AB-PIELMs: Adaptive-Basis Physics-Informed Extreme Learning Machines for Residual-Driven Domain Decomposition

Aldis Daniel
Department of Mechanical Engineering
Indian Institute of Technology Madras
Chennai 600036, India
me22b070@smail.iitm.ac.in

Vikas Dwivedi
CREATIS, INSA-Lyon; Inserm, U1044; CNRS UMR 5220
Université Lyon 1; Université de Lyon
Lyon, 69621, France
vikas.dwivedi@creatis.insa-lyon.fr

Balaji Srinivasan
Wadhwani School of Data Science and Artificial Intelligence
Indian Institute of Technology Madras
Chennai 600036, India
sbalaji@dsai.iitm.ac.in

Abstract

Physics-informed neural solvers often struggle with multiscale behavior and sharp gradients due to the use of global approximation spaces and nonconvex training. We introduce AB-PIELM, an adaptive-basis physics-informed extreme learning machine that combines domain decomposition with joint optimisation of spatial and spectral hyperparameters while retaining deterministic normal-equation training. The method adapts both the number and placement of subdomains and the locality of radial basis functions, enabling problem-dependent allocation of representational capacity without modifying the underlying solver. Experiments on oscillatory function approximation and singularly perturbed advection–diffusion equations demonstrate accurate solutions across a wide range of stiffness regimes while using substantially fewer neurons than existing neural and PIELM-based approaches. The learned hyperparameters are interpretable, concentrating resolution near sharp gradients. The framework also supports inverse problems, successfully recovering diffusion coefficients from sparse noisy observations using Bayesian optimisation. These results indicate that adaptive-basis PIELM formulations provide an efficient and stable alternative to gradient-trained neural PDE solvers and naturally extend to broader classes of differential equations.

1 Introduction

Recent years have seen rapid growth in machine-learning approaches for solving partial differential equations (PDEs) and surrogate modeling of physical systems. Among these, physics-informed neural networks (PINNs) embed governing equations and boundary conditions directly into the loss function, enabling data-efficient learning without labeled solutions (Raissi et al.). Subsequent developments have applied physics-informed learning to forward and inverse problems, parametric studies, and uncertainty quantification (Karniadakis et al.). However, most neural PDE solvers rely on global function approximators defined over the entire spatial domain. Such representations implicitly assume uniform smoothness and are therefore sensitive to stiffness, multiscale structure, and

sharp gradients. As a result, PINNs may exhibit slow convergence or failure to train, caused by gradient pathologies and ill-conditioned optimization landscapes (Wang et al., a; Krishnapriyan et al.; Wang et al., b). From a spectral perspective, this behavior reflects the low-frequency bias of neural networks, which limits the efficient representation of localized high-frequency features (Rahaman et al.).

To overcome these limitations, several works introduce localized neural approximations. Domain-decomposition methods such as cPINNs, XPINNs, and related parallel formulations train multiple networks on subdomains coupled through interface conditions (Jagtap et al.; D. Jagtap & Em Karniadakis; Shukla et al.; Klawonn et al.). Other approaches incorporate spectral enrichment, for example via Fourier feature mappings (Tancik et al.), or adaptive decomposition strategies (Botvinick-Greenhouse et al.). In parallel, deterministic alternatives replace gradient training with linear-in-parameter models. Extreme Learning Machines (ELMs) fix nonlinear feature maps and solve for coefficients directly (Huang et al.), and physics-informed extensions such as the normal-equation PIELM (NEPIELM) provide stable solvers that admit domain-decomposed formulations (Dwivedi & Srinivasan; Anderson et al.; Dwivedi et al.). While these methods improve robustness relative to global networks, domain partitions and basis parameters are typically prescribed, limiting adaptation to localized solution features.

Despite these advances, existing approaches either rely on nonconvex optimization or require manually designed decompositions and hyperparameters, leading to complicated training procedures, increased computational cost, and reduced interpretability compared to traditional numerical discretizations. In this work we propose AB-PIELM, a parameterisable domain-decomposed physics-informed extreme learning machine that explicitly identifies spatial and spectral hyperparameters and optimizes them within a single physics-informed objective, while retaining deterministic normal-equation training for fixed hyperparameters. The computational efficiency of the underlying PIELM allows for optimisation of the network architecture whereas existing methods rely on residual-based refinement of collocation distributions.

The main contributions of this work are:

- We identify key spatial and spectral hyperparameters in the formulation and propose their joint optimization within a single physics-informed objective.
- We rigorously compare our method against state of the art approaches like XTFC and distributed NE-PIELM on a benchmark ODE.
- We demonstrate the interpretability of the resulting domain partitions and basis parameters.
- We formulate the inverse problem and solve it using Bayesian optimisation, demonstrating the method's flexibility.

The results shown are in benchmark cases as a proof of concept and the approach naturally extends to PDEs. The code is made available at `https://github.com/catdisk04/AB-PIELM` to facilitate future research.

2 METHODOLOGY

The proposed method combines the ideas of Adaptive Mesh Refinement (AMR) and modern hyperparamter optimisation techniques in the context of PIELMs. This allows for a more flexible and adaptive architecture that can better capture sharp gradients and complex solution features while retaining the computational efficiency of PIELMs that are trained using the normal equation formulation (Dwivedi & Srinivasan).

2.0.1 AB-PIELM SOLUTION SCHEME

In this section, we provide a brief overview of the distributed formulation used in this work.

We consider linear partial differential equations of the form

$$\mathcal{L}u(x) = f(x), \qquad x \in \Omega \subset \mathbb{R}^d, \tag{1}$$

subject to boundary conditions $\mathcal{B}u(x) = g(x)$ over $x \in \partial\Omega$, where $\mathcal{L}$ is a linear differential operator and $\mathcal{B}$ is a boundary operator. The domain Ω is partitioned into M non-overlapping subdomains

$\{\Omega_m\}_{m=1}^M$ with interfaces $\Gamma_{mn} = \partial\Omega_m \cap \partial\Omega_n$. Each network has output layer weights β and hidden layer basis functions Φ that are local to the subdomain. The solution in each subdomain is given by

$$\hat{u}_m(x) = \Phi_m(x)\beta_m, \qquad x \in \Omega_m. \tag{2}$$

Let $\{x_i^{(m)}\}$ denote collocation points in Ω_m. We define the discrete PDE operator P_m and the discrete boundary operator B_m by

$$P_m\beta_m = \left(\mathcal{L}\Phi_m(x_i^{(m)})\beta_m\right)_i, \tag{3}$$

$$B_m\beta_m = \left(\mathcal{B}\Phi_m(x_i^{(m)})\beta_m\right)_i. \tag{4}$$

For each interior interface Γ_{mn}, we define trace and flux operators

$$C_{m|n}\beta_m = \Phi_m(x)\beta_m\big|_{x\in\Gamma_{mn}}, \tag{5}$$

$$D_{m|n}\beta_m = \partial_n\Phi_m(x)\beta_m\big|_{x\in\Gamma_{mn}}, \tag{6}$$

where ∂_n denotes the outward normal derivative on $\partial\Omega_m$.

The global objective function is defined as

$$J(\beta) = J_{PDE}(\beta) + J_{BC}(\beta) + J_{INT}(\beta), \tag{7}$$

where

$$J_{PDE}(\beta) = \sum_{m=1}^{M} \|P_m\beta_m - f_m\|_2^2, \tag{8}$$

$$J_{BC}(\beta) = \gamma_b \sum_{m=1}^{M} \|B_m\beta_m - g_m\|_2^2, \tag{9}$$

$$J_{INT}(\beta) = \sum_{(m,n)\in\mathcal{I}} \Big(\gamma_c\|C_{m|n}\beta_m - C_{n|m}\beta_n\|_2^2 + \gamma_d\|D_{m|n}\beta_m - D_{n|m}\beta_n\|_2^2\Big), \tag{10}$$

and $\mathcal{I}$ denotes the set of neighboring subdomain pairs and $\gamma_b, \gamma_c, \gamma_d > 0$ are penalty parameters.

The first-order optimality condition $\nabla_\beta J = 0$ yields the block linear system

$$A\beta = b, \tag{11}$$

with diagonal blocks

$$A_{mm} = P_m^\top P_m + \gamma_b B_m^\top B_m + \sum_{n\in\mathcal{N}(m)} \left(\gamma_c C_{m|n}^\top C_{m|n} + \gamma_d D_{m|n}^\top D_{m|n}\right), \tag{12}$$

and off-diagonal coupling blocks

$$A_{mn} = -\left(\gamma_c C_{m|n}^\top C_{n|m} + \gamma_d D_{m|n}^\top D_{n|m}\right), \qquad m \neq n. \tag{13}$$

The right-hand side is given by

$$b_m = P_m^\top f_m + \gamma_b B_m^\top g_m. \tag{14}$$

The resulting system is symmetric and positive semi-definite, with block sparsity induced by the domain decomposition. The system is solved directly as

$$\beta = A^\dagger b, \tag{15}$$

where $A^\dagger$ denotes the Moore-Penrose pseudoinverse of A.

2.1 JOINT SPATIAL–SPECTRAL OPTIMIZATION

The PDE loss (Eq 7) of the PIELM network depends on the set of structural hyperparameters $\theta = (\theta_{\text{spatial}}, \theta_{\text{spectral}})$. The spatial hyperparameters specify the domain decomposition via the interface locations

$$\theta_{\text{spatial}} = \{x_{\text{if}}^{(k)}\}_{k=1}^{M-1}, \tag{16}$$

where the number of domains M is also a hyperparameter, thereby defining the subdomains $\Omega_m(\theta_{\text{spatial}})$. The spectral hyperparameters control the locality of the RBF basis in each subdomain through the width parameters

$$\theta_{\text{spectral}} = \{\sigma_m\}_{m=1}^{M}. \tag{17}$$

$$\sigma_m = \alpha_m \, \Delta c_m, \tag{18}$$

where Δc_m denotes RBF center spacing and σ_m the RBF widthin Ω_m.

The loss term $J_{PDE}(\beta; \theta)$ therefore depends on θ through both the discrete interface operators and the basis functions of the PIELM formulation. We jointly optimize

$$\min_{\theta \in \Theta} \; \min_{\beta} \; J(\beta; \theta), \tag{19}$$

where Θ is the hyperparameter search space. For fixed θ, the inner minimization is solved exactly as described for the underlying PIELM. This joint optimization adapts both the domain decomposition and the spectral sharpness of the basis in a problem-dependent manner, concentrating resolution near localized features while preserving global coherence across interfaces. The dimensionality of β, θ_{spatial}, and θ_{spectral} are dependant on M. Hence, optimisation of M is separated from the optimisation of θ and executed in an outer loop in an incremental fashion to result in the optimal number of domains M^* being as small as possible in the interest of model simplicity.

Since each local network has a fixed number of RBFs and collocation points, this framework results in adaptiveness in three levels: The number of domains, number of RBFs and the total number of collocation points. The methodology is summarised in Algorithm 1.

Algorithm 1 AB-PIELM Hyperparameter Optimisation Scheme

Require: Boundary Conditions
Require: Hyperparameter search space
Require: Initialise M, J_{min}, J_{min}^{M} and θ
while not *converged* **do**
  while $N_{iter} < N_{max}$ **do**
    Generate collocation set
    $\beta^* = \arg\min_\beta J_{PDE}(\beta; \theta)$ (Solve for output layer weights using normal equation)
    Obtain J from the trained optimal network
    Evaluate performance on testing set
    Update hyperparameters using TPE algorithm
    if $J < J_{min}$ **then**
      Update J_{min} = J
      Store optimal hyperparameters θ^*
    end if
  end while
  if $J_{min} < J_{min}^{M}$ **then**
    Update J_{min}^{M} and store optimal hyperparameters θ^* for M
    Increment M
    Reset J_{min}
  else
    Update *converged* to True
    Store optimal hyperparameters θ^* for M^*
    break
  end if
end while
Obtain final trained model with optimal hyperparameters θ^* and M^*
Evaluate performance on validation set
return Trained AB-PIELM model

3 NUMERICAL EXPERIMENTS

To evaluate the effectiveness and robustness of our approach, we consider two cases:

1. Oscillatory function (Taylor et al.)
2. Singularly perturbed differential equation (De Florio et al.)

The *sinc* function and the advection–diffusion equation serve as complementary benchmark problems. The oscillatory *sinc* function with decaying amplitude evaluates the expressive capacity and training behaviour of the approximation architecture, while the advection–diffusion equation — a canonical singularly perturbed model — assesses the ability of the domain-decomposed PIELM to approximate PDE solutions under stiffness and varying Péclet numbers while maintaining interface continuity and differentiability across subdomains.

3.1 PROBLEM 1: SINC FUNCTION APPROXIMATION

This test case considers the scalar *sinc* function for a function approximation benchmark. In this work, the normalized *sinc* function is defined as

$$sinc(x) = \begin{cases} 1, & x = 0, \\ \dfrac{\sin(n\pi x)}{n\pi x}, & x \neq 0, \end{cases} \tag{20}$$

where $n = 10$, $x \in [-1.5, 1.5]$

The loss is purely data-driven and constructed from sampling $sinc(x)$ at a set of collocation points. This setup isolates the effect of the architecture and domain decomposition on approximating an oscillatory function and thereby provides an useful analsis of how the number of neurons and the interface location influence the representation of oscillatory data.

3.2 PROBLEM 2: ADVECTION–DIFFUSION

The one-dimensional advection diffusion equation is defined as follows:

$$-\nu \frac{\mathrm{d}^2 u}{\mathrm{d}x^2} + \frac{\mathrm{d}u}{\mathrm{d}x} = 0, \qquad x \in (0, 1), \tag{21}$$

with the Dirichlet boundary conditions

$$u(0) = 0, \qquad u(1) = 1, \tag{22}$$

where $\nu > 0$ is the diffusion coefficient.

3.3 INVERSE PROBLEM SETUP

We aim to identify the diffusion coefficient ν in the advection-diffusion equation equation 21 by fitting the model to synthetic data that is sparse and noisy, generated from a known value of ν.

For a fixed ν and model hyperparamters determined as defined for the forward problem, the trained output layer weights β minimise the physics-informed loss and results in

$$\hat{\mathbf{u}}(\nu) = \Phi_{obs}(\nu)\beta(\nu) \tag{23}$$

where $\Phi_{obs}(\nu)$ is the matrix of basis functions evaluated at the observation points. The inverse problem is then formulated as the minimisation of a outer objective

$$J_{inv}(\nu) = \|\hat{\mathbf{u}}(\nu) - \mathbf{u}_{data}\|_2^2 + \lambda_{PDE} J_{PDE}(\beta(\nu); \nu), \tag{24}$$

where the J_{PDE} part of the objective contributed by the discrete PDE operator and λ_{PDE} is a penalty parameter that balances the data fit and physics-informed terms. The inverse problem is

$$\nu^* = \underset{\nu \in (\nu_{min}, \nu_{max})}{\arg\min} \quad J_{inv}(\nu) \tag{25}$$

where ν_{min} and ν_{max} define the user-defined search space for the physical parameter ν. The outer optimization is performed using Bayesian optimization with a TPE algorithm. This nests the forward model in a outer inverse optimisation loop, allowing for parameter recovery that is simultaneously constrained by data and physics.

4 Results and Discussion

4.1 Problem 1: *sinc* Function Approximation

Figure 1 shows the MSE residual as a function of the number of subdomains. An elbow is observed at 5 subdomains, which is selected as the optimal number. The AB-PIELM approximation of the *sinc* function using 5 subdomains is shown in Figure 2a, while the MSE residual plot is shown in Figure 2b. Overall, the results demonstrate that the proposed framework can effectively approximate highly oscillatory functions by optimising the domain decomposition.

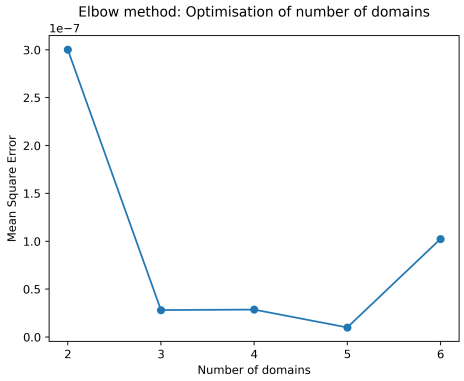
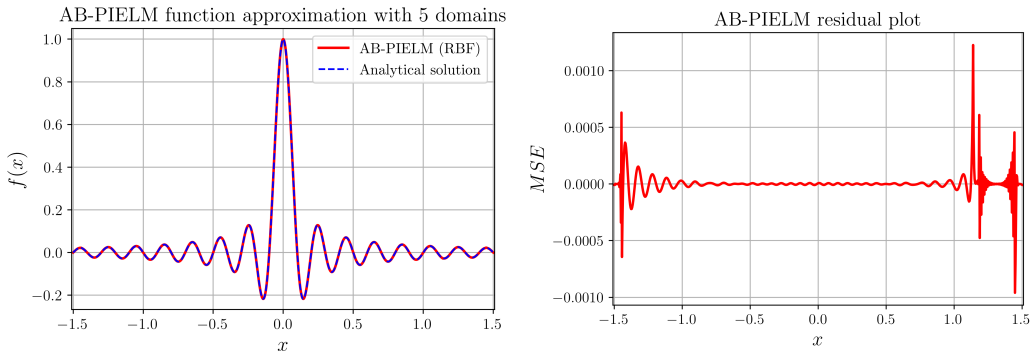


Figure 1: Plot showing the optimisation the number of subdomains on the MSE residual for the *sinc* function approximation using the Elbow method.



(a) (b)

Figure 2: AB-PIELM approximation of the *sinc* function using 5 subdomains (a) and the corresponding MSE residual plot (b).

4.2 Problem 2: Advection–Diffusion

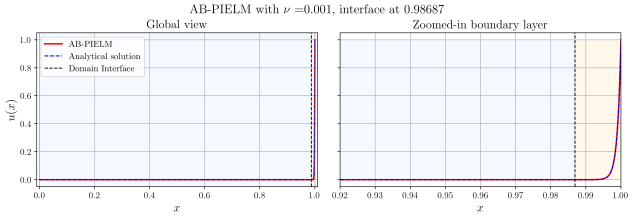
Figure 3 shows that the AB-PIELM solutions closely match the analytical solution of the advection–diffusion equation. The Mean Absolute Error (MAE) residual plot is shown in Figure 4. The

Table 1: Comparison of performance metrics with existing literature for *sinc* function approximation

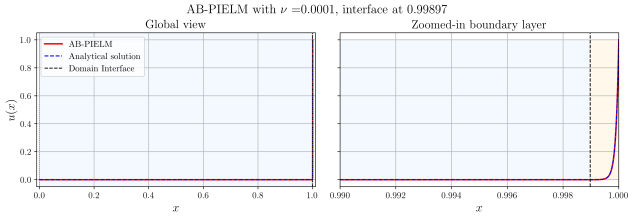
Metric	PINN-LM (Taylor et al.)	AB-PIELM
Number of neurons	9620	250
MSE	1.64×10^{-7}	9.91×10^{-9}

performance of the AB-PIELM is also compared with other methods, namely PIELM and XTFC. These methods were the best performers in a comparison of Neural Networks tested on the advection–diffusion problem in (De Florio et al.). The performance of the vanilla distributed NEPIELM (DNEPIELM) is also reported for comparison.The results are summarized in Table 2.

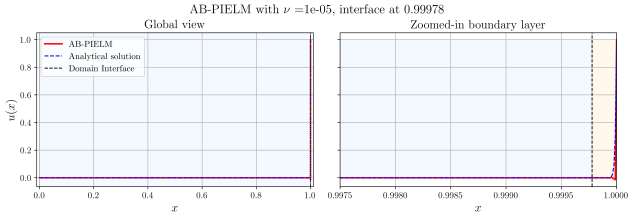
In summary, the AB-PIELM is significantly more accurate than the PIELM in all the values of ν considered, while using two orders of magnitude fewer neurons (PIELM is reported to entirely fail for the $\nu = 0.0001$ case). The AB-PIELM also outperforms the XTFC method in the two cases with the sharpest gradients ($\nu = 0.0001$ and $\nu = 0.00001$) while using two orders of magnitude fewer neurons, thereby demonstrating the better or comparable accuracy of the proposed method with significantly reduced model size.



(a)



(b)

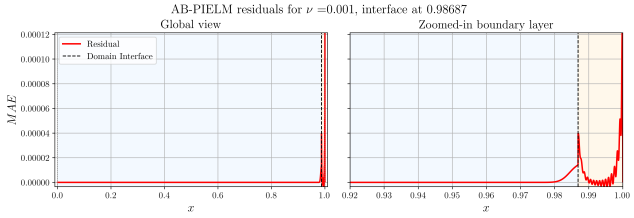


(c)

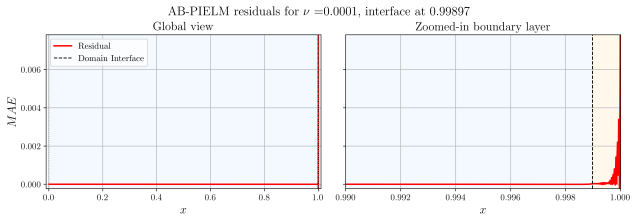
Figure 3: Advection–diffusion equation solutions for different diffusion coefficients: (a) $\nu = 10^{-3}$, (b) $\nu = 10^{-4}$, and (c) $\nu = 10^{-5}$. The AB-PIELM solutions are compared against the analytical solutions.

4.2.1 Interpretibility of optimised hyperparameters

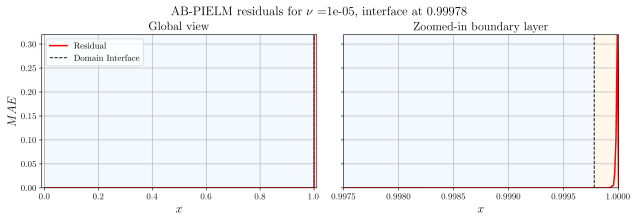
The optimisation of hyperparameters in AB-PIELM reveals important insights into the model's behavior. The number of subdomains and the placement of interfaces are critical factors that influence the model's representation capacity and accuracy. Table 3 shows that for all three values of the diffusion coefficient ν, the optimised interface is located close to the sharp gradient region thus maximising the representation capacity where it is most needed. It is also observed that the σ values in the left domain (smooth region) are significantly larger than those in the right domain (sharp



(a)



(b)



(c)

Figure 4: MAE residual plots for the advection–diffusion equation for different diffusion coefficients: (a) $\nu = 10^{-3}$, (b) $\nu = 10^{-4}$, and (c) $\nu = 10^{-5}$.

Table 2: Comparison of performance metrics number of neurons L_{tot} and Mean Absolute Error for advection–diffusion equation at different diffusion coefficient ν values using different methods from literature(De Florio et al.)XTFC (De Florio et al.)

ν	Metric	PIELM	XTFC	DNEPIELM	AB-PIELM
0.001	L_{tot}	10000	10000	100	100
	MAE	4.89×10^{-5}	7.09×10^{-13}	2.25×10^{-4}	1.85×10^{-7}
0.0001	L_{tot}	10000	10000	400	100
	MAE	5.0×10^{-1}	1.39×10^{-3}	1.99×10^{-4}	4.56×10^{-7}
0.00001	L_{tot}	-	-	400	100
	MAE	-	-	4.37×10^{-4}	6.41×10^{-6}

gradient region). This indicates that the model requires a finer resolution (smaller σ) to accurately capture the steep gradient, while a coarser resolution (larger σ) suffices in the smooth region.

4.3 Inverse Problem Results

Table 4 shows the recovered diffusion coefficient ν^* for different noise levels in the synthetic data. AB-PIELM is able to accurately recover the true diffusion coefficient even in the presence of noise upto 5%, demonstrating the robustness of the method for inverse problems. The results also show that as the noise level increases, the recovered ν^* deviates more from the true value, which is expected due to the increased uncertainty in the data. However, even at a noise level of $\sigma^2_{noise} = 0.05$, the recovered ν^* remains reasonably close to ν_{true}, indicating that AB-PIELM can effectively handle noisy data in inverse problem settings.

Limitations

Table 3: Comparison of hyperparameters over domain 1 (left) containing the smooth region and domain 2 (right) containing the sharp gradient for advection–diffusion equation at different diffusion coefficients

Diffusion coefficient	Domain	Domain length	σ
$\nu = 0.001$	Domain 1	0.9869	0.0025
	Domain 2	0.0131	9.311×10^{-4}
$\nu = 0.0001$	Domain 1	0.9990	0.0026
	Domain 2	0.0010	4.310×10^{-5}
$\nu = 0.00001$	Domain 1	0.99999	0.1715
	Domain 2	0.00001	5.509×10^{-8}

Table 4: Recovered diffusion coefficient ν for different noise levels

ν_{true}	σ^2_{noise}	Recovered ν^*
0.005	0	0.0049172
	0.01	0.0049917
	0.05	0.0054590
0.0005	0	0.0005181
	0.01	0.0005236
	0.05	0.0004491

- **Linear operators:** The normal–equation formulation relies on linearity of the PDE in the solution coefficients. Nonlinear PDEs would require outer iterative linearization, reducing the deterministic training advantage.
- **Scaling to high dimensions:** Domain decomposition introduces interface constraints whose number grows rapidly with dimension, leading to larger coupled systems and increased memory cost.
- **Unsteady problems:** The present formulation addresses steady equations. Extending to time-dependent PDEs would require time marching or space–time decompositions, significantly increasing system size and conditioning complexity.
- **Penalty parameters:** The weights in the physics-informed objective must be specified a priori since they define the objective itself and cannot be optimized within the same loss.

5 Conclusion

This work introduced a distributed adaptive physics-informed extreme learning machine that augments the normal-equation PIELM formulation with optimisable spatial and spectral hyperparameters. By allowing both the domain partition and the locality of the basis functions to adapt to the solution structure, the method addresses a central limitation of global neural PDE solvers: the inability to efficiently represent localized behaviour using a uniform approximation space. The resulting formulation retains deterministic training while enabling problem-dependent allocation of representational capacity. Numerical experiments demonstrated accurate approximation of oscillatory targets and robust solutions of singularly perturbed advection–diffusion equations, as well as stable recovery of the diffusion coefficient from sparse and noisy observations. The learned hyperparameters were also interpretable, with interfaces concentrating near sharp gradients and basis widths varying according to local solution smoothness.

References

Samuel Anderson, Victorita Dolean, Ben Moseley, and Jennifer Pestana. ELM-FBPINN: efficient finite-basis physics-informed neural networks. URL https://arxiv.org/abs/2409.01949.

Jonah Botvinick-Greenhouse, Wael H. Ali, Mouhacine Benosman, and Saviz Mowlavi. AB-PINNs: Adaptive-basis physics-informed neural networks for residual-driven domain decomposition. URL https://arxiv.org/abs/2510.08924.

Ameya D. Jagtap and George Em Karniadakis. Extended physics-informed neural networks (XPINNs): A generalized space-time domain decomposition based deep learning framework for nonlinear partial differential equations. 28(5):2002–2041. ISSN 1815-2406, 1991-7120. doi: 10.4208/cicp.OA-2020-0164. URL https://global-sci.org/cicp/article/view/6911.

Mario De Florio, Enrico Schiassi, Francesco Calabrò, and Roberto Furfaro. Physics-informed neural networks for 2nd order ODEs with sharp gradients. 436:115396. ISSN 03770427. doi: 10.1016/j.cam.2023.115396. URL https://linkinghub.elsevier.com/retrieve/pii/S0377042723003400.

Vikas Dwivedi and Balaji Srinivasan. A normal equation-based extreme learning machine for solving linear partial differential equations. 22(1):014502. ISSN 1530-9827, 1944-7078. doi: 10.1115/1.4051530. URL https://asmedigitalcollection.asme.org/computingengineering/article/22/1/014502/1111555/A-Normal-Equation-Based-Extreme-Learning-Machine.

Vikas Dwivedi, Enrico Schiassi, Monica Sigovan, and Bruno Sixou. Gated x-TFC: Soft domain decomposition for forward and inverse problems in sharp-gradient PDEs. URL https://arxiv.org/abs/2510.01039.

Guang-Bin Huang, Qin-Yu Zhu, and Chee-Kheong Siew. Extreme learning machine: Theory and applications. 70(1):489–501. ISSN 09252312. doi: 10.1016/j.neucom.2005.12.126. URL https://linkinghub.elsevier.com/retrieve/pii/S0925231206000385.

Ameya D. Jagtap, Ehsan Kharazmi, and George Em Karniadakis. Conservative physics-informed neural networks on discrete domains for conservation laws: Applications to forward and inverse problems. 365:113028. ISSN 00457825. doi: 10.1016/j.cma.2020.113028. URL https://linkinghub.elsevier.com/retrieve/pii/S0045782520302127.

George Em Karniadakis, Ioannis G. Kevrekidis, Lu Lu, Paris Perdikaris, Sifan Wang, and Liu Yang. Physics-informed machine learning. 3(6):422–440. ISSN 2522-5820. doi: 10.1038/s42254-021-00314-5. URL https://www.nature.com/articles/s42254-021-00314-5.

Axel Klawonn, Martin Lanser, and Janine Weber. Machine learning and domain decomposition methods - a survey. 1(1):2. ISSN 2948-1597. doi: 10.1007/s44207-024-00003-y. URL https://link.springer.com/10.1007/s44207-024-00003-y.

Aditi S. Krishnapriyan, Amir Gholami, Shandian Zhe, Robert M. Kirby, and Michael W. Mahoney. Characterizing possible failure modes in physics-informed neural networks. URL http://arxiv.org/abs/2109.01050.

Nasim Rahaman, Aristide Baratin, Devansh Arpit, Felix Draxler, Min Lin, Fred A. Hamprecht, Yoshua Bengio, and Aaron Courville. On the spectral bias of neural networks. URL http://arxiv.org/abs/1806.08734.

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. 378:686–707. ISSN 00219991. doi: 10.1016/j.jcp.2018.10.045. URL https://linkinghub.elsevier.com/retrieve/pii/S0021999118307125.

Khemraj Shukla, Ameya D. Jagtap, and George Em Karniadakis. Parallel physics-informed neural networks via domain decomposition. URL http://arxiv.org/abs/2104.10013.

Matthew Tancik, Pratul P. Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan T. Barron, and Ren Ng. Fourier features let networks learn high frequency functions in low dimensional domains. URL http://arxiv.org/abs/2006.10739.

John Taylor, Wenyi Wang, Biswajit Bala, and Tomasz Bednarz. Optimizing the optimizer for data driven deep neural networks and physics informed neural networks. URL https://arxiv.org/abs/2205.07430.

Sifan Wang, Yujun Teng, and Paris Perdikaris. Understanding and mitigating gradient flow pathologies in physics-informed neural networks. 43(5):A3055–A3081, a. ISSN 1064-8275, 1095-7197. doi: 10.1137/20M1318043. URL https://epubs.siam.org/doi/10.1137/20M1318043.

Sifan Wang, Xinling Yu, and Paris Perdikaris. When and why PINNs fail to train: A neural tangent kernel perspective. 449:110768, b. ISSN 00219991. doi: 10.1016/j.jcp.2021.110768. URL https://linkinghub.elsevier.com/retrieve/pii/S002199912100663X.