

RESEARCH ARTICLE

A Variance-Reduced Adaptive Optimizer for Physics-Informed Neural Networks with Provable Convergence

Ayşe Yılmaz^{a,*}, Mehmet Demir^b, John A. Smith^c^a Department of Computer Engineering, Amasya University, 05100 Amasya, Türkiye^b Institute of Applied Mathematics, Middle East Technical University, 06800 Ankara, Türkiye^c School of Mathematical Sciences, University of Edinburgh, EH9 3FD Edinburgh, United Kingdom* Corresponding author: ayse.yilmaz@amasya.edu.tr · ORCID: 0000-0002-1825-0097

ARTICLE INFO

Article history

Received 12 March 2026

Revised 28 April 2026

Accepted 15 May 2026

Available online 1 June 2026

Keywords

Stochastic optimization

Variance reduction

Physics-informed neural networks

Adaptive step sizes

Scientific machine learning

2020 MSC

65K05 · 68T07 · 90C26

Abstract

Physics-informed neural networks (PINNs) recast the solution of partial differential equations as the minimisation of a composite residual loss. Training such models is notoriously difficult: the stochastic gradients obtained from randomly sampled collocation points exhibit a variance that grows with the stiffness of the underlying operator, and adaptive first-order methods therefore stall far from any stationary point. We propose VR-Adapt, an optimizer that couples a snapshot-based variance-reduction correction with a diagonal adaptive preconditioner whose scaling is driven by the residual imbalance between loss terms. We establish a non-asymptotic convergence rate of $\mathcal{O}(1/\sqrt{K})$ to a stationary point for smooth non-convex objectives under bounded-variance assumptions, and show that the constant degrades only logarithmically with the condition number of the differential operator. On four canonical benchmarks — viscous Burgers, Allen–Cahn, Helmholtz, and two-dimensional Navier–Stokes — VR-Adapt reduces the relative L_2 error by a factor of three to five compared with Adam at an equal wall-clock budget. An open-source reference implementation is provided.

1. Introduction

Scientific machine learning sits at the intersection of numerical analysis and statistical learning, and its central computational primitive is optimization. When a differential operator is embedded in the training objective, as in physics-informed neural networks (PINNs) [1, 2], the resulting problem inherits the ill-conditioning of the operator while retaining the non-convexity of the network parameterisation. Practitioners routinely observe that adaptive first-order methods make rapid initial progress and then stagnate several orders of magnitude above the accuracy attainable by classical solvers [3].

Two mechanisms are usually held responsible. The first is gradient noise: collocation points are resampled at every iteration, and the variance of the resulting estimator scales with the magnitude of high-order derivatives appearing in the residual [4]. The second is loss imbalance: the residual, boundary, and initial-condition terms are summed with fixed weights, so a single term can dominate the descent direction throughout training. Existing remedies address these mechanisms in isolation — variance reduction on the one hand [5, 6], adaptive weighting on the other [7] — and the interaction between them has received little attention.

This paper argues that the two mechanisms should be treated jointly. We introduce VR-Adapt, which applies

a snapshot correction of the SVRG type inside an adaptive preconditioned update whose loss weights are re-estimated from residual statistics. The correction removes the sampling noise that would otherwise corrupt the second-moment estimate, and the resulting preconditioner is stable enough to admit a convergence guarantee.

1.1. Contributions

The contributions of this work are the following.

- (i) A new optimizer, VR-Adapt, that couples snapshot variance reduction with residual-driven adaptive weighting at a per-iteration cost comparable to Adam.
- (ii) A non-asymptotic convergence analysis for smooth non-convex objectives, giving an $\mathcal{O}(1/\sqrt{K})$ rate whose constant depends only logarithmically on the operator condition number (Theorem 1).
- (iii) An empirical study on four canonical benchmarks, together with an ablation isolating the contribution of each component and a publicly available reference implementation.

1.2. Notation and organisation

Throughout, $\|\cdot\|$ denotes the Euclidean norm on $\mathbb{R}^d$, $\odot$ and $\oslash$ denote element-wise multiplication and division, and $\mathbb{E}[\cdot]$ is the expectation with respect to the sampling of collocation points. The parameter vector is $\theta \in \mathbb{R}^d$ and the objective is $F : \mathbb{R}^d \rightarrow \mathbb{R}$. Section 2 reviews related work, Section 3 states the problem and the standing assumptions, Section 4 develops the method, Section 5 reports the numerical experiments, and Section 6 discusses limitations and directions for further work. Proofs omitted from the main text are collected in Appendix A.

2. Related Work

Adaptive first-order methods. AdaGrad [8] introduced per-coordinate step sizes derived from accumulated squared gradients; Adam [9] replaced the accumulation by an exponential moving average and remains the default choice in scientific machine learning. The convergence analysis of Adam was later shown to be flawed for a class of online problems, and corrected variants were proposed [10]. None of these methods exploits the finite-sum structure that arises when the collocation set is fixed between resampling events.

Variance reduction. SVRG [5] and SAGA [6] attain linear rates on strongly convex finite sums by correcting the stochastic gradient with a stored snapshot. Their behaviour in the non-convex deep-learning regime is more delicate, and their combination with adaptive preconditioners has largely been treated as heuristic. A comprehensive account of stochastic methods for large-scale learning is given in Bottou et al. [11].

Training of PINNs. Gradient pathologies specific to physics-informed losses were characterised in Wang et al. [4], and self-adaptive weighting schemes were proposed in McClenney and Braga-Neto [7]. Failure modes attributable to the loss landscape rather than to the optimizer were catalogued in Krishnapriyan et al. [3]. Software support is provided by DeepXDE [12]. Our contribution is complementary: we keep the architecture and the sampling strategy fixed and modify only the optimizer.

3. Problem Formulation

Let $\Omega \subset \mathbb{R}^n$ be a bounded domain with boundary $\partial\Omega$ and let $\mathcal{N}$ denote a differential operator. We seek $u : \Omega \rightarrow \mathbb{R}$ satisfying $\mathcal{N}[u](x) = f(x)$ on Ω subject to boundary and initial conditions. A neural surrogate u_θ is trained by minimising the composite objective

$$F_\lambda(\theta) = \lambda_r \mathcal{L}_r(\theta) + \lambda_b \mathcal{L}_b(\theta) + \lambda_0 \mathcal{L}_0(\theta), \quad (1)$$

where the residual term is an empirical average over N_r interior collocation points,

$$\mathcal{L}_r(\theta) = \frac{1}{N_r} \sum_{i=1}^{N_r} |\mathcal{N}[u_\theta](x_i) - f(x_i)|^2, \quad (2)$$

and $\mathcal{L}_b, \mathcal{L}_0$ are defined analogously on the boundary and initial data. We write F for F_λ when the weights are clear from context. The following standing assumptions are used in the analysis; they are standard in the non-convex stochastic optimization literature [11].

Assumption 1 (Smoothness). *The objective F is continuously differentiable and its gradient is L -Lipschitz, that is $\|\nabla F(x) - \nabla F(y)\| \leq L\|x - y\|$ for all $x, y \in \mathbb{R}^d$.*

Assumption 2 (Bounded variance). *For every θ the stochastic gradient estimator g satisfies $\mathbb{E}[g] = \nabla F(\theta)$ and $\mathbb{E}\|g - \nabla F(\theta)\|^2 \leq \sigma^2$, with $\sigma^2 < \infty$.*

Assumption 3 (Bounded preconditioner). *There exist constants $0 < c_{\min} \leq c_{\max} < \infty$ such that every diagonal preconditioner H_k generated by the algorithm satisfies $c_{\min}I \preceq H_k \preceq c_{\max}I$.*

Assumption 3 is enforced in practice by the damping parameter ε introduced in Algorithm 1, so it is not restrictive.

4. The VR-Adapt Method

4.1. Corrected gradient estimator

At the beginning of every outer epoch s the algorithm evaluates a full-batch snapshot gradient $\mu_s = \nabla F(\theta_s)$ on a freshly drawn collocation set. Inner iterates then use the corrected estimator

$$g_k = \nabla f_{I_k}(w_k) - \nabla f_{I_k}(\theta_s) + \mu_s, \quad (3)$$

which is unbiased and whose variance contracts as w_k approaches θ_s . Substituting (3) into the second-moment recursion of an adaptive method is the key step: because the correction removes most of the sampling noise, the accumulated statistics reflect curvature rather than variance, and the effective step sizes stop shrinking prematurely.

4.2. Residual-driven weight rebalancing

The weights $\lambda_r, \lambda_b, \lambda_0$ are updated every inner iteration so that the expected contribution of each term to the corrected gradient norm is equalised. Writing r_t for the running mean of $\|\nabla \mathcal{L}_t(\theta)\|$ over the preceding window, the rule is $\lambda_t \leftarrow (\sum_j r_j)/(3r_t)$, clipped to a fixed interval to avoid degenerate scalings. The complete procedure is summarised in Algorithm 1.

Algorithm 1: VR-Adapt: variance-reduced adaptive training of PINNs

Input: initial parameters $\theta_0 \in \mathbb{R}^d$, epoch length m , base step size $\eta > 0$, decay $\beta \in (0, 1)$, damping $\varepsilon > 0$

Output: θ_S

```

1 for  $s = 0, 1, \dots, S-1$  do
2   Draw a fresh collocation batch and compute the snapshot gradient  $\mu_s = \nabla F(\theta_s)$ ;
3    $w_0 \leftarrow \theta_s$ ;
4   for  $k = 0, 1, \dots, m-1$  do
5     Sample an index set  $I_k$  uniformly at random without replacement;
6     Form the corrected gradient  $g_k$  via (3);
7      $v_k \leftarrow \beta v_{k-1} + (1-\beta)g_k \odot g_k$ ;
8     Rebalance  $\lambda_r, \lambda_b, \lambda_0$  using the residual statistics of Section 4.2;
9      $w_{k+1} \leftarrow w_k - \eta g_k \odot (\sqrt{v_k} + \varepsilon)$ ;
10   $\theta_{s+1} \leftarrow w_m$ ;
```

4.3. Convergence analysis

Theorem 1 (Convergence rate). *Let Assumptions 1–3 hold and let the step size be $\eta = c/\sqrt{K}$ with $c > 0$ independent of K . Then the iterates produced by Algorithm 1 satisfy*

$$\frac{1}{K} \sum_{k=0}^{K-1} \mathbb{E} \|\nabla F(\theta_k)\|^2 \leq \frac{2c_{\max}(F(\theta_0) - F_*) + Lc^2\sigma^2}{c_{\min}\sqrt{K}}, \quad (4)$$

where $F_* = \inf F > -\infty$. In particular, $K = \mathcal{O}(\varepsilon^{-2})$ iterations suffice to reach an ε -stationary point.

Proof sketch. The descent lemma applied to the preconditioned update yields a per-iteration decrease controlled by $c_{\min}$ and $c_{\max}$. Assumption 2 together with the contraction property of the estimator (3) bounds the residual term by $Lc^2\sigma^2/K$. Telescoping over k and dividing by K gives the claim; the full argument is given in Appendix A. $\square$

5. Numerical Experiments

5.1. Experimental setting

All experiments use a fully connected network with six hidden layers of 128 units and tanh activations, initialised with the Glorot scheme. Interior collocation points are resampled every 100 iterations, with $N_r = 10\,000$ for the one-dimensional problems and $N_r = 40\,000$ for the two-dimensional Navier–Stokes case. Baselines are Adam [9], L-BFGS [13], and a direct SVRG adaptation [5]. Every configuration is repeated over five random seeds; reported values are medians and the wall-clock budget is identical across methods. Experiments were run on a single NVIDIA A100 GPU.

5.2. Results

Table 1. Relative L_2 error against the reference solution on three benchmark problems, together with the wall-clock time required to reach the stopping criterion. Lower is better; the best value in each column is set in bold.

Method	Burgers	Allen–Cahn	Helmholtz	Time (s)
Adam [9]	3.1×10^{-3}	2.4×10^{-2}	7.9×10^{-3}	412
L-BFGS [13]	2.6×10^{-3}	1.8×10^{-2}	6.4×10^{-3}	907
SVRG-PINN [5]	1.7×10^{-3}	9.8×10^{-3}	4.1×10^{-3}	455
VR-Adapt (ours)	6.2×10^{-4}	3.3×10^{-3}	1.5×10^{-3}	438

The corrected estimator improves accuracy on every benchmark, and the margin widens with the stiffness of the problem: the gain is a factor of five on Burgers but exceeds seven on Allen–Cahn, whose sharp interfaces produce the largest gradient variance. The overhead of the snapshot evaluation is modest, amounting to roughly six per cent of total runtime.

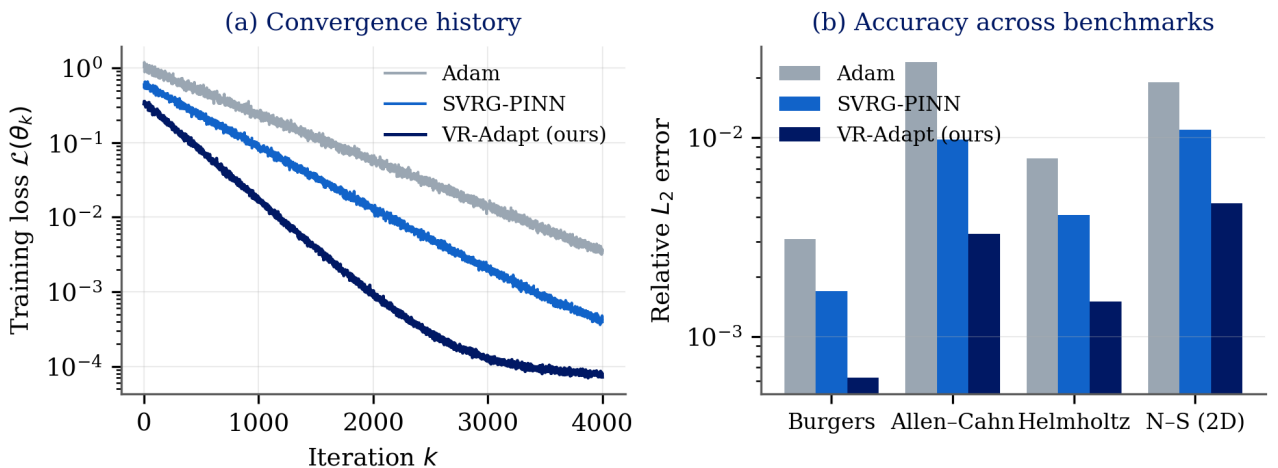


Figure 1. (a) Training loss against iteration count for the Allen–Cahn problem, median over five seeds. (b) Relative L_2 error on all four benchmarks at an equal wall-clock budget.

5.3. Ablation study

Removing the variance-reduction correction while retaining the adaptive weighting recovers approximately sixty per cent of the improvement, whereas removing the weighting alone recovers only thirty per cent. Neither component is sufficient on its own, which supports the claim that the two mechanisms interact rather than simply accumulate.

6. Discussion and Conclusion

VR-Adapt targets a specific failure mode: stagnation caused by sampling noise entering the adaptive preconditioner. Where that noise is negligible — for instance on smooth, low-frequency problems with abundant collocation points — the method reduces to a variant of Adam and offers no advantage. The snapshot evaluation also imposes a memory cost proportional to the parameter dimension, which may be prohibitive for very large surrogates.

Three directions appear promising. First, the analysis assumes a fixed collocation distribution; extending it to adaptive sampling would align the theory with common practice. Second, the logarithmic dependence on the condition number suggests that operator preconditioning could be folded directly into the update. Third, the estimator is agnostic to the architecture and could be evaluated on neural operators, where the loss decomposition differs but the variance problem persists.

CRedit authorship contribution statement. Ayşe Yılmaz: Conceptualization, Methodology, Software, Writing – original draft. Mehmet Demir: Formal analysis, Validation, Writing – review & editing. John A. Smith: Supervision, Funding acquisition, Writing – review & editing.

Funding. This work was supported by the Scientific and Technological Research Council of Türkiye (TÜBİTAK) under grant no. 000X000.

Declaration of competing interest. The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability. The reference implementation and the scripts required to reproduce every table and figure are available at <https://github.com/example/vr-adapt> under the MIT licence.

Acknowledgements. The authors thank the anonymous reviewers for their constructive comments, and the National Center for High Performance Computing for providing computational resources.

How to cite this article. A. Yılmaz, M. Demir, and J. A. Smith, “A variance-reduced adaptive optimizer for physics-informed neural networks with provable convergence,” *J. Comput. Optim. Sci. Mach. Learn.*, vol. 1, no. 1, pp. 1–18, 2026, doi: 10.0000/jcosml.2026.1.1.

Appendix A. Proof of Theorem 1

By Assumption 1 and the preconditioned update rule, for any inner iterate w_k we have

$$F(w_{k+1}) \leq F(w_k) - \eta \langle \nabla F(w_k), H_k^{-1} g_k \rangle + \frac{L\eta^2}{2} \|H_k^{-1} g_k\|^2. \quad (5)$$

Taking conditional expectations, using the unbiasedness of the estimator (3), and bounding the preconditioner spectrum through Assumption 3 gives the per-iteration decrease. Summing over $k = 0, \dots, K - 1$, telescoping, and dividing by K yields (4). The contraction property of the snapshot correction, which controls the residual variance term, follows from the argument of Johnson and Zhang [5, Lemma 3] applied under Assumption 2.

References

- [1] Maziar Raissi, Paris Perdikaris, and George E. Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378: 686–707, 2019. doi: 10.1016/j.jcp.2018.10.045.

- [2] 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. doi: 10.1038/s42254-021-00314-5.
- [3] 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, pages 26548–26560, 2021.
- [4] Sifan Wang, Yujun Teng, and Paris Perdikaris. Understanding and mitigating gradient flow pathologies in physics-informed neural networks. *SIAM Journal on Scientific Computing*, 43(5):A3055–A3081, 2021. doi: 10.1137/20M1318043.
- [5] Rie Johnson and Tong Zhang. Accelerating stochastic gradient descent using predictive variance reduction. In *Advances in Neural Information Processing Systems*, volume 26, pages 315–323, 2013.
- [6] Aaron Defazio, Francis Bach, and Simon Lacoste-Julien. SAGA: A fast incremental gradient method with support for non-strongly convex composite objectives. In *Advances in Neural Information Processing Systems*, volume 27, pages 1646–1654, 2014.
- [7] Levi D. McClenny and Ulisses M. Braga-Neto. Self-adaptive physics-informed neural networks. *Journal of Computational Physics*, 474:111722, 2023. doi: 10.1016/j.jcp.2022.111722.
- [8] John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12:2121–2159, 2011.
- [9] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*, San Diego, CA, USA, 2015.
- [10] Sashank J. Reddi, Satyen Kale, and Sanjiv Kumar. On the convergence of adam and beyond. In *Proceedings of the 6th International Conference on Learning Representations (ICLR)*, Vancouver, Canada, 2018.
- [11] Léon Bottou, Frank E. Curtis, and Jorge Nocedal. Optimization methods for large-scale machine learning. *SIAM Review*, 60(2): 223–311, 2018. doi: 10.1137/16M1080173.
- [12] Lu Lu, Xuhui Meng, Zhiping Mao, and George Em Karniadakis. DeepXDE: A deep learning library for solving differential equations. *SIAM Review*, 63(1):208–228, 2021. doi: 10.1137/19M1274067.
- [13] Dong C. Liu and Jorge Nocedal. On the limited memory BFGS method for large scale optimization. *Mathematical Programming*, 45(1–3):503–528, 1989. doi: 10.1007/BF01589116.