(ICIC2024, Oral)

S1 The pseudocods of G-RAS

Algorithm 1: The proposed G-RAS.

1 Use Random-R to sample the initial residual points $\mathcal{T}$;
2 Train the PINNs for a finite number of iterations;
3 **repeat**
4 | $\mathcal{T} \longleftarrow$ Sample new residual points according to the PDF of Eq. (6) in main text;
5 | Train the PINNs for a finite number of iterations;
6 **until** *The total error meets the predefined threshold, or the number of iterations meets the maximum limit*;

Algorithm 1 is the pseudocode of G-RAS. First, we use Random-R, a uniform non-adaptive sampling method [4], to pick the initial residual points $\mathcal{T}$ for training (Line 1). Furthermore, we proceed to train the PINNs for a finite number of iterations, as indicated in Line 2. Next, we choose a group of residual points in G-RAS based on the PDF $p(\mathbf{t})$ defined in Eq. (5) in main text on Line 4. The points that are sampled randomly based on the PDF defined in Eq. (5) in main text may be used to substitute the original residual points mentioned in Line 4. We utilize the new residual points $\mathcal{T}$ to train the PINNs for a finite number of iterations (Line 5). Finally, we iterate the procedure until either the cumulative error or the number of iterations surpasses the pre-defined threshold (Lines 3-6).

S2 Experiment Settings

Table 1. Detail settings of G-RAS in solving PDEs.

Equations	Depth	Width	Optimizer	Learning Rate	Iterations
Diffusion	4	32	Adam	0.001	50000
Burgers	4	64	Adam+L-BFGS	0.001	115000
Allen-Cahn	4	64	Adam+L-BFGS	0.001	115000
Wave	6	100	Adam+L-BFGS	0.001	115000
Diffusion-reaction	4	20	Adam	0.001	110000
Korteweg-de Vries	4	100	Adam	0.001	100000

In our experiments, we set $a_{max} = 2$, $a_{min} = 0$, $b_{max} = 10$, and $b_{min} = 0$ in Eq. (5) in the main paper for all PDEs. In addition, the details of the experiments are shown in Table 1. Following the convention [4], the network architecture in our study consists of multi-layer perceptrons, including 4 to 6 layers with a width ranging from 20 to 100. The optimizers we use are either Adam [1] or a

hybrid of Adam and L-BFGS [3]. The learning rate is 0.001, and the iterations range from 50000 to 115000. Furthermore, we employ the hyperbolic tangent (tanh) as the activation function. The codes we use are implemented with the DeepXDE library [2], and the results shown are the mean values obtained from five independent runs. We employ the L_2 and L_1 relative error metrics to assess the predictive accuracy of the solution $\hat{u}$ and the estimated coefficients $\hat{\delta}$ [2, 4].

References

1. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
2. Lu, L., Meng, X., Mao, Z., Karniadakis, G.E.: Deepxde: A deep learning library for solving differential equations. SIAM Review **63**(1), 208–228 (2021)
3. Moritz, P., Nishihara, R., Jordan, M.: A linearly-convergent stochastic l-bfgs algorithm. In: Artificial Intelligence and Statistics. pp. 249–258. PMLR (2016)
4. Wu, C., Zhu, M., Tan, Q., Kartha, Y., Lu, L.: 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)