Proceedings of the 4th ITB Graduate School Conference
Innovation and Discovery for Sustainability
July 6, 2023

Development of Pressure Estimator and Velocity Field Corrections for Particle Image Velocimetry Using Physics-Informed Neural Network

Calvin Christian Chandra*, Luqman Fathurrohim, Pramudita Satria Palar & Lavi Rizki Zuhal

Flow Diagnostics Laboratory, Faculty of Mechanical and Aerospace Engineering, Institut Teknologi Bandung, Jalan Ganesa 10, Bandung 40132, Indonesia
*Email: calvincc5700@gmail.com

Abstract. Flow diagnostics using particle image velocimetry (PIV) has always been a viable option, but errors or faults in the experiment can lead to misinterpreted data. Meanwhile, physics-informed neural network (PINN) usage has been on the rise because of its versatility. This work intends to analyze the possibilities of implementing PINN for PIV and test on a couple of flow cases to observe whether misinterpretations in PIV output can be minimized, along with providing pressure prediction in the analysis domain. This work modifies an already existing PINN program to better suit PIV applications which is then implemented on uniform flow and backstep flow. The PINN was tested on uniform flow and backstep flow which show that the PINN can produce a denser velocity prediction and predict a pressure field without any prior pressure data. Also, it is capable of filling in gaps of missing data and correct invalid velocity data. results for individual cases are satisfactory for both velocity and pressure predictions but can be improved further.

Keywords: *deep learning; flow pressure estimator; flow velocity field corrector; particle image velocimetry; physics-informed neural network.*

1 Introduction

There is a branch of aerospace engineering which is flight physics with one concern of how fluid motion affects a body. Assessment of these motions is necessary to obtain relevant output, state of the flow, and flow phenomenon that are present in each case. The assessment methods can be categorized into 3 groups which are analytical, experimental, or numerical analysis. Focusing on experimental fluid motion analysis, several methods can be implemented with one category being optical flow analysis which is based on the interaction (refraction, scattering, etc.) of light with inhomogeneous media [1]. One common optical flow experiment that has been around for decades is particle image velocimetry (PIV) which is a quantitative visualization technique by seeding the fluid with clearly distinguishable and structured tracer particles to non-intrusively

follow the flow [2]. Given how established PIV is, a lot of analysis methods are already available, but a more recent approach is to implement physics-informed neural network (PINN) with aim to be able reconstruct a dense velocity field based on experimental data that has sparse flow motion information and estimate a flow pressure field without any prior pressure data. In addition, the problem regarding errors in PIV experiments will be assessed. These errors are introduced in numerous ways, with common examples such as invalid measurements, relative uncertainty, measurement error (includes systematic and random errors), etc. [2]. The results of these errors vary but one being outlier velocity vectors that have incorrect value and/or direction (as depicted in **Figure 1**).

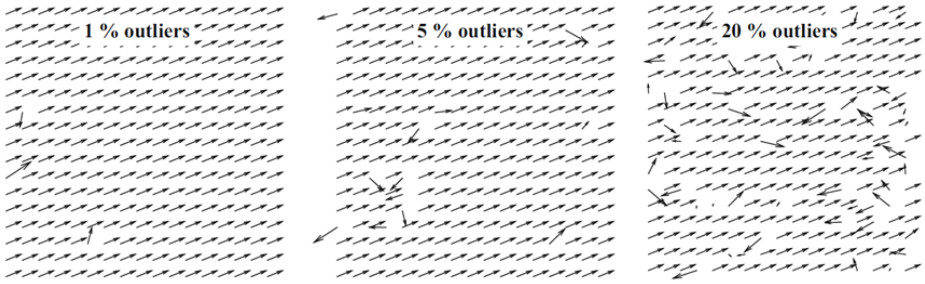


Figure 1 Vector fields with different amounts of invalid measurements [2].

Another problem relating with PIV are regions which cannot be evaluated called gaps [3]. This can be caused by reasons such as limited access for flow illumination, shadows from obstruction, etc. causing missing velocity vectors. These invalid or missing velocity vectors are expected to be corrected or filled in by PINN. Furthermore, the PINN can be used to produce pressure field estimations, useful to substitute common methods of pressure measurements which installation are not always convenient or even impossible.

2 Physics-Informed Neural Network

PINN is a framework that combines data-driven neural networks and physical laws. The neural network functions as a universal function approximator, meanwhile the physics laws are connected through partial differential equations (PDEs) into the loss function of the neural network. This will result the PINN able to integrate known information (boundary conditions, initial conditions, and measured data) [4]. One example of PINN architecture is depicted in **Figure 2** implementing a residual neural network approach with the relation of input and output as follows.

$$(\mathbf{U}, p) = F_{NN}(\mathbf{X}, t; \mathbf{\Theta}) \tag{1}$$

Eq. (1) adduce the neural network (F_{NN}) with trainable variables ($\boldsymbol{\Theta}$) will have an input of space coordinates ($\mathbf{X}$) and time (t). Which will then have an output of velocity vector ($\mathbf{U}$) and pressure (p).

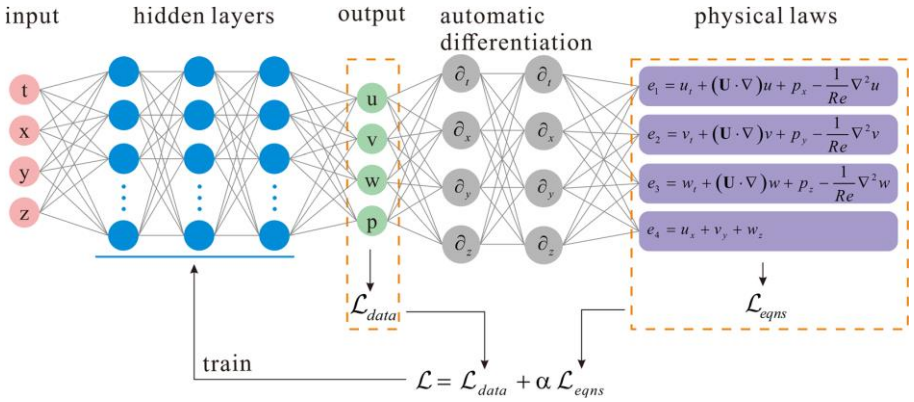


Figure 2 PINN architecture using residual neural network and incompressible navier-stokes equations [4].

Before using PINN, it initially needs to be trained to configure its parameters which are iteratively updated by minimizing the loss function. The loss function of the PINN combines the data and physical loss. In this case, the data of PIV output are available, and the physical loss will come from the PDE calculations at randomly distributed points (namely collocation points) throughout the domain. Therefore, the total loss function will be described as follows.

$$J_{Total} = J_g + \alpha J_d \tag{2}$$

J_{Total} being the total loss, J_g is the PDE loss, J_d is the data loss, and α is the loss weighting coefficient to control the trade-off between J_g and J_d

$$J_g = \frac{1}{N_g}\sum_{i-1}^{N_g}\left(\mathbf{f}(\boldsymbol{x}^i)\right)^2 \tag{3}$$

$$J_d = \frac{1}{N_d}\sum_{i-1}^{N_d}\left(\mathbf{u}^{\mathrm{i}}_{pred} - \mathbf{u}^{\mathrm{i}}_{d}\right)^2 \tag{4}$$

N_g and N_d being the number of collocation points and data points respectively, $\mathbf{f}$ being the residual of predicted value compared to the analytical PDE solution, and $\mathbf{u}_{pred}$ and $\mathbf{u}_{\mathrm{d}}$ are the predicted values from the neural network and defined data values, respectively.

In solving fluid flow problems, the PDEs are the Navier-Stokes equations and 2 formulations of it will be referred in this work. First, the 2-dimensional steady, incompressible form in velocity-pressure formulation is addressed as follows.

$$\begin{aligned} \rho(\boldsymbol{u}\cdot\nabla)\boldsymbol{u} &= -\nabla p + \mu\nabla^2\boldsymbol{u} && in\ \Omega \\ \nabla\cdot\boldsymbol{u} &= 0 && in\ \Omega \\ \boldsymbol{u} &= \boldsymbol{u}_\Gamma && in\ \Gamma_{\mathrm{D}} \\ \partial_n\boldsymbol{u} &= 0 && in\ \Gamma_{\mathrm{N}} \end{aligned} \tag{5}$$

With $\boldsymbol{u}$ defined as the velocity vector, p is pressure, μ is the fluid's viscosity, and ρ is the fluid's density. Also, there are the Dirichlet and Neumann boundary conditions denoted by Γ_D and Γ_N respectively.

Second, the 2-dimensional steady, incompressible form in Cauchy stress tensor formulation is addressed as follows with the addition of Cauchy stress tensor, σ.

$$\begin{aligned} \rho(\boldsymbol{u}\cdot\nabla)\boldsymbol{u} &= -\nabla\cdot\sigma && in\ \Omega \\ \sigma &= -p\boldsymbol{I} + \mu(\nabla\boldsymbol{u} + \nabla\boldsymbol{u}^T) && in\ \Omega \\ \nabla\cdot\boldsymbol{u} &= 0 && in\ \Omega \\ \boldsymbol{u} &= \boldsymbol{u}_\Gamma && in\ \Gamma_{\mathrm{D}} \\ \partial_n\boldsymbol{u} &= 0 && in\ \Gamma_{\mathrm{N}} \end{aligned} \tag{6}$$

Both formulations are then used to form the residuals of the PINN prediction by means of rearranging the momentum and continuity equations, then applying for PINN prediction results. The resulting residual formulation for both Navier-Stokes formulations are listed in **Eq. (7)** for velocity-pressure and **Eq. (8)** for stress tensor.

$$\begin{aligned} g_u &= u\partial_x u + v\partial_y u + \frac{1}{\rho}\partial_x p - \frac{\mu}{\rho}\left(\partial_{xx}u + \partial_{yy}u\right) \\ g_v &= u\partial_x v + v\partial_y v + \frac{1}{\rho}\partial_y p - \frac{\mu}{\rho}\left(\partial_{xx}v + \partial_{yy}v\right) \end{aligned} \tag{7}$$

$$\begin{aligned} g_u &= u\partial_x u + v\partial_y u - \frac{\mu}{\rho}\left(\partial_x\sigma_{xx} + \partial_y\sigma_{xy}\right) \\ g_v &= u\partial_x v + v\partial_y v - \frac{\mu}{\rho}\left(\partial_x\sigma_{yx} + \partial_y\sigma_{yy}\right) \\ g_{\sigma_{xx}} &= -p + \mu(2\partial_x u) \\ g_{\sigma_{yy}} &= -p + \mu\left(2\partial_y v\right) \\ g_{\sigma_{xy}} &= \mu\left(\partial_y u + \partial_x v\right) \\ g_p &= p - \frac{1}{2}\left(\sigma_{xx} + \sigma_{yy}\right) \end{aligned} \tag{8}$$

3 Methods and Flow Test Cases

3.1 Methods

This work will implement 2 base programs. First, To obtain the velocity vector data, some of the code in the program created by Octavianus [5] will be utilized which will analyze the pair particle images, create an array output containing the information of velocity in the x- and y- direction, as well as visualizing the velocity vectors in a quiver plot. Next, the PINN program created by Putra [6] is modified so that it can take the velocity vectors as its input rather than boundary conditions, hence making it suitable for PIV applications. Two cases will be assessed in this work which are uniform flow and backstep flow with the general flowchart of the program depicted in **Figure 3**.

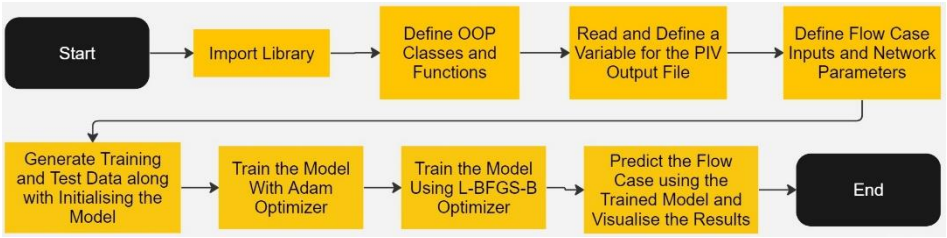


Figure 3 General Flowchart of the PINN Program for PIV Implementation

3.2 PIV Uniform Flow

The referred uniform flow has an average velocity of around -0.3 m/s in the x-axis and -4.3 m/s in the y-axis which is visualized in **Figure 4**. Additionally, the PINN parameters for this case can be seen in **Table 1**.

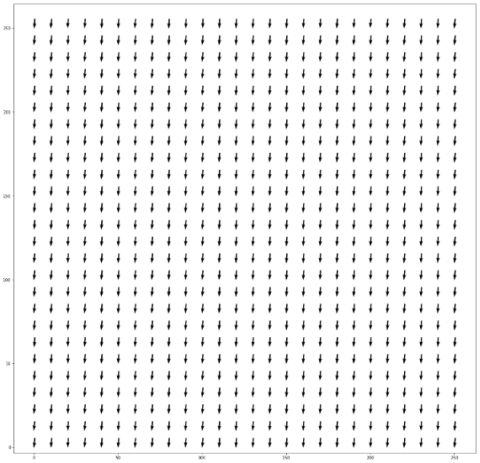


Figure 4 PIV uniform flow visual representation.

Table 1 PINN program parameters for PIV uniform flow.

General Parameter	Value
Number of Collocation Points	5000
Number of Rows	256
Number of Columns	256
Number of Hidden Layers	4
Number of Perceptrons per Hidden Layer	50
Loss Weighting Coefficient, α	1
Domain Size (m)	3×3

There will be 2 variations for the PINN analysis. The first is analysis without any prior alteration to the input velocity data. Meanwhile the second will have data group alterations to simulate major misinterpretations from the PIV experimentation. Implemented alterations are removal of velocity data points within a boundary and changing velocity data values within a boundary. The parameters of each group alterations can be seen in **Table 2** and **Table 3**.

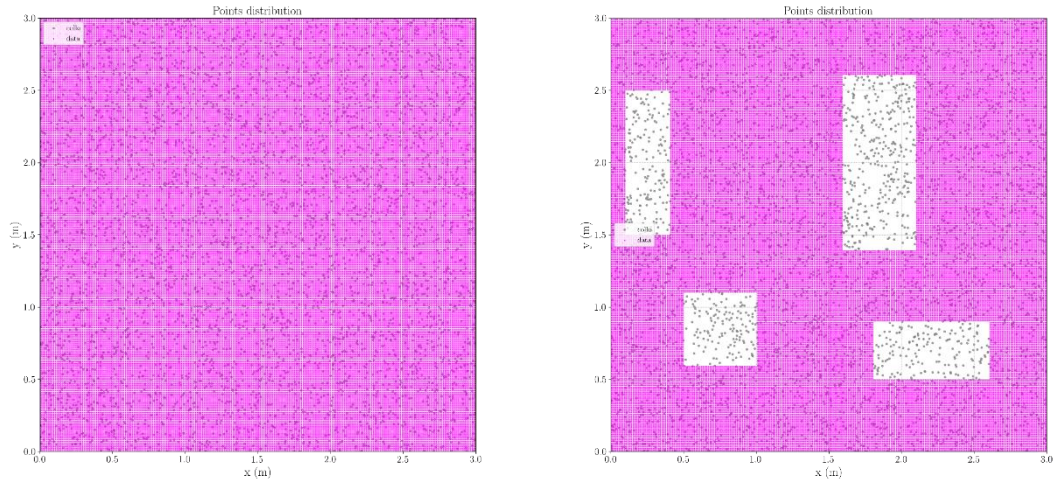
Table 2 Removed data points' boundary limits.

Cluster No.	Min. x limit [m]	Max. x limit [m]	Min. y limit [m]	Max. y limit [m]
1	0.5	1	0.6	1.1
2	1.6	2.1	1.4	2.6
3	0.1	0.4	1.5	2.5
4	1.8	2.6	0.5	0.9

Table 3 Data points' velocity change boundary limits and values.

Cluster No.	Min. x limit [m]	Max. x limit [m]	Min. y limit [m]	Max. y limit [m]	New x-Direction Velocity [m/s]	New y-Direction Velocity [m/s]
1	0.1	0.5	0.2	0.5	0.5	-
2	1.4	1.8	2.5	2.8	1.5	-
3	2.5	2.8	1.3	1.7	-1	-
4	0.3	0.6	0.1	0.4	-	0.5
5	1.2	1.6	1.5	2.1	-	-1
6	2.5	2.9	1.6	2.2	-	1.5

The PINN analysis domain for both variations can then be generated and is visualized in **Figure 5**.



(a) (b)

Figure 5 PINN analysis domain definition for PIV uniform flow (a) without altering input velocity data points, (b) with input velocity data points alterations.

3.3 PIV Backstep Flow

The referred backstep flow has Reynolds' number of 1500 and is visualized in **Figure 6**. Similar to uniform flow, this case also has 2 variations and implements PINN parameters listed in **Table 1**. The alterations also implement those listed in **Table 2** and **Table 3**, and the generated PINN analysis domain is also similar to **Figure 5**.

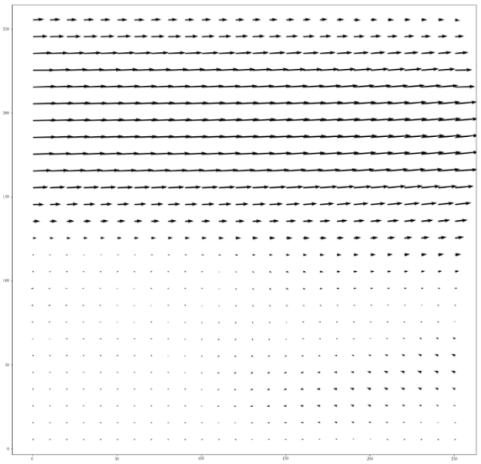
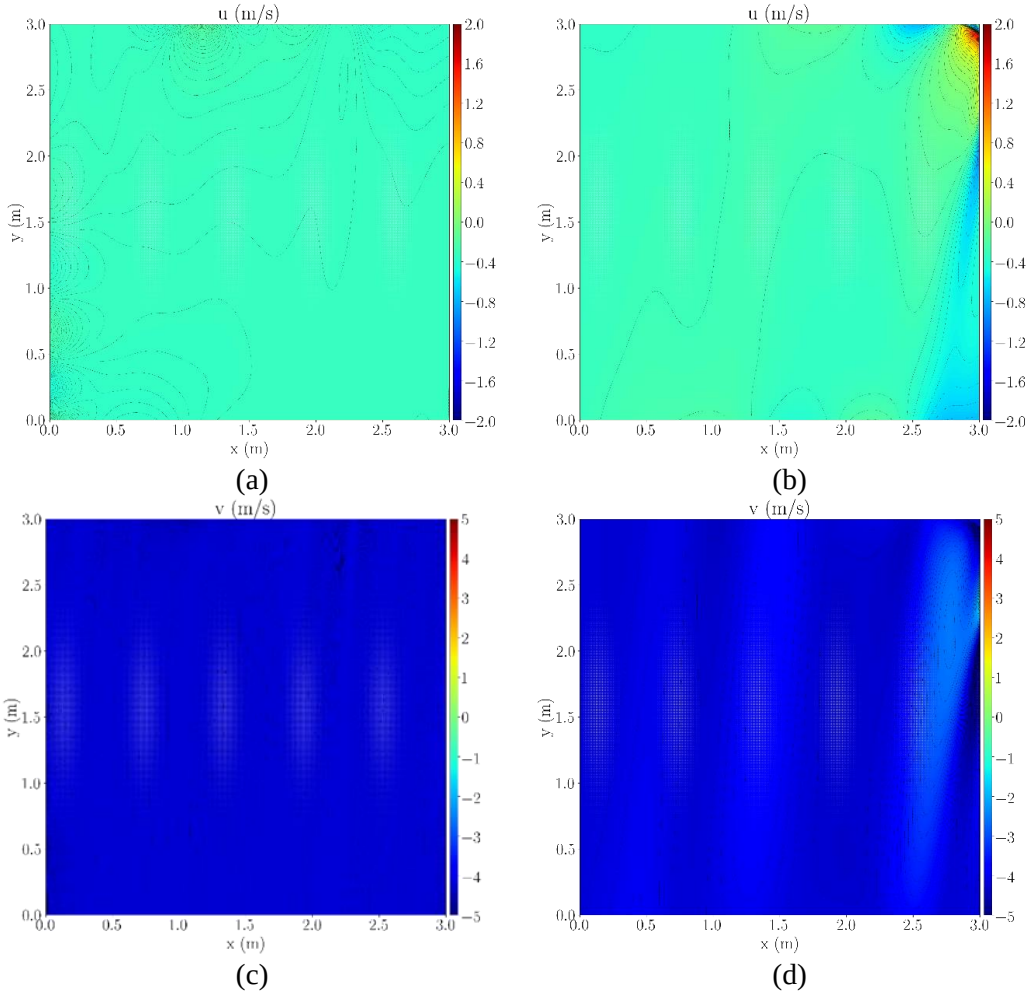


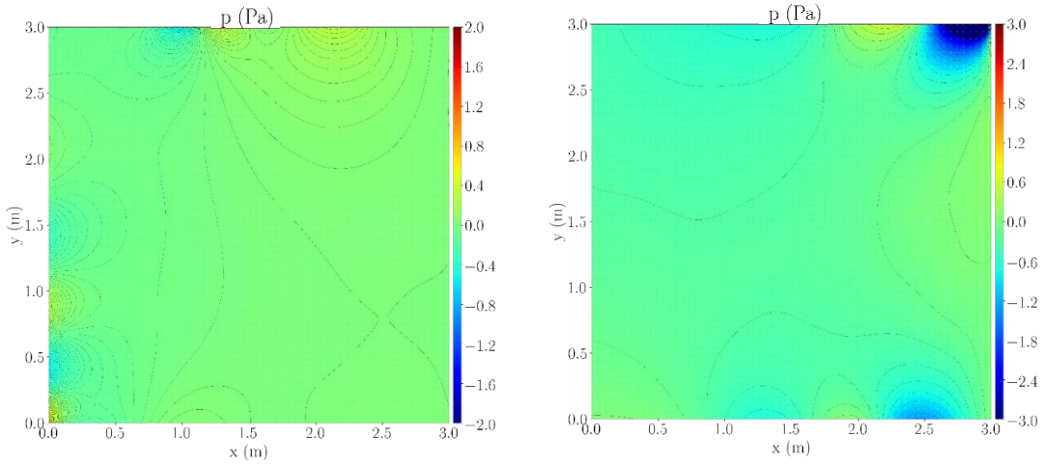
Figure 6 PIV backstep flow visual representation.

4 Results and Analysis

This section showcases the result after PINN analysis which compares the PINN capability on altered and non-altered data.

4.1 PIV Uniform Flow

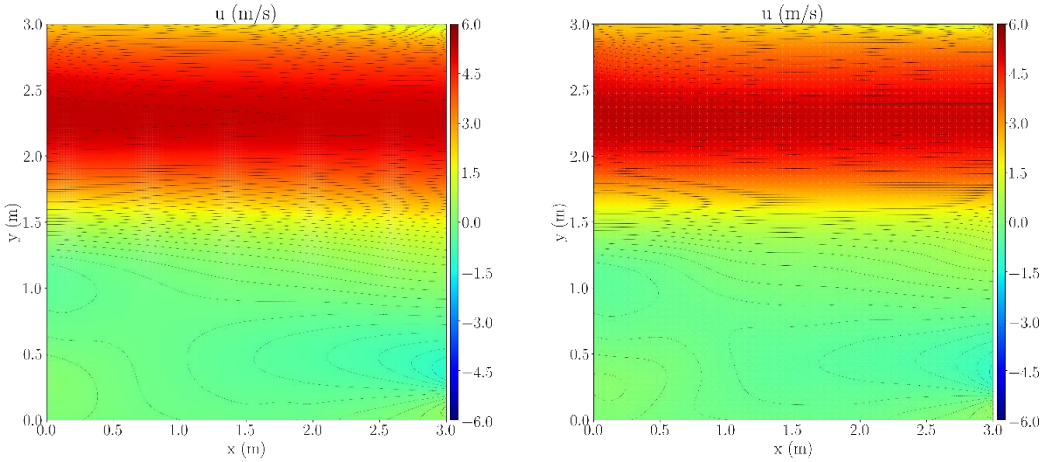




(e) (f)

Figure 7 Analyzed fields for uniform flow using PINN without velocity data alterations for (a) x-axis velocity, (c) y-axis velocity, (e) pressure, and with velocity data alterations for (b) x-axis velocity, (d) y-axis velocity, (f) pressure.

4.2 PIV Backstep Flow



(a) (b)

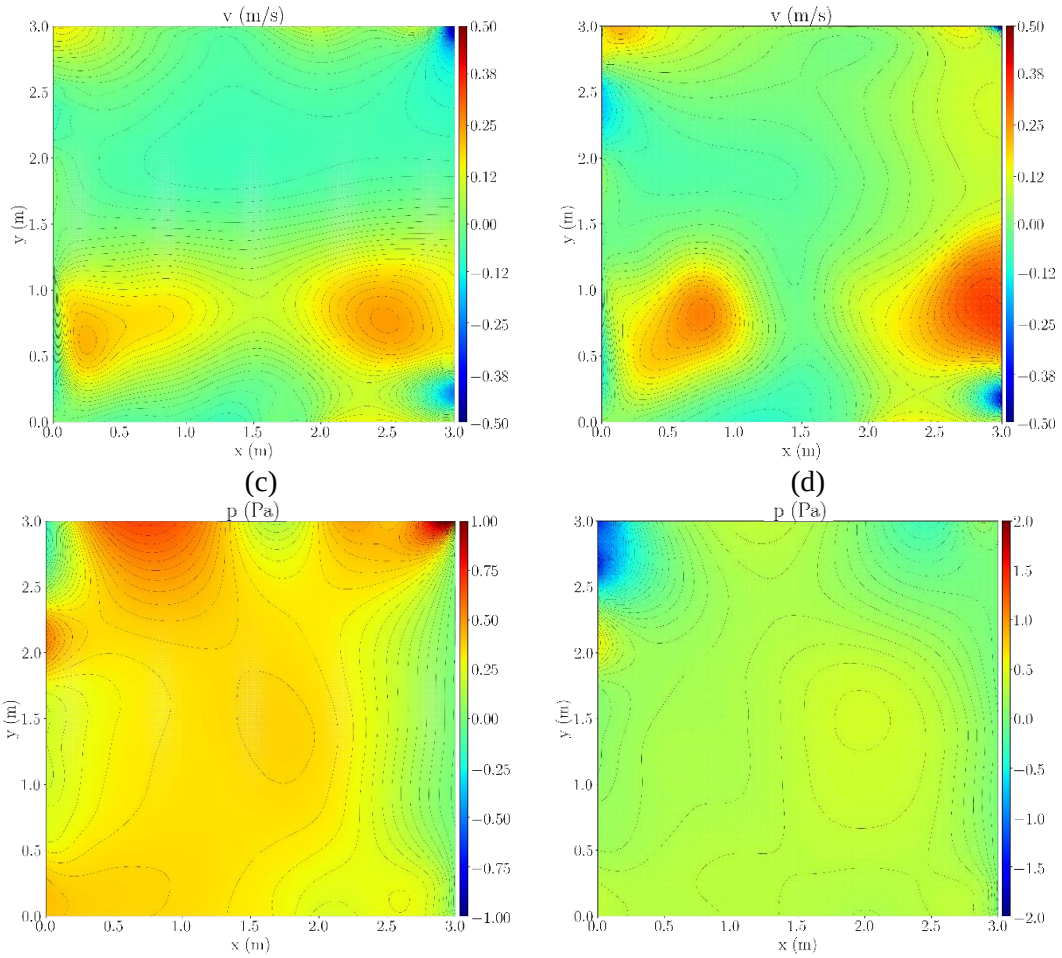


Figure 8 Analyzed fields for backstep flow using PINN without velocity data alterations for (a) x-axis velocity, (c) y-axis velocity, (e) pressure, and with velocity data alterations for (b) x-axis velocity, (d) y-axis velocity, (f) pressure.

4.3 Analysis

Referring from the results, there are several insights that can be obtained with the first being that, from part **(a)** and **(b)** of **Figure 7** and **Figure 8**, the PINN shows capability to correct defects in data by filling in gaps of data and produce outputs that overwrites prior incorrect velocity vector data. Meanwhile, part **(e)** and **(f)** of **Figure 7** and **Figure 8** reveal that a pressure field prediction is also possible to be produced without any prior pressure data. The results themselves, which are depicted as pressure fields also signify the capability of the PINN to reconstruct

a denser velocity prediction compared to the initial PIV prediction, limited by the resolution of the particle images.

From the uniform flow case, **Figure 7 (a)** and **(c)** show results that are satisfactory with uniform colors throughout the domain indicating the intended uniform flow. **Figure 7 (e)** is also considered to be appropriate because if referring to **Eq. (5)**, there is no velocity gradient in uniform flow which result in no pressure gradient and can mean constant pressure throughout the domain. Although, some pockets of pressure differences are present because of the slight imperfections in input velocity data which resulted in the flow isn't ideally uniform. Meanwhile, **Figure 7 (b)**, **(d)**, and **(f)** reveal that the significant data alterations lead to a more undesirable result which could be interpreted as minimizing errors in the input velocity vector data is still a recommended practice before further processing with PINN.

From the backstep flow case, **Figure 8 (a)** show accordance with general backstep (such as shown in **Figure 9**) with distribution of the jet velocity having the largest at the middle and dissipates as it gets further. Also, it captures the backward flow caused by induced rotational flow caused by the jet. **Figure 8 (c)** is also in accordance with the jet naturally 'fall' due to gravity and thus have negative y-axis velocity. Meanwhile, **Figure 8 (e)** when compared to **Figure 10** show that pressure prediction for the backstep flow does share similarities, but can be improved. On the other hand, **Figure 8 (b)**, **(d)**, and **(f)** does further suggest that errors in very high amounts result in a more undesirable prediction from the PINN and is recommended to minimize errors in the input velocity vector data.

Figure 9 Backstep Flow Reference Quiver Plot [7]

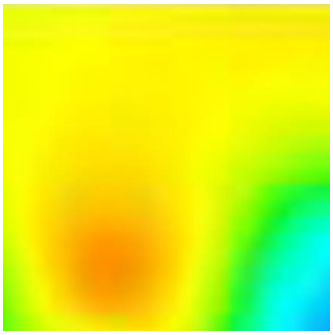


Figure 10 Backstep Flow Reference Pressure Contour [8]

5 Conclusion

The PINN is a deep learning model with implemented physical models that has the capabilities for flow diagnostics problems, one being particle image velocimetry (PIV), producing an output not only of a denser velocity prediction, but also pressure field predictions without any prior pressure data. It also has the capabilities to fill in gaps of missing data and its' likeliness to correct invalid data because of the assimilated simplified Navier-Stokes equation. The PINN's performance was tested on uniform flow and backstep flow which reveal satisfactory results for both velocity and pressure predictions but can be improved further. Lastly, alterations to the input data to simulate large quantities of error does have a negative effect towards the PINN analysis and is still recommended to minimize experiment errors prior to PINN analysis.

Nomenclature

$\mathbf{f}$	=	residual between predicted value and the analytical PDE solution
F_{NN}	=	neural network
J_d	=	data loss
J_g	=	PDE loss
J_{Total}	=	total loss function
N_d	=	number of data points
N_g	=	number of collocation points
p	=	pressure
t	=	time
$\boldsymbol{u}$	=	velocity vector
$\mathbf{u}_d$	=	defined data values
$\mathbf{u}_{pred}$	=	predicted values from the neural network
$\mathbf{U}$	=	velocity vector
$\mathbf{X}$	=	space coordinates
α	=	loss weighting coefficient
Γ_D	=	dirichlet boundary condition

Γ_N = neumann boundary condition
μ = fluid dynamic viscosity
Ω = fluid domain
ρ = fluid density
σ = cauchy stress tensor
$\boldsymbol{\Theta}$ = trainable variables

References

[1] Westerweel, J., *Fundamentals of digital particle image velocimetry*, Measurement Science and Technology, pp. 1379-1392, 1997.

[2] Raffel, M., Willert, C. E., Scarano, F., Kähler, C. J., Wereley, S. T. & Kompenhans, J., *Particle Image Velocimetry: A Practical Guide*, Cham: Springer International Publishing AG, 2018.

[3] Sciacchitano, A., Dwight, R. P. & Scarano, F., *Navier–Stokes simulations in gappy PIV data*, Experiments in Fluids, pp. 1421-1435, 2012.

[4] Wang, H., Liu, Y. & Wang, S., *Dense velocity reconstruction from particle image velocimetry/particle tracking velocimetry using a physics-informed neural network*, Physics of Fluids, **34**(1), 2022.

[5] Octavianus, F. Y., *Visual-Based Fluid Motion Estimator with Deep Learning*, unpublished.

[6] Putra, C., *On Physics-Informed Neural Network (PINN) in Solving Viscous Fluid Flow Problems*, unpublished.

[7] Zhang, M. & Piggott, M. D., *Unsupervised Learning of Particle Image Velocimetry*, Jul. 2020.

[8] Hossain, M. A., Rahman, M. T. & Ridwan, S., *Numerical Investigation of Fluid Flow Through A 2D Backward Facing Step Channel*, International Journal of Engineering Research & Technology, **2**(10), Oct. 2013.