CHEBYSHEV-AUGMENTED ONE-SHOT TRANSFER LEARNING FOR PINNS ON NONLINEAR DIFFERENTIAL EQUATIONS

Yiqi Rao
Harvard University
herryrao@g.harvard.edu

Pavlos Protopapas
Harvard University
pprotopapas@g.harvard.edu

ABSTRACT

Physics-Informed Neural Networks (PINNs) offer a flexible paradigm for solving differential equations by embedding governing laws into the training objective. A persistent limitation is instance specificity: standard PINNs typically require retraining for each new forcing term, boundary/initial condition, or parameter setting. One-shot transfer learning (OTL) addresses this bottleneck for linear operators by freezing a pretrained latent representation and computing optimal output weights in closed form, but for nonlinear problems closed-form adaptation is generally unavailable because the loss is nonconvex in the output layer.

In this paper we substantially broaden the class of nonlinearities amenable to one-shot PINN transfer by combining OTL with Chebyshev polynomial surrogates. We approximate general smooth weakly nonlinear terms by truncated Chebyshev expansions over a prescribed solution range, yielding a polynomial nonlinearity that can be handled by a perturbative decomposition into linear subproblems. A multi-head PINN learns a reusable latent space associated with the dominant linear operator; at test time, solutions to new instances are obtained via a sequence of closed-form linear solves in the output layer, without retraining the network body.

We provide a unified derivation of the framework for ODEs and PDEs and demonstrate accuracy and fast online adaptation on nonlinear benchmarks, including non-polynomial and singular ODE nonlinearities as well as a reaction–diffusion PDE with saturating kinetics, demonstrating the method's utility in many-query regimes.

1 INTRODUCTION

PINNs have become a widely used tool for solving forward and inverse differential-equation problems by minimizing residuals of the governing equations alongside boundary/initial conditions (Lagaris et al., 1998; Raissi et al., 2019; Karniadakis et al., 2021; Desai et al., 2022). Despite their flexibility, a core drawback remains: they are typically trained separately for each problem instance. Even when two problems share the same operator but differ only in forcing or boundary data, standard PINN approaches generally require a new optimization run, limiting their utility in many-query, real-time, or interactive settings where rapid adaptation is required.

Transfer learning for PINNs aims to amortize training across a family of related problems. A particularly strong form is one-shot transfer learning (OTL): for linear ODE/PDE families, a shared network body can be frozen and the output-layer weights computed in closed form, yielding fast adaptation with a single matrix inversion (Desai et al., 2022; Lei et al., 2023). However, the presence of nonlinear terms renders the optimization problem nonconvex in the output weights, precluding closed-form adaptation in general.

A promising recent direction is to restore linear solvability via perturbative decomposition. For nonlinearities that appear as a small polynomial perturbation, the nonlinear problem can be expressed as a sequence of linear subproblems whose right-hand sides depend on lower-order solutions (Lei et al., 2023; Auroy & Protopapas, 2025; Alexandrino et al., 2026). Each linear subproblem can then be

solved by one-shot weight updates. While effective, this framework is restricted to weak polynomial nonlinearities and to regimes where the perturbative expansion remains valid.

In this work, we propose to relax the restrictive polynomial assumption by approximating weak nonlinear terms with truncated Chebyshev polynomial expansions (Trefethen, 2019; Boyd, 2001; Mason & Handscomb, 2002). This transforms a broad class of smooth nonlinearities into polynomial surrogates, enabling perturbative continuation and one-shot transfer learning. Concretely, we:

- construct Chebyshev surrogates for general weak nonlinear terms, including non-polynomial response functions, over prescribed solution ranges;
- derive a perturbative reduction into a sequence of linear subproblems, each solved via one-shot output-layer updates using a pretrained multi-head PINN body;
- validate the proposed framework on nonlinear ODE and PDE benchmarks, demonstrating accuracy and fast online adaptation.

The proposed method targets many-query regimes in which the dominant linear operator is fixed and nonlinear effects are moderate over bounded solution ranges.

The remainder of the paper is organized as follows: Section 3 introduces the problem setting, Section 4 presents the Chebyshev-augmented one-shot framework, Section 5 reports experimental results on nonlinear ODE and PDE benchmarks, Section 6 concludes with a discussion of limitations and future directions, and Section 7 concludes the work.

2 Related work

PINNs and failure modes. PINNs trace back to early neural-network solvers for differential equations (Lagaris et al., 1998) and have been popularized by modern autograd-based formulations (Raissi et al., 2019; Karniadakis et al., 2021). Known failure modes include spectral bias and poor extrapolation (Krishnapriyan et al., 2021; Xu et al., 2019), motivating improved architectures, sampling, and training strategies.

Multi-instance training and transfer. Bundle-style and multi-head training amortize computation by training on multiple instances simultaneously (Flamant et al., 2020; Zou & Karniadakis, 2023). One-shot transfer learning for linear families provides closed-form adaptation by reducing inference to a least-squares solve in the output layer (Desai et al., 2022). Operator-learning methods such as DeepONet and Fourier/Neural Operators provide alternative amortization mechanisms, learning maps between function spaces (Lu et al., 2021; Li et al., 2021; Kovachki et al., 2023). Unlike operator-learning approaches, which aim to learn mappings between function spaces, one-shot transfer methods retain explicit equation structure and target fast adaptation under a fixed operator.

Perturbative PINNs for nonlinear problems. Perturbative OTL frameworks recover closed-form adaptation for certain nonlinear ODEs/PDEs by expanding solutions in a small parameter and solving a sequence of linear problems (Lei et al., 2023; Auroy & Protopapas, 2025; Alexandrino et al., 2026). Our method retains the one-shot adaptation principle but uses Chebyshev approximation to broaden the nonlinearity class beyond polynomials.

Chebyshev approximation and spectral methods. Chebyshev polynomials provide near-minimax polynomial approximations with strong error guarantees for smooth analytic functions and form the backbone of many spectral methods (Trefethen, 2019; Boyd, 2001). We leverage these approximation properties not for discretization, but to construct polynomial surrogates that make nonlinear one-shot transfer tractable.

3 Problem setting

We study one-shot transfer for families of weak nonlinear differential equations in a perturbative regime, targeting many-query settings. Here, a query refers to solving the same differential operator under new forcing terms, boundary conditions, or parameters. Let s denote the independent

variables: $s = t$ for ODEs and $s = (x, t)$ for PDEs, where $x \in \Omega_x \subset \mathbb{R}^d$ and $t \in [0, T]$. We write $\Omega = \Omega_x \times [0, T]$ for space–time (with $\Omega = [0, T]$ in the ODE case) and $\partial\Omega$ for the collection of boundary/initial sets. For clarity we present a scalar unknown $u : \Omega \to \mathbb{R}$; extensions to vector-valued systems follow component-wise without conceptual changes.

We consider equation families of the form

$$\mathcal{D}u(s) + \varepsilon \mathcal{N}\big(u(s)\big) = f(s; \eta), \qquad s \in \Omega, \tag{1}$$

subject to initial/boundary constraints

$$\mathcal{B}u(s) = b(s; \eta), \qquad s \in \partial\Omega, \tag{2}$$

where $\mathcal{D}$ is a fixed linear differential operator that defines the dominant dynamics and is shared across the family, $\mathcal{N}$ is a pointwise nonlinearity which may be non-polynomial, $\mathcal{B}$ collects the boundary/initial operators (Dirichlet/Neumann and initial conditions), and η indexes instance-dependent inputs such as forcing terms and boundary/initial data.

The scalar ε plays the role of a perturbation strength parameter. In some applications, ε is a physical coefficient already present in the model. When the original equation does not include such a multiplier, we treat ε as a formal homotopy parameter; we solve the family $\mathcal{D}u + \varepsilon \mathcal{N}(u) = f$ and evaluate the approximation. Equivalently, when admissible, one may induce a small effective ε by rescaling the unknown and/or coefficients so that the nonlinear contribution is moderate over the solution regime of interest.

Our objective is to design a method with a clear offline/online split. Offline, we train once to learn a latent representation associated with the linear operator $\mathcal{D}$ from a bundle of linear tasks. Online, for each new nonlinear instance equation 1–equation 2 (i.e., each new query), we compute an approximation of u without retraining the network body or modifying the learned latent representation. The online computation should reduce to a small number of closed-form linear solves in the output layer (one-shot transfer), rather than iterative gradient-based optimization.

4 METHOD

4.1 OVERVIEW

As described above, our framework follows a clear offline/online split: we learn an operator-aware representation offline once, then solve new nonlinear instances online via a small number of closed-form output-layer updates.

In the offline stage, we train a multi-head PINN on a bundle of linear tasks that share the same linear operator $\mathcal{D}$. This learns a shared feature map $H(s) \in \mathbb{R}^h$ (the network body), which captures operator-dependent structure. After training, we freeze H. With H fixed, any linear instance

$$\mathcal{D}u = g, \qquad \mathcal{B}u = h$$

can be solved by a convex least-squares solve over the output-layer weights. The corresponding system matrix used in the fit depends only on $\mathcal{D}$ and $\mathcal{B}$, so it can be assembled and factorized once and reused across all the linear tasks.

In the online stage, given a nonlinear instance equation 1–equation 2 (i.e., a new query), we first approximate the pointwise nonlinearity $\mathcal{N}(u)$ by a truncated Chebyshev expansion over a prescribed solution range $u \in [u_{\min}, u_{\max}]$. This produces a surrogate map $\mathcal{N}_m(u)$ represented in the Chebyshev basis whose coefficients are computed by Gauss–Chebyshev quadrature and whose evaluation uses the standard three-term recurrence (details in Appendix A.1.1).

We then construct a truncated perturbation expansion in ε, $u(s; \varepsilon) \approx \sum_{j=0}^{p} \varepsilon^j u_j(s)$, substitute it into the surrogate equation $\mathcal{D}u + \varepsilon \mathcal{N}_m(u) = f$, and match powers of ε. This yields $p + 1$ linear subproblems sharing the operator $\mathcal{D}$; the right-hand side at order j depends only on the previously computed lower-order terms $\{u_0, \ldots, u_{j-1}\}$. We impose evenly-splitted boundary/initial constraints on every order, so that the truncated series satisfies the original boundary/initial constraints up to truncation error. Each linear subproblem is then solved by one-shot adaptation using the frozen feature map H.

Section 4.2 states the Chebyshev surrogate used in the online stage, and Section 4.3 presents the resulting perturbative linear recursion. Section 4.4 describes the multi-head pretraining procedure used to learn a reusable operator-aware feature map for $\mathcal{D}$, while Section 4.5 presents the closed-form one-shot update that computes output weights. Finally, Section 4.6 summarizes the complete online pipeline for solving nonlinear instances by sequential one-shot solves across perturbation orders and reconstructing u from the truncated series.

4.2 CHEBYSHEV SURROGATE

The goal of this step is to replace a general nonlinear term by a polynomial surrogate on a bounded solution range, enabling perturbative decomposition and closed-form adaptation in later stages.

We approximate the pointwise nonlinearity $\mathcal{N}(u)$ by a truncated Chebyshev series on a prescribed range $u \in [u_{\min}, u_{\max}]$ by mapping u affinely to $\xi \in [-1, 1]$ and expanding in Chebyshev polynomials of the first kind. Concretely, we use the surrogate

$$\mathcal{N}(u) \approx \mathcal{N}_m(u) := \sum_{\ell=0}^{m} c_\ell \, T_\ell(\Phi(u)), \tag{3}$$

where $\Phi : [u_{\min}, u_{\max}] \to [-1, 1]$ is the standard affine map. The coefficients $\{c_\ell\}$ are computed by weighted Chebyshev projections, approximated in practice by Gauss–Chebyshev quadrature; evaluation uses the standard three-term recurrence. Full details of the affine map, coefficient formulas, quadrature rules, and stable evaluation are provided in Appendix A.1.1.

4.3 PERTURBATIVE EXPANSION AND LINEAR SUBPROBLEM RECURSION

We solve the surrogate problem obtained by replacing $\mathcal{N}$ with $\mathcal{N}_m$ in equation 1:

$$\mathcal{D}u(s) + \varepsilon \mathcal{N}_m\big(u(s)\big) = f(s;\eta), \qquad \mathcal{B}u = b(\cdot;\eta). \tag{4}$$

We seek a truncated series in the same perturbation parameter ε, which we treat as small in the regime of interest, of the form

$$u(s;\varepsilon) \approx \sum_{j=0}^{p} \varepsilon^j u_j(s). \tag{5}$$

Substituting equation 5 into equation 4 requires expanding $\mathcal{N}_m(u(s;\varepsilon))$ in powers of ε. To preserve numerical stability, we perform this expansion directly in the Chebyshev basis using a Chebyshev recurrence lifted to truncated ε-series; the explicit forcing construction is given in Appendix A.1.2. Matching powers of ε yields a sequence of linear problems with shared operator $\mathcal{D}$:

$$\mathcal{D}u_0(s) = f(s;\eta), \qquad \mathcal{B}u_0 = b_0(\cdot;\eta), \tag{6}$$

$$\mathcal{D}u_j(s) = -\mathcal{G}_{j-1}(s), \qquad \mathcal{B}u_j = b_j(\cdot;\eta), \qquad j = 1, \ldots, p. \tag{7}$$

Here, the boundary/initial data $\{b_j\}$ are chosen so that the truncated series $\sum_{j=0}^{p} \varepsilon^j u_j$ satisfies the original constraints up to truncation error, for example by evenly splitting the constraints across orders. Thus the nonlinear problem reduces to $p + 1$ linear subproblems governed by the same operator $\mathcal{D}$. Each linear solve in equation 6–equation 7 is carried out by one-shot adaptation with the frozen feature map H (Section 4.5).

4.4 MULTI-HEAD PRETRAINING ON LINEAR BUNDLES

All subproblems in equation 6–equation 7 share the same dominant linear operator $\mathcal{D}$, so we learn a reusable feature map only once. This offline training cost is amortized across all subsequent linear solves.

We parameterize the solution with a shared network body $H_\theta(s)$ and K linear heads. A clear illustration of the multi-head architecture is shown in Appendix A.1.3 (Figure 4). The shared body is intended to capture operator-specific structure common to all tasks, while the heads account for task-specific forcing and boundary data. In settings where the linear part contains higher-order

derivatives, we use the standard first-order reformulation with auxiliary variables (details and an example are given in Appendix A.1.3).

In the PDE case, it is convenient to view the frozen feature map as a matrix $\mathbf{H}_\theta(s) \in \mathbb{R}^{2\times h}$ whose rows correspond to the two components of the first-order state. Each head uses a single weight vector $W_k \in \mathbb{R}^h$ that is shared across the components of the state vector, ensuring a consistent linear combination of the feature map for both the primary and auxiliary variables.

$$\hat{\mathbf{u}}^{(k)}(s) = \mathbf{H}_\theta(s)\, W_k, \qquad \text{i.e.,} \quad \hat{u}^{(k)}(s) = H_{\theta,u}(s)^\top W_k,\ \hat{y}^{(k)}(s) = H_{\theta,y}(s)^\top W_k, \qquad k = 1, \dots, K. \tag{8}$$

We train the shared parameters θ and all head weights $\{W_k\}_{k=1}^K$ by minimizing a weighted sum of physics-informed residual losses and constraint losses (and an optional data loss when manufactured solutions are available). The explicit task specification and loss definitions are provided in Appendix A.1.3.

4.5 One-shot head solve for any linear subproblem

With the feature map $\mathbf{H}(s) = \mathbf{H}_\theta(s)$ frozen, each linear subproblem sharing $\mathcal{D}$ is solved by optimizing only the output weights, following the one-shot transfer construction in (Desai et al., 2022) and its perturbative extensions (Lei et al., 2023; Auroy & Protopapas, 2025). We approximate the state by

$$\hat{\mathbf{u}}(s) = \mathbf{H}(s)W,$$

in particular the primary field is $\hat{u}(s) = H_u(s)^\top W$.

Let $\{s_n\}_{n=1}^{N_r} \subset \Omega$ and $\{\bar{s}_n\}_{n=1}^{N_b} \subset \partial\Omega$ be the fixed interior and constraint sampling sets. These sampling sets are shared across all linear subproblems and perturbation orders. Let $\mathbf{A}_r$ denote the stacked matrix collecting $(\mathcal{D}\mathbf{H})(s_n)$ over interior points, and let $\mathbf{A}_b$ denote the stacked matrix collecting $(\mathcal{B}H_u)(\bar{s}_n)^\top$ over constraint points; explicit stacking and dimensions are given in Appendix A.1.4. The corresponding targets are stacked as

$$\mathbf{f}^* := \big[\mathbf{f}^*(s_n)\big]_{n=1}^{N_r}, \qquad \mathbf{b}^* := \big[b^*(\bar{s}_n)\big]_{n=1}^{N_b}.$$

These targets encode the forcing and constraint data for the specific linear instance being solved. The one-shot head is obtained by minimizing the same quadratic objective used in training (with θ fixed):

$$\min_W \ \frac{w_{\mathrm{pde}}}{N_r}\big\|\mathbf{A}_r W - \mathbf{f}^*\big\|_2^2 + \frac{w_{\mathrm{bc}}}{N_b}\big\|\mathbf{A}_b W - \mathbf{b}^*\big\|_2^2. \tag{9}$$

This objective is convex in W and admits a unique solution under standard full-rank conditions; thus we have the closed-form update

$$W^* = \mathbf{M}^{-1}\mathbf{q}^*, \tag{10}$$

with

$$\mathbf{M} = \frac{w_{\mathrm{pde}}}{N_r}\mathbf{A}_r^\top\mathbf{A}_r + \frac{w_{\mathrm{bc}}}{N_b}\mathbf{A}_b^\top\mathbf{A}_b, \qquad \mathbf{q}^* = \frac{w_{\mathrm{pde}}}{N_r}\mathbf{A}_r^\top\mathbf{f}^* + \frac{w_{\mathrm{bc}}}{N_b}\mathbf{A}_b^\top\mathbf{b}^*. \tag{11}$$

For fixed $\mathcal{D}$, $\mathcal{B}$, frozen $\mathbf{H}$, and fixed sampling sets, $\mathbf{M}$ is constant across instances and across perturbation orders. Consequently, $\mathbf{M}$ depends only on the linear operator, the constraint type, and the sampling strategy. Hence $\mathbf{M}^{-1}$ can be precomputed once and reused, and each new linear solve reduces to constructing $\mathbf{q}^*$. This reuse is the key to achieving fast online adaptation in many-query settings.

4.6 Online solve for nonlinear instances

Given a nonlinear instance equation 1–equation 2, the online stage performs a sequence of one-shot linear solves while keeping the pretrained feature map $\mathbf{H}$ fixed. No gradient-based retraining is performed in the online stage. We first select a working range $[u_{\min}, u_{\max}]$ and construct the Chebyshev surrogate $\mathcal{N}_m$ as in Section 4.2. We then compute the perturbation coefficients $\{u_j\}_{j=0}^p$ by solving the linear subproblems equation 6–equation 7 sequentially.

For each order j, the linear subproblem provides an interior forcing $g_j(s)$ (with $g_0 = f$ and, for $j \geq 1$, g_j determined by the Chebyshev-based forcing construction in Appendix A.1.2) and a constraint

target b_j consistent with the constraint splitting used in equation 6–equation 7. Both the forcing and constraint targets are assembled deterministically from previously computed lower-order solutions. In the first-order PDE example (Appendix A.1.3), we lift the scalar forcing to a vector target by

$$\mathbf{f}^{(j)}(s) = \begin{bmatrix} g_j(s) \\ 0 \end{bmatrix},$$

and apply the constraint operator only to the primary component u_j. With the fixed matrices $\mathbf{A}_r, \mathbf{A}_b$ (Appendix A.1.4), we form the corresponding stacked targets $\mathbf{f}^{(j)}$ and $\mathbf{b}^{(j)}$, assemble $\mathbf{q}^{(j)}$ from equation 11, and compute the head in closed form:

$$W_j = \mathbf{M}^{-1}\mathbf{q}^{(j)}, \qquad j = 0, 1, \ldots, p. \tag{12}$$

Since $\mathbf{M}^{-1}$ is reused, the cost of each order-j solve is dominated by forming the right-hand side $\mathbf{q}^{(j)}$. The order-j solutions are then given by the primary-component reconstruction $u_j(s) = H_u(s)^\top W_j$, and the final approximation is

$$u(s;\varepsilon) = \sum_{j=0}^{p} \varepsilon^j u_j(s) = \sum_{j=0}^{p} \varepsilon^j H_u(s)^\top W_j. \tag{13}$$

This reconstruction completes the online solve for a nonlinear query using only closed-form linear algebra operations.

4.7 Algorithmic summary

An algorithmic summary of the offline/online pipeline is provided in Appendix A.1.5.

5 Results

We evaluate the proposed Chebyshev-augmented one-shot perturbative PINN on three nonlinear benchmarks: two second-order ODEs with pointwise nonlinearities and one reaction–diffusion PDE with a rational reaction term. The goal of these experiments is to assess accuracy and online efficiency under one-shot nonlinear transfer. And to demonstrate the efficiency, we compare the runtime of our method with a regular baseline where we freeze the pretrained feature map and retrains only the linear heads via gradient descent at test time. All experiments follow the offline/online workflow described in Section 4. We report mean squared error (MSE) metrics and time per online solve; full implementation details, architectures, sampling, metric definitions, timing protocol, baseline training settings, and ablation results are provided in Appendix A.2 and Appendix A.3.

5.1 Benchmarks and evaluation setup

ODE benchmarks. We consider the following families on $t \in [0, 5]$:

$$\textbf{ODE1:} \quad u''(t) + \delta\, u'(t) + \alpha\, u(t) + \varepsilon \cos\big(u(t)\big) = \beta \cos(\omega t), \tag{14}$$

$$\textbf{ODE2:} \quad u''(t) + \delta\, u'(t) + \alpha\, u(t) + \varepsilon\, u(t)^{-2} = \gamma - e^{-t}. \tag{15}$$

ODE1 tests the ability of the surrogate-and-perturbation pipeline to handle a smooth non-polynomial nonlinearity $\cos(u)$, which cannot be represented exactly by finite polynomial perturbations, while ODE2 probes a more challenging singular nonlinearity u^{-2} (the test distribution is chosen to keep trajectories away from $u = 0$; see Appendix A.2).

PDE benchmark. On $\Omega = [0, 1] \times [0, 1]$, we consider

$$\textbf{PDE1:} \quad u_t(x,t) = D\, u_{xx}(x,t) + \varepsilon \left(\frac{u(x,t)}{u(x,t)+1} - \delta\, u(x,t) \right) + f(x,t), \tag{16}$$

with constant boundary/initial conditions and a forcing term f derived from manufactured solutions. This benchmark targets a reaction–diffusion regime where the dominant linear operator is parabolic and the nonlinear reaction term is rational in u.

Baseline: head retraining. In addition to our closed-form one-shot online solve, we consider a baseline that preserves the same pretrained feature map but replaces the closed-form head computation with iterative optimization. Specifically, we freeze the shared body H_θ learned in the offline stage and optimize only the linear head W for the target nonlinear instance using gradient descent on the same objective function. Training hyperparameters and settings are reported in Appendix A.2.7.

What is measured. For ODE1–ODE2, we report the mean squared equation residual evaluated on a uniform grid of 100 time points and averaged across 100 test instances. For PDE1, we report the mean squared error against the manufactured truth on a fixed 61×61 evaluation grid, averaged across 32 test instances. All reported online times exclude offline training and the one-time precomputation of $\mathbf{M}^{-1}$ and reflect only the per-query cost of the online stage (Appendix A.2.6). For the baseline, we report the wall-clock time to reach a prescribed error threshold (early stopping). Specifically, we stop learning once the mean squared error falls below $\tau \in \{5 \times 10^{-4},\ 5 \times 10^{-3},\ 10^{-2}\}$ respectively for each benchmark and record the average time of the runs that successfully reach the threshold.

5.2 SUMMARY

Table 1 summarizes the main quantitative outcomes across all three benchmarks, including the online settings used (nonlinearity strength ε, perturbation order p, Chebyshev degree m, and quadrature size M). Across all cases, the online stage consists of sequential one-shot solves across orders $\{u_j\}_{j=0}^{p}$ with a fixed pretrained feature map and a task-invariant inverse $\mathbf{M}^{-1}$.

Table 1: Summary results of our method on all the systems investigated. Training with a small number of heads is sufficient to enable accurate one-shot transfer to unseen nonlinear instances within the prescribed regime. All times are reported for runs on a CPU.

Benchmark	**Nonlinearity**	**Online** (ε, p, m, M)	**MSE**	**Online time (s)**	**Baseline time (s)**
ODE1	$\cos(u)$	(0.5, 12, 20, 1000)	3.83×10^{-6}	9.63×10^{-2}	8.86
ODE2	u^{-2}	(0.1, 12, 20, 1000)	5.31×10^{-5}	7.85×10^{-2}	24.95
PDE1	$\frac{u}{u+1} - \delta u$	(0.5, 20, 30, 1000)	7.12×10^{-5}	9.61×10^{-2}	221.58

5.3 ODE1 RESULTS: COSINE NONLINEARITY

Using the configuration in Table 1, ODE1 achieves a mean equation-residual MSE of 3.83×10^{-6} across 100 test instances, while retaining 9.63×10^{-2}s online time per instance. Beyond residual metrics, we compare the reconstructed series solution against a numerical reference solver (RK45). As shown in Figure 1 (left), the one-shot transfer prediction closely tracks the numerical trajectories despite being obtained without any gradient-based retraining at test time for a subset of test instances.

To visualize the agreement over time, Figure 1 (right) plots the mean discrepancy

$$\Delta(t) \ := \ \frac{1}{N} \sum_{i=1}^{N} \Big(u_{\mathrm{TL}}^{(i)}(t) - u_{\mathrm{ref}}^{(i)}(t) \Big),$$

computed over the test suite on the same uniform grid. The discrepancy remains small throughout $[0, 5]$, indicating that the online surrogate-and-perturbation recursion yields a stable reconstruction over the time interval.

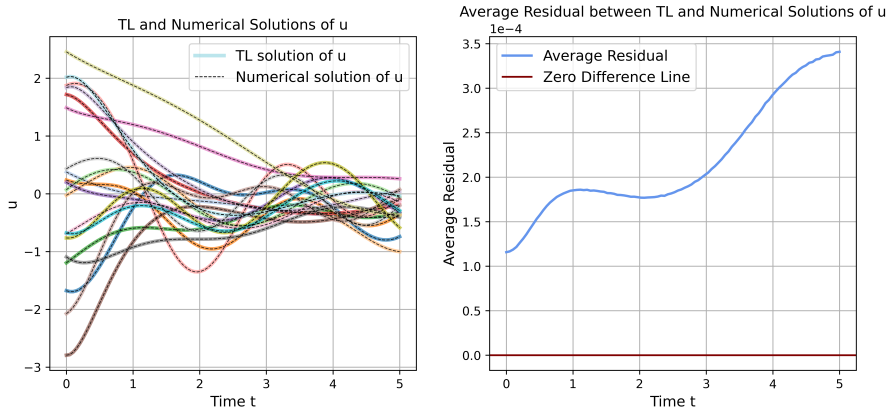

Figure 1: ODE1 equation 14. **Left:** overlay of one-shot transfer predictions $u_{\mathrm{TL}}(t)$ (solid) and numerical reference solutions $u_{\mathrm{ref}}(t)$ (dashed) for a representative subset of test instances. **Right:** mean solution discrepancy over the same subset of test instances (red line indicates $\Delta(t) = 0$).

5.4 ODE2 RESULTS: INVERSE-SQUARE NONLINEARITY

ODE2 is more sensitive due to the inverse-square term u^{-2}, which can amplify errors if trajectories approach $u = 0$. Under the configuration in Table 1, the method attains a mean equation-residual MSE of 5.31×10^{-5} across 100 instances with an average online time of 7.85×10^{-2} seconds per solve. Despite the increased difficulty, and the potential amplification of approximation errors near singularities, the trajectory overlays in Figure 2 (left) show that the one-shot transfer solutions remain in close agreement with the numerical references across the interval.

Figure 2 (right) reports the mean discrepancy $\Delta(t)$ as in ODE1. The discrepancy remains small in magnitude relative to the solution scale in Figure 2 (left), supporting the feasibility of the Chebyshev surrogate combined with one-shot perturbative solves in this regime.

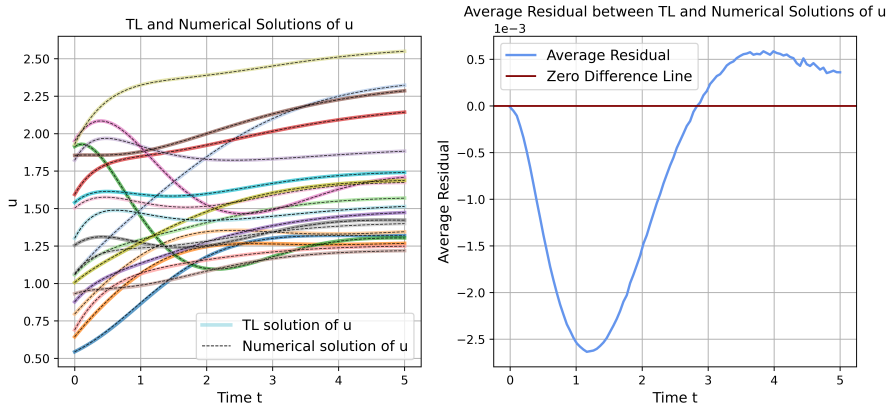

Figure 2: ODE2 equation 15. **Left:** overlay of one-shot transfer predictions $u_{\mathrm{TL}}(t)$ (solid) and numerical reference solutions $u_{\mathrm{ref}}(t)$ (dashed) for a representative subset of test instances. **Right:** mean solution discrepancy over the same subset of test instances (red line indicates $\Delta(t) = 0$).

5.5 PDE1 RESULTS: REACTION–DIFFUSION WITH MANUFACTURED FORCING

For PDE1, the forcing term $f(x, t)$ is constructed from manufactured solutions, enabling direct evaluation of solution error on a fixed grid. Using the configuration in Table 1, the method achieves an average solution MSE of 7.12×10^{-5} on a 61×61 evaluation grid, with an average online time of 9.61×10^{-2} seconds per solve.

Figure 3 visualizes a representative prediction. The left panel shows the predicted field $u_{\mathrm{TL}}(x,t)$, while the right panel reports the pointwise squared error $\big(u_{\mathrm{TL}}(x,t) - u_{\mathrm{true}}(x,t)\big)^2$. The error remains small over most of the domain, with localized increases near the boundary regions in this instance. These boundary-localized errors are consistent with truncation effects in the perturbative reconstruction and finite Chebyshev approximation.

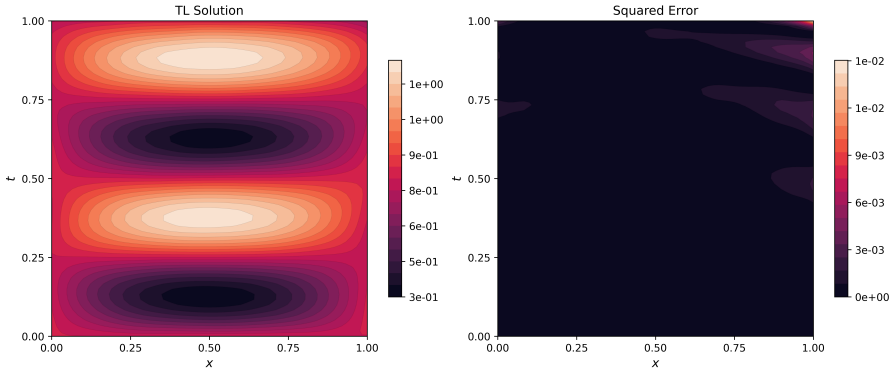

Figure 3: PDE1 equation 16. **Left:** predicted field $u_{\mathrm{TL}}(x,t)$ on the $[0,1]\times[0,1]$ space–time domain. **Right:** pointwise squared error $\big(u_{\mathrm{TL}}(x,t) - u_{\mathrm{true}}(x,t)\big)^2$ against the manufactured solution on the same grid.

5.6 Brief discussion

Across all three benchmarks, the proposed method achieves low MSE with relatively short online solve times per instance. ODE1 demonstrates strong agreement for a smooth bounded nonlinearity, while ODE2 highlights that the approach remains effective for a singular inverse-power term under a small-nonlinearity regime. PDE1 confirms that the same offline/online pipeline transfers to a spatiotemporal parabolic operator with a rational reaction term, yielding accurate reconstructions against a manufactured ground truth.

In addition, we notice that the conventional gradient descent baseline is substantially slower at test time even though it freezes the same pretrained feature map and optimizes only the final linear head. This is orders of magnitude larger than the $\mathcal{O}(10^{-1})$s per-instance online cost of our one-shot solves in Table 1. Moreover, the baseline is less reliable across instances and can fail to reach comparable accuracy without many iterations, highlighting both the efficiency and robustness benefits of the proposed online adaptation framework.

Taken together, these results demonstrate that Chebyshev surrogates can effectively extend one-shot perturbative PINNs beyond polynomial nonlinearities while retaining fast online adaptation. To quantify sensitivity to key online hyperparameters, we perform ablation studies over perturbative strength ε, perturbation order p, and Chebyshev degree m for ODE1 and PDE1 in Appendix A.3. We also report training and optimization settings for the baseline in Appendix A.2.7.

6 Discussion

Our framework inherits the strengths of one-shot transfer for linear operators: fast inference and reusability of latent features (Desai et al., 2022). Meanwhile, it extends applicability to a broad class of nonlinearities via Chebyshev surrogates. This combination enables efficient many-query adaptation while preserving explicit equation structure.

Limitations. The method is not a universal replacement for operator learning. When nonlinearities are very strong, solutions leave bounded ranges, or chaotic behaviors develop, surrogate polynomials may require high degree or piecewise treatment, and the perturbation sequence may become unstable. In such regimes, full operator-learning approaches or instance-specific PINN training strategies may be more appropriate.

7 CONCLUSION

We presented a Chebyshev-augmented framework for one-shot transfer learning in PINNs applied to nonlinear ODEs and PDEs. By approximating general nonlinear terms with Chebyshev polynomials and coupling this surrogate with a perturbative expansion, we reduce a nonlinear problem to a sequence of linear subproblems, each solvable by closed-form output-layer updates in a pretrained latent space. This construction preserves the efficiency and reusability of one-shot transfer while extending its applicability beyond polynomial nonlinearities.

The resulting method targets many-query settings, offering fast adaptation across new forcing terms and boundary/initial conditions without retraining network bodies or modifying the learned feature representation. This framework is particularly relevant for practitioners who seek to rapidly obtain approximate solutions to differential equations with the same dominant operator but under varying initial/boundary conditions, forcings, or coefficients.

To our knowledge, this is the first framework that combines Chebyshev approximation with one-shot perturbative PINNs to enable closed-form nonlinear adaptation beyond polynomial nonlinearities.

Future directions include adaptive or piecewise Chebyshev surrogates to extend the perturbative regime, integration with data-rich or high-dimensional settings, and further optimization of the online pipeline for large-scale many-query applications.

DECLARATION OF AI USAGE

The authors used ChatGPT to assist with language refinement to improve readability and wording. The tool was not used to generate experimental results or to formulate mathematical arguments. All equations, derivations, and interpretations were critically reviewed by the authors, and the authors take full responsibility for the contents of the work.

REFERENCES

Duarte Alexandrino, Ben Moseley, and Pavlos Protopapas. Ptl-pinns: Perturbation-guided transfer learning with physics-informed neural networks for nonlinear systems. *arXiv preprint arXiv:2601.12093*, 2026.

Samuel Auroy and Pavlos Protopapas. One-shot transfer learning for nonlinear pdes with perturbative pinns. In *Machine Learning and the Physical Sciences Workshop (NeurIPS)*, 2025. arXiv:2511.11137.

John P. Boyd. *Chebyshev and Fourier Spectral Methods*. Dover, 2001.

Shaan Desai, Marios Mattheakis, Hayden Joy, Pavlos Protopapas, and Stephen Roberts. One-shot transfer learning of physics-informed neural networks. *arXiv preprint arXiv:2110.11286*, 2022.

Cedric Flamant, Pavlos Protopapas, and David Sondak. Solving differential equations using neural network solution bundles. *arXiv preprint arXiv:2006.14372*, 2020.

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.

Nikola Kovachki, Zongyi Li, Burigede Liu, Kamyar Azizzadenesheli, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Neural operator: Learning maps between function spaces with applications to PDEs. *Journal of Machine Learning Research*, 2023.

Aditi S. Krishnapriyan, Amir Gholami, Shandian Zhe, Robert M. Kirby, and Michael W. Mahoney. Characterizing possible failure modes in physics-informed neural networks. In *NeurIPS*, 2021.

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

Wanzhou Lei, Pavlos Protopapas, and Joy Parikh. One-shot transfer learning for nonlinear odes. *arXiv preprint arXiv:2311.14931*, 2023.

Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Fourier neural operator for parametric partial differential equations. *arXiv preprint arXiv:2010.08895*, 2021.

Lu Lu, Pengzhan Jin, Guofei Pang, Zhongqiang Zhang, and George Em Karniadakis. Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators. *Nature Machine Intelligence*, 3(3):218–229, 2021.

John C. Mason and David C. Handscomb. *Chebyshev Polynomials*. Chapman and Hall/CRC, 2002.

Maziar Raissi, Paris Perdikaris, and George Em 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.

Lloyd N. Trefethen. *Approximation Theory and Approximation Practice*. SIAM, 2 edition, 2019.

Zhi-Qin John Xu, Yaoyu Zhang, and Yanyang Xiao. Training behavior of deep neural network in frequency domain. *arXiv preprint arXiv:1807.01251*, 2019.

Zongren Zou and George Em Karniadakis. L-hydra: Multi-head physics-informed neural networks. *arXiv preprint arXiv:2301.02152*, 2023.

A Appendix

The code has been made publicly available on: `https://github.com/ryqherry/Cheby-PINNs`.

A.1 Additional method details

A.1.1 Chebyshev surrogate construction via Gauss–Chebyshev quadrature

The goal of this step is to replace a general nonlinear term by a polynomial surrogate on a bounded solution range, enabling perturbative decomposition and closed-form adaptation in later stages.

We approximate the pointwise nonlinearity $\mathcal{N}(u)$ by a truncated Chebyshev series on a prescribed range $u \in [u_{\min}, u_{\max}]$. Define the affine map $\Phi : [u_{\min}, u_{\max}] \to [-1, 1]$ by

$$\xi = \Phi(u) := \frac{2u - (u_{\max} + u_{\min})}{u_{\max} - u_{\min}}, \qquad u = \Phi^{-1}(\xi) = \frac{u_{\max} - u_{\min}}{2}\xi + \frac{u_{\max} + u_{\min}}{2}. \tag{17}$$

Let $\tilde{\mathcal{N}}(\xi) := \mathcal{N}(\Phi^{-1}(\xi))$.

We employ Chebyshev polynomials of the first kind $\{T_\ell\}_{\ell \geq 0}$ defined by the three-term recurrence

$$T_0(\xi) = 1, \qquad T_1(\xi) = \xi, \qquad T_{\ell+1}(\xi) = 2\xi\, T_\ell(\xi) - T_{\ell-1}(\xi), \quad \ell \geq 1. \tag{18}$$

In the main paper we use the truncated surrogate $\mathcal{N}(u) \approx \mathcal{N}_m(u)$ defined in equation 3.

The coefficients are given by weighted Chebyshev projections:

$$c_0 = \frac{1}{\pi} \int_{-1}^{1} \frac{\tilde{\mathcal{N}}(\xi)}{\sqrt{1 - \xi^2}}\, d\xi, \tag{19}$$

$$c_\ell = \frac{2}{\pi} \int_{-1}^{1} \frac{\tilde{\mathcal{N}}(\xi)\, T_\ell(\xi)}{\sqrt{1 - \xi^2}}\, d\xi, \qquad \ell \geq 1. \tag{20}$$

We approximate equation 19–equation 20 using Gauss–Chebyshev quadrature (first kind). With M quadrature nodes $\xi_j = \cos\theta_j$ and $\theta_j = \frac{(2j-1)\pi}{2M}$, $j = 1, \ldots, M$, we use

$$c_0 \approx \frac{1}{M} \sum_{j=1}^{M} \tilde{\mathcal{N}}(\xi_j), \tag{21}$$

$$c_\ell \approx \frac{2}{M} \sum_{j=1}^{M} \tilde{\mathcal{N}}(\xi_j)\, T_\ell(\xi_j), \qquad \ell \geq 1. \tag{22}$$

In implementation, for each node ξ_j we compute $\{T_\ell(\xi_j)\}_{\ell=0}^m$ by iterating equation 18. We choose M sufficiently large to reduce numerical error when $\tilde{\mathcal{N}}$ varies rapidly near $\xi = \pm 1$.

Given coefficients $\{c_\ell\}$, evaluation of $\mathcal{N}_m(u)$ at a point proceeds by computing $\xi = \Phi(u)$, generating $\{T_\ell(\xi)\}_{\ell=0}^m$ via equation 18, and forming the sum in equation 3. We keep the Chebyshev representation throughout; in particular, we do not require any conversion to a monomial basis to carry out the perturbative recursion below.

A.1.2 Perturbative expansion and linear subproblem recursion

Substituting the truncated series ansatz equation 5 into the surrogate problem equation 4 requires expanding $\mathcal{N}_m(u(s;\varepsilon))$ in powers of ε. To preserve numerical stability, we perform this expansion directly in the Chebyshev basis.

Let $\xi(s;\varepsilon) := \Phi(u(s;\varepsilon))$. Since Φ is affine,

$$\xi(s;\varepsilon) = \xi_0(s) + \sum_{j=1}^{p} \varepsilon^j \xi_j(s), \qquad \xi_0(s) = \Phi(u_0(s)), \qquad \xi_j(s) = \alpha\, u_j(s)\ (j \geq 1), \tag{23}$$

where $\alpha = \frac{2}{u_{\max} - u_{\min}}$.

For each ℓ, define coefficients $\tau_{\ell,j}(s)$ by the truncated series

$$T_\ell\big(\xi(s;\varepsilon)\big) = \sum_{j=0}^{p} \varepsilon^j\, \tau_{\ell,j}(s) + \mathcal{O}(\varepsilon^{p+1}). \tag{24}$$

That is, $\tau_{\ell,j}(s)$ denotes the coefficient of ε^j in the expansion of $T_\ell(\xi(s;\varepsilon))$. We compute $\{\tau_{\ell,j}\}$ using the Chebyshev recurrence lifted to ε-series. Initialize

$$\tau_{0,0} = 1, \quad \tau_{0,j} = 0\ (j \geq 1), \qquad \tau_{1,j} = \xi_j, \quad j = 0, 1, \ldots, p, \tag{25}$$

and for $\ell \geq 1$ define

$$\tau_{\ell+1,j} = 2 \sum_{k=0}^{j} \xi_k\, \tau_{\ell,j-k} - \tau_{\ell-1,j}, \qquad j = 0, 1, \ldots, p, \tag{26}$$

where the sum $\sum_{k=0}^{j} \xi_k\, \tau_{\ell,j-k}$ is the coefficient of ε^j in the product $\xi(s;\varepsilon)\, T_\ell(\xi(s;\varepsilon))$ for the recurrence. This follows from the Cauchy product of truncated ε-series. That is, multiplying the truncated ε-series $\xi(s;\varepsilon) = \sum_{k=0}^{p} \varepsilon^k \xi_k(s)$ and $T_\ell(\xi(s;\varepsilon)) = \sum_{r=0}^{p} \varepsilon^r \tau_{\ell,r}(s)$ yields the coefficient

$$[\varepsilon^j]\, \xi\, T_\ell = \sum_{k=0}^{j} \xi_k\, \tau_{\ell,j-k}.$$

Using equation 3 and equation 24, the surrogate nonlinearity expands as

$$\mathcal{N}_m(u(s;\varepsilon)) = \sum_{\ell=0}^{m} c_\ell\, T_\ell\big(\xi(s;\varepsilon)\big) = \sum_{j=0}^{p} \varepsilon^j\, \mathcal{G}_j(s) + \mathcal{O}(\varepsilon^{p+1}), \qquad \mathcal{G}_j(s) := \sum_{\ell=0}^{m} c_\ell\, \tau_{\ell,j}(s). \tag{27}$$

Thus, $\mathcal{G}_j(s)$ collects all contributions to the nonlinearity at order ε^j and depends only on lower-order solution components through $\tau_{\ell,j}$.

Substituting equation 5 and equation 27 into equation 4 and matching powers of ε yields the linear subproblems equation 6–equation 7 in the main paper.

A.1.3 Multi-head pretraining details

All subproblems in equation 6–equation 7 share the same dominant linear operator $\mathcal{D}$, so we learn a reusable feature map only once. This offline training cost is amortized across all subsequent linear solves. For differential equations whose linear part contains higher-order derivatives, our

implementation follows the standard first-order reformulation: we introduce auxiliary variables so that the linear operator acts on a vector-valued state. For example, for a diffusion-type operator $u_t - \kappa u_{xx}$ we introduce $y := u_x$ and define the state $\mathbf{u} := [u, y]^\top$, yielding the linear system

$$\mathcal{D}\mathbf{u} = \begin{bmatrix} u_t - \kappa\, y_x \\ u_x - y \end{bmatrix}, \qquad \mathbf{f}(s) = \begin{bmatrix} f(s) \\ 0 \end{bmatrix}, \tag{28}$$

with constraints applied to the primary component u (e.g. Dirichlet/initial values). In ODE settings, similar auxiliary variables can be introduced when $\mathcal{D}$ contains higher-order derivatives (Lei et al., 2023).

We parameterize the solution with a shared network body $H_\theta(s)$ and K linear heads as in equation 8. The following Figure 4 illustrates the general architecture of the multi-head PINN we use for pretraining

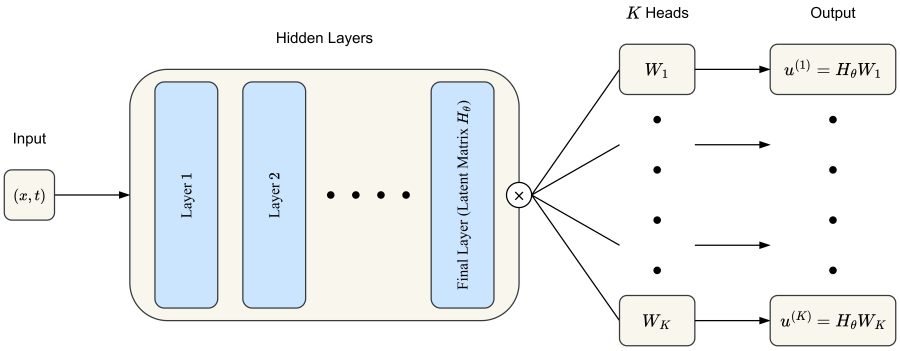

Figure 4: Multi-head PINN architecture used in the offline stage. A shared body produces features $H_\theta(s)$ and each head corresponds to a linear output layer with weights W_k, $k = 1, \ldots, K$.

For head k we specify a linear instance with fixed $(\mathcal{D}, \mathcal{B})$ and task-dependent targets

$$\mathcal{D}\mathbf{u} = \mathbf{f}_k \text{ in } \Omega, \qquad \mathcal{B}u = b_k \text{ on } \partial\Omega, \tag{29}$$

where $\mathcal{B}$ acts on the primary field u (the first component of $\mathbf{u}$). The forcing $\mathbf{f}_k$ and boundary/initial data b_k vary across heads, while the operator $\mathcal{D}$ and constraint type $\mathcal{B}$ remain fixed. Let $\{s_n\}_{n=1}^{N_r} \subset \Omega$ be interior points and $\{\bar{s}_n\}_{n=1}^{N_b} \subset \partial\Omega$ be constraint points.We define the per-head physics-informed losses that enforce the governing equations and constraints for each linear task in the training bundle

$$\mathcal{L}_{\text{pde}}^{(k)}(\theta, W_k) := \frac{1}{N_r} \sum_{n=1}^{N_r} \left\| \mathcal{D}\hat{\mathbf{u}}^{(k)}(s_n) - \mathbf{f}_k(s_n) \right\|_2^2, \tag{30}$$

$$\mathcal{L}_{\text{bc}}^{(k)}(\theta, W_k) := \frac{1}{N_b} \sum_{n=1}^{N_b} \left| \mathcal{B}\hat{u}^{(k)}(\bar{s}_n) - b_k(\bar{s}_n) \right|^2. \tag{31}$$

When manufactured solutions are available for the training bundle, we add a data loss term

$$\mathcal{L}_{\text{data}}^{(k)}(\theta, W_k) := \frac{1}{N_r} \sum_{n=1}^{N_r} \left| \hat{u}^{(k)}(s_n) - u_k^{\text{ref}}(s_n) \right|^2, \tag{32}$$

and set its weight to zero otherwise.

We train the shared parameters θ and all head weights $\{W_k\}_{k=1}^K$ by minimizing

$$\min_{\theta, \{W_k\}} \sum_{k=1}^{K} \left(w_{\text{pde}}\, \mathcal{L}_{\text{pde}}^{(k)}(\theta, W_k) + w_{\text{bc}}\, \mathcal{L}_{\text{bc}}^{(k)}(\theta, W_k) + w_{\text{data}}\, \mathcal{L}_{\text{data}}^{(k)}(\theta, W_k) \right), \tag{33}$$

with $w_{\text{pde}}, w_{\text{bc}} \geq 0$ and $w_{\text{data}} \geq 0$. After training, we freeze θ (hence $\mathbf{H}$) and discard the training heads; only the frozen feature map is retained for all subsequent one-shot solves on new tasks.

A.1.4 ONE-SHOT HEAD SOLVE CONSTRUCTION

Let $\{s_n\}_{n=1}^{N_r} \subset \Omega$ and $\{\bar{s}_n\}_{n=1}^{N_b} \subset \partial\Omega$ be the fixed interior and constraint sampling sets. These sampling sets are shared across all linear subproblems and perturbation orders. We form the stacked operator–feature matrices

$$\mathbf{A}_r := \begin{bmatrix} (\mathcal{D}\mathbf{H})(s_1) \\ \vdots \\ (\mathcal{D}\mathbf{H})(s_{N_r}) \end{bmatrix} \in \mathbb{R}^{(qN_r)\times h}, \qquad \mathbf{A}_b := \begin{bmatrix} (\mathcal{B}H_u)(\bar{s}_1)^\top \\ \vdots \\ (\mathcal{B}H_u)(\bar{s}_{N_b})^\top \end{bmatrix} \in \mathbb{R}^{N_b\times h}, \tag{34}$$

where q is the state dimension (which depends on the system we aim to solve; here, $q = 2$ for the first-order diffusion system equation 28). The corresponding targets are stacked as

$$\mathbf{f}^* := \big[\mathbf{f}^*(s_n)\big]_{n=1}^{N_r} \in \mathbb{R}^{qN_r}, \qquad \mathbf{b}^* := \big[b^*(\bar{s}_n)\big]_{n=1}^{N_b} \in \mathbb{R}^{N_b}.$$

These targets encode the forcing and constraint data for the specific linear instance being solved. For system equation 28, $\mathbf{f}^*(s) = [g(s), 0]^\top$ contains the scalar forcing g in the primary equation and a zero target for the auxiliary constraint.

A.1.5 ALGORITHMIC SUMMARY

Algorithm 1 Chebyshev-augmented one-shot perturbative PINN (offline/online pipeline)

Require: Dominant linear operator $\mathcal{D}$ and constraint operator $\mathcal{B}$; sampling sets $\{s_n\}_{n=1}^{N_r} \subset \Omega$, $\{\bar{s}_n\}_{n=1}^{N_b} \subset \partial\Omega$; weights $w_{\mathrm{pde}}, w_{\mathrm{bc}} > 0$; Chebyshev degree m, quadrature size M; perturbation order p.
Require: Nonlinear instance $(f(\cdot;\eta), b(\cdot;\eta))$ and target ε.
Offline: multi-head pretraining and precomputation.
Train the multi-head model equation 8 by minimizing equation 33; freeze $\mathbf{H}(s) = \mathbf{H}_\theta(s)$.
Assemble fixed matrices $\mathbf{A}_r, \mathbf{A}_b$ from equation 34.
Form $\mathbf{M}$ using equation 11 and store $\mathbf{M}^{-1}$.
Online: Chebyshev surrogate + perturbative one-shot solves.
Choose $[u_{\min}, u_{\max}]$ and compute Chebyshev coefficients $\{c_\ell\}_{\ell=0}^m$ via equation 21–equation 22.
Initialize order-0 targets from equation 6 and form $\mathbf{q}^{(0)}$ via equation 11; set $W_0 = \mathbf{M}^{-1}\mathbf{q}^{(0)}$.
for $j = 1$ **to** p **do**
    Build the forcing term for order j using the Chebyshev lifted recurrence equation 24–equation 27 and the order equations equation 7.
    Form $\mathbf{q}^{(j)}$ from the stacked targets $(\mathbf{f}^{(j)}, \mathbf{b}^{(j)})$ via equation 11 and compute $W_j = \mathbf{M}^{-1}\mathbf{q}^{(j)}$.
end for
Reconstruct $u(s;\varepsilon) = \sum_{j=0}^p \varepsilon^j H_u(s)^\top W_j$.

A.2 EXPERIMENTAL DETAILS

This section collects implementation details, architectures, sampling, and metric definitions referenced in Section 5.

A.2.1 METRICS

ODE residual MSE. For each ODE instance, the equation residual is evaluated on $N_t = 100$ uniformly spaced points in $[0, 5]$. We report the mean squared residual, and then average over 100 test instances.

PDE solution MSE. For PDE1 we evaluate the pointwise squared error against the manufactured solution on a fixed 61×61 (x, t) grid and average over the grid; the reported value is averaged across the full test suite.

A.2.2 CHEBYSHEV SURROGATE SETTINGS

All surrogates use Gauss–Chebyshev quadrature with $M = 1000$ nodes.

ODE1. We approximate $\cos(u)$ on $[u_{\min}, u_{\max}] = [-4.0, 4.0]$ with degree $m = 20$.

ODE2. We approximate u^{-2} on $[u_{\min}, u_{\max}] = [0.5, 6.0]$ with degree $m = 20$.

PDE1. We approximate $\frac{u}{u+1} - \delta u$ on $[u_{\min}, u_{\max}] = [-0.9, 1.0]$ with degree $m = 30$.

A.2.3 CONSTRAINT SPLITTING ACROSS PERTURBATION ORDERS

We use an even split of constant boundary/initial constraints across perturbation orders. If the target constraint value is b, then each order solves with

$$b_{\text{each}} = \frac{b}{\sum_{j=0}^{p} \varepsilon^j},$$

so that reconstructing $u^{(p)}(s;\varepsilon) = \sum_{j=0}^{p} \varepsilon^j u_j(s)$ satisfies the original constraint at the target ε.

A.2.4 ODE ARCHITECTURES AND TRAINING SETTINGS

Both ODE benchmarks (ODE1–ODE2) use the same multi-head network body with Sigmoid Linear Unit (SiLU) activations and $K = 10$ heads. The optimizer is Adam with learning rate 4×10^{-4} and a StepLR scheduler (step size 100, $\gamma = 0.92$). Loss weights are $\alpha_{\text{ode}} = 0.5$ and $\alpha_{\text{ic}} = 1.5$. ODE1 is trained for 5000 iterations and ODE2 for 8000 iterations, each using 50 training points on $[0, 5]$.

Train/test distributions. The full parameter ranges, initial-condition ranges, and test-suite sizes are those listed in the main text (Section 5) and in the project configuration.

A.2.5 PDE1 ARCHITECTURE AND TRAINING SETTINGS

First-order system. To enforce the diffusion operator, we introduce $y = u_x$ and enforce a first-order system at interior points, with boundary/initial conditions applied to u only.

Network and optimization. The PDE network takes $(x, t) \in \mathbb{R}^2$ as input, uses SiLU activations, and has $K = 16$ heads. Training uses Adam with initial learning rate 10^{-3} and StepLR decay (step size 100, $\gamma = 0.98$) for 20000 iterations. Interior sampling uses a 50×50 grid; boundary sampling uses 100 points on $t = 0$ and 100 points on each spatial boundary $x \in \{0, 1\}$. Loss weights are $w_{\text{pde}} = w_{\text{bc}} = w_{\text{data}} = 1$.

Manufactured solutions. Offline heads use $u(x,t) = A \sin(2\pi x) \sin(k\pi t) + b$ with $A \in \{0.5, -0.5\}$ and k taking 8 evenly spaced values in $[1, 2]$, yielding 16 tasks. The test set uses $u(x,t) = A\, x(x-1) \sin(kt) + b$ with $A \in \{1, 2\}$, $k \in \{\pi, 2\pi, 3\pi, 4\pi\}$, and $b \in \{0.2, 0.4, 0.6, 0.8\}$, and defines $f(x,t)$ by substitution into equation 16.

A.2.6 TIMING PROTOCOL

Reported online times include surrogate construction and the sequential computation of orders $\{u_j\}_{j=0}^{p}$, and exclude offline training and the one-time precomputation of $\mathbf{M}^{-1}$.

A.2.7 GRADIENT DESCENT BASELINE

Frozen trunk and head parameterization. This baseline employs the same pretrained multi-head backbone as the proposed method and freezes all shared-body parameters. At test time, we optimize only the final linear head parameters W that map the frozen features to the solution state. Thus, this baseline measures the cost of iterative per-instance adaptation while keeping the representation fixed, in contrast to our closed-form one-shot head solve.

Objective. For ODE1–ODE2, we minimizes a physics-informed objective consisting of the ODE residual loss (enforcing the first-order system) plus an initial-condition (IC) loss. For PDE1, we minimizes the PDE residual loss on interior points plus a boundary/initial-condition loss on constraint points. All baseline runs use the same nonlinear instances (test suite) as used for evaluating our one-shot method.

Optimization, scheduling, and stopping. We use Adam with a StepLR schedule. For ODE2 and PDE1, we additionally apply gradient clipping with max-norm 1.0 to improve stability. We employ an early-stopping criterion based on the mean-squared residual of the primary differential equation, with a maximum iteration cap. If the early-stopping criterion is not met within the cap, the run is marked as reaching the maximum iteration limit.

Timing protocol. Baseline online time includes the full per-instance head optimization loop until early stopping or the iteration cap. Reported runtimes are averaged across test instances that terminate before the cap. As with our method, all online times exclude offline multi-head training and any one-time precomputations.

Table 2: Baseline: optimization and stopping (CPU).

Benchmark	Adam LR	StepLR (step,γ)	Stop / cap
ODE1	10^{-2}	(100, 0.92)	5×10^{-4} / 2×10^{4}
ODE2	10^{-2}	(100, 0.92)	5×10^{-3} / 2×10^{4}
PDE1	3×10^{-3}	(200, 0.96)	1×10^{-2} / 4×10^{3}

Table 3: Baseline: sampling and loss weights.

Benchmark	Points per iter	Weights	Stabilization / init
ODE1	$N_t = 100$	$(w_{\text{ode}}, w_{\text{ic}}) = (0.5, 1.5)$	none / zero
ODE2	$N_t = 100$	$(w_{\text{ode}}, w_{\text{ic}}) = (0.5, 1.5)$	clip(1.0) / random
PDE1	$I = 60$, $B = 200$	$(w_{\text{pde}}, w_{\text{bc}}) = (1, 1)$	clip(1.0) / zero

A.3 Ablation study

We report ablations over perturbation strength ε, perturbation order p, and Chebyshev degree m for ODE1 and PDE1. Unless otherwise specified, all other online settings follow the default configurations used in Table 1, and MSE is computed using the same metrics described in Appendix A.2.1.

For ODE1, all ablation studies are evaluated on the first parameter/initial-condition instance drawn from the same random test-generation procedure used in the main experiments. For PDE1, all ablation studies are evaluated on a fixed manufactured instance with parameters $A = 2$, $k = 2\pi$, and $b = 0.4$ (as defined in Appendix A.2.5), with all other settings unchanged.

Table 4: ODE1 ablation over perturbation strength ε.

ε	MSE
0.05	3.40×10^{-8}
0.10	1.53×10^{-6}
0.20	7.94×10^{-7}
0.50	2.56×10^{-6}
0.80	2.74×10^{-5}

Table 5: ODE1 ablation over perturbation order p.

p	MSE
1	6.64×10^{-4}
2	9.97×10^{-5}
3	4.06×10^{-5}
4	2.63×10^{-5}
5	5.44×10^{-6}
6	2.91×10^{-6}
7	7.55×10^{-6}
8	1.75×10^{-6}
9	2.51×10^{-6}
10	1.80×10^{-7}
11	5.73×10^{-7}
12	2.56×10^{-6}
13	2.46×10^{-6}
14	1.51×10^{-5}
15	8.75×10^{-6}
16	3.82×10^{-7}
17	2.58×10^{-6}
18	5.64×10^{-6}
19	3.82×10^{-7}
20	2.38×10^{-7}

Table 6: ODE1 ablation over Chebyshev degree m.

m	MSE
1	4.52×10^{-1}
2	1.01×10^{-1}
3	1.01×10^{-1}
4	2.54×10^{-3}
5	2.54×10^{-3}
6	1.23×10^{-5}
7	1.23×10^{-5}
8	4.92×10^{-7}
9	4.84×10^{-7}
10	1.11×10^{-7}
11	1.22×10^{-7}
12	2.46×10^{-5}
13	3.54×10^{-5}
14	7.71×10^{-6}
15	1.46×10^{-5}
16	9.23×10^{-7}
17	9.06×10^{-7}
18	6.67×10^{-6}
19	6.58×10^{-6}
20	2.56×10^{-6}

Table 7: PDE1 ablation over perturbation strength ε.

ε	MSE
0.05	3.84×10^{-6}
0.10	3.94×10^{-6}
0.20	4.66×10^{-6}
0.50	2.91×10^{-5}
0.80	2.41×10^{-4}

Table 8: PDE1 ablation over perturbation order p.

p	MSE
1	1.42×10^{-5}
2	4.00×10^{-5}
3	3.63×10^{-5}
4	3.23×10^{-5}
5	3.00×10^{-5}
6	2.92×10^{-5}
7	2.89×10^{-5}
8	2.88×10^{-5}
9	2.89×10^{-5}
10	2.90×10^{-5}
11	2.91×10^{-5}
12	2.91×10^{-5}
13	2.91×10^{-5}
14	2.91×10^{-5}
15	2.91×10^{-5}
16	2.91×10^{-5}
17	2.91×10^{-5}
18	2.91×10^{-5}
19	2.91×10^{-5}
20	2.91×10^{-5}

Table 9: PDE1 ablation over Chebyshev degree m.

m	MSE
2	4.10×10^{-4}
4	1.86×10^{-4}
6	8.50×10^{-5}
8	2.75×10^{-5}
10	3.50×10^{-5}
12	2.75×10^{-5}
14	2.93×10^{-5}
16	2.92×10^{-5}
18	2.90×10^{-5}
20	2.91×10^{-5}
22	2.91×10^{-5}
24	2.91×10^{-5}
26	2.91×10^{-5}
28	2.91×10^{-5}
30	2.91×10^{-5}
32	2.91×10^{-5}
34	2.91×10^{-5}
36	2.91×10^{-5}
38	2.91×10^{-5}
40	2.91×10^{-5}