_{yy})\Big)^2 \tag{3.16}$$

The derivation of the above MSEs from the Navier–Stokes equations can be found in Appendix A. In equations (3.15) and (3.16), the subscripts x and y on u, v and p denote the partial derivative in the specified direction. Equations (3.13) and (3.14) can be implemented using TensorFlow's GradientTape method [17]. By training a neural network using the overall loss function given in equation (3.12), the Navier–Stokes equations can be incorporated into the ML training process; the final trained model will therefore be in respect of the Navier–Stokes equations. This process thus creates a physics-obeying PINN.

Chapter 4

Methodology

The goal of the present project is comparing the performance of a PINN compared to an MLP for predicting the flow properties of a 2D steady-state lid-driven cavity. In this chapter, the metrics for comparing the two methods are defined.

4.1 Multi-Layer Perceptron Model and Physics-Informed Neural Network Trained Within Small-Data Regime

Raissi et al. posits that a benefit of the PINN is its ability to achieve convergence within a small-data training regime [1]. The motivation behind this experiment is to compare the performance of a PINN to a standard data-driven MLP given a real-world scenario wherein one might be interested in mapping out the flow field within a lid-driven cavity at a single given Reynolds number but one only has a limited sample of training points.

Two flow regimes are chosen for this test. The first is a lid-driven cavity flow with a Reynolds number of 1000, which is selected due to the presence of secondary vortices in the lower left and right corners of the cavity (see Figure 5.2 (a)). The second is a flow with a Reynolds number of 8000, this is selected due to the presence of tertiary vortices in the lower right corner of the flow.

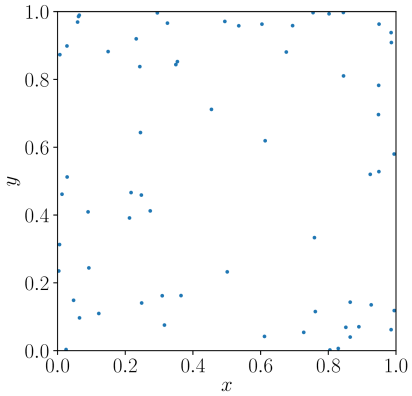
To achieve this comparison, the size of the data-set used in the training process is intentionally constrained. From the entire available set of CFD-derived flow data, a smaller subset of the data is randomly selected for use in training. The size of the training set is varied from 8 to 2048 points within the cavity. This constrained subset of data is used to train both a data-driven MLP and a PINN.

The spatial locations of these randomly sampled points within the lid driven cavity are plotted in Figure 4.1. Subsequently, a PINN and an MLP are trained to completion using this constrained subset of data.

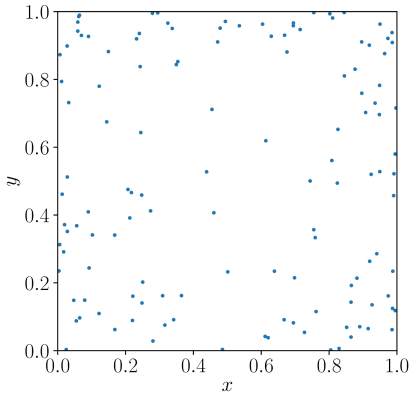
The two ML approaches are compared qualitatively and quantitatively. In the qualitative assessment, the output streamline plots are visually compared against the CFD data. Particular attention is paid to the construction of secondary and tertiary vortices within the lid-driven cavity. In the quantitative assessment, the validation MSE of both approaches are compared. For this experiment, the validation MSE is defined as the mean squared error between the predicted output tuple from either model against the actual output tuple from CFD of the subset of data that is not seen by either the MLP and the PINN during the training process.

4.2 Pressure recovery using Physics-Informed Neural Network

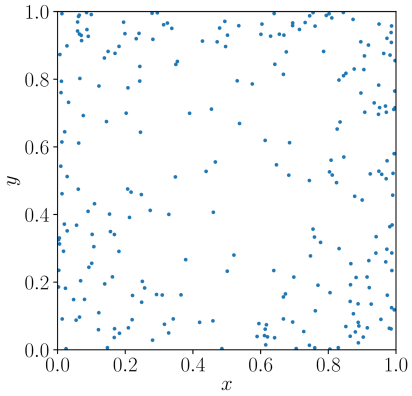
Raissi et al. posit that PINNs can achieve "qualitatively accurate" recovery of the pressure field without being explicitly trained with data from the pressure field. This makes intuitive sense: the Navier–Stokes equations for incompressible flow encapsulate the differential pressure term throughout the cavity; this way, the function approximation of the PINN on the pressure field is in the form of an integration of the derivative pressure terms. As a result, without an anchoring reference pressure point, pressure recovery is only



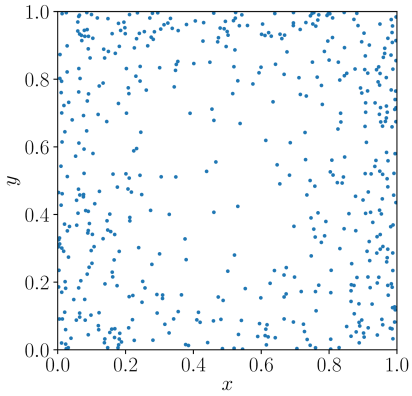
(a) Sampling locations for 64 samples.



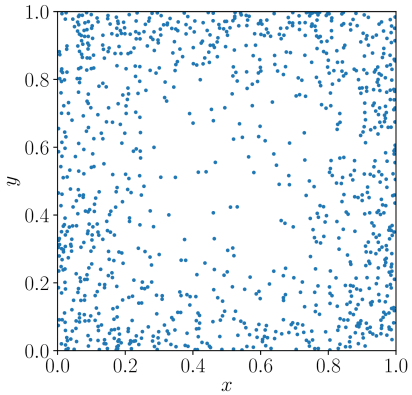
(b) Sampling locations for 128 samples.



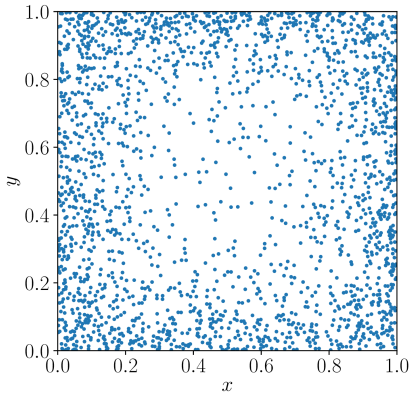
(c) Sampling locations for 256 samples.



(d) Sampling locations for 512 samples.



(e) Sampling locations for 1024 samples.



(f) Sampling locations for 2048 samples.

Figure 4.1: Location of sampling points within the cavity for different numbers of training points. (a) 64 samples. (b) 128 samples. (c) 256 samples. (d) 512 samples. (e) 1024 samples. (f) 2048 samples.

possible up to a constant.

The present work attempts to verify and extend the above postulate laid down by Raissi et al.. In this segment, two approaches to pressure recovery are adopted. The first approach directly follows from methodology employed by Raissi et al.—pressure data is excluded entirely from the training process; hence, the expectation is a recovery of the pressure field except scaled by a constant error throughout the cavity. This happens because the Navier–Stokes equations only include the derivative pressure terms. The second approach then follows from the understanding that accurate pressure recovery is not possible without at least one reference point. Viewing the neural network as an approximate integrator of the differential pressure terms in the Navier–Stokes equation, it is hypothetically possible to recover the original pressure field by providing a single pressure point to the training process. For this section, a standard data-driven MLP is not trained for the sake of comparison. This is because the recovery of the pressure field using an MLP trained without seeing the pressure field is completely stochastic in nature; hence it offers no scientific merit as a basis of comparison.

The result of training a PINN without any training pressure data is achieved by redefining the loss function to ignore the pressure field during the training process:

$$MSE_{0,\,new} = \frac{1}{N}\sum(u - u_{pred})^2 + \frac{1}{N}\sum(v - v_{pred})^2 \tag{4.1}$$

A more practical implementation of the PINN pressure prediction is training it with a single pressure point. This has real-life implications—if attempting to model the pressure flow around a real-life body, a PINN could be used to extrapolate a good approximation of the pressure field from a

single known reference point. By modifying equation (4.1), the loss function incorporating one point on the pressure field can be obtained:

$$MSE_{0,\,newer} = \frac{1}{N}\sum(u - u_{pred})^2 + \frac{1}{N}\sum(v - v_{pred})^2 + (p_{ref} - p_{ref,\,pred})^2 \quad (4.2)$$

In this formulation of the loss function, p_{ref} and $p_{ref,\,pred}$ are the actual and predicted pressures at a single fixed point of reference within the cavity.

Chapter 5

Experimental Procedure

5.1 Computational Fluid Dynamics Data for the Lid-Driven Cavity

OpenFOAM CFD[1] software was utilised for generating high-resolution lid-driven cavity flow data using the modified Navier–Stokes equations described in equations (3.4) and (3.5) at various points throughout the cavity. This data is expressed in terms of the horizontal velocity u, the vertical velocity v, and the pressure p at each given spatial point with Cartesian coordinates (x, y) within a fixed flow regime characterised by Re. This data is then used to train both the data-driven ML as well as the PINN.

5.1.1 Validation of OpenFOAM Data

The output of any ML model is only as accurate as the data used to train it; hence, it is of vital importance to verify the accuracy of the data obtained from the OpenFOAM CFD simulation. This can be done by taking the centre-line velocity profiles of CFD data generated by OpenFOAM and overlaying this against centre-line velocity profiles obtained using alternative numerical methods found in other literature.

[1] OpenFOAM is an open-source CFD software. Specifically, OpenFOAM's SimpleFoam CFD solver, which uses the Semi-Implicit Method for Pressure Linked Equations (SIMPLE) algorithm to solve for incompressible steady-state flow. More documentation can be found in [20].

Ghia et al. and Erturk et al.'s numerical simulations of the lid-driven cavity flow represent two of the most cited reference works for numerical solutions of the lid-driven cavity flow. To capture a large range of Reynolds numbers, CFD data obtained from OpenFOAM at three Reynolds numbers—$Re = 1000$, $Re = 5000$ and $Re = 10\,000$—are plotted and compared against Ghia et al. and Erturk et al.'s results [2, 3].

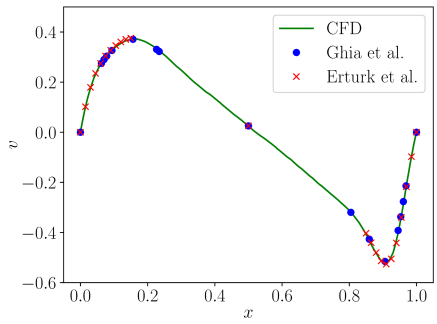
Figure 5.1 plots the centre-line velocities from OpenFOAM CFD and from Ghia et al. and Erturk et al. for both the horizontal and vertical centre-lines at Reynolds numbers of 1000, 5000 and 10 000. Figure 5.1 demonstrates a good correlation between the OpenFOAM CFD data and numerical simulations performed by both Ghia et al. and Erturk et al.. Hence, we can be confident of the accuracy of OpenFOAM's CFD simulation result. At this point, training of the data-driven MLP can begin.

5.2 Construction of Both the Multi-Layer Perceptron Model and the Physics-Informed Neural Network

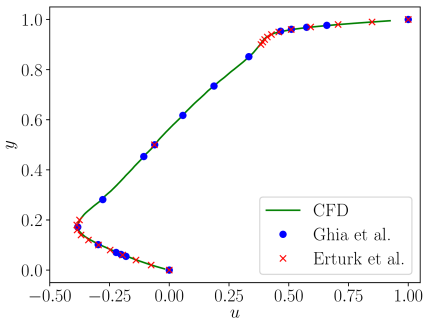
With the CFD data validated, the construction of the MLP and PINN can begin. Google's TensorFlow library [17] provides a suite of methods suitable for constructing and training neural networks. Specifically, the Keras deep learning framework (which is built into TensorFlow) [18] is adopted for constructing both the data-driven MLP and the PINN.

In order to fully explore the differences between an MLP and a PINN, an optimised underlying neural network must be constructed for both approaches. This is done through optimisation of the neural network hyperparameters.

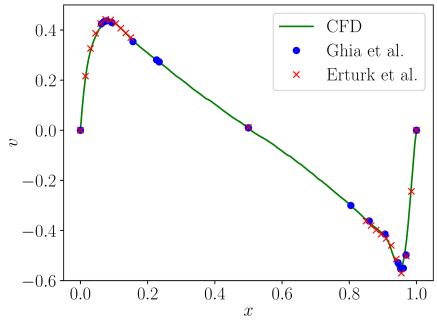
Both the data-driven MLP and the PINN are modelled as sequential, fully-connected neural networks. As elaborated in Chapter 3, the key difference between the two approaches is the loss function adopted—the MLP minimises



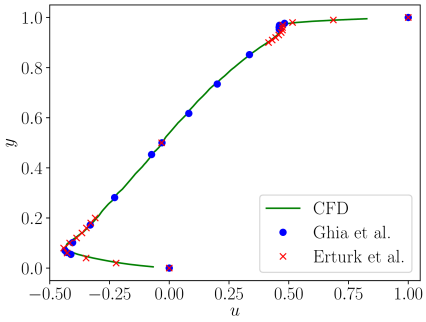
(a) Vertical velocity v along the cavity's horizontal centre-line, $Re = 1000$.



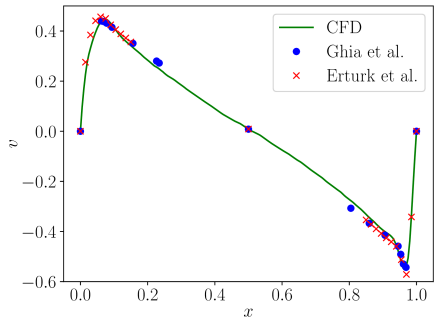
(b) Horizontal velocity u along the cavity's vertical centre-line, $Re = 1000$.



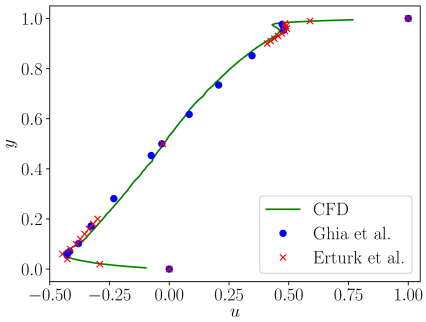
(c) Vertical velocity v along the cavity's horizontal centre-line, $Re = 5000$.



(d) Horizontal velocity u along the cavity's vertical centre-line, $Re = 5000$.



(e) Vertical velocity v along the cavity's horizontal centre-line, $Re = 10\,000$.



(f) Horizontal velocity u along the cavity's vertical centre-line, $Re = 10\,000$.

Figure 5.1: Centre-line velocity profile comparison between OpenFOAM CFD and numerical solutions provided by Ghia et al. [3] and Erturk et al. [2], for multiple Reynolds numbers. (a), (b) $Re = 1000$. (c), (d) $Re = 5000$. (e), (f) $Re = 10\,000$.

MSE as defined in equation (3.8) while the PINN minimises MSE as defined in equation (3.12). Apart from this, in order to ensure a fair comparison between both the MLP and PINN, the same hyperparameters are applied to both the

MLP and the PINN. Both the data-driven MLP and the PINN are constructed with the same number of hidden layers, the same number of neurons per layer and the same activation function.

The data-driven MLP approach is trained by minimising loss using the Adam optimiser [19]. The PINN is trained in a two-stage process, first by minimising loss using the Adam optimiser [19], then by using the limited-memory Broyden–Fletcher–Goldfarb–Fanno (L-BFGS) optimiser [21], following the method adopted by Raissi et al. [1]. The L-BFGS optimiser guarantees global convergence [21]; applying this method for the MLP results in a risk of over-fitting.[2] Conversely, for the PINN, the additional constraint on the Navier–Stokes equations theoretically negates the risk of over-fitting [1], hence, the L-BFGS optimiser can be safely used.

An iterative approach is adopted in order to find the best architecture neural network architecture. This is done by independently training both the MLP and the PINN, then comparing the ending validation MSE. For this process, the validation MSE is defined as the MSE between the predicted output tuple from either model against the actual output tuple from CFD of the subset of data that is not seen by either the MLP or the PINN during the training process.

$$MSE_{val} = \frac{1}{3N} \sum_{i=1}^{N} \Big(f(X_{unseen}) - Y_{unseen} \Big) \tag{5.1}$$

[2]Over-fitting is the phenomenon whereby the neural network parameters become tuned specifically for the input-output pairs 'seen' by the neural network during training, such that when a new, unseen input is introduced, the neural network cannot correctly generate the desired output. For this reason, during the MLP training process, training data is split into two subsets—a training set and a validation set. Then, the neural network is trained using only the training subset, while the MSE of the validation subset is watched. The occurrence of over-fitting can be observed if the validation MSE begins to diverge from the training MSE; in this case, the neural network automatically ceases training. By observing both the training MSE and the validation MSE, over-fitting can be managed and avoided.

The optimum hyperparameters can be selected by minimising the validation MSE while keeping in mind other practical considerations such as training time and model complexity. Since the goal of PINNs is to improve outcomes when training in a small-data regime, 128 spatial points from a flow with Reynolds number 1000 are randomly selected from the cavity for the training process; this data is then used to train both the MLP and the PINN. An added practical benefit of using a constrained data set in the training process is a reduction in overall training time.

To start, the number of layers is fixed at 3 and the number of neurons is fixed at 50 while the activation function is varied. The effect of changing the activation function on ending MSE is observed. The best activation function is the one which minimises ending MSE for both the MLP and the PINN. Subsequently, with the best activation function from the previous test set, the number of neurons can be varied, fixing the number of layers. Finally, fixing the activation function and the number of neurons per layer, the number of layers can be varied, and the number of layers that provides the minimum MSE is kept.

5.2.1 Optimising the Activation Function

The PINN and MLP are initially created with 3 hidden layers with 50 neurons per layer. The activation function is varied. Both neural networks are trained with 128 samples of flow at $Re = 1000$. The validation MSEs for both methods are recorded in Table 5.1.

Table 5.1: Validation Mean Squared Error for Different Activation Functions

Activation function	**PINN MSE**	**MLP MSE**
tanh	1.37×10^{-4}	1.31×10^{-3}
sigmoid	2.44×10^{-4}	3.30×10^{-4}
swish	1.81×10^{-4}	2.85×10^{-4}

From Table 5.1, tanh is chosen as the activation function as it performs the best for both the PINN and the MLP. It is worth noting that while other activation functions are available, they are not usable with the PINN. An example is the Rectified Linear Unit (ReLU) family of activation functions. For these functions, the second-order derivative of ReLU is 0 everywhere. This becomes a problem for the PINN, which incorporates second-order differential terms within the loss function. Applying these activation functions causes the second-order terms to become identically 0. This is not reflective of the actual Navier–Stokes equations. If the activation function is set to ReLU, the PINN to be unable to be trained properly, and the PINN is unable to achieve convergence. Hence, they have been excluded from this testing.

5.2.2 Optimising the Number of Neurons

Fixing the activation function as tanh and the number of layers to 3, the number of neurons are now varied between 10 and 100. As before, the PINN and MLP are trained using 128 points on the $Re = 1000$ flow. The results are plotted in Table 5.2.

Table 5.2: Validation Mean Squared Error for Different Numbers of Neurons

No. of neurons	**PINN MSE**	**MLP MSE**
10	3.37×10^{-4}	1.86×10^{-3}
25	1.60×10^{-4}	2.53×10^{-3}
50	1.37×10^{-4}	1.31×10^{-3}
75	1.41×10^{-4}	2.39×10^{-3}
100	1.46×10^{-4}	1.74×10^{-3}

From Table 5.2, the optimum number of neurons per layer is found to be 50. Increasing the number of neurons from this does not improve validation MSE for either the PINN or the MLP; in contrast, it causes training time to rapidly increase. Hence, 50 neurons per layer is chosen as the optimum number.

5.2.3 Optimising the Number of Layers

Lastly, fixing the activation number as tanh and the number of neurons per layer to 50, the number of layers are varied from 3 to 8. Again, both the PINN and MLP are trained using 128 samples from the $Re = 1000$ flow and tested using the remaining unseen points; the results are plotted in Table 5.3.

Table 5.3: Validation Mean Squared Error for Different Numbers of Neurons

No. of layers	**PINN MSE**	**MLP MSE**
3	1.37×10^{-4}	1.31×10^{-3}
4	1.27×10^{-4}	7.14×10^{-4}
5	1.60×10^{-4}	6.42×10^{-4}
6	8.90×10^{-5}	5.57×10^{-4}
7	1.01×10^{-4}	4.73×10^{-4}
8	7.82×10^{-5}	5.42×10^{-4}

From Table 5.3, the optimum number of layers is 7 for the MLP and 8 for the PINN. Ultimately, 8 layers are chosen for the construction of both neural networks. This decision is made by considering the relative reduction in performance of the MLP for switching from 7 to 8 hidden layers against the reduction in performance of the PINN when switching from 8 to 7 hidden layers.

5.2.4 Final Construction of Both Neural Networks

From the iterative selection process, the optimum neural network architecture is found. This turns out to be a neural network with 8 hidden layers with 50 neurons each, using a tanh activation function.

5.3 Small-Data Regime Comparisons

In this experiment, the convergence performance of the PINN and the MLP are both qualitatively and quantitatively compared, by training both models within a constrained small-data regime.

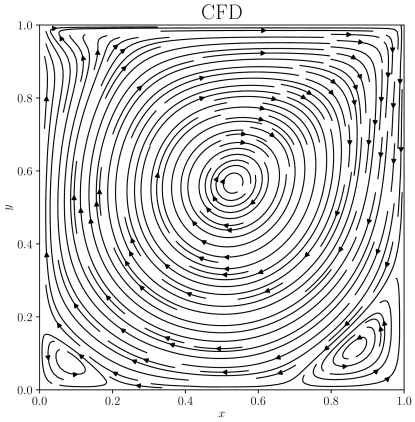
The streamline plots for the lid-driven cavity flow at a Reynolds number of 1000 predicted by the PINN and the MLP, together with the original streamline plot obtained using OpenFOAM CFD, are plotted in Figure 5.2, highlighting the differences between both ML approaches. It can be seen that the PINN output is qualitatively superior to that of the MLP when only 64 points of sampling data are used for training. The MLP is unable to accurately predict the secondary vortices at both the lower left and right corners of the cavity. Furthermore, the MLP is unable to accurately place the core of the primary vortex within the centre of the cavity.

In Figure 5.2, the MSE of the velocity vector $\mathbf{u}$ is shown by the shading of the cavity at each spatial point, with a deeper colour representing a greater deviation from the true velocity. It is evident that the velocity field recovered by the PINN is of a much higher accuracy than that of the MLP.

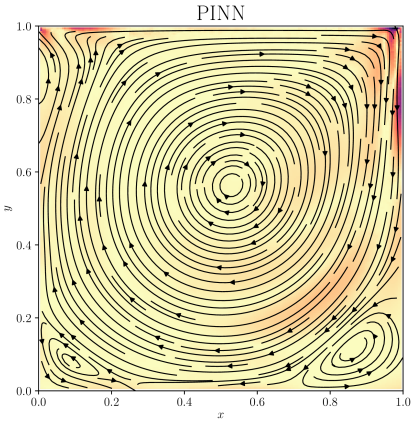
A quantitative assessment of the performance of a PINN compared to the MLP can be performed by recording the validation MSE of either approach after training is completed. Here, an additional metric is defined: residual MSE, which is given as follows:

$$MSE_{resid} = MSE_c + MES_m \tag{5.2}$$

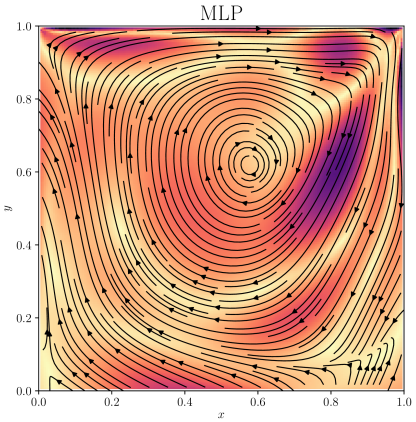
MSE_{resid} records the adherence of the PINN to the Navier–Stokes equations. By definition, this metric applies only to the PINN approach. The validation MSEs from the PINN and the MLP as well as the residual MSEs



(a) Streamlines in original CFD.



(b) Streamlines for PINN trained with 64 sampling points.



(c) Streamlines for MLP trained with 64 sampling points.

Figure 5.2: Comparison between original streamline plots and ML-derived streamline plots for the lid-driven cavity at $Re = 1000$. The MSE of the velocity vector $\mathbf{u}$ at each spatial point is demarcated by the colour. (a) Original streamline plot. (b) Streamlines obtained after training PINN with 64 samples. (c) Streamlines obtained after training MLP with 64 samples.

achieved after training using 8 to 2048 samples for the flow at $Re = 1000$ are plotted in Table 5.4.

To observe the performance of the PINN at higher Reynolds numbers, further testing is done by comparing the PINN and MLP performance for the lid-driven cavity at a flow regime where $Re = 8000$. Figure 5.3 shows the CFD streamline plot as well as the streamlines predicted using a PINN and an MLP at both 64 training samples and 256 training samples. At 64 training samples,

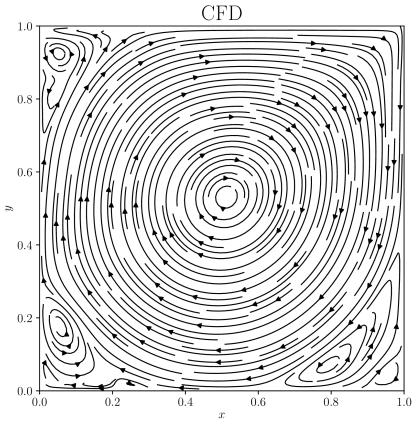
Table 5.4: Mean Squared Error for Different Numbers of Training Samples

Training samples	**PINN MSE**	**Residual MSE**	**MLP MSE**
8	3.72×10^{-2}	3.57×10^{-16}	4.80×10^{-2}
16	1.05×10^{-2}	8.47×10^{-15}	2.30×10^{-2}
32	3.75×10^{-3}	3.14×10^{-10}	8.79×10^{-3}
64	3.72×10^{-4}	4.71×10^{-8}	5.62×10^{-3}
128	8.26×10^{-5}	2.49×10^{-7}	5.42×10^{-4}
256	8.16×10^{-5}	2.74×10^{-6}	3.13×10^{-4}
512	5.15×10^{-5}	3.49×10^{-6}	1.04×10^{-4}
1024	3.00×10^{-5}	2.24×10^{-6}	9.74×10^{-5}
2048	2.41×10^{-5}	2.90×10^{-6}	6.83×10^{-5}

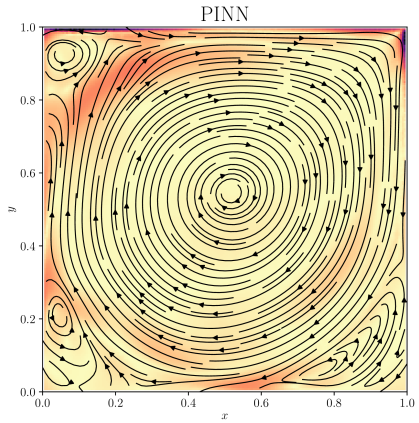
there is a clear difference between the PINN and MLP streamline predictions. The PINN trained using 64 samples, is able to predict the secondary vortices at the bottom left, bottom right and top left corners of the cavity; in stark contrast, the MLP trained using 64 sampling points is only able to predict the primary vortex. Further, the MLP prediction of the location of the primary vortex is wrong. While both predictions show specific areas of high MSE (where the neural network is unable to accurately predict the true velocity at any given point), the MLP prediction is clearly inferior to the PINN throughout the cavity.

Increasing the number of samples, the PINN trained using 256 points is actually able to recover the tertiary vortices forming at the lower left hand corner. The prediction using the MLP trained using 256 points is also qualitatively superior to the MLP trained using 64 points, and is able to predict the three secondary vortices at the bottom left, bottom right and top left corner; however, it remains unable to predict the tertiary vortex forming at the lower right corner.

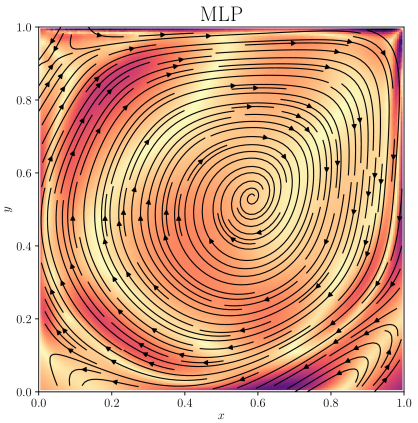
The validation MSEs for the PINN and MLP as well as the residual MSE for the PINN are tabulated in Appendix B. Similarly to the flow at $Re = 1000$, the validation MSE for the PINN is lower than that of the MLP



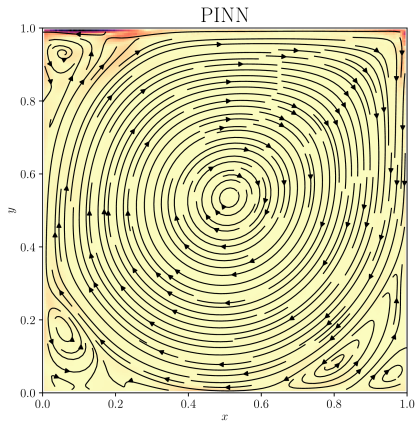
(a) Streamlines in original CFD.



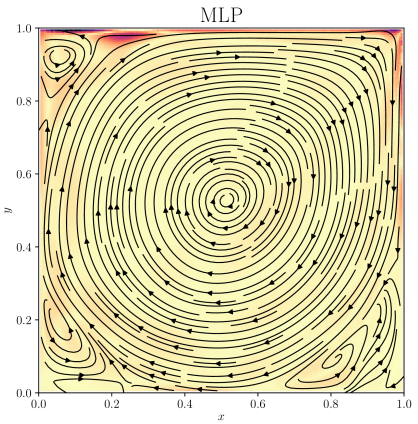
(b) Streamlines for PINN trained with 64 sampling points.



(c) Streamlines for MLP trained with 64 sampling points.



(d) Streamlines for PINN trained with 256 sampling points.



(e) Streamlines for MLP trained with 256 sampling points.

Figure 5.3: Comparison between original streamline plots and ML-derived streamline plots for the lid-driven cavity at $Re = 8000$. The MSE of the velocity vector $\mathbf{u}$ at each spatial point is demarcated by the colour. (a) Original streamline plot. (b) PINN, 64 samples. (c) MLP, 64 samples. (d) PINN, 256 samples. (e) MLP, 256 samples.

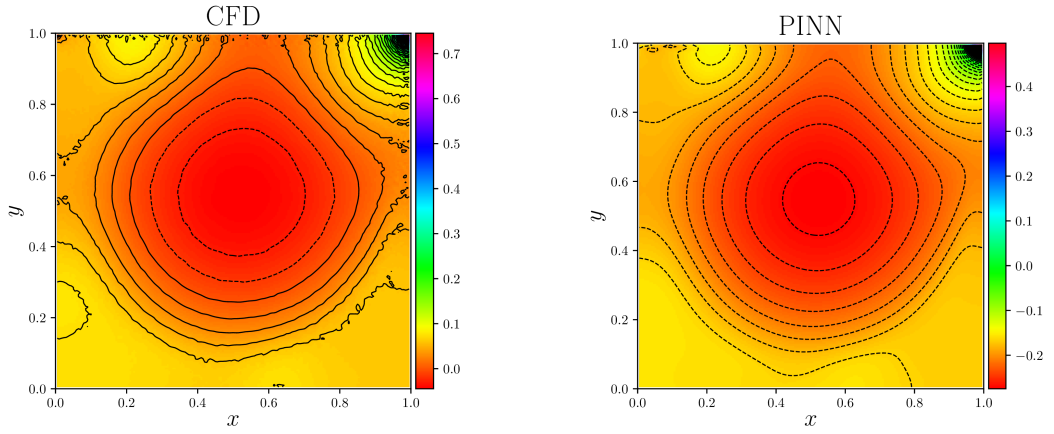
for almost all training samples used, except for 1024 samples.

5.3.1 Analysis of Small-Data Regime Training for Both Approaches

It can be seen that the PINN outperforms the MLP in predicting the flow field around unseen data for all the tested numbers of training samples. Another observation is that increasing the number of training samples improves validation MSE for both the PINN and the MLP. This makes intuitive sense, as a larger amount of training data would undoubtedly improve model training.

Comparing the validation MSE for the PINN and the MLP in Table 5.4, it seems that there is an ideal range of training samples for which the PINN outperforms the MLP. At very low numbers of training samples, the training process is highly stochastic for both the PINN and the MLP; as a result, the overall accuracy of the predicted model is limited. Then, there is a range of numbers where the performance of the PINN vastly outmatches that of the MLP. For 64 training samples of the flow at $Re = 1000$, the PINN outperforms the MLP by an order of magnitude. However, at yet higher numbers of training samples, the PINN begins to lose its advantage over the MLP, and the difference in validation loss between the PINN and the MLP begins to decrease—at 2048 training samples, the relative performance of the PINN and the MLP is much less.

Looking at the residual loss might paint a picture as to why the above phenomena is observed. The residual loss term incorporates the x and y derivatives of (u, v, p). Yet, the actual calculation of the derivative is not necessarily accurate. The neural network, after all, acts as a function approximator of the 'true' function, hence the derivative terms are similarly merely approximate derivatives of the 'true' differential terms. This explains why the residual loss term increases as the number of training samples increase, rather than decrease as expected: the neural network structure imposes an upper limit on the actual



(a) Exact pressure distribution

(b) PINN pressure recovery

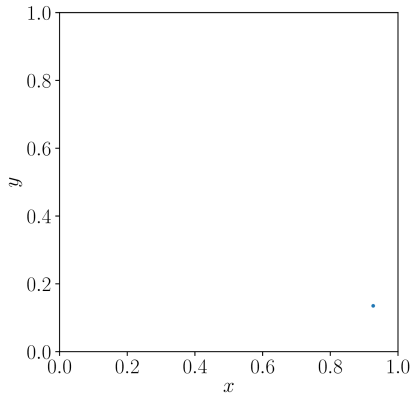
Figure 5.4: Comparison between exact pressure field and pressure field predicted by the PINN trained without referencing the pressure field, for a flow regime $Re = 2000$. Note the different scaling along the colour bars for both plots, indicating a constant error between both pressure fields. (a) Exact pressure distribution. (b) Pressure distribution recovered using a PINN.

achievable accuracy of the differential equations within the neural network.

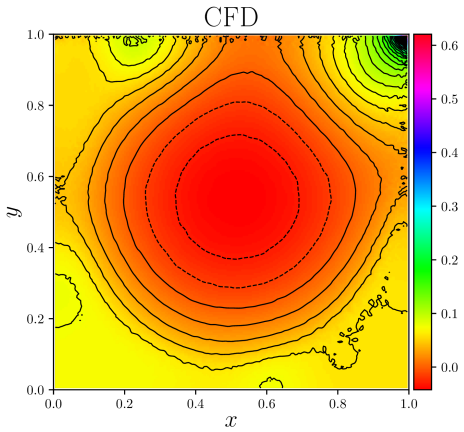
5.4 Pressure Recovery Using Physics-Informed Neural Network

This experiment tests the ability of the PINN to recover the pressure field when trained without observing it. First, a PINN is trained replacing MSE_0 in equation (3.12) with $MSE_{0,\,new}$ as defined in equation (4.1). This PINN is trained using 256 randomly sampled points within the cavity for a flow regime of $Re = 2000$. Figure 5.4 compares the original exact pressure field from CFD versus the pressure field recovered by the PINN for the flow regime of $Re = 2000$. The pressure distribution plots are remarkably similar, but for a constant difference between the two plots. This result is in good agreement with Raissi et al.'s findings.

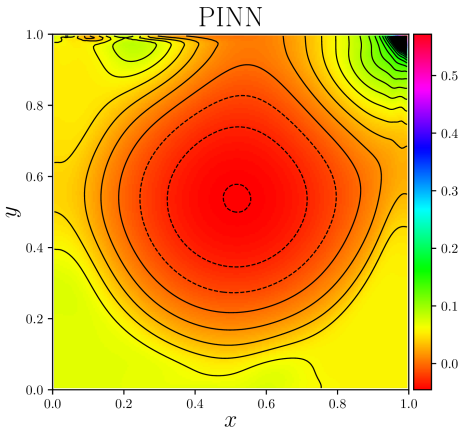
Subsequently, the PINN is trained in a $Re = 4000$ flow regime. As before, 256 points in the cavity are used for the training. A single point in the pressure field is randomly selected for the calculation of the MSE as given in



(a) Location of singular reference point within the pressure field.



(b) Exact pressure distribution



(c) PINN pressure distribution

Figure 5.5: Comparison between exact pressure field and pressure field predicted by the PINN trained with one reference point in the pressure field, for a flow regime $Re = 4000$. (a) Location of the reference point. (b) Exact pressure distribution. (c) Pressure distribution recovered using a PINN.

equation (4.2). Figure 5.5 compares the PINN predicted pressure field against the exact pressure for a flow in $Re = 4000$. The pressure field is able to be recovered to a qualitatively high level of accuracy using a PINN trained using just one reference point for the pressure field.

Additional figures for pressure recovery are included in Appendix B for completeness. The PINN shows excellent promise in pressure field recovery. When trained without any pressure reference, the PINN is able to return a qualitatively accurate representation of the exact pressure, albeit off by a

constant. When trained with a single pressure point, the PINN is able to recover the exact pressure field.

5.4.1 Analysis of Pressure Recovery

The present work is able to replicate the results of Raissi et al. in accurately recovering the pressure field of the lid-driven cavity flow when trained without observing pressure field data. This confirms the postulate laid down by Raissi et al. [1]. Raissi et al. rightfully point out that the pressure within a CFD-generated flow field that relies on the Navier–Stokes equations can only be accurate up to a constant [1], hence consider their version of the PINN recovery an adequate demonstration of the ability of their PINN to estimate of the true pressure field within the flow.

The above testing is able to take Raissi et al.'s work one step further by considering an anchoring reference point from which the rest of the pressure field can be recovered. This is born from the hypothetical scenario wherein a PINN is used to model an empirical flow field with empirically obtained velocity and pressure data. In such a scenario, the quantitatively accurate recovery of the pressure field is of great scientific merit. The above testing demonstrates that the PINN has the ability to accurately recover the entire pressure field with as little as a single reference point. This highlights the potential applications of PINNs for generating quantitatively accurate predictions within super low data regimes and signifies the possible use of PINNs in unsupervised learning.

Chapter 6

Conclusion and Future Work

6.1 Conclusion

In this study, a PINN approach to modelling the lid-driven cavity flow is compared against an MLP approach to the lid-driven cavity flow problem. In the lid-driven cavity flow, flow obeys the incompressible steady-state Navier–Stokes equations. Flow field data for a wide range of Reynolds numbers are obtained using OpenFOAM CFD. This data is then validated against existing literature.

Subsequently, the CFD data is used as the basis for training a data-driven MLP as well as a PINN. The data-driven MLP is a naive neural network which minimises the MSE between the predicted and actual flow field values. The PINN is a neural network which minimises the deviation from the Navier–Stokes equations in addition to the MSE between the predicted and actual flow values. This is achieved through defining a custom loss function that encodes the Navier–Stokes equations using TensorFlow's automatic differentiation capabilities, then minimising this custom loss function during the training process.

The merits of the PINN approach to designing ML models for solving of scientific programming problems have been adequately demonstrated through this work. Within a small-data regime, PINNs demonstrate better predictive accuracy both quantitatively and qualitatively compared to the data-driven

MLP approach.

Recovery of the pressure field using a PINN also shows great promise and excellent practicality. The PINN is able to qualitatively recover the pressure field when trained without seeing any pressure data, and is further able to fully recover the pressure field when trained with a single reference point. This has excellent potential use in scientific computing, when there is a need to accurately model a pressure field when very little training data is available. o]

6.2 Weaknesses of Physics-Informed Neural Networks

While PINNs have clear advantages over their data-driven counterparts, this project has revealed some weaknesses of the PINN.

One weakness of the PINN is identified in Chapter 5. The differential terms are—similarly to the neural network itself—only approximate representations of the true underlying mathematics. This places a constraint on the maximum achievable accuracy of the trained model. Furthermore, the PINN does have practical disadvantages compared with the MLP. Due to the nature of a PINN requiring a callback to the gradients at every iteration, the training time of a PINN could end up taking much longer than an MLP as each epoch takes significantly more computing power than the simple back propagation of gradients performed during each step of the MLP training process.

When the size of the sampling data is sufficiently large, the above factors in combination make the choice between a PINN and an MLP rather more complicated.

6.3 Recommendations for Future Research

6.3.1 Relative Weighting of Mean Squared Error Components

The MSE used to train the PINN (defined in equation (3.12) encodes both the MSE of the actual output values (defined as MSE_0) and the MSE of the Navier–Stokes equations. However, less intuitively, this definition of MSE implicitly assigns a relative weight towards each MSE component—in this case, the components are equally weighted, or to be rather more precise, the components are not explicitly weighted.

Changing the relative weighting of each MSE component would lead to the model being attuned to the highest weighted component. As an explanatory example, by increasing the relative weight of MSE_m, the PINN will attempt to train itself to best align with the requirement for the conservation of momentum throughout the cavity.

Theoretically, all MSE components should approach 0 as the PINN increases in real accuracy measured in validation MSE. However, the PINN remains an imperfect approximator of the true underlying function and as a result the TensorFlow Gradient terms are imperfect approximators of the true derivatives. It will be interesting to study how varying the relative weights of each MSE component will affect overall model accuracy.

6.3.2 Boundary Conditions

One physical condition not encoded by the current PINN are the boundary conditions of the lid-driven cavity. These boundary conditions include the driving velocity at the top of the cavity, as well as the no-penetration condition at the boundary walls. This could further improve the PINN. Looking at Figure 5.2, the failure of the PINN to properly account for velocities along the

cavity edge is evident. Enforcing the boundary conditions within the PINN could possible improve the training of the PINN further.

6.3.3 Unsupervised Learning

If the boundary conditions are taken into account, it may be possible to generate the flow field entirely unsupervised or almost entirely unsupervised. Taking inspiration from the PINN's ability to recover the pressure field without seeing the pressure data, it might be possible to apply the same logic to the entire flow field. This possibility warrants future investigation, because if it were possible to obtain the lid-driven cavity flow using a PINN trained using unsupervised learning, then an entirely new paradigm for flow field predictions could be realised.

6.3.4 Alternative Forms of the Navier–Stokes Equations

One possible modification to the present work entails encoding a different form of the Navier–Stokes equations. An example is the vorticity-streamfunction formulation. Such a formulation could potentially have different effects on the final accuracy of the trained PINN.

6.3.5 Applying Physics-Informed Neural Networks to Recovering Incomplete Data Sets

Following the successful recovery of the data in the pressure field, an additional potentially useful application of a PINN is in the recovery of incomplete data from a set of readings. Real world data collection is not always consistent, and sensor or human error could lead to corruption of certain parts of data within a larger stream. A well-designed PINN could possible achieve excellent recovery of missing data, if required.

Bibliography

[1] M. Raissi, P. Perdikaris, and G. 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, 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0021999118307125

[2] E. Erturk, T. C. Corke, and C. Gökçöl, "Numerical solutions of 2-D steady incompressible driven cavity flow at high Reynolds numbers," *International Journal for Numerical Methods in Fluids*, vol. 48, no. 7, pp. 747–774, 2005. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/fld.953

[3] 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. [Online]. Available: https://www.sciencedirect.com/science/article/pii/0021999182900584

[4] R. Temam, *Navier–Stokes Equations: Theory and Numerical Analysis*, 2nd ed., J. L. Lions, G. Papanicolaou, and R. T. Rockafellar, Eds. North Holland, 1979.

[5] J. Tu, G. H. Yeoh, and C. Liu, *Computational Fluid Dynamics*, 3rd ed. Butterworth-Heinemann, 2019.

[6] S. L. Brunton, B. R. Noack, and P. Koumoutsakos, "Machine learning for fluid mechanics," *Annual Review of Fluid Mechanics*,

vol. 52, no. 1, pp. 477–508, 2020. [Online]. Available: https://doi.org/10.1146/annurev-fluid-010719-060214

[7] D. C. Psichogios and L. H. Ungar, "A hybrid neural network-first principles approach to process modeling," *AIChE Journal*, vol. 38, no. 10, pp. 1499–1511, 1992. [Online]. Available: https://aiche.onlinelibrary.wiley.com/doi/abs/10.1002/aic.690381003

[8] M. Raissi, A. Yazdani, and G. E. Karniadakis, "Hidden fluid mechanics: A Navier–Stokes informed deep learning framework for assimilating flow visualization data," 2018.

[9] M. Raissi and G. E. Karniadakis, "Hidden physics models: Machine learning of nonlinear partial differential equations," *Journal of Computational Physics*, vol. 357, pp. 125 – 141, 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0021999117309014

[10] D. Kochkov, J. A. Smith, A. Alieva, Q. Wang, M. P. Brenner, and S. Hoyer, "Machine learning accelerated computational fluid dynamics," 2021.

[11] Y. Zhu, N. Zabaras, P.-S. Koutsourelakis, and P. Perdikaris, "Physics-constrained deep learning for high-dimensional surrogate modelling and uncertainty quantification without labelled data," *Journal of Computational Physics*, vol. 394, pp. 56 – 81, 2019. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0021999119303559

[12] R. Tipireddy, P. Perdikaris, P. Stinis, and A. Tartakovsky, "A comparative study of physics-informed neural network models for learning unknown dynamics and constitutive relations," 2019.

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

[14] A. D. Jagtap, K. Kawaguchi, and G. E. Karniadakis, "Adaptive activation functions accelerate convergence in deep and physics-informed neural networks," *Journal of Computational Physics*, vol. 404, p. 109136, Mar 2020. [Online]. Available: http://dx.doi.org/10.1016/j.jcp.2019.109136

[15] Y. Yang and P. Perdikaris, "Adversarial uncertainty quantification in physics-informed neural networks," *Journal of Computational Physics*, vol. 394, pp. 136 – 152, 2019. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0021999119303584

[16] T. Mitchell, *Machine Learning*. McGraw Hill, 1997.

[17] M. Abadi *et al.*, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015, software available from tensorflow.org. [Online]. Available: https://www.tensorflow.org/

[18] F. Chollet *et al.*, "Keras," https://keras.io, 2015.

[19] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2017.

[20] M. Alleto. SimpleFoam. [Online]. Available: https://openfoamwiki.net/index.php/SimpleFoam

[21] D. C. Liu and J. Nocedal, "On the limited memory BFGS method for large scale optimization," *Mathematical Programming*, vol. 45, pp. 503 – 528, 1989.

Appendix A

Derivations

A.1 Derivation of MSE for PINN

The derivation of the various components of *MSE* in Equation 3.12 may be of interest to the reader. We start from Equation 3.4 and Equation 3.5:

$$\nabla \cdot \mathbf{u} = 0 \tag{A.1}$$

$$\mathbf{u} \cdot \nabla \mathbf{u} = -\nabla p + \frac{1}{Re} \nabla^2 \mathbf{u} \tag{A.2}$$

Equation A.1 simply becomes:

$$\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} = 0 \tag{A.3}$$

$$u_x + v_y = 0 \tag{A.4}$$

On the other hand, equation (A.2) can be split into horizontal and vertical momentum components:

$$u \cdot \frac{\partial u}{\partial x} + v \cdot \frac{\partial u}{\partial y} = -\frac{\partial p}{\partial x} + \frac{1}{Re}(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}) \tag{A.5}$$

$$u \cdot \frac{\partial v}{\partial x} + v \cdot \frac{\partial v}{\partial y} = -\frac{\partial p}{\partial y} + \frac{1}{Re}(\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2}) \tag{A.6}$$

Rearranging:

$$u \cdot u_x + v \cdot u_y + p_x - \frac{1}{Re}(u_{xx} + u_{yy}) = 0 \tag{A.7}$$

$$u \cdot v_x + v \cdot v_y + p_y - \frac{1}{Re}(v_{xx} + v_{yy}) = 0 \tag{A.8}$$

Equations (A.7) and (A.8) then become MSE_{mx} and MSE_{my}, respectively.

Appendix B

Additional Tables and Diagrams

B.1 Small-Data Sampling Regime

Table B.1 records the validation MSE of both the PINN and the MLP, as well as the residual MSE of the PINN, for the flow at $Re = 8000$.

Notably, the validation MSE for the PINN is slightly higher than that of the MLP trained using 1024 samples. To see the effect of this, the streamlines of the PINN and MLP trained using 1024 samples are plotted in Figure B.1. A qualitative look at the streamlines for both predictions indicates that they have their strengths and weaknesses. The top left secondary vortex is more accurately modelled by the PINN output. However, the flow along the bottom

Table B.1: Mean Squared Error for $Re = 8000$

Training samples	**PINN MSE**	**Residual MSE**	**MLP MSE**
8	2.96×10^{-2}	2.10×10^{-16}	4.63×10^{-2}
16	1.68×10^{-2}	2.29×10^{-14}	4.78×10^{-2}
32	8.55×10^{-3}	9.54×10^{-12}	1.40×10^{-2}
64	2.24×10^{-3}	7.89×10^{-9}	6.87×10^{-3}
128	1.36×10^{-3}	1.55×10^{-7}	3.14×10^{3}
256	1.08×10^{-3}	2.14×10^{-6}	1.35×10^{-3}
512	1.96×10^{-4}	1.06×10^{-5}	6.15×10^{-4}
1024	1.93×10^{-4}	1.97×10^{-5}	1.64×10^{-4}
2048	1.15×10^{-4}	2.49×10^{-5}	1.01×10^{-4}

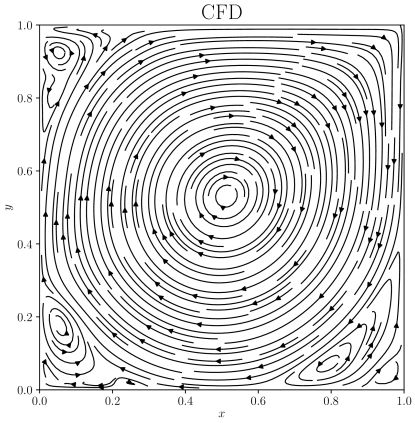
of the cavity is modelled more accurately by the MLP. Both approaches are unable to accurately model flow near the edges of the cavity, where the no-penetration condition should apply.[1] Interestingly, while the PINN trained using 256 samples was able to predict the appearance of a tertiary vortex at the lower left corner of the cavity, the PINN trained using 1024 samples appears oblivious to this. Overall, it is difficult to qualitatively judge the performance of the PINN looking at the streamlines alone. This observation lends some merit to the notion that the validation MSE alone may not necessarily be a good comparative indicator of the performance of either model, especially when the difference between the PINN and the MLP are very marginal.

B.2 Pressure Recovery

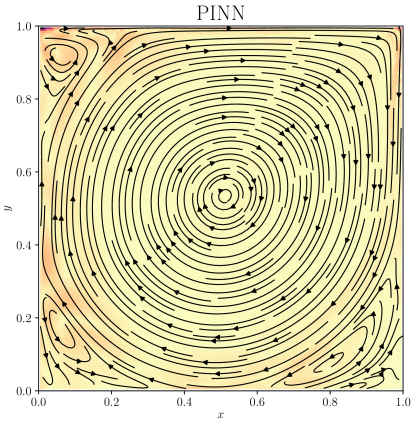
Figure B.2 shows the pressure recovery plot when the PINN is trained using 256 samples at $Re = 2000$ and with one reference point. The figure demonstrates good pressure recovery by the PINN.

Figure B.3 shows the pressure recovery for the PINN trained using 256 samples without any reference point in the pressure field. Interestingly, we observe a very good recovery of the original pressure field, despite the PINN not seeing any pressure data during the training process. This is due to the stochastic nature of the training process without any input data—there is, of course, some chance that the PINN is able to fully recover the pressure field without seeing it, and this is a good demonstration of that phenomena. It is worth pointing out that this is not the general case.

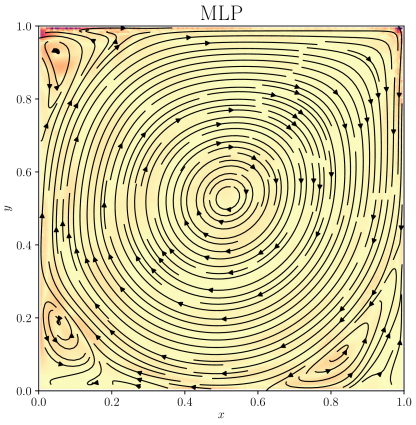
[1] Streamlines should be parallel to the cavity walls at the boundaries. Encoding these boundary conditions into the PINN loss function could well improve its predictive performance, which is the motivation for the suggestion in Chapter 6.



(a) Streamlines in original CFD.

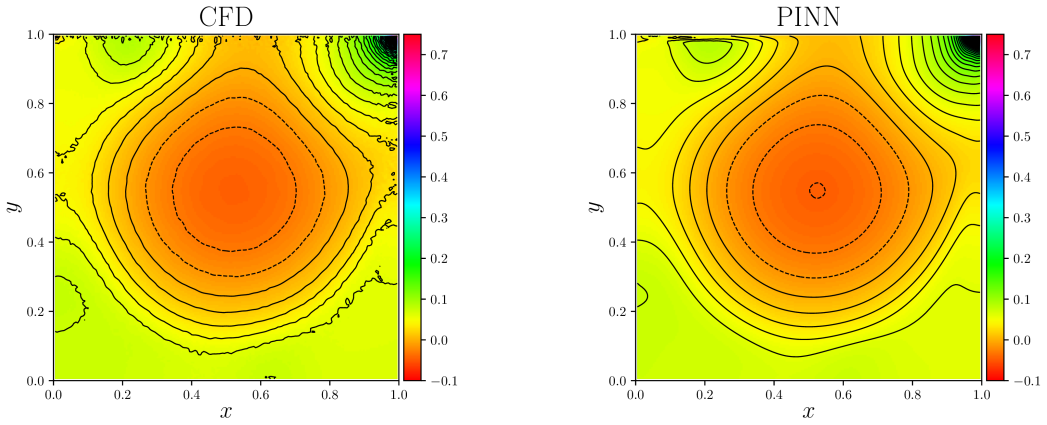


(b) Streamlines for PINN trained with 1024 sampling points.



(c) Streamlines for MLP trained with 1024 sampling points.

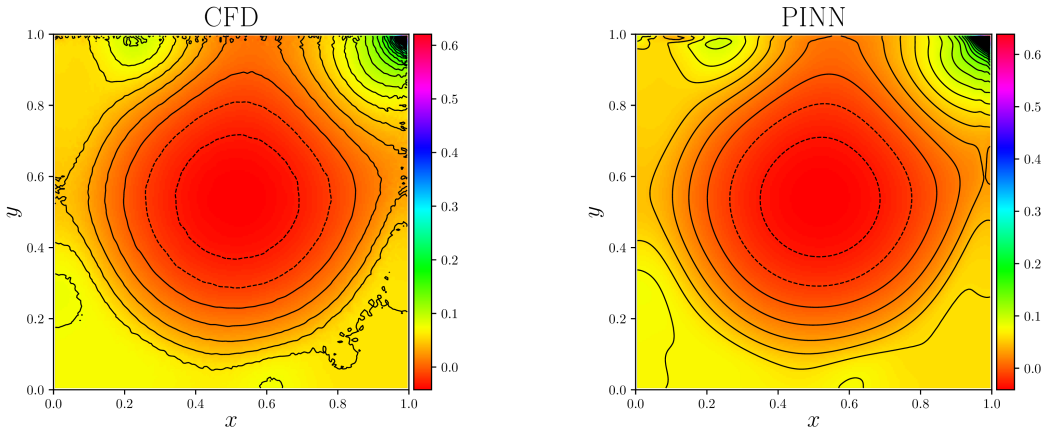
Figure B.1: Comparison between original streamline plots and ML-derived streamline plots for the lid-driven cavity at $Re = 8000$. The MSE of the velocity vector $\mathbf{u}$ at each spatial point is demarcated by the colour. (a) Original streamline plot. (b) PINN, 1024 samples. (c) MLP, 1024 samples.



(a) Exact pressure distribution

(b) PINN pressure distribution

Figure B.2: Comparison between exact pressure field and pressure field predicted by the PINN trained with one reference point in the pressure field, for a flow regime $Re = 2000$. (a) Location of the reference point. (b) Exact pressure distribution. (c) Pressure distribution recovered using a PINN.



(a) Exact pressure distribution

(b) PINN pressure distribution

Figure B.3: Comparison between exact pressure field and pressure field predicted by the PINN trained without any reference in the pressure field, for a flow regime $Re = 4000$. (a) Location of the reference point. (b) Exact pressure distribution. (c) Pressure distribution recovered using a PINN.