RICH REPRESENTATIONS, SMALL HEADS: FAST PDE SOLVES AND FLOWS IN WEIGHT SPACE

Anonymous authors
Paper under double-blind review

ABSTRACT

The study of embeddings of partial differential equations (PDEs) using Physics-Informed Neural Networks (PINNs) involves learning a latent space basis that represents a family of solutions. For linear PDEs, a sufficiently expressive learned basis yields approximate solutions for unseen equation parameters (EPs) or initial conditions (ICs) through a direct solve for the output-head weights, without retraining the shared network. This procedure is referred to as one-shot transfer learning (OTL). For nonlinear equations, however, a direct closed-form solution for the head weights is generally unavailable, and existing extensions of OTL instead use approaches that rely on linearizing the nonlinearity, such as perturbative decompositions or Chebyshev surrogates. In this work, we investigate second-order methods that adapt the head weights to the nonlinear problem directly while keeping the learned basis fixed.

We train multi-head PINNs on families of linear and nonlinear PDEs with varying EPs and ICs, learning a shared basis for each equation with a separate linear head for each training instance. This low-dimensional learned basis restricts adaptation to a small space of solution coefficients, making second-order optimization tractable. With the learned basis fixed, we investigate two second-order methods for obtaining new solutions at unseen EPs or ICs. The first applies Gauss–Newton to the least-squares problem for the head weights. The second uses continuation in head-weight space, for which we derive an analytic expression, and returns the path of heads between a trained instance and the target. We compare both methods against training a new head in terms of accuracy and computational cost. On the heat, wave, Allen–Cahn and Burgers equations, both methods improve the accuracy of a trained head for the same computational cost, improving the efficiency of transfer learning in PINNs.

1 INTRODUCTION

Physics-informed neural networks (PINNs) (Raissi et al., 2019) solve partial differential equations (PDEs) by minimizing the residual of the equation on a set of collocation points without requiring solution data. In many applications, however, one is interested not in a single solution but in a whole family of them, such as for instance the solutions of the same PDE for different values of an equation parameter (EP) or for different initial conditions (ICs). Training a new PINN for every member of the family, also referred to as *instance*, is expensive; transfer learning (TL) aims at reusing a pretrained network to obtain the solution of an unseen instance at a much lower cost.

A convenient setting for TL is multi-head (MH) training (Pellegrin et al., 2022; Zou & Karniadakis, 2023), where a shared body learns a set of basis functions spanning a so-called *latent space* for the whole family, and a linear output layer, or *head*, projects them onto each training solution. For an unseen instance the body is frozen and only a new head has to be found. For linear PDEs this is a linear least-squares problem solved in closed form, known as one-shot TL (OTL) (Desai et al., 2022). For nonlinear PDEs, however, this closed form is lost. Current extensions of OTL recover it through perturbative expansions or Chebyshev surrogates of the nonlinearity (Lei et al., 2023; Auroy & Protopapas, 2025; Alexandrino et al., 2026; Rao & Protopapas, 2026), which require a small parameter. Otherwise, the head is trained with a first-order optimizer, which is slow on the poorly conditioned PINN loss (Krishnapriyan et al., 2021; Rathore et al., 2024).

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.

In this work we start from a simple observation: since only the head is adapted, the TL problem has as many unknowns as basis functions in the pretrained body. Second-order methods, which are usually too expensive for PINN training, then become affordable if the latent space basis is chosen to be small. Here, we study two such second-order methods. The first applies Gauss–Newton (GN) to the nonlinear least-squares problem for the head at the target instance; for linear PDEs it reduces to OTL in a single step. The second method, which we call Hessian continuation, follows the minimizer of the least-squares problem as the EP or IC is varied from a trained instance to the target by integrating an ordinary differential equation for the head derived from the implicit function theorem. Since the residual on a frozen basis cannot vanish, this equation involves the full Hessian of the loss rather than its Gauss–Newton approximation. Unlike GN, the continuation returns the whole path of heads between the two instances, each a transferred head for its own instance, together with their derivative with respect to the EP or IC. To make this path interpretable, we train the latent space basis to be orthonormal, such that the head weights are L^2 coordinates of the network output in a frame shared by all instances and distances between heads are hence distances between outputs.

We test both methods on two linear (heat and wave) and two nonlinear (Burgers and Allen–Cahn) equations in one spatial dimension, varying either an EP or the IC. Both second-order methods obtain a new head in at most a few hundred milliseconds on a CPU. At equal wall-clock time, their errors are a median of 3.7 times smaller than those of the *Adam* baseline, and up to 61 times smaller. GN is the cheaper of the two, while the continuation reaches the same accuracy in seven of the eight experiments and additionally provides the full paths in weight space.

Our contributions are:

- A second-order treatment of TL for nonlinear PDEs on a frozen MH basis which handles the nonlinearity directly without perturbative expansions or surrogates;
- The derivation of the Hessian continuation in the weight space of the heads, which returns the path of intermediate heads between a trained instance and the target, together with their sensitivity to the EP or IC;
- An approximately orthonormal latent basis in which the head weights are the L^2 coordinates of the network output, so that trained and transferred heads, and the paths between them, live in a common frame;
- A systematic comparison of GN, the Hessian continuation and first-order head training for TL, in accuracy and cost, on linear and nonlinear PDEs with varying EPs and ICs.

2 Related Work

Physics-informed neural networks. PINNs were introduced by Dissanayake & Phan-Thien (1994) and Lagaris et al. (1998) and popularized by Raissi et al. (2019) (see Karniadakis et al. (2021) for a review). Their loss is ill-conditioned, since the differential operators in the residual spread the spectrum of its curvature (Krishnapriyan et al., 2021; Rathore et al., 2024). Second-order optimizers improve the accuracy of first-order ones by orders of magnitude (Müller & Zeinhofer, 2023; Jnini et al., 2025; Jnini & Vella, 2025; Dangel et al., 2024; Rathore et al., 2024), and for the nonlinear PDE classes analyzed by Bonfanti et al. (2024), the Hessian of the PDE residual with respect to network parameters need not vanish in the infinite-width limit. These methods act on all network parameters, so the curvature matrix is expensive to form and invert (Dangel et al., 2024). In our TL setting only a linear head is adapted, and the exact Jacobian and Hessian are cheap to assemble.

Transfer learning in PINNs. TL methods differ in what is adapted for a new instance. *Zero-shot* methods adapt nothing at test time: the EP or IC enters as an extra input (Flamant et al., 2020) or through a hypernetwork that generates the PINN weights, or part of them (de Avila Belbute-Peres et al., 2021; Cho et al., 2023). Hence, the accuracy is set by how well the training covers the family. *Raw* TL fine-tunes some or all layers of a pretrained network (Goswami et al., 2020; Xu et al., 2023; Mustajab et al., 2024; Gao et al., 2022; Penwarden et al., 2023), at the cost of a gradient-based optimization per instance. *MH* TL lies in between: the body is frozen and only a new head is obtained (Zou & Karniadakis, 2023; Pellegrin et al., 2022). It has been used to reach stiff regimes (Seiler et al., 2025; Tarancón-Álvarez et al., 2025) and to study the geometry of the solution

manifold (Jimenez et al., 2026). The new head is either trained by gradient descent (Pellegrin et al., 2022; Tarancón-Álvarez et al., 2025) or, for linear PDEs, obtained by OTL (Desai et al., 2022). Nonlinear extensions of OTL use perturbative expansions (Lei et al., 2023; Auroy & Protopapas, 2025; Alexandrino et al., 2026) or Chebyshev surrogates (Rao & Protopapas, 2026); they require a small parameter, and their error grows with the strength of the nonlinearity (Auroy & Protopapas, 2025). We instead keep the nonlinear residual and solve for the head with second-order methods. Moreover, whereas Jimenez et al. (2026) orthogonalize the matrix of head weights, we make the basis itself orthonormal such that the head weights are L^2 coordinates of the network output.

Fixed-basis solvers and neural operators. Extreme learning machines and random-feature methods also solve for output coefficients on a frozen basis (Dwivedi & Srinivasan, 2020; Chen et al., 2022; Datar et al., 2026), with Newton-type or nonlinear least-squares iterations for nonlinear PDEs (Dong & Li, 2021; Huang & Wu, 2025; Awojinrin et al., 2026; Yang, 2026), closely related to our GN step. Their basis, however, is random, prescribed or meta-learned without the PDE (Yang, 2026) and it is not trained with one head per instance. Neural operators such as DeepONet (Lu et al., 2021) and FNO (Li et al., 2021a) learn the map from inputs to solutions, possibly with a physics loss (Wang et al., 2021; Li et al., 2021b), and transfer by fine-tuning (Goswami et al., 2022; Xu et al., 2022; Subramanian et al., 2023). Our body plays the role of a DeepONet trunk, but each head is solved from the residual of its own instance, without reference solutions. An extended comparison is given in Appendix A.

3 Problem statement

We consider a family of PDEs whose instances differ by an EP value or an IC. These instances are labelled by ξ. A MH PINN is first trained on D instances $\{\xi_j\}_{j=1}^{D}$. Its body provides B basis functions $\mathbf{H}_\theta(x,t)$, shared by all instances, and each instance j has its own head $\mathbf{w}_j \in \mathbb{R}^B$, such that its solution is $\mathbf{w}_j^\top \mathbf{H}_\theta$ up to the reparametrization that imposes the IC and boundary conditions (Eq. 6).

Given a new instance $\xi^\star$ unseen during training, we want to obtain the new head $\mathbf{w}^\star \in \mathbb{R}^B$ for which $(\mathbf{w}^\star)^\top \mathbf{H}_\theta$ solves the PDE at $\xi^\star$, with the pretrained body $\mathbf{H}_\theta$ kept frozen. Since the frozen basis cannot represent the new solution exactly, $\mathbf{w}^\star$ is defined as the head that minimizes the PDE residual on a set of collocation points without using any reference solution. For a linear PDE this is a linear least-squares problem with a closed-form solution (OTL). For a nonlinear PDE it is a nonlinear least-squares problem, usually solved by training the head with a first-order optimizer. Our first question is whether second-order methods (affordable here because B is kept small) solve it faster and at least as accurately.

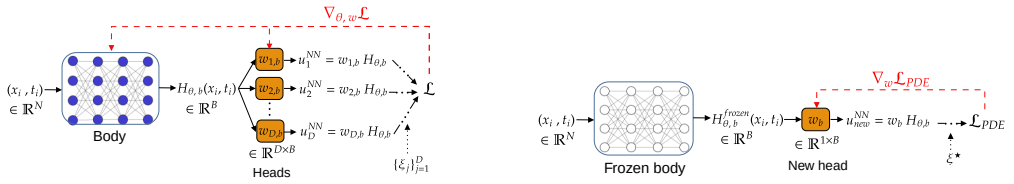
Our second question concerns the latent space between instances. As ξ moves from a trained instance ξ_0 to a new $\xi^\star$, the optimal head traces a path $\mathbf{w}^\star(\xi)$ in weight space. Since the basis is made (approximately) orthonormal, the head weights are coordinates of the network output, and hence this path describes how the part of the solution carried by the head changes with the EP or IC. Methods that solve only at $\xi^\star$, such as GN, return the endpoint of this path but not the path itself. We want to compute the path and its derivative $\partial \mathbf{w}^\star / \partial \xi$ directly, rather than by solving the least-squares problem at every intermediate instance. As well as information on the sensitivity to the EP or IC, this would provide a description of the local structure of the solution family in latent coordinates, which a set of isolated endpoints cannot reveal.

GN (Section 4.2.2) addresses the first question and the Hessian continuation (Section 4.2.3) both. Section 5 compares them with a first-order baseline, in accuracy against reference solutions and in wall-clock cost, and with one another.

4 Methods

4.1 Embedding space via multi-head training

The first step is to learn a latent space whose basis spans the solutions of a PDE family for several EP values or ICs. To this end, we use MH training (Figure 1a). A shared body $\mathbf{H}_\theta : \mathbb{R}^2 \to \mathbb{R}^B$, which



(a) MH training. (b) Transfer learning.

Figure 1: (a) The shared body maps the collocation points (x_i, t_i) to B basis functions $H_{\theta,b}$. One bias-free linear head $w_{b,j}$ per instance ξ_j projects them onto that instance's solution u_j^{NN}, and all heads enter the loss of Eq. (12). The reparametrization of Eq. (6) is not shown. Solid black arrows: forward pass; dashed red arrows: backpropagation. (b) For an unseen instance $\xi^\star$ the body is frozen and only a new head $\mathbf{w}^\star \in \mathbb{R}^B$ is sought, from the PDE residual alone.

depends only on the independent variables (x, t), is followed by a bias-free linear layer holding all heads, $\mathbf{w} \in \mathbb{R}^{B\times D}$, one per training instance. For a batch of collocation points, the raw output of head j is

$$u_j^{\mathrm{NN}}(x_i, t_i; \xi_j) = \sum_{b=1}^{B} w_{b,j}\, H_{\theta,b}(x_i, t_i), \tag{1}$$

where $i = 1, \ldots, N$ labels the collocation points and $j = 1, \ldots, D$ the training instances, each labelled by $\xi_j \in \mathbb{R}^{d_\xi}$. Since the body is shared, the dependence on ξ_j is carried entirely by the head, $w_{b,j} \equiv w_b(\xi_j)$. Moreover, a single automatic-differentiation call returns the derivatives of all heads at all points, so a training step costs one backward pass per derivative order, independently of D.

We consider four PDEs in one spatial dimension, two linear and two nonlinear:

$$\text{Heat:} \quad u_t - \nu\, u_{xx} = 0, \tag{2}$$

$$\text{Wave:} \quad u_{tt} - \nu^2\, u_{xx} = 0, \tag{3}$$

$$\text{Burgers:} \quad u_t + u\, u_x - \nu\, u_{xx} = 0, \tag{4}$$

$$\text{Allen–Cahn:} \quad u_t - \nu\, u_{xx} + k\,(u^3 - u) = 0. \tag{5}$$

In these, the EP ν is the diffusivity, the wave speed, the viscosity and the interface diffusivity, respectively. The reaction rate of the Allen–Cahn equation is fixed to $k = 1$.

The IC and the boundary conditions (BCs) are imposed exactly by reparametrizing the output (Lagaris et al., 1998):

$$u_j(x,t) \;=\; A_j(x,t) + \underbrace{\tilde{x}(1-\tilde{x})\big(1 - e^{-t^p}\big)}_{\equiv\, g(x,t)}\; u_j^{\mathrm{NN}}(x,t), \tag{6}$$

with $\tilde{x} = (x - x_{\min})/(x_{\max} - x_{\min})$. Here, $p = 2$ for equations of second order in time and $p = 1$ otherwise. The particular part A_j carries the initial profile $u_{0,j}$ along with any Dirichlet data. The envelope satisfies $g(x, 0) = 0$ and $g(x_{\min}, t) = g(x_{\max}, t) = 0$, and for $p = 2$ also $g_t(x, 0) = 0$, such that u_j satisfies the IC and BCs by construction. For all four equations we use homogeneous Dirichlet BCs, $u_{0,j}(x_{\min}) = u_{0,j}(x_{\max}) = 0$, and for the wave equation $u_t(x, 0) = 0$. Each head is trained on the residual of its own instance, $\mathcal{R}_{\xi_j}[u_j]$, through the PINN loss

$$\mathcal{L}_{\mathrm{PDE}} = \frac{1}{ND} \sum_{i=1}^{N} \sum_{j=1}^{D} \Big(\mathcal{R}_{\xi_j}[u_j(x_i, t_i)]\Big)^2. \tag{7}$$

The two settings below differ only in how ξ_j enters this residual.

4.1.1 Varying equation parameter

In the multi-EP setting the IC is the same for every head, $u_{0,j} = u_0$ and $A_j = A$, and the heads are labelled by the EP, $\xi = \nu$ with $d_\xi = 1$. Head j is trained on the residual at ν_j, namely $\mathcal{R}_{\nu_j}$, for D values of ν sampled over the range of interest (see Table 2). Since the reparametrization (Eq. 6) is identical across heads, ν enters the residual only through the equation coefficients instead of through A.

4.1.2 VARYING IC VIA A VAE DECODER

In the multi-IC setting the EP is fixed, and each head has its own initial profile $u_{0,j}$ which enters Eq. (6) through A_j. TL to an unseen IC then requires a continuous, low-dimensional parametrization of the ICs: a profile sampled on the spatial grid lives in $\mathbb{R}^{n_x}$, which is far too large to serve as ξ. We therefore compress a bank of profiles using a variational autoencoder (VAE) (Kingma & Welling, 2014; Rezende et al., 2014), trained beforehand and subsequently frozen. Its decoder $\mathcal{D} : \mathbb{R}^{d_z} \to \mathbb{R}^{n_x}$ is a truncated sine expansion whose coefficients are given by a small MLP with `SiLU` activations,

$$u_{0,j}(x) \;=\; \mathcal{D}(\mathbf{z}_j)(x) \;=\; \sum_{k=1}^{K} c_k(\mathbf{z}_j)\, \sin\left(\frac{k\pi(x - x_{\min})}{x_{\max} - x_{\min}}\right), \tag{8}$$

where we choose $d_z = 2$, so that $\xi = \mathbf{z} \in \mathbb{R}^2$, and $K \in \{4, 10\}$ modes.[1] Since the decoder is band-limited, the ICs stay smooth and resolved on the grid. Furthermore, it vanishes at both boundaries, making every IC compatible with the BCs. Its derivative $\partial\mathcal{D}/\partial\mathbf{z}$ is available analytically, as required by the Hessian continuation (Section 4.2.3), and the path $\mathbf{w}(\mathbf{z})$ followed by the continuation inherits its smoothness, hence the smooth activation. The latent dimension $d_z = 2$ is too small for the VAE to fully converge, but this is not a problem here: we only use the decoder, and the ICs are generated by sampling directly in $\mathbf{z}$-space, $\mathbf{z} \sim \mathcal{N}(\mu, \sigma) \approx \mathcal{N}(0, 5.9)$, the approximate empirical aggregate posterior of the VAE.

4.1.3 ORTHONORMAL LATENT BASIS

The MH construction fixes the basis only up to its span: any invertible $B \times B$ mixing of the basis functions, reabsorbed into the heads, leaves every training solution unchanged. We fix this freedom by making the basis orthonormal over the domain. Treating the collocation points as samples of the domain, the empirical Gram matrix

$$G_{ab} = \frac{1}{N}\sum_{i=1}^{N} H_{\theta,a}(x_i, t_i)\, H_{\theta,b}(x_i, t_i) \simeq \langle H_{\theta,a}, H_{\theta,b}\rangle \tag{9}$$

should be the identity, which we impose softly through the penalty

$$\mathcal{L}_\perp = \lambda_\perp \|G - \mathbb{I}_B\|^2, \tag{10}$$

evaluated on the same batch as the residual. The heads then become coordinates: writing Eq. (1) as $u_j^{\mathrm{NN}} = \mathbf{w}_j^\top \mathbf{H}_\theta$,

$$w_{b,j} \;=\; \langle u_j^{\mathrm{NN}}, H_{\theta,b}\rangle\,, \qquad \|u_j^{\mathrm{NN}}\|^2 \;=\; \|\mathbf{w}_j\|^2. \tag{11}$$

Each weight is the projection of the network output onto one basis function, the norm of a head is the L^2 norm of that output, and distances between heads are distances between outputs in a frame shared by all instances. The full field $u_j = A_j + g\, u_j^{\mathrm{NN}}$ of Eq. (6) adds the particular part and the envelope. Hence, with a shared IC (multi-EP), differences between solutions are the enveloped differences between outputs, $u_i - u_j = g\,(\mathbf{w}_i - \mathbf{w}_j)^\top \mathbf{H}_\theta$. With varying ICs, A_j changes too and head distances measure the part of the solution carried by the head only. The total training loss is

$$\mathcal{L} = \mathcal{L}_{\mathrm{PDE}} + \mathcal{L}_\perp. \tag{12}$$

4.2 TRANSFER LEARNING

For an unseen instance $\xi^\star$, in TL we seek a new head $\mathbf{w}^\star \in \mathbb{R}^B$ with the pretrained body being frozen (Figure 1b).[2] Evaluating the reparametrized field of Eq. (6) at the N collocation points and inserting it into the residual of the target equation defines

$$F_i(\mathbf{w}, \xi^\star) \equiv \mathcal{R}_{\xi^\star}\big[u(x_i, t_i)\big]\,, \qquad \mathbf{F} : \mathbb{R}^B \to \mathbb{R}^N, \tag{13}$$

[1] PINNs converge more easily for ICs with little high-frequency content, particularly for the nonlinear equations. We choose $d_z = 2$ so that the instance space can be covered densely by the training heads and the weight space can be displayed as a map over the $\mathbf{z}$-plane (Figs. 4 and 8).

[2] TL can be used for interpolation, inside the training range, or for extrapolation, outside of it. The same procedure applies to joint variations $(\nu^\star, \mathbf{z}^\star)$.

with components F_i. Since $N \gg B$, the system $\mathbf{F} = \mathbf{0}$ is overdetermined and the new head is defined as its least-squares solution,

$$\mathbf{w}^\star(\xi^\star) = \arg\min_{\mathbf{w}} \Phi(\mathbf{w}, \xi^\star)\,, \qquad \Phi(\mathbf{w}, \xi) = \tfrac{1}{2}\,\mathbf{F}^\top(\mathbf{w}, \xi)\,\mathbf{F}(\mathbf{w}, \xi). \tag{14}$$

The residual cannot vanish, as every head, either trained or transferred, carries a non-zero residual. Furthermore, note that $\mathbf{w}^\star$ is in fact a minimizer of Φ, not a root of $\mathbf{F}$. None of the methods presented below uses a reference solution.

4.2.1 First-order baseline

The standard approach minimizes Φ with a first-order optimizer, as in MH training, backpropagating only into the B weights of the new head. We use *Adam* as the baseline in all experiments.

4.2.2 Second-order TL: Gauss–Newton

Since B is small ($\sim\ \mathcal{O}(10)$ in our experiments), second-order methods are affordable. The reparametrization is affine in $\mathbf{w}$, so for a linear PDE the residual is affine too, $\mathbf{F}(\mathbf{w}, \xi^\star) = \mathbf{J}(\xi^\star)\,\mathbf{w} + \mathbf{b}(\xi^\star)$, with a Jacobian $\mathbf{J} = \partial\mathbf{F}/\partial\mathbf{w}$ independent of $\mathbf{w}$. Equation 14 is then solved by a single pseudo-inverse, $\mathbf{w}^\star = -\mathbf{J}^+\mathbf{b}$, which is the OTL solve (Desai et al., 2022). For the Burgers and Allen–Cahn equations, $\mathbf{F}$ is quadratic and cubic in $\mathbf{w}$, and no closed form exists. The GN iteration is the direct extension of OTL to this case. Linearizing $\mathbf{F}(\mathbf{w}+\delta\mathbf{w}) \simeq \mathbf{F}(\mathbf{w}) + \mathbf{J}(\mathbf{w}, \xi^\star)\,\delta\mathbf{w}$ and minimizing the resulting quadratic model gives $\mathbf{J}^\top\mathbf{J}\,\delta\mathbf{w} = -\mathbf{J}^\top\mathbf{F}$, i.e.

$$\mathbf{w}^{(n+1)} \;=\; \mathbf{w}^{(n)} \;-\; \lambda\,\mathbf{J}^+\big(\mathbf{w}^{(n)}, \xi^\star\big)\,\mathbf{F}\big(\mathbf{w}^{(n)}, \xi^\star\big), \tag{15}$$

with $\mathbf{J}^+$ the Moore–Penrose pseudo-inverse (Moore, 1920; Penrose, 1955) and damping $\lambda = 1$. Each step is solved by a truncated SVD with relative cutoff r_{cond}, since the frozen basis has near-null directions along which the residual is insensitive to $\mathbf{w}$ (Appendix D). For a linear PDE, one step reproduces OTL from any starting point, so the same routine covers all four equations.

GN starts from the trained head nearest to the target, $\mathbf{w}^{(0)} = \mathbf{w}(\xi_0)$ with $\xi_0 = \xi_{j_0}$ and $j_0 = \arg\min_j \|\xi^\star - \xi_j\|$, which is also the starting point of the continuation. Starting from $\mathbf{w}^{(0)} = \mathbf{0}$ is reported as an ablation (Appendix B).

GN has two limitations that motivate the next method. First, it discards the residual-weighted term $\sum_i F_i\,\partial^2 F_i/\partial w^j \partial w^k$ of the Hessian of Φ. It therefore converges quadratically only when the residual vanishes, and linearly otherwise, which is the generic case here. Second, it is a local method that returns only the head at $\xi^\star$.

4.2.3 Second-order TL: Hessian continuation

Instead of the minimizer at a single instance, the Hessian continuation tracks the path of minimizers $\mathbf{w}^\star(\xi)$ as ξ varies continuously from a trained instance ξ_0 to the target $\xi^\star$. Along this path the gradient of Φ with respect to $\mathbf{w}$ vanishes,

$$\nabla_{\mathbf{w}}\Phi = \tfrac{1}{2}\,\nabla_{\mathbf{w}}\big(\mathbf{F}^\top\mathbf{F}\big) = \mathbf{0} \quad\Longrightarrow\quad \partial_j\Phi = (\partial_j F_i)\,F_i = 0\,, \tag{16}$$

with a sum over repeated indices. These are B equations for the B unknowns at each ξ, and the same condition that GN drives to zero at $\xi^\star$, since $\mathbf{J}^+\mathbf{F} = \mathbf{0} \Leftrightarrow \mathbf{J}^\top\mathbf{F} = \mathbf{0}$. Differentiating Eq. (16) with respect to the components ξ^a and solving for $\partial w^k/\partial\xi^a$ gives

$$\frac{\partial w^k}{\partial \xi^a} \;=\; -\,\big(H_{ww}^{-1}\big)^{kj}\,\big(H_{w\xi}\big)_{ja}\,, \tag{17}$$

with

$$\big(H_{ww}\big)_{jk} = \frac{\partial^2 F_i}{\partial w^j \partial w^k}F_i + \frac{\partial F_i}{\partial w^j}\frac{\partial F_i}{\partial w^k}\,, \qquad \big(H_{w\xi}\big)_{ja} = \frac{\partial^2 F_i}{\partial w^j \partial \xi^a}F_i + \frac{\partial F_i}{\partial w^j}\frac{\partial F_i}{\partial \xi^a}\,. \tag{18}$$

Here H_{ww} is the full Hessian of Φ, including the residual-weighted term that GN discards, which does not vanish because $\mathbf{F} \neq \mathbf{0}$. Equation 17 is the implicit function theorem applied to Eq. (16). Together with the initial condition

$$\mathbf{w}(\xi_0) \;=\; \arg\min_{\mathbf{w}'}\ \Phi(\mathbf{w}', \xi_0)\,, \tag{19}$$

it defines, by the implicit function theorem, a unique branch of stationary points of Φ through $\mathbf{w}(\xi_0)$, locally wherever H_{ww} is nonsingular; if H_{ww} stays positive definite along the path, the branch consists of local minimizers. Note that Φ may have other stationary points at the same ξ, and the continuation follows the branch of its start head (see Appendix B.4). This is natural-parameter continuation (Allgower & Georg, 2003) applied to the minimizers of a least-squares problem rather than to roots (see Appendix A).

The trained head does not satisfy Eq. (19) exactly, since MH training does not reach $\mathbf{J}^\top\mathbf{F} = \mathbf{0}$. We therefore first project it onto this condition, iterating Eq. (15) at fixed ξ_0 until $\|\mathbf{J}^\top\mathbf{F}\|$ falls below a tolerance or stops decreasing. We then integrate Eq. (17) along the straight segment $\xi(s) = \xi_0 + s(\xi^\star - \xi_0)$, $s \in [0, 1]$, which gives B coupled ODEs,

$$\frac{dw^k}{ds} \;=\; -\left(H_{ww}^{-1}\right)^{kj}\left(H_{w\xi}\right)_{ja}\frac{d\xi^a}{ds}\,. \tag{20}$$

We integrate them with the fourth-order Runge–Kutta method (RK4) in a fixed number of steps, reassembling H_{ww} and $H_{w\xi}$ at every stage and solving the linear system instead of inverting H_{ww}. As long as the path stays smooth and H_{ww} invertible, the endpoint does not depend on the chosen curve, which only affects the discretization error. In the multi-EP setting, $\xi = \nu$ and Eq. (20) reduces to $dw^k/d\nu = -(H_{ww}^{-1})^{kj}(H_{w\nu})_j$.

Unlike GN, the continuation returns a curve. Up to the integration error, every intermediate head $\mathbf{w}(s)$ is a stationary point of Φ at its own instance $\xi(s)$, on the branch of the start head, and is therefore a transferred head for that instance; the derivative $\partial\mathbf{w}^\star/\partial\xi$ is available at every point. GN reaches the same endpoint whenever both methods land on the same stationary point, which is the case on all multi-EP targets and most multi-IC ones (see Appendix B.4), but recovering the intermediate heads would require one least-squares solve per intermediate instance.

5 Results

We compare the two second-order methods with the *Adam* baseline in accuracy and cost, and examine the weight-space paths returned by the continuation. For each PDE we train two MH models, eight in total: one over EP values ($\xi = \nu$, $d_\xi = 1$) with a fixed IC, with $D = 16$ heads and $B = 8$ basis functions, and one over ICs ($\xi = \mathbf{z}$, $d_\xi = 2$) with a fixed EP, with $D = 100$ and $B = 48$. Each TL method is evaluated on 15 unseen instances $\{\xi_k^\star\}_{k=1}^{15}$. GN iterates up to convergence, the continuation takes 4 RK4 steps from ξ_0 to $\xi^\star$, and *Adam* runs for the wall-clock time of the slower second-order method. We report the median relative L^2 error of the transferred solution against analytic (heat, wave) or finite-difference (Burgers, Allen–Cahn) references, and the median wall time per transfer.[3] Training settings, metric definitions, the timing protocol, ablations, and validation and extrapolation results are given in Appendix B.

Table 1: TL to unseen $\xi^\star$ (15 targets per model): median relative L^2 error against the reference solution and median wall time (online) per transfer, on CPU. The *Adam* baseline runs for the wall-clock time of the slower second-order method (the Hessian continuation).

		Gauss–Newton (GN)		**Hessian + RK4**		**Baseline (*Adam*)** equal longest time (Hessian)
	PDE	rel. L^2	ms	rel. L^2	ms	rel. L^2
EP	heat	2.8×10^{-3}	2.2	2.8×10^{-3}	10	7.1×10^{-3}
	wave	4.4×10^{-2}	2.8	4.4×10^{-2}	14	1.5×10^{-1}
	Allen–Cahn	4.1×10^{-3}	7.3	4.1×10^{-3}	21	4.2×10^{-3}
	Burgers	3.7×10^{-2}	8.2	3.7×10^{-2}	26	4.7×10^{-2}
IC	heat	1.6×10^{-3}	18	1.6×10^{-3}	116	1.6×10^{-2}
	wave	7.2×10^{-3}	20	7.2×10^{-3}	120	4.4×10^{-1}
	Allen–Cahn	2.1×10^{-2}	68	5.2×10^{-2}	203	2.5×10^{-1}
	Burgers	7.4×10^{-2}	60	7.4×10^{-2}	248	3.0×10^{-1}

[3] Medians are reported since in the multi-IC setting 3 Burgers and 2 Allen–Cahn targets land on a wrong stationary point of Φ for every TL method, with errors above 0.3. This is a limitation of the nonconvex head problem rather than of any solver, and one that would dominate the means (Appendix B.1).

5.1 ACCURACY AND COST

At equal wall-clock time, the median error of the second-order methods matches or improves on that of the *Adam* baseline in all eight experiments (Table 1): it is a median of 3.7 times smaller, and up to 61 times smaller for the wave equation with varying IC. Only for Allen–Cahn with varying EP are the three methods within a few percent of each other. The gains are largest in the IC experiments, which have a larger basis and a two-dimensional instance space, and are slower for all methods. The errors on unseen instances are comparable to those of the trained heads at known instances (Appendix B), so the error floor is set by the frozen basis rather than by the TL method. Figure 2 shows one transfer for the Burgers equation with varying IC, the most demanding case; the other equations are shown in Appendix C.

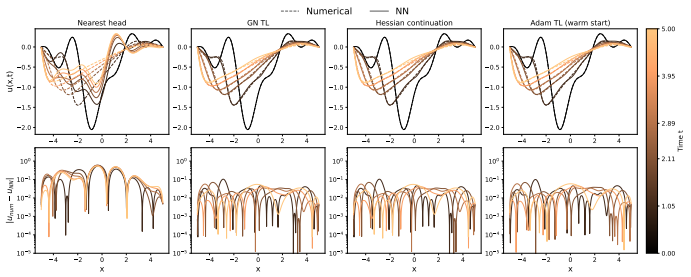


Figure 2: TL to an unseen IC for the Burgers equation: solutions (top row) and absolute error against the numerical reference (bottom row). From left to right: the nearest trained head $\mathbf{w}_0(\mathbf{z}_0)$, used as starting point by all methods, and the new heads from GN (Section 4.2.2), the Hessian continuation (Section 4.2.3) and *Adam* (Section 4.2.1).

5.2 METHOD COMPARISON: INTERPRETABILITY OF THE WEIGHT SPACE

The two second-order methods reach the same accuracy in seven of the eight experiments, and GN is 2.9 to 6.4 times cheaper, since at fixed ξ it never assembles $\partial\mathbf{F}/\partial\xi$ (which in the IC case includes the decoder Jacobian) or the second-order blocks. The two methods, however, return different objects. GN returns only the head at $\xi^\star$. The continuation returns the path $\mathbf{w}^\star(\xi(s))$ from ξ_0 to $\xi^\star$, each point of which solves its own intermediate instance, together with $\partial\mathbf{w}^\star/\partial\xi$ along it.

Figure 3 shows these paths in the multi-EP setting. Figure 4 shows them for the Burgers equation with varying IC, projected on the leading right singular vector $\mathbf{v}_1$ of $\mathbf{J}$ (see Appendix D). The surface interpolates the trained heads and all points of the continuation paths, which satisfy Eq. (16). Up to the RK4 error and the finite expressivity of the basis, this is therefore the solution of the TL problem over the whole range of ICs parametrized by $\mathbf{z}$, or in other words a map of the optimal head over the latent IC space in the coordinates of Section 4.1.3. Interactive 3D versions of these weight spaces, with the corresponding solutions, are provided in the supplementary material below.

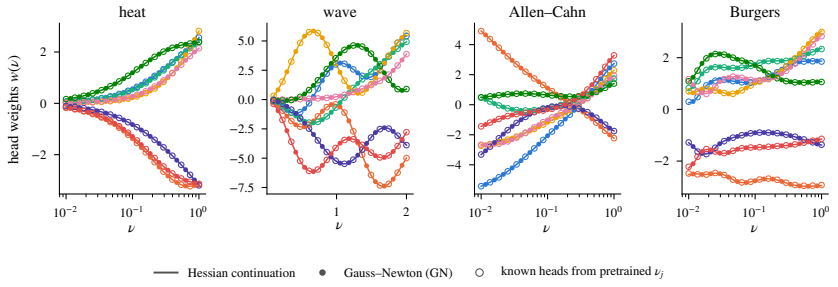


Figure 3: Weight space $\mathbf{w}(\nu)$ of the multi-EP models. Each color curve corresponds to one latent basis function. Lines: Hessian continuation integrated across the trained range of ν. Open circles: trained heads at the ν_j. Filled dots: GN at the 15 unseen midpoints $\nu^\star$. The continuation passes through both.

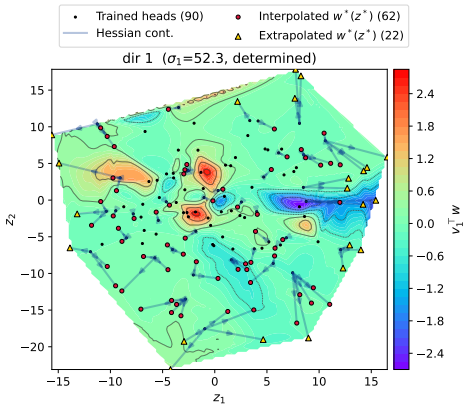


Figure 4: Weight space $\mathbf{w}(\mathbf{z})$ for the Burgers equation, projected on the leading direction $\mathbf{v}_1$ (color contours of $\mathbf{v}_1^\top \mathbf{w}$). Black points: trained heads. Red circles and yellow triangles: transferred heads $\mathbf{w}^\star(\mathbf{z}^\star)$ for 62 unseen ICs inside the training domain of $\mathbf{z}$ (interpolation) and 22 outside it (extrapolation; Appendix B.3). Blue arrows: Hessian continuation paths from the nearest trained head $\mathbf{w}_0$ to $\mathbf{w}^\star$.

6 Conclusions

Setup and findings. We have shown that, once a MH PINN has learned a shared basis for a family of PDE solutions, the head of an unseen instance can be obtained with second-order methods, for linear and nonlinear equations alike. Since only the head is adapted, the problem has as many unknowns as basis functions, and the exact Jacobian and Hessian are cheap to assemble. GN solves the nonlinear least-squares problem for the head directly and extends OTL to nonlinear PDEs without perturbative expansions. The Hessian continuation follows the minimizer from a trained instance to the target and returns, in addition, the path of heads in between and their derivative with respect to the EP or IC. On the heat, wave, Burgers and Allen–Cahn equations, with varying EPs and ICs, both methods obtain a new head in at most a few hundred milliseconds on a CPU and, at equal wall-clock time, reach errors a median of 3.7 times smaller than first-order head training, and up to 61 times smaller.

The two methods serve different purposes. When only the solution at the target is required, GN is the natural choice, being 2.9 to 6.4 times cheaper than the Hessian continuation and equally accurate in seven of the eight experiments. When the dependence of the solution on the EP or IC is of interest, the continuation provides it in a single integration, and the orthonormality of the basis makes the resulting paths interpretable as paths of the network output in a common L^2 frame.

Limitations. All experiments are in one spatial dimension with a low-dimensional instance space ($d_\xi \leq 2$), and the accuracy of every transferred head is bounded by how well the frozen basis represents the family, rather than by the TL method. The continuation also assumes that the Hessian of the loss with respect to the head remains invertible along the path, and it follows the branch of its start head. Natural extensions are higher-dimensional domains, joint variations of EPs and ICs, larger IC spaces, and the use of the continuation paths to study the geometry of the solution manifold.

AI use statement

In this work, we used generative AI tools (Claude, Anthropic) to assist with programming, including the implementation of the methods, to give feedback on the consistency of the manuscript, and to formulate two remarks in Appendix A: the relation between prior fixed-basis solvers and Gauss–Newton, and the condition-number estimate of the head problem. We have not used generative AI tools for the other tasks with required disclosure. Additionally, we used them to search and summarize the literature, check and format references, and draft and edit parts of the text. We have reviewed all AI-assisted work: the code was verified and tested by the authors, every reference was

checked against its source, and both remarks were verified. We take responsibility for the final content of this work.

Ethics statement

The authors have read and adhere to the ICLR Code of Ethics. This work develops numerical methods for PDEs and uses only synthetic data, generated from analytic and finite-difference solutions of the heat, wave, Burgers and Allen–Cahn equations. It involves no human subjects, personal data or sensitive information, and we are not aware of potential harms specific to it.

Reproducibility statement

Section 4 gives the complete formulation of the pipeline: the reparametrization that imposes the IC and BCs (Eq. 6), the training losses (Eqs. 7, 10 and 12), the IC decoder (Eq. 8), the least-squares problem for the head (Eq. 14), the GN iteration (Eq. 15), and the continuation (Eqs. 17–20), including the projection of the initial head. Section 5 specifies the evaluation protocol: the number of heads and basis functions, the number of unseen targets, the GN tolerance, the number of RK4 steps, the time budget of the baseline and the error metric. Appendix B gives the architectures, sampling, timing protocol and baseline settings, and Appendix C documents the quality of the trained models. The supplementary material contains interactive visualizations of the weight spaces, as well as the used code released anonymously in the repository Weight_Space_Flows.

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.

Eugene L. Allgower and Kurt Georg. *Introduction to Numerical Continuation Methods*, volume 45 of *Classics in Applied Mathematics*. SIAM, 2003.

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.

Gbenga T. Awojinrin, Abdul-Akeem Olawoyin, and Rami M. Younis. A convex quasilinearization method for solving nonlinear PDEs with physics-informed neural networks. *arXiv preprint arXiv:2606.18175*, 2026.

Mathieu Blondel, Quentin Berthet, Marco Cuturi, Roy Frostig, Stephan Hoyer, Felipe Llinares-López, Fabian Pedregosa, and Jean-Philippe Vert. Efficient and modular implicit differentiation. In *Advances in Neural Information Processing Systems*, volume 35, 2022.

Andrea Bonfanti, Giuseppe Bruno, and Cristina Cipriani. The challenges of the nonlinear regime for physics-informed neural networks. In *Advances in Neural Information Processing Systems*, volume 37, 2024.

Jingrun Chen, Xurong Chi, Weinan E, and Zhouwang Yang. Bridging traditional and machine learning-based algorithms for solving PDEs: The random feature method. *Journal of Machine Learning*, 1(3):268–298, 2022.

Woojin Cho, Kookjin Lee, Donsub Rim, and Noseong Park. Hypernetwork-based meta-learning for low-rank physics-informed neural networks. In *Advances in Neural Information Processing Systems*, volume 36, pp. 11219–11231, 2023.

Eric C. Cyr, Mamikon A. Gulian, Ravi G. Patel, Mauro Perego, and Nathaniel A. Trask. Robust training and initialization of deep neural networks: An adaptive basis viewpoint. In *Proceedings of The First Mathematical and Scientific Machine Learning Conference*, volume 107 of *PMLR*, pp. 512–536, 2020.

Felix Dangel, Johannes Müller, and Marius Zeinhofer. Kronecker-factored approximate curvature for physics-informed neural networks. In *Advances in Neural Information Processing Systems*, volume 37, pp. 34582–34636, 2024.

Chinmay Datar, Taniya Kapoor, Abhishek Chandra, Qing Sun, Erik Lien Bolager, Iryna Burak, Anna Veselovska, Massimo Fornasier, and Felix Dietrich. Fast training of accurate physics-informed neural networks without gradient descent. In *International Conference on Learning Representations*, 2026. arXiv:2405.20836.

Filipe de Avila Belbute-Peres, Yi-fan Chen, and Fei Sha. HyperPINN: Learning parameterized differential equations with physics-informed hypernetworks. In *The Symbiosis of Deep Learning and Differential Equations (NeurIPS 2021 Workshop)*, 2021.

Shaan Desai, Marios Mattheakis, Hayden Joy, Pavlos Protopapas, and Stephen Roberts. One-shot transfer learning of physics-informed neural networks, 2022. URL `https://arxiv.org/abs/2110.11286`.

M. W. M. G. Dissanayake and N. Phan-Thien. Neural-network-based approximations for solving partial differential equations. *Communications in Numerical Methods in Engineering*, 10(3):195–201, 1994.

Suchuan Dong and Zongwei Li. Local extreme learning machines and domain decomposition for solving linear and nonlinear partial differential equations. *Computer Methods in Applied Mechanics and Engineering*, 387:114129, 2021.

Vikas Dwivedi and Balaji Srinivasan. Physics informed extreme learning machine (PIELM) – a rapid method for the numerical solution of partial differential equations. *Neurocomputing*, 391: 96–118, 2020.

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

Yihang Gao, Ka Chun Cheung, and Michael K. Ng. SVD-PINNs: Transfer learning of physics-informed neural networks via singular value decomposition. In *2022 IEEE Symposium Series on Computational Intelligence (SSCI)*, pp. 1443–1450, 2022.

G. H. Golub and C. F. Van Loan. *Matrix Computations*. Johns Hopkins University Press, 4 edition, 2013.

Somdatta Goswami, Cosmin Anitescu, Souvik Chakraborty, and Timon Rabczuk. Transfer learning enhanced physics informed neural network for phase-field modeling of fracture. *Theoretical and Applied Fracture Mechanics*, 106:102447, 2020.

Somdatta Goswami, Katiana Kontolati, Michael D. Shields, and George Em Karniadakis. Deep transfer operator learning for partial differential equations under conditional shift. *Nature Machine Intelligence*, 4(12):1155–1164, 2022.

P. C. Hansen. *Rank-Deficient and Discrete Ill-Posed Problems: Numerical Aspects of Linear Inversion*. SIAM, 1998.

Jianguo Huang and Haohao Wu. The neural network basis method for nonlinear partial differential equations and its Gauss–Newton optimizer. *Communications in Nonlinear Science and Numerical Simulation*, 143:108608, 2025. doi: 10.1016/j.cnsns.2025.108608.

Yao Huang, Wenrui Hao, and Guang Lin. HomPINNs: Homotopy physics-informed neural networks for learning multiple solutions of nonlinear elliptic differential equations. *Computers & Mathematics with Applications*, 121:62–73, 2022.

Raul Jimenez, Svitlana Mayboroda, Pavlos Protopapas, Leonid Sarieddine, David N. Spergel, and Pedro Tarancón-Álvarez. Physics-informed neural embeddings of PDE solution families. *arXiv preprint arXiv:2607.06348*, 2026.

Anas Jnini and Flavio Vella. Dual natural gradient descent for scalable training of physics-informed neural networks. *Transactions on Machine Learning Research*, 2025. arXiv:2505.21404.

Anas Jnini, Flavio Vella, and Marius Zeinhofer. Gauss–Newton natural gradient descent for physics-informed computational fluid dynamics. *Computers & Fluids*, pp. 106955, 2025. arXiv:2402.10680.

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.

Diederik P. Kingma and Max Welling. Auto-encoding variational Bayes. In *International Conference on Learning Representations*, 2014. arXiv:1312.6114.

Aditi S. Krishnapriyan, Amir Gholami, Shandian Zhe, Robert M. Kirby, and Michael W. Mahoney. Characterizing possible failure modes in physics-informed neural networks. In *Advances in Neural Information Processing Systems*, volume 34, pp. 26548–26560, 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. doi: 10.1109/72.712178.

Wanzhou Lei, Pavlos Protopapas, and Joy Parikh. One-shot transfer learning for nonlinear ODEs. In *NeurIPS 2023 Workshop on The Symbiosis of Deep Learning and Differential Equations*, 2023. arXiv:2311.14931.

Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Fourier neural operator for parametric partial differential equations. In *International Conference on Learning Representations*, 2021a.

Zongyi Li, Hongkai Zheng, Nikola Kovachki, David Jin, Haoxuan Chen, Burigede Liu, Kamyar Azizzadenesheli, and Anima Anandkumar. Physics-informed neural operator for learning partial differential equations. *arXiv preprint arXiv:2111.03794*, 2021b. Published in ACM/IMS Journal of Data Science, doi:10.1145/3648506.

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.

E. H. Moore. On the reciprocal of the general algebraic matrix. *Bulletin of the American Mathematical Society*, 26(9):394–395, 1920. doi: 10.1090/S0002-9904-1920-03322-7.

Johannes Müller and Marius Zeinhofer. Achieving high accuracy with PINNs via energy natural gradient descent. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *PMLR*, pp. 25471–25485, 2023.

Abdul Hannan Mustajab, Hao Lyu, Zarghaam Rizvi, and Frank Wuttke. Physics-informed neural networks for high-frequency and multi-scale problems using transfer learning. *Applied Sciences*, 14(8):3204, 2024.

Raphaël Pellegrin, Blake Bullwinkel, Marios Mattheakis, and Pavlos Protopapas. Transfer learning with physics-informed neural networks for efficient simulation of branched flows. *arXiv preprint arXiv:2211.00214*, 2022.

R. Penrose. A generalized inverse for matrices. *Mathematical Proceedings of the Cambridge Philosophical Society*, 51(3):406–413, 1955. doi: 10.1017/S0305004100030401.

Michael Penwarden, Shandian Zhe, Akil Narayan, and Robert M. Kirby. A metalearning approach for physics-informed neural networks (PINNs): Application to parameterized PDEs. *Journal of Computational Physics*, 477:111912, 2023.

M. Raissi, P. Perdikaris, and G. E. Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686–707, 2019.

Yiqi Rao and Pavlos Protopapas. Chebyshev-augmented one-shot transfer learning for PINNs on nonlinear differential equations. *arXiv preprint arXiv:2605.01634*, 2026.

Pratik Rathore, Weimu Lei, Zachary Frangella, Lu Lu, and Madeleine Udell. Challenges in training PINNs: A loss landscape perspective. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *PMLR*, pp. 42159–42191, 2024.

Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *PMLR*, pp. 1278–1286, 2014.

Emilien Seiler, Wanzhou Lei, and Pavlos Protopapas. Stiff transfer learning for physics-informed neural networks. *arXiv preprint arXiv:2501.17281*, 2025.

Shashank Subramanian, Peter Harrington, Kurt Keutzer, Wahid Bhimji, Dmitriy Morozov, Michael W. Mahoney, and Amir Gholami. Towards foundation models for scientific machine learning: Characterizing scaling and transfer behavior. In *Advances in Neural Information Processing Systems*, volume 36, pp. 71242–71262, 2023.

Pedro Tarancón-Álvarez, Pablo Tejerina-Pérez, Raul Jimenez, and Pavlos Protopapas. Efficient pinns via multi-head unimodular regularization of the solutions space. *Communications Physics*, 8(1):335, 2025.

Sifan Wang, Hanwen Wang, and Paris Perdikaris. Learning the solution operator of parametric partial differential equations with physics-informed DeepONets. *Science Advances*, 7(40):eabi8605, 2021.

Chen Xu, Ba Trung Cao, Yong Yuan, and Günther Meschke. Transfer learning based physics-informed neural networks for solving inverse problems in engineering structures under different loading scenarios. *Computer Methods in Applied Mechanics and Engineering*, 405:115852, 2023.

Wuzhe Xu, Yulong Lu, and Li Wang. Transfer learning enhanced DeepONet for long-time prediction of evolution equations. *arXiv preprint arXiv:2212.04663*, 2022.

Zichuan Yang. MetaColloc: Optimization-free PDE solving via meta-learned basis functions. *arXiv preprint arXiv:2605.12368*, 2026.

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

A Extended related work

Fixed-basis solvers and Gauss–Newton. Awojinrin et al. (2026) linearize the PDE about the current iterate and solve each linear problem by least squares on a trial space that is linear in its parameters. For a fixed basis this coincides with a Gauss–Newton step on the coefficients (Huang & Wu, 2025), i.e. with Eq. 15 for $\lambda = 1$. Huang & Wu (2025) use this Gauss–Newton optimizer to solve nonlinear PDEs on a basis given by shallow neural networks. Dong & Li (2021) use the same linearize-and-solve iteration and a nonlinear least-squares solver with random features, and Cyr et al. (2020) view the hidden layers of a trained network as an adaptive basis whose last layer is obtained by least squares. None of these works trains the basis with one head per instance of a PDE family, and thus the coefficients cannot be interpreted as coordinates in the latent space.

Conditioning of the head problem. The singular values σ_i of $\mathbf{J}$ span one to two orders of magnitude (Appendix D). The condition number of $\mathbf{J}$, defined as $\kappa(\mathbf{J}) = \sigma_{\max}/\sigma_{\min}$, is then $\kappa(\mathbf{J}) \sim 10$–$10^2$, and that of the Gauss–Newton matrix is $\kappa(\mathbf{J}^\top\mathbf{J}) = \kappa(\mathbf{J})^2 \sim 10^2$–$10^4$. The head problem thus inherits, in a milder form, the poor conditioning of the PINN loss (Krishnapriyan et al., 2021; Rathore et al., 2024): the convergence rate of gradient descent is governed by this ratio, whereas Gauss–Newton and the Hessian continuation invert the curvature directly.

Continuation. The Hessian continuation of Section 4.2.3 follows the head as the EP or IC is changed gradually from a trained instance to the target. This is a standard technique in numerical analysis, known as natural-parameter continuation (Allgower & Georg, 2003). It works as long as the Hessian of the loss with respect to the head weights, H_{ww} in Eq. (18), stays invertible along the path. If it becomes singular, the path can turn back, and the EP or IC can no longer be used to follow it; standard remedies exist for this case, such as pseudo-arclength continuation (Allgower & Georg, 2003). The main difference with the classical setting is that there the residual is driven to zero, whereas here the residual cannot vanish ($N \gg B$), so we follow the minimum of the loss instead. For this reason the full Hessian of the loss enters Eq. (17), and not only its Gauss–Newton approximation $\mathbf{J}^\top\mathbf{J}$. The same derivative of the head with respect to the EP or IC is what implicit differentiation computes for general optimization problems (Blondel et al., 2022). In PINNs, continuation has so far been used during training, e.g. as curriculum regularization (Krishnapriyan et al., 2021) or as homotopy methods for multiple solutions (Huang et al., 2022), but, to our knowledge, not in the weight space of a frozen body.

B Implementation details

This appendix collects the implementation details, architectures, sampling, metric definitions, timing protocol and baseline settings referenced in Section 5, the ablation on the starting point of Gauss–Newton, the validation of the transferred heads against the trained ones, and the extrapolation results.

B.1 Metrics

Relative L^2 error. All accuracies are $\epsilon(\mathbf{w}) = \|u(\mathbf{w}) - u_{\mathrm{ref}}\|_2/\|u_{\mathrm{ref}}\|_2$ for the field u of Eq. 6 on a fixed space–time grid: 201×101 uniform points in $x \in [-5, 5]$, $t \in [0, 5]$ (multi-EP) and the 1000×20 grid of the IC bank (multi-IC). None of the transfer methods sees u_{ref}.

Reference solutions. Multi-EP: closed form for heat and wave with $u_0 = \sin(\kappa x)$, $\kappa = 2\pi/5$, i.e. $u = \sin(\kappa x)\, e^{-\nu\kappa^2 t}$ and $u = \sin(\kappa x)\cos(\nu\kappa t)$; explicit finite differences for Burgers and Allen–Cahn (1601 points, forward Euler with $\Delta t = 0.4\,\Delta x^2/\nu$ capped at 10^{-3}, first-order upwind advection). Multi-IC: the solver that generated the bank, on its 1000-point grid: Crank–Nicolson (heat, $\Delta t = 0.5\,\Delta x^2/\nu$), leapfrog (wave, Courant number 0.5), Crank–Nicolson diffusion with second-order upwind advection (Burgers, Courant number 0.4) and with an explicit reaction term (Allen–Cahn, $\Delta t = 10^{-3}$).

Statistic. Medians over the 15 targets of each model, for errors and wall times. In the multi-IC setting 3 Burgers and 2 Allen–Cahn targets land on a wrong stationary point of Φ for every method (errors above 0.3) and would dominate the means; the multi-EP setting has no such target.

B.2 ARCHITECTURES AND TRAINING SETTINGS

Table 2 lists the models. All are trained on one GPU with the PINN-only loss of Eq. 12; the IC and BCs are imposed by Eq. 6 with $u_{0,j}(\pm 5) = 0$ and $u_t(x, 0) = 0$ for the wave equation. In the multi-EP models each head receives its own Latin-hypercube batch of collocation points, resampled at every step. In the multi-IC models the 10^5 points of an epoch are uniform in (x, t) and assigned to random heads, the derivatives u_0', u_0'' of the particular part are exact from the sine series, and the Gram matrix of Eq. 9 is estimated on 4096 fresh points per epoch. The last 10 heads of each multi-IC model are never used as starting points; they serve as held-out heads in Section B.8.

Table 2: Architecture and training settings. EP values and trained-head errors are listed in the order heat / wave / Allen–Cahn / Burgers; the trained-head error is the median relative L^2 error of the jointly trained heads at their own instance.

	multi-EP	**multi-IC**
instance label ξ	ν ($d_\xi = 1$)	$\mathbf{z} \in \mathbb{R}^2$ ($d_\xi = 2$)
domain	$x \in [-5, 5]$, $t \in [0, 5]$, $u(\pm 5, t) = 0$	
heads D	16	100
basis size B	8	48
body $\mathbf{H}_\theta$	5×128, tanh	4×128, tanh
EP values ν	16 log-spaced in $[10^{-2}, 1]$	0.1 / 1 / 0.5 / 0.2
(wave)	16 linear in $[0.1, 2]$	
initial conditions	$u_0 = \sin(2\pi x/5)$, all heads	$\mathcal{D}(\mathbf{z}_j)$, $K = 10$ (Allen–Cahn: 4)
collocation points	1024/head/step, Latin hypercube	10^5/epoch, uniform in (x, t)
orthonormality $\lambda_\perp$	10^{-3}	2×10^{-2}
optimiser	Adam 10^{-3}, $\times 0.955/10^3$ steps	Adam 10^{-3}, $\times 0.985/10^3$ epochs
steps / epochs	10^5	10^5
seed	0	41
wall time (one GPU)	34–42 min	30–45 min
trained-head error ($\times 10^{-3}$)	3.0 / 43 / 4.5 / 37	2.2 / 7.1 / 7.3 / 48

IC decoder and bank. The VAE of Section 4.1.2 is shared by the four equations: an encoder of three 1-D convolutions (32, 64, 128 channels, kernel 5, stride 2) and a 128-unit layer to $(\mu, \log \sigma^2) \in \mathbb{R}^2$, and the decoder MLP $2 \to 256 \to 256 \to K$ with SiLU activations, $K = 10$, $n_x = 1000$. It is trained on 4000 profiles vanishing at the walls (random superpositions of Gaussian, Lorentzian and sech bumps, chirps, sigmoids and triangles, and random Fourier–polynomial–wavelet series) for 1500 epochs with $\beta = 10^{-2}$ (200-epoch warm-up), seed 123; its aggregate posterior has mean $\simeq (-0.5, -0.6)$ and standard deviation $\simeq (6.0, 5.9)$. The bank is 100 latents drawn from this Gaussian (seed 333) and accepted if $0.15 \le \max_x |u_0| \le 3.5$, the same for the four equations; for Allen–Cahn the decoder is truncated to $K = 4$ modes and $\nu = 0.5$, which keeps the fronts resolved on the grid (with $K = 10$ and $\nu = 0.1$ plain PINN training did not converge). Each IC is solved numerically on 1000 points and 20 times in $[0, 5]$.

B.3 SAMPLING OF THE UNSEEN TARGETS

Multi-EP. The 15 targets are the midpoints between consecutive trained values, $\sqrt{\nu_k \nu_{k+1}}$ for the log-spaced grids and $(\nu_k + \nu_{k+1})/2$ for the wave equation; the start is the nearest trained head, half a spacing away (0.15 in $\ln \nu$; 0.063 in ν for wave).

Multi-IC. The 15 targets are new latents from the aggregate Gaussian of the bank with the same amplitude screen (seeds 11 to 25, shared by the four equations), solved numerically; the start is the nearest of the 90 training heads in $\mathbf{z}$ (median distance 1.07, maximum 2.08; median spacing between trained latents 1.08). The unseen ICs of Figures 4 and 8 are instead uniform in the bounding box of the trained latents enlarged by 20% per side, so that a quarter to a third of them fall outside the training box.

B.4 SECOND-ORDER METHODS

Both methods use the residual of Eq. 13 on a fixed collocation set on which the frozen body and the envelope are evaluated once per model: 80×80 interior points of a uniform grid in double precision (multi-EP) and the 1000×20 bank grid in single precision (multi-IC). All fields are affine in $\mathbf{w}$, so $\mathbf{F}$, $\mathbf{J}$, $\partial\mathbf{F}/\partial\xi$ and the blocks of Eq. 18 are dense $N \times B$ linear algebra, with the pointwise derivatives of the residual from automatic differentiation; all blocks were checked against finite differences.

Gauss–Newton. $\lambda = 1$. Each step is a least-squares solve with the singular values below $r_{\mathrm{cond}}\,\sigma_{\max}$ discarded, $r_{\mathrm{cond}} = 10^{-8}$ (multi-IC) and machine precision (multi-EP). It stops when $\|\mathbf{F}\|$ changes by less than 10^{-10} (multi-EP) or 10^{-6} (multi-IC, the single-precision floor) between iterates, or after 30 iterations; for Burgers and Allen–Cahn the step is halved (at most 8 times) if it would increase $\|\mathbf{F}\|$. The linear equations converge in one step; the nonlinear ones take 12 to 13 evaluations of $\mathbf{F}$ and $\mathbf{J}$ (multi-EP) and a median of 6 steps (multi-IC).

Hessian continuation. The start head is projected onto $\mathbf{J}^\top\mathbf{F} = \mathbf{0}$ at ξ_0 by Gauss–Newton (multi-EP: until $\|\mathbf{F}\|$ changes by less than 10^{-12}; multi-IC: until $\|\mathbf{J}^\top\mathbf{F}\| < 10^{-10}$ or stagnates; at most 20 steps), which moves it by 0.03 to 0.2% (multi-EP) and 3 to 11% (multi-IC) of its norm. Eq. 20 is then integrated with RK4 in 4 sub-steps along the straight segment (16 tangent evaluations), each solving the $B \times B$ system in double precision by least squares (multi-EP) or by a symmetric eigendecomposition with eigenvalues below $10^{-12}\lambda_{\max}$ discarded (multi-IC); no corrector. Four sub-steps reproduce the 16-sub-step endpoint to three digits in the median and coincide with the Gauss–Newton head to better than 10^{-5} in weight space, except on 3 Burgers and 8 Allen–Cahn multi-IC targets, where the continuation ends on a different stationary point than Gauss–Newton (a worse one on 6 of the Allen–Cahn targets) independently of the number of sub-steps: the continuation follows the branch of the start head, whereas the Gauss–Newton iteration can jump to another basin. This is the origin of the Allen–Cahn difference in Table 1. The projection takes 23 to 29% (multi-EP) and 31 to 43% (multi-IC) of the continuation's wall time and depends only on the start head.

B.5 TIMING PROTOCOL

All transfers are timed on one CPU (Apple M3, 8 cores, 24 GB; PyTorch 2.4, single process), in the same process and on the same precomputed system for the three methods. Reported times are optimisation time only: the assembly of $\mathbf{F}$ and its derivatives and the linear algebra, plus the decoding of the target IC in the multi-IC case. Excluded are the one-off evaluation of the body on the grid (seconds per model), the periodic evaluation of the error that produces the Adam traces, and any diagnostic. Gauss–Newton and the continuation are deterministic and are timed three times after a warm-up call (median); Adam is run once. One Adam step costs 0.22 to 0.37 ms (multi-EP, 2048-row minibatches) and 1.0 to 1.4 ms (multi-IC, all 20 000 rows). Training on the cluster GPU is not part of any reported time.

B.6 FIRST-ORDER BASELINE

Settings. Adam on the mean squared residual of Eq. 14 on the same collocation system, body frozen, from the nearest trained head: learning rate 10^{-2} decayed exponentially to 5×10^{-4} with minibatches of 2048 rows (multi-EP) and to 10^{-3} on the full residual (multi-IC), 2000 steps, no early stopping, seed 0.

Equal time. The error of the Adam iterate is recorded every 10 steps (evaluation excluded from the clock); the "equal time" entry of Table 1 is its value at the last recorded step within the wall time of the Hessian continuation on the same target (30 to 50 steps in the multi-EP setting, 80 to 170 in the multi-IC one), median over the targets.

No fixed budget. Table 3: for each target, exponential schedules of 100 to 5000 steps (20 000 for the wave ICs), stopped at the first iterate at or below the Gauss–Newton error (a best case for Adam, since it uses the reference) and, as an observable rule, at the iterate where the residual has improved by less than 1% over 200 steps on the longest schedule. Adam reaches the Gauss–Newton error

on every multi-EP target after 7 to 25 times the Gauss–Newton wall time, but on 6, 10, 4 and 10 of the 15 multi-IC targets only, and its converged error stays above Gauss–Newton's for the wave, Allen–Cahn and Burgers ICs.

Table 3: Adam (warm start) without a fixed budget. "To GN accuracy": first iterate at or below the Gauss–Newton error of the same target, best over exponential schedules of 100 to 5000 steps (20 000 for the wave ICs); median steps and wall time over the targets that reach it, and how many of the 15 do (strictly / within 5%). "Converged": iterate at which the residual has improved by less than 1% over the previous 200 steps on the longest schedule. Wall times are steps times the measured cost per step; the Gauss–Newton column is that of Table 1.

		Gauss–Newton	**Adam (warm start)**					
			to GN accuracy			converged		
	PDE	rel. L^2	steps	ms	reached	rel. L^2	steps	ms
EP	heat	2.8×10^{-3}	70	20	15/15	2.8×10^{-3}	270	85
	wave	4.4×10^{-2}	250	81	15/15	4.4×10^{-2}	440	152
	Allen–Cahn	4.1×10^{-3}	60	23	15/15	4.2×10^{-3}	280	106
	Burgers	3.7×10^{-2}	280	109	13/13	3.7×10^{-2}	390	212
IC	heat	1.6×10^{-3}	210	208	6/14	1.6×10^{-3}	2520	2495
	wave	7.2×10^{-3}	3050	4362	10/14	6.9×10^{-3}	5470	7822
	Allen–Cahn	2.1×10^{-2}	60	79	4/5	5.7×10^{-2}	3910	5161
	Burgers	7.4×10^{-2}	340	398	10/15	8.1×10^{-2}	1180	1381

B.7 Ablation: Gauss–Newton from $\mathbf{w}^{(0)} = \mathbf{0}$

Table 4 repeats the transfers with Gauss–Newton started from $\mathbf{w}^{(0)} = \mathbf{0}$. The cold start reaches the same minimiser on every multi-EP target and on all but 1 (Burgers) and 3 (Allen–Cahn) of the multi-IC targets, with 25 to 40% more evaluations (multi-EP) and one to two more steps (multi-IC); the three Allen–Cahn targets land on a wrong stationary point (errors of 0.5 to 0.6), which doubles the median of that row. Adam from $\mathbf{0}$ is 2 to 18 times worse than from the nearest head, except for the heat equation.

Table 4: Gauss–Newton from the nearest trained head (as in Table 1) and from $\mathbf{w}^{(0)} = \mathbf{0}$ on the same 15 targets per model: median relative L^2 error and cost, as the mean number of evaluations of $\mathbf{F}$ and $\mathbf{J}$ (multi-EP) or the median number of steps (multi-IC). Adam from $\mathbf{0}$: median error after 1500 steps (multi-EP) and 2000 steps (multi-IC); from the nearest head it reaches 2.8×10^{-3}, 4.4×10^{-2}, 4.1×10^{-3}, 4.1×10^{-2} (multi-EP) and 1.8×10^{-3}, 4.5×10^{-2}, 8.4×10^{-2}, 1.0×10^{-1} (multi-IC).

		GN from nearest head		**GN from $\mathbf{w}^{(0)} = \mathbf{0}$**		**Adam from 0**
	PDE	rel. L^2	evals	rel. L^2	evals	rel. L^2
EP	heat	2.8×10^{-3}	4	2.8×10^{-3}	4	3.2×10^{-3}
	wave	4.4×10^{-2}	4	4.4×10^{-2}	4	7.8×10^{-1}
	Allen–Cahn	4.1×10^{-3}	12.5	4.1×10^{-3}	17.1	1.2×10^{-2}
	Burgers	3.7×10^{-2}	13.3	3.7×10^{-2}	16.7	9.1×10^{-2}
IC	heat	1.6×10^{-3}	2	1.6×10^{-3}	2	1.9×10^{-3}
	wave	7.2×10^{-3}	2	7.2×10^{-3}	2	1.2×10^{-1}
	Allen–Cahn	2.1×10^{-2}	6	4.1×10^{-2}	8	3.4×10^{-1}
	Burgers	7.4×10^{-2}	6	7.4×10^{-2}	7	1.5×10^{-1}

B.8 Validation on the trained heads

Table 5: Gauss–Newton at a trained instance, started from the nearest *other* head (leave-one-out; the 16 trained ν of the multi-EP models and the 10 held-out heads of the multi-IC models), reaches the error of the jointly trained head, with a median ratio of 0.89 to 1.06. The errors at the unseen targets of Table 1 equal those of the trained heads, except for the Burgers and Allen–Cahn multi-IC models (1.5 and 2.9 times larger): the error is set by the frozen basis, not by the transfer.

Table 5: Transferred heads against trained heads. "Trained": median relative L^2 error of the jointly trained heads at their own instance. "Leave-one-out": Gauss–Newton at a trained instance started from the nearest *other* head, median error and median ratio to that instance's own head. "Unseen": Gauss–Newton at the unseen targets of Table 1.

		trained heads	leave-one-out		unseen targets
	PDE	rel. L^2	rel. L^2	ratio to oracle	rel. L^2
EP	heat	3.0×10^{-3}	2.9×10^{-3}	0.96	2.8×10^{-3}
	wave	4.3×10^{-2}	4.2×10^{-2}	1.00	4.4×10^{-2}
	Allen–Cahn	4.5×10^{-3}	4.6×10^{-3}	1.00	4.1×10^{-3}
	Burgers	3.7×10^{-2}	3.6×10^{-2}	0.98	3.7×10^{-2}
IC	heat	2.2×10^{-3}	1.2×10^{-3}	0.89	1.6×10^{-3}
	wave	7.1×10^{-3}	5.2×10^{-3}	1.01	7.2×10^{-3}
	Allen–Cahn	7.3×10^{-3}	6.5×10^{-3}	1.00	2.1×10^{-2}
	Burgers	4.8×10^{-2}	3.5×10^{-2}	1.06	7.4×10^{-2}

B.9 EXTRAPOLATION

Table 6. Multi-EP: targets s anchor spacings past both ends of the trained range (0.307 in $\ln \nu$; 0.127 in ν for wave, whose lower end admits only $s = 0.5$), from the extreme trained head, with RK4 sub-steps proportional to s (8 per spacing, at least 4) and Adam for 1500 steps. The three methods agree to two digits and their error grows with s: it exceeds 10% at one spacing for all but the heat equation, and for Burgers the nearest head is already as accurate as the transferred one. Multi-IC: 20 latents per equation outside the training box (2.8 spacings from it in the median, 5 spacings from the nearest head), with the settings of Table 1. Gauss–Newton keeps its interpolation accuracy for heat, wave and Allen–Cahn and degrades to 0.18 for Burgers (basis floor 0.07); the continuation with a fixed 4 sub-steps is under-resolved at these distances (1 to 2 times the Gauss–Newton error in the median, 2 to 5 targets per equation above 10 times), and Adam at 2000 steps is 5 to 17 times worse for wave and Allen–Cahn. In both settings the limitation is the frozen basis, trained on the range only.

Table 6: Extrapolation. Multi-EP: median relative L^2 error at s anchor spacings past the trained range of ν (both ends), for the untouched nearest head, Gauss–Newton, the Hessian continuation and Adam (1500 steps) from that head. Multi-IC: median over 20 unseen ICs outside the training box of the latents, for the nearest head, Gauss–Newton, the continuation (4 sub-steps) and Adam (2000 steps).

Multi-EP

PDE	s	nearest head	Gauss–Newton	Hessian + RK4	Adam
heat	0.5	5.7×10^{-2}	1.5×10^{-2}	1.5×10^{-2}	1.6×10^{-2}
	1	1.2×10^{-1}	3.6×10^{-2}	3.6×10^{-2}	3.7×10^{-2}
	2	2.4×10^{-1}	1.1×10^{-1}	1.1×10^{-1}	1.1×10^{-1}
	4	5.2×10^{-1}	3.5×10^{-1}	3.5×10^{-1}	3.5×10^{-1}
wave	0.5	1.5×10^{-1}	8.1×10^{-2}	8.1×10^{-2}	8.2×10^{-2}
	1	4.3×10^{-1}	3.4×10^{-1}	3.4×10^{-1}	3.4×10^{-1}
	2	8.2×10^{-1}	7.9×10^{-1}	7.9×10^{-1}	7.8×10^{-1}
	4	1.4	1.5	1.5	1.3
Allen–Cahn	0.5	1.2×10^{-1}	8.0×10^{-2}	8.0×10^{-2}	8.0×10^{-2}
	1	2.2×10^{-1}	1.4×10^{-1}	1.4×10^{-1}	1.4×10^{-1}
	2	4.2×10^{-1}	3.0×10^{-1}	3.0×10^{-1}	3.0×10^{-1}
	4	8.2×10^{-1}	6.4×10^{-1}	6.4×10^{-1}	6.4×10^{-1}
Burgers	0.5	1.4×10^{-1}	1.4×10^{-1}	1.4×10^{-1}	1.5×10^{-1}
	1	1.9×10^{-1}	1.8×10^{-1}	1.8×10^{-1}	2.1×10^{-1}
	2	3.1×10^{-1}	3.0×10^{-1}	3.0×10^{-1}	3.6×10^{-1}
	4	5.9×10^{-1}	5.6×10^{-1}	5.6×10^{-1}	6.7×10^{-1}

Multi-IC (targets outside the training box)

PDE	nearest head	Gauss–Newton	Hessian + RK4	Adam
heat	1.9×10^{-1}	1.9×10^{-3}	6.1×10^{-3}	2.0×10^{-3}
wave	9.4×10^{-1}	7.0×10^{-3}	2.0×10^{-2}	1.1×10^{-1}
Allen–Cahn	9.1×10^{-1}	1.8×10^{-2}	1.3×10^{-1}	1.6×10^{-1}
Burgers	7.9×10^{-1}	1.8×10^{-1}	2.6×10^{-1}	2.6×10^{-1}

C MULTI-IC: EMBEDDING SPACES AND REMAINING EQUATIONS

This appendix collects, for the four equations, the quality of the trained embedding space, i.e. the loss history of the MH training (Figure 5) and the solutions carried by the trained heads (Figure 6), and the TL results for the three equations not shown in the main text (Figures 7 and 8). The quantitative comparison for all equations is in Table 1.

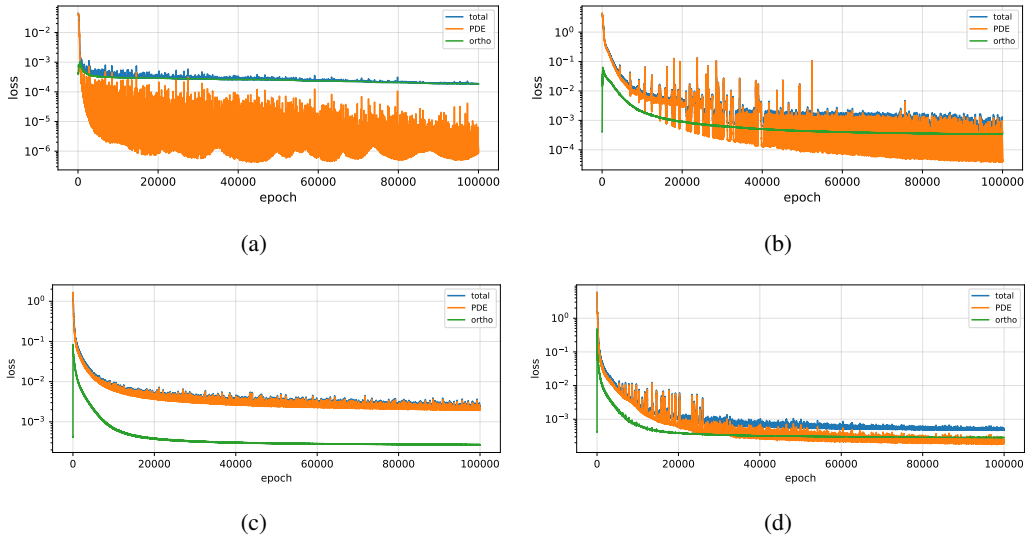


Figure 5: Loss history of the MH training for 10^5 epochs for the heat (a), wave (b), Burgers (c) and Allen–Cahn (d) equations. Total loss in blue, PDE loss (Eq. 7) in orange, orthogonality loss (Eq. 10) in green.

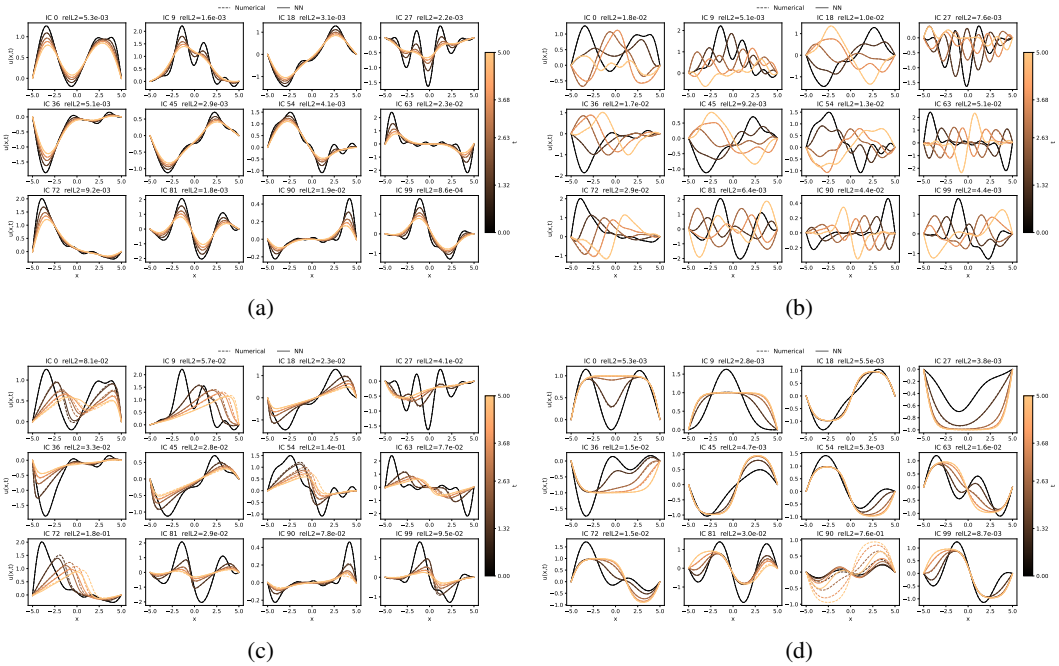
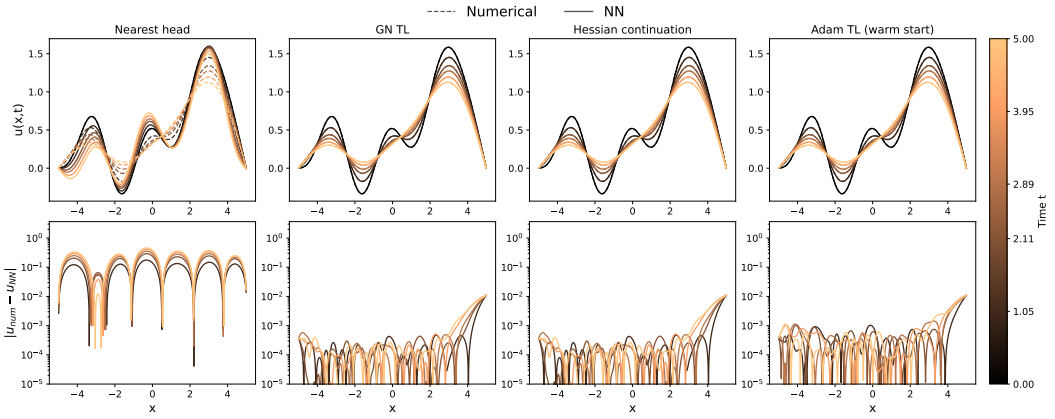
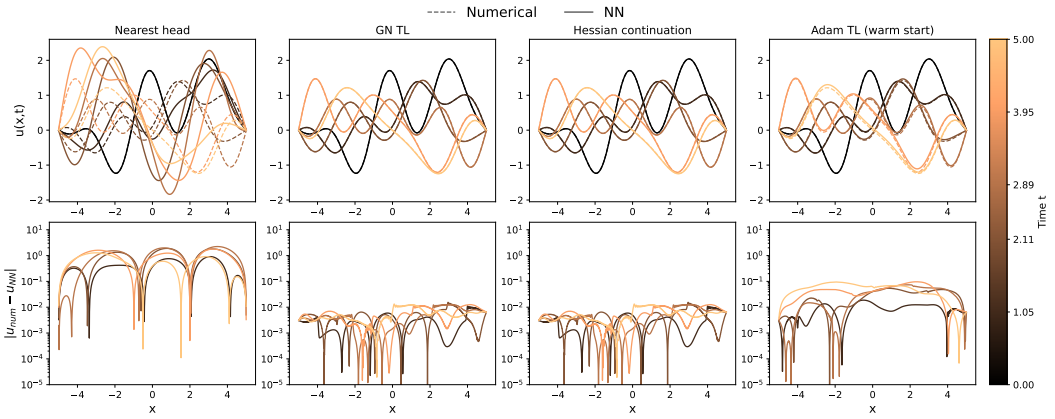


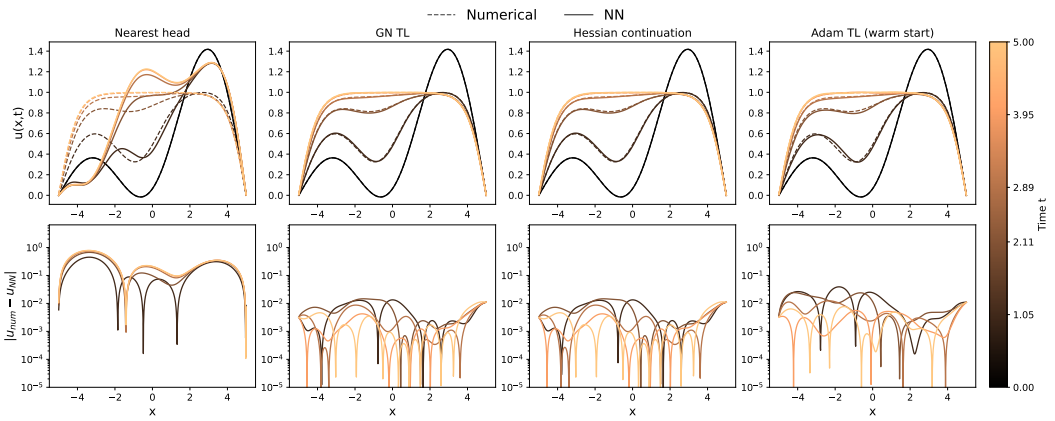
Figure 6: Solutions from a representative subset of the trained heads against the numerical reference, for the heat (a), wave (b), Burgers (c) and Allen–Cahn (d) equations, ordered from highest relative L^2 error.



(a)



(b)



(c)

Figure 7: TL to an unseen IC for the heat (a), wave (b) and Allen–Cahn (c) equations. Layout as in Figure 2.

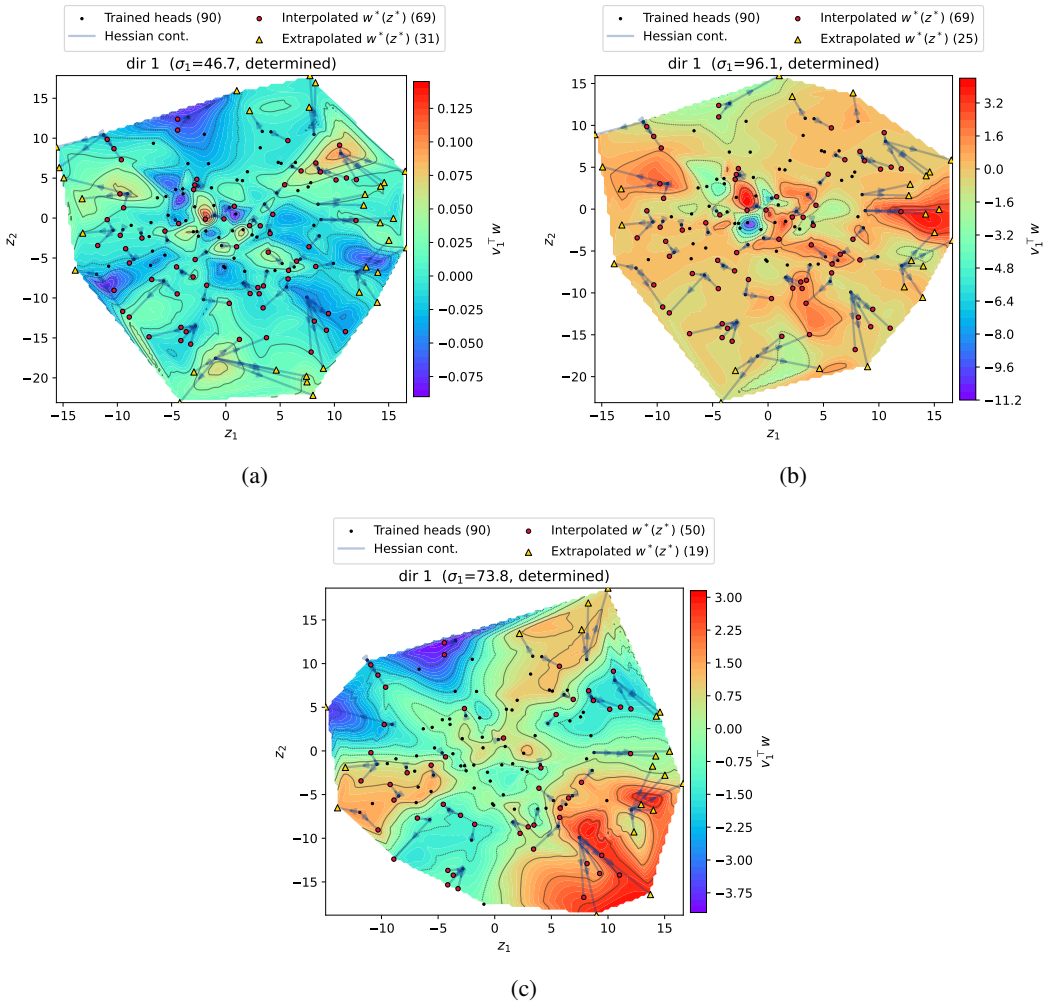


Figure 8: Weight space $\mathbf{w}(\mathbf{z})$ for the heat (a), wave (b) and Allen–Cahn (c) equations, to be compared with the Burgers case of Figure 4. Legend as in Figure 4.

D PRINCIPAL DIRECTIONS IN WEIGHT SPACE

Since the trajectories of Section 4.2.3 live in $\mathbb{R}^B$, displaying them requires a choice of plane. Let $\mathbf{J} = \partial\mathbf{F}/\partial\mathbf{w}$ be assembled at the trained head that starts the path, and let $\mathbf{J} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top$ be its singular value decomposition (Golub & Van Loan, 2013; Hansen, 1998). The right singular vectors $\mathbf{v}_i$ form an orthonormal frame of weight space, ordered by $\sigma_i = \|\mathbf{J}\mathbf{v}_i\|$, the first-order change of the residual under a unit displacement of the head along $\mathbf{v}_i$. We display heads through their coordinates $\mathbf{v}_i^\top \mathbf{w}$, reading the leading directions as those the PDE constrains most strongly and the trailing ones as those it constrains least. This procedure does not discard any of the B basis components, as the frame is a rotation and the ordering only serves to choose what to plot.

The basis is orthonormal (Section 4.1.3), so a displacement of given length changes the field by the same amount in every direction, whereas the residual it produces is weighted by σ_i, and the σ_i span one to two orders of magnitude. The least-squares problem of Eq. (14) therefore pins the leading coordinates down tightly and leaves the trailing ones comparatively free.