# Probing optimisation in physics-informed neural networks

Nayara Fonseca[*,†], Veronica Guidetti[*,‡], Will Trojak[*,†]

[†] IBM Research Europe, Daresbury, WA44AD, United Kingdom
[‡] University of Modena and Reggio Emilia Department of Physics, Informatics and Mathematics Via G. Campi 213/a, 41125, Modena, Italy

## Main contributions

- We present a **novel, low-cost approach to study optimiser behaviour** via the definition of a **local reference frame and the curvature of the training trajectory** in the NN parameter space (see Fig. 1).
- We observed that **PINNs convergence is significantly influenced by the optimisation algorithm**. In particular, **second-order optimisation algorithms** which do not strictly follow gradient directions **exhibit better performance**.
- Studying the linear advection equation, we see a **negative correlation between the convergence error and the local path curvature in the NN parameter space**, suggesting that PINNs' optimal solutions lie in highly curved regions.

## Physics Informed Neural Networks (PINNs)

PINNs [4] aim to provide a universal regressor that can solve any PDE/ODE problem. In summary, for a PDE system as:

$$\begin{cases} G(\mathbf{x}, u^l(\mathbf{x}), \nabla u^l(\mathbf{x}), u^l(\vec{x}), \nabla^2 u^l(\mathbf{x}), u^l(\mathbf{x}), \dots) = 0 & \text{PDE} \\ u^l(t_0, \vec{x}) = u_0^l(\vec{x}) & \text{Initial conditions (IC)} \\ u^l(t, \vec{x})|_{\partial\mathcal{D}} = f^l(\mathbf{x}) & \text{Boundary conditions (BC)} \end{cases}$$

where $G$ is a PDE, $\mathbf{x} \equiv (t, \vec{x}) \in \mathcal{D} \subset \mathbb{R}^D$, $\mathbf{u}(\mathbf{x}) \in \mathbb{R}^{n_o}$, and $l = 1, \dots, n_o$. IC and BC can be generalized to Dirichelet, Von Neumann or Robin cases. We sample collocation points from the bulk ($\mathcal{D}$), the boundary ($\partial\mathcal{D}$), and the IC domain ($\mathcal{D}_0$) and define a highly non-linear transformation, $\hat{\mathbf{u}}$, acting on $x \in \mathcal{D}$:

$$\hat{\mathbf{u}}(\mathbf{x}, \boldsymbol{\omega}) = \mathcal{NN}(\mathbf{x}, \boldsymbol{\omega})$$

where $\mathcal{NN}$ is a NN architecture with parameters $\boldsymbol{\omega}$. Solving the Cauchy system is then rephrased as minimizing the following piece-wise objective function:

$$\begin{aligned} \mathcal{L}(\mathbf{x}, \omega) = & \sum_{k=1,\dots,N_{\mathcal{D}}} \langle \left[ G(\mathbf{x}, \hat{u}^k(\mathbf{x}, \boldsymbol{\omega}), \nabla \hat{u}^k(\mathbf{x}, \omega), \dots) \right]^2 \rangle_{\mathbf{x}=(t,\vec{x})\in\mathcal{D}} + & \leftarrow \quad \mathcal{L}_{\mathcal{D}} \\ & \sum_{l=1,\dots,N_{\mathcal{D}_0}} \langle \left[ \hat{u}^l(t_0, \vec{x}, \boldsymbol{\omega}) - u_0^l(\vec{x}) \right]^2 \rangle_{\mathbf{x}=(t_0,\vec{x})\in\mathcal{D}_0} + & \leftarrow \quad \mathcal{L}_{\mathcal{D}_0} \\ & \sum_{m=1,\dots,N_{\partial\mathcal{D}}} \langle [\hat{u}^m(t, \mathbf{x}, \boldsymbol{\omega}) - f^m(t, \mathbf{x})]^2 \rangle_{\mathbf{x}=(t,\vec{x})\in\partial\mathcal{D}} & \leftarrow \quad \mathcal{L}_{\partial\mathcal{D}}\,, \end{aligned}$$

where $\langle\cdot\rangle_{\mathbf{x}\in X}$ represents the average value computed on the collocation points belonging to $X$. See Fig. 1 for a pictorial view of PINNs optimisation.

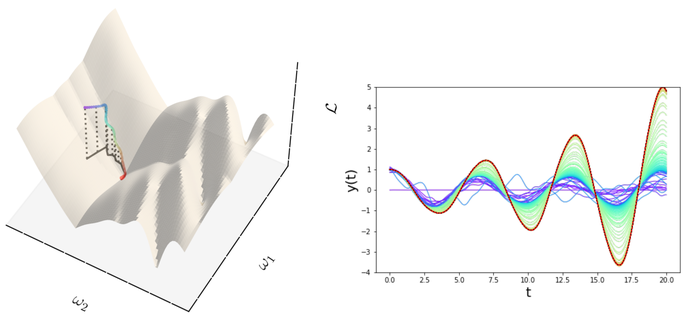


Figure 1. Pictorial view of PINN optimisation. Left: Optimisation trajectory on the loss surface or on a distorted loss as seen by the optimiser (rainbow line). The projection of this curve onto the parameter space (black curve on grey plane). Right: example of PINN output throughout a successfull training trajectory.

## Convergence and optimisers

Recently PINNs were shown to exhibit **poor convergence properties** (e.g. this recent review [1]). Indeed, **PINNs are affected by a spectral bias**, i.e. they struggle to learn high-frequency modes. Several works analyzed and modified parts of the PINNs training pipeline to improve convergence. They studied the effects of different architectures, loss function formulations and weighting, and ways to treat domain and collocation points. However, little attention has been devoted to the role of optimisers.

**This work aimed to improve the understanding of how PINNs performance is affected by the optimiser choice.**

**Case Study.** Specifically, we considered the following algorithms spanning different optimiser categories:

- **GD**: Gradient descent
- **LBFGS**: a second order quasi-Newton method
- **ADAM**: an adaptive stochastic GD algorithm
- **BBI**: a relativistic, energy-conserving optimiser with chaotic jumps

We study the (1+1)-dimensional **Linear Advection Equation** defined as:

$$\begin{cases} \partial_t u + \beta\, \partial_x u = 0, \quad \text{for} \quad \mathrm{u} : \mathrm{T} \times \Omega \mapsto \mathbb{R}, \quad \Omega = [0, 2\pi), \mathrm{T} \in [0, 1], \\ u(x, 0) = \sin(x), \\ u(0, t) = u(2\pi, t), \end{cases}$$

The number of training epochs was 5k, that of collocation point was 2k, and we use a 2- and a 4-hidden layer MLP with 25 and 50 neurons per layer, respectively. Thus, the larger network is in the over-parameterised regime, i.e. it has more parameters than training data. We use $\tanh$ activation function in every layer and let $\beta = \{1, 5, 15, 30\}$. For each architecture and optimiser we perform a learning rate grid search to optimise the training regime.

## Tracking the local curvature

Let us define the weight update rule for the $k$th update as $\Delta\boldsymbol{\omega}_k \equiv \boldsymbol{\omega}_{k+1} - \boldsymbol{\omega}_k = \mathbf{V}(\boldsymbol{\omega}_k, \boldsymbol{\eta})$, where $\boldsymbol{\eta}$ are the model hyperparameters. Assuming the continuum time limit, the trajectory in the parameter space during training can be described by:

$$\dot{\boldsymbol{\omega}} = \mathbf{V}(\boldsymbol{\omega}, \boldsymbol{\eta})\,; \tag{1}$$

see, for example, the projected black curve in Fig. 1. Despite Eq. (1) does not have a dynamic interpretation (except for GD), it can be efficiently used to **understand the kinematics of the optimiser trajectories**. To analyse the motion in the parameter space as described by (1), we introduce the unit vector tangential to the training trajectory, denoted by $\hat{\mathbf{T}}$:

$$\hat{\mathbf{T}} = \frac{\dot{\boldsymbol{\omega}}}{\|\dot{\boldsymbol{\omega}}\|_2}, \quad \text{where} \quad \|\dot{\boldsymbol{\omega}}\|_2 = \sqrt{\langle\dot{\boldsymbol{\omega}}, \dot{\boldsymbol{\omega}}\rangle}. \tag{2}$$

For an $N_\omega$ parameter neural network, $\hat{\mathbf{T}}$ is a time-dependent vector in an $N_\omega$-dimensional space. We can monitor and define the local curvature of the training trajectory, $\kappa_\omega$, as:

$$\kappa_\omega = \left\| \frac{\mathrm{d}\hat{\mathbf{T}}}{\mathrm{d}\omega} \right\|_2 = \frac{1}{\|\dot{\boldsymbol{\omega}}\|_2} \left\| \frac{\mathrm{d}\hat{\mathbf{T}}}{\mathrm{d}t} \right\|_2 = \frac{1}{\|\dot{\boldsymbol{\omega}}\|_2^2} \left[ \ddot{\boldsymbol{\omega}} \cdot \ddot{\boldsymbol{\omega}} - \left( \frac{\mathrm{d}\|\dot{\boldsymbol{\omega}}\|_2}{\mathrm{d}t} \right)^2 \right]^{1/2}. \tag{3}$$

$\kappa_\omega$ represents a local geometric quantity, i.e., the local curvature in the parameter space removing the effects of time and trajectory speed.

## Results

- All configurations failed to produce good approximations for $\beta = 30$ **[Fig. 2a]**;
- Large networks achieve lower error, especially for BBI **[Fig. 2a]**;
- When convergence is achieved, we find a clear negative relation between the final values of MSE and the curvature $\kappa_\omega$. LBFGS reaches the highest $\kappa_\omega$ and the lowest error values **[Fig. 2b]**;
- For each optimiser, we find a clear negative/positive correlation between $\kappa_\omega$ that holds throughout the whole training trajectory. In particular, successful optimisers show negative correlation, i.e., their optimal solutions lie in highly curved regions **[Fig. 2c]**.

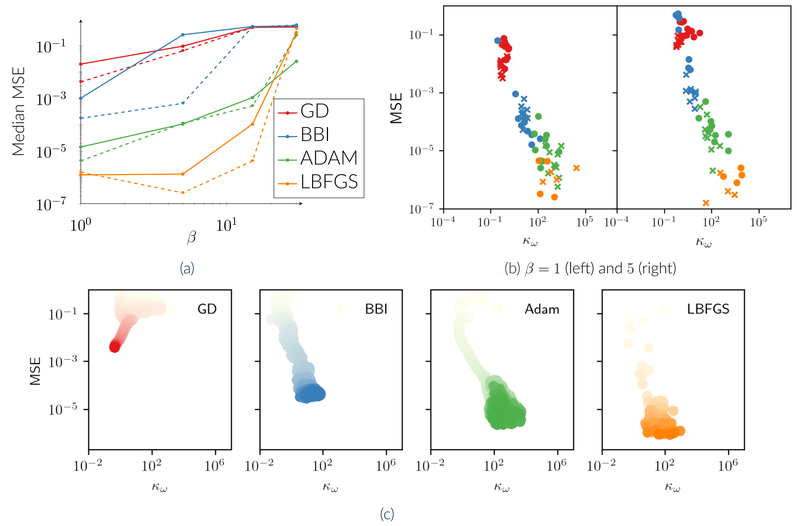


Figure 2. (a) Median MSE over 10 samples for various optimisers. *Solid/dashed* lines refer to small/large MLP. (b) Relation between final values of convergence (MSE) and curvature ($\kappa_\omega$) for $\beta = 1$ (left) and 5 (right). Dots/crosses refer to small/large MLP. (c) Examples of the relation between MSE and $\kappa_\omega$ during training (from pale to intense color) for different optimisers ($\beta = 1$).

## Outlooks

We are currently working on two promising research directions:

- Comparison of our approach with the costly Hessian-based methods relating local curvature and accuracy, [3].
- Exploration of the connection between model generalisation and the flatness of minima for problems requiring high accuracy [2].

## References

[1] Z. Hao, S. Liu, Y. Zhang, C. Ying, Y. Feng, H. Su, and J. Zhu.
Physics-informed machine learning: A survey on problems, methods and applications, arxiv:2211.08064.

[2] W. R. Huang, Z. Emam, M. Goldblum, L. Fowl, J. K. Terry, F. Huang, and T. Goldstein.
Understanding generalization through visualizations, 2020.

[3] E. J. Michaud, Z. Liu, and M. Tegmark.
Precision machine learning, *Entropy*, 25(1):175,2023.

[4] M. Raissi, P. Perdikaris, and G. E. Karniadakis.
Physics informed deep learning (part i): Data-driven solutions of nonlinear partial differential equations, 2017.