POINTWISE DETECTION OF WELL-TRAINED REGIONS IN PHYSICS-INFORMED NEURAL NETWORKS

Anonymous authors
Paper under double-blind review

ABSTRACT

Physics-informed neural networks (PINNs) rely on residual losses to enforce governing equations, but a small residual at an interior point does not guarantee that the predicted value matches the solution selected by the boundary or initial data. We study this pointwise detection problem: how to identify, during training, which domain points are already well trained using only quantities available to the PINN. We introduce a boundary-seeded detector that grows a well-trained set through local parent–child admissions. A candidate is admitted only when its own residual is small, its parent's residual remains small, and the parent's prediction has not drifted from the value stored at admission. The residual threshold is predicted from PDE-level features by a small structural meta-model, avoiding online estimation of stability constants or parameter-space sampling. The same primitive supports two uses: a conservative coverage-based stopping signal and an adaptive-domain training scheme with stored-value self-distillation. Across multiple PDE families, the detector reduces the false-stop rate from at least 40.9% for aggregate residual and residual–boundary criteria to 6.8%, while the detection-driven frontier improves training on a stiff convection benchmark and Helmholtz.

1 INTRODUCTION

A partial differential equation (PDE) defines an unknown solution $u^\star$ by requiring it to satisfy the governing equation $\mathcal{F}[u^\star](\mathbf{x}) = 0$ in the interior domain Ω, where $\mathcal{F}$ is a differential operator defining the governing equation, and the prescribed boundary or initial data $u^\star(\mathbf{x}) = g(\mathbf{x})$ on $\partial\Omega$. Physics-informed neural networks (PINNs) (Lagaris et al., 1998; Raissi et al., 2019) approximate $u^\star$ by a neural network u_θ and train it by minimizing an interior residual loss $\mathbb{E}_{\mathbf{x}\in\Omega}[|r_\theta(\mathbf{x})|^2]$, where $r_\theta(\mathbf{x}) := \mathcal{F}[u_\theta](\mathbf{x})$, together with boundary or initial-condition penalties on $\partial\Omega$ (Karniadakis et al., 2021; Lu et al., 2021). However, a small residual at a point only indicates that u_θ approximately satisfies the governing equation there; it does not imply that the pointwise error $|u_\theta(\mathbf{x}) - u^\star(\mathbf{x})|$ is small. Thus, being *well trained* at $\mathbf{x}$ means that $u_\theta(\mathbf{x})$ is close to the true value $u^\star(\mathbf{x})$, not merely that $|r_\theta(\mathbf{x})|$ is small. This gap is fundamental during PINN training because $u^\star(\mathbf{x})$ is unobserved at interior points, making the residual the most accessible but incomplete signal for pointwise correctness (Mishra & Molinaro, 2023; Liu et al., 2023). This motivates the central question of this paper: during training, how can we detect which individual points in the domain are already well trained using only quantities available to the PINN? Such pointwise detection is a primitive rather than a stopping rule itself: once a well-trained set is identified, it can be used either as a conservative stopping signal or as a data-driven frontier for adaptive-domain training.

A simple example shows why a pointwise residual threshold is insufficient. On the transport equation $u_t + \beta u_x = 0$ with a nonconstant initial condition, any constant function has zero interior residual because both derivatives vanish, yet it can be far from the solution induced by the initial condition. Adding boundary or initial-condition losses does not fully remove this ambiguity. A small boundary loss indicates agreement on the supervised slice, but it does not by itself certify that this information has propagated to the interior. In propagation-failure regimes, correct boundary or initial supervision may coexist with a low-residual but incorrect interior prediction (Wu et al., 2025). Our experiments confirm that this issue is not merely an artifact of batch averaging: aggregate thresholds on residual or residual–boundary losses produce many false stops, and applying a residual threshold pointwise on grid candidates also frequently accepts incorrect interior points. Thus, the key issue is not simply

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

how to choose a better residual threshold, but how to decide whether local residual satisfaction at a candidate point is anchored to already reliable information.

A related line of uncertainty-aware PINN methods estimates confidence from agreement across parameter-space samples, such as ensembles or Bayesian posterior samples (Haitsiukevich & Ilin, 2023; Innerebner et al., 2025). While useful for uncertainty estimation, agreement is not the same as pointwise correctness: multiple samples can agree on the same wrong-phase or low-complexity solution (Krishnapriyan et al., 2021; Pilar et al., 2025). Moreover, parameter-space sampling adds substantial cost and makes the diagnostic depend on the chosen model family and sampling procedure (Innerebner et al., 2025). We instead seek a lightweight detector based only on quantities already available in ordinary PINN training: pointwise residuals, boundary or initial anchors, and stored predictions at previously admitted points.

Stability-guided pointwise admission. Our detector is inspired by classical error analysis. For PDEs satisfying suitable local stability estimates, the error in a neighborhood can be controlled by the residual in that neighborhood together with boundary mismatch (LeVeque, 2007). This suggests that residual satisfaction at a candidate point should not be used in isolation, but should be checked relative to nearby points already admitted as well-trained. We therefore use a one-step parent–child admission rule: a candidate is admitted only if (i) its own residual is small, (ii) its nearby parent's residual remains small, and (iii) the parent's current prediction has not drifted far from its stored value. The first two checks enforce local PDE consistency, while the third verifies that the parent remains a stable anchor. The only PDE-dependent scalar is the residual threshold $\hat{\tau}$, which we predict from PDE-level features using a small structural meta-model rather than estimating stability constants online. Thus, the detector avoids ensembles or posterior sampling while remaining usable across architectures.

The admitted set supports two uses. First, as a passive observer, it provides a coverage-based stopping signal: training stops only when the admitted set covers the discretized domain. This is not an unconditional certificate, but in our experiments it substantially reduces false stops compared with aggregate residual and residual–boundary loss criteria. Second, the same admitted set can guide training: stored predictions provide self-distillation targets, while the residual loss is evaluated near the current detected frontier (Laine & Aila, 2017). As a result, the active training region becomes detection-driven rather than following a hand-designed temporal window schedule: the frontier advances only when newly trusted points are verified (Wang et al., 2024; Yan & He, 2024; Bekele, 2024; Guo et al., 2025). On challenging time-dependent and elliptic benchmarks, this detection-driven frontier improves final solution accuracy when the training bottleneck is procedural rather than purely representational.

In summary, this work comprises three **contributions**:

- **A pointwise detection primitive for well-trained regions.** We introduce a depth-one parent–child admission rule (Definition 1, Section 3) that combines candidate residual, parent residual, and parent value-drift checks, using only quantities available during ordinary PINN training and requiring no parameter-space sampling.
- **A conservative coverage-based stopping signal.** When used passively, the admitted set provides a stop criterion once it covers the discretized domain. Across parameter-interpolation, parameter-extrapolation, and PDE-type-extrapolation splits, it substantially reduces false stops compared with aggregate residual and residual–boundary loss criteria, though it is not an unconditional certificate.
- **Adaptive-domain training with buffer self-distillation.** When used actively, the same admitted set supplies stored-value self-distillation targets and determines the active training frontier. This detection-driven frontier improves training on a stiff convection benchmark and is also instantiated on Helmholtz at $k = 1$ as a static-PDE proof of concept; its applicability to higher wavenumbers and broader elliptic regimes is discussed as a limitation.

2 PRELIMINARIES

Setup. Following the standard PINN formulation (Krishnapriyan et al., 2021), we consider PDE-constrained learning problems of the form

$$\mathcal{F}[u^*](\mathbf{x}) = 0, \qquad \mathbf{x} \in \Omega, \tag{1}$$

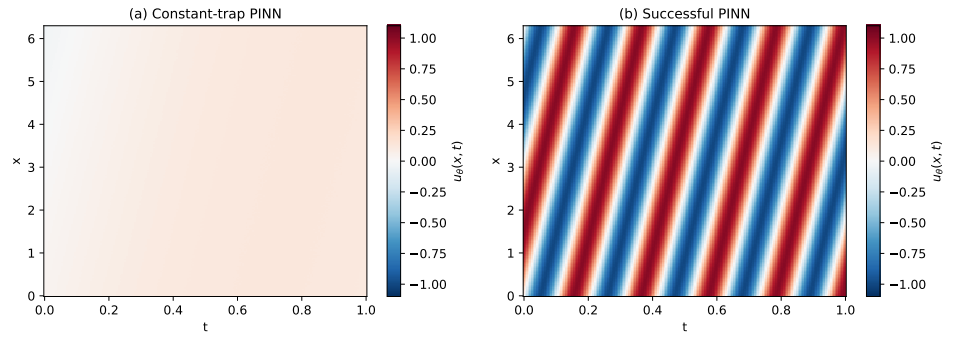

Figure 1: **The residual is not a reliable training signal.** Convection $u_t + \beta u_x = 0$ with $\beta = 30$. (a) Trapped PINN: rel-$L_2 = 1.01$, residual MSE 4.2×10^{-6}. (b) Successful PINN: rel-$L^2 = 6.5 \times 10^{-3}$, residual MSE 5.4×10^{-5}. The trapped run has the *smaller* residual but is two orders of magnitude worse in solution error.

together with value-type supervised boundary and/or initial conditions

$$u^*(\mathbf{x}) = g(\mathbf{x}), \qquad \mathbf{x} \in \partial\Omega. \tag{2}$$

Here Ω denotes the interior space–time domain, $\partial\Omega$ denotes the supervised boundary of this domain, and $\mathcal{F}$ is the differential operator defining the governing equation. For time-dependent equations, $\mathbf{x}$ contains both spatial and temporal coordinates, e.g., $\mathbf{x} = (x, t)$, and the initial-time slice is included in $\partial\Omega$; for steady-state equations, $\mathbf{x}$ contains only spatial coordinates. Throughout the paper, $\mathbf{x}$ denotes a generic point in $\overline{\Omega} := \Omega \cup \partial\Omega$; when role disambiguation is needed, we write $\mathbf{x}_c$ for a candidate point under verification and $\mathbf{x}_p$ for its parent.

A PINN parameterizes the unknown solution by a neural network $u_\theta : \overline{\Omega} \to \mathbb{R}$ and uses automatic differentiation to evaluate the pointwise PDE residual

$$r_\theta(\mathbf{x}) := \mathcal{F}[u_\theta](\mathbf{x}). \tag{3}$$

Training then imposes the PDE as a soft constraint by minimizing a weighted objective of the form

$$\mathcal{L}_{\mathrm{PINN}}(\theta) = \lambda_r \, \mathbb{E}_{\mathbf{x}\in\Omega}\left[r_\theta(\mathbf{x})^2\right] + \lambda_b \, \mathbb{E}_{\mathbf{x}\in\partial\Omega}\left[\big(u_\theta(\mathbf{x}) - g(\mathbf{x})\big)^2\right]. \tag{4}$$

The pointwise error

$$e_\theta(\mathbf{x}) := u_\theta(\mathbf{x}) - u^*(\mathbf{x}) \tag{5}$$

is unobservable during training because u^* is unavailable in the interior. Thus, the residual $r_\theta(\mathbf{x})$ is the most direct training-time signal available at interior points, but only measures local satisfaction of the PDE constraint, not whether $u_\theta(\mathbf{x})$ matches the solution selected by the boundary or initial data.

The residual is not a reliable signal. A small pointwise residual does not, by itself, imply that an interior prediction is trustworthy. The residual checks local satisfaction of the differential equation at the sampled point, but not whether the predicted value is anchored to the solution selected by the boundary or initial data. For example, for the linear transport equation $u_t + \beta u_x = 0$ with a nonconstant initial condition, any constant function $u \equiv c$ has zero pointwise residual everywhere, since both u_t and u_x vanish, even though it can be far from the solution induced by the initial condition. The same issue appears in actual PINN training: aggregate residual and boundary losses can fall below their thresholds while the prediction remains far from the true solution. Figure 1 illustrates this failure mode on convection: the trapped run attains a smaller residual than the successful run (4.2×10^{-6} vs. 5.4×10^{-5}), yet its rel-L_2 error is two orders of magnitude larger (1.01 vs. 6.5×10^{-3}).

Verification notation. The detector maintains a finite well-trained set $\mathcal{W}$ and a detection shell $\mathcal{D}$. Each admitted point $\mathbf{x} \in \mathcal{W}$ stores a buffered value $v_{\mathbf{x}}$ and a parent pointer $\mathrm{par}(\mathbf{x})$. Here $v_{\mathbf{x}}$ denotes the value associated with $\mathbf{x}$ in the detector's buffer, and $\mathrm{par}(\mathbf{x})$ denotes its parent point in the detection tree. Boundary or initial anchors have no parent, denoted by $\mathrm{par}(\mathbf{x}) = \perp$, and have $v_{\mathbf{x}} = g(\mathbf{x})$. The shell $\mathcal{D}$ contains candidate points $\mathbf{x}_c$ at L_∞-distance δ from $\mathcal{W}$, each linked to the parent point

$\mathbf{x}_p \in \mathcal{W}$ that spawned it. Initially, $\mathcal{W}$ is a discretization of the supervised boundary/initial set $\partial\Omega$, and $\mathcal{D}$ is the corresponding interior δ-shell. We use $\hat{\tau}$ for the PDE-dependent residual threshold, ϵ_B for boundary/initial admission, and ϵ for parent-value drift. Only $\hat{\tau}$ is PDE-dependent and is predicted by the meta-model in Section 3.3; ϵ_B and ϵ are fixed auxiliary tolerances.

3 METHOD: POINTWISE DETECTIVE VERIFICATION

We first present the detective algorithm (Section 3.1) and then provide a one-step stability bound that motivates its three admission checks (Section 3.2). We then describe how the PDE-dependent threshold is chosen (Section 3.3) and how the same primitive is used for passive stopping and active adaptive-domain training (Section 3.4).

3.1 THE DETECTIVE ALGORITHM

Definition 1 (Parent–child admission rule)**.** A supervised point $\mathbf{x} \in \partial\Omega$ is admitted as an anchor if

$$|u_\theta(\mathbf{x}) - g(\mathbf{x})| \le \epsilon_B,$$

in which case we add $\mathbf{x}$ to the initial well-trained set $\mathcal{W}$, set $\mathrm{par}(\mathbf{x}) = \perp$, and set $v_{\mathbf{x}} = g(\mathbf{x})$. An interior candidate $\mathbf{x}_c$ with parent $\mathbf{x}_p \in \mathcal{W}$ and $\|\mathbf{x}_c - \mathbf{x}_p\|_\infty = \delta$ is admitted if, at the current parameters θ,

$$|r_\theta(\mathbf{x}_c)| < \hat{\tau}, \qquad \textbf{(child_ok)} \tag{6}$$

$$|r_\theta(\mathbf{x}_p)| < \hat{\tau}, \qquad \textbf{(parent_res_ok)} \tag{7}$$

$$\left|u_\theta(\mathbf{x}_p) - v_{\mathbf{x}_p}\right| \le \epsilon, \qquad \textbf{(parent_val_ok)} \tag{8}$$

where $\hat{\tau}$ is the PDE-dependent residual threshold and ϵ is the value-drift tolerance. When $\mathbf{x}_c$ is admitted, we add $\mathbf{x}_c$ to $\mathcal{W}$, set $v_{\mathbf{x}_c} = u_\theta(\mathbf{x}_c)$, and set $\mathrm{par}(\mathbf{x}_c) = \mathbf{x}_p$.

Throughout, $\hat{\tau}$ is supplied by the meta-model of Section 3.3; in Sections 3.1–3.2 we treat it as a fixed scalar. Supervised anchors provide the base case: since $v_{\mathbf{x}} = g(\mathbf{x}) = u^*(\mathbf{x})$ for $\mathbf{x} \in \partial\Omega$, their buffered values are exact, while ϵ_B only controls whether the current network prediction is close enough to the supervised value for the anchor to be activated.

The three checks play distinct roles. Condition (6) tests whether the candidate locally satisfies the PDE. Condition (7) re-checks the parent at the current parameters, guarding against anchors whose local residual has degraded since promotion. Condition (8) compares the parent's current prediction with the value stored when that parent was admitted, and therefore tests whether the local anchor used for propagation has remained stable. The *wavefront* is the subset of $\mathcal{W}$ that currently has children in the detection shell. A persistent wrong-phase solution may continue to satisfy the PDE residual locally, but it changes predictions at previously admitted parents; once this drift exceeds ϵ, (8) fails on parents of active candidates and the wavefront stops advancing.

Detection cycle. Training and verification are decoupled. Every K optimizer steps, we run a detection cycle (Algorithm 1). For each candidate $\mathbf{x}_c$ in the current detection shell, we evaluate the three checks in Definition 1 at the current parameters. Candidates that pass all checks are promoted together, assigned buffered values $v_{\mathbf{x}_c} = u_\theta(\mathbf{x}_c)$, and used to spawn the next δ-shell. Repeating this procedure up to S times per check allows the wavefront to advance at a controlled pace. The cadence K and shell budget S are implementation knobs chosen per experiment. Algorithm 1 rechecks only parents of active candidates; already admitted interior points are not globally revalidated or retracted. Thus, a failed check stalls further propagation from the affected wavefront rather than deleting previously admitted points.

Computational cost. Each detection cycle evaluates u_θ and r_θ on candidates and their parents. Since each candidate has one parent, the cost per check is $O(|\mathcal{D}|)$ forward-and-residual evaluations, where $\mathcal{D}$ is the current detection shell. A cycle is run once every K optimizer steps and may expand up to S shells per cycle, so its amortized per-step cost is $O((S/K)\,|\mathcal{D}|)$. For the active use of Section 3.4, the buffer loss in Eq. (11) adds $O(|\mathcal{W}|)$ forward evaluations per gradient step on a fixed buffer subsample, which is the dominant overhead once the admitted set is large. In practice, we use $K{=}2000$ and $S = 5$ in Table 3 and a buffer subsample of size at most 512, so all variants run within the same order of wall-clock as a vanilla PINN at the same architecture and total step budget.

3.2 THEORETICAL FOUNDATION

The admission rule is motivated by a two-step local argument. First, because the PINN residual is obtained by automatic differentiation from a smooth network, small residuals at the two endpoints of a parent–child edge control the residual in the small neighborhood between them. Second, a local residual-stability condition turns this small residual defect, together with a trusted parent value, into a bound on the child error. The first step is network/operator regularity; the second step is the PDE-side condition under which the detector is intended to operate.

Algorithm 1 Detective cycle (one check per K steps)

Input: θ; $\mathcal{W}$; $\mathcal{D}$; $\hat{\tau}$; ϵ, ϵ_B; S.
for $s = 1, \dots, S$ **do**
  $\mathcal{P} \leftarrow \emptyset$
  for $\mathbf{x}_c \in \mathcal{D}$, $\mathbf{x}_p \leftarrow \mathrm{par}(\mathbf{x}_c)$ **do**
    if $|r_\theta(\mathbf{x}_c)| < \hat{\tau}$, $|r_\theta(\mathbf{x}_p)| < \hat{\tau}$, $\left|u_\theta(\mathbf{x}_p) - v_{\mathbf{x}_p}\right| \le \epsilon$ **then**
      $\mathcal{P} \leftarrow \mathcal{P} \cup \{\mathbf{x}_c\}$; $\tilde{v}_{\mathbf{x}_c} \leftarrow u_\theta(\mathbf{x}_c)$
    end if
  end for
  Add $\mathbf{x}_c \in \mathcal{P}$ to $\mathcal{W}$ with $v_{\mathbf{x}_c} = \tilde{v}_{\mathbf{x}_c}$
  $\mathcal{D} \leftarrow (\mathcal{D} \setminus \mathcal{P}) \cup (\mathrm{shell}_\delta(\mathcal{P}) \setminus \mathcal{W})$
  if $\mathcal{P} = \emptyset$ **or** $\mathcal{D} = \emptyset$ **then break**
end for
return $\mathcal{W}, \mathcal{D}$

Let $\mathcal{N}_\delta(\mathbf{x}_p, \mathbf{x}_c)$ denote the local neighborhood associated with a parent–child pair. We choose $\mathcal{N}_\delta(\mathbf{x}_p, \mathbf{x}_c)$ so that it is covered by the two endpoint balls of radius $\delta/2$: for every $\mathbf{z} \in \mathcal{N}_\delta(\mathbf{x}_p, \mathbf{x}_c)$,

$$\min\{\|\mathbf{z} - \mathbf{x}_p\|_\infty, \|\mathbf{z} - \mathbf{x}_c\|_\infty\} \le \delta/2.$$

This holds, for example, when $\mathcal{N}_\delta(\mathbf{x}_p, \mathbf{x}_c)$ is the closed segment between the two endpoints, or any local cell satisfying this endpoint-cover property.

Assumption 1 (Network-induced residual regularity)**.** For the current network u_θ, the residual $r_\theta = \mathcal{F}[u_\theta]$ is Lipschitz on each parent–child neighborhood $\mathcal{N}_\delta(\mathbf{x}_p, \mathbf{x}_c)$ with constant L_r:

$$|r_\theta(\mathbf{z}_1) - r_\theta(\mathbf{z}_2)| \le L_r \|\mathbf{z}_1 - \mathbf{z}_2\|_\infty, \qquad \mathbf{z}_1, \mathbf{z}_2 \in \mathcal{N}_\delta(\mathbf{x}_p, \mathbf{x}_c).$$

This assumption follows when u_θ has the required smooth, bounded derivatives and $\mathcal{F}$ is formed from bounded additions, multiplications, and derivatives of u_θ; a sufficient condition is given in Appendix A.2.

Assumption 2 (Local residual stability along an admitted edge)**.** For each admitted parent–child pair $(\mathbf{x}_p, \mathbf{x}_c)$ with $\|\mathbf{x}_c - \mathbf{x}_p\|_\infty = \delta$, there exist constants $C_{\mathrm{stab}} > 0$ and $C_{\mathrm{reg}} \ge 0$ such that the error $e_\theta := u_\theta - u^*$ satisfies

$$|e_\theta(\mathbf{x}_c) - e_\theta(\mathbf{x}_p)| \le C_{\mathrm{stab}} \|r_\theta\|_{L^\infty(\mathcal{N}_\delta(\mathbf{x}_p, \mathbf{x}_c))} + C_{\mathrm{reg}}\delta. \tag{9}$$

Assumption 2 is a local, discretized stability condition: it says that over a small parent–child neighborhood, the error cannot change much unless the PDE residual is large. It is motivated by residual-based a posteriori error analysis, where solution error is controlled by residual defects and stability constants (Estep et al., 2002; Beck et al., 2000). We do not assume that it holds uniformly for all PDEs or in every geometric direction; rather, it characterizes the local stability regime in which boundary-anchored propagation is meaningful.

Proposition 1 (Conditional one-step error bound)**.** *Fix the current parameters θ. Let $\mathbf{x}_p \in \mathcal{W}$ be an admitted parent with stored value $v_{\mathbf{x}_p}$ satisfying*

$$\left|v_{\mathbf{x}_p} - u^*(\mathbf{x}_p)\right| \le E_{\mathbf{x}_p}.$$

Let $\mathbf{x}_c$ be an interior candidate with $\|\mathbf{x}_c - \mathbf{x}_p\|_\infty = \delta$. Suppose the three admission checks of Definition 1 hold at the current parameters:

$$|r_\theta(\mathbf{x}_c)| < \hat{\tau}, \quad |r_\theta(\mathbf{x}_p)| < \hat{\tau}, \quad \left|u_\theta(\mathbf{x}_p) - v_{\mathbf{x}_p}\right| \le \epsilon.$$

Then under Assumptions 1–2,

$$|u_\theta(\mathbf{x}_c) - u^*(\mathbf{x}_c)| \le E_{\mathbf{x}_p} + \epsilon + C_{\mathrm{stab}}\left(\hat{\tau} + \frac{1}{2}L_r\delta\right) + C_{\mathrm{reg}}\delta. \tag{10}$$

In particular, the weaker but simpler bound with $C_{\mathrm{stab}}(\hat{\tau} + L_r\delta)$ also holds.

Proof is given in Appendix A. The proposition mirrors the detector: **child_ok** and **parent_res_ok** control the residual on the local parent–child neighborhood through Assumption 1, while **parent_val_ok** transfers the stored parent error bound to the current network. Supervised anchors provide the base case: for $\mathbf{x} \in \partial\Omega$, $v_{\mathbf{x}} = g(\mathbf{x}) = u^*(\mathbf{x})$, so $E_{\mathbf{x}} = 0$ at admission.

3.3 CHOOSING THE THRESHOLD $\hat{\tau}$

The one-step bound in Proposition 1 contains the term $C_{\mathrm{stab}}(\hat{\tau} + \frac{1}{2}L_r\delta)$. Thus, the same numerical residual threshold can correspond to very different error scales across PDEs, because the local stability constant and residual-to-error map depend on the operator, coefficients, geometry, and boundary or initial conditions. This makes a single hand-tuned $\hat{\tau}$ difficult to transfer across PDE families.

We therefore treat $\hat{\tau}$ as a lightweight calibration parameter rather than an analytically estimated stability constant. Given a PDE, a small structural meta-model predicts a positive threshold $\hat{\tau} = \exp(h_\phi(\phi(\mathrm{PDE})))$ from architecture-independent PDE features such as dimension, derivative order, coefficient scales, operator class, source magnitude, and flags for time dependence, nonlinearity, and advection. The labels are residual scales observed at convergence in a set of reference PINN runs: for each run, we use the 90-th percentile of $|r_\theta|$ at the first checkpoint where the relative L_2 error falls below the target tolerance. The meta-model is not intended to recover C_{stab} or to provide a formal guarantee; it only supplies a per-PDE residual scale at which the detector operates. Training details, feature definitions, and leave-family / leave-class diagnostics are reported in Appendix B.

3.4 USING THE DETECTED SET

The detector produces a boundary-anchored set $\mathcal{W}$ of points that have passed the parent–child admission rule. We use this set in two ways, corresponding to two practical questions in PINN training. First, can we stop training once the entire domain is deemed trustworthy? Second, can the trusted region itself guide where the model should train next? These applications are not separate algorithms; they are two ways of consuming the same detected set.

Passive use: stopping. As a stopping signal, the detective runs as a passive observer of a standard PINN training loop. Training stops when $\mathcal{W}$ covers the discretized domain, equivalently when no detection-shell candidates remain. The motivation is practical: aggregate residual and boundary losses can both be small while the solution error remains large, so a stop rule should require pointwise coverage by trusted points rather than convergence of batch statistics alone. Empirically, the full rule substantially reduces false stops compared with aggregate residual–boundary criteria and with pointwise ablations that remove parent validation.

Active use: adaptive-domain training. The same set $\mathcal{W}$ can also be written back into training. The motivation is different from stopping: difficult PINN regimes often fail because optimization advances into regions before earlier regions are reliable, so a detected trust frontier provides a data-driven way to decide where training should focus next. Each admitted point contributes a stored-value target, giving the buffer loss

$$\mathcal{L}_{\mathrm{buf}}(\theta) = \frac{1}{|\mathcal{W}|}\sum_{\mathbf{x}\in\mathcal{W}} \big(\max\{|u_\theta(\mathbf{x}) - v_{\mathbf{x}}| - \epsilon, 0\}\big)^2. \tag{11}$$

The detected frontier also determines the active collocation region. For time-dependent PDEs, we restrict residual collocation to a margin ahead of the latest verified time, $\Omega \cap \{t \le t_{\mathrm{front}} + M\Delta t\}$, where t_{front} is the largest time coordinate in $\mathcal{W}$. For static PDEs, where no temporal ordering exists, we use the same principle spatially: the active region is the union of cells within an M-shell margin of the current boundary-seeded wavefront. Thus, the training domain grows only when the detector verifies newly trusted points, rather than following a hand-designed temporal or spatial schedule. The total loss is $\mathcal{L}_{\mathrm{PINN}} + \lambda_{\mathrm{buf}}\mathcal{L}_{\mathrm{buf}}$.

To avoid freezing early admission errors into the buffer, we refresh non-anchor buffered values by an exponential moving average (Tarvainen & Valpola, 2017; Laine & Aila, 2017) at each detection cycle,

$$v_{\mathbf{x}} \leftarrow (1-\alpha)v_{\mathbf{x}} + \alpha u_\theta(\mathbf{x}), \qquad \mathbf{x} \in \mathcal{W} \setminus \partial\Omega. \tag{12}$$

Boundary or initial anchors are not updated. The EMA is an empirical stabilization device for active training, not part of the static one-step guarantee in Proposition 1. Thus, both applications use the same admission logic and threshold, but only differ in whether $\mathcal{W}$ is read as a stopping signal or written into the loss as supervision and domain restriction.

4 EXPERIMENTS

This section reports results for Application 1, the stop-criterion use of the detective. Application 2 (the buffer-supervised adaptive-domain benchmark) is reported separately and shares no findings with this section. All comparisons use the *final* relative L_2 error at the end of training, never best-in-run. Meta-model training, dataset construction, and the per-run breakdown are deferred to Appendix B; throughout Section 4.1, the meta-model is treated as a fixed predictor that maps PDE structure to $\hat{\tau}$.

4.1 APPLICATION 1: STOP CRITERION DRIVEN BY THE PREDICTED THRESHOLD

Table 1: Outcome breakdown for the full detective using the predicted $\hat{\tau}$. Trained: 32 configurations across the 8 PDE families used to train the meta-model. Unseen: 12 configurations across three held-out families.

Regime	Successful Stop	False Stop	Missed Det.	Correct Non-conv.
Trained families	10 (31.3%)	2 (6.3%)	5 (15.6%)	15 (46.9%)
Unseen families	7 (58.3%)	1 (8.3%)	3 (25.0%)	1 (8.3%)

We first evaluate the detective as a stop criterion. This application tests the central claim of the paper: low residual values, even when boundary loss convergence is achieved, are not sufficient to determine whether the learned solution is pointwise trustworthy. A run is *converged* when its final relative L_2 error is at most $\varepsilon = 5 \times 10^{-3}$, and a stop criterion *fires* when its stopping condition is met within the step budget. We report four outcomes: Successful Stop, False Stop, Missed Detection, and Correct Non-convergence. False Stop is the primary safety metric because it captures the failure mode in which a criterion declares convergence even though the solution error remains above the tolerance. Per-run tables are provided in Appendix B.

Predicted thresholds across PDE families. We first test whether the predicted threshold $\hat{\tau}$ provides a useful residual scale across PDEs. The evaluation contains 44 configurations: 32 configurations from the 8 PDE families used to train the threshold meta-model, and 12 configurations from three held-out families (`wave_1d`, `nonlinear_poisson_2d`, and `reaction_diffusion_1d`). No test coefficient appears in the meta-model training set: the trained-family configurations include both coefficient interpolation and coefficient extrapolation, while the held-out families are unseen at the family level. Table 1 reports the outcome breakdown.

Table 2: Stop criterion vs. residual-based baselines on the same 44 benchmarks. L_r and L_b are batch means of the residual and boundary losses. Pointwise ablations isolate the contribution of parent residual validation and parent-value drift validation.

Stop condition	False Stop
$L_r < 10^{-2}$	$30/44$ (68.2%)
$L_r < 10^{-3}$	$21/44$ (47.7%)
$L_r < 10^{-2}$, $L_b < 10^{-3}$	$27/44$ (61.4%)
$L_r < 10^{-3}$, $L_b < 10^{-4}$	$18/44$ (40.9%)
child-only pointwise residual	$19/44$ (43.2%)
child + parent residual, no value-check	$7/44$ (15.9%)
detective, full rule (ours)	$\mathbf{3/44}$ ($\mathbf{6.8}$%)

The detective produces $3/44$ False Stops in total: $2/32$ on trained families and $1/12$ on unseen families. Thus, the predicted threshold does not provide a formal no-false-positive guarantee, but it places the residual scale in a useful range across both seen and unseen PDE families. The remaining Missed Detections correspond to runs whose final error satisfies the convergence tolerance but whose wavefront does not cover the domain within the step budget; these cases reflect the propagation speed of the detected frontier rather than only threshold calibration. Overall, this experiment supports the role of $\hat{\tau}$ as a lightweight calibration mechanism for the detector, not as an independent certificate of convergence. A step-by-step visualization of the detective wavefront on convection $\beta = 50$, illustrating how the admitted set automatically advances from the supervised initial slice as training progresses, is provided in Figure 2 (Appendix B.4).

Why boundary-anchored pointwise verification matters. We next compare against stopping rules that use the same training signals but consume them differently. The batch baselines stop

when the mean residual loss, or the mean residual and boundary losses together, stay below fixed thresholds for three consecutive checks. These baselines test whether aggregate convergence of the standard PINN losses is enough. The aggregate thresholds are hand-tuned constants on L_r and L_b rather than pointwise applications of the predicted $\hat{\tau}$: the comparison is designed to isolate aggregate-versus-pointwise consumption of the residual signal, not tuned-versus-predicted threshold. We also include pointwise ablations of the detective: a child-only residual rule, and a rule that uses both child and parent residuals but removes the parent-value drift check.

Table 2 shows a clear separation between these criteria. All aggregate residual or residual–boundary baselines produce at least $18/44$ False Stops (40.9%), confirming that convergence of the losses optimized by a PINN does not reliably imply small solution error. Applying the residual threshold pointwise to candidate samples is also insufficient: the child-only rule still produces $19/44$ False Stops. Adding the parent residual check reduces this number to $7/44$, showing the value of local parent–child consistency. The full detective further reduces False Stops to $3/44$ by adding the parent-value check, which prevents the wavefront from advancing from anchors whose predictions have drifted. This ablation supports the main mechanism of the method: trustworthy pointwise detection requires residual satisfaction to be tied to a stable chain of already admitted points.

Sensitivity to wavefront speed. Finally, we vary the shell budget S, which controls the number of shell-expansion rounds the detector may perform at each verification check. This experiment tests a practical consequence of the theory: the detector should not advance the trust frontier arbitrarily fast, because each promotion assumes that the parent–child neighborhood is locally stable at the current network parameters. Table 3 shows the expected trade-off. With $S = 1$, the wavefront advances too slowly: the criterion produces no Successful Stops on the trained-family suite and yields many Missed Detections. With $S = 200$, the wavefront can advance aggressively, increasing Successful Stops to $13/32$ but also increasing False Stops to $5/32$. The default $S = 5$ is more conservative: it gives $10/32$ Successful Stops with $2/32$ False Stops. We therefore use $S = 5$ as a safety-oriented compromise in the main experiments.

Table 3: Shell budget S on the trained-family suite (32 configurations), full four-class breakdown.

S	Succ.	False	Missed	Non-conv.
1	0 (0.0%)	1 (3.1%)	15 (46.9%)	16 (50.0%)
200	13 (40.6%)	5 (15.6%)	2 (6.2%)	12 (37.5%)
5 (ours)	10 (31.3%)	2 (6.3%)	5 (15.6%)	15 (46.9%)

4.2 Application 2: buffer-supervised adaptive-domain training

We next evaluate whether the detected set can improve PINN training when it is written back into the loss: if $\mathcal{W}$ identifies regions that are already trustworthy, then $\mathcal{W}$ can provide both stable self-distillation targets and a data-driven frontier for deciding where residual training should focus next. We test this in two complementary regimes: a high-speed convection problem, where PINNs often collapse to wrong-phase solutions, and a static Helmholtz problem with no temporal ordering, which probes whether the same boundary-seeded frontier is useful beyond time-marching intuition.

Time-dependent benchmark: convection at $\beta = 30$. We use the one-dimensional convection equation $u_t + 30u_x = 0$ on $[0, 2\pi] \times [0, 1]$ with $u(x, 0) = \sin(x)$. This benchmark is a natural stress test for our method because residual minimization alone admits low-residual trivial solutions, while accurate prediction requires propagating the initial condition across many oscillatory cycles. Table 4 compares our method against published Bayesian/ensemble pseudo-labeling baselines, the region-optimized PINN of (Wu et al., 2024), and internal ablations. The full method, detective with EMA buffer (D+B), reaches 6.9×10^{-3} final relative L_2, improving over the strongest published baseline, ensemble with pseudo-labels, at 1.02×10^{-2}, and over RoPINN at 1.18×10^{-2}. Internal variants show that neither the detected frontier nor the buffer alone fully explains the gain. The most informative comparison is D+B versus Win.+Buf: both methods use EMA buffer self-distillation, but Win.+Buf advances the active region by a fixed linear time-window schedule, whereas D+B advances only when the detective verifies newly trusted points. The detective-driven frontier reduces the final error from 8.4×10^{-3} to 6.9×10^{-3}, isolating the contribution of detection-driven domain expansion from that of generic time-windowing and self-distillation. Removing the buffer (AW) degrades to 7.9×10^{-3}, and removing detective-driven domain restriction altogether (Buf., Win.)

Table 4: Convection $\beta = 30$, final rel-L_2. Published (Innerebner et al., 2025; Wu et al., 2024): Van. vanilla; B+PL / Bay. Bayesian $\pm$ pseudo-labels; E+PL / Ens. ensemble $\pm$ pseudo-labels; RoPINN region-optimized PINN. Internal: Det. detective frontier alone; Buf. EMA buffer, full domain; Win. / Win.+Buf fixed window $\mp$ buffer; AW auto-window alone; D+B full method. Pairs D+B vs. Win.+Buf, D+B vs. Det. isolate frontier and buffer.

Method	Van.	B+PL	Bay.	Ens.	E+PL	RoPINN
rel-$L_2 \downarrow$	1.59×10^{-2}	1.43×10^{-2}	1.21×10^{-2}	1.19×10^{-2}	1.02×10^{-2}	1.18×10^{-2}
Method	Det.	Buf.	Win.	Win.+Buf	AW	D+B
rel-$L_2 \downarrow$	1.08×10^{-2}	1.00×10^{-2}	1.00×10^{-2}	8.4×10^{-3}	7.9×10^{-3}	$\mathbf{6.9\times10^{-3}}$

yields 1.0×10^{-2}, indicating that the two components are complementary. We further ablate the buffer-loss weight λ_{buf} in Appendix C; the results suggest that overly large buffer weights can degrade performance. We vary detector and active-training parameters on one seed with $300{,}000$ AdamW updates (Appendix Table 10); the reported ranges retain lower errors than the fixed Vanilla baseline, although an overly strict residual threshold can stall domain expansion.

Static benchmark: Helmholtz on the unit square. We then test whether the same primitive is useful when the PDE has no time direction. For the Helmholtz equation $\nabla^2 u + k^2 u = f$ on $[0,1]^2$, the detected frontier expands spatially from the supervised boundary instead of advancing along time. This benchmark is important because it decouples the detector's value from temporal causality: any gain must arise from boundary-anchored spatial trust propagation and buffer stabilization, not from a time-window schedule. Table 5 shows that, at $k = 1$, D+B reduces the final relative L_2 error from 6.76×10^{-3} to 3.67×10^{-3}, a 46% improvement over vanilla PINN. The buffer-only variant (Buf.) shows that self-distillation alone can be harmful on the static benchmark: at $k = 10$ it matches vanilla, and at $k = 1$ it is worse than vanilla. The gap between D+B and Buf. suggests that, without a detective-driven frontier, early sparse buffered records may act as unbalanced self-distillation constraints. At $k = 10$, the full method still slightly improves over vanilla PINN, suggesting that the detective does not introduce harmful self-distillation even in a harder-to-converge regime. Because buffered values are written only at admitted points and refreshed by EMA, unreliable predictions are less likely to become fixed training targets that obstruct optimization. We additionally compare Vanilla, residual-based adaptive distribution (RAD), and both detective variants on Helmholtz at $k = 1$ across three seeds under a shared warm-up and update budget (Appendix Table 8); both detective variants achieve lower mean errors than Vanilla and RAD. Together, the two benchmarks show that the detected set can identify well-trained regions well enough to advance training when propagation is the bottleneck, while also filtering out unreliable values so that the buffer does not hinder training when the PDE is more difficult.

Table 5: Helmholtz $\nabla^2 u + k^2 u = f$ on $[0,1]^2$, final relative L_2 error. For static PDEs, the detector expands the active region spatially from the boundary. Method abbreviations: Van. = vanilla PINN; Det. = detective-driven frontier without EMA buffer; Buf. = EMA buffer without detective-driven frontier; D+B = full method, detective-driven frontier with EMA buffer.

k	Van.	Det.	Buf.	D+B
1	6.76×10^{-3}	6.07×10^{-3}	1.04×10^{-2}	$\mathbf{3.67\times10^{-3}}$
10	8.37×10^{-2}	7.88×10^{-2}	8.37×10^{-2}	$\mathbf{7.70\times10^{-2}}$

5 Conclusion

We present a pointwise detective verification mechanism for PINNs that uses a depth-one parent–child rule and a lightweight meta-model to identify well-trained regions. As a stop criterion, it reduces false stops to $3/44$ across 44 configurations spanning 11 PDE families, compared to at least $18/44$ for aggregate residual or residual-boundary criteria on the same suite. As a training driver with an EMA buffer and a wavefront-restricted collocation domain, it achieves 6.9×10^{-3} relative L_2 error on the architecture-parity convection $\beta = 30$ benchmark of (Innerebner et al., 2025), improving upon the strongest published baseline, and also outperforms vanilla PINN on the static Helmholtz benchmark at $k = 1$. Both uses share the same admission logic and threshold predictor, differing only in passive monitoring v.s. active loss modification.

AI USE STATEMENT

In this work, we used generative AI tools to formulate mathematical claims, provide key elements of mathematical proofs, implement methods, assist with translation, and assist in writing proofs. We also used these tools to summarize or analyze existing literature. We did not use generative AI tools to design or provide feedback on research methodology or experiments, develop theoretical models or conceptual frameworks, support qualitative or thematic data analysis, interpret results, generate synthetic datasets, propose or refine hypotheses, or clean and reformat datasets. We have reviewed all AI-assisted work and take responsibility for the final content, including any text, claims, or artifacts produced with the assistance of generative AI.

REPRODUCIBILITY STATEMENT

We provide detailed information to support reproduction of our results. The assumptions and complete proofs for the theoretical results are given in Section 3.2 and Appendix A. Appendix B describes the threshold meta-model, its training data and procedure, and per-run evaluation results. Appendix C provides the experimental protocols, ablations, and sensitivity analyses for adaptive-domain training, while Appendix D reports the implementation details, architectures, optimization settings, random seeds, computational resources, and hyperparameters. We plan to release the implementation and experimental data to facilitate independent reproduction.

REFERENCES

Roland Beck, Ralf Hiptmair, Ronald H. W. Hoppe, and Barbara Wohlmuth. Residual based a posteriori error estimators for eddy current computation. *ESAIM: Mathematical Modelling and Numerical Analysis*, 34(1):159–182, 2000.

Yared W Bekele. Physics-informed neural networks with curriculum training for poroelastic flow and deformation processes. *arXiv preprint arXiv:2404.13909*, 2024.

James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: composable transformations of Python+NumPy programs, 2018. URL `http://github.com/jax-ml/jax`.

Donald Estep, Michael Holst, and Duane Mikulencak. Accounting for stability: a posteriori error estimates based on residuals and variational analysis. *Communications in Numerical Methods in Engineering*, 18(1):15–30, 2002.

Yuan Guo, Zhuojia Fu, Jian Min, Shiyu Lin, Xiaoting Liu, Youssef F. Rashed, and Xiaoying Zhuang. Long-term simulation of physical and mechanical behaviors using curriculum-transfer-learning based physics-informed neural networks. *Neural Networks*, 2025. doi: 10.1016/j.neunet.2025.107868.

Katsiaryna Haitsiukevich and Alexander Ilin. Improved training of physics-informed neural networks with model ensembles. In *2023 International Joint Conference on Neural Networks (IJCNN)*, 2023. URL `https://arxiv.org/abs/2204.05108`.

Kevin Innerebner, Franz M. Rohrhofer, and Bernhard C. Geiger. B-PL-PINN: Stabilizing PINN training with Bayesian pseudo labeling, 2025. URL `https://arxiv.org/abs/2507.01714`. To appear in IJCNN 2025.

George Em Karniadakis, Ioannis G Kevrekidis, Lu Lu, Paris Perdikaris, Sifan Wang, and Liu Yang. Physics-informed machine learning. *Nature Reviews Physics*, 3(6):422–440, 2021.

Aditi Krishnapriyan, Amir Gholami, Shandian Zhe, Robert Kirby, and Michael W Mahoney. Characterizing possible failure modes in physics-informed neural networks. *Advances in Neural Information Processing Systems*, 34:26548–26560, 2021.

Isaac E Lagaris, Aristidis Likas, and Dimitrios I Fotiadis. Artificial neural networks for solving ordinary and partial differential equations. *IEEE Transactions on Neural Networks*, 9(5):987–1000, 1998.

Samuli Laine and Timo Aila. Temporal ensembling for semi-supervised learning. In *International Conference on Learning Representations (ICLR)*, 2017. URL https://arxiv.org/abs/1610.02242.

Randall J LeVeque. *Finite difference methods for ordinary and partial differential equations: Steady-state and time-dependent problems*. SIAM, 2007.

Shuheng Liu, Xiyue Huang, and Pavlos Protopapas. Residual-based error bound for physics-informed neural networks. In *Uncertainty in Artificial Intelligence (UAI), Proceedings of Machine Learning Research, vol. 216*, 2023. URL https://proceedings.mlr.press/v216/liu23b.html.

Lu Lu, Xuhui Meng, Zhiping Mao, and George E Karniadakis. Deepxde: A deep learning library for solving differential equations. *SIAM Review*, 63(1):208–228, 2021.

Siddhartha Mishra and Roberto Molinaro. Estimates on the generalization error of physics-informed neural networks for approximating PDEs. *IMA Journal of Numerical Analysis*, 43(1):1–43, 2023.

Philipp Pilar, Markus Heinonen, and Niklas Wahlström. Repulsive ensembles for Bayesian inference in physics-informed neural networks. *arXiv preprint arXiv:2505.17308*, 2025.

Maziar Raissi, Paris Perdikaris, and George 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*, 378:686–707, 2019.

Antti Tarvainen and Harri Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 30, 2017. URL https://arxiv.org/abs/1703.01780.

Sifan Wang, Shyam Sankaran, and Paris Perdikaris. Respecting causality for training physics-informed neural networks. *Computer Methods in Applied Mechanics and Engineering*, 421: 116813, 2024. ISSN 0045-7825. doi: https://doi.org/10.1016/j.cma.2024.116813. URL https://www.sciencedirect.com/science/article/pii/S0045782524000690.

Chenxi Wu, Min Zhu, Qinyang Tan, Yadhu Kartha, and Lu Lu. A comprehensive study of non-adaptive and residual-based adaptive sampling for physics-informed neural networks. *Computer Methods in Applied Mechanics and Engineering*, 403:115671, 2023.

Haixu Wu, Huakun Luo, Yuezhou Ma, Jianmin Wang, and Mingsheng Long. RoPINN: Region optimized physics-informed neural networks. In *Advances in Neural Information Processing Systems*, volume 37, 2024. URL https://arxiv.org/abs/2405.14369.

Haixu Wu, Yuezhou Ma, Hang Zhou, Huikun Weng, Jianmin Wang, and Mingsheng Long. ProPINN: Demystifying propagation failures in physics-informed neural networks. *arXiv preprint arXiv:2502.00803*, 2025.

Da Yan and Ligang He. Dp-pinn: A dual-phase training scheme for improving the performance of physics-informed neural networks. In *Computational Science – ICCS 2024*, pp. 19–32, Cham, 2024. Springer Nature Switzerland. ISBN 978-3-031-63749-0.

A PROOFS

A.1 PROOF OF PROPOSITION 1

Let

$$e_\theta(\mathbf{x}) := u_\theta(\mathbf{x}) - u^*(\mathbf{x}).$$

By the triangle inequality,

$$|e_\theta(\mathbf{x}_c)| \le |e_\theta(\mathbf{x}_p)| + |e_\theta(\mathbf{x}_c) - e_\theta(\mathbf{x}_p)|.$$

We first control the parent error at the current parameters. By the parent-value check and the stored parent error bound,

$$|e_\theta(\mathbf{x}_p)| = |u_\theta(\mathbf{x}_p) - u^*(\mathbf{x}_p)| \le \left|u_\theta(\mathbf{x}_p) - v_{\mathbf{x}_p}\right| + \left|v_{\mathbf{x}_p} - u^*(\mathbf{x}_p)\right| \le \epsilon + E_{\mathbf{x}_p}.$$

It remains to control the error increment from $\mathbf{x}_p$ to $\mathbf{x}_c$. By Assumption 2,

$$|e_\theta(\mathbf{x}_c) - e_\theta(\mathbf{x}_p)| \le C_{\mathrm{stab}} \|r_\theta\|_{L^\infty(\mathcal{N}_\delta(\mathbf{x}_p, \mathbf{x}_c))} + C_{\mathrm{reg}}\delta.$$

We now bound the residual on $\mathcal{N}_\delta(\mathbf{x}_p, \mathbf{x}_c)$ using the two endpoint checks. Let $\mathbf{z} \in \mathcal{N}_\delta(\mathbf{x}_p, \mathbf{x}_c)$ be arbitrary. By the endpoint-cover property of $\mathcal{N}_\delta(\mathbf{x}_p, \mathbf{x}_c)$, there exists an endpoint $\tilde{\mathbf{x}} \in \{\mathbf{x}_p, \mathbf{x}_c\}$ such that

$$\|\mathbf{z} - \tilde{\mathbf{x}}\|_\infty \le \delta/2.$$

If $\tilde{\mathbf{x}} = \mathbf{x}_p$, then **parent_res_ok** gives

$$|r_\theta(\tilde{\mathbf{x}})| < \hat{\tau}.$$

If $\tilde{\mathbf{x}} = \mathbf{x}_c$, then **child_ok** gives

$$|r_\theta(\tilde{\mathbf{x}})| < \hat{\tau}.$$

In either case, Assumption 1 gives

$$|r_\theta(\mathbf{z})| \le |r_\theta(\tilde{\mathbf{x}})| + L_r \|\mathbf{z} - \tilde{\mathbf{x}}\|_\infty < \hat{\tau} + \frac{1}{2} L_r \delta.$$

Since $\mathbf{z}$ was arbitrary,

$$\|r_\theta\|_{L^\infty(\mathcal{N}_\delta(\mathbf{x}_p, \mathbf{x}_c))} \le \hat{\tau} + \frac{1}{2} L_r \delta.$$

Combining the parent-error bound, the local propagation-stability bound, and the residual-neighborhood bound yields

$$|u_\theta(\mathbf{x}_c) - u^*(\mathbf{x}_c)| = |e_\theta(\mathbf{x}_c)| \le E_{\mathbf{x}_p} + \epsilon + C_{\mathrm{stab}} \left(\hat{\tau} + \frac{1}{2} L_r \delta \right) + C_{\mathrm{reg}}\delta,$$

which proves Proposition 1. □

A.2 A SUFFICIENT CONDITION FOR NETWORK-INDUCED RESIDUAL REGULARITY

Assumption 1 is stated at the residual level because our experiments include nonlinear PDE operators, for which the residual Lipschitz constant depends on both the operator and the current network state. For many PINN operators, the condition follows from smoothness and boundedness of the neural network derivatives.

For example, suppose the residual operator can be written locally as a finite composition of additions, multiplications, bounded coefficient functions, source terms, and derivatives of u_θ up to order n. If all involved derivatives $\partial^\alpha u_\theta$ are bounded and Lipschitz on $\mathcal{N}_\delta(\mathbf{x}_p, \mathbf{x}_c)$, and all coefficient functions and source terms are bounded and Lipschitz, then $r_\theta = \mathcal{F}[u_\theta]$ is Lipschitz on $\mathcal{N}_\delta(\mathbf{x}_p, \mathbf{x}_c)$. Indeed, sums of Lipschitz functions are Lipschitz, and products of bounded Lipschitz functions are Lipschitz:

$$|ab - a'b'| \le |a|\,|b - b'| + |b'|\,|a - a'|.$$

Applying this repeatedly to the finite expression defining $\mathcal{F}[u_\theta]$ gives a finite constant L_r such that

$$|r_\theta(\mathbf{z}_1) - r_\theta(\mathbf{z}_2)| \le L_r \|\mathbf{z}_1 - \mathbf{z}_2\|_\infty, \qquad \mathbf{z}_1, \mathbf{z}_2 \in \mathcal{N}_\delta(\mathbf{x}_p, \mathbf{x}_c).$$

B META-MODEL DETAILS AND SUPPLEMENTARY RESULTS FOR APPLICATION 1

B.1 META-MODEL ARCHITECTURE AND TRAINING

The meta-model h_ϕ is a two-layer MLP with 32 hidden units and $\tanh$ activation, trained on the 10-dimensional structural features described in Section 3.3. Inputs are standardized per feature. The output is the logarithm of the predicted threshold, ensuring positivity. Training loss is mean-squared error in log space with ℓ_2 regularization ($\lambda = 10^{-3}$). Labels are the 90-th percentile of $|r_\theta|$ at the *convergence moment*, defined as the first logged training step where the relative L^2 error first drops below 5×10^{-3} for three consecutive checkpoints. No training-dynamics features (loss trajectory, gradient statistics) are included, so the meta-model prediction depends only on the PDE, not on the architecture or optimizer.

The training set comprises 254 converged PINN runs across 8 PDE families (`advdiff_1d`, `burgers_1d`, `convection_1d`, `convection_2d`, `heat_1d`, `helmholtz_2d`, `poisson_1d`, `poisson_2d`), with 5 random seeds per coefficient setting. Coefficients used at training time and at evaluation time are disjoint by construction.

B.2 PER-RUN RESULTS ON THE TRAINED-FAMILY SUITE

Table 6 reports the full per-run breakdown for the 32 trained-family configurations at $S = 5$ and a 50,000-step budget. "interp" denotes coefficients inside the meta-model's training range; "extrap" denotes coefficients outside that range.

PDE	Coeff	Type	rel-L^2	$\hat{\tau}$	Steps	Outcome
advdiff_1d	0.008	interp	9.73e-03	8.43e-03	50,000	Correct Non-conv.
advdiff_1d	0.03	interp	1.53e-03	8.44e-03	42,000	Successful Stop
advdiff_1d	0.15	interp	1.89e-03	8.50e-03	50,000	Miss Det.
advdiff_1d	0.5	extrap	2.41e-02	8.66e-03	50,000	Correct Non-conv.
burgers_1d	0.008	interp	1.55e-03	1.16e-02	50,000	Miss Det.
burgers_1d	0.03	interp	4.86e-04	1.17e-02	50,000	Miss Det.
burgers_1d	0.15	interp	4.55e-03	1.18e-02	29,000	Successful Stop
burgers_1d	0.5	extrap	1.64e-02	1.21e-02	14,000	False Stop
convection_1d	3.0	interp	1.43e-02	1.36e-02	50,000	Correct Non-conv.
convection_1d	8.0	interp	2.09e-02	2.17e-02	50,000	Correct Non-conv.
convection_1d	25.0	interp	7.77e-01	5.94e-02	50,000	Correct Non-conv.
convection_1d	80.0	extrap	1.00e+00	4.31e-02	50,000	Correct Non-conv.
convection_2d	0.75	interp	1.02e-02	8.99e-03	50,000	Correct Non-conv.
convection_2d	1.25	interp	1.70e-02	9.44e-03	50,000	Correct Non-conv.
convection_2d	4.0	interp	3.58e-02	1.29e-02	50,000	Correct Non-conv.
convection_2d	7.0	extrap	9.07e-02	1.82e-02	50,000	Correct Non-conv.
heat_1d	0.03	interp	2.03e-03	8.48e-03	50,000	Miss Det.
heat_1d	0.08	interp	4.42e-03	8.50e-03	39,000	Successful Stop.
heat_1d	0.3	interp	1.62e-03	8.55e-03	28,000	Successful Stop
heat_1d	2.0	extrap	4.53e-02	9.11e-03	50,000	Correct Non-conv.
helmholtz_2d	0.25	extrap	1.59e-03	4.71e-02	24,000	Successful Stop
helmholtz_2d	1.5	interp	2.95e-03	4.48e-02	25,000	Successful Stop
helmholtz_2d	4.0	interp	5.62e-02	3.12e-02	50,000	Correct Non-conv.
helmholtz_2d	9.0	extrap	8.35e-02	4.24e-01	50,000	Correct Non-conv.
poisson_1d	0.75	interp	1.67e-04	1.99e-02	20,000	Successful Stop
poisson_1d	1.25	interp	3.16e-04	2.49e-02	13,000	Successful Stop
poisson_1d	3.5	interp	3.87e-03	1.89e-01	40,000	Successful Stop
poisson_1d	6.0	extrap	1.61e-02	5.42e-01	20,000	False Stop
poisson_2d	0.75	interp	3.13e-03	4.09e-02	27,000	Successful Stop
poisson_2d	1.25	interp	1.45e-03	5.62e-02	50,000	Missed Det.
poisson_2d	2.5	interp	2.04e-02	1.74e-01	50,000	Correct Non-conv.
poisson_2d	4.0	extrap	2.92e-02	6.19e-01	50,000	Correct Non-conv.

Table 6: Per-run results for the 32 trained-family configurations at $S = 5$.

B.3 UNSEEN-FAMILY SUITE AND EXTENDED-BUDGET RUN FOR `wave_1d`

Table 7 reports the per-run breakdown on the 12-configuration unseen-family suite. The step budget is 50,000 for `nonlinear_poisson_2d` and `reaction_diffusion_1d`, and 80,000 for `wave_1d` (the underlying PINN takes longer on the wave equation, for reasons unconnected to the threshold).

PDE	Coeff	rel-L^2	$\hat{\tau}$	Steps	Outcome
nonlinear_poisson_2d	0.1	2.38e-03	3.58e-02	38,000	Successful Stop
nonlinear_poisson_2d	0.5	3.68e-03	3.61e-02	30,000	Successful Stop
nonlinear_poisson_2d	1.0	2.39e-02	3.64e-02	28,000	False Stop
nonlinear_poisson_2d	3.0	4.64e-03	3.93e-02	29,000	Successful Stop
reaction_diffusion_1d	0.01	5.73e-04	1.05e-02	47,000	Successful Stop
reaction_diffusion_1d	0.05	9.85e-04	1.05e-02	49,000	Successful Stop
reaction_diffusion_1d	0.1	1.03e-03	1.06e-02	50,000	Missed Det.
reaction_diffusion_1d	0.5	3.98e-03	1.08e-02	17,000	Successful Stop
wave_1d	0.5	9.38e-04	1.16e-02	40,000	Successful Stop
wave_1d	1.0	7.06e-04	1.20e-02	80,000	Missed Det.
wave_1d	2.0	1.54e-03	1.43e-02	80,000	Missed Det.
wave_1d	4.0	1.55e-02	4.05e-02	80,000	Correct Non-conv.

Table 7: Per-run results for the 12 unseen-family configurations at $S = 5$.

B.4 Detective progression visualization on convection $\beta = 50$

Figure 2 visualizes how the detective wavefront expands during training on the one-dimensional convection problem $u_t + 50\,u_x = 0$ with the predicted threshold $\hat{\tau} = 0.0570$. Each column corresponds to a snapshot at training step $\{0\text{k}, 30\text{k}, 50\text{k}, 80\text{k}, 120\text{k}\}$; the top row shows the admitted set $\mathcal{W}$ (blue) and the active detection shell $\mathcal{D}$ (red) over the (t, x) grid, the middle row shows the PINN prediction u_θ, and the bottom row shows the pointwise absolute error. Coverage grows from 1% to 79% as the predicted PINN learns the solution, illustrating that the wavefront automatically advances from the supervised initial slice $\{t = 0\}$ into the interior in step with the underlying solution accuracy: the maximum pointwise error decreases by an order of magnitude over the same window, and the relative L^2 error drops from 1.29 to 0.043.

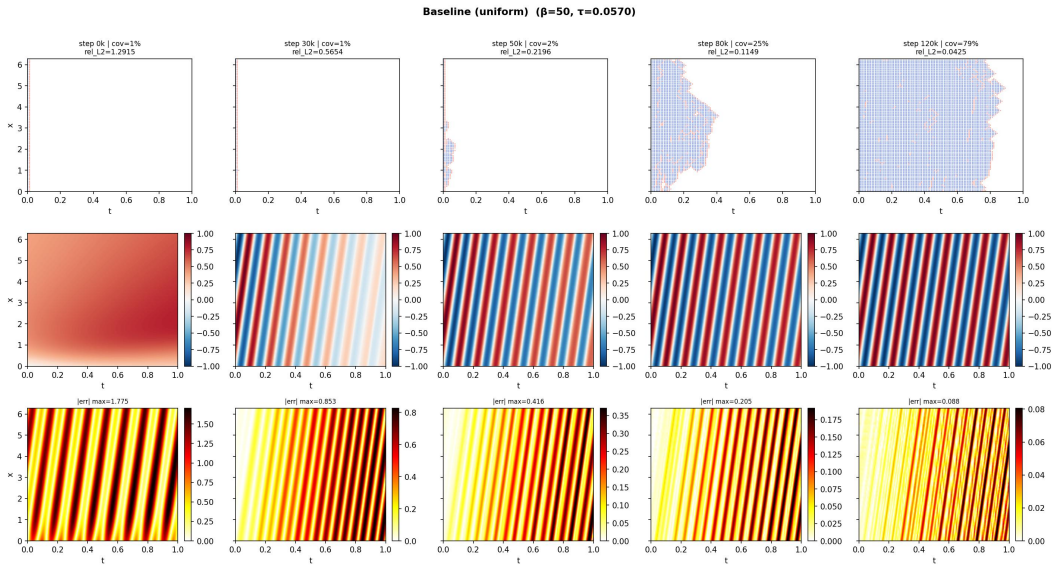

Figure 2: Detective progression on convection $\beta = 50$ with predicted threshold $\hat{\tau} = 0.0570$. Top: admitted set (blue) and detection shell (red); middle: PINN prediction; bottom: $|u_\theta - u^\star|$. Snapshots at training steps 0k, 30k, 50k, 80k, 120k. Coverage grows from 1% to 79% as the boundary-anchored frontier advances into the interior.

C Application 2 experimental details

Convection benchmark protocol. For the time-dependent benchmark, we use the one-dimensional convection equation $u_t + \beta u_x = 0$ with $\beta = 30$ on $(x, t) \in [0, 2\pi] \times [0, 1]$ and initial condition

$u(x, 0) = \sin(x)$. All methods in Table 4 use the same architecture-parity protocol as the published baselines: a 4×50 MLP with tanh activation, the same input feature map, and AdamW with constant learning rate 10^{-3}. Method abbreviations are defined in the caption of Table 4.

Compute budget vs. published baselines. Our internal methods use a strictly smaller optimization budget than the strongest published baselines. Each run consists of 3×10^5 AdamW steps followed by a single short L-BFGS pass of 500 steps. By contrast, the ensemble pseudo-label baseline of (Innerebner et al., 2025) runs Algorithm 1 for 100 outer iterations, each consisting of roughly 5,000 Adam steps and up to 20,000 L-BFGS steps, totaling on the order of 5×10^5 Adam steps and up to 2×10^6 L-BFGS steps per run. Thus, the comparison in Table 4 is favorable to the published baselines on both phases of optimization: they receive more first-order steps and dramatically more L-BFGS refinement, yet the detective-driven frontier with EMA buffer reaches a lower final relative L^2 error.

Static Helmholtz protocol. For the static benchmark, we solve the Helmholtz equation $\nabla^2 u + k^2 u = f$ on $[0, 1]^2$ for $k \in \{1, 10\}$. All internal methods use the same 4×50 MLP and the same training budget of 5×10^4 Adam steps. Since the problem has no temporal coordinate, the detective-driven active domain is defined spatially: the detected set is initialized on the supervised boundary, and the active collocation region is the union of cells within an M-shell margin of the current boundary-seeded wavefront. The same EMA buffer mechanism is used as in the time-dependent benchmark, except that boundary anchors are kept fixed and are not updated by EMA. We report mean final relative L^2 error across three seeds.

Controlled comparison with residual-adaptive sampling. We additionally compare ordinary training (Vanilla), residual-based adaptive distribution (RAD) (Wu et al., 2023) with $k = c = 1$, detective-guided training, and detective-guided training with the buffer on Helmholtz at $k = 1$. Within this comparison, all methods share the 4×50 tanh network, seeds $\{0, 42, 123\}$, seed-wise initialization, base loss weights, batch sizes, optimizer, update budget, and evaluation grid. The common budget is 50,000 AdamW steps, including a shared seed-wise 10,000-step full-domain warm-up, with no L-BFGS. Method-specific differences are residual-based resampling, detective-guided training support, and the optional buffer loss. Table 8 reports this additional controlled comparison separately from Table 5.

Table 8: Additional controlled comparison on two-dimensional Helmholtz at $k = 1$. Values are final full-domain relative L_2 errors, reported as mean $\pm$ sample standard deviation over three seeds.

Method	Relative L_2
Vanilla	0.00265 ± 0.00181
RAD	0.00742 ± 0.00986
Detective-only	$\mathbf{0.00145} \pm 0.00055$
Detective + buffer	0.00175 ± 0.00133

Buffer-loss weight ablation. The buffer-loss weight λ_{buf} is the only hyperparameter introduced by Application 2 beyond $\hat{\tau}$. We sweep it on the convection $\beta = 30$ benchmark with the same 4×50 MLP and report the mean final relative L^2 across three seeds (Table 9). The trend is monotonic in the regime tested: smaller λ_{buf} produces lower final error, and $\lambda_{\text{buf}} = 0.5$ is the lowest value we tested. Larger weights drive the network toward the EMA's moving average rather than letting the residual loss continue to refine the solution, in line with the precision-asymmetry argument of Section 4.2; the optimum is robustly small. The main text uses $\lambda_{\text{buf}} = 0.5$ throughout.

Additional active-training parameter sensitivity. We additionally run a single-seed sensitivity pilot on convection $\beta = 30$ with 300,000 AdamW updates and no L-BFGS. Seed 42 is used for both parameter selection and reporting; Vanilla is not retuned, so this is a sensitivity study rather than a held-out fairness comparison. Table 10 reports the parameter-grid ranges and local checks. The M and K checks vary one parameter at a time around the detective-only setting $(\hat{\tau}, \epsilon, K, M) = (0.064, 0.04, 2000, 5)$. The α and ϵ checks use the detective-plus-buffer setting $(\hat{\tau}, \epsilon, \lambda_{\text{buf}}, \alpha, K, M) = (0.128, 0.02, 0.2, 0.1, 2000, 5)$ as their center.

λ_{buf}	Mean rel-L^2	Std
0.5	$\mathbf{6.94 \times 10^{-3}}$	1.82×10^{-3}
1.0	8.69×10^{-3}	2.05×10^{-3}
3.0	1.09×10^{-2}	1.97×10^{-3}
5.0	1.09×10^{-2}	1.34×10^{-3}

Table 9: Buffer-loss weight ablation for detective + EMA buffer on convection $\beta = 30$, mean final relative L^2 across three seeds.

Table 10: Single-seed active-training sensitivity at the 300,000-step AdamW endpoint. Grid rows give error ranges; slash-separated errors follow the order of the tested settings. The buffer-weight row reports the minimum over the searched $\hat{\tau}$ values for each weight.

Sensitivity check	Tested settings	Relative L_2
Detective-only grid	$\hat{\tau} = 0.064$–0.192; $\epsilon = 0.01$–0.04	0.01704–0.021996
Detective + buffer grid	$\hat{\tau} = 0.096$–0.192; $\lambda_{\text{buf}} = 0.02$–$0.20$	0.01756–0.02329
Buffer weight λ_{buf}	0.02/0.05/0.1/0.2	0.01873/0.01905/ 0.01922/0.01756
Moving-average rate α	0.03/0.1/0.3	0.01780/0.01756/ 0.01933
Value drift ϵ	0.01/0.02/0.04	0.01968/0.01756/ 0.02031
Look-ahead M	2/5/10	0.01737/0.01704/ 0.01808
Cadence K	1000/2000/4000	0.01980/0.01704/ 0.01850

Every setting in the displayed ranges outperforms the same-protocol Vanilla result of 0.02676, supporting stability across these tested ranges on this seed. The response is not monotone: in a separate one-factor check of the default detective-plus-buffer configuration, $\hat{\tau} = 0.032$ stalls at 78.6% coverage with 0.18899 error at the 300,000-step AdamW endpoint. These results characterize local sensitivity; they do not establish cross-seed or cross-family robustness.

D Implementation details and reproducibility

We list the full hyperparameter configuration used throughout the experiments to support reproduction. All experiments are implemented in JAX (Bradbury et al., 2018) with Optax for the optimiser and run on a workstation equipped with two NVIDIA GeForce RTX 4090 GPUs (24 GB each); each individual run uses a single GPU. The detective algorithm runs on the host with NumPy bookkeeping (grid-based shell construction, vectorised parent-scatter via `np.unique` inverse) and offloads only the residual and prediction batches to the GPU through a JIT-compiled forward function, so the same code reproduces on a single-GPU machine without modification. The shared algorithmic constants are: shell spacing $\delta = 10^{-2}$ (giving a 101-point grid per axis on the unit cube), value-drift tolerance $\epsilon = 10^{-2}$, well-set supervision batch 512, promotion interval $K = 2000$ optimiser steps, shell budget $S = 5$ promotion rounds per cycle, stall budget $N_{\text{stall}} = 20000$ steps, and target tolerance $\varepsilon = 5 \times 10^{-3}$ for meta-model labels (with $\varepsilon = 5 \times 10^{-3}$ used as the convergence contract reported in §4.1). The PINN is a tanh-activated MLP with Fourier feature inputs; per-PDE settings (depth, width, learning rate, total steps, interior batch, boundary batch, boundary loss weight λ_b) are: `poisson_1d` (5×128, lr 10^{-3}, 12,000 steps, $1024/256$, λ_b=10); `convection_1d`, `heat_1d` (5×128, 20,000 steps, $2048/512$, λ_b=10); `burgers_1d`, `advdiff_1d` (5×128, 25,000 steps, $2048/512$, λ_b=10); `poisson_2d`, `convection_2d` (5×192, 25,000 steps, $2048/768$, λ_b=10); `helmholtz_2d` (5×192, 20,000 steps, $2048/768$, λ_b=10); `wave_1d` (5×256, 80,000 steps, $2048/512$, λ_b=20, cosine learning-rate decay to $\alpha = 10^{-2}$, velocity-IC penalty weight 20); `nonlinear_poisson_2d` (5×192, 30,000 steps, $2048/768$, λ_b=10); `reaction_diffusion_1d` (5×128, 25,000 steps, $2048/512$, λ_b=10). All Application 1 experiments enforce a global step budget of $\max(\text{per-PDE steps}, 50{,}000)$

in addition to the per-PDE specification above. Per-config wall-clock time on a single RTX 4090 ranges from ~ 5 minutes for the smallest in-distribution configurations to ~ 13 minutes for `wave_1d` at the $80{,}000$-step budget; the full 44-configuration sweep used in Tables 1 and 3 completes in approximately 4 hours on a single GPU.

Application 2 hyperparameters. For the convection $\beta = 30$ benchmark of Section 4.2 we follow the architecture-parity protocol of (Innerebner et al., 2025). The base model is a 4-layer plain MLP with width 50 and $\tanh$ activation (approximately $7{,}800$ parameters); inputs are mapped through the same Fourier feature embedding used by (Innerebner et al., 2025) to enforce the periodic boundary condition. We use the AdamW optimiser with a constant learning rate of 1×10^{-3} and zero weight decay, train for 3×10^5 Adam steps, and run a 500-iteration L-BFGS finisher applied uniformly to every variant in the ablation; each batch samples $2{,}000$ residual collocation points and 200 initial/boundary points. The detective wavefront uses a 100×100 uniform grid over $[0, 2\pi] \times [0, 1]$, a meta-model-predicted residual threshold $\hat{\tau} = 6.4 \times 10^{-2}$, value-drift tolerance $\epsilon = 10^{-2}$, shell budget $S = 5$ promotion rounds per cycle, and a detection cycle every $2{,}000$ optimiser steps. The buffer-loss weight is $\lambda_{\text{buf}} = 0.5$ (chosen by the ablation in Table 9); the EMA buffer uses decay $\alpha = 0.1$, applied once per detection cycle to all admitted points except boundary anchors, which are kept at the exact initial-condition values $\sin(x)$. The collocation domain is clipped to a margin around the wavefront frontier $t_{\text{front}} + M\Delta t$ with $M = 5$ shells; whenever the frontier advances, residual collocation points are resampled inside the new active region. We report the mean and standard deviation of the final relative L^2 error across three random seeds $\{0, 42, 123\}$. For the static Helmholtz benchmark of Section 4.2, the architecture is the same 4×50 MLP, the optimiser is AdamW with constant learning rate 10^{-3}, the budget is 5×10^4 Adam steps with no L-BFGS finisher, and the wavefront expands outward from $\partial[0, 1]^2$ in L^∞ rather than along time; all other detective parameters match the convection setup.