A DATA-PARALLEL ADDITIVELY PRECONDITIONED TRUST-REGION STRATEGY FOR PHYSICS-INFORMED NEURAL NETWORKS

Bindi Çapriqi[1] **Shega Likaj**[1,2] **Ken Trotti**[1] **Rolf Krause**[1]
[1]King Abdullah University of Science and Technology (KAUST)
[2]Università della Svizzera italiana (USI)
{bindi.capriqi,shega.likaj,ken.trotti,rolf.krause}@kaust.edu.sa

ABSTRACT

Physics-informed neural networks (PINNs) are trained by minimizing a composite objective that enforces boundary conditions, initial conditions (for time-dependent problems), and interior PDE residuals. We adapt the Additively Preconditioned Trust-Region Strategy (APTS) to PINN training by introducing a parallel additive preconditioning stage across replicated models. We form data subdomains by randomly partitioning the interior collocation set. Each iteration combines this parallel preconditioning step with a global trust-region correction, governed by trust-region acceptance and adaptive radius updates. We evaluate the approach on representative PDE benchmarks (Heat, Burgers, Laplace) and compare against Adam and L-BFGS using solution-error and PDE-residual metrics.

1 INTRODUCTION

Physics-informed neural networks (PINNs) approximate solutions of partial differential equations (PDEs) by parameterizing the solution with a neural network u_θ and fitting parameters θ to satisfy the PDE and associated initial/boundary conditions. Training is generally nonconvex and can be challenging in practice due to loss-term imbalance and resampling-induced gradient noise. As a result, first-order optimizers such as Adam often require problem-dependent learning-rate and weighting choices, while quasi-Newton methods such as L-BFGS can be sensitive to ill-conditioning (Urbán et al., 2025; Rathore et al., 2024). These challenges motivate globalization strategies with explicit acceptance decisions, such as trust-region mechanisms, to improve robustness and reduce sensitivity to hyperparameters

We propose a data-parallel additively preconditioned trust-region strategy (APTS) scheme (Trotti et al., 2025) for PINN training in which "subdomains" are defined by random partitions of the interior collocation set $\mathcal{C}$ (batches), while the initial-condition and boundary-condition, denoted by $\mathcal{I}$ and $\mathcal{B}$, are shared across all replicas. This differs from geometric domain decomposition used in cPINNs, XPINNs and related domain-decomposed or finite-basis PINN variants(Jagtap & Karniadakis, 2020; Jagtap et al., 2020; Moseley et al., 2023). At each outer iteration we run a two-stage trust-region scheme: (i) a preconditioning stage based on local solves on collocation partitions, and (ii) a global refinement stage on the full objective. Each stage applies the trust-region acceptance test and radius update. See Appendix A.3 for an overview of TR.

We evaluate the method on representative PDE benchmarks (Heat, Burgers, Laplace) and compare against Adam and L-BFGS in terms of solution error and PDE residuals.Our method follows the nonlinear preconditioning viewpoint: local subproblem solves are combined additively to form a global preconditioned direction, in the spirit of additive Schwarz nonlinear preconditioning (ASPIN) (Cai & Keyes, 2002) and the additively preconditioned trust-region framework (Groß, 2009).

2 METHOD

Consider the one-dimensional spatial domain $\Omega = [0, 1]$ and the generic time-dependent PDE

$$F\big(t,\ x,\ u(t,x),\ \nabla u(t,x),\ \nabla^2 u(t,x),\ \ldots,\ \nabla^k u(t,x)\big) = 0, \qquad (t,x) \in (0,T) \times \Omega. \quad (1)$$

Let $u_\theta(t, x)$ denote a PINN surrogate with parameters θ. Training proceeds by minimizing 2

$$L(\theta) = L_\Omega(\theta) + L_{\mathrm{IC}}(\theta) + L_{\mathrm{BC}}(\theta), \quad (2)$$

where L_Ω enforces the PDE residual at interior collocation points, and L_{IC} and L_{BC} enforce the initial and boundary conditions, respectively (Raissi et al., 2019). The dataset is created by sampling interior, initial, and boundary points

$$\begin{aligned}
\mathcal{C} &= \{(x_j, t_j) : x_j \in \Omega,\ t_j \in (0, T],\ j = 1, \ldots, N_{\mathrm{coll}}\},\\
\mathcal{I} &= \{(x_j, 0) : x_j \in \Omega \setminus \partial\Omega,\ j = 1, \ldots, N_{\mathrm{IC}}\},\\
\mathcal{B} &= \{(x_j, t_j) : x_j \in \partial\Omega,\ t_j \in [0, T],\ j = 1, \ldots, N_{\mathrm{BC}}\},
\end{aligned}$$

and, in order to prevent overfitting, these sets are resampled every r global iterations.

At global iteration k, APTS proceeds in two stages: a preconditioning stage followed by a global TR stage. In the preconditioning stage, we form D disjoint "subdomains" by partitioning the interior collocation set $\mathcal{C}_k = \bigsqcup_{d=1}^{D} \mathcal{C}_k^d$. We then instantiate D model replicas (copies) $u_{\theta_k^d},\ d = 1, \ldots, D$, each initialized from the same global parameters $\theta_k^d := \theta_k$. Replica d is associated with the local objective

$$L_k^d(\theta) \ = \ L_\Omega(\theta; \mathcal{C}_k^d) \ + \ L_{\mathrm{IC}}(\theta; \mathcal{I}_k) \ + \ L_{\mathrm{BC}}(\theta; \mathcal{B}_k), \quad (3)$$

where only the interior residual term depends on the partition $\mathcal{C}_k^d$ while the initial-condition and boundary-condition sets $\mathcal{I}$ and $\mathcal{B}$ are shared across all subdomain solves. J local trust-region steps are performed in parallel to train each replica obtaining inner updates $s_{k,j}^d$ and the cumulative local correction $s_k^d := \sum_{j=1}^{J} s_{k,j}^d$. To enforce $\|s_k^d\| \le \Delta_k$, we constrain every inner step to satisfy $\|s_{k,j}^d\| \le \Delta_k/J$ by fixing the local trust-region radius to Δ_k/J, i.e., using Δ_k/J as both the initial and maximum local radius. Then, by the triangle inequality, $\|s_k^d\| \le \sum_{j=1}^{J} \|s_{k,j}^d\| \le \Delta_k$.

The data-parallel preconditioning direction is then the average of the replica corrections, $\bar{s}_k := \frac{1}{D} \sum_{d=1}^{D} s_k^d$. We treat $\bar{s}_k$ as a trial step and set $\theta_{k+1/2} := \theta_k + \bar{s}_k$. If $L(\theta_{k+1/2}) < L(\theta_k)$ we accept the step; otherwise we reject it and set $\theta_{k+1/2} = \theta_k$. This is a simplified trust-region acceptance test, and the preconditioning stage concludes by updating the radius to $\Delta_{k+1/2}$ using trust-region heuristics (Conn et al., 2000). Afterwards, we compute a global trust-region step s_k for the full objective $L(\theta)$ at $\theta_{k+1/2}$ with radius $\Delta_{k+1/2}$. The global stage applies the same accept/reject principle, shrinking the radius and retrying until a step is accepted[1].For more information complete pseudocode description is provided in Appendix A.4, Algorithm 1.

To compute both the local updates s_k^d and the global step s_k, we use a quadratic trust-region model of the objective. Let g and H denote the gradient and Hessian of the objective used in the current stage (local L_k^d or global L_k) at the current iterate. In practice, we do not form H explicitly; instead, the solver accesses curvature through Hessian-vector products computed by auto-differentiation. We formulate the trust-region problem

$$\min_{\|s\| \le \Delta} \ m(s), \qquad m(s) := g^\top s + \tfrac{1}{2} s^\top \big(H + \lambda I\big) s, \quad (4)$$

where $\lambda \ge 0$ is a user-chosen damping parameter introduced for numerical stabilization and improved conditioning. We compute an approximate solution using Steihaug's truncated conjugate-gradient method (TR-CG) (Steihaug, 1983).

3 EXPERIMENTS AND RESULTS

We evaluate APTS on three PDE benchmarks (Heat, Burgers, Laplace), compare it against full-batch Adam and L-BFGS, and study the effect of the number of data subdomains $D \in \{2, 4, 8\}$.

[1] In practice, this repeats on average only 1.01 times per iteration.

All methods use the same network architecture and initialization: an MLP with $\tanh$ activations and layer sizes $[2, 20, 20, 20, 1]$. Each run is trained for 2000 outer iterations and repeated over 3 random seeds (plots report the mean with variability). Interior points are resampled every 10 iterations, while IC/BC points are chosen from a fixed uniform grid, with a stage-wise sampling schedule (curriculum) that changes the relative sizes of collocation/IC/BC sets during training of fraction 0.3 transitioning from $(N_{\text{coll}}, N_{\text{IC}}, N_{\text{BC}}) = (1000, 800, 50)$ to $(1000, 100, 100)$,to stabilize early training by emphasizing the initial-condition term before switching to a more balanced allocation later.

To better align per-iteration work with APTS's two-stage structure (preconditioner + global TR), we run two full-batch optimizer steps per outer iteration for Adam and L-BFGS. Although the D local subdomain solves are independent and thus data-parallel, our current prototype executes them sequentially; consequently, wall-clock timings are not representative of the intended parallel runtime.

We use the following algorithm configurations in all experiments. **APTS:** global TR uses $\Delta_{\max} = 10$, $\Delta_{\min} = 1e^{-10}$, $\eta_1 = 0.1$, $\eta_2 = 0.5$, $\gamma_\downarrow = 0.5$, $\gamma_\uparrow = 2.0$. The preconditioner uses $J = 3$ local solves per subdomain, combined by uniform averaging; local TR-CG uses 12 CG iterations and damping 10^{-3}. **Adam:** full-batch with learning rate 9×10^{-3}. **L-BFGS:** full-batch PyTorch L-BFGS with history size $m = 10$; all other hyperparameters use the PyTorch defaults.

We report the error, measured as the RMSE between model predictions and the exact solution, along with the loss trajectories, and the preconditioner acceptance rate[2] for APTS with $D = 4$ subdomains in Figure 1. Additional results for $D = 2$ and $D = 8$ are provided in the Appendix A.1 in Figures 2–3, while the error heatmaps for the $D = 4$ subdomains can be found in Appendix A.2, Figure 4. Across all problems, APTS consistently improves over Adam and, in several cases, achieves performance that is competitive with L-BFGS in terms of accuracy. In particular, we observe that APTS can match or surpass the baselines in the decay of the solution error over outer iterations, even when the corresponding loss (residual) decay is less pronounced. The preconditioning stage achieves acceptance rates typically above 50%, indicating that the data-parallel additive correction frequently produces a descent direction and contributes materially to the overall progress of the two-stage scheme. We observe that Adam is noticeably more sensitive to collocation resampling than APTS and L-BFGS: after resampling events, Adam frequently exhibits very high increases in the objective and/or solution error before recovering, whereas APTS and L-BFGS produce smoother trajectories under the same resampling schedule. Consistent with the known sensitivity of Adam to nonstationary objectives, we observe larger transients in Adam's loss/error trajectories following resampling events compared with APTS and L-BFGS under the same schedule.

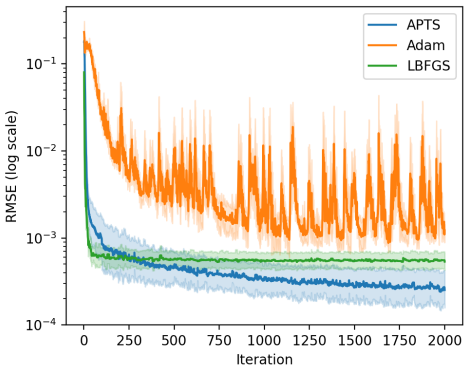
4 Limitations

We view the present study as a foundation for extending APTS to broader regimes, architectures, and genuinely parallel implementations. First, despite aligning outer-iteration counts and using two full-batch steps per outer iteration for Adam/L-BFGS, the effective computational work can still differ across methods. Second, we focus on three PDEs with a single network architecture and curriculum/resampling schedule, so behavior may change across PDE regimes (e.g., stiffness/viscosity), architectures/activations, sampling strategies, or soft vs. hard IC/BC enforcement. Lastly, our APTS variant reflects specific design choices, and systematic tests are needed to identify which components drive stability and performance.

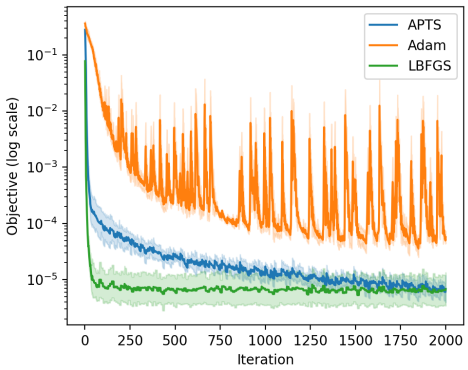
5 Conclusion

We investigated a data-parallel APTS for PINN training and compared it against full-batch Adam and L-BFGS on three PDE benchmarks. Across all problems, APTS improves over Adam and in several cases achieves performance competitive with L-BFGS in terms of solution accuracy. Notably, APTS often exhibits faster decay of the solution error over outer iterations even when the corresponding loss (residual) decreases less aggressively. A recurring discrepancy between error and loss trajectories emerges across benchmarks. Understanding when and why reductions in the PINN

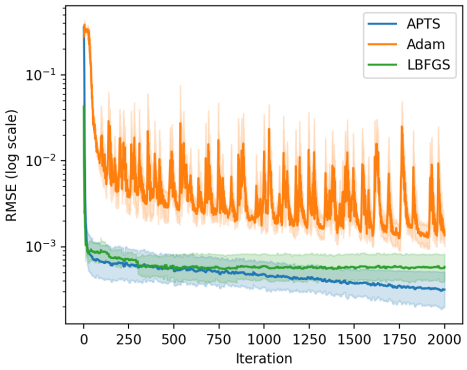
[2]The fraction of outer iterations in which the preconditioning trial step $\bar{s}_k$ is accepted in the preconditioning stage (prior to the global TR refinement).



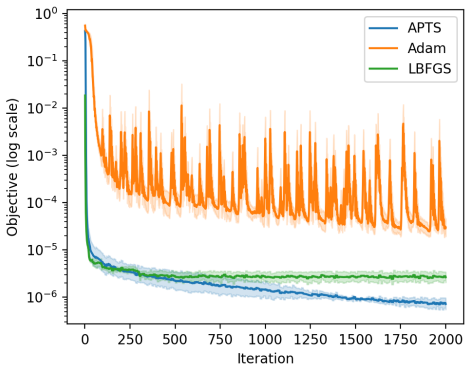
(a) Heat equation error evolution.



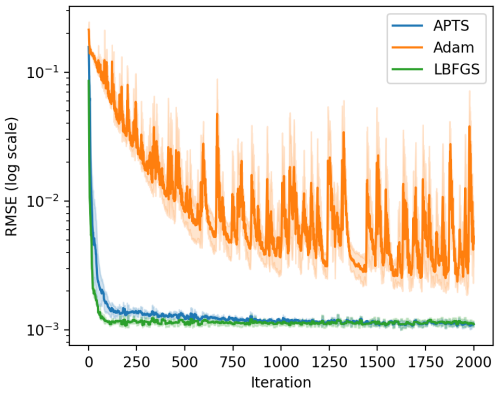
(b) Heat equation loss evolution.



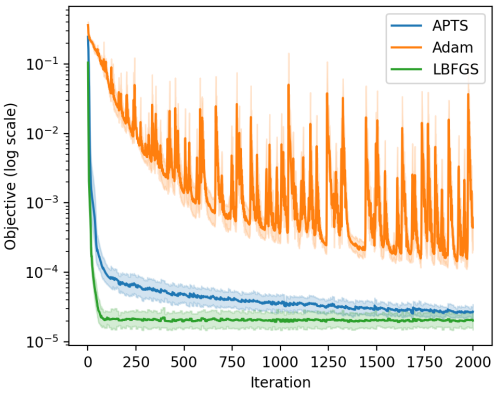
(c) Burgers equation error evolution.



(d) Burgers equation objective evolution.



(e) Laplace equation error evolution.



(f) Laplace equation objective evolution.

Figure 1: APTS ($D = 4$) on three PDE benchmarks. Left column: error vs. outer iteration. Right column: training objective $L(\theta)$ vs. outer iteration. Preconditioner acceptance rates: Heat 0.4322, Burgers 0.8857, Laplace 0.6185.

loss translate into improved solution accuracy and how this relationship depends on the optimizer is an interesting direction for future work. Overall, the results suggest that combining additive data-partitioned corrections with trust-region globalization is a promising route to more robust PINN optimization, motivating future work on truly parallel implementations and broader benchmark coverage. In the future, we intend to test the method on geometric domain decompositions (e.g., XPINNs), develop a truly parallel implementation to quantify speedups and communication overheads, and refine the algorithm, including aggregation/weighting of subdomain corrections, cheaper local/global TR iteration, and more noise-aware globalization.

REFERENCES

Xiao-Chuan Cai and David E. Keyes. Nonlinearly preconditioned inexact newton algorithms. *SIAM Journal on Scientific Computing*, 24:183–200, 2002. doi: 10.1137/S106482750037620X.

Andrew R. Conn, Nicholas I. M. Gould, and Philippe L. Toint. *Trust-Region Methods*. MPS-SIAM Series on Optimization. SIAM, 2000. doi: 10.1137/1.9780898719857.

C. Groß. *A Unifying Theory for Nonlinear Additively and Multiplicatively Preconditioned Globalization Strategies: Convergence Results and Examples From the Field of Nonlinear Elastostatics and Elastodynamics*. PhD thesis, Bonn International Graduate School, University of Bonn, Bonn, Germany, 2009.

Ameya D. Jagtap and George E. Karniadakis. Extended physics-informed neural networks (xpinns): A generalized space-time domain decomposition based deep learning framework for nonlinear partial differential equations. *Communications in Computational Physics*, 28(5):2002–2041, 2020. doi: 10.4208/cicp.oa-2020-0164.

Ameya D. Jagtap, Ehsan Kharazmi, and George E. Karniadakis. Conservative physics-informed neural networks on discrete domains for conservation laws: Applications to forward and inverse problems. *Computer Methods in Applied Mechanics and Engineering*, 365:113028, 2020. doi: 10.1016/j.cma.2020.113028.

Ben Moseley, Andrew Markham, and Tarje Nissen-Meyer. Finite Basis Physics-Informed Neural Networks (FBPINNs): A scalable domain decomposition approach for solving differential equations. 49(4):62, 2023. ISSN 1019-7168, 1572-9044. doi: 10.1007/s10444-023-10065-9.

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.

Pratik Rathore, Weimu Lei, Zachary Frangella, Lu Lu, and Madeleine Udell. Challenges in training pinns: A loss landscape perspective, 2024. ICML 2024 Oral.

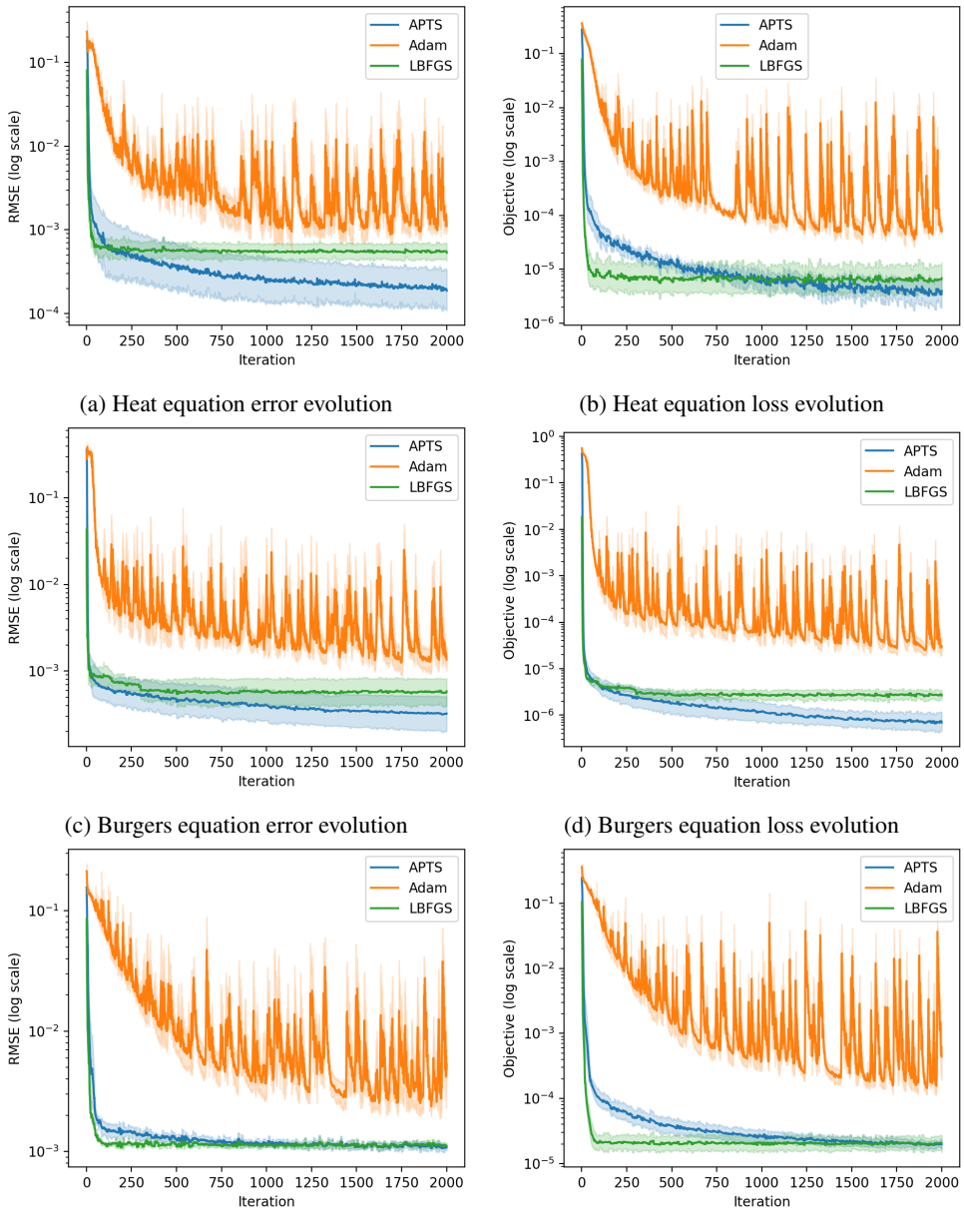
Trond Steihaug. The conjugate gradient method and trust regions in large scale optimization. *SIAM Journal on Numerical Analysis*, 20(3):626–637, 1983. doi: 10.1137/0720042.

K. Trotti, S. Cruz Alegría, R. Krause, and A. Kopaničáková. Parallel trust-region approaches in neural network training. In *Proceedings of the MATH+ Thematic Einstein Semester 2023: Mathematical Optimization for Machine Learning*, pp. 107–120. De Gruyter, Berlin, 2025. ISBN 9783111376776.

J. F. Urbán, P. Stefanou, and J. A. Pons. Why are physics-informed neural networks so hard to optimize? *Journal of Computational Physics*, 523:113656, 2025. doi: 10.1016/j.jcp.2024.113656.

A Appendix

A.1 Results for APTS with 2 and 8 subdomains



(a) Heat equation error evolution

(b) Heat equation loss evolution

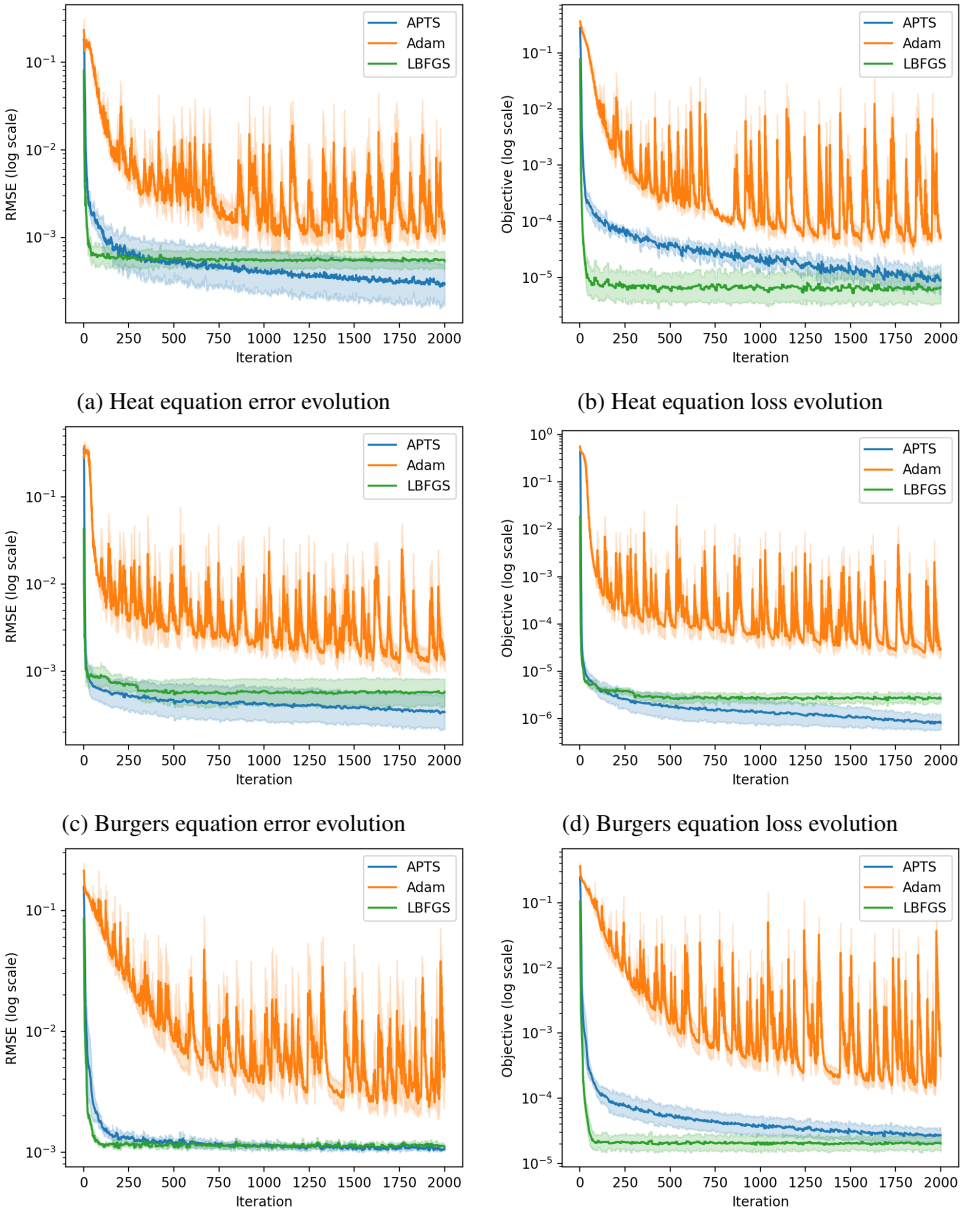
(c) Burgers equation error evolution

(d) Burgers equation loss evolution

(e) Laplace equation error evolution

(f) Laplace equation loss evolution

Figure 2: APTS ($D = 2$) on three PDE benchmarks. Left column: error vs. outer iteration. Right column: training objective $L(\theta)$ vs. outer iteration. Preconditioner acceptance rates: heat 0.6993, Burgers 0.7780, Laplace 0.8795.



(a) Heat equation error evolution

(b) Heat equation loss evolution

(c) Burgers equation error evolution

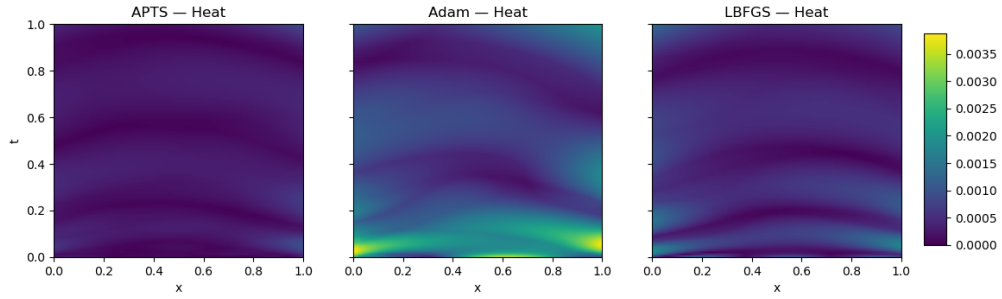
(d) Burgers equation loss evolution

(e) Laplace equation error evolution

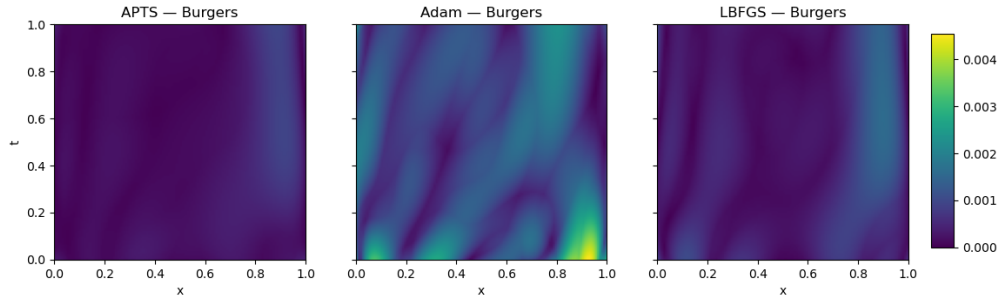
(f) Laplace equation loss evolution

Figure 3: APTS ($D = 8$) on three PDE benchmarks. Left column: error vs. outer iteration. Right column: training objective $L(\theta)$ vs. outer iteration. Preconditioner acceptance rates: Heat 0.3057, Burgers 0.8112, Laplace 0.4507.

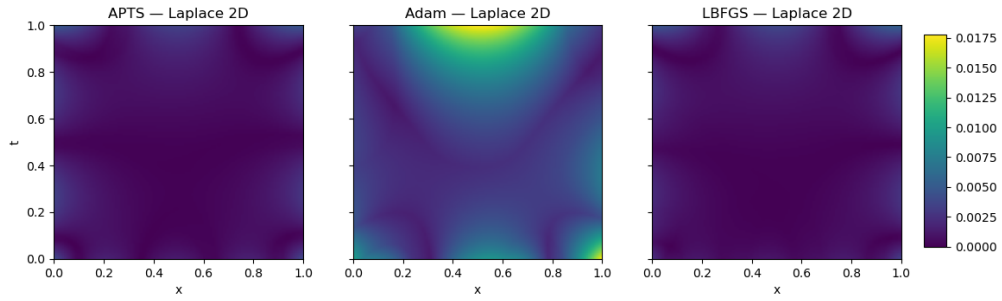
A.2 ERROR HEATMAPS



(a) Heat equation error heatmap



(b) Burgers equation error heatmap



(c) Laplace equation error heatmap

Figure 4: Error heatmaps for the three PDE benchmarks for APTS ($D = 4$ subdomain), Adam, and LBFGS.

A.3 TRUST-REGION METHODS

Trust-region (TR) methods are a class of iterative algorithms for nonlinear optimization that construct a local approximation of the objective function and minimize this approximation within a region around the current iterate where the model is considered reliable (Conn et al., 2000). Rather than taking an unrestricted step based solely on the local model, the method explicitly limits the step length to remain within a *trust region*. This mechanism provides stability and improves robustness when the local quadratic model is only accurate in a neighborhood of the current point.

Given the current iterate $\theta^{(k)}$, the objective $L(\theta)$ is locally approximated by the quadratic model

$$\psi(s) = \langle \nabla L(\theta^{(k)}), s \rangle + \frac{1}{2} \langle s, B^{(k)} s \rangle, \tag{5}$$

where $\nabla L(\theta^{(k)})$ is the gradient of the loss at the current iterate and $B^{(k)}$ is the Hessian of L at $\theta^{(k)}$ or an approximation of it (e.g. a quasi-Newton update).

The step is obtained by solving the trust-region subproblem

$$s^{(k)} = \arg\min_{s \in \mathbb{R}^n} \psi(s) \qquad \text{such that} \qquad \|s\| \le \Delta^{(k)}, \tag{6}$$

where $\Delta^{(k)}$ denotes the trust-region radius. This constraint limits the step length and ensures that the quadratic model is only trusted locally.

After computing the candidate step $s^{(k)}$, the method evaluates how well the quadratic model predicted the actual decrease in the objective. This is measured using the ratio

$$\rho^{(k)} = \frac{L(\theta^{(k)}) - L(\theta^{(k)} + s^{(k)})}{\psi(0) - \psi(s^{(k)})}. \tag{7}$$

If $\rho^{(k)}$ is close to 1, the model prediction is considered reliable, the step is accepted, and the trust-region radius may be increased. If the ratio is far from 1, the step may be rejected and the radius may be reduced. In this way, the trust-region mechanism adapts the step size dynamically based on how accurately the quadratic model represents the objective. In our implementation, the trust-region subproblem is solved approximately using Steihaug's truncated conjugate-gradient method (Steihaug, 1983), which requires only Hessian–vector products rather than forming the Hessian matrix explicitly.

A.4 Data-parallel APTS-PINN Algorithm

Algorithm 1 Data-parallel APTS for PINN training (two-stage trust-region iteration)

Require: Initial iterate θ_0 and trust-region radius Δ_0, with radius bounds $\Delta_{\min} \le \Delta_k \le \Delta_{\max}$; D subdomains, J local TR steps, resampling period r

Require: TR update parameters: acceptance thresholds $0 < \eta_1 < \eta_2 < 1$ and shrink/expand factors $0 < \gamma_\downarrow < 1 < \gamma_\uparrow$; full loss L in equation 2

for $k = 0, 1, 2, \dots$ **do**
  if $k \bmod r = 0$ **then**
    $(\mathcal{C}_k, \mathcal{I}_k, \mathcal{B}_k)$ = resample$([0,T] \times \Omega)$
  else
    $(\mathcal{C}_k, \mathcal{I}_k, \mathcal{B}_k) = (\mathcal{C}_{k-1}, \mathcal{I}_{k-1}, \mathcal{B}_{k-1})$
  end if
  Stage I: additive preconditioning (local solves)
  partition $\mathcal{C}_k = \bigsqcup_{d=1}^{D} \mathcal{C}_k^d$
  for each $d = 1, \dots, D$ **in parallel do**
    set replica $\theta_k^d \leftarrow \theta_k, \quad s_k^d \leftarrow 0, \quad \Delta_{\text{loc}} \leftarrow \Delta_k / J$
    for $j = 1, \dots, J$ **do**
      form local objective $L_k^d(\cdot)$ using $\mathcal{C}_k^d$ and shared $(\mathcal{I}_k, \mathcal{B}_k)$ ▷ equation 3
      compute a local TR step $s_{k,j}^d$ for L_k^d with radius Δ_{loc}
      update $s_k^d \leftarrow s_k^d + s_{k,j}^d; \quad \theta_k^d \leftarrow \theta_k^d + s_{k,j}^d$
    end for
  end for
  assemble preconditioner step $\bar{s}_k \leftarrow \frac{1}{D} \sum_{d=1}^{D} s_k^d; \quad \theta_{\text{trial}} \leftarrow \theta_k + \bar{s}_k$
  $(\theta_{k+1}, \Delta_{k+1}) \leftarrow$ TR-Accept$(L, \theta_k, \theta_{\text{trial}}, \Delta_k)$
  if rejected **then**
    $\theta_{k+1} \leftarrow \theta_k$
  end if
  Stage II: global trust-region refinement (full objective)
  repeat
    compute global TR step s_{k+1} for L_k at θ_{k+1} with radius Δ_{k+1}
    $\theta_{\text{trial}} \leftarrow \theta_{k+1} + s_{k+1}$
    $(\theta_{k+1}, \Delta_{k+1}) \leftarrow$ TR-Accept$(L_k, \theta_{k+1}, \theta_{\text{trial}}, \Delta_{k+1})$
  until accepted and if rejected θ_{k+1} is not updated
  enforce $\Delta_{k+1} \in [\Delta_{\min}, \Delta_{\max}]$
end for
return θ_{k+1}