IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2017.DOI

When Random Features Beat the Mesh: Benchmarking Physics-Informed Extreme Learning Machines against Finite Elements

ANDREA SACCHETTI
In situte for Mathematics and Natural Sciences, University of Applied Sciences and Arts Northwestern Switzerland (FHNW), Windisch, CH-5210 Switzerland (e-mail: andrea.sacchetti@fhnw.ch)

Corresponding author: Andrea Sacchetti (e-mail: andrea.sacchetti@fhnw.ch).

ABSTRACT Physics-informed extreme learning machines (PIELMs) have emerged as optimisation-free surrogates to gradient-based physics-informed neural networks, yet their quantitative advantage over classical finite-element methods (FEM) has remained limited to regular-grid domains. This work delivers the first systematic, large-scale head-to-head study between a minimalist single-R_m PIELM and FEM on four elliptic benchmarks of escalating difficulty with random collocation points: (i) a 2-D Poisson verification case, (ii) a 2-D heat-conduction problem with an internal cavity, (iii) a 2-D hyperbolic PDE featuring steep gradients, and (iv) the 3-D extension of (ii). More than 4×10^3 random hyper-parameter realisations are executed, spanning seven decades in training loss and error. For the two smooth 2-D benchmarks the PIELM attains sub-10^{-8} root-mean-square (RMS) errors while being at least an order of magnitude faster than equally accurate FEM. On the hyperbolic test it retains its speed edge and lowers the FEM error by one order of magnitude. In 3-D the PIELM reaches RMS values within a factor 4 of the FEM optimum yet requires only a fraction of the wall-time, revealing dimensionality limits but preserving favourable accuracy-per-second scaling.

Statistical analysis uncovers an almost linear log–log correlation between the collocation residual and the RMS error, enabling a practical a-posteriori stopping criterion. Sensitivity sweeps identify the hidden-layer width M and the weight scale R_m as the dominant hyper-parameters, whereas the interior-to-boundary collocation ratio primarily governs stability. Error visualisations expose complementary failure modes between PIELM (local spikes) and FEM (global diffusion), suggesting the need for more robust collocation-points selection strategies.

Overall, the evidence positions the single-R_m PIELM as a compelling alternative for fast, high-accuracy solutions of low- to moderate-dimensional, smooth PDEs, and charts a roadmap for overcoming the remaining challenges in high-dimensional and highly non-smooth regimes.

INDEX TERMS Enter key words or phrases in alphabetical order, separated by commas. For a list of suggested keywords, send a blank e-mail to keywords@ieee.org or visit http://www.ieee.org/organizations/pubs/ani_prod/keywrd98.txt

I. INTRODUCTION

Neural-network-based methods for solving partial differential equations (PDEs) have seen renewed interest, starting with early spectral–collocation network approaches [1], [2], and culminating in the development of physics-informed neural networks (PINNs) [3], [4]. PINNs embed PDE residuals and boundary conditions into a neural network's loss function, enabling mesh-free, flexible simulation across complex domains and facilitating inverse problem solving [3], [4].

Despite their appeal, PINNs often face practical challenges including slow convergence and poor handling of stiff or high-gradient problems, due to spectral bias and imbalanced loss terms [5], [6]. Additionaly, our previous work showed

that they may struggle to actually outperform FEM at least on simple problems [7]. As a faster alternative, extreme learning machines (ELMs) fix hidden-layer weights randomly and solve only the output layer through linear least squares [8]–[10]. When informed by physics, they yield physics-informed ELMs (PIELMs), offering orders-of-magnitude speedups over gradient-based PINNs with similar accuracy [11]. Among early applications of ELM to PDEs, Panghal and Kumar demonstrated that an optimization-free ELM approach could dramatically reduce computation time while delivering high accuracy for both ODEs and PDEs—even in 3D settings—compared to traditional optimization-based neural solvers [12]. Their results reinforce the appeal of PIELMs for rapid prototyping and low-latency inference. Building on the classical PIELM framework, Dong and Yang [8] introduced a minimal yet powerful ELM-based solver for PDEs by focusing on the optimal selection of the random-weight magnitude parameter R_m. Their approach, using a differential evolution strategy to minimize the residual norm, significantly boosts the accuracy of PIELMs—reducing solution errors by several orders of magnitude compared to naïvely initialized models. Notably, their PIELM formulation, though simple in architecture, outperforms even high-order FEM in selected benchmarks including Poisson, Burgers', and nonlinear Helmholtz equations, thus establishing a strong baseline for the class of PIELM methods. Supporting these findings, Fabiani et al. [13] applied PIELMs to bifurcation analysis in nonlinear PDEs, reporting that PIELM consistently outperforms finite differences (FD) and matches or exceeds FEM in both accuracy and computational cost, particularly for medium- to large-scale problems. An alternative acceleration strategy is the Extreme Theory of Functional Connections (X-TFC) [14], which analytically imposes boundary conditions via constrained expressions and trains the free parameters using ELM principles. This hybridization enables faster convergence and increased stability compared to gradient-based PINNs.

Calabrò et al. [15] extended PIELMs to handle PDEs with steep gradients, and Dong & Yang [16] demonstrated that proper scaling allows PIELMs to outperform traditional FEM methods on canonical smooth PDE benchmarks. Bayesian variants—namely BPIELMs—provide uncertainty quantification via Bayesian linear regression in the output layer [17]. Further extending ELM-based solvers to inverse problems, the method by Dong & Yang [16] applies local randomized neural networks and domain decomposition to accurately infer PDE parameters alongside the solution field. By leveraging non-gradient-based solvers like nonlinear least squares and variable projection methods, their approach achieves exponential convergence and outperforms conventional PINNs in both speed and accuracy, especially in the noise-free regime. To further improve expressiveness, Ni et al. [18] proposed the Hierarchical-Layer Concatenated Extreme Learning Machine (HLConcELM), which retains the ELM's fast training paradigm while enhancing accuracy through logically connected multi-layer hidden units. The model achieves exponential accuracy improvements across diverse PDE types—including advection, Helmholtz, KdV, and Schrödinger equations—without sacrificing computational efficiency. Dong and Li [19] further demonstrated that their locELM method, based on domain decomposition and local random-weight training, can outperform both PINNs and the Deep Galerkin Method (DGM) in accuracy and training time. Notably, locELM achieved accuracy levels on par with, or even surpassing, those of classical FEM on a range of benchmark problems, while maintaining much lower computational cost. A recent application-oriented study by Yan et al. [20] applied an PIELM framework to the structural analysis of composite thin-walled structures, successfully predicting deflection, vibration, and buckling modes with accuracy comparable to FEM. Their domain-decomposition approach, combined with ELM training, offers mesh-free efficiency and demonstrates that such hybrid solvers can handle real-world engineering problems beyond synthetic benchmarks. Dong and Yang [21] also explored an alternative to ELMs via a variable projection framework, combining Newton iteration and analytical elimination of output-layer parameters to achieve exponential convergence for both linear and nonlinear PDEs. While this approach sacrifices some speed compared to FEM, it delivers superior accuracy—outperforming both FEM and PINNs by orders of magnitude in certain regimes—thus complementing the strengths of ELM-based methods from earlier work.

Despite their impressive performance, many of the recent enhancements to physics-informed ELMs—such as domain decomposition, enforced analytical boundary constraints, or specialized architectures—introduce significant limitations when considering real-world deployment. In practical engineering scenarios, boundary conditions are rarely available in closed form, and any attempt to manually decompose complex geometries without meshing defeats the core appeal of mesh-free methods. Similarly, the use of regular collocation grids, as seen in prior work [8], is rarely feasible in realistic geometries. In this study, we therefore deliberately take a step back and focus on benchmarking the highly-performing yet simple formulation introduced by Dong and Yang [8], while also relaxing the regular-grid assumption to test its robustness under more realistic sampling conditions. Our aim is to systematically assess the true potential and generalisability of this minimalist PIELM variant against classical FEM across a curated suite of benchmark problems:

1) A 2D Poisson problem (based on [8]), tested with random collocation points.
2) A 2D heat conduction benchmark from our previous work [7], where standard PINNs previously underperformed FEM.
3) A hyperbolic PDE with sharp gradients, posing a classical challenge for FEM.
4) A 3D extension of the above heat problem to test effects of increased dimensionality.

Our study focuses on:

- Correlation between final training loss and solution accuracy.
- Comparative performance across problem types, collocation strategies, and dimensions.
- Detailed analysis of PIELM strengths (speed, accuracy) and limitations (boundary artifacts, sensitivity to non-smooth solutions).

Our contributions include:

- A large-scale empirical benchmark between PIELMs and FEM on four PDEs.
- Demonstration that training loss serves as a reliable a-posteriori accuracy indicator.
- Identification of regimes where PIELMs outperform (smooth PDEs) and underperform (sharp gradient & high dimensionality).
- Insight into typical failure modes such as localized misfits and plateauing behavior despite extensive hyper-parameter tuning.

For clarity, the present study focuses exclusively on the head–to–head benchmarking between the minimalist single–R_m PIELM and FEM. A detailed comparison between conventional PINNs and FEM was already provided in our previous work [7], which identified FEM as the relevant classical baseline. Repeating that analysis here would not contribute additional insight but would obscure the specific contribution of the single–R_m formulation evaluated in this paper. The objective of this work is not to perform a classical FEM convergence study, but to compare end-to-end solver performance under realistic usage conditions. Accordingly, FEM is treated as a mature reference method, while PIELM is analysed through a broad exploration of its admissible parameter space. The reported results should therefore be interpreted as achievable accuracy-cost envelopes rather than as optimal solver configurations for either method.

The remainder of this paper is structured as follows: Section II describes mathematical framework; Section III presents the benchmark problems; Section IV shows the results; Section V discusses them; and Section VI contains concluding remarks and future research directions.

II. NUMERICAL METHODS

This section introduces the PIELM collocation solver used throughout the paper, the Monte-Carlo strategy adopted for hyper-parameter exploration, and the finite-element (FEM) reference solutions against which PIELM is benchmarked.

A. EXTREME LEARNING MACHINE (SINGLE-R_M FORMULATION)

Following Dong & Yang's optimal–R_m recipe [8], we approximate the solution $u : \Omega \to \mathbb{R}$ of a linear elliptic PDE

$$\mathcal{L}[u](\mathbf{x}) = f(\mathbf{x}), \qquad \mathcal{B}[u](\mathbf{x}) = g(\mathbf{x}), \quad \mathbf{x} \in \Omega \subset \mathbb{R}^d, \tag{1}$$

with a single-hidden-layer network

$$u_\beta(\mathbf{x}) = \sum_{j=1}^{M} \beta_j \, \sigma\big(R_m \, \mathbf{w}_j \cdot \mathbf{x} + b_j\big), \qquad \mathbf{w}_j, b_j \sim \mathcal{U}[-1, 1], \tag{2}$$

where

- $\mathcal{L}[u]$ - Differential operator defining the PDE
- $f(\mathbf{x})$ - Driving term of the PDE
- $\mathcal{B}[u]$ - Differential operator defining the boundary conditions
- $g(\mathbf{x})$ - Boundary condition function
- M – number of hidden nodes,
- σ – activation (Gaussian by default),
- R_m – single scaling parameter controlling the support of σ,
- β_j – trainable output weights collected in $\boldsymbol{\beta} \in \mathbb{R}^M$.

Unless otherwise stated, the activation function σ is Gaussian. This choice follows Dong and Yang [8], who observed only marginal accuracy differences among smooth activations such as Gaussian and $\tanh$ for elliptic PDEs, while the Gaussian offers superior stability under repeated differentiation. Adopting it here ensures consistency with that reference and facilitates direct comparison.

Because (2) is linear in β, the training stage reduces to a single least-squares solve:

$$\underbrace{\begin{bmatrix} \mathcal{L}V_1(\mathbf{x}_1) & \dots & \mathcal{L}V_M(\mathbf{x}_1) \\ \vdots & & \vdots \\ \mathcal{L}V_1(\mathbf{x}_Q) & \dots & \mathcal{L}V_M(\mathbf{x}_Q) \\ \hline \mathcal{B}V_1(\mathbf{x}_{Q+1}) & \dots & \mathcal{B}V_M(\mathbf{x}_{Q+1}) \\ \vdots & & \vdots \\ \mathcal{B}V_1(\mathbf{x}_{Q+Q_b}) & \dots & \mathcal{B}V_M(\mathbf{x}_{Q+Q_b}) \end{bmatrix}}_{\mathbf{A} \in \mathbb{R}^{(Q+Q_b)\times M}} \boldsymbol{\beta} = \underbrace{\begin{bmatrix} f(\mathbf{x}_1) \\ \vdots \\ f(\mathbf{x}_Q) \\ \hline g(\mathbf{x}_{Q+1}) \\ \vdots \\ g(\mathbf{x}_{Q+Q_b}) \end{bmatrix}}_{\mathbf{f}}, \tag{3}$$

where

- $\{\mathbf{x}_p\}_{p=1}^{Q}$ – **interior** collocation points drawn uniformly at random inside Ω,
- $\{\mathbf{x}_q\}_{q=Q+1}^{Q+Q_b}$ – **boundary** points obtained by rejection sampling on $\partial\Omega$,
- $V_j(\mathbf{x}) = \sigma\big(R_m \, \mathbf{w}_j \cdot \mathbf{x} + b_j\big)$ are the basis functions.

a: Training loss and error metrics.

The train loss is the 2-norm residual of (3), $\mathcal{L}_{\text{train}} = \|\mathbf{A}\boldsymbol{\beta} - \mathbf{f}\|_2^2$, whereas accuracy is evaluated on an independent test set $\{\tilde{\mathbf{x}}_k\}_{k=1}^{N}$ as the root-mean-square (RMS) error

$$\varepsilon_{\text{RMS}} = \sqrt{\frac{1}{N} \sum_{k=1}^{N} \big[u_\beta(\tilde{\mathbf{x}}_k) - u_{\text{exact}}(\tilde{\mathbf{x}}_k)\big]^2}. \tag{4}$$

It is implicitly assumed here, that the exact solution $u_{\text{exact}}(\mathbf{x}_k)$ for the considered benchmark is known in its analytical form.

b: Compile versus solve time.

We distinguish (i) compile time—automatic differentiation and assembly of $\mathbf{A}$—from (ii) solve time, i.e. the QR factorisation of (3). Unless stated otherwise, wall-clock times in Section IV refer to the sum of both stages.

B. HYPER-PARAMETER EXPLORATION

PIELM performance hinges on four user choices:

1) width of the hidden layer M or number of parameters (10^2–10^3),
2) scale R_m of the hidden weights (10^0–10^1),
3) size of the inner domain discretization n_{dom},
4) size of the boundary discretization n_{bnd}.

Because their interplay is highly problem-dependent, we adopt a **Monte-Carlo sweep**: for each benchmark we draw 1024 random quadruples (M,R_m,n_{dom},n_{bnd}) from log-uniform priors and record $\mathcal{L}_{\text{train}}, \varepsilon_{\text{RMS}}$ and wall times. The resulting data cloud underpins the loss–accuracy correlation reported later.

To isolate the effect of single parameters we optionally perform refined sweeps: fix the hyper-parameters for which an optimal value can be envisaged out of the points cloud and scan the remaining one(s) on a reduced Monte-Carlo run (128 parameters sets).

The full Monte–Carlo exploration of 1024 realisations per benchmark corresponds to a total CPU time of roughly $3-5$ h on a single core. This broad search was intended to characterise the overall sensitivity of the method in a "green–field" manner. In practical deployments, once suitable defaults are known, only a limited number of trial runs is needed–typically by transferring experience from similar problems or by coarse sampling followed by local refinement. This type of calibration effort is comparable to mesh-parameter tuning in FEM and should be regarded as part of the normal setup cost.

C. REFERENCE FINITE-ELEMENT SOLUTIONS

Reference solutions are generated with FEniCS [22]:

- conforming linear (P_1) elements on triangular/tetrahedral meshes created by Gmsh [23];
- Dirichlet and Neumann boundary conditions enforced weakly;
- linear systems solved with conjugate-gradient preconditioned by algebraic multigrid (`hypre_boomerAMG`).

For each benchmark we refine the mesh until the element count exceeds the memory capacity of our device (16 GB). This choice is not intended as a substitute for a convergence-driven mesh refinement study, but to approximate the upper envelope of FEM accuracy and computational cost under practical constraints. Mesh size is therefore used here as a proxy for computational cost rather than as a convergence parameter. Wall-clock times are measured with the same timer routine used for PIELM, ensuring fair comparison. The reported FEM wall-times comprise both the mesh generation in Gmsh and the subsequent assembly and linear solve in FEniCS. For the two–dimensional cases the meshing stage accounts for 85–95 % of the total wall-time, whereas in the largest 3-D benchmark the solution stage dominates with up to 83 %. This ensures that the comparison with PIELM reflects the complete end-to-end cost of each method. The FEM implementation is intended to provide a robust reference solution rather than to explore solver-specific optimisations. The solver tolerance was chosen sufficiently strict so that, within the accuracy regime considered here, the linear solver error does not materially affect the reported FEM results. Accordingly, the observed FEM errors primarily reflect discretisation effects rather than solver artefacts. A single solver strategy was adopted for all benchmarks in order to maintain a consistent FEM workflow across two- and three-dimensional problems. While direct solvers may be effective for selected two-dimensional cases, solver choice is not the focus of this work, and per-benchmark solver optimisation is intentionally avoided.

D. SOFTWARE AND HARDWARE

All PIELM results were produced in `Jax 0.6.0` with double-precision and JIT compilation enabled; least-squares solves use `jax.numpy.linalg.lstsq` (QR with column pivoting). Computations ran on an Intel i7-1165G7 @ 2.8 GHz; no hardware accelerators were used, so timings correspond to single-CPU performance.

III. BENCHMARK PROBLEMS

To assess the accuracy, robustness, and computational cost of the single–R_m PIELM solver, we consider four elliptic test–cases of increasing geometric and analytical complexity. All of them admit closed–form solutions, which enables an exact evaluation of the RMS error (4). Table 1 summarises the key features; detailed formulations follow.

TABLE 1. Overview of benchmark problems.

Problem	Dimension d	Geometry
Poisson verification	2	$[0,2]^2$ square
Original (heat conduction)	2	square with circular hole
Hyperbolic singularity	2	same as above
Original, 3-D extension	3	cube with spherical hole

All selected benchmarks admit analytic ground truths, ensuring exact error quantification and a level comparison between methods. Future work will extend this analysis to non-convex and singular problems (e.g., crack-tip fields) where the behaviour of PIELMs relative to FEM remains to be established.

A. POISSON VERIFICATION CASE

The first benchmark reproduces the 2-D Poisson problem of Dong & Yang [8], serving as a sanity check for our PIELM implementation:

$$\begin{aligned} -\nabla^2 u &= f(\mathbf{x}) \quad \text{in } \Omega = [0,2]^2, \\ u &= g(\mathbf{x}) \quad \text{on } \partial\Omega, \end{aligned} \tag{5}$$

with the analytic solution

$$u_{\text{exact}}(x,y) = \Big[-2\cos\big(\tfrac{3}{2}\pi x + \tfrac{2}{5}\pi\big) - \tfrac{3}{2}\cos\big(3\pi x - \tfrac{1}{5}\pi\big)\Big] \cdot \Big[-2\cos\big(\tfrac{3}{2}\pi y + \tfrac{2}{5}\pi\big) - \tfrac{3}{2}\cos\big(3\pi y - \tfrac{1}{5}\pi\big)\Big], \tag{6}$$

so that $f = -\nabla^2 u_{\text{exact}}$ and $g = u_{\text{exact}}$. Because the boundary data are smooth and the domain is convex, the solution enjoys full H^2 regularity, allowing us to isolate discretisation errors from singular behaviour.

a: Purpose.

This test confirms that random interior sampling attains the same accuracy as the uniform tensor grid of [8] while avoiding the restrictive requirement of a Cartesian domain.

B. ORIGINAL BENCHMARK (2-D)

The "Original" problem, introduced in our earlier PINN–FEM comparison [7], models steady heat conduction on a square plate of side b with a concentric circular cavity of radius a:

$$-\nabla^2 T = 4 - \frac{2a}{r}, r = \sqrt{x^2+y^2}, \quad \text{in } \Omega, \tag{7a}$$

$$T = 0 \quad \text{on } \Gamma_{\text{hole}}\ (r=a), \tag{7b}$$

$$T = (r-a)^2 \quad \text{on } \Gamma_{\text{top/bot}}\ (|y| = b/2), \tag{7c}$$

$$\nabla T \cdot \mathbf{n} = b(1-a/r) \quad \text{on } \Gamma_{\text{left/right}}\ (|x| = b/2), \tag{7d}$$

where $\Omega = [-\frac{b}{2}, \frac{b}{2}]^2 \setminus \{r < a\}$ is the domain and $\mathbf{n}$ denotes the outward normal. The exact solution

$$T_{\text{exact}}(r) = (r-a)^2 \tag{8}$$

is C^1 but exhibits a jump in the radial derivative at $r = a$.

a: Purpose.

Compared with (5), the curved internal boundary probes the PIELM's ability to handle irregular collocation sets and provides a direct PIELM–vs–PINN accuracy baseline.

C. HYPERBOLIC DIVERGENCE PROBLEM

To stress both PIELM and FEM with a steeper gradient we retain the geometry of (7) but prescribe a hyperbolic profile:

$$-\nabla^2 T = 0, \quad \text{in } \Omega, \tag{9a}$$

$$T = \frac{x+y}{r^2}, r = \sqrt{x^2+y^2}, \quad \text{on } (r=a, |y| = b/2), \tag{9b}$$

$$\nabla T \cdot \mathbf{n} = \operatorname{sgn}(x)\frac{-x^2 - 2xy + y^2}{(x^2+y^2)^2}, \quad \text{on } (|x| = b/2), \tag{9c}$$

The exact solution $T_{\text{exact}} = \frac{x+y}{r^2}$ features a nil-Laplacian but an arbitrarily large gradient, lowering boundary regularity. This is known to challenge low–order FEM [24] and therefore provide a useful benchmark for machine-learning solvers.

D. ORIGINAL BENCHMARK IN THREE DIMENSIONS

Finally, we lift (7) to $\mathbb{R}^3$:

$$-\nabla^2 T = 6 - \frac{4a}{r}, \quad r = \sqrt{x^2+y^2+z^2}, \quad \text{in } \Omega_3, \tag{10a}$$

$$T = (r-a)^2 \quad \text{on } \Gamma_{\text{cube}}\ (x,y,z = \pm b/2), \tag{10b}$$

$$\nabla T \cdot \mathbf{n} = 0 \quad \text{on } \Gamma_{\text{sphere}}\ (r=a). \tag{10c}$$

with $\Omega_3 = [-\frac{b}{2}, \frac{b}{2}]^3 \setminus \{r < a\}$. The analytic solution remains $T_{\text{exact}} = (r-a)^2$. Differently from the 2D version, we adopt Dirichlet conditions on the entirety of the exterior boundary and von Neumann on the interior. Apart from this slightly simplifying adjustment relative to Section III-B, the principal difficulty now lies in the curse of dimensionality for both mesh-based (FEM) and collocation-based (PIELM) approaches.

a: Purpose.

This case gauges how collocation density must scale with dimension and investigates whether the compile cost of PIELM grows faster or slower than matrix assembly in FEM.

IV. RESULTS

This section aggregates more than 4×1024 PIELM realisations and a comprehensive set of FEM reference runs. All wall-times were measured on the same Intel i7-1165G7 CPU and include the full compile+solve pipeline as defined in Section II-A.

A. TRAIN LOSS VERSUS ACCURACY

Fig. 1 compiles every Monte-Carlo sample across the four benchmarks. A general trend towards a power-law bordering on linearity relating these two indicators emerges, independently of the chosen benchmark. It is nonetheless worth mentioning that deviations from this general trend are indeed apparent in these data, particularly for the Original-3D

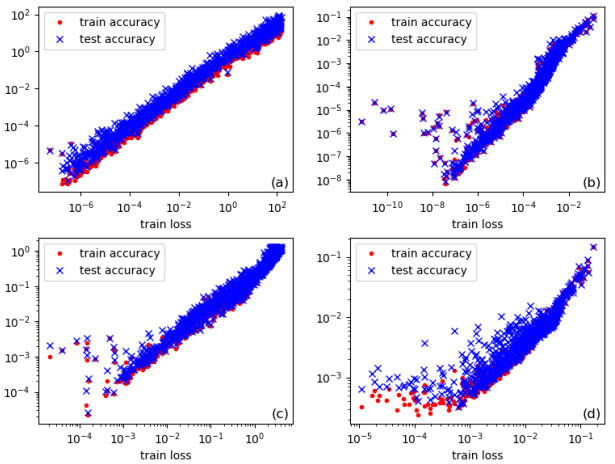

FIGURE 1. Global correlation between training loss and accuracy. Both training and test accuracies are shown. (a) Poisson, (b) Original 2D, (c) Hyperbolic, (d) Original 3D

benchmark. Almost all outliers lie above the diagonal and pertain to the low-loss range.

B. PERPROBLEM PERFORMANCE

The scatter plots in Figs. 2–5 highlight the results of the Monte-Carlo cloud for each benchmark problem; numeric optima for the best accuracy and the fastest wall time are summarised in Tables 2 and 3.

Problem	Method	RMS error	Times [s]
Poisson 2-D	PIELM	1.09×10^{-7}	2.33
	FEM	3.22×10^{-5}	189
Original 2-D	PIELM	8.37×10^{-9}	8.30
	FEM	3.94×10^{-5}	24.0
Hyperbolic 2-D	PIELM	2.57×10^{-5}	10.3
	FEM	3.01×10^{-4}	149
Original 3-D	PIELM	3.28×10^{-4}	36.3
	FEM	8.26×10^{-5}	135

TABLE 2. Best observed RMS error and corresponding full wall-time (compile + solve).

Problem	Method	RMS error	Times [s]
Poisson 2-D	PIELM	1.91×10^{-4}	1.69
	FEM	3.12×10^{-3}	0.68
Original 2-D	PIELM	3.84×10^{-3}	2.89
	FEM	3.39×10^{-4}	0.200
Hyperbolic 2-D	PIELM	1.10×10^{-1}	2.61
	FEM	3.02×10^{-3}	0.641
Original 3-D	PIELM	8.94×10^{-3}	3.32
	FEM	6.03×10^{-3}	0.0455

TABLE 3. Best observed full wall-time and corresponding RMS errors (compile + solve).

For each benchmark, we show the dependence of train and test accuracy for PIELM as a function of the hyperparameters, namely: the hidden weight scale R_m, the number of trainable parameters (width of the last layer), the fraction of points on the boundary, and the total number of collocation points. For the latter, we additionally display the dependence of the train and test losses and compare with the corresponding FEM simulation. The final main comparison between PIELM and FEM is shown in the dependence of the computational times upon the achieved accuracy.

a: Summary of the sweeps.

- **Poisson.** Random collocation achieves the same 10^{-6} RMS accuracy as the uniform grid used by Dong et al. while outperforming FEM in terms of wall-time for any accuracy below $\approx 10^{-3}$ (Fig. 2).
- **Original 2-D.** The best PIELM is six orders of magnitude more accurate than our earlier PINN but faster than FEM at equal error below $\sim 10^4$ (Fig. 3).
- **Hyperbolic.** PIELM can reach RMS error about one order of magnitude smaller compared with FEM, while keeping the wall-time 10 times below (Fig. 4).
- **Original 3-D.** PIELM fails to outperform FEM below $\approx 3 \times 10^{-4}$ accuracy but this higher performance comes at a cost in terms of wall-time of almost one order of magnitude (Fig. 5).

PIELM is characterized by a larger number of hyperparameters while FEM performance is basically governed by the size of the mesh with the remaining features deriving directly from it. For most PIELM hyperparameters, it is possible to identify an optimal value that minimizes the RMS error at least approximately or alternatively an overall trend. In some cases, this was not possible, specifically for the dependence on the total number of points in the Poisson and Original-2D benchmarks. In order to identify optimal values for this hyperparameter as well, a smaller (128 points) Monte-Carlo run was performed with the remaining hyperparameters fixed at their optimum. This is the run that is shown in the corresponding panels (e).

C. SOLUTION QUALITY: OPTIMAL PIELM PREDICTIONS

Fig. 6 visualises the best PIELM solution for each benchmark (top row of panels) alongside its pointwise deviation from the analytic ground truth (bottom row). Errors concentrate around the corners and the boundary regions.

D. SOLUTION QUALITY: OPTIMAL FEM PREDICTIONS

For completeness, Fig. 7 shows the corresponding FEM results on the finest meshes used. FEM errors seem to have a smoother structure, loosely following the shape of the solution and characterised by less extreme values than PIELM.

V. DISCUSSION

The dataset compiled in Section IV clarifies the three motivating questions posed in the Introduction and yields practical guidelines for future deployments of random-feature solvers.

A. TRAINING LOSS AS A RELIABLE A-POSTERIORI INDICATOR

Fig. 1 reveals an almost linear relationship in log–log space between the collocation residual $\mathcal{L}_{\text{train}}$ and the RMS error ε_{RMS} for all four benchmarks. Although the exact slope varies slightly from case to case, the band is narrow enough that a single trend line bounds the vast majority of runs. This finding suggests that $\mathcal{L}_{\text{train}}$ serves as a reliable *a-posteriori* error indicator. To the best of our knowledge, this is a novel finding for any type of PINNs and it would be definitely worth to investigate for other types of solvers because having a reliable predictor of how well the obtained solution approximates the true one is certainly a very useful feature. The practical consequence is that one can stop training as soon as the residual reaches a user-defined threshold without evaluating a separate test set—an attractive feature in industrial scenarios lacking ground-truth data.

The fact that in all outliers the loss is higher than the accuracy and both are in the low-values range suggests that this effect is brought about by a form of overfitting with the obtained solution displaying a much lower training loss than

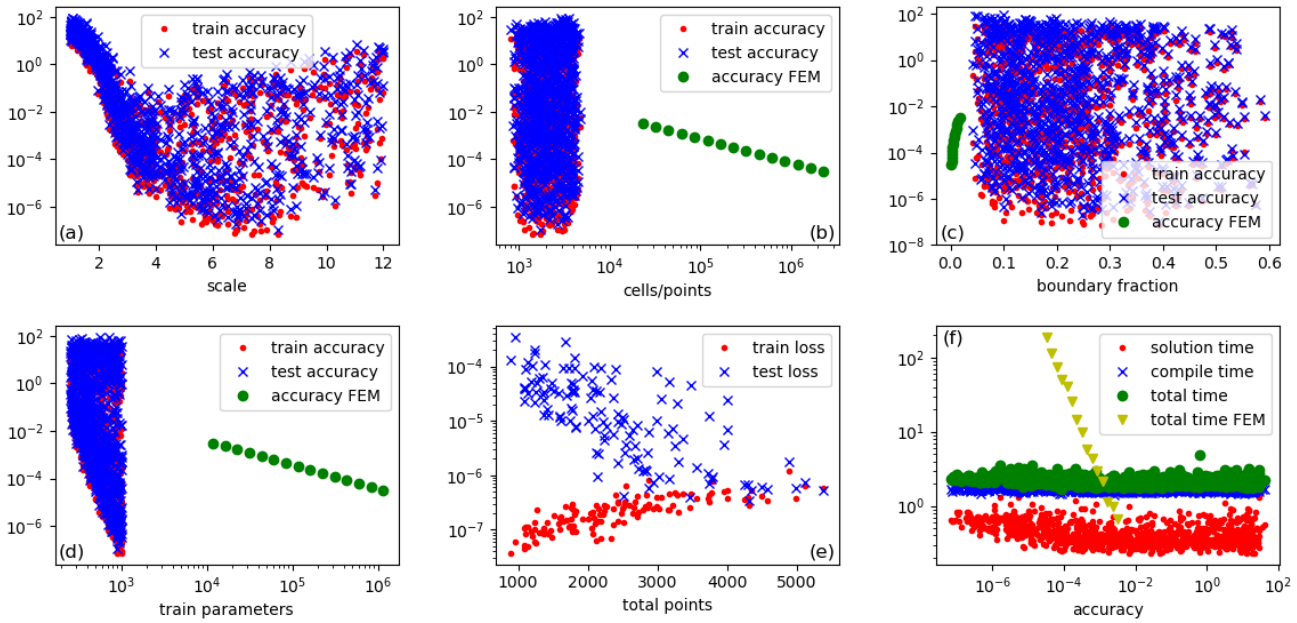

FIGURE 2. Hyperparameter sweep for the Poisson problem with dependence of the train and test accuracy (RMS errors) upon the scale parameter R_m (a), then total number of collocation points for PIELM and of cells for FEM (b), the fraction of cells/points on the boundary (c), and the number of trainable parameters (d). Panel (e) shows the train and test losses as a function of the total number of collocation points. In panel (f), the computational times as a function of the obtained accuracy are shown.

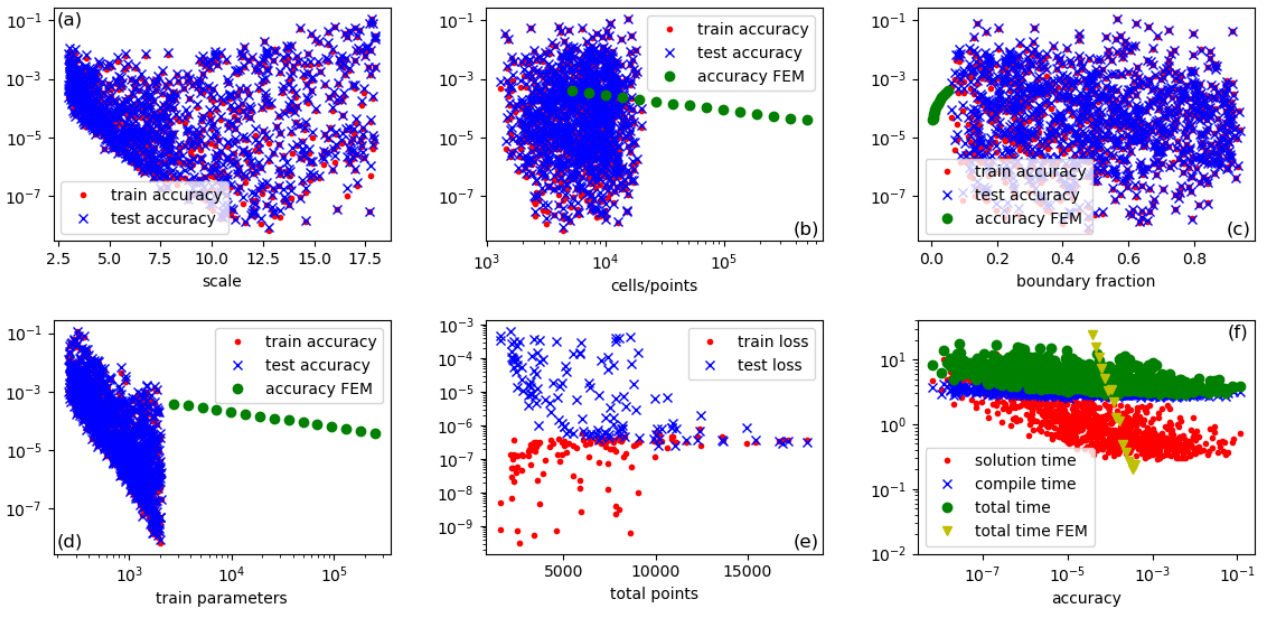

FIGURE 3. Hyperparameter sweep for the Original 2D problem with dependence of the train and test accuracy (RMS errors) upon the scale parameter R_m (a), then total number of collocation points for PIELM and of cells for FEM (b), the fraction of cells/points on the boundary (c), and the number of trainable parameters (d). Panel (e) shows the train and test losses as a function of the total number of collocation points. In panel (f), the computational times as a function of the obtained accuracy are shown.

the actual deviation from the real solution. This aspect must be further investigated before the training loss is uncritically considered as an exact predictor of the final accuracy.

A brief sensitivity check was conducted to quantify the effect of random weight initialisation. Sixty-four repeated runs at fixed hyperparameters–chosen from the best perform-

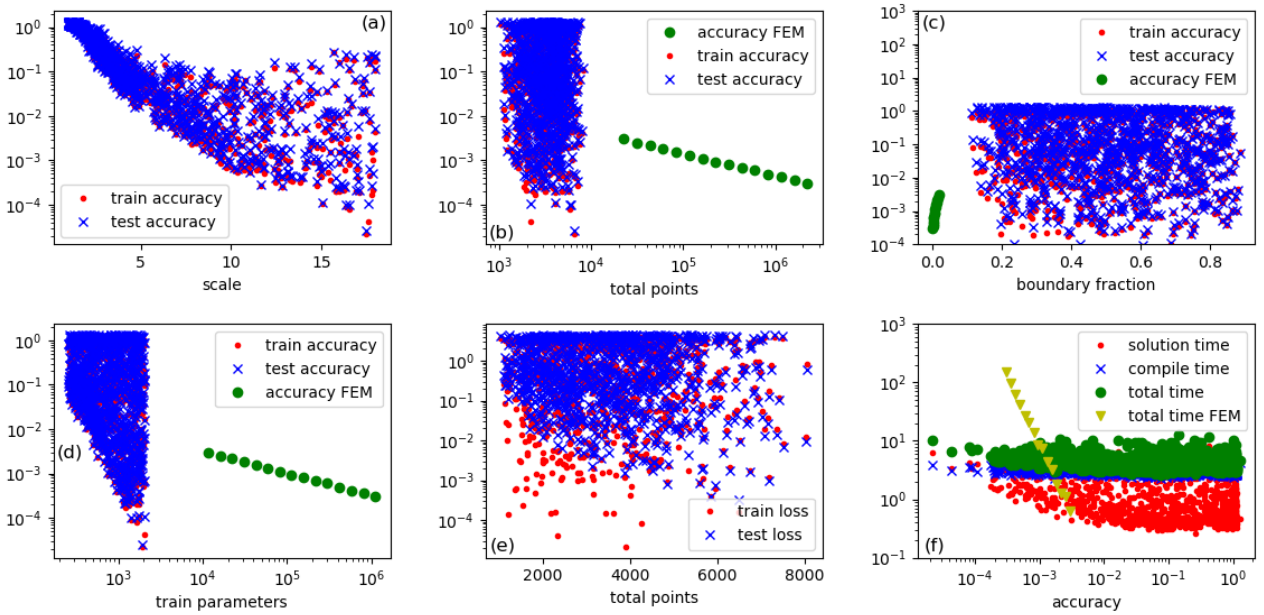

FIGURE 4. Hyperparameter sweep for the Hyperbolic problem with dependence of the train and test accuracy (RMS errors) upon the scale parameter R_m (a), then total number of collocation points for PIELM and of cells for FEM (b), the fraction of cells/points on the boundary (c), and the number of trainable parameters (d). Panel (e) shows the train and test losses as a function of the total number of collocation points. In panel (f), the computational times as a function of the obtained accuracy are shown.

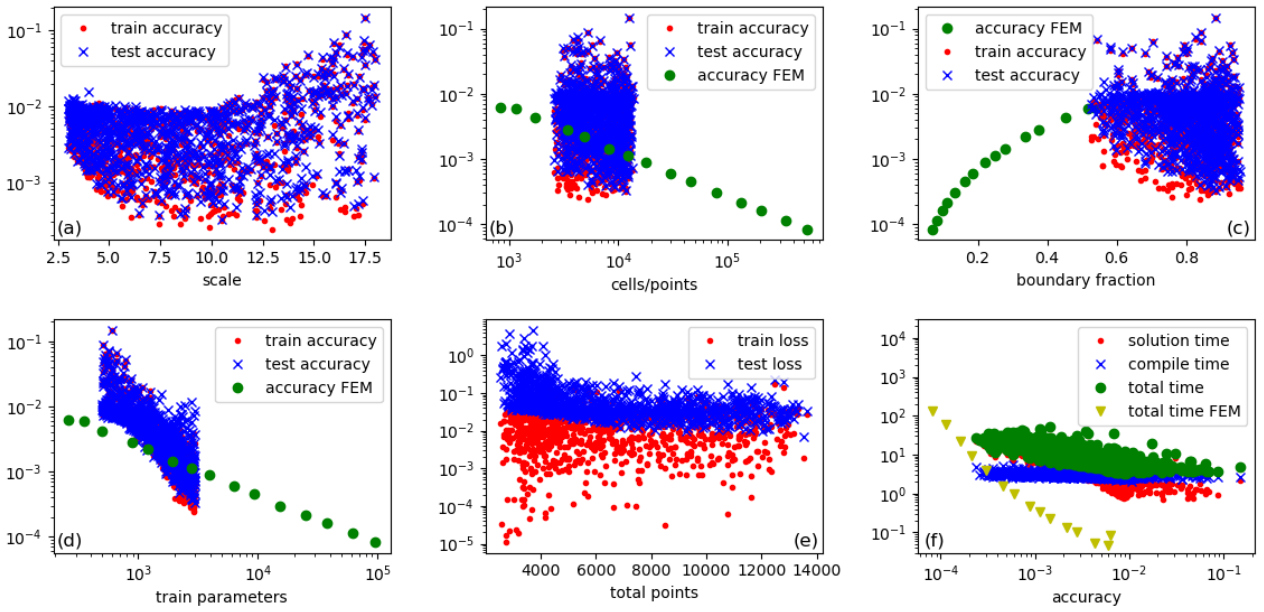

FIGURE 5. Hyperparameter sweep for the Original 3D problem with dependence of the train and test accuracy (RMS errors) upon the scale parameter R_m (a), then total number of collocation points for PIELM and of cells for FEM (b), the fraction of cells/points on the boundary (c), and the number of trainable parameters (d). Panel (e) shows the train and test losses as a function of the total number of collocation points. In panel (f), the computational times as a function of the obtained accuracy are shown.

ing configurations per problem–showed an RMS-error spread comparable to the lowest overall error of each benchmark. Therefore, the variance observed in Figs. 2-5 arises primarily from hyperparameter variation rather than from stochastic weight initialisation. This confirms that the inherently random component of PIELM does not constitute a dominant

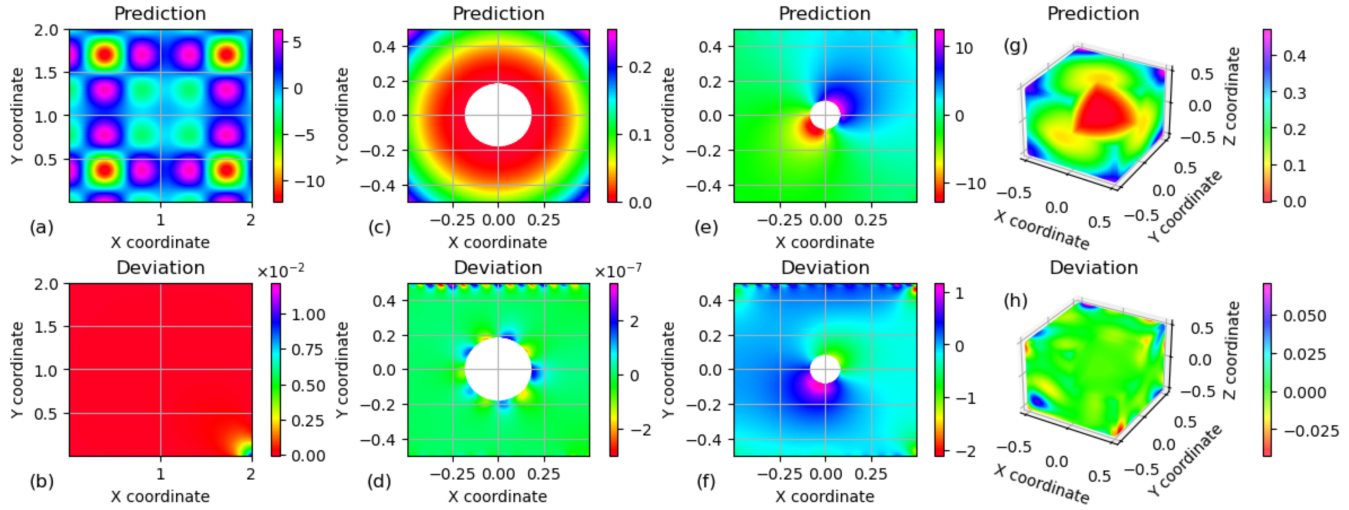

FIGURE 6. Optimal PIELM predictions (top) and error w.r.t. ground truth (bottom) for all four benchmarks: Poisson (panels a and b), Original 2D (panels (c and d), Hyperbolic (panels e and f), Original 3D (panels g and h).

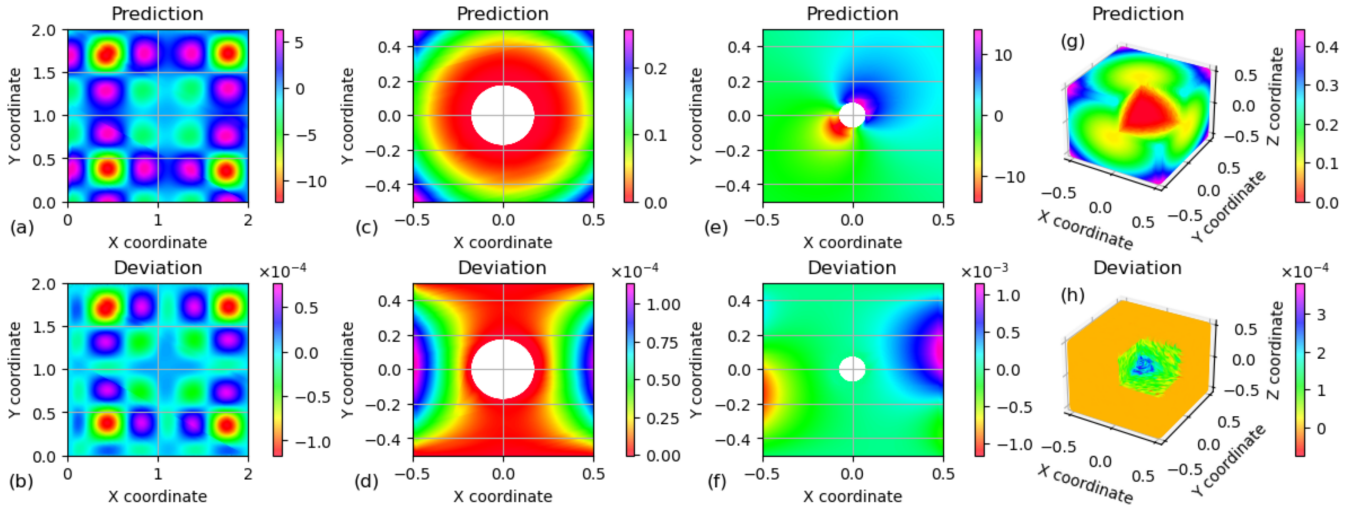

FIGURE 7. Optimal FEM predictions (top) and error w.r.t. ground truth (bottom) for all four benchmarks: Poisson (panels a and b), Original 2D (panels (c and d), Hyperbolic (panels e and f), Original 3D (panels g and h).

uncertainty source.

B. WHEN DOES PIELM OUTPERFORM FEM?

Table 2, Table 3, and the parameter clouds in Figs. 2–5 show a mixed picture. PIELM appears to outperform FEM both in terms of accuracy and wall-time in at least 3 of the benchmarks, while holding its ground in the fourth where FEM's better accuracy rapidly results in a significantly higher computational cost. It is worth mentioning that for the Original 2D benchmark we observe a massive improvement with respect to our previous work focused on conventional PINNs [7]: accuracy could be improved by as much as 4 orders of magnitude while wall time on the same device remains at least a factor 10 lower.

In general, FEM displays a steeper correlation between wall-time and accuracy whereas it seems that PIELM can achieve higher accuracy without significant increase in computational costs. As already mentioned, PIELMs are characterized by a larger number of hyperparameters, whereas the mesh size is the main driver in FEM. This results, among other things, in a very low range of ratios between interior and exterior cells at least forthe 2-dimensional cases, whereas in PIELM this parameter can be virtually varied at will. This is one additional advantage of mesh-less techniques where the collocation points can be chosen independently from one another. On the other hand, the large number of hyperparameters can be also seen as a weakness in PIELM because it means that optimisaztion can be more tedious and time-consuming. It is therefore desirable that the algorithms for selecting the collocation points are improved to a maturity level similar to that of meshing techniques where one parameter is enough to effectively define the resulting

mesh. This is particularly true in view of the fact that the compiling time, consisting of collocation points selection and matrix construction, dominates the overall wall-time in PIELM in a similar fashion as meshing-time does in FEM. Care must be thus taken in avoiding unnecessary iterations of this step for instance by means of exploiting the fact that single collocation points can be adjusted or added without affecting the remaining ones thanks to the fact that, unlike mesh, they are not directly correlated.

Solution plots (Figs. 6, 7) indicate that FEM errors are smoother and less localised, whereas PIELM errors spike at specific locations. This suggests that a certain degree of overfitting might be happening but, apart from that, it is an effect that deserves further investigation before more complex applications are tackled.

The local error spikes visible in Fig. 6 are reproducible across random weight initialisations and stem from sparse collocation near boundaries or re-entrant corners. They contribute little to the global RMS error but can be efficiently reduced by locally increasing the collocation density–an analogue to mesh refinement in FEM that does not require remeshing.

Although PIELMs lack the formal variational guarantees of FEM, they remain partially interpretable: the predicted field is an explicit linear combination of known basis functions, and the collocation residual directly measures physical consistency. Rather than replacing FEM, PIELMs complement classical methods by offering mesh-free solutions for cases where meshing is prohibitive or frequent re-meshing is required. We view them as a pragmatic bridge between data-driven and first-principles solvers whose theoretical foundations are still evolving.

The above is an analysis of accuracy, computational cost, and qualitative error behaviour of FEM and PIELM across a set of representative elliptic benchmark problems. While the results are necessarily problem-dependent, several recurring patterns emerge. To improve readability and to synthesise these observations, Table 4 summarises the qualitative characteristics of the two approaches as observed in the present benchmarks. The table is intended as a descriptive summary rather than as a set of general recommendations. The comparison highlights that the two approaches address complementary aspects of numerical solution workflows. FEM benefits from strong theoretical foundations, mature tooling, and well-established mechanisms for local error control, particularly in the presence of singularities. In contrast, PIELM offers a mesh-free formulation with low setup effort and competitive accuracy for smooth problems, at the cost of reduced local error control. Importantly, these observations are limited to the benchmark problems considered here. Broader or prescriptive conclusions would require substantially wider validation across different PDE classes, geometries, and boundary conditions.

TABLE 4. Qualitative comparison based on observations in the present benchmarks

Aspect	FEM	PIELM
Meshing requirement	Required	Not required
Local accuracy control	Strong	Limited
Behaviour near singularities	Robust	Local deviations
Global accuracy (smooth PDEs)	High	High
Setup complexity	Moderate	Low
Method maturity	Established	Emerging

VI. CONCLUSIONS

In this study we revisited the minimalist, single–R_m formulation of physics-informed extreme learning machines (PIELMs) and carried out the first large-scale, head-to-head comparison with classical finite-element methods (FEM) on four elliptic benchmarks of increasing geometric and analytical complexity without the limitation of regular grids for the collocation points. More than 4×10^3 random hyper-parameter realisations were executed and contrasted with optimally refined FEM solutions, allowing us to draw the following, data-driven conclusions.

1) Accuracy and speed trade-off. For the two smooth, two-dimensional problems (Poisson verification and the *Original 2-D* heat benchmark) PIELM achieved sub-10^{-7} and sub-10^{-8} RMS errors, respectively, while remaining at least one order of magnitude faster than FEM at comparable accuracy. On the hyperbolic test with steep gradients PIELM preserved its speed advantage and even reduced the FEM error by an order of magnitude, although the gap narrowed when targeting very low accuracies. In three dimensions PIELM could not match the best FEM accuracy yet still delivered competitive results at a fraction of the wall-time, confirming favourable accuracy–per-second scaling but also revealing a dimensionality limit for the present, non-adaptive formulation.

2) Training loss as an *a-posteriori* error indicator. Across all benchmarks the collocation residual L_{train} and the RMS error ε_{RMS} followed an almost linear trend in log–log space spanning seven decades. This empirical law turns L_{train} into a practical stopping criterion when reference solutions are unavailable, although a handful of low-loss outliers warns that overfitting may still occur, especially in the 3-D case.

3) Hyper-parameter sensitivity. Out of the four free parameters $(M, R_m, n_{\text{dom}}, n_{\text{bnd}})$ the width M and the weight scale R_m dominated both accuracy and compile cost, whereas the interior–to–boundary ratio mostly controlled stability. Reasonable defaults emerged from the sweeps, suggesting that extensive tuning can often be avoided once a coarse Monte-Carlo search is performed.

4) Typical failure modes. Error visualisations revealed that PIELM inaccuracies are localised around re-entrant corners, (exterior) boundaries and sharp fronts, while FEM errors remain smoother but accumulate globally when the mesh is under-refined. These complementary patterns suggest that PIELM's overall better accuracy might come at a price in terms of fidelity of reproduction of the local features.

5) Practical guidelines and outlook.

- Use the training residual as the primary convergence metric; stop once a certain threshold for L_{train} is achieved.
- Allocate a suitable fraction of collocation points to the boundary to avoid spurious artifacts.
- Prefer moderate widths and perform a short grid search over R_m; further scaling returns diminishing gains while inflating compile time.
- Exploit the embarrassingly parallel nature of the Monte-Carlo sweep—unlike gradient-based PINNs, each realisation is an independent, one-shot solve.

Future work should therefore (i) incorporate adaptive or variance-based point selection to mitigate the curse of dimensionality, (ii) investigate GPU-accelerated QR solvers to shorten compile stages, and (iii) explore strategies to optimize the hyperparmeters in a computationally economic way. Extending the present study to time-dependent and non-elliptic PDEs, as well as to noisy or partially observed data, will further clarify the domain of applicability of physics-informed extreme learning machines.

Overall, the evidence compiled herein positions the single–R_m PIELM as a compelling alternative to both FEM and gradient-based PINNs for fast, high-accuracy solutions of low- to moderate-dimensional, smooth PDEs, while also charting the roadmap for overcoming its current limitations in high-dimensional and highly non-smooth regimes.

ACKNOWLEDGMENT

The author wishes to thank Urs-Martin Küenzi for invaluable technical contribution, insightful comments, and fruitful discussions. ChatGPT 5.2 has been employed to check and improve the language, the logic, and the structure of this manuscript.

REFERENCES

[1] K. S. Lee and I. S. Kang, "Neural Algorithms for Solving Differential Equations," *J. Comp. Phys.*, vol. 91, pp. 110–131, Dez. 1990, https://doi.org/10.1016/0021-9991(90)90007-N.

[2] I. E. Lagaris, A. Likas and D. I. Fotiadis, "Artificial neural networks for solving ordinary and partial differential equations," *IEEE Trans. Neural Netw.*, vol. 9(5), pp. 987-–1000, Sep. 1998, https://doi.org/10.1109/72.712178.

[3] 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," *J. Comput. Phys.*, vol. 378, pp. 686-–707 Feb. 2019, https://doi.org/10.1016/j.jcp.2018.10.045.

[4] G. E. Karniadakis et al., "Physics-informed machine learning," *Nat. Rev. Phys.*, vol. 3, pp. 422-–440, May 2021, https://doi.org/10.1038/s42254-021-00314-5.

[5] S. Wang, Y. Teng and P. Perdikaris, "Understanding and mitigating gradient flow pathologies in physics-informed neural networks," *SIAM J. Sci. Comput.*, vol. 43(5), pp. A3055-–A3081, 2021, https://doi.org/10.1137/20M1318043.

[6] S. Mishra and R. Molinaro, "Estimates of generalization error for physics-informed neural networks for the Burgers equation," *IMA J. Num. Anal.*, vol. 42(2), pp. 981–1022, Apr. 2021, https://doi.org/10.1093/imanum/drab032.

[7] A. Sacchetti et al., "Neural Networks to Solve Partial Differential Equations: A Comparison With Finite Elements," *IEEE Acc.*, vol. 10, pp. 32271–32279, Mar. 2022, https://doi.org/10.1109/ACCESS.2022.3160186.

[8] S. Dong and J. Yang, "On Computing the Hyperparameter of Extreme Learning Machines: Algorithm and Application to Computational PDEs, and Comparison with Classical and High-Order Finite Elements," *Comput. Methods Appl. Mech. Eng.*, vol. 463, 111290, Aug. 2022, https://doi.org/10.1016/j.jcp.2022.111290.

[9] G.-B. Huang, Q.-Y. Zhu and C.-K. Siew, "Extreme learning machine: Theory and applications," *Neurocomputing*, vol. 70(1–3), pp. 489-–501, Dez. 2006, https://doi.org/10.1016/j.neucom.2005.12.126.

[10] G.-B. Huang et al., "Extreme learning machine for regression and multiclass classification," *IEEE Trans. Syst. Man Cybern. B*, vol. 42(2), pp. 513-–529, Oct. 2012, https://doi.org/10.1109/TSMCB.2011.2168604.

[11] V. Dwivedi and B. Srinivasan, "Physics-informed extreme learning machines (PIELM): A rapid method for the numerical solution of partial differential equations," *Neurocomp.*, vol. 391, pp. 96–118, May 2020, https://doi.org/10.1016/j.neucom.2019.12.099.

[12] S. Panghal and M. Kumar, 'Optimization free neural network approach for solving ordinary and partial differential equations," *Engineering with Computers*, vol. 37, pp. 2989-–3002, Feb. 2021, https://doi.org/10.1007/s00366-020-00985-1.

[13] G. Fabiani et al., 'Numerical solution and bifurcation analysis of nonlinear partial differential equations with extreme learning machines," *Journal of Scientific Computing*, vol. 89, pp. 44, Oct. 2021, https://doi.org/10.1007/s10915-021-01650-5.

[14] E. Schiassi et al., "Extreme theory of functional connections: A fast physics-informed neural network method for solving ordinary and partial differential equations," *Neurocomputing*, vol. 457, pp. 334-–356, Oct. 2021, https://doi.org/10.1016/j.neucom.2021.06.015.

[15] F. Calabrò, G. Fabiani and C. Siettos, "Extreme learning machine collocation for the numerical solution of elliptic PDEs with sharp gradients," *Comput. Methods Appl. Mech. Eng.*, vol. 387, pp. 114188, Dez. 2021, https://doi.org/10.1016/j.cma.2021.114188.

[16] S. Dong and Y. Wang, "A method for computing inverse parametric PDE problems with random-weight neural networks," *J. Comput. Phys.*, vol. 489, pp. 112263, Sep. 2023, https://doi.org/10.1016/j.jcp.2023.112263.

[17] X. Liu et al., "Bayesian physics-informed extreme learning machine for forward and inverse PDE problems with noisy data," *Neurocomputing*, vol. 549, pp. 126425, Sep. 2023, https://doi.org/10.1016/j.neucom.2023.126425.

[18] N. Ni and S. Dong, 'Numerical Computation of Partial Differential Equations by Hidden-Layer Concatenated Extreme Learning Machine," *J. Sc. Comp.*, vol. 95, pp. 35, Mar. 2023, https://doi.org/10.1007/s10915-023-02162-0.

[19] S. Dong and Z. Li, 'Local extreme learning machines and domain decomposition for solving linear and nonlinear partial differential equations," *Comput. Methods Appl. Mech. Eng.*, vol. 387, pp. 114129, Dez. 2021, https://doi.org/10.1016/j.cma.2021.114129.

[20] C.-A. Yan, R. Vescovini and L. Dozio, "A framework based on physics-informed neural networks and extreme learning for the analysis of composite structures," *Comput. Methods Appl. Mech. Eng.*, vol. 265, pp. 106761, Jun. 2022, https://doi.org/10.1016/j.compstruc.2022.106761.

[21] S. Dong and J. Yang, 'Numerical approximation of partial differential equations by a variable projection method with artificial neural networks," *Comput. Methods in Appl. Mech. Eng.*, vol. 398, pp. 115284, Aug. 2022, https://doi.org/10.1016/j.cma.2022.115284.

[22] "Automated Solution of Differential Equations by the Finite Element Method: The FEniCS Book." Heidelberg, Germany: Springer Berlin, 2016 https://10.1007/978-3-642-23099-8.

[23] C. Geuzaine and J.-F. Remacle, 'Gmsh: A 3-D finite element mesh generator with built-in pre- and post-processing facilities," *Num. Methods Eng.*, vol. 79, pp. 1309–1331, May 2009, https://doi.org/10.1002/nme.2579.

[24] L. Demkowicz, "Computing with hp-Adaptive Finite Elements, Vol. 1: One and Two Dimensional Elliptic and Maxwell Problems", Chapman & Hall/CRC, 2006, https://doi.org/10.1201/9781420011685.

ANDREA SACCHETTI was born in Italy in 1977. He received the BS and MS (2001) as well as the Ph.D. degree (2005) in physics from the University of Rome "La Sapienza".

From 2005 to 2008 he was a Post-Doc fellow at the ETH Zurich. Furthermore, he collected industrial experience between 2008 and 2018 in various R&D-strong companies like Bruker Biospin, Sensirion, and Kistler Instruments. Since 2018, he is a full professor for physics at the University of Applied Sciences and Arts Northwestern Switzerland. His research interests include applications of experimental physics and of algorithms and simulations to product development and industrial research.

Prof. Sacchetti authored numerous publications and patents in different fields of basic and applied physics and is also referee of several international journals.