Physics-Informed Residual Flows

Jephte Abijuru[1], Mayank Nagda[1], Phil Ostheimer[1], Sebastian Josef Vollmer[2], Jan C. Aurich[1], Marius Kloft[1], Sophie Fellenz[1]
[1]RPTU University Kaiserslautern-Landau, Germany
[2]DFKI Kaiserslautern, Germany
`abijuru@rptu.de`

Abstract

Physics-Informed Neural Networks (PINNs) embed physical laws into deep learning models. However, conventional PINNs often suffer from failure modes leading to inaccurate solutions. We trace these failure modes to two structural pathologies: gradient shattering, where gradients degrade with depth and provide little training signal, and flow mismatch, where training pushes predictions along trajectories that diverge from the PDE solution path. We introduce ResPINNs, which reformulate PINNs as residual flows, differentiable networks that iteratively refine their own predictions through explicit corrective steps, in the spirit of classical iterative solvers. Our analysis shows that this design mitigates both pathologies by keeping updates aligned with descent and by preserving informative gradients across depth. Across canonical PDEs and the *PINNacle* benchmark suite, residual flows improve accuracy by up to two orders of magnitude.

1 Revisiting Failure Modes in PINNs

In recent years, Physics-Informed Neural Networks (PINNs) [Raissi et al., 2019] have emerged as a neural surrogate for numerically solving partial differential equations (PDEs) by embedding PDE residuals, initial, and boundary conditions into the training loss and leveraging automatic differentiation. Despite recent advances, PINNs remain vulnerable to intrinsic failure modes. Krishnapriyan et al. [2021] document several types of PDEs that are especially challenging, often due to parameters that induce high-frequency or complex solution behaviors. In such cases, PINNs may fail to propagate initial conditions accurately. A typical manifestation is the emergence of overly smooth solutions which can minimize empirical loss while ignoring temporal dynamics. To address these shortcomings, various strategies have been proposed, including optimization techniques [Wu et al., 2024, Wang et al., 2021, Liu et al., 2025, Wong et al., 2022, Bu and Karpatne, 2021], adaptive sampling [Daw et al., 2023], architectural modifications [Zhao et al., 2024, Xu et al., 2025, Wang et al., 2021].

Orthogonal to these advances, we revisit the failure modes of PINNs from lenses of optimization dynamics and representation flow identifying two structural problems. First, *gradient shattering*: As depth increases, the input–output Jacobians of PINNs decorrelate exponentially, while their norms either vanish or explode. Since PDE residuals require repeated differentiation of the network outputs, this effect is amplified in PINNs, making optimization unstable even when the residual loss is small. Second, *flow mismatch*: training updates in latent space need not align with true descent directions, so the network can satisfy residual constraints locally while drifting globally, failing to propagate initial conditions. To address these issues, we introduce *residual flows*: networks that iteratively refine predictions through small corrections around the identity. This view connects directly to three established perspectives: (i) residual networks, where skip connections stabilize gradients; (ii) neural ODEs, where depth corresponds to integrating a continuous-time flow; and (iii) classical iterative solvers, where predictor–corrector updates progressively reduce error. In the PINN setting, these formulations coincide: residual flows stabilize Jacobians and keep updates aligned with loss descent.

Our contributions are threefold. **Diagnosis:** we trace PINN failures to *gradient shattering* and *flow mismatch*. **Reformulation and Unification:** we propose *residual flows*, a view that encompasses prior sequence-modeling designs as special cases of stepwise refinement. **Evidence:** theory and experiments show that this residual-flow structure underlies stable optimization and accurate PDE solutions.

2 Mitigating Failure Modes with Residual Flows

PINNs can minimize residual loss yet diverge from true dynamics. We interpret this as a latent-space flow problem governed by two stabilizing principles: (i) update–descent alignment and (ii) near-identity Jacobians.

Preliminaries. Let $u : \Omega \times [0,T] \to \mathbb{R}^m$ satisfy operators $\mathcal{O} \in \{\mathcal{F}, \mathcal{I}, \mathcal{B}\}$ (interior, initial, boundary) with $\mathcal{O}(u) = 0$ on corresponding regions $\chi_{\mathcal{O}}$. A PINN u_θ minimizes a weighted residual loss $L(u_\theta) = \sum_{\mathcal{O}} \lambda_{\mathcal{O}} \, \mathbb{E}_{(x,t)\sim\chi_{\mathcal{O}}} \|\mathcal{O}(u_\theta)\|^2$.

Gradient shattering. Let $J_\theta(x,t) = \nabla_{(x,t)} u_\theta(x,t) \in \mathbb{R}^{m\times(d+1)}$. Mean-field analysis (Thm. E.1, App. E) shows that correlations in J_θ decay exponentially with depth while norms vanish or explode, destroying derivative signals essential for PDE residuals. Preventing this requires Jacobians close to identity.

Residual-flow formulation. To understand gradient misalignment in PINNs, training can be viewed as a latent flow indexed by solver time k, $\frac{dz(k)}{dk} = T(z(k), k;\, x, t)$ with $z(0) = E(x,t)$. A discrete step with implicit step size α gives $z_{k+1} = z_k + T_k(z_k; \alpha)$, a small residual correction advancing z_k toward the PDE solution. Residual networks, neural ODEs, and progressive refinement schemes all instantiate this structure.

Local descent and Jacobian neutrality. For a loss $\mathcal{L}(z_k)$ with β-Lipschitz gradient:
Lemma 2.1 (Depth-aware descent). *For updates $z_{k+1} = z_k + T_k(z_k)$, $k{=}0\ldots K{-}1$, let $J_\ell = \partial z_{\ell+1}/\partial z_\ell$. If $\nabla\mathcal{L}$ is β-Lipschitz near $\{z_k\}$, then for some $\beta_k \le \beta(\prod_{\ell=k}^{K-1} \|J_\ell\|_2)^2$,*

$$\mathcal{L}(z_K) \le \mathcal{L}(z_0) + \sum_{k=0}^{K-1} \Big[\langle \nabla_{z_k} \mathcal{L}(z_k), T_k(z_k) \rangle + \tfrac{\beta_k}{2} \|T_k(z_k)\|^2 \Big].$$

Lemma 2.1 implies that the first-order loss change at step k is governed by $\langle \nabla_{z_k}\mathcal{L}(z_k), T_k(z_k)\rangle$, so each update behaves as a gradient step whose effect scales with its alignment to $-\nabla_{z_k}\mathcal{L}(z_k)$.[1] Misaligned or overly large updates cause *flow mismatch*, producing drift despite decreasing loss. Alignment alone, however, does not ensure stability: even well-aligned updates can yield vanishing or exploding gradients when Jacobian spectra drift. Residual formulations mitigate this by enforcing near-identity Jacobians.
Theorem 2.2 (Jacobian neutrality). *If $J_k = I + A_k$ with $\|A_k\|_2 \le \alpha_k < 1$, then $|\sigma_i(J_k) - 1| \le \alpha_k$ and $\kappa(J_k) \le \frac{1+\alpha_k}{1-\alpha_k}$. Thus, gradients remain bounded through depth.*

Together, these results explain stability: alignment mitigates flow mismatch, while Jacobian neutrality suppresses shattering. Proofs and detailed statements are deffered to Appendix C.

Empirical illustration. Figure 1 shows Jacobian spectra for a 1D convection PDE. Residual PINNs (ResPINNs) maintain near-unit singular values and stable conditioning, whereas standard PINNs exhibit spectral dispersion, confirming smoother latent-space evolution and reduced failure modes.

3 Empirical Evaluation

Experimental setup. We evaluate three realizations of the residual-flow formulation: (1) **ResPINN**, a discrete residual network where each block performs a correction $h_{k+1} = h_k + \alpha f(h_k; \theta_k)$;

[1]The *gradient alignment* is $\mathrm{GA}_k = \frac{\langle T_k(z_k), -\nabla_{z_k}\mathcal{L}(z_k)\rangle}{\|T_k(z_k)\| \, \|\nabla_{z_k}\mathcal{L}(z_k)\|}$; $\mathrm{GA}_k > 0$ indicates descent, $\mathrm{GA}_k = 0$ neutrality, and $\mathrm{GA}_k < 0$ ascent.

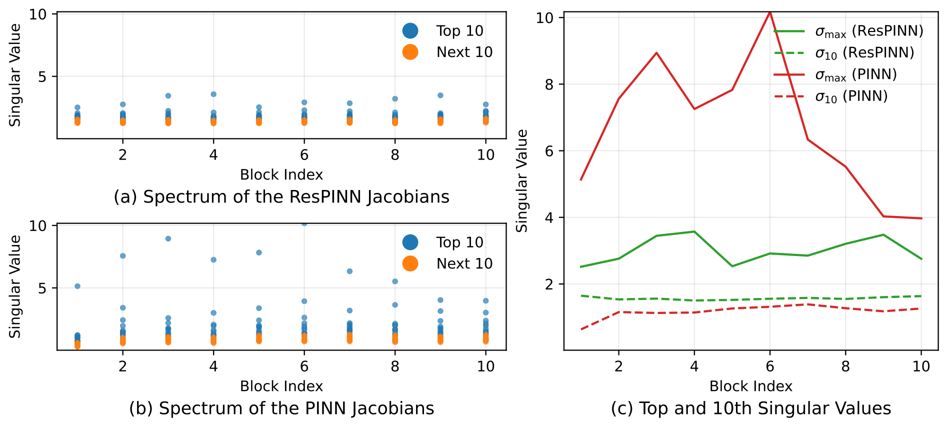

Figure 1: Jacobian spectra across depth for the 1D convection problem. ResPINNs maintain near-unit singular values and stable conditioning, while standard PINNs show spectral dispersion, consistent with the Jacobian neutrality result.

Table 1: Quantitative results on four PDE benchmarks (rMAE, rRMSE). Lower is better.

Model	Wave		Reaction		Convection		Heat	
	rMAE	rRMSE	rMAE	rRMSE	rMAE	rRMSE	rMAE	rRMSE
PINNs	0.410	0.414	0.980	0.979	0.851	0.899	0.896	0.940
FLS	0.102	0.119	0.022	0.039	0.173	0.197	0.749	0.787
PINNsFormer	0.356	0.363	0.015	0.030	0.453	0.522	0.213	0.224
KAN	0.143	0.146	0.017	0.034	0.605	0.659	0.090	0.104
PINNMamba	0.020	0.020	0.009	0.022	0.019	0.020	0.054	0.058
ResPINN (ours)	**0.013**	**0.015**	**0.005**	**0.008**	**0.003**	**0.005**	**0.004**	**0.005**

(2) **O-PINN**, a continuous-depth variant integrating $\dot{h} = f_\theta(h, t)$ with an RK4 solver; and (3) **Progressive Residual Flow**, a curriculum model that incrementally adds residual blocks, analogous to multistage refinement in numerical solvers. All models use three-layer blocks of width 64 with explicit skip connections and are trained on 101×101 space–time grids using L–BFGS and wavelet activations [Zhao et al., 2024]. Benchmarks include four canonical PDEs (Wave, Reaction–Diffusion, Convection, Heat) and the *PINNacle* suite [Zhongkai et al., 2024] of 16 additional PDEs such as Burgers, Poisson, and Navier–Stokes. Baselines cover standard PINNs [Raissi et al., 2019], FLS [Wong et al., 2022], QRes [Bu and Karpatne, 2021], KANs [Liu et al., 2025], and sequence-based models [Zhao et al., 2024, Xu et al., 2025]. Dataset handling, optimizer settings, and further details appear in Appendices F– G.2.

Benchmark performance. We evaluate whether residual flows mitigate known failure modes in PINNs. Table 1 reports relative mean absolute error (rMAE) and relative root mean squared error (rRMSE) across four canonical PDEs (Wave, Reaction–Diffusion, Convection, Heat). Classical PINNs perform poorly on Reaction and Convection, consistent with prior observations. Sequence-based models such as PINNsFormer and PINNMamba include shallow residual components but lack full residual refinement. In contrast, **ResPINN**, which stacks residual updates throughout depth, achieves the lowest errors —often by an order of magnitude—confirming that end-to-end residual flows provide consistent gains. Additional per-equation analyses appear in Appendix D.

Iterative refinement and gradient alignment. To test whether residual flows behave as iterative refiners rather than feature learners, we measure two diagnostics. First, the relative update magnitude $\|T_i(z_i)\|/\|z_i\|$ decreases monotonically with depth in ResPINNs, indicating progressively smaller corrections. Second, the cosine alignment between $T_i(z_i)$ and the negative loss gradient remains positive throughout training, consistent with descent-aligned refinement. Figure 3 summarizes these effects; full curves for additional PDEs appear in Appendix I.

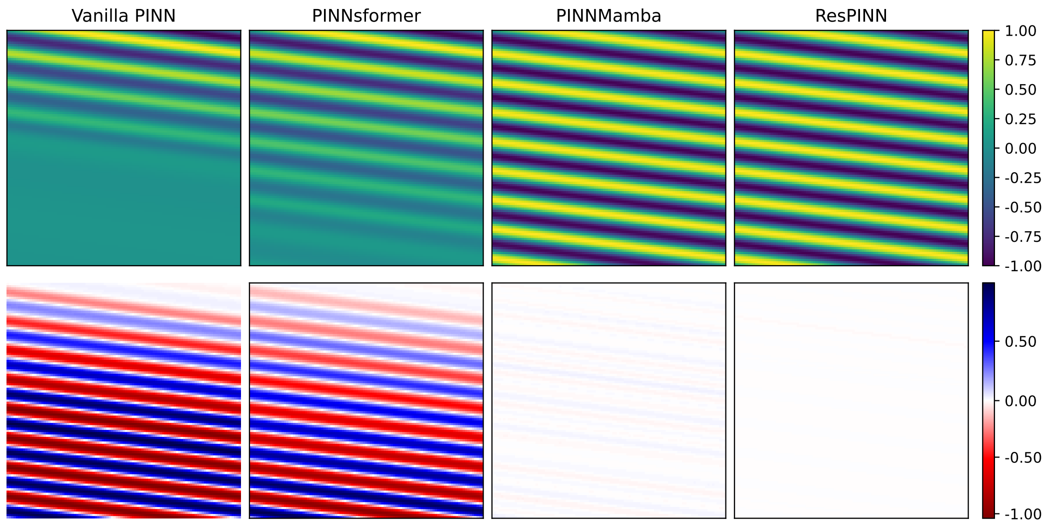

Figure 2: Qualitative comparison on Convection PDE. Top: predicted solutions. Bottom: pointwise errors. ResPINN predictions remain stable across the domain, while standard PINNs drift.

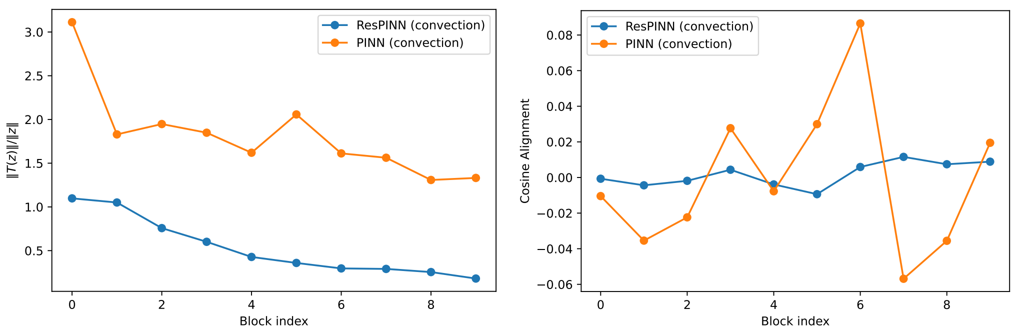

Figure 3: Left: relative update size $\|T_i(z_i)\|/\|z_i\|$ across depth. Right: gradient alignment. ResPINNs operate in the small-step, descent-aligned regime.

Ablation and generalization. We compare discrete (ResPINN) and continuous (O-PINN) residual flows under $\tanh$ and wavelet activations. Detailed tables appear in Table 4 (Appendix H). Results show complementary strengths: O-PINN benefits families requiring smooth integration, while ResPINN remains competitive across discrete tasks. On the broader *PINNacle* benchmark [Zhongkai et al., 2024], ResPINN attains stable solutions across evaluated PDEs, which span multiscale and stiff systems. Comprehensive results and resource analyses are provided in Appendix H. Appendix A provides a roadmap for reproducing the reported values.

4 Conclusion

We presented PINNs as *residual flows*, networks that solve PDEs through iterative small-step refinement. This view reveals two stabilizing factors: *gradient alignment*, ensuring descent-oriented updates, and *Jacobian neutrality*, maintaining well-conditioned mappings. Preliminary results on canonical PDEs and the *PINNacle* suite show that residual-flow architectures (ResPINN, O-PINN, and progressive variants) deliver consistent improvements and interpretable optimization behavior. This framework offers a basis for exploring how residual formulations, continuous-time limits, and solver-inspired parameterizations can unify and stabilize neural PDE solvers, as well as extend toward deeper architectures and direct residual flows in solution space.

Acknowledgement

MK and SF acknowledge support by the BMFTR award 01IS24071A, by the DFG through FOR 5359 (ID 459419731), TRR 375 (ID 511263698), SPP 2298 (Id 441826958), and SPP 2331 (441958259, 553345933, 466468799), and by the Carl-Zeiss Foundation through the initiatives AI-Care and Process Engineering 4.0.

References

David Balduzzi, Marcus Frean, Lennox Leary, JP Lewis, Kurt Wan-Duo Ma, and Brian McWilliams. The shattered gradients problem: If resnets are the answer, then what is the question? In *International conference on machine learning*, pages 342–350. PMLR, 2017.

Jie Bu and Anuj Karpatne. Quadratic residual networks: A new class of neural networks for solving forward and inverse problems in physics involving pdes. In *Proceedings of the 2021 SIAM International Conference on Data Mining (SDM)*, pages 675–683. SIAM, 2021.

Ricky TQ Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. *Advances in neural information processing systems*, 31, 2018.

Arka Daw, Jie Bu, Sifan Wang, Paris Perdikaris, and Anuj Karpatne. Mitigating propagation failures in physics-informed neural networks using retain-resample-release (R3) sampling. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett, editors, *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pages 7264–7302. PMLR, 2023. URL `https://proceedings.mlr.press/v202/daw23a.html`.

Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real NVP. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. URL `https://openreview.net/forum?id=HkpbnH9lx`.

Klaus Greff, Rupesh K. Srivastava, and Jürgen Schmidhuber. Highway and residual networks learn unrolled iterative estimation. In *International Conference on Learning Representations*, 2017. URL `https://openreview.net/forum?id=Skn9Shcxe`.

Eldad Haber and Lars Ruthotto. Stable architectures for deep neural networks. *Inverse problems*, 34 (1):014004, 2017.

Eldad Haber, Keegan Lensink, Eran Treister, and Lars Ruthotto. IMEXnet a forward stable deep neural network. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 2525–2534. PMLR, 09–15 Jun 2019. URL `https://proceedings.mlr.press/v97/haber19a.html`.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

Stanisław Jastrzebski, Devansh Arpit, Nicolas Ballas, Vikas Verma, Tong Che, and Yoshua Bengio. Residual connections encourage iterative inference. In *International Conference on Learning Representations*, 2018. URL `https://openreview.net/forum?id=SJa9iHgAZ`.

Durk P Kingma and Prafulla Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. *Advances in neural information processing systems*, 31, 2018.

Aditi Krishnapriyan, Amir Gholami, Shandian Zhe, Robert Kirby, and Michael W Mahoney. Characterizing possible failure modes in physics-informed neural networks. *Advances in Neural Information Processing Systems*, 34:26548–26560, 2021.

Zongyi Li, Daniel Zhengyu Huang, Burigede Liu, and Anima Anandkumar. Fourier neural operator with learned deformations for pdes on general geometries. *Journal of Machine Learning Research*, 24(388):1–26, 2023.

Ziming Liu, Yixuan Wang, Sachin Vaidya, Fabian Ruehle, James Halverson, Marin Soljacic, Thomas Y. Hou, and Max Tegmark. KAN: Kolmogorov–arnold networks. In *The Thirteenth International Conference on Learning Representations*, 2025. URL `https://openreview.net/forum?id=Ozo7qJ5vZi`.

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, 2021a.

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, 2021b. doi: 10.1137/19M1274067.

Yiping Lu, Bin Zhong, Jian Li, and Bin Dong. Beyond finite layer neural networks: Deep networks as dynamical systems. In *ICLR*, 2018.

Jeffrey Pennington, Samuel Schoenholz, and Surya Ganguli. The emergence of spectral universality in deep networks. In *International Conference on Artificial Intelligence and Statistics*, pages 1924–1932. PMLR, 2018.

Ben Poole, Subhaneil Lahiri, Maithra Raghu, Jascha Sohl-Dickstein, and Surya Ganguli. Exponential expressivity in deep neural networks through transient chaos. *Advances in neural information processing systems*, 29, 2016.

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.

Danilo Rezende and Shakir Mohamed. Variational inference with normalizing flows. In *International conference on machine learning*, pages 1530–1538. PMLR, 2015.

Yogesh Verma, Markus Heinonen, and Vikas Garg. ClimODE: Climate forecasting with physics-informed neural ODEs. In *The Twelfth International Conference on Learning Representations*, 2024. URL `https://openreview.net/forum?id=xuY33XhEGR`.

Sifan Wang, Yujun Teng, and Paris Perdikaris. Understanding and mitigating gradient pathologies in physics-informed neural networks. *CoRR*, abs/2001.04536, 2020. URL `https://arxiv.org/abs/2001.04536`.

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.

Sifan Wang, Bowen Li, Yuhan Chen, and Paris Perdikaris. Piratenets: Physics-informed deep learning with residual adaptive networks. *Journal of Machine Learning Research*, 25(402):1–51, 2024.

Jian Cheng Wong, Chin Chun Ooi, Abhishek Gupta, and Yew-Soon Ong. Learning in sinusoidal spaces with physics-informed neural networks. *IEEE Transactions on Artificial Intelligence*, 5(3):985–1000, 2022.

Chenxi Wu, Min Zhu, Qinyang Tan, Yadhu Kartha, and Lu Lu. A comprehensive study of non-adaptive and residual-based adaptive sampling for physics-informed neural networks. *Computer Methods in Applied Mechanics and Engineering*, 403:115671, 2023.

Haixu Wu, Huakun Luo, Yuezhou Ma, Jianmin Wang, and Mingsheng Long. Ropinn: Region optimized physics-informed neural networks. In *Advances in Neural Information Processing Systems*, 2024.

Chenhui Xu, Dancheng Liu, Yuting Hu, Jiajie Li, Ruiyang Qin, Qingxiao Zheng, and Jinjun Xiong. Sub-sequential physics-informed learning with state space model. In *Forty-second International Conference on Machine Learning*, 2025. URL `https://openreview.net/forum?id=V7VnjJxBlg`.

Ge Yang and Samuel Schoenholz. Mean field residual networks: On the edge of chaos. *Advances in neural information processing systems*, 30, 2017.

Yuan Yin, Matthieu Kirchmeyer, Jean-Yves Franceschi, Alain Rakotomamonjy, and patrick gallinari. Continuous PDE dynamics forecasting with implicit neural representations. In *The Eleventh International Conference on Learning Representations*, 2023. URL `https://openreview.net/forum?id=B73niNjbPs`.

Zhiyuan Zhao, Xueying Ding, and B. Aditya Prakash. PINNsformer: A transformer-based framework for physics-informed neural networks. In *The Twelfth International Conference on Learning Representations*, 2024. URL `https://openreview.net/forum?id=DO2WFXU1Be`.

Hao Zhongkai, Jiachen Yao, Chang Su, Hang Su, Ziao Wang, Fanzhi Lu, Zeyu Xia, Yichi Zhang, Songming Liu, Lu Lu, et al. Pinnacle: A comprehensive benchmark of physics-informed neural networks for solving pdes. *Advances in Neural Information Processing Systems*, 37:76721–76774, 2024.

A Reproducibility Statement.

All PDE setups (governing equations, domains, analytic solutions, and meshes) are detailed in Appendix F. Theoretical results and proofs appear in Appendix C, with the mean-field shattering adaptation in Appendix E. Architectural and solver specifications for ResPINN, O-PINN, and Progressive Flow are given in Appendix G, and additional alignment/refinement diagnostics are in Appendix I. PINNacle task definitions and results are reported in Appendix H. An anonymous code respository containing implementations of residual flows and scripts reproducing the experiments is available at `https://anonymous.4open.science/r/resflows-0FD5`

B Related Work

Failure Modes in PINNs Despite recent advances in PINN literature, they remain vulnerable to intrinsic failure modes. Krishnapriyan et al. [2021] document several types of PDEs that are especially challenging, often due to parameters that induce high-frequency or complex solution behaviors. In such cases, PINNs may fail to propagate initial conditions accurately. A typical manifestation is the emergence of overly smooth solutions which can minimize empirical loss while ignoring temporal dynamics. To address these shortcomings, various strategies have been proposed, including optimization techniques [Wu et al., 2024, Wang et al., 2021, Liu et al., 2025, Wong et al., 2022, Bu and Karpatne, 2021], adaptive sampling [Daw et al., 2023, Wu et al., 2023], architectural modifications [Zhao et al., 2024, Xu et al., 2025, Wang et al., 2021]. Nevertheless, these methods do not explicitly address the instability caused by noisy gradients, often referred to as gradient shattering, which can mislead training and limit robustness. We propose residual flows as a mechanism to align these gradients with the optimization.

Neural Operators. Neural operators such as DeepONet [Lu et al., 2021a] and FNOs [Li et al., 2023] are data driven *surrogate models* that approaximate the PDE solution operator from labeled data, whereas PINNs rely on enforcing PDE residuals and boundary/initial conditions. Since our work focuses on PINNs, we benchmark mainly against PINN variants and restrict our analysis to failure modes specific to this class of methods.

Flows in Machine Learning. The idea of flows is well-established in Machine Learning. Normalizing flows learn invertible maps that transform a base density into a data density by composing simple bijections and tracking Jacobian determinants for exact likelihoods [Rezende and Mohamed, 2015, Dinh et al., 2017, Kingma and Dhariwal, 2018, Chen et al., 2018]. This probabilistic goal is orthogonal to ours: we do not model densities or require invertibility. Our flow perspective instead concerns optimization dynamics of PINNs exhibiting failure modes.

Residual networks and connections. Residual connections stabilize training by composing near-identity transformations [He et al., 2016, Greff et al., 2017, Jastrzebski et al., 2018, Haber and Ruthotto, 2017, Lu et al., 2018, Chen et al., 2018]. In PINNs, they appear in several proposals, often alongside other architectural changes, so their specific contribution is unclear [Wang et al., 2020, Zhao et al., 2024, Xu et al., 2025]. PINNsFormer [Zhao et al., 2024] and PINNMamba [Xu et al., 2025] both employ residual connections but attribute gains to sequence modeling, while PirateNets [Wang et al., 2024] explore adaptive residual scaling and physics-informed initialization without connecting them to failure modes.

Continuous-depth limits and Neural ODEs. Taking residual networks to the continuous-depth limit yields Neural ODEs, parameterized by vector fields and solved numerically [Chen et al., 2018]. Connections to solver stability and residual architectures have been emphasized [Lu et al., 2018, Haber et al., 2019], and continuous-depth models have been adapted to scientific machine learning [Yin et al., 2023, Verma et al., 2024].

Positioning. Our contribution differs from likelihood-based flows and prior PINN adaptations. We explicitly characterize two structural failure modes in PINNs, gradient shattering and flow mismatch, and propose a residual flow formulation that enforces gradient alignment and Jacobian neutrality. The mechanism is architecture-agnostic: it can be instantiated as a residual stack, as a continuous-depth Neural ODE with explicit solvers, or as a purely iterative refinement scheme without ODE machinery. This unifies discrete residual nets, continuous flows, and solver-style iterations under a single stabilization principle tailored to PINNs.

C Proofs

We study feature evolution through a latent flow induced by residual transformations in continuous network time:

$$\frac{dz(k)}{dk} \;=\; T\big(z(k), k;\, x, t\big), \qquad k \in [0, K], \qquad z(0) = z_0 := E(x,t) \in \mathbb{R}^{d_h}. \tag{1}$$

Remark C.1 (Analytical surrogate). Equation equation 1 is not intended as the literal dynamics of fully connected PINNs, but as an analytical surrogate that lets us study feature evolution and gradient misalignment using the language of residual flows.

Lemma C.2 (Integral form). *If $T(\cdot,\cdot)$ is continuous, then z is a solution of equation 1 on $[0,K]$ if and only if*

$$z(k) \;=\; z_0 + \int_0^k T\big(z(\tau), \tau;\, x, t\big)\, d\tau, \qquad 0 \le k \le K. \tag{2}$$

Proof. *(⇒) Integrate equation 1 from 0 to k to obtain equation 2.*
(⇐) If equation 2 holds and $T(z(\tau),\tau)$ is continuous in τ, then by the fundamental theorem of calculus the map $k \mapsto z(k)$ is differentiable with $\frac{dz}{dk} = T(z(k),k)$ and $z(0) = z_0$, i.e., z solves equation 1. □

Theorem C.3 (Banach contraction mapping). *Let $(X, \|\cdot\|_X)$ be a Banach space and let $F : X \to X$ satisfy*

$$\|F(z) - F(z')\|_X \;\le\; c\,\|z - z'\|_X, \qquad \forall z, z' \in X,$$

for some $0 < c < 1$. Then F admits a unique fixed point $z^* \in X$, and the iterates $z^{(n+1)} = F(z^{(n)})$ converge to z^* for any initial $z^{(0)} \in X$.

Theorem C.4 (Existence and uniqueness of a solution of a residual flow). *Let $T : \mathbb{R}^{d_h} \times [0,K] \to \mathbb{R}^{d_h}$ be continuous and assume there exists $L > 0$ such that*

$$\|T(z_1,k) - T(z_2,k)\| \;\le\; L\,\|z_1 - z_2\|, \qquad \|T(z,k)\| \;\le\; L(1 + \|z\|), \tag{3}$$

for all $z, z_1, z_2 \in \mathbb{R}^{d_h}$ and $k \in [0,K]$. Then the IVP 1 admits a unique solution $z \in C([0,K], \mathbb{R}^{d_h})$.

Proof. Fix $\delta > 0$ and consider the Banach space $X = C([0,\delta], \mathbb{R}^{d_h})$ with norm $\|z\|_X = \sup_{0\le s\le\delta} \|z(s)\|$. Define the flow operator

$$(\mathcal{F}z)(k) \;:=\; z_0 + \int_0^k T(z(\tau), \tau)\, d\tau.$$

If $\mathcal{F}z = z$, then by Lemma C.2, z solves the IVP on $[0,\delta]$.

For $z, z' \in X$ and $k \in [0, \delta]$,

$$\|(\mathcal{F}z)(k) - (\mathcal{F}z')(k)\| \leq \int_0^k \|T(z(\tau), \tau) - T(z'(\tau), \tau)\| \, d\tau \leq L\delta \|z - z'\|_X.$$

Thus $\|\mathcal{F}z - \mathcal{F}z'\|_X \leq L\delta\|z - z'\|_X$, so $\mathcal{F}$ is a contraction whenever $L\delta < 1$. By Theorem C.3, $\mathcal{F}$ has a unique fixed point in X, which is the unique solution on $[0, \delta]$. Repeating the argument on successive intervals of length δ extends the solution uniquely to all of $[0, K]$. □

Definition C.5 (Discrete Residual Step)**.** Let $\Delta k > 0$ and $k_n := n\,\Delta k$ for $n = 0, \ldots, N$ with $N\Delta k = K$. The explicit Euler discretization of the residual flow $\frac{dz(k)}{dk} = T(z(k), k;\, x, t)$ with $z(0) = z_0 := E(x, t)$ is

$$z_{n+1} \;=\; z_n + \Delta k\, T(z_n, k_n;\, x, t), \qquad z_0 = E(x, t).$$

Equivalently, this is a residual update with $T_n(z_n) := \Delta k\, T(z_n, k_n;\, x, t)$.

Definition C.6 (Convergence/order)**.** Let $z(\cdot)$ denote the (unique) solution of the IVP on $[0, K]$. A time-stepping scheme producing $\{z_n\}_{n=0}^N$ is said to *converge with order p* on $[0, K]$ if there exists a constant C, independent of Δk, such that

$$\max_{0 \leq n \leq N} \big\|z(k_n) - z_n\big\| \;\leq\; C\,(\Delta k)^p.$$

Theorem C.7 (First-order convergence of the residual flows)**.** *Assume the hypotheses of existence/uniqueness hold (global Lipschitz and linear growth in z for T), and that the solution z is twice continuously differentiable on* $[0, K]$*. Let* $\{z_n\}$ *be defined by C.5. Then the discrete formulation of the residual flows converges with order* 1*:*

$$\max_{0 \leq n \leq N} \big\|z(k_n) - z_n\big\| \;\leq\; C_K\, \Delta k,$$

where C_K depends on K, the Lipschitz constant L of T in z, and $\max_{k \in [0,K]} \|\ddot{z}(k)\|$*, but is independent of* Δk*.* Sketch. *Taylor expand $z(k_{n+1}) = z(k_n) + \Delta k\, \dot{z}(k_n) + R_n$ with $\|R_n\| \leq C\,(\Delta k)^2$. Using $\dot{z}(k_n) = T(z(k_n), k_n)$ and subtracting the Euler step gives the error recurrence $e_{n+1} \leq (1 + L\Delta k)\, e_n + C\,(\Delta k)^2$, where $e_n := \|z(k_n) - z_n\|$. Apply the discrete Grönwall lemma to obtain $e_n \leq C\, \frac{e^{Lk_n} - 1}{L}\, \Delta k \leq C\, \frac{e^{LK} - 1}{L}\, \Delta k$.*

Proposition C.8 (Gradient alignment in residual flows)**.** *Let $\mathcal{L} : \mathbb{R}^{d_h} \to \mathbb{R}$ be continuously differentiable, and let $z : [0, K] \to \mathbb{R}^{d_h}$ be a continuously differentiable solution of the residual flow IVP equation 1. Then, for all $k \in [0, K]$,*

$$\frac{d}{dk}\, \mathcal{L}\big(z(k)\big) \;=\; \big\langle \nabla\mathcal{L}\big(z(k)\big),\, T\big(z(k), k;\, x, t\big)\big\rangle. \tag{4}$$

1. *If*
$$\big\langle \nabla\mathcal{L}(z(k)),\, T(z(k), k)\big\rangle \;\leq\; 0 \quad \textit{for all } k \in [0, K], \tag{5}$$
then $\mathcal{L}(z(k))$ is nonincreasing on $[0, K]$.

2. *If there exists a constant $c \in (0, 1]$ such that*
$$\frac{\big\langle T(z(k), k),\, -\nabla\mathcal{L}(z(k))\big\rangle}{\|T(z(k), k)\|\, \|\nabla\mathcal{L}(z(k))\|} \;\geq\; c \quad \textit{and} \quad \|T(z(k), k)\| > 0 \quad \textit{for all } k \in I \subset [0, K], \tag{6}$$
then $\mathcal{L}(z(k))$ is strictly decreasing on I.

Proof. The chain rule gives equation 4. Under equation 5, $\frac{d}{dk}\mathcal{L}(z(k)) \leq 0$ for all k, so $\mathcal{L}(z(k))$ is nonincreasing.

For equation 6, write

$$\frac{d}{dk}\, \mathcal{L}(z(k)) = -\; \|\nabla\mathcal{L}(z(k))\|\, \|T(z(k), k)\|\, \frac{\langle T(z(k), k), -\nabla\mathcal{L}(z(k))\rangle}{\|T(z(k), k)\|\, \|\nabla\mathcal{L}(z(k))\|} \tag{7}$$

$$\leq\; -c\, \|\nabla\mathcal{L}(z(k))\|\, \|T(z(k), k)\|. \tag{8}$$

On any interval I where $c > 0$ and $\|T(z(k), k)\| > 0$, the right-hand side is strictly negative, hence $\mathcal{L}(z(k))$ is strictly decreasing on I. □

Proof of Lemma 2.1. Since $\mathcal{L}$ has β-Lipschitz continuous gradient, we have for any u, v in a neighborhood of z_k:

$$\mathcal{L}(v) \le \mathcal{L}(u) + \langle \nabla \mathcal{L}(u), v - u \rangle + \tfrac{\beta}{2}\|v - u\|^2.$$

Apply this inequality with $u = z_k$ and $v = z_{k+1} = z_k + T_k(z_k)$:

$$\mathcal{L}(z_{k+1}) \le \mathcal{L}(z_k) + \langle \nabla_{z_k} \mathcal{L}(z_k), T_k(z_k) \rangle + \tfrac{\beta}{2}\|T_k(z_k)\|^2.$$

To capture the effect of depth, note that subsequent updates depend on how T_j is transformed through the Jacobians $J_\ell = \partial z_{\ell+1}/\partial z_\ell$. The gradient at z_k is related to the gradient at z_{k+1} by the chain rule:

$$\nabla_{z_k} \mathcal{L}(z_{k+1}) = J_k^\top \nabla_{z_{k+1}} \mathcal{L}(z_{k+1}).$$

Rolling this back from step K to step k shows that each local Lipschitz constant is scaled by the squared operator norms of the Jacobians:

$$\|\nabla^2 \mathcal{L}(z_k)\|_2 \;\le\; \beta \Big(\prod_{\ell=k}^{K-1} \|J_\ell\|_2 \Big)^2.$$

Therefore there exists a local smoothness constant $\beta_k \le \beta \big(\prod_{\ell=k}^{K-1} \|J_\ell\|_2 \big)^2$ such that

$$\mathcal{L}(z_{k+1}) \le \mathcal{L}(z_k) + \langle \nabla_{z_k} \mathcal{L}(z_k), T_k(z_k) \rangle + \tfrac{\beta_k}{2}\|T_k(z_k)\|^2.$$

This completes the proof. □

D Dissecting Existing Approaches to Failure Modes

A central challenge in PINNs for time-dependent PDEs is the propagation of initial conditions across time. Several recent works have sought to address this by introducing explicit sequence modeling. For example, Krishnapriyan et al. [2021] proposed recursive sequence-to-sequence training, rolling solutions forward in time with separate networks. While effective for short horizons, this strategy is memory- and compute-intensive, and does not generalize reliably outside the training window. More recent approaches have adapted modern sequence architectures: Zhao et al. [2024] introduced a transformer-based framework (PINNsFormer), while Xu et al. [2025] proposed state-space models (PINNMamba). Both report improved accuracy and robustness, attributing their gains to the ability of attention or structured recurrence to capture long-range temporal dependencies.

At first glance, these results seem to suggest that sophisticated sequence modules are essential for overcoming failure modes in time-dependent PINNs. Yet this conclusion is not entirely satisfying: improvements could equally stem from side effects such as increased parameterization, altered optimization dynamics, or more flexible local mappings. In other words, what appears as a benefit of "long-range temporal modeling" may instead be an artifact of broader architectural changes. This motivates a sharper question: *are sequence modules truly the driving factor behind the reported improvements, or are we attributing gains to the wrong mechanism?* To probe this, we designed a controlled ablation study. Training setup, initialization, and sampling were kept fixed, and only the internal sequence modules were varied. Specifically, self-attention and state-space operators were replaced with deliberately simple local mappings (a linear projection or a shallow MLP), with parameter counts carefully matched within $\pm 10\%$. This isolates the effect of explicit sequence modeling from confounding factors such as model capacity or optimization differences.

The evidence in Table 2 challenges the conventional explanation. The table compares both transformer-based (PINNsFormer) and state-space–based (PINNMamba) architectures against ablated versions where their sequence modules are removed or replaced with simpler alternatives. For PINNsFormer, the encoder is retained while attention is stripped out and substituted either with a linear projection or a shallow MLP. For PINNMamba, the state-space operator is removed outright or replaced by an MLP with matched parameter count. Across all three PDE benchmarks, these simplified variants perform comparably to the original models, indicating that explicit attention or structured recurrence is not essential for maintaining accuracy. This suggests that the improvements attributed to sophisticated sequence modules may instead arise from a different architectural mechanism.

Across both transformer- and state-space–based PINNs, one component remains consistent: the use of *residual pathways* that carry predictions forward through incremental corrections. Unlike attention

Table 2: Ablation study on Convection, Reaction and Wave PDEs. The relative MAE values are reported. Removing attention and replacing it with linear mappings preserves or even improves performance, despite the drastic reduction in complexity.

Model	Convection (rMAE)	Reaction (rMAE)	Wave (rMAE)
PINNsFormer (Original)	0.510	0.015	0.270
Encoder Only	0.043	0.017	0.058
-Attention + Linear	0.012	0.022	0.022
-Attention + MLP	0.009	0.016	0.142
PINNMamba (Original)	0.019	0.010	0.020
-SSM	0.012	0.013	0.029
-SSM+MLP	0.063	0.014	0.015

or structured recurrence, these pathways are present in every variant tested, including the simplified ablations. This observation points to residual connections—not sequence modules—as the common mechanism underlying stability and accuracy.

Why might residual pathways play such a critical role? At a high level, they enforce an update rule that keeps each layer close to the identity, nudging predictions forward through small, controlled steps rather than drastic transformations. This structure has several consequences that help explain the observed robustness:

(H1) Because updates are incremental, optimization becomes more stable: each layer only needs to make small corrections, reducing the risk of divergence.

(H2) The skip connections implicit in residual design bias the layer Jacobians toward the identity, which mitigates gradient shattering and helps preserve information across depth.

(H3) The repeated corrections accumulate like iterations of a solver, progressively refining the solution in the manner of predictor–corrector schemes.

Taken together, these hypotheses recast the source of robustness in time-dependent PINNs: not the sophistication of sequence modules, but the refinement dynamics induced by residual flows. In the remainder of this paper, we have put these hypotheses to the test.

E Mean-field gradient shattering for PINN Jacobians

Our analysis of gradient shattering follows directly from the mean-field studies of deep random networks by Poole et al. [2016], Balduzzi et al. [2017], Pennington et al. [2018], and Yang and Schoenholz [2017]. We adapt their derivations to the input–output Jacobians relevant for PINNs.

Theorem E.1 (Adapted from prior work on shattered gradients)**.** *Consider a depth-L, width-n fully-connected network with random Gaussian initialization as in Poole et al. [2016], Balduzzi et al. [2017]. Let $J_\theta(z) = \nabla_z u_\theta(z)$ denote the input–output Jacobian at input z. In the mean-field limit $n \to \infty$ the following hold:*

(A) ***Exponential decorrelation.*** *For nearby inputs z, z', correlations between Jacobians decay exponentially with depth:*

$$\mathbb{E}\big[\cos(J_\theta(z), J_\theta(z'))\big] = \mathcal{O}(\rho^L), \quad \rho \in (0,1).$$

(B) ***Norm growth/decay.*** *Jacobian norms scale exponentially with depth:*

$$\mathbb{E}\|J_\theta(z)\|_F^2 = \Theta(\gamma^L),$$

with $\gamma = 1$ only on the edge-of-chaos manifold; generically $\gamma \neq 1$, yielding vanishing or exploding norms.

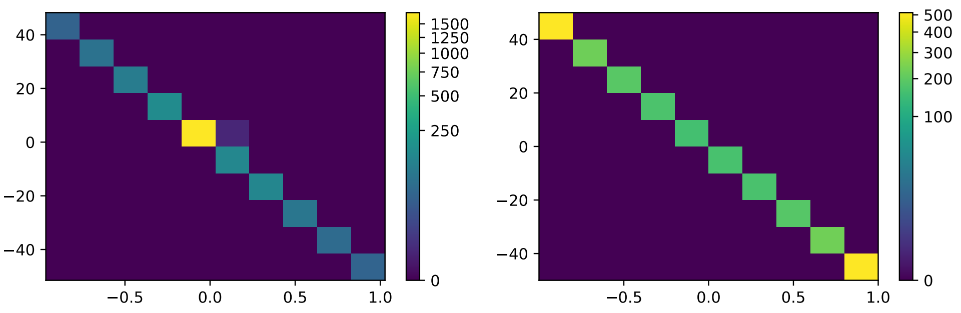

Figure 4: A 2D histogram of input-output Jacobian for convection PDE evaluated on a 50x50 equispaced grid. (left) A trained PINN. Few gradients follow the linearity of the solution while the most mass is concentrated around 0 which is a sign of vanishing gradients. (right) The Jacobian of the analytical solution.

Proof sketch. The argument mirrors Poole et al. [2016], Pennington et al. [2018]. Pre-activations converge to Gaussian processes in the mean-field limit, and input sensitivities evolve via multiplicative recursions depending on $\mathbb{E}[\phi'(u)^2]$. Cross-input correlations shrink by a factor $\rho < 1$ per layer, while sensitivity norms scale by γ. Full details can be found in the cited works; here we simply specialize the analysis to the input–output Jacobians of PINNs.

Empirical illustration. The mean-field analysis predicts vanishing or exploding Jacobian norms and exponential loss of correlation across nearby inputs. Figure 4 provides an empirical counterpart: for the convection PDE, we plot the distribution of input–output Jacobian entries on a 50×50 evaluation grid. For a trained PINN, most Jacobian values concentrate near zero, indicating collapsed sensitivities, with only a few gradients reflecting the true structure of the solution. By contrast, the analytical Jacobian remains well spread, showing the expected variation across space–time. This behavior has been also observed on other PDES where PINNs exhibit failure modes.

F PDE Setups and Metrics

F.1 Metrics

In our experiments, we report three metrics: the training loss (defined in Eq. (2)), the relative mean absolute error (rMAE), and the relative root mean squared error (rMSE). For a set of evaluation points $\mathcal{S}$, model prediction u_θ, and ground-truth solution u^*, we define

$$\mathrm{rMAE} = \frac{\sum_{x\in\mathcal{S}} \left|u_\theta(x) - u^*(x)\right|}{\sum_{x\in\mathcal{S}} \left|u^*(x)\right|}, \qquad \mathrm{rMSE} = \sqrt{\frac{\sum_{x\in\mathcal{S}} \left(u_\theta(x) - u^*(x)\right)^2}{\sum_{x\in\mathcal{S}} \left(u^*(x)\right)^2}}. \tag{9}$$

Note that both $u_\theta(x)$ and $u^*(x)$ can take positive or negative values; consequently, rMAE and rMSE may exceed 1.

F.2 Benchmarks

To comprehensively test our algorithm, we include four benchmarks. The first three correspond to canonical PDEs widely used in the PINN literature (see Figure 7), while the last one is the large-scale *PINNacle* benchmark Zhongkai et al. [2024].

1D–Reaction. This one-dimensional nonlinear ODE models chemical reactions:

$$\frac{\partial u}{\partial t} - \rho u(1-u) = 0, \quad x \in (0, 2\pi),\ t \in (0, 1),$$

with initial and boundary conditions

$$u(x,0) = \exp\left(-\frac{(x-\pi)^2}{2(\pi/4)^2}\right), \quad u(0,t) = u(2\pi,t).$$

The analytic solution is

$$u(x,t) = \frac{h(x)e^{\rho t}}{h(x)e^{\rho t} + 1 - h(x)}, \quad h(x) = \exp\left(-\frac{(x-\pi)^2}{2(\pi/4)^2}\right),$$

with $\rho = 5$. Prior work Raissi et al. [2019], Krishnapriyan et al. [2021] identified this case as a "PINN failure mode" due to the nonlinear term, and its sharp interior boundary adds further difficulty. Following PINNsFormer Xu et al. [2025], we sample 101 points on the initial/boundary sets and a 101×101 grid on the residual domain. Evaluation uses the same mesh.

1D–Wave. A standard hyperbolic PDE from acoustics and fluid dynamics:

$$\frac{\partial^2 u}{\partial t^2} - 4\frac{\partial^2 u}{\partial x^2} = 0, \quad x \in (0,1),\ t \in (0,1),$$

with initial and boundary conditions

$$u(x,0) = \sin(\pi x) + \tfrac{1}{2}\sin(\beta\pi x), \quad \frac{\partial u(x,0)}{\partial t} = 0, \quad u(0,t) = u(1,t) = 0.$$

The analytic solution is

$$u(x,t) = \sin(\pi x)\cos(2\pi t) + \tfrac{1}{2}\sin(\beta\pi x)\cos(2\beta\pi t),$$

with $\beta = 3$. Compared to Reaction and Convection, the solution is smoother, making it easier for deep models. Training/evaluation meshes are sampled as in Reaction.

1D–Convection. A hyperbolic PDE relevant in fluids, atmosphere, and heat transfer:

$$\frac{\partial u}{\partial t} + \beta\frac{\partial u}{\partial x} = 0, \quad x \in (0,2\pi),\ t \in (0,1),$$

with

$$u(x,0) = \sin(x), \quad u(0,t) = u(2\pi,t).$$

The analytic solution is $u(x,t) = \sin(x - \beta t)$, where we set $\beta = 50$. Despite its simple closed form, this problem is challenging for PINNs due to the high-frequency oscillations and sharp loss landscape Krishnapriyan et al. [2021]. Training/evaluation meshes follow the same setup as above.

Heat Equation: The heat equation is a second-order parabolic PDE that describes heat distribution in a given region over time. It is a classic example of a diffusive system, which presents challenges related to numerical stiffness.

- **Equation**: $\frac{\partial u}{\partial t} = \alpha\frac{\partial^2 u}{\partial x^2}$, with $\alpha = 0.1$.
- **Domain**: $(x,t) \in [0,1] \times [0,1]$.
- **Initial Condition**: $u(x,0) = \sin(\pi x)$.
- **Boundary Conditions**: $u(0,t) = 0$ and $u(1,t) = 0$ (Dirichlet).
- **Analytical Solution**: $u(x,t) = \sin(\pi x)e^{-\alpha\pi^2 t}$.

PINNacle. The fourth benchmark is *PINNacle* Zhongkai et al. [2024], built on DeepXDE Lu et al. [2021b]. It comprises 20 PDE tasks covering fluid dynamics, heat conduction, nonlinear and multiscale phenomena, and high-dimensional settings. We found that several subtasks are unsolved by existing methods (e.g., Heat–2d-LT, NS–2d-LT, Wave–2d-MS, Kuramoto–Sivashinsky). These involve long-time dynamics or high-order derivatives, which present challenges beyond those noted in the original paper. To focus on training paradigms rather than backbone design, we omit these four hardest cases and evaluate on the remaining 16 tasks. Dataset details are summarized in Table 3.

Table 3: PDE benchmarks from PINNacle Zhongkai et al. [2024]. We list input dimensionality, training/testing sizes, and representative simplified equations. All PDEs here are second-order. Full formalizations, coefficient meanings, and boundary/initial conditions appear in Zhongkai et al. [2024].

PDE	Dimension	N_{train}	N_{test}	Key Equation
Burgers	1D+Time (1d-C)	16384	12288	$\frac{\partial u}{\partial t} + u \cdot \nabla u - \nu \Delta u = 0$
	2D+Time (2d-C)	98308	82690	same form in 2D
Poisson	2D (2d-C)	12288	10240	$-\Delta u = 0$
	2D (2d-CG)	12288	10240	$-\Delta u + k^2 u = f(x, y)$
	3D (3d-CG)	49152	40960	$-\mu_i \Delta u + k_i^2 u = f(x, y, z),\ i = 1, 2$
	2D (2d-MS)	12288	10329	$-\nabla(a(x)\nabla u) = f(x, y)$
Heat	2D+Time (2d-VC)	65536	49189	$\frac{\partial u}{\partial t} - \nabla(a(x)\nabla u) = f(x, t)$
	2D+Time (2d-MS)	65536	49189	$\frac{\partial u}{\partial t} - \frac{1}{(500\pi)^2} u_{xx} - \frac{1}{\pi^2} u_{yy} = 0$
	2D+Time (2d-CG)	65536	49152	$\frac{\partial u}{\partial t} - \Delta u = 0$
Navier–Stokes	2D (2d-C)	14337	12378	$u \cdot \nabla u + \nabla p - \frac{1}{Re}\Delta u = 0,\ \nabla \cdot u = 0$
	2D (2d-CG)	14055	12007	same form
Wave	1D+Time (1d-C)	12288	10329	$u_{tt} - 4u_{xx} = 0$
	2D+Time (2d-CG)	49170	42194	$\left[\nabla^2 - \frac{1}{c(x)} \frac{\partial^2}{\partial t^2}\right] u(x, t) = 0$
Chaotic (GS)	2D+Time	65536	61780	$\begin{cases} u_t = \varepsilon_1 \Delta u + b(1-u) - uv^2, \\ v_t = \varepsilon_2 \Delta v - dv + uv^2 \end{cases}$
High-dim	5D (P-Nd)	49152	67241	$-\Delta u = \frac{\pi^2}{4} \sum_{i=1}^{n} \sin\left(\frac{\pi}{2} x_i\right)$
	5D+Time (H-Nd)	65537	49152	$\frac{\partial u}{\partial t} = k\Delta u + f(x, t)$

G ODE Solvers and Residual Flows

For completeness, we recall the connection between residual updates and classical numerical ODE solvers. Consider an ODE

$$\frac{dh(t)}{dt} = f(h(t), t), \qquad h(0) = h_0.$$

G.1 Residual Flow Solvers

Forward Euler. The simplest explicit solver advances in steps of size $\alpha > 0$ via

$$h_{k+1} = h_k + \alpha\, f(h_k, t_k).$$

This is precisely the form of a residual block: each step applies a correction around the identity.

Runge–Kutta (RK4). Higher-order solvers reduce truncation error by evaluating f at intermediate points. The classical fourth-order Runge–Kutta scheme computes

$$\begin{aligned} k_1 &= f(h_k, t_k), \\ k_2 &= f(h_k + \tfrac{\alpha}{2} k_1, t_k + \tfrac{\alpha}{2}), \\ k_3 &= f(h_k + \tfrac{\alpha}{2} k_2, t_k + \tfrac{\alpha}{2}), \\ k_4 &= f(h_k + \alpha k_3, t_k + \alpha), \\ h_{k+1} &= h_k + \tfrac{\alpha}{6}(k_1 + 2k_2 + 2k_3 + k_4). \end{aligned}$$

ResPINNs correspond to Euler-like discrete updates , while *O-PINNs* instantiate the continuous limit using RK4 integration with weight sharing. Implementattion details follow.

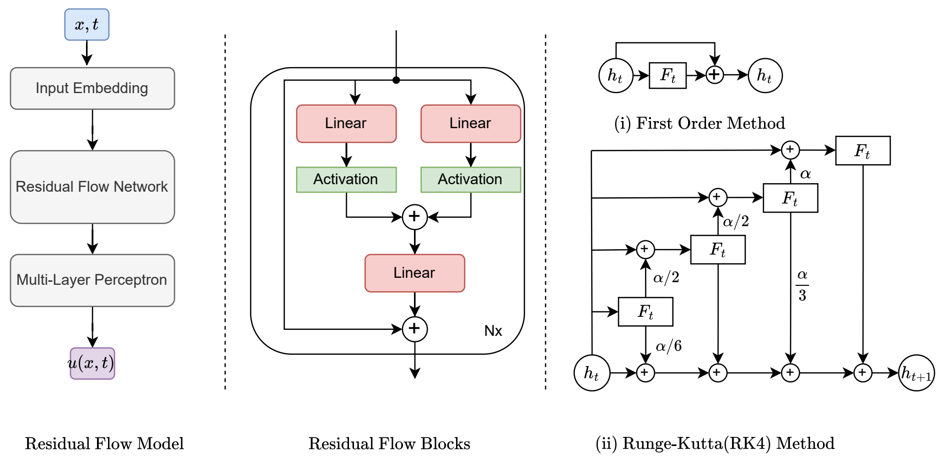

Figure 5: ResPINN overview. Inputs (x, t) are encoded to a latent state $h(0)$, which is iteratively refined by a residual flow in pseudo-time s. The flow is realized either as a stacked residual (Euler) network or as a higher-order explicit solver RK4. The terminal state $h(S)$ is decoded to the PDE solution $u(x, t)$.

	Wave		Reaction		Convection	
Model	rMAE	rRMSE	rMAE	rRMSE	rMAE	rRMSE
O-PINN + tanh	0.038	0.039	0.018	0.035	0.014	0.016
O-PINN + wavelet	0.053	0.059	0.003	0.005	0.003	0.003
ResPINN + tanh	0.030	0.030	0.008	0.017	0.015	0.016
ResPINN + wavelet	0.070	0.074	0.008	0.009	0.006	0.006

Table 4: Ablation on activation functions for continuous (O-PINN) and discrete (ResPINN) residual flow models. Results are reported on Wave, Reaction, and Convection PDEs using relative rMAE and rRMSE.

G.2 Implementation of Residual Flows

ResPINN (discrete residual stack). A fixed-depth network composed of $K = 10$ residual blocks, each block containing three fully connected layers of width 64 with a skip connection. A linear encoder maps inputs to latent space, and a single fully connected output head maps back to the PDE solution.

O-PINN (continuous residual flow). Uses the same residual block as the vector field f_θ, but instead of stacking layers explicitly, the dynamics are integrated with RK4. This yields a continuous-depth model whose trajectory corresponds to an effectively deeper residual flow.

Progressive Flow. Starts with three residual blocks and adds two new blocks at each training stage while freezing earlier ones. Both encoder and decoder are linear projections, ensuring that representational capacity resides in the blocks. At each stage, the final projection layer is re-initialized and trained as a predictor of the PDE solution, providing a direct probe of iterative refinement.

An overview of ResPINN and O-PINN archirectures is shown in Figure 5.

H Ablation and Generalization

H.1 Residual Networks and Neural ODEs

To disentangle the effect of discretization from architectural or activation choices, we compare the continuous-depth formulation (*O-PINN*, integrated with a fixed-RK4 ODE solver) against its

Table 5: Results on PINNacle. Baseline results are from Wu et al. [2024], Xu et al. [2025]. OOM means Out-of-Memory.

	PINN		PINNsFormer		PINNMamba		ResPINN	
Equation	rMAE	rRMSE	rMAE	rRMSE	rMAE	rRMSE	rMAE	rRMSE
Burgers 1d-C	1.1e-2	3.3e-2	9.3e-3	1.4e-2	3.7e-3	1.1e-3	4.6e-3	1.4e-3
Burgers 2d-C	4.5e-1	5.2e-1	OOM	OOM	OOM	OOM	OOM	OOM
Poisson 2d-C	7.5e-1	6.8e-1	7.2e-1	6.6e-1	6.2e-1	5.7e-1	7.8e-1	7.1e-1
Poisson 2d-CG	5.4e-1	6.6e-1	5.4e-1	6.3e-1	1.2e-1	1.4e-1	4.4e-3	8.6e-3
Poisson 3d-CG	4.2e-1	5.0e-1	OOM	OOM	OOM	OOM	OOM	OOM
Poisson 2d-MS	7.8e-1	6.4e-1	1.3e+0	1.1e+0	7.2e-1	6.0e-1	9.0e-1	7.5e-1
Heat 2d-VC	1.2e+0	9.8e-1	OOM	OOM	OOM	OOM	OOM	OOM
Heat 2d-MS	4.7e-2	6.9e-2	OOM	OOM	OOM	OOM	6.5e-3	4.5e-3
Heat 2d-CG	2.7e-2	2.3e-2	OOM	OOM	OOM	OOM	OOM	OOM
NS 2d-C	6.1e-2	5.1e-2	OOM	OOM	OOM	OOM	OOM	OOM
NS 2d-CG	1.8e-1	1.1e-1	1.0e-1	7.0e-2	1.1e-2	7.8e-3	1.4e-2	9.8e-3
Wave 1d-C	5.5e-1	5.5e-1	5.0e-1	5.1e-1	1.0e-1	1.0e-1	3.4-2	3.7e-2
Wave 2d-CG	2.3e+0	1.6e+0	OOM	OOM	OOM	OOM	OOM	OOM
Chaotic GS	2.1e-2	9.4e-2	OOM	OOM	OOM	OOM	OOM	OOM
High-dim PNd	1.2e-3	1.1e-3	OOM	OOM	OOM	OOM	OOM	OOM
High-dim HNd	1.2e-2	5.3e-3	OOM	OOM	OOM	OOM	OOM	OOM

discrete counterpart (*ResPINN*), each trained with either $\tanh$ or wavelet activations. This ablation allows us to test whether the improvements stem from the residual flow discretization itself or from particular activation functions. The results in Table 4 show that O-PINN and ResPINN exhibit complementary strengths: the continuous formulation benefits some PDE families (especially with wavelet activations), while discrete residual stacks remain competitive elsewhere.

H.2 Experiments on Complex Problems

To assess generalization, we evaluate on *PINNacle* [Zhongkai et al., 2024]. On challenging multiscale tasks, baselines such as PINNsFormer [Zhao et al., 2024] and PINNMamba [Xu et al., 2025] either fail to converge or run into out-of-memory errors, whereas *ResPINN* trains successfully while maintaining comparable accuracy on the remaining tasks. Details of the PINNacle experiments are shown in Table 5.

I Additional Alignment plots

Gradient Alignment To validate the theoretical analysis in Section 2, we examine how residual updates align with the local loss gradient across depth. Figures 7 and 8 plot the cosine and inner-product measures of alignment for the 1D convection problem. In ResPINNs, both quantities remain near zero, consistent with small-step, near-isometric transformations that stabilize optimization without oversteering along the loss gradient. Standard PINNs, by contrast, show large oscillations and sign changes, indicating alternating over- and under-alignment that amplifies gradient noise and destabilizes propagation. These trends empirically support the view that residual flows operate in a descent-aligned but numerically stable regime, as predicted by the depth-aware descent lemma and Jacobian neutrality theorem.

Progressive residual refinement. Figures 9 and 10 illustrate the stagewise behavior of progressive residual flows. Each training stage appends new residual blocks while freezing earlier ones, forming an explicit refinement hierarchy. For the convection problem with $\beta = 50$, early stages reproduce typical PINN failure modes, failing to capture temporal dynamics, whereas later stages introduce increasingly accurate temporal features as residual corrections accumulate. Similarly, on the wave equation, absolute error maps show systematic error reduction with depth, indicating that each residual stage refines rather than relearns the solution. These patterns empirically support the interpretation of

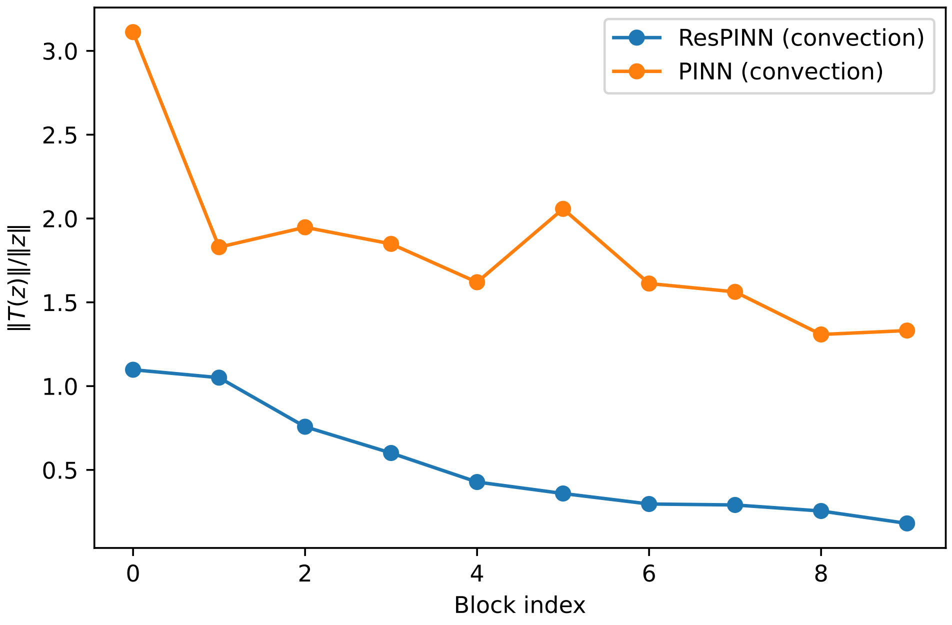

Figure 6: Relative transformation magnitude $\|T(z_k)\|/\|z_k\|$ per block for the convection problem. ResPINNs keep ratios near unity, suppressing spectral growth and stabilizing gradient flow. In contrast, PINNs amplify inputs more strongly, reflecting anisotropy and poor conditioning.

progressive residual flows as multistage iterative solvers that stabilize training through incremental correction.

J Qualitative Error Analysis

Figures 11 and 12 visualize predicted solutions and corresponding pointwise errors for the wave and reaction PDEs. Across both equations, residual-flow PINNs preserve accurate propagation of temporal and spatial features, maintaining stability throughout the domain. Baseline models, including standard PINNs and sequence-based variants, exhibit localized drift or smoothing, consistent with gradient misalignment and Jacobian instability discussed in Section 2. These maps complement the quantitative benchmarks by showing that residual refinement mitigates characteristic PINN failure modes even in regions of steep gradients or oscillatory behavior.

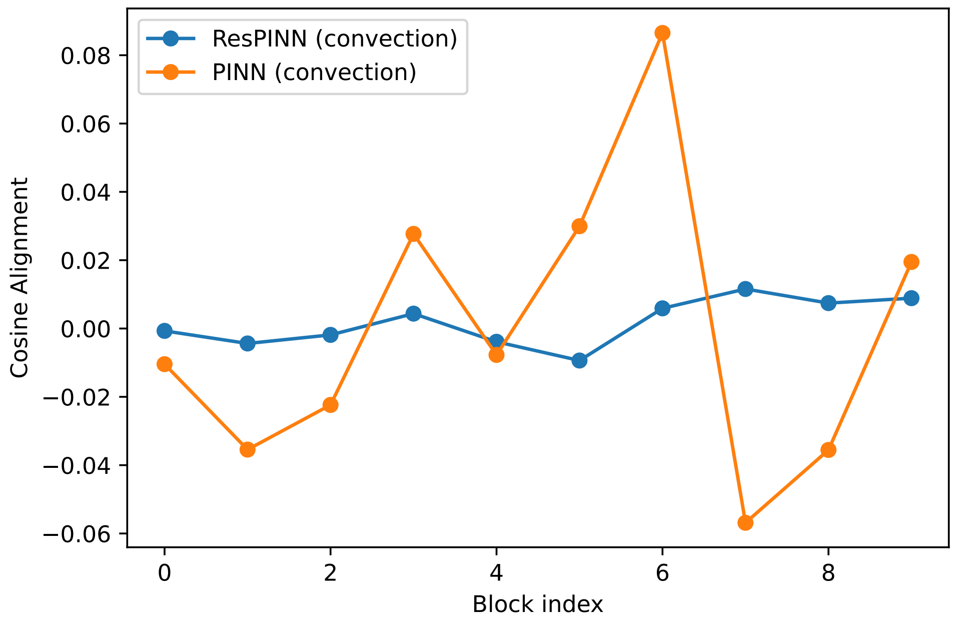

Figure 7: Cosine alignment between block updates and local loss gradients for the convection problem. ResPINNs remain close to zero, indicating residual updates act primarily as stabilizers rather than directly following descent directions. PINNs oscillate between positive and negative values, reflecting inconsistent alignment and unstable propagation.

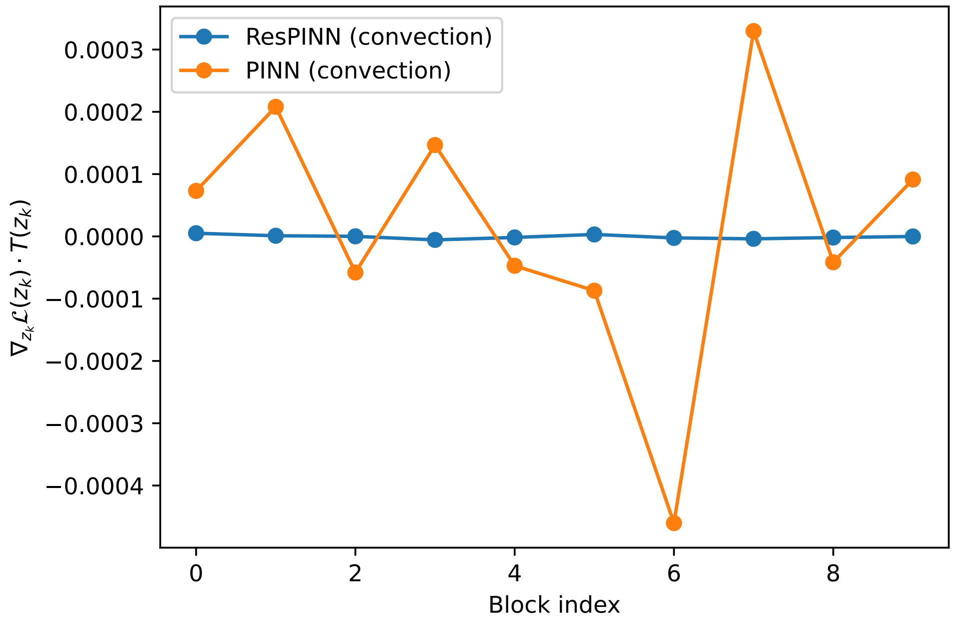

Figure 8: Inner product between block update $T(z_k)$ and the local loss gradient across block depth for the 1D convection problem. ResPINNs maintain values close to zero, consistent with near-isometric transformations. Standard PINNs exhibit larger fluctuations, indicating unstable amplification of activation directions.

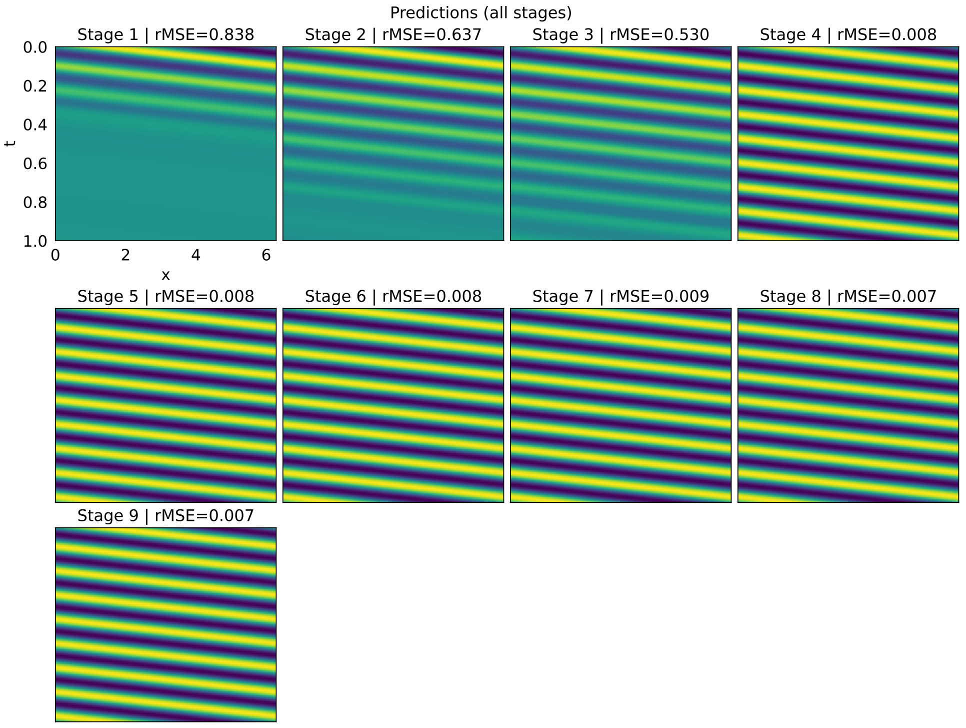

Figure 9: Predicted solutions across blocks. Earlier blocks run into failure modes where they fail to capture temporal dynamics of the convection PDE on $\beta = 50$. With more residual steps, the model captures increasingly fine temporal dynamics.

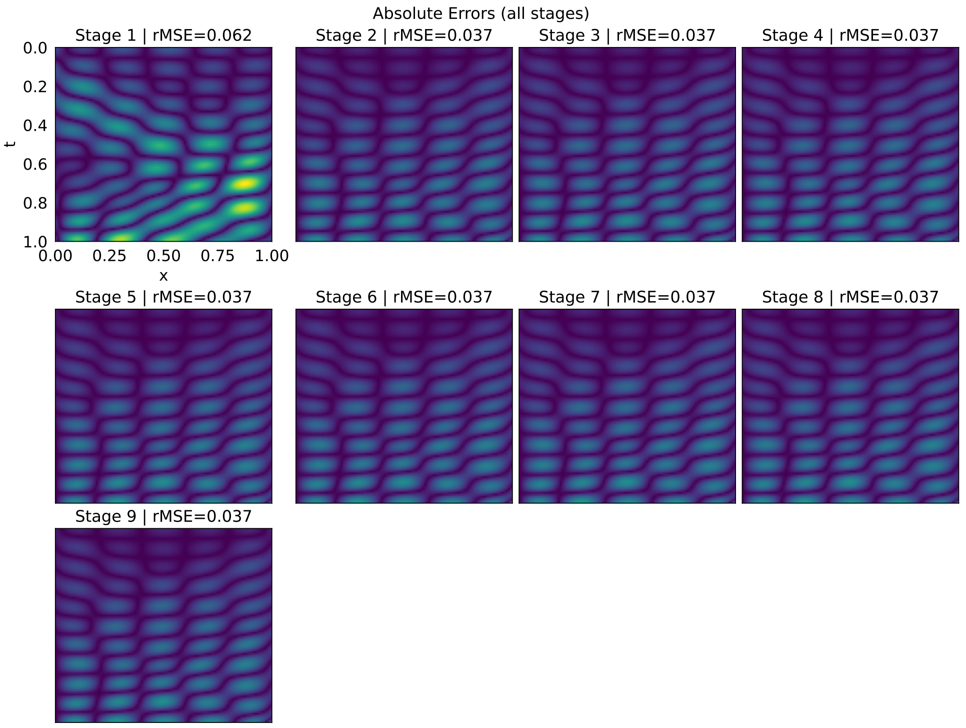

Figure 10: Absolute Errors across blocks on wave PDE.

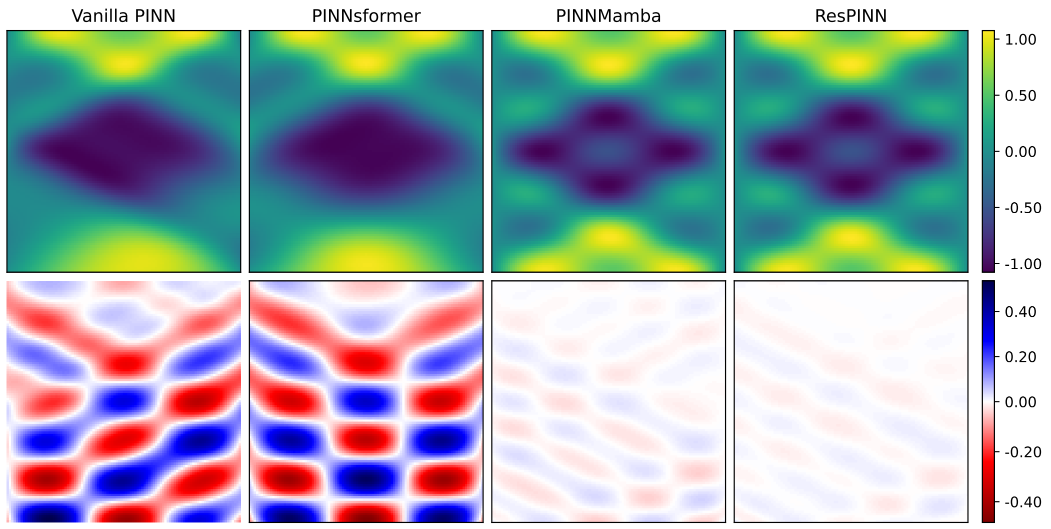

Figure 11: Qualitative comparison on 1D wave PDE. Top: predicted solutions. Bottom: pointwise errors.

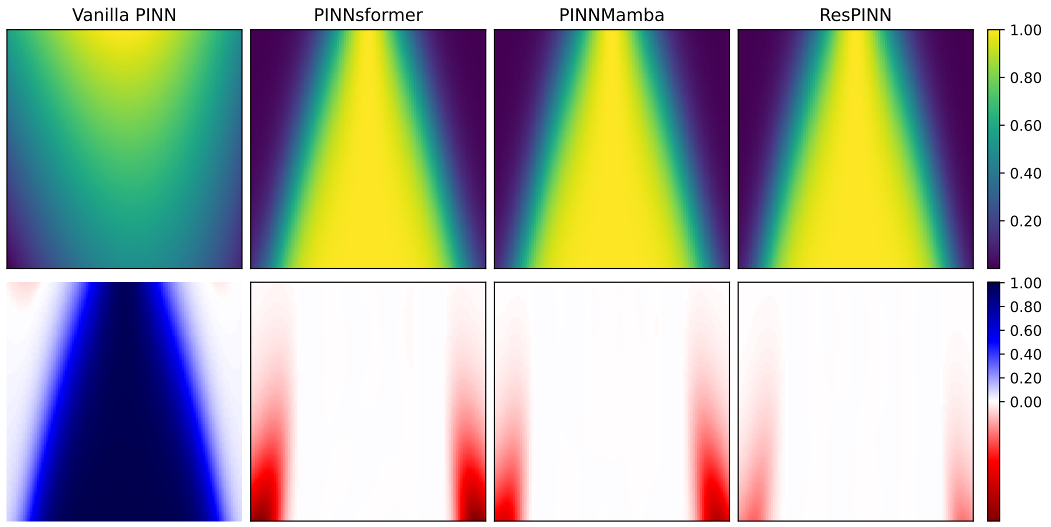

Figure 12: Qualitative comparison on 1D Reaction PDE. Top: predicted solutions. Bottom: pointwise errors.