PINN application for compressible gas flow transient analysis

SangJoon Lee
Electronics and Telecommunications Research Institute (ETRI)
Gwangju, Republic of Korea
sjlee85@etri.re.kr

Byung-Tak Lee
ETRI
Gwangju, Republic of Korea
bytelee@etri.re.kr

Seok Kap Ko
ETRI
Gwangju, Republic of Korea
softgear@etri.re.kr

***Abstract*—Simulating transient flow inside pipeline networks has been an important topic in the field of civil engineering safety. The recent development of the Physics-Informed Neural Network (PINN) shed light to solve this problem by the means of machine learning. In this paper, we apply PINN to a transient compressible fluid pipeline flow problem. We show that PINN requires a specific data normalization to preserve all necessary physics information for accurate training. We compare the PINN prediction with the estimation provided by finite difference method (FDM), which previously has been the main tool to solve such problems. As a result, we obtained a mesh-free PINN model with the difference of less than 0.6% for pressure mapping and less than 2.3% for mass flow mapping, compared to the FDM analysis.**

***Index Terms*—Physics-Informed Neural Network, Compressible fluid dynamics, Data normalization, Darcy-Weisbach equation, Mesh-free model**

I. Introduction

Analyzing fluid flow in a pipeline network has long been an important topic in the industry, from both economic and safety point of view. For example, there is an ongoing trend of digital twin implementation of gas supply lines, for the purpose of safety training and urban planning.

However, this problem is a typical example of fluid dynamics, which is one of the more difficult problems in the field of both physics and mathematics. Even if we simplify the problem by assuming that the system is turbulence-free, the transient-state analysis boils down to solving system of partial differential equations, which is often challenging in practice.

Early studies on this problem was done by Stoner [1] and Yow [2] using classical method of characteristics. Choy et al. [3] showed that finite difference method (FDM) can be a better tool to solve these type of problems than the characteristics method. Shabaik et al. [4] further compared several numerical methods in both accuracy and response time for real-time monitoring scheme. Nam et al. [5] extended the method to multi-node setting. It turns out that one of the main limitation of FDM is that it is mesh dependent. To obtain high resolution mesh on the spatial domain, the time mesh needs to be exponentially more dense, making the overall computation slow and often infeasible. As a result, FDM method can only afford a sparse mesh over the spatial domain for practical purpose.

This work was supported by Electronics and Telecommunications Research Institute (ETRI) grant funded by the Korean government [23ZK1140, Honam region regional industry-based ICT convergence technology advancement support project].

Recently the development of machine learning and neural networks allowed various numerical solution methods for PDEs [6]. In particular, the introduction of physics-informed neural network (PINN) [7] has allowed various physics problems with only small amount of data to be approached like a PDE with initial and boundary conditions, numerically solved under machine learning scheme. Since then, PINN has been applied to wide range of physical applications, such as hyperelasticity problems [8], inverse problems [9] [10], thermal modeling [11] [12] and power systems [13]. Fluid dynamics, which is mostly based on the Navier-Stokes equation, has also seen PINN implementation for many particular problems [14] [15] [16] [17] [18] [19] [20] [21] [22] [23] [24] [25] [26] [27]. For our case of practice, the main advantage of PINN is that unlike FDM, PINN is virtually mesh free, since a successful training would imply that the collocation loss converges on an arbitrary set of interior points. Therefore PINN can provide predictions with much higher resolution over the spatial domain. In our paper, we will implement PINN to simulate transient pipeline flow under various scenarios, and compare it with the previous FDM method.

II. The PDE Model

We consider the flow of a compressible fluid in a straight segment of a pipeline with circular cross-section. For our analysis, two different quantities may be considered; the pressure p of the fluid and the mass flow q at a given point. If we know those two, the fluid density ρ and the volume flow ν can also be deduced.

For a given segment of a pipe, the mass and momentum balance can be described by the Darcy–Weisbach equation as follows.

$$\begin{gathered}\frac{\partial \rho}{\partial t}+\frac{\partial(\rho \nu)}{\partial x}+\frac{q_{\text{Leak}}}{A \Delta x}=0 \\ \frac{\partial(\rho \nu)}{\partial t}-\frac{\partial\left(\rho \nu^2\right)}{\partial x}+\frac{\partial p}{\partial x}-\frac{q_{\text{Leak}} \nu}{A \Delta x}+\frac{f \rho \nu^2}{2 D}+\rho g \sin \psi=0\end{gathered} \tag{1}$$

f is the friction factor which we assume to be constant throughout the pipe, a is the wave speed constant, A and D

are area and diameter of the pipe at position x respectively. q_{Leak} is the leakage mass of the pipe, if applicable. ψ is the pipeline inclination angle.

In this paper, we consider both q_{Leak} and ψ to be 0 so the related terms vanish. Moreover, the flow velocity ν is assumed to be much less than the acoustic velocity a, which allows the term $\partial(\rho\nu^2)/\partial x$ to be ignored.

We also use the facts $a^2 = p/\rho$ and $q = \rho A\nu$, and assume that A is not dependent on x (straight pipe with constant cross-sectional area throughout) to rewrite (1) as follows.

$$\begin{aligned} p_t + \frac{a^2}{A} q_x &= 0 \\ q_t + A p_x + \frac{f a^2}{2AD} \frac{q|q|}{p} &= 0 \end{aligned} \tag{2}$$

III. PINN-FEASIBLE PDE

A. Steady-state analysis

The PDE model discussed above does not provide the initial/boundary condition, and even in practice they are often not fully provided. In order to specify them from the limited control/observed data, for each of our experiment, we will assume that the system is initially in a steady state. If the system is in a steady state, both p and q (as well as ρ and ν) are time-invariant, that is, $p_t = q_t = 0$. Plugging in these to the first equation in (2), we get $q_x = 0$, which implies that $q(t, x) = q_{\text{ss}}$ is a constant function.

Because q_{ss} is a constant, the second equation of (2) becomes

$$p_x + \frac{\left(f a^2 q_{\text{ss}} |q_{\text{ss}}|\right)/(2A^2 D)}{p} = 0$$

Solving this equation subject to the condition $p_{\text{ss}}(0) = p_0$ and $p_{\text{ss}}(L) = p_L$ yields the closed form solution of $p_{\text{ss}}(x)$, which can then be used to solve for the closed form solution of q_{ss}, as described as follows.

$$\begin{aligned} p_{ss}(x) &= \sqrt{p_0^2 - \frac{p_0^2 - p_L^2}{L} x} \\ q_{ss}(x) &= \sqrt{\frac{DA}{fa^2} \cdot \frac{p_0^2 - p_L^2}{L}} \end{aligned} \tag{3}$$

where p_0 and p_L are inlet/outlet pressure (we assume that $p_0 \geq p_L$) and L is the length of the pipe. This suggests that the initial inlet/outlet pressure information is sufficient for us to establish the initial condition as a whole.

Figure 1 shows how the initial and boundary conditions are set up, using an example case which we will further discuss in section IV.

B. PINN-feasible normalization

It is important to note that each of the position domain x, time domain t, pressure output p (which must be computed in pascal unit), and flow output q require data normalization, as their values are generally large in practical application. Moreover, unlike many machine learning models, the PDE

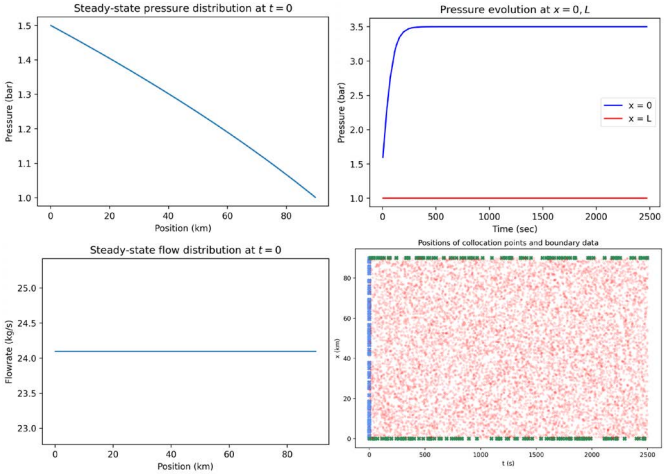

Fig. 1. Left: Pressure and flow map at $t = 0$, computed from steady-state analysis. Top Right: Observed inlet/outlet pressure over time. Bottom right: Sample points over time-spatial domain to compute PINN loss. The interior collocation points does not require actual data sampling.

(2) also needs to be modified accordingly. This is because the structure of PINN relies on the physical-mathematical relation of the data as well as the data itself. Without these adjustments PINN will not train properly.

Suppose $p(t, x)$ and $q(t, x)$ have domain $t \in [0, T]$ and $x \in [0, L]$. We substitute $p(t, x)$ and $q(t, x)$ with new functions $\bar{p}(u, v)$ and $\bar{q}(u, v)$, such that the domain is within $[0, 1] \times [0, L/X]$ and the range is also within similar interval.

$$\begin{aligned} \bar{p}(u, v) &= \frac{1}{P}\left(p(Tu, Xv) - P\right) = \frac{p(t, x)}{P} - 1 \\ \bar{q}(u, v) &= \frac{1}{Q}\left(q(Tu, Xv) - Q\right) = \frac{q(t, x)}{Q} - 1 \end{aligned} \tag{4}$$

In other words, we use T, X as input scale factor and P, Q as output scale/normalization factor.

Formula (4), combined with the input scaling $t = Tu$ and $x = Xv$ suggest the following.

$$\begin{aligned} \bar{p}_u = \frac{\partial}{\partial u}\left(\frac{p}{P} - 1\right) &= \frac{1}{P}\frac{\partial p}{\partial t}\frac{du}{dt} = \frac{T}{P} p_t \\ \bar{q}_u &= \frac{T}{Q} q_t \\ \bar{p}_v = \frac{\partial}{\partial v}\left(\frac{p}{P} - 1\right) &= \frac{1}{P}\frac{\partial p}{\partial x}\frac{dv}{dx} = \frac{X}{P} p_x \\ \bar{q}_v &= \frac{X}{Q} q_x \\ p &= (\bar{p} + 1)P \\ q &= (\bar{q} + 1)Q \end{aligned} \tag{5}$$

Substituting p_t, p_x, q_t, q_x, p and q in (2) with relations in (5) yield the following PINN-feasible PDE.

$$\begin{aligned} \frac{P}{T}\bar{p}_u + \frac{Q}{X}\frac{a^2}{A}\bar{q}_v &= 0 \\ \frac{Q}{T}\bar{q}_u + \frac{AP}{X}\bar{p}_v + \frac{f a^2 Q^2 (\bar{q} + 1)|\bar{q} + 1|}{2DAP(\bar{p} + 1)} &= 0 \end{aligned} \tag{6}$$

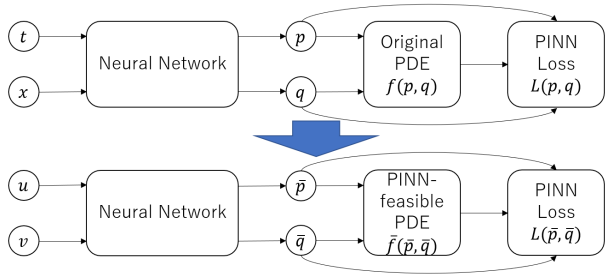

Fig. 2. Top: Overall structure of raw-data PINN. Due to the high discrepancy between input and output variables, this model does not learn very well. Bottom: Overall structure of normalized PINN. As the input and output variables normalize to each other, the loss-computing PDE must be modified accordingly to preserve the physics information.

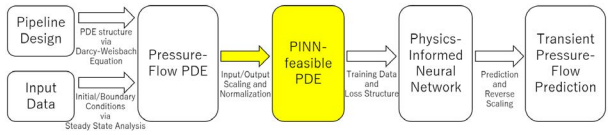

Fig. 3. Overall system flowchart. The pipeline design is embedded into the Darcy-Weisbach equation to establish the PDE structure. The input data collected from the inlet/outlet position of the pipe, along with steady state analysis, provide the initial and boundary conditions. This PDE is then scaled into PINN-feasible modified PDE, which is trained by PINN. Once the training is complete, it can be used to predict the overall time-spatial pressure and flow map.

We comment that while the choice of input-scaling factor T, X are fairly straightforward (one would usually use median or mean of the input data), the choice of output-scaling factor P, Q are more heuristic, as they would depend on the unknown output quantities. The steady-state analysis gives us a good idea of what P, Q should be, but depending on the type of the problem, if the model output changes greatly over time, the choice of P, Q based on the initial observation may not be adequate.

We will use (6) for our PINN to train $\bar{p}_\theta$ and $\bar{q}_\theta$. For the rest of the paper, we will use $L = 90km$ and $f = 0.003$. Figure 2 compare the overall structure of raw-data PINN and normalized PINN, and figure 3 shows the schematic architecture of our approach.

IV. Case: Supply Pressure Spike

Here we assume that the pipe has uniform cross-section of $1m^2$, and is initially in steady state with inlet pressure 1.5bar and outlet pressure 1.0bar. The inlet pressure will then rapidly increase up to 3.5bar, under the following formula.

$$p(t,0) = 2 \cdot \left(\frac{2}{1+e^{-50t}} - 1 \right) + 1.5$$
$$p(t,L) = 1 \tag{7}$$

We will predict the system over 2500 seconds time window. The initial pressure distribution and mass flow can be recovered from (3).

$$p(0,x) = \sqrt{1.5^2 - \frac{(1.5^2 - 1.0^2)}{90000} x}$$
$$q(0,x) \approx 24.10 \tag{8}$$

The inputs t and x will be scaled by $T = 2500$ and $X = 10^5$. The outputs p and q will be scaled and normalized by $P = 1.25 \cdot 10^5$ and $Q = q(0,x)$. Using these factors for (6), the modified $\bar{p}, \bar{q}$ are now a trainable function.

A. PINN Training

We build a standard neural network that will take two (u, v) inputs and two $(\bar{p}^\theta, \bar{q}^\theta)$ outputs, with 8 hidden layers of 30 nodes each. Each hidden layers include $\tanh$ as activation function.

The resulting $\bar{p}^\theta$ and $\bar{q}^\theta$ are then compared with the true $\bar{p}$ and $\bar{q}$ on $N_0 = N_b = 100$ sampled points, each along the initial and boundary conditions $\bar{p}(0,v), \bar{q}(0,v), \bar{p}(u,0)$ and $\bar{p}(u, L/X)$. Moreover, the PDE criterion (6) will be checked over $N_f = 10,000$ sampled collocation points within the time-spatial domain. The bottom right figure in Figure 1 illustrates the point sampling scheme. The overall mean square error of the PINN is defined as follows.

$$\begin{aligned}
MSE_0 &= \frac{1}{N_0}\sum_{i=1}^{N_0}\left(\bar{p}^\theta(0,v_i) - \bar{p}(0,v_i)\right)^2 \\
&\quad + \frac{1}{N_0}\sum_{i=1}^{N_0}\left(\bar{q}^\theta(0,v_i) - \bar{q}(0,v_i)\right)^2 \\
MSE_b &= \frac{1}{N_b}\sum_{i=1}^{N_b}\left(\bar{p}^\theta(u_i,0) - \bar{p}(u_i,0)\right)^2 \\
&\quad + \frac{1}{N_b}\sum_{i=1}^{N_b}\left(\bar{p}^\theta(u_i,L/X) - \bar{p}(u_i,L/X)\right)^2 \\
MSE_f &= \frac{1}{N_f}\sum_{i=1}^{N_f}\left[\left(\frac{P}{T}\bar{p}_u^\theta + \frac{Q}{X}\frac{a^2}{A}\bar{q}_u^\theta\right)^2\right. \\
&\quad \left. + \left(\frac{Q}{T}\bar{q}_u^\theta + \frac{AP}{X}\bar{p}_v^\theta + \frac{fa^2Q^2(\bar{q}^\theta+1)|\bar{q}^\theta+1|}{2DAP(\bar{p}^\theta+1)}\right)^2\right] \\
MSE &= MSE_0 + MSE_b + MSE_f
\end{aligned} \tag{9}$$

We train this PINN for 400,000 epochs under different learning rates, including adaptive scheme. For adaptive scheme, the learning rate step down every $100,000$ epochs, from 10^{-3} to 10^{-6}. It turns out that the adaptive learning rate method yields the best result, which we will use for the model generation in this paper. Figure 4 illustrates the comparison between these schemes as well as the loss evolution under the adaptive scheme.

B. Comparison with FDM result

To verify our result, we solve the same system with a simple explicit forward-time centred-space (FTCS) scheme. The time grid was divided into $N_t = 5,000$ sub-interval and the spatial grid was divided into $N_x = 16$ sub-interval. It is necessary that $N_t \gg N_x$; otherwise the solution starts to oscillate as the iteration proceeds and might even completely diverge.

Figure 5 compare the time-spatial map of p and q for both methods. Figure 6 compares the time evolution of p and q for

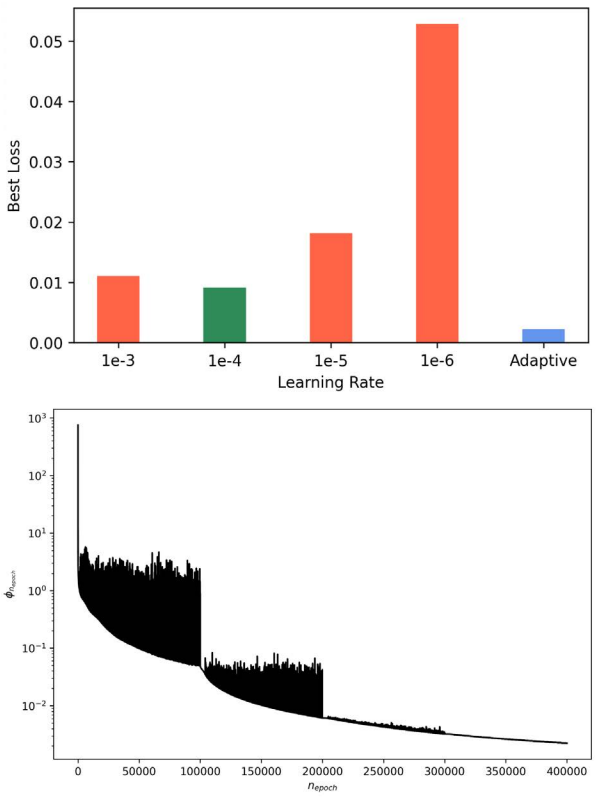

Fig. 4. Top: Best loss value after $400,000$ epochs of training under given learning rate. Bottom: Evolution of loss value MSE under the adaptive shceme.

specific (quarter) points along the pipe, as well as the pipeline snapshot at $t = 2500$. We can observe that both PINN and FTCS shows similar convergence to the long-time steady-state solution on the pressure side, but PINN shows a smoother convergence on the flow side.

To measure the similarity between the PINN prediction p^θ, q^θ and the FTCS estimation p^F, q^F, we will compare the norm between the difference of the two and p^θ, q^θ. That is, we take the sum of absolute difference between the two over the mesh of FTCS method, and divide it by the sum of PINN prediction result.

$$\begin{aligned}\text{sim}(p^\theta, p^F) &= \frac{\|p^\theta - p^F\|_1}{\|p^\theta\|_1} \approx \frac{\sum |p^\theta(t_i, x_i) - p^F(t_i, x_i)|}{\sum |p^\theta(t_i, x_i)|} \\ \text{sim}(q^\theta, q^F) &= \frac{\|q^\theta - q^F\|_1}{\|q^\theta\|_1} \approx \frac{\sum |q^\theta(t_i, x_i) - q^F(t_i, x_i)|}{\sum |q^\theta(t_i, x_i)|}\end{aligned} \tag{10}$$

In our experiment, $\text{sim}(p^\theta, p^F) \approx 0.00560$ and $\text{sim}(q^\theta, q^F) \approx 0.02225$. Note that neither p^θ, q^θ nor p^F, q^F are exact solution. In fact, we could argue that p^F, q^F are more prone to error which has been suggested by previous studies on FDM approach. This also explains why q^θ, q^F are less similar to each other than p^θ, p^F, as the second equation in (2) inherently amplifies more error in case of FTCS.

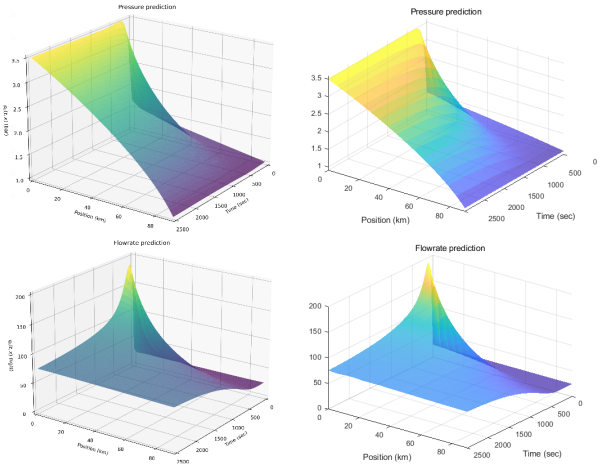

Fig. 5. Prediction map of pressure and mass flow through the pipe. Left images are prediction from PINN and right images are estimation from classic FTCS method.

V. Conclusion and Discussion

Our result show that PINN can be a useful tool to simulate and estimate transient flow subject to relevant situations, but each problem must be carefully normalized for proper training. Once PINN is properly applied, compared to FDM methods, the mesh-free nature of PINN allows higher resolution predictions and ability to deal with drastic change in the system. Also, PINN method shows better adaptability to different initial and boundary conditions.

PINN based analysis still has a lot of room to improve. To train a much more complex system, the loss components in (9) should be prioritized such that the initial and boundary conditions are imposed first. This issue has been recognized from the earliest development of PINN [28] [29], and many improvements have been proposed since [30] [31]. We look forward to blend these methods into our model the future.

Also, our ultimate goal is to implement PINN based model for larger system of pipeline network. Current stage of model only applies to a single segment of pipeline, and while it can be directly expanded to multi-nodal pipeline network theoretically, it will take extreme amount of time to train such model. To work around this, we can utilize hydraulic modelling solutions that can compute steady state equilibrium of universal network [32], and then use it to setup boundary conditions for the local models as described in this paper.

One of the limitation of PINN, although the same could be said for FDM methods as well, is that if the initial and boundary conditions change then the model should be re-trained accordingly. Because of this, there have been studies on different type of machine learning structure that could learn the entire family of PDEs and extrapolate beyond the domain of interest [33] [34] [35]. While these physics-encoded neural network (PENN) structures are more difficult to implement and handle, we believe that PENN could potentially solve the initial and boundary conditions dependence issue.

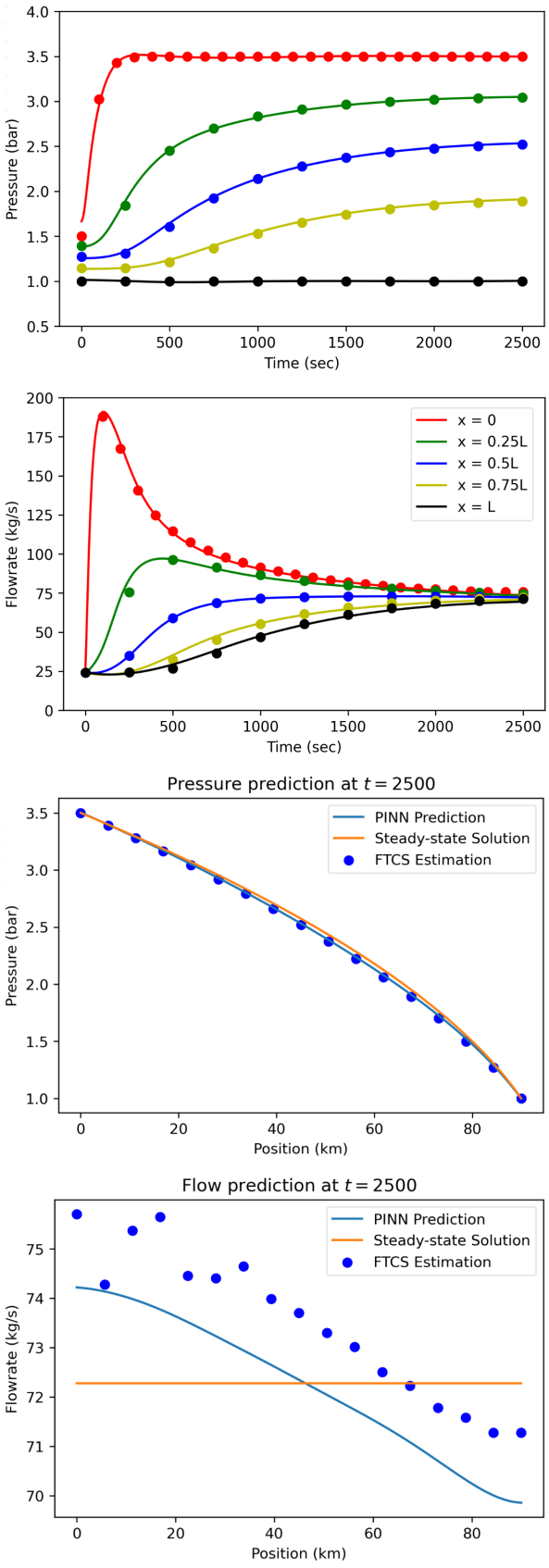

Fig. 6. Comparison of pressure and flow predictions by PINN (solid line) and FTCS (dotted points). First two images are time evolution for quater points of the pipe. Last two images are snapshots at $t = 2500$ with steady-state solution.

Appendix A
Nomenclature

Symbol	Description
a	wave speed (m/s)
A	cross-sectional area of the pipe (m^2)
D	pipe diameter (m)
L	length of pipe (m)
f	friction coefficient
g	acceleration of gravity (m/s^2)
p	fluid pressure (bar)
q	fluid mass flow rate (kg/s)
t	time (s)
x	position along the pipe (m)
ρ	fluid density (kg/m^3)
ν	fluid velocity (m/s)
$p_{\text{ss}}, q_{\text{ss}}$	steady-state pressure and flow rate
T, X, P, Q	input/output scaling factor
N_0, N_b	number of sample points along boundary conditions
N_f	number of collocation sample points
ψ	pipeline inclination angle (deg)
q_{Leak}	mass flow rate of pipe leakage (kg/s)

Acknowledgement

This work was supported by Electronics and Telecommunications Research Institute (ETRI) grant funded by the Korean government [23ZK1140, Honam region regional industry-based ICT convergence technology advancement support project]. The authors would like to express their deep gratitude to the research group members for constructive discussion and support.

References

[1] M. A. Stoner, "Analysis and Control of Unsteady Flows in Natural Gas Piping Systems," *Journal of Basic Engineering*, vol. 91, no. 3, pp. 331–338, 09 1969. [Online]. Available: https://doi.org/10.1115/1.3571107

[2] W. Yow, "Numerical Error on Natural Gas Transient Calculations," *Journal of Basic Engineering*, vol. 94, no. 2, pp. 422–428, 06 1972. [Online]. Available: https://doi.org/10.1115/1.3425438

[3] F. K. Choy, M. J. Braun, and H. S. Wang, "Transient Pressure Analysis in Piping Networks due to Valve Closing and Outlet Pressure Pulsation," *Journal of Pressure Vessel Technology*, vol. 118, no. 3, pp. 315–325, 08 1996. [Online]. Available: https://doi.org/10.1115/1.2842194

[4] H. E. Emara-Shabaik, Y. A. Khulief, and I. Hussaini, "Simulation of transient flow in pipelines for computer-based operations monitoring," *International Journal for Numerical Methods in Fluids*, vol. 44, no. 3, pp. 257–275, 2004. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/fld.636

[5] J.-H. Nam, C.-Y. Cho, S.-P. Jang, S.-H. Lim, D.-H. Shin, and T.-Y. Chung, "A study on pipeline network analysis for predicting pressure and flow rate transients in city-gas supply lines," *Journal of the Korean Institute of Gas*, vol. 12, no. 2, pp. 85–91, 2008.

[6] J. Blechschmidt and O. G. Ernst, "Three ways to solve partial differential equations with neural networks — a review," *GAMM-Mitteilungen*, vol. 44, no. 2, p. e202100006, 2021. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/gamm.202100006

[7] 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, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0021999118307125

[8] V. M. Nguyen-Thanh, X. Zhuang, and T. Rabczuk, "A deep energy method for finite deformation hyperelasticity," *European Journal of Mechanics - A/Solids*, vol. 80, p. 103874, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0997753819305352

[9] A. D. Jagtap, E. Kharazmi, and G. E. Karniadakis, "Conservative physics-informed neural networks on discrete domains for conservation laws: Applications to forward and inverse problems," *Computer Methods in Applied Mechanics and Engineering*, vol. 365, p. 113028, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0045782520302127

[10] X. Meng and G. E. Karniadakis, "A composite neural network that learns from multi-fidelity data: Application to function approximation and inverse pde problems," *Journal of Computational Physics*, vol. 401, p. 109020, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0021999119307260

[11] F. Bragone, K. Morozovska, P. Hilber, T. Laneryd, and M. Luvisotto, "Physics-informed neural networks for modelling power transformer's dynamic thermal behaviour," *Electric Power Systems Research*, vol. 211, p. 108447, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0378779622005855

[12] S. Cai, Z. Wang, S. Wang, P. Perdikaris, and G. E. Karniadakis, "Physics-Informed Neural Networks for Heat Transfer Problems," *Journal of Heat Transfer*, vol. 143, no. 6, 04 2021, 060801. [Online]. Available: https://doi.org/10.1115/1.4050542

[13] G. S. Misyris, A. Venzke, and S. Chatzivasileiadis, "Physics-informed neural networks for power systems," in *2020 IEEE Power & Energy Society General Meeting (PESGM)*, 2020, pp. 1–5.

[14] L. Sun, H. Gao, S. Pan, and J.-X. Wang, "Surrogate modeling for fluid flows based on physics-constrained deep learning without simulation data," *Computer Methods in Applied Mechanics and Engineering*, vol. 361, p. 112732, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S004578251930622X

[15] K. O. Lye, S. Mishra, and D. Ray, "Deep learning observables in computational fluid dynamics," *Journal of Computational Physics*, vol. 410, p. 109339, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0021999120301133

[16] J. Magiera, D. Ray, J. S. Hesthaven, and C. Rohde, "Constraint-aware neural networks for riemann problems," *Journal of Computational Physics*, vol. 409, p. 109345, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0021999120301194

[17] Z. Mao, A. D. Jagtap, and G. E. Karniadakis, "Physics-informed neural networks for high-speed flows," *Computer Methods in Applied Mechanics and Engineering*, vol. 360, p. 112789, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0045782519306814

[18] H. Wessels, C. Weißenfels, and P. Wriggers, "The neural particle method – an updated lagrangian physics informed neural network for computational fluid dynamics," *Computer Methods in Applied Mechanics and Engineering*, vol. 368, p. 113127, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0045782520303121

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

[20] X. Jin, S. Cai, H. Li, and G. E. Karniadakis, "NSFnets (Navier-Stokes flow nets): Physics-informed neural networks for the incompressible Navier-Stokes equations," *Journal of Computational Physics*, vol. 426, p. 109951, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0021999120307257

[21] M. Raissi, A. Yazdani, and G. E. Karniadakis, "Hidden fluid mechanics: Learning velocity and pressure fields from flow visualizations," *Science*, vol. 367, no. 6481, pp. 1026–1030, 2020. [Online]. Available: https://www.science.org/doi/abs/10.1126/science.aaw4741

[22] M. Yin, X. Zheng, J. D. Humphrey, and G. E. Karniadakis, "Non-invasive inference of thrombus material properties with physics-informed neural networks," *Computer Methods in Applied Mechanics and Engineering*, vol. 375, p. 113603, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S004578252030788X

[23] G. Kissas, Y. Yang, E. Hwuang, W. R. Witschey, J. A. Detre, and P. Perdikaris, "Machine learning in cardiovascular flows modeling: Predicting arterial blood pressure from non-invasive 4d flow mri data using physics-informed neural networks," *Computer Methods in Applied Mechanics and Engineering*, vol. 358, p. 112623, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0045782519305055

[24] X. I. A. Yang, S. Zafar, J.-X. Wang, and H. Xiao, "Predictive large-eddy-simulation wall modeling via physics-informed neural networks," *Phys. Rev. Fluids*, vol. 4, p. 034602, Mar 2019. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevFluids.4.034602

[25] Q. Lou, X. Meng, and G. E. Karniadakis, "Physics-informed neural networks for solving forward and inverse flow problems via the boltzmann-bgk formulation," *Journal of Computational Physics*, vol. 447, p. 110676, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0021999121005714

[26] S. Cai, Z. Wang, F. Fuest, Y. J. Jeon, C. Gray, and G. E. Karniadakis, "Flow over an espresso cup: inferring 3-d velocity and pressure fields from tomographic background oriented schlieren via physics-informed neural networks," *Journal of Fluid Mechanics*, vol. 915, p. A102, 2021.

[27] D. Lucor, A. Agrawal, and A. Sergent, "Physics-aware deep neural networks for surrogate modeling of turbulent natural convection," *ArXiv*, vol. abs/2103.03565, 2021.

[28] S. Wang, Y. Teng, and P. Perdikaris, "Understanding and mitigating gradient flow pathologies in physics-informed neural networks," *SIAM Journal on Scientific Computing*, vol. 43, no. 5, pp. A3055–A3081, 2021. [Online]. Available: https://doi.org/10.1137/20M1318043

[29] C. Rao, H. Sun, and Y. Liu, "Physics-informed deep learning for computational elastodynamics without labeled data," *Journal of Engineering Mechanics*, vol. 147, no. 8, p. 04021043, 2021.

[30] N. Sukumar and A. Srivastava, "Exact imposition of boundary conditions with distance functions in physics-informed deep neural networks," *Computer Methods in Applied Mechanics and Engineering*, vol. 389, p. 114333, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0045782521006186

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

[32] O. Awe, S. Okolie, and O. Fayomi, "Review of water distribution systems modelling and performance analysis softwares," *Journal of Physics: Conference Series*, vol. 1378, no. 2, p. 022067, dec 2019. [Online]. Available: https://dx.doi.org/10.1088/1742-6596/1378/2/022067

[33] C. Rao, H. Sun, and Y. Liu, "Hard encoding of physics for learning spatiotemporal dynamics," *arXiv preprint arXiv:2105.00557*, 2021.

[34] Z. Li, N. B. Kovachki, K. Azizzadenesheli, B. liu, K. Bhattacharya, A. Stuart, and A. Anandkumar, "Fourier neural operator for parametric partial differential equations," in *International Conference on Learning Representations*, 2021. [Online]. Available: https://openreview.net/forum?id=c8P9NQVtmnO

[35] R. T. Q. Chen, Y. Rubanova, J. Bettencourt, and D. K. Duvenaud, "Neural ordinary differential equations," in *Advances in Neural Information Processing Systems*, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, Eds., vol. 31. Curran Associates, Inc., 2018. [Online]. Available: https://proceedings.neurips.cc/paper/2018/file/69386f6bb1dfed68692a24c8686939b9-Paper.pdf