DOI: 10.61782/fa.2025.0602

FORUM ACUSTICUM
EURONOISE 2025

NEURAL NETWORK-BASED SOLUTIONS FOR THE LINEARIZED EULER EQUATIONS IN OUTDOOR SOUND PROPAGATION

Hessel Juliust[1,2] **Katharina Baumann**[1] **Arthur Schady**[1]
Sirine Gharbi[1] **Felix Dietrich**[2]
[1] Institute of Atmospheric Physics, German Aerospace Center
[2] School of Computation, Information and Technology, Technische Universität München

ABSTRACT

The Linearized Euler Equations (LEE) are a system of partial differential equations that provide a framework for modeling outdoor acoustic wave propagation, capturing atmospheric and topographic effects. Solving LEE using traditional numerical methods demands fine spatial and temporal resolutions, leading to high computational costs over large domains. This study uses established neural network approaches to solve LEE and evaluates their performance and scalability for outdoor acoustic wave propagation. Included approaches are Physics-Informed Neural Networks (PINNs) and the sampled network-based Extreme Learning Machine Ordinary Differential Equations (ELM-ODE). We use a SIREN-based architecture in our PINNs for wave-like solutions. In the 1D case, PINNs achieved higher accuracy but required significantly more training time due to the complexity of training over a spatiotemporal domain. Meanwhile, ELM-ODE provided competitive accuracy with much lower computational cost. In the 2D case, ELM-ODE again showed computational advantages over SIREN-PINNs while delivering great accuracy. However, its scalability is constrained by the coupled velocity–pressure matrix, which increases costs in cases like outdoor noise mapping, where velocity data eventually are unused. Addressing boundary conditions, such as frequency-dependent impedance, remains a challenge.

*__Corresponding author__: *hessel.juliust@dlr.de.*

Keywords: *linearized euler equations, outdoor acoustics, PINNs, ELM-ODE, sampled networks*

1. INTRODUCTION

Modeling outdoor acoustic wave propagation is important with a lot of applications, including noise mapping for urban planning [1], environmental monitoring [2], and aviation acoustics [3]. The propagation of sound in the atmosphere is affected by atmospheric factors such as wind, temperature gradients, and topographic features [2]. These factors introduce complexities into predicting acoustic wave behavior accurately, especially when considering high-frequency waves or large spatial domains.

The Linearized Euler Equations (LEE) provide a mathematical framework for modeling outdoor acoustics by capturing the interactions among acoustic velocity, pressure fluctuations, and the background atmospheric flow [4, 5]. Unlike simpler acoustic models, the LEE take account for the convective effects and the compressible nature of sound waves, making them particularly well-suited for simulating realistic outdoor conditions. However, traditional numerical methods for solving the LEE, such as finite-difference time-domain (FDTD) schemes [6, 7], often require substantial computational resources. This is especially true for scenarios involving high frequencies or extended spatial domains [8].

Recent advances in machine learning offer a new way to solve LEE. In particular, *physics-informed neural networks* (PINNs) have been proposed as an effective means to solve partial differential equations (PDEs) by embedding the governing equations directly into the loss function during training [9]. This approach ensures that the neural network follows the underlying physics. How-

ever, PINNs can be computationally demanding and may encounter convergence issues when dealing with high-frequency oscillations and multi-scale features [10], especially for solving time dependent PDEs like LEE which need to be trained for the whole time domain.

An alternative strategy, the *sampled neural network method* to solve PDE has been introduced to overcome some of these challenges. In the work of Datar *et al.* [11], the authors propose the *Extreme Learning Machine Ordinary Differential Equations* (ELM-ODE) framework, which applies a shallow neural network with fixed, randomly sampled weights to represent spatial basis functions. The time evolution of the solution is then captured by solving a system of ordinary differential equations for the time-dependent coefficients. This decoupling of the spatial and temporal components leads to a dramatic reduction in training time while still achieving high accuracy in modeling the wave dynamics inherent in the LEE.

In this paper, we present a comparative study of the PINNs and ELM-ODE approaches for solving the Linearized Euler Equations in the context of outdoor acoustic propagation. We present how to implement each framework to solve LEE and assess the methods' performance in terms of accuracy and computational efficiency. Our goal is to explore the strengths and limitations of these techniques for simulation of outdoor acoustics.

2. METHODS

2.1 Linearized Euler Equations

The linearized Euler equations have become a widely accepted model for simulating outdoor sound propagation. Their time-domain formulation offers several key advantages. First, a single simulation can generate comprehensive frequency responses. Second, they accommodate nonlinear effects while accounting for moving, realistic sources. When dealing with a moving medium, it is essential to use numerical methods that accurately capture both the medium's velocity and the acoustic pressure [12]. For instance, models described in [7] employ finite-difference time-domain (FDTD) methods to simulate sound wave motion in a moving medium. Atmospheric conditions significantly influence outdoor sound propagation, further highlighting the need to consider the medium's dynamics. The LEE consists of coupled equations for the velocity and pressure components, defined as follows:

The velocity component of LEE is given by:

$$\begin{aligned}\frac{\partial \mathbf{v}_{\text{acc}}}{\partial t} + (\mathbf{v}_{\text{met}} \cdot \nabla)\mathbf{v}_{\text{acc}} + (\mathbf{v}_{\text{acc}} \cdot \nabla)\mathbf{v}_{\text{met}} \\ = -\alpha_{\text{met}} \nabla p_{\text{acc}} - \alpha_{\text{acc}} \nabla p_{\text{met}} + \nu \nabla^2 \mathbf{v}_{\text{acc}}.\end{aligned} \tag{1}$$

Similarly, the pressure component is given by:

$$\begin{aligned}\frac{\partial p_{\text{acc}}}{\partial t} + \mathbf{v}_{\text{met}} \cdot \nabla p_{\text{acc}} + \mathbf{v}_{\text{acc}} \cdot \nabla p_{\text{met}} \\ = -\kappa p_{\text{met}} \nabla \cdot \mathbf{v}_{\text{acc}} - \kappa p_{\text{acc}} \nabla \cdot \mathbf{v}_{\text{met}}.\end{aligned} \tag{2}$$

The coefficients are defined as:

$$\alpha_{\text{met}} = \frac{1}{\rho_{\text{met}}}, \alpha_{\text{acc}} \quad = -\frac{1}{\kappa} \frac{p_{\text{acc}}}{p_{\text{met}}} \frac{1}{\rho_{\text{met}}}, \kappa = \frac{c_p}{c_v}.$$

The numerical solver will then solve for $\mathbf{v}_{\text{acc}}$ (the velocity of the particle) and p_{acc} (the pressure of the acoustic wave), with other variables such as p_{met} (the pressure of the flow field), $\mathbf{v}_{\text{met}}$ (the velocity of the flow field), ρ_{met} (the air density), and κ (the ratio of specific heats at constant pressure (c_p) and constant volume (c_v)) given. Note that κ can also be obtained from the speed of sound (c), p_{met}, and ρ_{met}.

2.2 Physics-Informed Neural Networks (PINNs)

Physics-Informed Neural Networks (PINNs) are neural networks trained to satisfy partial differential equations (PDEs), initial conditions (IC), and boundary conditions (BC) simultaneously [13]. The network receives spatiotemporal coordinates as input and outputs the physical fields to be learned. These outputs are differentiated with respect to the inputs using automatic differentiation [14], and then substituted into the PDE to obtain the residual. The residual quantifies how much the predicted fields violate the governing equations and is evaluated at a set of collocation points in the domain. The total loss is defined as:

$$\mathcal{L} = \mathcal{L}_{\text{phys}} + \mathcal{L}_{\text{IC}} + \mathcal{L}_{\text{BC}}, \tag{3}$$

where each term is a mean-squared error loss for the PDE residual $\mathcal{L}_{\text{phys}}$, the initial condition loss $\mathcal{L}_{\text{IC}}$, and the boundary condition loss $\mathcal{L}_{\text{BC}}$. Note that it can also include additional loss term if data are available such as from old simulation or measurements.

When applied to the Linearized Euler Equations (LEE), the neural network takes spatial-temporal input coordinates (x, t) or (x, y, t) and outputs pressure and velocity components as continuous functions. The network utilizes sinusoidal activation functions based on the SIREN

architecture [15], where each layer applies

$$\sigma(z) = \sin(\omega_0 z)$$

with a fixed frequency scaling constant ω_0 in order to approximate the wavy behaviour better.

2.3 Sampled Neural Network Approach

The Sampled Neural Network approach used in this study follows the Extreme Learning Machine-based ODE (ELM-ODE) method introduced in [11]. This method approximates the solution of a time-dependent PDE by separating spatial and temporal components. The spatial part is represented using a shallow neural network with fixed, randomly sampled weights, while the temporal part is evolved by solving an ordinary differential equation (ODE) system.

Let $\phi_k(x) = \sigma(w_k^\top x + b_k)$ denote the k-th basis function, where σ is a nonlinear activation function also w_k and b_k are weights and biases sampled from prescribed distributions (e.g., $w_k \sim \mathcal{N}(\eta_1, \eta_2)$ and $b_k \sim \mathcal{U}[-\eta_3, \eta_3]$), as originally proposed in the Extreme Learning Machine framework [16]. The solution is expressed as:

$$\hat{u}(x,t) = \sum_{k=1}^{K} c_k(t) \cdot \phi_k(x), \tag{4}$$

where $c_k(t)$ are time-dependent coefficients.

This ansatz is substituted into the governing PDE, and the residual is evaluated at a set of collocation points. The resulting system is projected onto the fixed spatial basis, resulting an ODE system for the temporal coefficients $c(t) = [c_1(t), \ldots, c_K(t)]^\top$, which is then integrated forward in time using standard solvers. For linear PDEs such as the simplified Linearized Euler Equations (LEE), the resulting ODE system remains linear.

Boundary conditions can be incorporated in two main ways, either by constructing basis functions that satisfy the boundary constraints, or by introducing additional terms to the ODE system that enforce the conditions weakly.

3. FORMULATING SOLUTIONS TO LEE WITH NEURAL NETWORKS

We started from the full Linearized Euler Equations given in Equations (1) and (2), however, in order to focus on the dominant convective and acoustic effects, we neglect the nonlinear advection due to acoustic perturbations, the weak coupling with background pressure gradients, viscous dissipation, and the divergence of the background flow.

Therefore, omitting these effects, the governing system simplifies to:

$$\frac{\partial \mathbf{v}_{\text{acc}}}{\partial t} = -\left(\mathbf{v}_{\text{met}} \cdot \nabla\right) \mathbf{v}_{\text{acc}} - \frac{1}{\rho_{\text{met}}} \nabla p_{\text{acc}}, \tag{5}$$

$$\frac{\partial p_{\text{acc}}}{\partial t} = -\left(\mathbf{v}_{\text{met}} \cdot \nabla\right) p_{\text{acc}} - \kappa\, P_{\text{met}}\, \nabla \cdot \mathbf{v}_{\text{acc}}. \tag{6}$$

3.1 Formulation of LEE in the PINNs Framework

We first nondimensionalize the governing equations. Let L_{ref} denote a characteristic length scale, and define the reference time scale as $T_{\text{ref}} = L_{\text{ref}}/c$, where c is the speed of sound. Pressure is scaled by a reference value p_c, and velocity by $v_c = p_c/(\rho_{\text{ref}} c)$, where ρ_{ref} is the reference density. This yields a nondimensional domain $(x, y, t) \in [-1, 1]^2 \times [0, 1]$, with all physical constants absorbed into the transformed variables. The nondimensional background wind velocities are defined as $M_x = v_{\text{met},x}/c$ and $M_y = v_{\text{met},y}/c$. Note that this can vary along the spaces or be constant values.

We apply this nondimensionalizing to Equations (5) and (6), resulting in the following system:

$$\frac{\partial v_{acc,x}}{\partial t} = -M_x \frac{\partial v_{acc,x}}{\partial x} - M_y \frac{\partial v_{acc,x}}{\partial y} - \frac{\partial p}{\partial x}, \tag{7}$$

$$\frac{\partial v_{acc,y}}{\partial t} = -M_x \frac{\partial v_{acc,y}}{\partial x} - M_y \frac{\partial v_{acc,y}}{\partial y} - \frac{\partial p}{\partial y}, \tag{8}$$

$$\frac{\partial p}{\partial t} = -M_x \frac{\partial p}{\partial x} - M_y \frac{\partial p}{\partial y} - \frac{\partial v_{acc,x}}{\partial x} - \frac{\partial v_{acc,y}}{\partial y}. \tag{9}$$

Then, we define a neural network that takes spatiotemporal coordinates (x, y, t) as input and outputs the predicted fields $\tilde{v}_{acc,x}$, $\tilde{v}_{acc,y}$, and $\tilde{p}$. The residuals are defined as:

$$\mathcal{R}_{v_{\text{acc},x}} = \frac{\partial \tilde{v}_{\text{acc},x}}{\partial t} + M_x \frac{\partial \tilde{v}_{\text{acc},x}}{\partial x} + M_y \frac{\partial \tilde{v}_{\text{acc},x}}{\partial y} + \frac{\partial \tilde{p}}{\partial x}, \tag{10}$$

$$\mathcal{R}_{v_{\text{acc},y}} = \frac{\partial \tilde{v}_{\text{acc},y}}{\partial t} + M_x \frac{\partial \tilde{v}_{\text{acc},y}}{\partial x} + M_y \frac{\partial \tilde{v}_{\text{acc},y}}{\partial y} + \frac{\partial \tilde{p}}{\partial y}, \tag{11}$$

$$\mathcal{R}_p = \frac{\partial \tilde{p}}{\partial t} + M_x \frac{\partial \tilde{p}}{\partial x} + M_y \frac{\partial \tilde{p}}{\partial y} + \frac{\partial \tilde{v}_{\text{acc},x}}{\partial x} + \frac{\partial \tilde{v}_{\text{acc},y}}{\partial y}. \tag{12}$$

These residuals are evaluated at collocation points and minimized during training. In the one-dimensional variant, all y-dependent terms are removed.

The neural network takes as input the spatiotemporal coordinates (x, y, t) and outputs the nondimensional fields p, v_x, and v_y. We employ a fully connected deep neural network using sinusoidal activation functions (SIREN) [15]. Derivatives of the output fields with respect to the inputs are computed via automatic differentiation [14].

The total loss minimized during training is composed of three terms such as in Equations (3), where the physics-based residual loss is defined as

$$\mathcal{L}_{\text{PDE}} = \text{MSE}(\mathcal{R}_{v_x}) + \text{MSE}(\mathcal{R}_{v_y}) + \text{MSE}(\mathcal{R}_p). \quad (13)$$

The initial condition loss $\mathcal{L}_{\text{IC}}$ is computed from a Gaussian pressure pulse centered in the domain with zero initial velocity. Boundary conditions are incorporated through $\mathcal{L}_{\text{BC}}$, enforced by included as supervised data points.

Training points are sampled uniformly across the nondimensional domain. The network is optimized using the AdamW optimizer, combined with a cosine annealing learning rate schedule [17]. Optionally, dynamic loss weighting based on Neural Tangent Kernel (NTK) sensitivity are applied in 2D case to balance contributions from different residuals during training [18, 19].

3.2 LEE Solution via ELM-ODE

Following the sampled neural network approach described above, we approximate the acoustic fields using a shallow NN with fixed, randomly sampled weights and biases. In this formulation, the only unknowns are the time-dependent coefficients. In particular, we write the Galerkin ansatz for the fields as

$$v_{acc,x}(t,x,y) = c_0(t) + \sum_{k=1}^{K} c_k(t)\,\phi(w_k^1\,x + w_k^2\,y + b_k), \quad (14)$$

$$v_{acc,y}(t,x,y) = d_0(t) + \sum_{k=1}^{K} d_k(t)\,\phi(w_k^1\,x + w_k^2\,y + b_k), \quad (15)$$

$$p_{acc}(t,x,y) = e_0(t) + \sum_{k=1}^{K} e_k(t)\,\phi(w_k^1\,x + w_k^2\,y + b_k), \quad (16)$$

where ϕ is the chosen activation function, and $\{w_k^1, w_k^2, b_k\}$ are fixed through random sampling. In this way, we seek only the time-dependent coefficients $c(t)$, $d(t)$, and $e(t)$ rather than training the full set of network parameters.

3.2.1 *Galerkin Projection of the Derivatives*

The time derivatives of the fields are obtained by differentiating the coefficients:

$$\frac{\partial v_{acc,x}}{\partial t} = c_{0,t}(t) + \sum_{k=1}^{K} c_{k,t}(t)\,\phi(w_k^1\,x + w_k^2\,y + b_k), \quad (17)$$

$$\frac{\partial v_{acc,y}}{\partial t} = d_{0,t}(t) + \sum_{k=1}^{K} d_{k,t}(t)\,\phi(w_k^1\,x + w_k^2\,y + b_k), \quad (18)$$

$$\frac{\partial p_{acc}}{\partial t} = e_{0,t}(t) + \sum_{k=1}^{K} e_{k,t}(t)\,\phi(w_k^1\,x + w_k^2\,y + b_k). \quad (19)$$

Spatial derivatives are computed by differentiating the NN ansatz. For instance, the x-derivative of p_{acc} is given by

$$\frac{\partial p_{acc}}{\partial x} = e_0(t) + \sum_{k=1}^{K} e_k(t)\,\phi'(w_k^1\,x + w_k^2\,y + b_k)\,|w_k^1|, \quad (20)$$

with analogous expressions for the spatial derivatives of $v_{acc,x}$ and $v_{acc,y}$.

These evaluations at the collocation points are collected into matrices: let $\mathbf{A}^{(i)}$ denote the matrix of basis function evaluations for the i-th field and $\mathbf{B}_x^{(i)}$ and $\mathbf{B}_y^{(i)}$ denote the corresponding spatial derivative matrices (with $i = 1$ for $v_{acc,x}$, $i = 2$ for $v_{acc,y}$, and $i = 3$ for p_{acc}). We then form the block matrices

$$\mathbf{A} = \begin{bmatrix} \mathbf{A}^{(1)} & 0 & 0 \\ 0 & \mathbf{A}^{(2)} & 0 \\ 0 & 0 & \mathbf{A}^{(3)} \end{bmatrix}, \quad (21)$$

$$\mathbf{B}^{(x)} = \begin{bmatrix} \mathbf{B}_x^{(1)} & 0 & 0 \\ 0 & \mathbf{B}_x^{(2)} & 0 \\ 0 & 0 & \mathbf{B}_x^{(3)} \end{bmatrix}, \quad (22)$$

$$\mathbf{B}^{(y)} = \begin{bmatrix} \mathbf{B}_y^{(1)} & 0 & 0 \\ 0 & \mathbf{B}_y^{(2)} & 0 \\ 0 & 0 & \mathbf{B}_y^{(3)} \end{bmatrix}. \quad (23)$$

3.2.2 Assembly of the ODE System

The coupling terms from the Linearized Euler Equations, such as the pressure gradient and the divergence of velocity, are incorporated into the coupling matrix

$$\mathbf{K} = \begin{bmatrix} 0 & 0 & \alpha_{met}\,\mathbf{B}_x^{(3)} \\ 0 & 0 & \alpha_{met}\,\mathbf{B}_y^{(3)} \\ \kappa p_{met}\,\mathbf{B}_x^{(1)} & \kappa p_{met}\,\mathbf{B}_y^{(2)} & 0 \end{bmatrix}. \quad (24)$$

Defining the combined coefficient vector

$$\mathbf{Y}(t) = \begin{bmatrix} c(t) \\ d(t) \\ e(t) \end{bmatrix}, \quad (25)$$

the Galerkin projection leads to:

$$\mathbf{A}\frac{\partial \mathbf{Y}}{\partial t} = -\Big[V_{met,x}\,\mathbf{B}^{(x)} + V_{met,y}\,\mathbf{B}^{(y)} + \mathbf{K}\Big]\mathbf{Y}.$$

Because the NN weights and biases are already obtained by sampling, we solve only for the coefficients. Inverting $\mathbf{A}$ (or solving the associated least-squares problem) yields

$$\begin{aligned} \frac{\partial \mathbf{Y}}{\partial t} &= -\mathbf{A}^{-1}\Big[V_{met,x}\,\mathbf{B}^{(x)} + V_{met,y}\,\mathbf{B}^{(y)} + \mathbf{K}\Big]\mathbf{Y} \\ &\equiv \mathbf{B}_{combined}\,\mathbf{Y}. \end{aligned} \quad (26)$$

This linear ODE system for $\mathbf{Y}(t)$ is then integrated in time using standard solvers, after which the full solution is reconstructed by evaluating the NN basis functions weighted by the computed coefficients.

4. PARAMETER DETAILS

4.1 One Dimensional Case

The one-dimensional case serves as a preliminary test to compare the accuracy, convergence, and computational performance of both PINNs and ELM-ODE methods under simplified conditions. The physical parameters used in this setup are listed in Tab. 1, while the configuration details for the PINNs and ELM-ODE implementations are given in Tab. 2 and 3, respectively. Both methods are tested on the same domain and initial conditions, allowing for a fair comparison. This setup provides insights into the strengths and limitations of each approach before scaling up to higher-dimensional problems.

Table 1. Physical parameters for 1D simulation

Parameter	Value
Space Domain	$x \in [-100, 100]$ m
Time Domain	$t \in [0, 1]$ s
Medium flow velocity	5 m/s
Speed of sound	340 m/s
Initial pressure	Gaussian at $x = 0$, $\sigma = 5$ m
Boundary condition	Total reflection

Table 2. PINNs configuration for 1D simulation

Component	Value
Architecture	3 hidden layers, 30 neurons each (SIREN)
Activation	Sinusoidal, $\omega_0 = 3.0$
Input/Output	(x, t) input; $(v_{acc},\ p_{acc})$ output
Collocation points	8,000 (PDE), 200 (BC), full grid (IC)
Optimizer	Adam (lr=10^{-3}, exponential decay)
Epochs	500,000
Computing Device	Nvidia P100

Table 3. ELM-ODE configuration for 1D simulation

Component	Value
Neuron Basis	600 per field (velocity and pressure)
Activation	$\tanh$
Sampling	Randomized with scaling
Boundary	Total Reflection
Basis Fitting	Least-squares projection with regularization (10^{-6})
Spatial Resolution	400 points over the domain
Time Integration	RK45
Computing Device	Intel Xeon Gold 5218R (1 core)

4.2 Two Dimensional Case

The two-dimensional case extends the previous 1D setup to a more realistic scenario of outdoor sound propagation

with a horizontal background wind. It evaluates the methods' performance in capturing wavefront changes, directional propagation, and boundary interactions.

Table 4. Physical parameters for 2D simulation

Parameter	Value
Space Domain	$x, y \in [-100, 100]$ m
Time Domain	$t \in [0, 1]$ s
Medium Flow velocity	$V_x = 40$ m/s, $V_y = 0$ m/s
Initial pressure	Gaussian, $\sigma = 20$ m
Boundary condition	Total reflection

Table 5. PINNs configuration for 2D simulation

Component	Value
Architecture	6 hidden layers, 80 neurons each with SIREN activation ($\omega_0 = 1$)
Input/Output	(x, y, t) input; (p, v_x, v_y) output
Collocation points	50,000 (PDE), 3,000 (BC), 5,000 (IC)
Optimizer	AdamW (lr=10^{-3}, cosine annealing)
Epochs	400,000
NTK	EMA with $\beta = 0.99$
Computing Device	Nvidia A100

Table 6. ELM-ODE configuration for 2D simulation

Component	Value
Neuron Basis	1500 per field (velocity v_x, velocity v_y, pressure)
Activation	$\tanh$
Sampling	Randomized with scaling
Boundary	Total Reflection
Basis Fitting	Least-squares projection with regularization (10^{-8})
Spatial Resolution	500 points per dimension
Time Integration	RK45
Computing Device	Intel Xeon Gold 5218R (1 core)

5. RESULTS AND DISCUSSION

5.1 1D Cases

Fig. 1 compares the solutions of the 1D Linearized Euler Equation obtained using finite-difference (FD), ELM-ODE, and PINNs approaches. The FD solution acts as the ground truth for evaluating the performance of neural network-based methods.

Both ELM-ODE and PINNs solutions accurately reproduce the overall wave dynamics, including reflections at domain boundaries and the propagation of the pressure pulse across the simulation domain. However, noticeable differences exist in their error distributions. The absolute error for the ELM-ODE solution (bottom-left panel of Fig. 1) primarily concentrates along the primary wavefront paths, indicating that ELM-ODE slightly underpredicts rapid changes. In contrast, the error distribution for the PINNs solution (bottom-right panel) shows more evenly dispersed, finer-scale errors across the domain. This behavior suggests that while PINNs captures the overall wave patterns very well, it exhibits smaller-scale oscillations or numerical artifacts.

Quantitatively, as shown in Tab. 7, PINNs achieve a lower mean absolute error (0.001851 Pa) than the ELM-ODE method (0.007404 Pa), confirming that PINNs offer better accuracy overall in the tested scenario. However, this comes at the cost of significantly higher computational demands, PINNs require approximately one hour of training, while ELM-ODE produces results within seconds.

Table 7. Accuracy and cost for 1D simulation

Metric	PINNs	ELM-ODE
MAE	1.851×10^{-3}	7.404×10^{-3}
Training Time	1 h 2 min	2.98 s

5.2 2D Cases

Extending the methods from Section 5.1 to two dimensions allows for more realistic outdoor acoustic propagation scenarios, such as a planar Gaussian pressure pulse advected by a horizontal background wind. Fig. 2 compares the solutions at two different time instants, $t = 0.2$ s and $t = 0.6$ s. Both PINNs and ELM-ODE solutions closely match the finite-difference (FD) reference, capturing wavefront propagation and boundary reflections effectively. At early times ($t = 0.2$ s) (as it is written in Tab. 8),

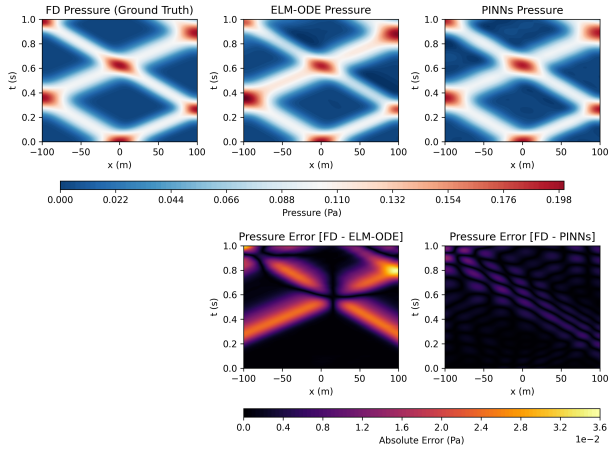

Figure 1. Comparison of pressure fields (top row) and absolute error (bottom row) for the 1D Linearized Euler Equation simulation. The FD solution serves as ground truth.

both methods yield mean absolute errors (MAE) in the order of 10^{-3}. At later times ($t = 0.6\,\mathrm{s}$), the error increases to the order of 10^{-2}, consistent with expected dispersion and diffusion. Spatially, PINNs errors are more uniformly distributed across the domain, while ELM-ODE shows errors along high-gradient regions.

Despite these similar accuracies, the two methods differ significantly in runtime. The PINNs approach required roughly 7.6 hours of training, whereas ELM-ODE finished in about 23 minutes. Important is, most of the ELM-ODE cost does not come from integrating the ordinary differential equation system, rather, it is spent assembling and factorizing the large block matrices (as discussed around Equations (21), (22), (23), (24), and (26)). Once those matrices are formed and factorized, the subsequent time-integration step involves repeated matrix-vector multiplications, which are relatively cheap.

An additional observation is that both methods must solve for all three fields (v_x, v_y, p). Even if we are mainly interested in the acoustic pressure p, the Linearized Euler Equations intrinsically couple pressure and velocity via gradients and divergences, making it physically incorrect to omit the velocity unknowns. Consequently, the coefficient vector in ELM-ODE must span three fields, substantially inflating the block matrices and their factorization cost. In three dimensions, the velocity field gains a third component, further increasing memory usage and computational effort. Nevertheless, for moderate 2D problems at practical accuracy, ELM-ODE can be significantly faster than a PINN-based solution while maintaining physically consistent results.

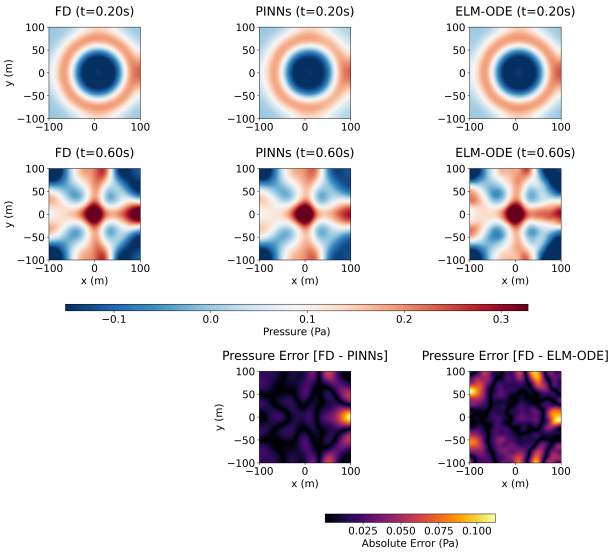

Figure 2. Comparison of FD, PINNs, and ELM-ODE solutions at $t = 0.2\,\mathrm{s}$ and $t = 0.6\,\mathrm{s}$ for the 2D domain. The bottom row shows the absolute errors for PINNs and ELM-ODE compared to the FD reference.

Table 8. Accuracy and cost for 2D simulation

Metric	PINNs	ELM-ODE
MAE $(t = 0.2)$ s	6.2221×10^{-3}	2.8990×10^{-3}
MAE $(t = 0.6)$ s	1.6631×10^{-2}	2.5553×10^{-2}
Training Time	7h 36m 52s	1362.424 s

6. CONCLUSION AND FUTURE WORK

We applied PINNs and ELM-ODE to solve the Linearized Euler Equations for outdoor acoustics in both 1D and 2D. ELM-ODE decoupled the spatial approximation from time integration and delivered faster solutions, despite with slightly larger errors in the same order. By contrast, PINNs attained slightly higher accuracy but demanded extensive training due to the full spatiotemporal residual.

Future work requires applying complex boundary conditions with frequency-dependent impedance and realistic ground surfaces to better match real-world acoustics. Scaling these methods to 3D remains challenging due to increased computational costs from the additional velocity component. Moreover, exploring alternative PDEs such as the convective Helmholtz equation or combining neural PDE solvers with measurement data or precomputed simulations appears promising.

7. REFERENCES

[1] M. Hornikx, "Ten questions concerning computational urban acoustics," *Building and Environment*, vol. 106, p. 409, 2016.

[2] J. E. Piercy, T. F. W. Embleton, and L. C. Sutherland, "Review of noise propagation in the atmosphere," *Journal of the Acoustical Society of America*, vol. 61, no. 6, pp. 1403–1418, 1977.

[3] R. Kapoor, N. Kloet, A. Gardi, A. Mohamed, and R. Sabatini, "Sound propagation modelling for manned and unmanned aircraft noise assessment and mitigation: A review," *Atmosphere*, vol. 12, no. 11, p. 1424, 2021.

[4] C. Tam, "Computational aeroacoustics: Issues and methods," *AIAA Journal*, vol. 33, no. 10, pp. 1788–1796, 1995.

[5] C. Bailly and D. Juve, "Numerical solution of acoustic propagation problems using linearized euler equations," *AIAA Journal*, vol. 38, no. 1, pp. 22–29, 2000.

[6] D. Botteldooren, "Finite-difference time-domain simulation of low-frequency room acoustic problems," *Journal of the Acoustical Society of America*, vol. 98, no. 6, pp. 3302–3308, 1995.

[7] R. Blumrich and D. Heimann, "A linearized eulerian sound propagation model for studies of complex meteorological effects," *The Journal of the Acoustical Society of America*, vol. 112, no. 2, pp. 446–455, 2002.

[8] N. Morales, V. Chavda, R. Mehra, and D. Manocha, "MPARD: A high-frequency wave-based acoustic solver for very large compute clusters," *Applied Acoustics*, vol. 121, pp. 82–94, 2017.

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

[10] A. Krishnapriyan, A. Gholami, S. Zhe, R. Kirby, and M. W. Mahoney, "Characterizing possible failure modes in physics-informed neural networks," in *Advances in Neural Information Processing Systems*, vol. 34, 2021.

[11] C. Datar, T. Kapoor, A. Chandra, Q. Sun, I. Burak, E. L. Bolager, A. Veselovska, M. Fornasier, and F. Dietrich, "Solving partial differential equations with sampled neural networks," *arXiv preprint arXiv:2405.20836*, 2024.

[12] T. Van Renterghem, "Efficient outdoor sound propagation modeling with the finite-difference time-domain (fdtd) method: a review," *International Journal of Aeroacoustics*, vol. 13, no. 5-6, pp. 385–404, 2014.

[13] G. E. Karniadakis, I. G. Kevrekidis, L. Lu, P. Perdikaris, S. Wang, and L. Yang, "Physics-informed machine learning," *Nature Reviews Physics*, vol. 3, no. 6, pp. 422–440, 2021.

[14] A. G. Baydin, B. A. Pearlmutter, A. A. Radul, and J. M. Siskind, "Automatic differentiation in machine learning: A survey," *Journal of Machine Learning Research*, vol. 18, no. 153, pp. 1–43, 2018.

[15] V. Sitzmann, J. Martel, A. Bergman, D. Lindell, and G. Wetzstein, "Implicit neural representations with periodic activation functions," *Advances in neural information processing systems*, vol. 33, pp. 7462–7473, 2020.

[16] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, "Extreme learning machine: theory and applications," *Neurocomputing*, vol. 70, no. 1-3, pp. 489–501, 2006.

[17] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," in *International Conference on Learning Representations (ICLR)*, 2019. arXiv:1711.05101.

[18] A. Jacot, F. Gabriel, and C. Hongler, "Neural tangent kernel: Convergence and generalization in neural networks," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 31, pp. 8571–8580, 2018.

[19] S. Wang, X. Yu, and P. Perdikaris, "When and why PINNs fail to train: A neural tangent kernel perspective," *Journal of Computational Physics*, vol. 449, p. 110768, 2021.