Forward and Inverse Problems with the Elastic Wave Equation: A Comparison of Traditional Numerical and Machine-Learning Methods

Tamanna Saini(tamannas@uoregon.edu) and Brittany A. Erickson (bae@uoregon.edu)
Department of Computer Science, University of Oregon

2025 SCEC ANNUAL MEETING Poster #308

Motivation

- GOAL: Compare grid-based Finite Difference Method(FDM) vs mesh-free Physics-Informed Neural Networks(PINNs) on the elastic wave equation for: the forward prediction of displacement u(x,t) and the inverse recovery of the shear modulus μ(x) from spare surface data.
- Why PINNs?: FDM is the accuracy/speed baseline. PINNs are mesh-free, easy incorporation of physics + constraints, and competitive accuracy for the hard enforcement implementation.
- Why Both?: Forward benchmarks set the accuracy/time baseline; inverse stresses data assimilation and parameter recovery.
- Design knobs: Soft vs hard constraint enforcement; L-BFGS optimizer implementation ; hotspot-aware collocation(see how well the method can learn an anomaly introduced in the equation for the forward problem) ; μ positivity/regularization(how well the method learns μ).

Physics-informed neural networks (PINNs)

Feed-forward deep neural network:

A single hidden layer with weights W and biases b

$$\ell(\mathbf{y};\theta) = \varphi(W\mathbf{y}+b), \quad \text{where } \theta = (W,b)$$

The recursive definition

$$\ell_0 = \mathbf{y},$$

$$\ell_k = \varphi_k(W_k \ell_{k-1} + b_k) \quad \text{for } 0 < k < L,$$

defines a feed-foward, deep neural network:

$$\mathcal{N}(\mathbf{y};\theta) = W_L \ell_{L-1} + b_L$$

PDEs as initial-boundary-value problems (IBVP):

differential operator → $\mathcal{L}[\mathbf{u};\lambda] = \mathbf{k}(\mathbf{x},t), \quad (\mathbf{x},t) \in \Omega \times [0,T],$ ← temporal domain; spatial domain

boundary operator → $\mathcal{B}[\mathbf{u};\lambda] = \mathbf{g}(\mathbf{x},t), \quad (\mathbf{x},t) \in \partial\Omega \times [0,T],$ ← boundary data

$\mathbf{u}(\mathbf{x},0) = \mathbf{u}_0(\mathbf{x}) \quad \mathbf{x} \in \Omega$ ← initial condition(s)

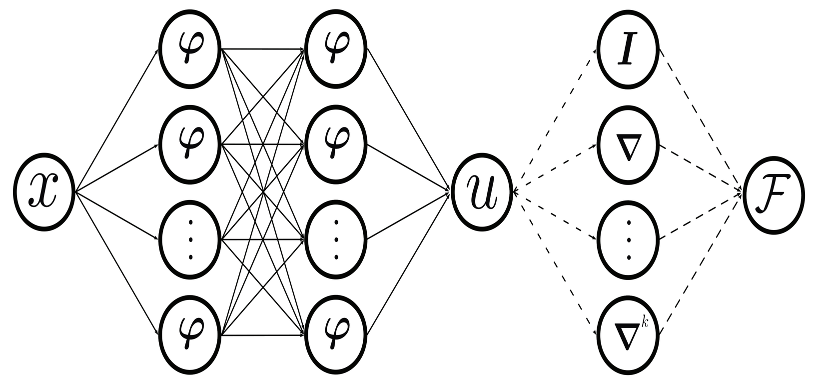


Figure 2: Network diagram for a generic PINN with activations φ, input x, output u and PDE $\mathcal{F}$. Network connections shown with dashed lines represent non-trainable parameters.

The Inverse-PINN objective:
Given ρ and limited surface observations, infer shear modulus μ(x) while predicting displacement u(x,t)
We aim to minimize the interior loss(residual) and learn the value of μ. Physics Residual (used in both forward and inverse):

$$\mathcal{R}[u,\mu](x,t) = \rho\,\partial_{tt}u(x,t) - \partial_x\big(\mu(x)\,\partial_x u(x,t)\big)$$

physics residual; Inverse(residula +data)

PINN architecture:

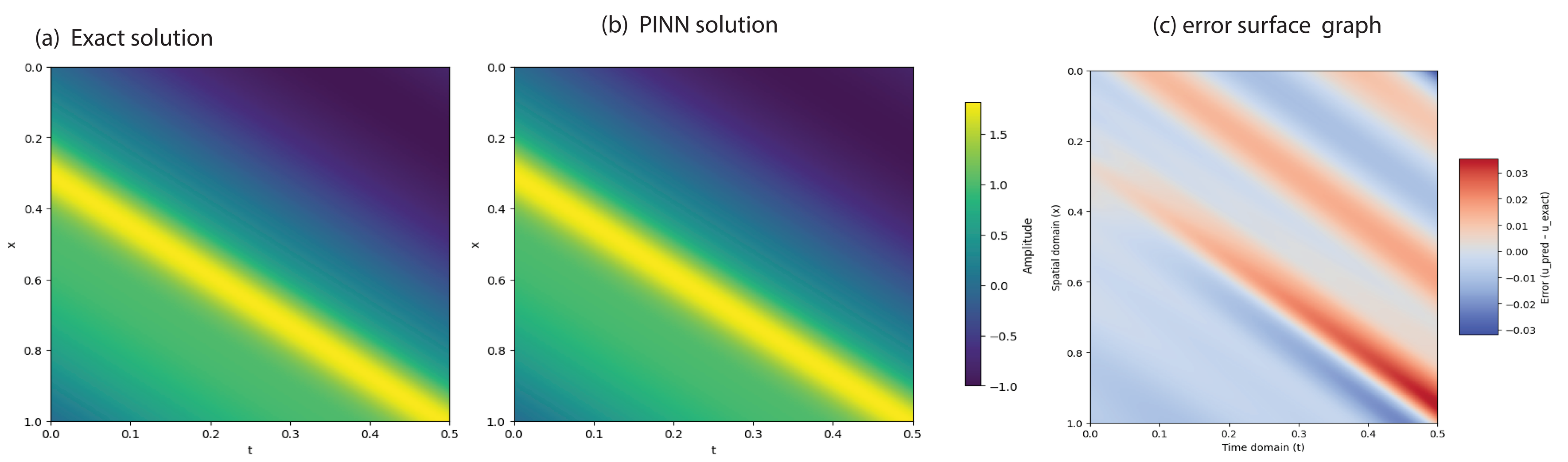
First we assume the solution to the IBVP is $u(\mathbf{x},t) \approx \mathcal{N}(\mathbf{x},t;\theta)$ and define the physics-informed neural network (PINN):

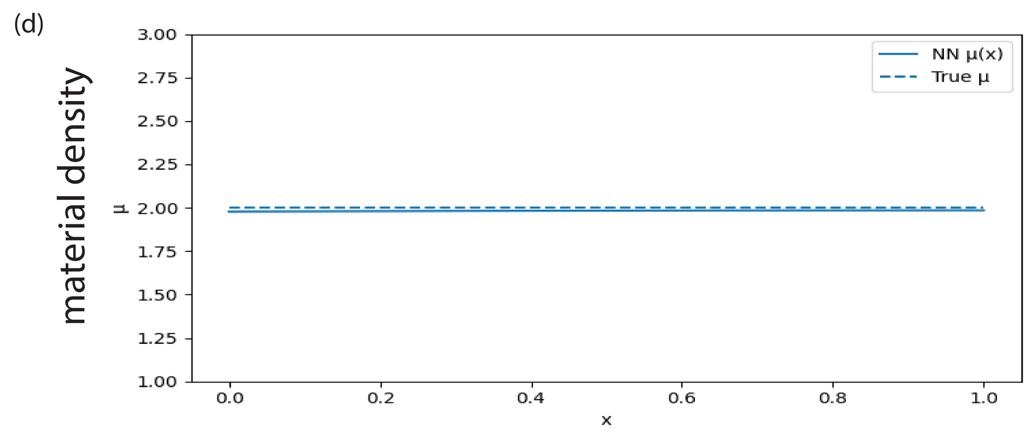
$$\mathcal{F} := \mathcal{L}[\mathcal{N};\lambda] - \mathbf{k}$$

Fig. 2 shows that both $\mathcal{F}$ and $\mathcal{N}$ have trainable nework parameters than can be learned by minimizing a mean squared error loss.

Verification: Learning PDEs, results for the forward and inverse problem

ID inverse problem: Inverting for μ





μ prediction

Figure 3: The plots show : a) Top-view of exact solution b)Top-view of PINN solution c)Signed difference between PINN predictions and exact solution d) predicted μ (dotted line) vs True μ value(solid line) plot.

Forward Problem for 1D elastic wave equation: Comparing FDM and LBFGS hard boundary enforcement solutions.

FDM implementation (amp=0.6)

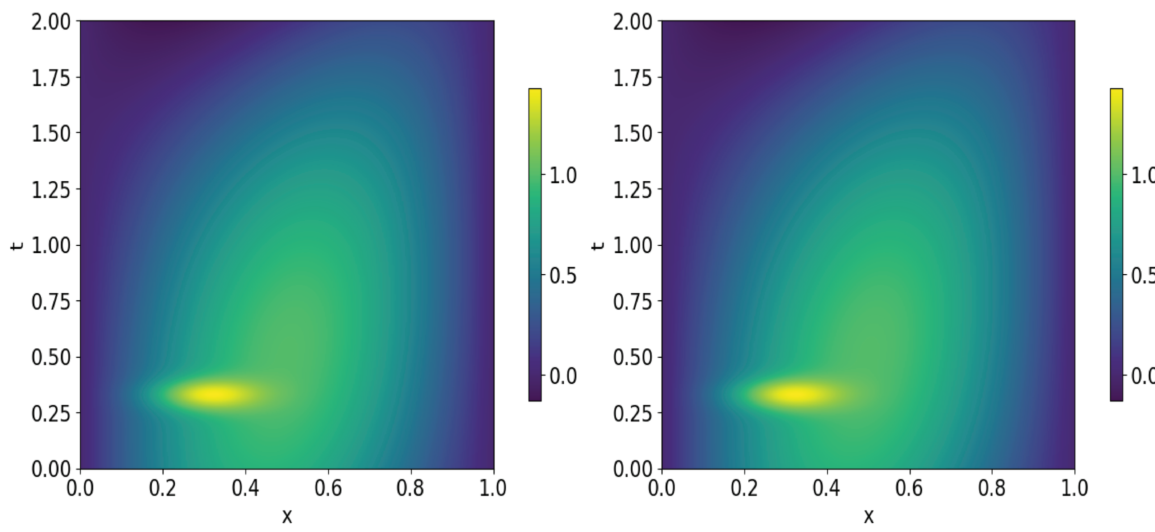


Figure 4: Top-view graph for exact solutipn(left) , top-view graph for FDM solution(right). Manufactured wavefield with narrow Gaussian perturbation(yellow bump) . Color encodes displacement u(x,t).

PINN implementation, with hard boundary enforcement for amp =0.6

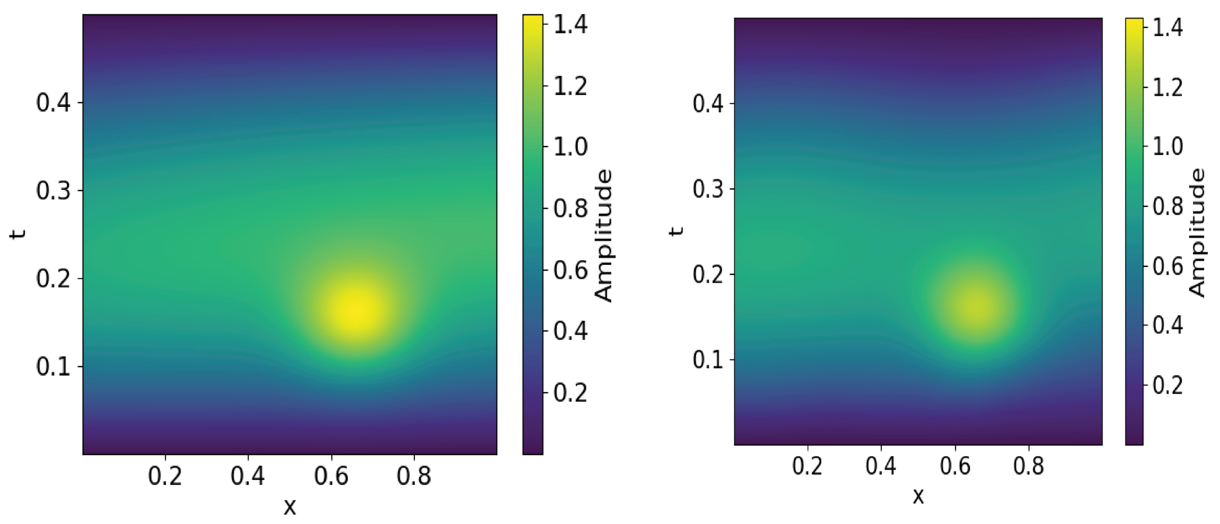


Figure 5: Top-view graph for exact solutipn(left) , top-view graph for PINN solution(right).

Table 1: Comparison of FDM vs. PINN

Method	L_2 (rel)	MSE	Total time (s)	Eval (s)	Iterations
FDM ($A=0$)	5.656E-5	5.823E-11	8.56	8.56	3200
FDM ($A=0.6$)	3.535E-6	2.280E-13	45.6	45.6	12800
Adam ($A=0$) (soft)	3.131E-3	4.000E-6	268	0.00092	6000
L-BFGS ($A=0$) (soft)	4.404E-4	8.664E-8	209	0.00088	10
Adam ($A=0.6$) (soft)	8.627E-2	3.786E-3	325	0.00089	6000
L-BFGS ($A=0.6$) (soft)	5.279E-2	1.417E-3	196	0.00070	10
Adam ($A=0.6$) (hard)	1.223E-4	7.622E-3	565	0.00261	4000
L-BFGS ($A=0.6$) (hard)	5.410E-5	1.486E-3	138	0.00013	10

Table 1: This table benchmarks every solver–optimizer combination across five metrics, for the forward problem

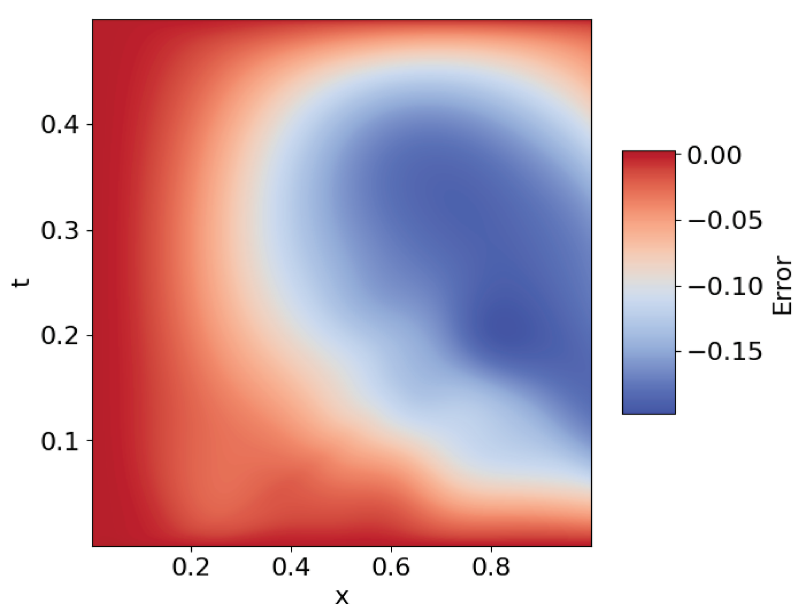


Figure 6: Error surface graph for exact vs PINN solution(fig 5).

PINN set-up and training

Step 1: Elastic Wave PDE:

$$\rho\,\partial_{tt}u(x,t) = \partial_x\big(\mu(x)\,\partial_x u(x,t)\big), \qquad (x,t) \in (0,L) \times (0,T].$$

Initial and Boundary conditions:

$$u(x,0) = u_0(x), \qquad \partial_t u(x,0) = v_0(x), \qquad u(0,t) = u(L,t) = 0.$$

Step 2: Hard enforced tial for u : (μ(x)≥0 via softplus)

$$u_\theta(x,t) = g(x,t) + B(x,t)\,N_\theta(x,t),$$

$$B(0,t) = B(L,t) = 0,\; g(x,0) = u_0(x),\; \partial_t g(x,0) = v_0(x).$$

Step 3: Define the mean-squared error loss

Forward Problem:

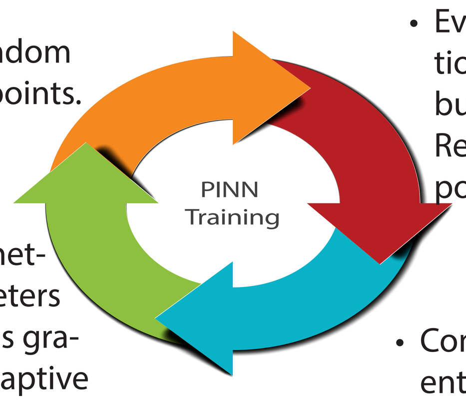
$$\mathcal{L}_{\mathrm{fwd}}(\theta) = \underbrace{\frac{1}{|\Omega|}\int_\Omega \big[\mathcal{R}[u_\theta,\mu](x,t)\big]^2\,\mathrm{d}x\,\mathrm{d}t}_{\mathcal{L}_{\mathrm{pde}}}$$

Inverse Problem:

$$\mathcal{L}_{\mathrm{inv}}(\theta,\phi) \approx \lambda_{\mathrm{pde}}\underbrace{\frac{1}{N_\Omega}\sum_{i=1}^{N_\Omega}\big[\mathcal{R}[u_\theta,\mu_\phi](x_i,t_i)\big]^2}_{\mathcal{L}_{\mathrm{pde}}} + \lambda_{\mathrm{data}}\underbrace{\frac{1}{N_{\mathrm{obs}}}\sum_{k=1}^{N_{\mathrm{obs}}}\big(u_\theta(x_k,t_k) - y_k\big)^2}_{\mathcal{L}_{\mathrm{data}}} + \lambda_\mu\underbrace{\mathcal{R}_\mu^{\mathrm{disc}}(\mu_\phi)}_{\mathrm{regularizer}}$$

*for inverse problems, known data is required.

Step 4: Training the PINN



- Generate random collocation points.
- Evaluate loss function - PyTorch's built in autograd. Resample interior points each epoch
- Compute gradient of loss function using backpropagation
- Update the network parameters using the loss gradient and adaptive moment estimation (e.g. Adam optimizer warmup, followed by L-BFGS).

- Ω: interior collocation set; (x_i, t_i) sampled by Sobol/LHS each epoch.
- Γ_obs: observation points for inverse; y_k are measured displacements.
- N_Ω, N_obs, N_x: counts of interior, observations, and spatial grid for μ-regularizer.
- λ_pde, λ_data, λ_μ: loss weights (tuned).
- Autodiff provides ∂_t, ∂_x and u_tt, (μ u_x)_x.

Summary, Future work & References

- In general, PINNs do not outperform traditional numerical methods for forward problems, HOWEVER they may offer improved and/or complimentary methods for inverse problems, and higher dimensional problems, allowing seamless integration of observational data, , particularly for applications requiring frequent point-wise evaluations where the instant query capabilities of PINNs surpass those of the FDM.
- Future work:
  - Sensitivity analysis of model outputs to model inputs, see how PINNs perform with real-world data.
  - Extend methods to 2D/3D elastic wave problems to evaluate computational feasibility and accuracy at scale.

References:
- C.ody Rucker, ``Physics-Informed Deep Learning of Rate-and-state Fault Friction,'' ,Journal of Geophysical Research: Solid Earth 2022
- Svetislav Savović et.al ,"A Comparative Study of the Explicit Finite Difference Method and Physics-Informed Neural Networks for Solving the Burgers' Equation", Axioms 2023
- Rasht, "Physics Informed Neural Networks (PINNs) for Wave Propagation and Full Waveform Inversions".