Challenges in Training PINNs: A Loss Landscape Perspective

Pratik Rathore[1], Weimu Lei[1], Zachary Frangella[1], Lu Lu[2], Madeleine Udell[1]

July 25, 2024

[1]Stanford University [2]Yale University

Formulation of PINNs

PDE formulation:

$$\mathcal{D}[u(x,t)] = 0, \quad x \in \Omega,$$
$$\mathcal{B}[u(x,t)] = 0, \quad x \in \partial\Omega$$

- $\mathcal{D}$ = differential operator, $\mathcal{B}$ = boundary/initial condition operator, $\Omega \subseteq \mathbb{R}^d$
- Example: convection PDE

$$\underbrace{\frac{\partial u}{\partial t} + \beta \frac{\partial u}{\partial x}}_{\mathcal{D}[u(x,t)]} = 0, \quad x \in (0, 2\pi), t \in (0, 1),$$
$$\underbrace{u(x,0) - \sin(x)}_{\mathcal{B}_1[u(x,t)]} = 0, \quad x \in [0, 2\pi],$$
$$\underbrace{u(0,t) - u(2\pi, t)}_{\mathcal{B}_2[u(x,t)]} = 0, \quad t \in [0, 1]$$

Formulation of PINNs, continued

- PINNs approximate $u(x, t)$ by a neural network $u(x, t; w)$.
- This neural network is trained using a non-linear least squares loss:

$$\underset{w \in \mathbb{R}^p}{\text{minimize}}\ L(w) = \underbrace{\frac{1}{2n_{\text{res}}} \sum_{i=1}^{n_{\text{res}}} \left(\mathcal{D}[u(x_r^i, t_r^i; w)] \right)^2}_{\text{PDE residual loss}} + \underbrace{\frac{1}{2n_{\text{bc}}} \sum_{j=1}^{n_{\text{bc}}} \left(\mathcal{B}[u(x_b^j, t_b^j; w)] \right)^2}_{\text{initial/boundary conditions loss}}$$

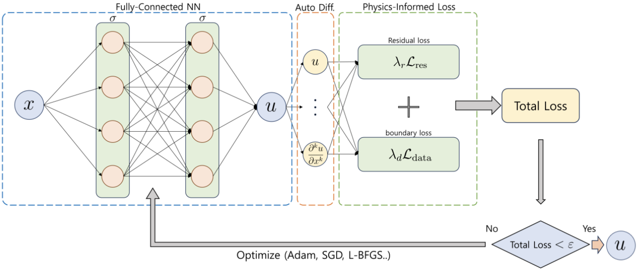

Figure 1: The PINN framework [Ko and Park, 2024].

Training challenges

- **High-precision solution needed**: The PINN loss L needs to be near-zero (often $< 10^{-4}$) to obtain a low ℓ_2-relative error (L2RE).

Training challenges

- **High-precision solution needed**: The PINN loss L needs to be near-zero (often $< 10^{-4}$) to obtain a low ℓ_2-relative error (L2RE).
- **Poor conditioning**: Previous work has suggested that the PINN loss is *ill-conditioned* (i.e., harder to optimize) [Krishnapriyan et al., 2021, De Ryck et al., 2023].

Training challenges

- **High-precision solution needed**: The PINN loss L needs to be near-zero (often $< 10^{-4}$) to obtain a low ℓ_2-relative error (L2RE).
- **Poor conditioning**: Previous work has suggested that the PINN loss is *ill-conditioned* (i.e., harder to optimize) [Krishnapriyan et al., 2021, De Ryck et al., 2023].
- **Non-convexity**: Hard to reach a global minimum! L-BFGS [Nocedal and Wright, 2006] is used for training PINNs [Raissi et al., 2019, Krishnapriyan et al., 2021, Hao et al., 2023], but it can encounter challenges due to saddle points [Dauphin et al., 2014].

- The *condition number*, κ, is determined by the spectrum of the Hessian of the loss, $H_L(w)$.
- Convergence rates of first-order methods are determined by κ.
- When the eigenvalues of $H_L(w)$ are spread out, κ is large (i.e., ill-conditioned), leading to slow convergence of first-order methods.

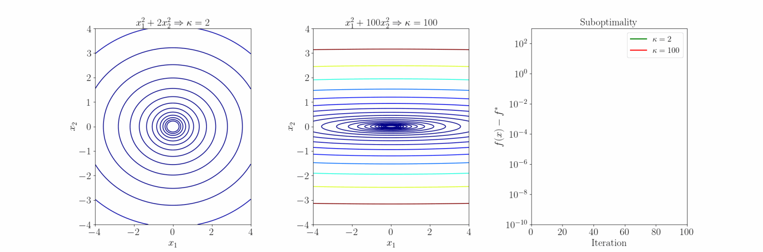

Figure 2: Convergence of gradient descent when $\kappa = 2$ and $\kappa = 100$.

The PINN loss is ill-conditioned

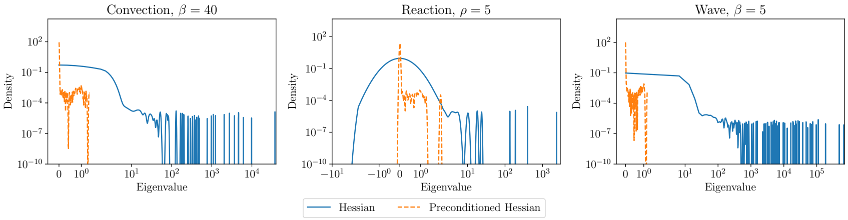

Figure 3: Spectral density of the Hessian and the preconditioned Hessian at the end of optimization.

- We use spectral density estimation [Ghorbani et al., 2019, Yao et al., 2020] to compute $\lambda(H_L(w))$.

The PINN loss is ill-conditioned

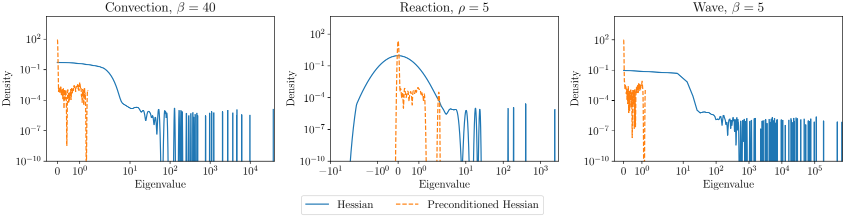

Figure 3: Spectral density of the Hessian and the preconditioned Hessian at the end of optimization.

- We use spectral density estimation [Ghorbani et al., 2019, Yao et al., 2020] to compute $\lambda(H_L(w))$.
- The PINN loss is ill-conditioned: $\lambda_{\max}(H_L(w)) > 10^3$ (Fig. 3)

The PINN loss is ill-conditioned

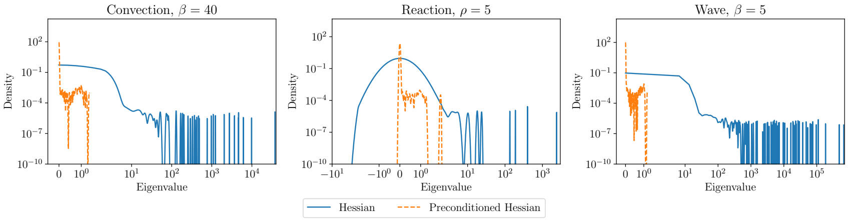

Figure 3: Spectral density of the Hessian and the preconditioned Hessian at the end of optimization.

- We use spectral density estimation [Ghorbani et al., 2019, Yao et al., 2020] to compute $\lambda(H_L(w))$.
- The PINN loss is ill-conditioned: $\lambda_{\max}(H_L(w)) > 10^3$ (Fig. 3)
- L-BFGS reduces $\lambda_{\max}(H_L(w))$ by at least 10^3 (Fig. 3).

Adam+L-BFGS outperforms Adam or L-BFGS alone

- Combining first-order + second-order optimization leads to the best performance (Fig. 4).

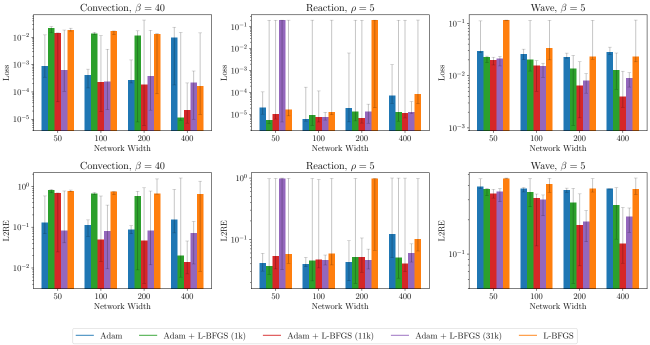

Figure 4: Across most architectures, Adam+L-BFGS outperforms Adam and L-BFGS alone. All methods are run for 41000 iterations each.

Theoretical benefits of first-order + second-order methods

Gradient-Damped Newton Descent (GDND)

1. Run K_{GD} steps of gradient descent, starting at w_0:

$$w_{k+1} = w_k - \eta \nabla L(w_k).$$

2. Run K_{DN} steps of Newton's method, starting at $\tilde{w}_0 = w_{K_{\mathrm{GD}}}$:

$$\tilde{w}_{k+1} = \tilde{w}_k - \eta \left(H_L(\tilde{w}_k) + \gamma I\right)^{-1} \nabla L(\tilde{w}_k).$$

Theoretical benefits of first-order + second-order methods

Gradient-Damped Newton Descent (GDND)

1. Run K_{GD} steps of gradient descent, starting at w_0:

$$w_{k+1} = w_k - \eta \nabla L(w_k).$$

2. Run K_{DN} steps of Newton's method, starting at $\tilde{w}_0 = w_{K_{\mathrm{GD}}}$:

$$\tilde{w}_{k+1} = \tilde{w}_k - \eta \left(H_L(\tilde{w}_k) + \gamma I\right)^{-1} \nabla L(\tilde{w}_k).$$

Theorem (Informal, Rathore et al. [2024]) There exists $K_{\mathsf{GD}} < \infty$ such that Phase 1 of GDND outputs a point $w_{K_{\mathsf{GD}}}$, for which Phase 2 of GDND satisfies

$$L(\tilde{w}_k) \leq \left(\frac{2}{3}\right)^k L(w_{K_{\mathsf{GD}}}).$$

Hence after $K_{\mathsf{DN}} \geq 3 \log\left(\frac{L(w_{K_{\mathsf{GD}}})}{\epsilon}\right)$ iterations, the output of GDND satisfies $L(\tilde{w}_{K_{\mathsf{DN}}}) \leq \epsilon$.

Convergence is fast and independent of the condition number!

Adam+L-BFGS is insufficient for optimizing the PINN loss

- Adam+L-BFGS does not reach a critical point: $\|\nabla L(w_{\mathrm{final}})\| > 10^{-3}$

Adam+L-BFGS is insufficient for optimizing the PINN loss

- Adam+L-BFGS does not reach a critical point: $\|\nabla L(w_{\text{final}})\| > 10^{-3}$
- Strong Wolfe line search in L-BFGS [Nocedal and Wright, 2006]:

$$L(w_k + \eta_k d_k) \leq L(w_k) + c_1 \eta_k \nabla L(w_k)^T d_k \qquad \text{(Sufficient decrease)}$$
$$|\nabla L(w_k + \eta_k d_k)^T d_k| \leq c_2 |\nabla L(w_k)^T d_k| \qquad \text{(Curvature)}$$

  where $0 < c_1 < c_2 < 1$ and d_k is a *descent direction*
- L-BFGS (incorrectly) selects $\eta_k = 0$, leading to early stopping.

Adam+L-BFGS is insufficient for optimizing the PINN loss

- Adam+L-BFGS does not reach a critical point: $\|\nabla L(w_{\text{final}})\| > 10^{-3}$
- Strong Wolfe line search in L-BFGS [Nocedal and Wright, 2006]:

$$L(w_k + \eta_k d_k) \leq L(w_k) + c_1 \eta_k \nabla L(w_k)^T d_k \qquad \text{(Sufficient decrease)}$$
$$|\nabla L(w_k + \eta_k d_k)^T d_k| \leq c_2 |\nabla L(w_k)^T d_k| \qquad \text{(Curvature)}$$

  where $0 < c_1 < c_2 < 1$ and d_k is a *descent direction*
- L-BFGS (incorrectly) selects $\eta_k = 0$, leading to early stopping.
- We develop a new second-order optimizer, NysNewton-CG, which only requires the sufficient decrease condition.

NNCG: a second-order method for training PINNs

- NysNewton-CG (NNCG) is an inexact Newton method that uses low-rank preconditioning [Frangella et al., 2023] to accelerate solving for the (damped) Newton direction

$$d_k = -(H_L(w_k) + \rho I)^{-1} \nabla L(w_k).$$

NNCG: a second-order method for training PINNs

- NysNewton-CG (NNCG) is an inexact Newton method that uses low-rank preconditioning [Frangella et al., 2023] to accelerate solving for the (damped) Newton direction

$$d_k = -(H_L(w_k) + \rho I)^{-1} \nabla L(w_k).$$

- NNCG selects the stepsize using only the sufficient decrease condition (i.e., *Armijo line search*)

$$L(w_k + \eta_k d_k) \leq L(w_k) + c_1 \eta_k \nabla L(w_k)^T d_k. \qquad \text{(Sufficient decrease)}$$

NNCG: a second-order method for training PINNs

- NysNewton-CG (NNCG) is an inexact Newton method that uses low-rank preconditioning [Frangella et al., 2023] to accelerate solving for the (damped) Newton direction

$$d_k = -(H_L(w_k) + \rho I)^{-1} \nabla L(w_k).$$

- NNCG selects the stepsize using only the sufficient decrease condition (i.e., *Armijo line search*)

$$L(w_k + \eta_k d_k) \leq L(w_k) + c_1 \eta_k \nabla L(w_k)^T d_k. \qquad \text{(Sufficient decrease)}$$

- NNCG can reduce both the PINN loss L and L2RE even after Adam+L-BFGS has stalled.

NNCG reduces both loss and error

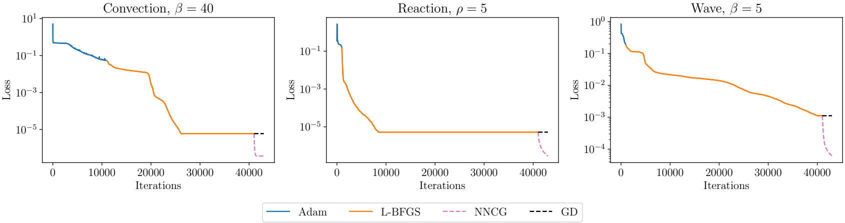

Figure 5: Performance of NNCG and gradient descent (GD) after Adam+L-BFGS. NNCG reduces the loss by a factor greater than 10 in all instances, while GD fails to make progress.

NNCG reduces both loss + error, continued

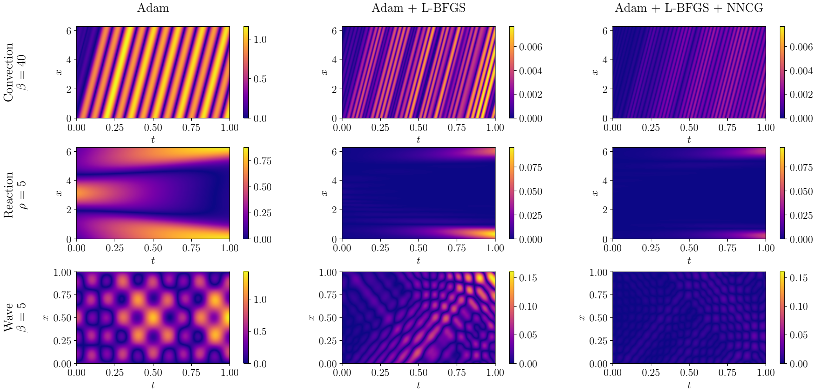

Figure 6: Absolute errors of the PINN solution at optimizer switch points. L-BFGS improves the solution obtained from first running Adam, and NNCG further improves the solution even after Adam+L-BFGS stops making progress.

Conclusion

- PINNs are challenging to train, since they need high-precision solutions and suffer from ill-conditioning and non-convexity.

Conclusion

- PINNs are challenging to train, since they need high-precision solutions and suffer from ill-conditioning and non-convexity.
- The PINN loss is ill-conditioned! Quasi-Newton methods like L-BFGS reduce the condition number.

Conclusion

- PINNs are challenging to train, since they need high-precision solutions and suffer from ill-conditioning and non-convexity.
- The PINN loss is ill-conditioned! Quasi-Newton methods like L-BFGS reduce the condition number.
- Combining first-order + second-order methods is a promising paradigm for training PINNs.

Conclusion

- PINNs are challenging to train, since they need high-precision solutions and suffer from ill-conditioning and non-convexity.
- The PINN loss is ill-conditioned! Quasi-Newton methods like L-BFGS reduce the condition number.
- Combining first-order + second-order methods is a promising paradigm for training PINNs.
- We develop NNCG, a second-order optimizer that improves PINN performance.

Conclusion

- PINNs are challenging to train, since they need high-precision solutions and suffer from ill-conditioning and non-convexity.
- The PINN loss is ill-conditioned! Quasi-Newton methods like L-BFGS reduce the condition number.
- Combining first-order + second-order methods is a promising paradigm for training PINNs.
- We develop NNCG, a second-order optimizer that improves PINN performance.
- Our insights could be used to improve the utility of PINNs for solving difficult PDEs.

Thanks for listening!

Poster: Hall C #301

Time: 11:30 AM – 1:00 PM CEST

References

Y. N. Dauphin, R. Pascanu, C. Gulcehre, K. Cho, S. Ganguli, and Y. Bengio. Identifying and attacking the saddle point problem in high-dimensional non-convex optimization. In *Advances in Neural Information Processing Systems*, 2014.

T. De Ryck, F. Bonnet, S. Mishra, and E. de Bézenac. An operator preconditioning perspective on training in physics-informed machine learning. *arXiv preprint arXiv:2310.05801*, 2023.

Z. Frangella, J. A. Tropp, and M. Udell. Randomized Nyström Preconditioning. *SIAM Journal on Matrix Analysis and Applications*, 44(2):718–752, 2023.

B. Ghorbani, S. Krishnan, and Y. Xiao. An Investigation into Neural Net Optimization via Hessian Eigenvalue Density. In *Proceedings of the 36th International Conference on Machine Learning*, 2019.

References

Z. Hao, J. Yao, C. Su, H. Su, Z. Wang, F. Lu, Z. Xia, Y. Zhang, S. Liu, L. Lu, and J. Zhu. PINNacle: A Comprehensive Benchmark of Physics-Informed Neural Networks for Solving PDEs. *arXiv preprint arXiv:2306.08827*, 2023.

S. Ko and S. H. Park. Vs-pinn: A fast and efficient training of physics-informed neural networks using variable-scaling methods for solving pdes with stiff behavior, 2024. URL `https://arxiv.org/abs/2406.06287`.

A. Krishnapriyan, A. Gholami, S. Zhe, R. Kirby, and M. W. Mahoney. Characterizing possible failure modes in physics-informed neural networks. In *Advances in Neural Information Processing Systems*, 2021.

J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer, 2nd edition, 2006.

References

M. Raissi, P. Perdikaris, and G. 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.

P. Rathore, W. Lei, Z. Frangella, L. Lu, and M. Udell. Challenges in training PINNs: A loss landscape perspective. In *Forty-first International Conference on Machine Learning*, 2024.

Z. Yao, A. Gholami, K. Keutzer, and M. W. Mahoney. PyHessian: Neural Networks Through the Lens of the Hessian. In *2020 IEEE International Conference on Big Data (Big Data)*, 2020.