BEYOND PREDICTION: EXTRACTING PRECONDITIONING STRUCTURE FROM FOURIER NEURAL OPERATORS

Anonymous authors
Paper under double-blind review

ABSTRACT

Neural operators provide rapid predictions, while classical iterative solvers deliver high-accuracy solutions, motivating growing interest in combining their strengths. Interpreting neural operators as learned approximations to inverse PDE operators naturally suggests Jacobian-based preconditioning. However, solution supervision alone does not ensure effective preconditioners, leading existing approaches to modify training objectives or impose additional constraints. Our theoretical analysis shows that Fourier neural operators (FNOs) trained for solution prediction can still contain recoverable preconditioning structure. Building on this insight, we introduce purification, a post-processing technique that extracts this structure from FNO Jacobians without retraining. Experiments on Poisson problems support our theoretical analysis. For Helmholtz problems, NOP reduces mean GMRES iterations by up to $8.2\times$ compared with unpreconditioned GMRES. For nonlinear convection–diffusion problems, NOP reduces mean cumulative GMRES iterations by up to $9.6\times$ compared with unpreconditioned Newton–GMRES. These results offer a new perspective on how neural operators can enhance iterative solvers through structure extraction.

1 INTRODUCTION

Partial differential equations (PDEs) arise in a wide range of scientific and engineering applications, including materials science (Bourdin et al., 2000), automotive aerodynamics (Han, 1989), and aircraft aerodynamics (Jameson et al., 1986). Neural operators have emerged as learned surrogates for PDE solution, approximating mappings from input functions to solution functions. Representative architectures include the deep operator network (DeepONet) (Lu et al., 2021) and the Fourier neural operator (FNO) (Li et al., 2021), which can provide rapid approximate solutions after training. These predictions offer inexpensive approximations that can be used directly when moderate accuracy is sufficient. However, more stringent accuracy requirements may call for further numerical refinement. This raises a practical question: beyond providing rapid predictions, how can trained neural operators accelerate classical iterative solvers?

A natural insight is that coarse predictions provided by neural operators can serve as initial points for iterative solvers. This strategy has been used to accelerate Newton and related iterative methods for nonlinear PDEs (Huang et al., 2020; Aghili et al., 2025; Jin et al., 2025). We also adopt the neural operator's prediction as an initial state. Moreover, in this paper, we go a step further because effective preconditioning is a crucial ingredient for achieving good performance in iterative solvers (Pernice & Walker, 1998).

Neural operators are commonly interpreted as learned approximations to inverse PDE operators. Consider steady-state boundary value problems with varying source s and fixed coefficients and boundary data:

$$\mathcal{N}(u) = s, \qquad \mathcal{N} : \mathcal{U} \to \mathcal{S}, \tag{1}$$

where $u \in \mathcal{U}$ and $s \in \mathcal{S}$ denote the solution and source functions, respectively, with $\mathcal{U}$ and $\mathcal{S}$ defined on a bounded domain $D \subset \mathbb{R}^d$ and the boundary conditions understood. Assuming that $\mathcal{N}$ and its local inverse $\mathcal{G} = \mathcal{N}^{-1}$ are differentiable, applying the chain rule to $\mathcal{G}(\mathcal{N}(u)) = u$ gives

$$\mathrm{D}\mathcal{G}(s)\,\mathrm{D}\mathcal{N}(u) = \mathcal{I}, \qquad s = \mathcal{N}(u), \tag{2}$$

where D denotes the Fréchet derivative. The neural operator $\mathcal{G}_\theta : \mathcal{S} \to \mathcal{U}$, with trainable parameters θ, approximates the source-to-solution map $\mathcal{G}$, suggesting its input derivative $\mathrm{D}\mathcal{G}_\theta(s)$ as

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

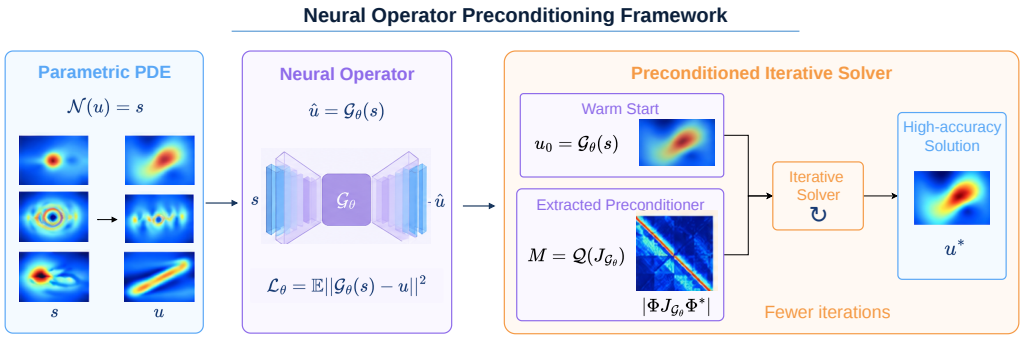


Figure 1: Neural Operator Preconditioning framework. A prediction-trained neural operator provides a warm start and a purified Jacobian preconditioner to accelerate iterative solution without retraining.

a potential preconditioner for $\mathrm{D}\mathcal{N}(u)$. However, solution prediction alone does not ensure accurate input derivatives (O'Leary-Roseberry et al., 2024). Our experiments also show that directly using the learned Jacobian as a preconditioner fails to yield convergence across all tested problems (Section 3). To improve preconditioning, FCG-NO uses a convergence-motivated loss and Krylov-subspace residual sampling (Rudikov et al., 2024), while Neural Preconditioned Born Series (NPBS) aligns its training loss with the preconditioned residual metric (Wang et al., 2026). These methods incorporate solver requirements into training. We instead ask: *Can a neural operator trained only for solution prediction still provide useful preconditioning information, even when its full learned Jacobian is unreliable as a preconditioner?*

Accurately learning the full inverse operator is not a prerequisite for effective preconditioning. The key is to understand the network's implicit learning biases and how they shape the preconditioning information encoded in its Jacobian. To this end, we first analyze FNOs in a simplified setting without nonlinear activations, where the FNO Jacobian is diagonal in the Fourier basis. Using this structure, we derive condition-number bounds for Fourier-diagonal systems and corresponding iteration convergence bounds. We examine these bounds numerically on periodic Poisson problems (Section 3.2). Introducing nonlinear activations, however, couples Fourier modes and can disrupt this diagonal structure. Inspired by this analysis, we introduce *purification*, which extracts Fourier-diagonal magnitudes from the source Jacobian of an FNO in general settings. As illustrated in Figure 1, we use the neural operator's prediction as an initial guess and the preconditioner extracted by purification to accelerate the subsequent iterative solve. Experiments on Helmholtz and nonlinear convection–diffusion equations show that purification yields effective preconditioners even when directly using the learned Jacobian as a preconditioner fails to achieve convergence (Section 3).

1.1 Contributions

- We show that FNOs trained only for solution prediction can retain useful preconditioning information even when their full Jacobians are unsuitable as preconditioners. In a simplified setting without nonlinear activations, we identify the Fourier-diagonal Jacobian structure and establish sufficient conditions under which prediction accuracy guarantees a reduction in the condition number.
- We introduce *purification*, which extracts Fourier-diagonal magnitudes from the learned source Jacobian without retraining or additional loss constraints. Within our Neural Operator Preconditioning (NOP) framework, purification complements neural initialization to accelerate iterative solvers. For SPD, Fourier-diagonal systems, we prove worst-case one-step GMRES optimality among preconditioners sharing the same Fourier-diagonal magnitudes.
- We demonstrate that purification recovers effective preconditioners for Helmholtz and nonlinear convection–diffusion equations where direct use of the learned Jacobian fails to achieve convergence.

1.2 Related Work

Hybrid iterative methods. Neural networks tend to learn low-frequency components first (Xu et al., 2020; Rahaman et al., 2019), whereas classical relaxation preferentially damps high-frequency errors in elliptic problems. This complementarity motivates hybrid methods that alternate neural corrections with classical relaxation or Krylov iterations (Zhang et al., 2024; Kopaničáková & Karniadakis, 2025; Hu & Jin, 2026). More recent methods explicitly exploit spectral or multilevel structure: FNS trains neural components to target error modes poorly reduced by smoothing (Cui et al., 2025a), while neural-enhanced multigrid methods combine learned components with specialized multilevel cycles for high-frequency heterogeneous Helmholtz problems (Cui et al., 2025b; Lerer et al., 2024).

Deep learning-based preconditioner. Classical preconditioners use physics-based or algebraic approximations of the system matrix. Data-driven alternatives learn preconditioning matrices, sparse factorizations, or approximate inverse actions (Sappl et al., 2019; Li et al., 2023; Häusner et al., 2024; Trifonov et al., 2026; Yang et al., 2025). Graph Neural Preconditioners target general sparse systems (Chen, 2025), while Graph Neural Multilevel Preconditioners impose an algebraic multigrid hierarchy to capture both local and global error modes (Zhang et al., 2026). Unlike these methods, which train networks specifically for preconditioning, we extract preconditioning structure post hoc from a neural operator trained only for solution prediction.

2 Method

To analyze the learned operator, we work in a finite-dimensional setting. Discretizing Equation 1 yields the discrete source-to-solution problem

$$\mathcal{N}_h(u_h) = s_h, \qquad u_h, s_h \in \mathbb{R}^n, \tag{3}$$

where h is the discretization scale and n is the number of unknowns. Coefficients and boundary data are fixed and incorporated into $\mathcal{N}_h$. We first analyze the FNO's structure and relate its training error to preconditioning quality, then use these insights to motivate purification.

2.1 Fourier Neural Operator

For the PDE in Equation 1, an FNO $\mathcal{G}_\theta : \mathcal{S} \to \mathcal{U}$ learns the source-to-solution map $\mathcal{G} = \mathcal{N}^{-1}$ by minimizing a prediction loss ℓ over N source–solution pairs:

$$\min_\theta \frac{1}{N} \sum_{m=1}^{N} \ell\Big(\mathcal{G}_\theta(s^{(m)}), u^{(m)}\Big), \qquad \mathcal{N}(u^{(m)}) = s^{(m)}. \tag{4}$$

As shown in Figure 2, the core FNO architecture with L Fourier layers is expressed as

$$\begin{aligned} \mathcal{G}_\theta(s) &= \big(\mathcal{P} \circ \mathcal{H}_{L-1} \circ \cdots \circ \mathcal{H}_0 \circ \mathcal{L}\big)(s), \\ \mathcal{H}_t(v) &= \sigma_t\big(\mathcal{W}_t(v) + \mathcal{F}^{-1}(\mathcal{R}_t \mathcal{F}(v))\big), \qquad t = 0, \ldots, L-1. \end{aligned} \tag{5}$$

Here v denotes the hidden representation.

On the discrete grid, the FNO maps the source vector s_h to an approximation of u_h. For simplicity, we omit bias terms in the following analysis. Its derivative with respect to the source is

$$J_{\mathcal{G}_\theta}(s_h) := \frac{\partial \mathcal{G}_\theta(s_h)}{\partial s_h} \in \mathbb{R}^{n \times n}. \tag{6}$$

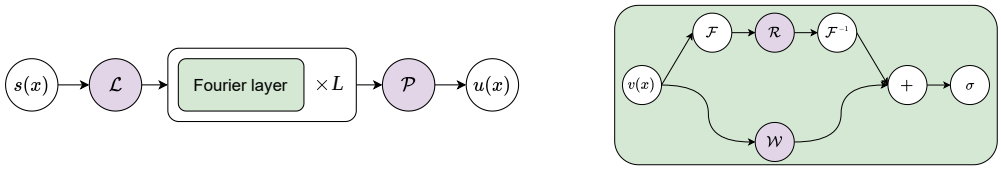
To analyze the structure of this Jacobian, we first consider an FNO without nonlinear activations. In this setting, $\mathcal{G}_\theta$ is a linear operator, so its Jacobian is independent of the input. Moreover, we show that this matrix is diagonalizable in the Fourier basis, as established below. Let $\Phi \in \mathbb{C}^{n \times n}$ denote the unitary spatial DFT.

Lemma 2.1. *Consider the FNO in Equation 5 without activations or padding. Its source Jacobian satisfies*

$$J_{\mathcal{G}_\theta}(s_h) = \Phi^* \operatorname{diag}\big(\psi_\xi(\theta)\big)_{\xi=1}^{n} \Phi. \tag{7}$$

Thus its eigenvalues are the Fourier multipliers $\psi_\xi(\theta)$, and its singular values are $|\psi_\xi(\theta)|$.

Proof. See Appendix A.1.



(a) FNO architecture (b) Fourier layer

Figure 2: FNO architecture. (a) $\mathcal{L}$ is a pointwise lifting transformation from the source to latent channels, and $\mathcal{P}$ projects the final features to the solution. (b) $\mathcal{F}$ and $\mathcal{F}^{-1}$ are the Fourier transform and its inverse; $\mathcal{R}$ applies learned channel-mixing matrices independently at retained Fourier modes and is zero on the remaining modes. $\mathcal{W}$ is a pointwise affine transformation, and σ is a pointwise activation, omitted in the last layer.

2.2 From Prediction Accuracy to Preconditioning

For linear PDEs whose discretization yields $A_h u_h = s_h$. When A_h is diagonalizable in the Fourier basis, Lemma 2.1 ensures that A_h and the FNO Jacobian are simultaneously diagonalizable. This allows us to relate prediction accuracy to the condition number of the preconditioned system.

Theorem 2.2. *Let $\mathcal{G}_\theta$ be an FNO without activations as in Lemma 2.1, trained on pairs $A_h u_h^{(m)} = s_h^{(m)}$, where $A_h = \Phi^* \operatorname{diag}(a_\xi)\Phi$ and $a_\xi \neq 0$. Let $S_N(\xi) = \frac{1}{N}\sum_m |[\Phi s_h^{(m)}]_\xi|^2 > 0$. If the training loss satisfies*

$$\mathcal{L}_\theta = \frac{1}{N}\sum_{m=1}^{N} \|\mathcal{G}_\theta(s_h^{(m)}) - u_h^{(m)}\|_2^2 \leq \varepsilon, \tag{8}$$

where

$$\varepsilon < \left(\frac{\max_\xi |a_\xi| - \min_\xi |a_\xi|}{\max_\xi |a_\xi| + \min_\xi |a_\xi|}\right)^2 \min_\xi \frac{S_N(\xi)}{|a_\xi|^2}, \tag{9}$$

then, with $\rho = \max_\xi |a_\xi| \sqrt{\varepsilon / S_N(\xi)}$,

$$\kappa_2(J_{\mathcal{G}_\theta} A_h) \leq \frac{1+\rho}{1-\rho} < \frac{\max_\xi |a_\xi|}{\min_\xi |a_\xi|} = \kappa_2(A_h). \tag{10}$$

Proof. See Appendix A.2.

When the source is sampled from a Gaussian random field (Lord et al., 2014), its spectrum gives an explicit frequency-dependent error estimate.

Corollary 2.3. *Consider the FNO and Fourier-diagonal system of Theorem 2.2, with $a_\xi > 0$ and N independent sources generated from a periodic Gaussian random field $\mathrm{GRF}(0, \sigma_s^2(-\Delta + \tau^2 I)^{-\gamma})$ on $[0,1]$, where GRF denotes a Gaussian random-field distribution parameterized by its mean and covariance operator. For any $0 < \delta < 1$, on an event of probability at least $1 - n\exp(-N\delta^2/4)$, the training-loss bound $\mathcal{L}_\theta \leq \varepsilon$ implies*

$$\frac{|\psi_\xi - a_\xi^{-1}|}{a_\xi^{-1}} \leq \frac{\sqrt{\varepsilon}}{\sigma_s\sqrt{1-\delta}} \frac{\left((2\pi\xi)^2 + \tau^2\right)^{\gamma/2}}{a_\xi^{-1}}. \tag{11}$$

Proof. See Appendix A.3.

2.3 Extracting Preconditioning Structure via Purification

The preceding analysis relies on an activation-free, unpadded architecture. In general FNO architectures, the derivatives of nonlinear activations vary across spatial locations and channels and can therefore couple Fourier modes. Padding and normalization can also disrupt the Fourier-diagonal structure. Fortunately, inspired by the preceding analysis, we propose *purification* to extract useful preconditioning structure from the learned Jacobian:

$$\begin{aligned} J_{\mathrm{spectral}}(s_h) &= \Phi J_{\mathcal{G}_\theta}(s_h)\Phi^*, \\ J_{\mathrm{purify}}(s_h) &= \Phi^* \operatorname{diag}\big(|J_{\mathrm{spectral}}(s_h)|\big)\Phi. \end{aligned} \tag{12}$$

Here $|\cdot|$ acts entrywise, and diag retains the diagonal and sets all off-diagonal entries to zero. Let $\boldsymbol{e}_\xi$ be the ξ-th standard basis vector. Each retained multiplier is extracted using a Jacobian–vector product (JVP), without forming the dense Jacobian:

$$d_\xi = \left[\Phi J_{\mathcal{G}_\theta}(s_h)(\Phi^* \boldsymbol{e}_\xi)\right]_\xi. \tag{13}$$

Writing $\boldsymbol{d} = (d_\xi)_{\xi=1}^n$ and $M = J_{\mathrm{purify}}(s_h)$, its action on $w \in \mathbb{R}^n$ is

$$Mw = \Phi^*\big(|\boldsymbol{d}| \odot \Phi w\big), \tag{14}$$

where $\odot$ denotes entrywise multiplication. Exact extraction requires $O(n)$ directional derivative evaluations. Once the multipliers are stored, each application costs $O(n \log n)$ using FFTs, with $O(n)$ storage.

The predicted solution is expected to be close to the high-accuracy solution. Motivated by the reuse principle underlying chord methods (Kelley, 2018), we construct M once at the supplied source and keep it fixed throughout the iterative solve. This avoids requiring the network to generalize to additional source inputs associated with intermediate iterates and eliminates repeated extraction costs. Appendix B gives the numerical algorithms and complete procedure.

For general FNO architectures, we now analyze purification at a fixed source. We assess purification using the worst-case residual reduction after one step of the generalized minimal residual (GMRES) method (Saad & Schultz, 1986). For $B \in \mathbb{R}^{n\times n}$, define the worst-case one-step GMRES residual factor by

$$\Gamma(B) := \max_{\substack{r\in\mathbb{R}^n \\ \|r\|_2=1}} \min_{c\in\mathbb{R}} \|r - cBr\|_2. \tag{15}$$

Here r is the normalized initial residual of the system with matrix B. For left-preconditioned systems, this is the preconditioned residual. We have the following result:

Theorem 2.4. *Let $A_h = \Phi^* \operatorname{diag}(a_\xi)\Phi$ be real SPD, with $a_\xi > 0$, and let $J \in \mathbb{R}^{n\times n}$. Define*

$$d_\xi = (\Phi J \Phi^*)_{\xi\xi} \neq 0, \qquad M = \Phi^* \operatorname{diag}(|d_\xi|)\Phi.$$

Among all real matrices P satisfying $|(\Phi P \Phi^*)_{\xi\xi}| = |d_\xi|$ for every ξ, M minimizes $\Gamma(PA_h)$. With $g_\xi = a_\xi|d_\xi|$, $g_{\min} = \min_\xi g_\xi$, and $g_{\max} = \max_\xi g_\xi$, in particular,

$$\Gamma(MA_h) = \frac{g_{\max} - g_{\min}}{g_{\max} + g_{\min}} \leq \Gamma(JA_h). \tag{16}$$

Proof. See Appendix A.4.

For SPD, Fourier-diagonal systems, Theorem 2.4 justifies purification of general FNO Jacobians: it minimizes the worst-case one-step GMRES residual factor for fixed diagonal magnitudes. In the setting of Theorem 2.2, purification preserves the reduced condition number while making the preconditioned system SPD. The condition-number reduction therefore translates into a tighter classical GMRES convergence bound (Saad, 2003).

3 Experiments

We test whether prediction-trained FNOs contain useful preconditioning structure, using Poisson1D to examine the theory, Helmholtz2D to test indefinite systems, nonlinear Darcy2D to test heterogeneous nonlinear diffusion, and Convec2D to study nonlinear transport.

3.1 Experimental setup

Models and comparisons. We compare no preconditioning (None), the direct FNO source Jacobian (Direct), its purified counterpart (Purify), and the fast Poisson solver (FPS), which serves as a classical preconditioning baseline for elliptic PDEs (Concus & Golub, 1973). FNOs are trained only on source–solution pairs. For each PDE parameter setting, neural initialization, Direct, and Purify use the same trained FNO. We compare zero and neural initial states on the same 50 test sources per setting in Cases II–IV. Appendix C.3 provides data-generation and training details, along with prediction results.

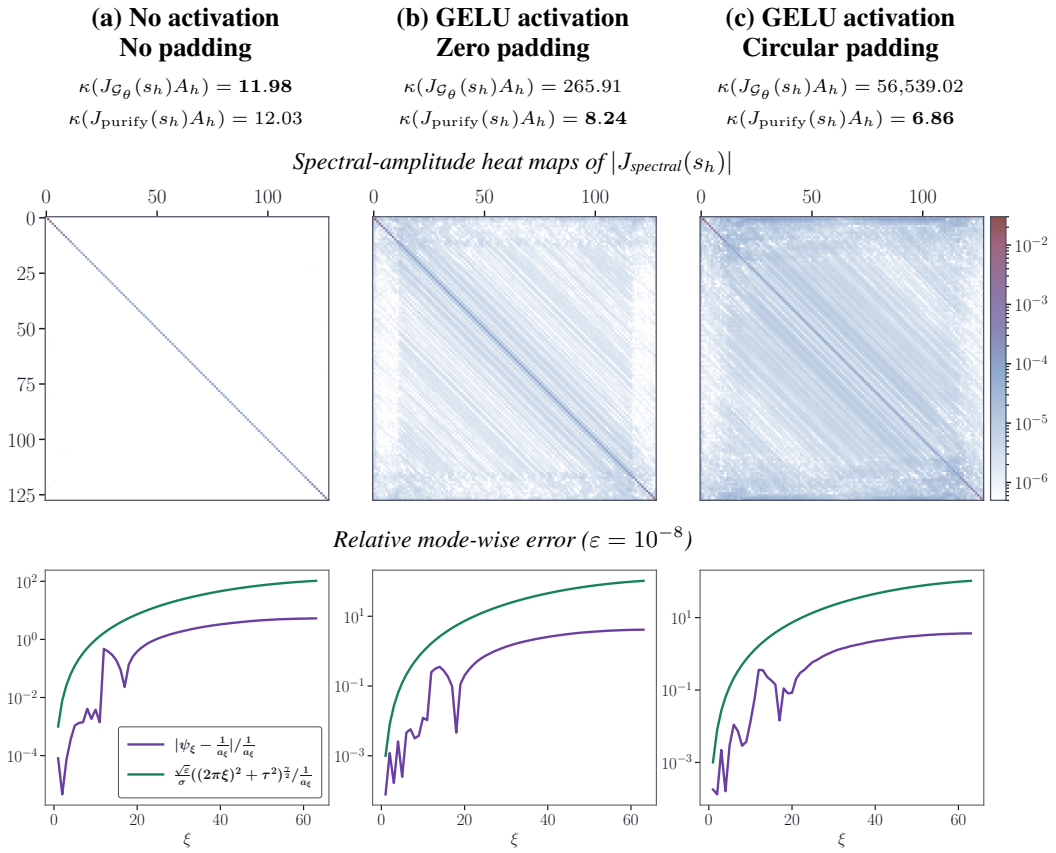


Figure 3: Poisson1D spectral structure at a fixed test source. Top: Fourier-space Jacobian amplitudes; bottom: relative diagonal-multiplier errors. The activation-free, unpadded model in (a) tests the diagonal theory; (b,c) show mode coupling and improved conditioning after purification.

Solvers and metrics. We use left-preconditioned GMRES for linear problems and Newton–GMRES for nonlinear problems; algorithms and solver settings are given in Appendix B. NI counts Newton updates, and LI counts GMRES iterations, accumulated across Newton steps for nonlinear problems. Success counts include all test sources; iteration means include successful solves only. Timing results are given in Appendix C.5.

Well-posedness. For the periodic Poisson operator, we use a rank-one correction to remove its constant nullspace and obtain a nonsingular discrete system, ensuring well-posedness. Details are given in Appendix C.2.

3.2 Poisson1D: verifying the theoretical analysis

We consider the one-dimensional Poisson equation with periodic boundary conditions:

$$\begin{cases} -u''(x) = s(x), & x \in [0,1], \\ u(0) = u(1), & u'(0) = u'(1). \end{cases} \tag{17}$$

At resolution $n = 128$, the corrected discrete Poisson matrix has condition number $\kappa(A_h) = 1660.35$. Appendix C.3.1 provides data-generation and training details, along with prediction results. Figure 3(a) shows that the activation-free, unpadded FNO has a Fourier-diagonal source Jacobian, as predicted by Lemma 2.1. Adding GELU and zero or circular padding introduces off-diagonal coupling in (b,c). Purification extracts this diagonal structure and reduces the preconditioned condition number. The lower row plots the relative spectral error $|\psi_\xi(\theta) - a_\xi^{-1}|/a_\xi^{-1}$. In (a–c), the measured errors lie below the error envelopes based on Corollary 2.3.

Our purification method inherits the resolution-transfer capability of FNO, which we empirically validate in Appendix C.4.

3.3 HELMHOLTZ2D: INDEFINITE SYSTEMS

We solve the variable-wavenumber Helmholtz problem with homogeneous Dirichlet boundary conditions:

$$\begin{cases} -\Delta u(x,y) - k(x,y)^2 u(x,y) = s(x,y), & (x,y) \in \Omega = [0,1]^2, \\ u(x,y) = 0, & (x,y) \in \partial\Omega, \end{cases} \tag{18}$$

where

$$k(x,y) = k_0\big[1 + 0.2\sin(2\pi x)\sin(2\pi y)\big]. \tag{19}$$

We use a 64×64 grid with $k_0 \in \{8, 12, 16\}$. Appendix C.3.2 provides data-generation and training details, along with prediction results. Table 1 shows that purification recovers useful preconditioning where Direct fails, although FPS remains more effective.

Table 1: Helmholtz2D success count / mean LI (out of 50 sources). Each wavenumber has zero and neural initialization columns. Means use successful solves only; − denotes no success. Bold marks the lowest mean LI per column.

	$k_0 = 8$		$k_0 = 12$		$k_0 = 16$	
Precond.	Zero Init.	Neural Init.	Zero Init.	Neural Init.	Zero Init.	Neural Init.
None	50 / 333.44	50 / 317.30	50 / 420.68	50 / 401.86	0 / −	6 / 1670.83
FPS	50 / **20.00**	50 / **19.00**	50 / **33.80**	50 / **32.08**	50 / **56.54**	50 / **56.40**
Direct	0 / −	0 / −	0 / −	0 / −	0 / −	0 / −
Purify	50 / 49.40	50 / 50.66	50 / 51.20	50 / 50.20	48 / 293.75	48 / 255.40

To assess our theoretical analysis empirically, we measure the diagonal energy fraction of $J_{\text{spectral}}(s_h)$ and the one-step GMRES residual factor, defined as

$$\mathcal{E}_{\text{diag}}(s_h) = \frac{\|\operatorname{diag}(J_{\text{spectral}}(s_h))\|_F^2}{\|J_{\text{spectral}}(s_h)\|_F^2}, \tag{20}$$

and

$$\rho_1(B, r) = \min_{c \in \mathbb{R}} \frac{\|r - cBr\|_2}{\|r\|_2}. \tag{21}$$

We evaluate the residual factor for $B = J_{\mathcal{G}_\theta}(s_h)A_h$ and $B = J_{\text{purify}}(s_h)A_h$, using the same direction $r = s_h/\|s_h\|_2$. As shown in Table 2, the diagonal entries constitute only $1/4096 \approx 0.024\%$ of all matrix entries, yet account for nearly 50% of the energy on average across the three wavenumbers (ranging from 31.54% to 60.40%). Purification also reduces the mean one-step residual factor at each wavenumber. Although $\rho_1(B, r)$ for a fixed direction does not guarantee GMRES convergence, these results support the effectiveness of purification and are consistent with the mechanism underlying Theorem 2.4.

Table 2: Helmholtz2D mean diagonal energy fraction and one-step GMRES residual factors. Smaller residual factors indicate stronger reduction.

k_0	$\mathcal{E}_{\text{diag}}$ (%)	$\rho_1(J_{\mathcal{G}_\theta}(s_h)A_h, r)$	$\rho_1(J_{\text{purify}}(s_h)A_h, r)$
8	60.40	0.964	0.415
12	48.93	0.942	0.516
16	31.54	0.984	0.968

3.4 DARCY2D: HETEROGENEOUS NONLINEAR DIFFUSION

We consider the nonlinear Darcy equation on $\Omega = [0,1]^2$,

$$\begin{cases} -\nabla\cdot\big(a(x,y)(1+u^2)\nabla u\big) = s(x,y), & (x,y) \in \Omega, \\ u(x,y) = 0, & (x,y) \in \partial\Omega, \end{cases} \tag{22}$$

where $a(x, y) = \exp(g(x, y))$ and g is a periodic Gaussian random field with zero spatial mean,

$$g \sim \mathrm{GRF}\big(0, \sigma_g^2(-\Delta + I)^{-2}\big), \qquad \sigma_g \in \{10, 20, 40\}. \tag{23}$$

We use a 64×64 cell-centered grid. Within each parameter setting, the coefficient field $a(x, y)$ is fixed across source–solution pairs. Appendix C.3.3 provides data-generation and training details, along with prediction results.

Table 3: Darcy2D success counts and mean NI / cumulative LI on a 64×64 grid. Means include successful solves only; bold marks the lowest LI per column. Direct fails on all 50 sources in every setting and initialization.

		Zero initialization			Neural initialization		
σ_g	Precond.	Success	NI	LI	Success	NI	LI
10	None	50/50	6.58	938.64	50/50	4.78	793.46
	FPS	50/50	6.14	**185.50**	50/50	4.04	**134.96**
	Purify	50/50	6.20	193.48	50/50	4.02	147.20
20	None	50/50	6.78	1358.04	50/50	4.98	1131.38
	FPS	50/50	6.60	**262.42**	50/50	4.34	**187.88**
	Purify	50/50	6.58	337.56	50/50	4.30	274.26
40	None	50/50	8.28	4824.44	50/50	5.34	4081.96
	FPS	50/50	7.96	**1100.50**	50/50	5.26	**940.48**
	Purify	50/50	7.80	1675.42	50/50	5.24	1497.12

Purification reduces mean cumulative LI relative to no preconditioning across all tested settings, whereas the direct Jacobian fails throughout (Table 3). At low coefficient heterogeneity, Purify achieves iteration counts comparable to FPS. As heterogeneity increases, Purify remains effective but offers smaller relative iteration savings.

3.5 CONVEC2D: NONLINEAR TRANSPORT AND INITIALIZATION

We consider the nonlinear convection–diffusion equation on $[0, 1]^2$ with periodic boundary conditions.

$$-\Delta u + Re\, u(u_x + u_y) = s(x, y), \qquad (x, y) \in [0, 1]^2. \tag{24}$$

We train and evaluate a separate FNO at each $Re \in \{100, 250, 500\}$. Appendix C.3.4 provides data-generation and training details, along with prediction results.

Table 4: Convec2D success counts and mean NI / cumulative LI. Means include successful solves only; bold marks the lowest LI per column. Direct fails on all 50 sources in every setting and initialization.

		Zero initialization			Neural initialization		
Re	Precond.	Success	NI	LI	Success	NI	LI
100	None	39/50	12.79	909.03	50/50	5.46	709.20
	FPS	45/50	9.24	227.51	50/50	4.04	**111.80**
	Direct	0/50	–	–	0/50	–	–
	Purify	46/50	9.22	**213.74**	50/50	4.18	135.68
250	None	4/50	12.00	1903.25	50/50	6.78	1503.64
	FPS	6/50	9.67	360.33	49/50	4.78	245.96
	Direct	0/50	–	–	0/50	–	–
	Purify	6/50	9.33	**248.17**	50/50	5.00	**210.78**
500	None	0/50	–	–	34/50	6.41	2537.94
	FPS	0/50	–	–	47/50	4.60	435.77
	Direct	0/50	–	–	0/50	–	–
	Purify	0/50	–	–	47/50	4.62	**263.23**

Neural initialization and purification complement each other as transport strengthens (Table 4). With neural initialization, Purify becomes increasingly competitive with FPS, achieving lower mean cumulative LI among successful solves, while Direct continues to fail. Sample residual histories appear

in Figure 4. Figure 5 shows the predicted and iterative solutions and their pointwise absolute difference on a 128×128 grid, where the solver is accelerated using the model trained on a 64×64 grid.

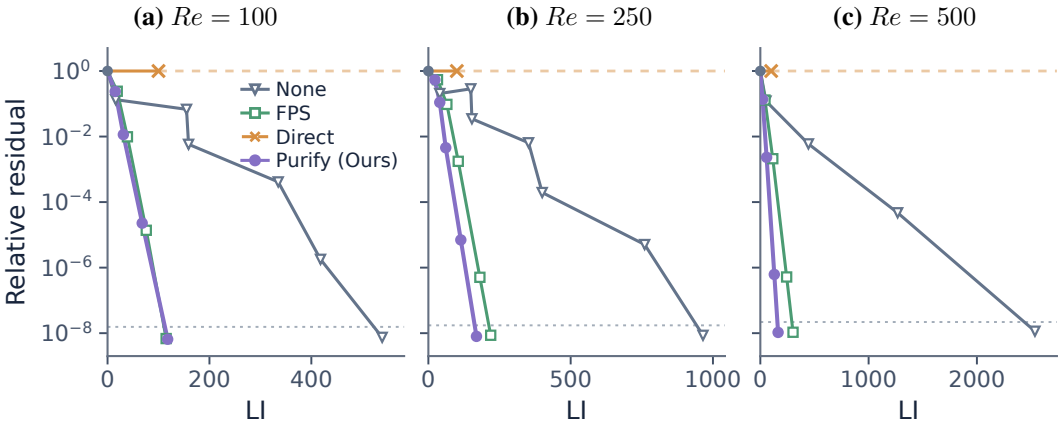


Figure 4: Convec2D nonlinear residual convergence histories of Newton–GMRES for test sample 916. The relative nonlinear residual $\|F_h(u_h^{(p)}; s_h)\|_2 / \|F_h(u_h^{(0)}; s_h)\|_2$ is plotted against cumulative inner GMRES iterations (LI).

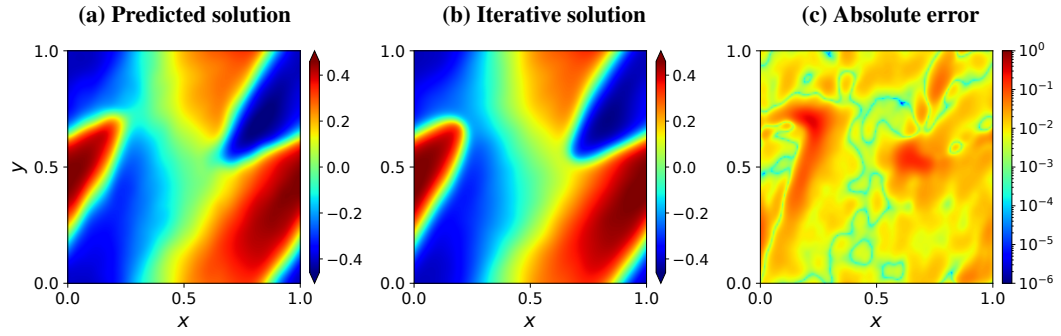


Figure 5: Convec2D solution comparison at $Re = 250$ for test sample 916 : FNO prediction (left), preconditioned iterative solution (middle), and their pointwise absolute difference (right).

4 Conclusion and Discussion

Our experiments support both the theoretical analysis and the effectiveness of the NOP framework. Poisson1D results support the predictions of Theorem 2.2 and Corollary 2.3, while Helmholtz2D provides empirical evidence consistent with the mechanism underlying Theorem 2.4. Across Helmholtz2D, Darcy2D, and Convec2D, direct Jacobian preconditioning fails in all tested settings, whereas purification reduces mean GMRES iterations efficiently. As nonlinear transport strengthens in Convec2D, purification combined with neural initialization also achieves lower mean cumulative GMRES iteration counts than FPS. Together, these results show that NOP can recover useful preconditioning information from FNOs trained only for solution prediction.

Purification retains useful spectral information while imposing a Fourier-diagonal structure favorable for preconditioning. However, this simplification also sacrifices some information. As nonlinearity and heterogeneity increase, learned FNOs may exhibit more complex coupling between Fourier modes, so discarding these couplings may remove more useful information. This motivates better extraction methods that balance information retention and preconditioning structure. Beyond FNOs, prediction-trained neural operators may also encode useful preconditioning information, motivating us to understand their learning biases before imposing additional training constraints.

AI USE STATEMENT

During the preparation of this manuscript, we used OpenAI Codex to assist with language refinement, LaTeX editing, and drafting the disclosure statements. The authors take full responsibility for the content of this paper, including all AI-assisted text and analysis.

ETHICS STATEMENT

This work develops neural-operator-based preconditioners for solving partial differential equations. Our experiments use numerically generated source–solution pairs and involve neither human participants nor personal data. We do not anticipate ethical concerns specific to the proposed method beyond those associated with scientific computing and learned numerical models. For safety-critical applications, solution accuracy and solver convergence should be independently validated under the intended operating conditions, including settings outside the training distribution.

REPRODUCIBILITY STATEMENT

Section 2 describes the method and states the assumptions of the theoretical results; proofs are provided in Appendix A. Appendix B and Algorithm 1 describe the common iterative solver framework. Section 3.1 specifies the solver protocol and reporting conventions. Appendix C collects benchmark discretizations, data-generation procedures, and training details, while Appendix C.5 provides a timing breakdown. We will make our code publicly available on GitHub upon acceptance of the paper.

REFERENCES

Joubine Aghili, Emmanuel Franck, Romain Hild, Victor Michel-Dansac, and Vincent Vigon. Accelerating the convergence of Newton's method for nonlinear elliptic PDEs using Fourier neural operators. *Communications in Nonlinear Science and Numerical Simulation*, 140:108434, 2025. doi: 10.1016/j.cnsns.2024.108434.

Blaise Bourdin, Gilles A. Francfort, and Jean-Jacques Marigo. Numerical experiments in revisited brittle fracture. *Journal of the Mechanics and Physics of Solids*, 48(4):797–826, 2000. doi: 10.1016/S0022-5096(99)00028-9.

Jie Chen. Graph neural preconditioners for iterative solutions of sparse linear systems. In *International Conference on Learning Representations*, volume 2025, pp. 23942–23967, 2025.

Paul Concus and Gene H. Golub. Use of fast direct methods for the efficient numerical solution of nonseparable elliptic equations. *SIAM Journal on Numerical Analysis*, 10(6):1103–1120, 1973. doi: 10.1137/0710092.

Chen Cui, Kai Jiang, Yun Liu, and Shi Shu. A hybrid iterative neural solver based on spectral analysis for parametric PDEs. *Journal of Computational Physics*, 538:114165, 2025a. doi: 10.1016/j.jcp.2025.114165.

Chen Cui, Kai Jiang, and Shi Shu. A neural multigrid solver for Helmholtz equations with high wavenumber and heterogeneous media. *SIAM Journal on Scientific Computing*, 47(3):C655–C679, 2025b. doi: 10.1137/24M1654397.

Stanley C Eisenstat and Homer F Walker. Choosing the forcing terms in an inexact Newton method. *SIAM Journal on Scientific Computing*, 17(1):16–32, 1996.

Vance Faber, Jörg Liesen, and Petr Tichý. Properties of worst-case GMRES. *SIAM Journal on Matrix Analysis and Applications*, 34(4):1500–1519, 2013. doi: 10.1137/13091066X.

Taeyoung Han. Computational analysis of three-dimensional turbulent flow around a bluff body in ground proximity. *AIAA Journal*, 27(9):1213–1219, 1989. doi: 10.2514/3.10248.

Paul Häusner, Ozan Öktem, and Jens Sjölund. Neural incomplete factorization: Learning preconditioners for the conjugate gradient method. *Transactions on Machine Learning Research*, 2024. URL https://openreview.net/forum?id=FozLrZ3CI5.

Jun Hu and Pengzhan Jin. A hybrid iterative method based on MIONet for PDEs: Theory and numerical examples. *Mathematics of Computation*, 95(359):1327–1359, 2026. doi: 10.1090/mcom/4086.

Jianguo Huang, Haoqin Wang, and Haizhao Yang. Int-Deep: A deep learning initialized iterative method for nonlinear problems. *Journal of Computational Physics*, 419:109675, 2020. doi: 10.1016/j.jcp.2020.109675.

Antony Jameson, Timothy J. Baker, and Nigel P. Weatherill. Calculation of inviscid transonic flow over a complete aircraft. In *24th Aerospace Sciences Meeting*. American Institute of Aeronautics and Astronautics, 1986. doi: 10.2514/6.1986-103.

Tianyu Jin, Georg Maierhofer, Katharina Schratz, and Yang Xiang. A fast neural hybrid Newton solver adapted to implicit methods for nonlinear dynamics. *Journal of Computational Physics*, 529:113869, 2025. doi: 10.1016/j.jcp.2025.113869.

C. T. Kelley. Numerical methods for nonlinear equations. *Acta Numerica*, 27:207–287, 2018. doi: 10.1017/S0962492917000113.

D. A. Knoll and D. E. Keyes. Jacobian-free Newton–Krylov methods: A survey of approaches and applications. *Journal of Computational Physics*, 193(2):357–397, 2004. doi: 10.1016/j.jcp.2003.08.010.

Alena Kopaničáková and George Em Karniadakis. DeepONet based preconditioning strategies for solving parametric linear systems of equations. *SIAM Journal on Scientific Computing*, 47(1): C151–C181, 2025. doi: 10.1137/24M162861X.

Béatrice Laurent and Pascal Massart. Adaptive estimation of a quadratic functional by model selection. *The Annals of Statistics*, 28(5):1302–1338, 2000. doi: 10.1214/aos/1015957395.

Bar Lerer, Ido Ben-Yair, and Eran Treister. Multigrid-augmented deep learning preconditioners for the Helmholtz equation using compact implicit layers. *SIAM Journal on Scientific Computing*, 46 (5):S123–S144, 2024. doi: 10.1137/23M1583302.

Yichen Li, Peter Yichen Chen, Tao Du, and Wojciech Matusik. Learning preconditioners for conjugate gradient PDE solvers. In *International Conference on Machine Learning*, pp. 19425–19439. PMLR, 2023.

Zongyi Li, Nikola Borislavov Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Fourier neural operator for parametric partial differential equations. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=c8P9NQVtmnO.

Gabriel J. Lord, Catherine E. Powell, and Tony Shardlow. *An Introduction to Computational Stochastic PDEs*, volume 50 of *Cambridge Texts in Applied Mathematics*. Cambridge University Press, 2014. doi: 10.1017/CBO9781139017329.

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*, 3:218–229, 2021. doi: 10.1038/s42256-021-00302-5.

Thomas O'Leary-Roseberry, Peng Chen, Umberto Villa, and Omar Ghattas. Derivative-informed neural operator: An efficient framework for high-dimensional parametric derivative learning. *Journal of Computational Physics*, 496:112555, 2024. doi: 10.1016/j.jcp.2023.112555.

Michael Pernice and Homer F. Walker. NITSOL: A Newton iterative solver for nonlinear systems. *SIAM Journal on Scientific Computing*, 19(1):302–318, 1998. doi: 10.1137/S1064827596303843.

Nasim Rahaman, Aristide Baratin, Devansh Arpit, Felix Draxler, Min Lin, Fred Hamprecht, Yoshua Bengio, and Aaron Courville. On the spectral bias of neural networks. In *International Conference on Machine Learning*, pp. 5301–5310. PMLR, 2019.

Alexander Rudikov, Vladimir Fanaskov, Ekaterina Muravleva, Yuri M. Laevsky, and Ivan Oseledets. Neural operators meet conjugate gradients: The FCG-NO method for efficient PDE solving. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 42766–42782. PMLR, 2024. URL https://proceedings.mlr.press/v235/rudikov24a.html.

Youcef Saad and Martin H. Schultz. GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM Journal on Scientific and Statistical Computing*, 7(3):856–869, 1986. doi: 10.1137/0907058.

Yousef Saad. *Iterative Methods for Sparse Linear Systems*. Society for Industrial and Applied Mathematics, Philadelphia, PA, 2 edition, 2003. doi: 10.1137/1.9780898718003.

Johannes Sappl, Laurent Seiler, Matthias Harders, and Wolfgang Rauch. Deep learning of preconditioners for conjugate gradient solvers in urban water related problems, 2019. URL https://arxiv.org/abs/1906.06925.

Vladislav Trifonov, Alexander Rudikov, Oleg Iliev, Yuri M. Laevsky, Ivan Oseledets, and Ekaterina Muravleva. Learning from linear algebra: A graph neural network approach to preconditioner design for conjugate gradient solvers. *Computational Methods in Applied Mathematics*, 26(2): 267–284, 2026. doi: 10.1515/cmam-2025-0170.

Juntao Wang, Xinliang Liu, and Jiwei Jia. Born-series-inspired residual metric for learned preconditioners. In *AI & PDE: ICLR 2026 Workshop on AI and Partial Differential Equations*, 2026. URL https://openreview.net/forum?id=eIXkwPgAc0.

Zhi-Qin John Xu, Yaoyu Zhang, Tao Luo, Yanyang Xiao, and Zheng Ma. Frequency principle: Fourier analysis sheds light on deep neural networks. *Communications in Computational Physics*, 28(5):1746–1767, 2020. doi: 10.4208/cicp.OA-2020-0085.

Zherui Yang, Zhehao Li, Kangbo Lyu, Yixuan Li, Tao Du, and Ligang Liu. Learning sparse approximate inverse preconditioners for conjugate gradient solvers on GPUs. In *Advances in Neural Information Processing Systems*, volume 38, 2025. doi: 10.52202/085713-0852.

Enrui Zhang, Adar Kahana, Alena Kopaničáková, Eli Turkel, Rishikesh Ranade, Jay Pathak, and George Em Karniadakis. Blending neural operators and relaxation methods in PDE numerical solvers. *Nature Machine Intelligence*, 6:1303–1313, 2024. doi: 10.1038/s42256-024-00910-x.

Zechen Zhang, Rui Peng Li, and Yousef Saad. Graph neural multilevel preconditioners for iterative solvers. In *Proceedings of the 32nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2026. doi: 10.1145/3770855.3818863.

A PROOFS AND IMPLEMENTATION DETAILS

Throughout, Φ is the unitary spatial DFT, $*$ denotes conjugate transpose, and $\kappa_2(B) = \|B\|_2\|B^{-1}\|_2$ for nonsingular B. Spatial and frequency indices range over n grid points and modes; in multiple dimensions, they enumerate the corresponding multi-indices.

A.1 PROOF OF LEMMA 2.1

We use tensor indices to separate spatial transformations from channel maps. Let i, j index grid points, ξ index Fourier modes, α index input channels, and β, β' index hidden channels. A repeated upper–lower index is summed; frequency sums are written explicitly.

Let $f^{i,\alpha}$ denote the input tensor, whose source channel is $f^{i,\alpha_s} = (s_h)_i$; any coordinate channels are held fixed. Define the source selector $(E_s)^\alpha = \delta^\alpha_{\alpha_s}$, so that

$$\frac{\partial f^{j,\alpha}}{\partial (s_h)_i} = \delta^j_i (E_s)^\alpha. \tag{25}$$

The lifting and projection matrices act on channels, whereas $(\Phi)^\xi_i$ and $(\Phi^*)^j_\xi$ act on spatial indices. For each fixed ξ, $(K_t(\xi))^{\beta'}_\beta$ is the learned channel matrix.

Proof. One Fourier layer. First consider a single layer without an activation or skip connection. Its tensor representation and source Jacobian are

$$\begin{aligned} \mathcal{G}_\theta(s_h)^j &= \sum_{\xi=1}^{n} (W_{\text{proj}})_{\beta'} (\Phi^*)^j_\xi (K_0(\xi))^{\beta'}_\beta (\Phi)^\xi_i (W_{\text{lift}})^\beta_\alpha f^{i,\alpha}, \\ (J_{\mathcal{G}_\theta})^j_i &= \sum_{\xi=1}^{n} (W_{\text{proj}})_{\beta'} (\Phi^*)^j_\xi (K_0(\xi))^{\beta'}_\beta (\Phi)^\xi_i (W_{\text{lift}})^\beta_\alpha (E_s)^\alpha. \end{aligned} \tag{26}$$

The Jacobian is independent of s_h. Channel maps and spatial Fourier transforms act on different tensor indices and can be interchanged. For a channel matrix $W : \mathbb{R}^{C_{\text{in}}} \to \mathbb{R}^{C_h}$, this follows from the Kronecker-product identity

$$(\Phi \otimes I_{C_h})(I_n \otimes W) = (I_n \otimes W)(\Phi \otimes I_{C_{\text{in}}}) = \Phi \otimes W. \tag{27}$$

We can therefore collect the channel contractions at each frequency:

$$\begin{aligned} (J_{\mathcal{G}_\theta})^j_i &= \sum_{\xi=1}^{n} (\Phi^*)^j_\xi \psi_\xi(\theta) (\Phi)^\xi_i, \\ \psi_\xi(\theta) &= (W_{\text{proj}})_{\beta'} (K_0(\xi))^{\beta'}_\beta (W_{\text{lift}})^\beta_\alpha (E_s)^\alpha. \end{aligned} \tag{28}$$

In matrix notation, $J_{\mathcal{G}_\theta} = \Phi^* \operatorname{diag}(\psi_\xi(\theta))\Phi$.

Adding the skip connection. The pointwise skip connection contributes the following term to the Jacobian:

$$\delta^j_i (W_{\text{proj}})_{\beta'} (W_0)^{\beta'}_\beta (W_{\text{lift}})^\beta_\alpha (E_s)^\alpha.$$

Using the identity

$$\delta^j_i = \sum_{\xi=1}^{n} (\Phi^*)^j_\xi (\Phi)^\xi_i,$$

we absorb this term into the same Fourier representation by replacing $K_0(\xi)$ with

$$B_0(\xi) := W_0 + K_0(\xi).$$

Stacking Fourier layers. Since spatial Fourier transforms and channel maps can be interchanged at every layer, adjacent forward and inverse transforms cancel between layers. With $B_t(\xi) := W_t + K_t(\xi)$, the resulting Fourier multipliers are

$$\psi_\xi(\theta) = W_{\text{proj}} B_{L-1}(\xi) \cdots B_0(\xi) W_{\text{lift}} E_s. \tag{29}$$

Thus the Fourier representation in Equation 28 extends to L layers, proving Equation 7 and independence from s_h. Unitary diagonalization gives the eigenvalues $\psi_\xi(\theta)$ and singular values $|\psi_\xi(\theta)|$. □

With nonlinear activations, the chain rule inserts spatially varying factors $\sigma_t'(z_t^{j\beta'})$, where z_t is the preactivation tensor. These factors need not commute with the spatial DFT and can therefore couple Fourier modes. Padding and cropping also introduce maps between grids, so the preceding cancellation argument no longer applies directly. Fixed coordinate channels do not change the source Jacobian in the activation-free model, since their source derivatives vanish.

A.2 Proof of Theorem 2.2

Proof. Since $\mathcal{G}_\theta(s_h) = J_{\mathcal{G}_\theta} s_h$, Parseval's identity gives

$$\begin{aligned}\mathcal{L}_\theta &= \frac{1}{N}\sum_m \big\|(J_{\mathcal{G}_\theta} - A_h^{-1})s_h^{(m)}\big\|_2^2 \\ &= \sum_\xi S_N(\xi)|\psi_\xi - a_\xi^{-1}|^2.\end{aligned} \tag{30}$$

Each summand is at most ε, so

$$|\psi_\xi - a_\xi^{-1}| \le \sqrt{\frac{\varepsilon}{S_N(\xi)}}, \qquad |a_\xi\psi_\xi - 1| \le \rho. \tag{31}$$

The singular values of the Fourier-diagonal matrix $J_{\mathcal{G}_\theta}A_h$ are $|a_\xi\psi_\xi|$. Therefore,

$$1-\rho \le |a_\xi\psi_\xi| \le 1+\rho, \qquad \kappa_2(J_{\mathcal{G}_\theta}A_h) \le \frac{1+\rho}{1-\rho}. \tag{32}$$

Unitary diagonalization also gives

$$\kappa_2(A_h) = \|A_h\|_2\|A_h^{-1}\|_2 = \frac{\max_\xi |a_\xi|}{\min_\xi |a_\xi|}. \tag{33}$$

Finally, $\rho^2 = \varepsilon/\min_\xi(S_N(\xi)/|a_\xi|^2)$, so the loss condition implies

$$\rho < \frac{\max_\xi |a_\xi| - \min_\xi |a_\xi|}{\max_\xi |a_\xi| + \min_\xi |a_\xi|} < 1, \qquad \frac{1+\rho}{1-\rho} < \kappa_2(A_h). \tag{34}$$

Combining these bounds proves the result. □

The loss threshold is sufficient but not necessary for condition-number reduction. For example, $J_{\mathcal{G}_\theta} = 2A_h^{-1}$ gives $\kappa_2(J_{\mathcal{G}_\theta}A_h) = 1$, even though the prediction error need not be small.

A.3 Proof of Corollary 2.3

Proof. Recall the empirical Fourier power and define its expectation:

$$\begin{aligned}S_N(\xi) &= \frac{1}{N}\sum_{m=1}^N \big|[\Phi s_h^{(m)}]_\xi\big|^2, \\ S(\xi) &:= \mathbb{E}[S_N(\xi)] = \mathbb{E}\big|[\Phi s_h]_\xi\big|^2.\end{aligned}$$

Under the unitary-DFT normalization of the Gaussian source covariance,

$$S(\xi) = \sigma_s^2\big((2\pi\xi)^2 + \tau^2\big)^{-\gamma}. \tag{35}$$

For each mode, $S_N(\xi)/S(\xi)$ is a normalized chi-square variable with N degrees of freedom for a real Fourier coefficient and $2N$ for a complex coefficient. The chi-square lower-tail bound of Laurent & Massart (2000, Lemma 1) therefore gives

$$\Pr(S_N(\xi) < (1-\delta)S(\xi)) \le \exp(-N\delta^2/4). \tag{36}$$

A union bound over the n modes shows that $S_N(\xi) \ge (1-\delta)S(\xi)$ holds simultaneously with probability at least $1 - n\exp(-N\delta^2/4)$. On this event, Equation 31 gives

$$|\psi_\xi - a_\xi^{-1}| \le \sqrt{\frac{\varepsilon}{S_N(\xi)}} \le \frac{\sqrt{\varepsilon}}{\sigma_s\sqrt{1-\delta}}\big((2\pi\xi)^2+\tau^2\big)^{\gamma/2}. \tag{37}$$

Dividing by $a_\xi^{-1} > 0$ proves the relative spectral-error bound. □

A.4 Proof of Theorem 2.4

We use the worst-case one-step GMRES residual factor defined in Equation 15.

Proof. For a real nonsingular matrix B, worst-case and ideal GMRES coincide at the first step (Faber et al., 2013):

$$\Gamma(B) = \min_{c\in\mathbb{R}} \|I - cB\|_2. \tag{38}$$

Fix $c \in \mathbb{R}$ and write the transformed matrix as

$$C_c := \Phi(I - cJA_h)\Phi^* = I - c(\Phi J\Phi^*)\,\mathrm{diag}(a_\xi).$$

Since Φ is unitary, multiplication by Φ and Φ^* preserves the spectral norm, so $\|I - cJA_h\|_2 = \|C_c\|_2$. The Cauchy–Schwarz inequality and the definition of the induced matrix norm give

$$|(C_c)_{\xi\xi}| = |e_\xi^* C_c e_\xi| \le \|e_\xi\|_2 \|C_c e_\xi\|_2 \le \|C_c\|_2.$$

Here, right multiplication by $\mathrm{diag}(a_\xi)$ scales column ξ by a_ξ, so

$$(C_c)_{\xi\xi} = 1 - ca_\xi(\Phi J\Phi^*)_{\xi\xi} = 1 - ca_\xi d_\xi.$$

Apply the triangle inequality $|z - w| \ge \big||z| - |w|\big|$, together with $a_\xi > 0$ and $g_\xi = a_\xi|d_\xi|$, gives

$$|1 - ca_\xi d_\xi| \ge \big|1 - |ca_\xi d_\xi|\big| = \big|1 - |c|g_\xi\big|.$$

Combining these steps and taking the maximum over ξ, we obtain

$$\begin{aligned} \|I - cJA_h\|_2 &= \|I - c(\Phi J\Phi^*)\,\mathrm{diag}(a_\xi)\|_2 \\ &\ge \max_\xi |1 - ca_\xi d_\xi| \\ &\ge \max_\xi \big|1 - |c|g_\xi\big|. \end{aligned} \tag{39}$$

The minimizing nonnegative scalar is

$$c_* := \arg\min_{c\ge 0} \max_\xi |1 - cg_\xi| = \frac{2}{g_{\max} + g_{\min}}.$$

Substituting this value gives

$$\min_{c\ge 0} \max_\xi |1 - cg_\xi| = \max_\xi |1 - c_* g_\xi| = \frac{g_{\max} - g_{\min}}{g_{\max} + g_{\min}}.$$

On the other hand, $MA_h = \Phi^*\,\mathrm{diag}(g_\xi)\Phi$, so $\|I - cMA_h\|_2 = \max_\xi |1 - cg_\xi|$. Applying Equation 38 to both matrices gives

$$\Gamma(MA_h) = \frac{g_{\max} - g_{\min}}{g_{\max} + g_{\min}} \le \Gamma(JA_h). \tag{40}$$

Conjugate-frequency symmetry of the Fourier diagonal of real J makes M real; since $|d_\xi| > 0$, it is SPD. The matrix MA_h is also real SPD, with Fourier eigenvalues $g_\xi > 0$. For any real P satisfying $|(\Phi P\Phi^*)_{\xi\xi}| = |d_\xi|$, the same argument gives

$$\Gamma(PA_h) \ge \frac{g_{\max} - g_{\min}}{g_{\max} + g_{\min}} = \Gamma(MA_h).$$

Since M belongs to this class and attains the bound, it is a minimizer. □

B NUMERICAL ALGORITHMS

We use GMRES for linear problems and Jacobian-free Newton–GMRES for nonlinear problems. All four variants share the solver framework and differ only in the fixed left-preconditioning action $M(w)$ described in Section B.2. For each variant, the initial solution is either zero or $\mathcal{G}_\theta(s_h)$, as specified in the experimental comparison. Solver implementation, parameter settings, and hardware are reported in Appendix C.1.

B.1 ITERATIVE SOLVERS

B.1.1 GMRES

For a linear system $Az = b$, left-preconditioned generalized minimal residual (GMRES) solves $MAz = Mb$ using operator actions $w \mapsto Aw$ and $w \mapsto M(w)$ (Saad & Schultz, 1986). Starting a restart cycle from z_0, let $r_0 = b - Az_0$. After j Krylov steps, GMRES computes

$$\begin{aligned} z_j &= \operatorname*{arg\,min}_{z \in z_0 + \mathcal{K}_j(MA, Mr_0)} \|M(b - Az)\|_2, \\ \mathcal{K}_j(MA, Mr_0) &= \operatorname{span}\{Mr_0, (MA)Mr_0, \ldots, (MA)^{j-1}Mr_0\}. \end{aligned} \tag{41}$$

Arnoldi iteration constructs an orthonormal basis of this space using $w \mapsto M(Aw)$, and Givens rotations incrementally update the QR factorization of the resulting small least-squares problem. At a restart, the latest iterate becomes the next cycle's initial guess. The parameter `restart` specifies the maximum number of Krylov steps per cycle, and `maxcycles` limits the number of cycles. For prescribed relative and absolute tolerances `rtol` and `atol`, convergence is checked using the true residual:

$$\|Az - b\|_2 \le \max\{\texttt{rtol}\, \|b\|_2, \texttt{atol}\}. \tag{42}$$

The nonlinear problems use adaptive inner tolerances, as described below.

B.1.2 JACOBIAN-FREE NEWTON–GMRES

Define the discrete nonlinear residual by $F_h(u_h; s_h) = \mathcal{N}_h(u_h) - s_h$. At Newton step p, the correction satisfies

$$A_p \delta u_h^{(p)} = -F_h(u_h^{(p)}; s_h), \qquad A_p = J_{\mathcal{N}_h}(u_h^{(p)}). \tag{43}$$

We apply the Jacobian through Jacobian–vector products (JVPs), without explicitly forming the matrix (Knoll & Keyes, 2004):

$$\mathcal{A}_p(w) = J_{\mathcal{N}_h}(u_h^{(p)})w. \tag{44}$$

GMRES approximately solves the correction equation from the zero vector, and Newton takes a full step,

$$u_h^{(p+1)} = u_h^{(p)} + \delta u_h^{(p)}. \tag{45}$$

The PDE Jacobian action changes with the Newton iterate, whereas M is constructed once and held fixed throughout the solve.

Inexact Newton and stopping criteria. Solving Equation 43 to high accuracy is unnecessary in the early Newton iterations, when the nonlinear residual is still large. We therefore use an inexact Newton method, terminating the inner GMRES solve once its true residual satisfies

$$\|\mathcal{A}_p(\delta u_h^{(p)}) + F_h(u_h^{(p)}; s_h)\|_2 \le \eta_p R_p. \tag{46}$$

where $R_p = \|F_h(u_h^{(p)}; s_h)\|_2$. The forcing term η_p controls the required accuracy of the linear solve and is selected adaptively using an Eisenstat–Walker (EW) type strategy (Eisenstat & Walker, 1996). This allows looser inner solves early on and tighter tolerances as Newton approaches convergence. The outer iteration stops when $R_p \le \tau_{\mathrm{NL}}$, where $\tau_{\mathrm{NL}} = \texttt{rtol}\, R_0 + \texttt{atol}$, with at most `maxiter` Newton updates. Here the tolerances apply to the outer nonlinear residual; the inner tolerance is set by η_p.

Algorithm 1 Jacobian-free Newton–GMRES with EW forcing

Require: Residual $F_h(\cdot; s_h)$, initial state $u_h^{(0)}$, fixed action $M(w)$
Require: `maxiter`, `restart`, `maxcycles`, `rtol`, `atol`

$R_0 \leftarrow \|F_h(u_h^{(0)}; s_h)\|_2$; $\tau_{\mathrm{NL}} \leftarrow \texttt{rtol}\, R_0 + \texttt{atol}$
for $p = 0, \ldots, \texttt{maxiter} - 1$ **do**
  if $R_p \le \tau_{\mathrm{NL}}$ **then**
    return $u_h^{(p)}$, success
  end if
  Select η_p using the EW strategy.
  Define $\mathcal{A}_p(w)$ by the JVP in Equation 44.
  Solve $\mathcal{A}_p(\delta u_h^{(p)}) = -F_h(u_h^{(p)}; s_h)$ with left-preconditioned GMRES.
  Use zero initial correction, `restart`, `maxcycles`, and Equation 46.
  if the inner solve fails to converge **then**
    return $u_h^{(p)}$, failure
  end if
  $u_h^{(p+1)} \leftarrow u_h^{(p)} + \delta u_h^{(p)}$; $R_{p+1} \leftarrow \|F_h(u_h^{(p+1)}; s_h)\|_2$
end for
return $u_h^{(\texttt{maxiter})}$, success if $R_{\texttt{maxiter}} \le \tau_{\mathrm{NL}}$, otherwise failure

B.2 Preconditioning actions

For an input vector $w \in \mathbb{R}^n$, the four actions are as follows.

None. Use the identity action $M(w) = w$.

Direct. Apply the trained FNO's source Jacobian using a JVP:

$$M(w) = J_{\mathcal{G}_\theta}(s_h)w. \tag{47}$$

FPS (fast Poisson solver). Extract the diagonal of the reference discrete Poisson operator L_h in this basis and use its entrywise reciprocals:

$$\begin{aligned} d_{\mathrm{FPS},\xi} &= \left[\Phi L_h(\Phi^* \boldsymbol{e}_\xi)\right]_\xi, \\ M(w) &= \Phi^* \operatorname{diag}\Big(\big(d_{\mathrm{FPS},\xi}^{-1}\big)_{\xi=1}^{n}\Big)\, \Phi w. \end{aligned} \tag{48}$$

Purify. Extract the Fourier diagonal of the same source Jacobian once, and use its magnitudes as FFT multipliers:

$$\begin{aligned} d_\xi &= \left[\Phi J_{\mathcal{G}_\theta}(s_h)(\Phi^* \boldsymbol{e}_\xi)\right]_\xi, \\ M(w) &= \Phi^* \operatorname{diag}\Big(\big(|d_\xi|\big)_{\xi=1}^{n}\Big)\, \Phi w. \end{aligned} \tag{49}$$

Both FPS and Purify cache their diagonal coefficients $d_{\mathrm{FPS},\xi}$ and d_ξ, respectively, for reuse throughout the iterative solve. Each preconditioner therefore requires $O(n)$ storage for these cached coefficients.

C Additional experimental details and results

C.1 Solver implementation and computational environment

Implementation and parameter settings. We evaluate JVPs using JAX forward-mode automatic differentiation. For linear problems, we use `restart` $= 400$, `maxcycles` $= 5$, `rtol` $= 10^{-8}$, and `atol` $= 10^{-10}$, allowing at most 2000 Krylov steps. For nonlinear problems, the outer Newton iteration uses `maxiter` $= 20$, `rtol` $= 10^{-8}$, and `atol` $= 10^{-8}$. Each inner GMRES solve uses `restart` $= 20$ and `maxcycles` $= n$, with its tolerance selected by the EW strategy in Equation 46.

Hardware and software. The server has two Intel Xeon Silver 4410Y CPUs (24 physical cores and 48 hardware threads in total), approximately 251.3 GiB of OS-visible memory, and four NVIDIA GeForce RTX 4090 GPUs with 24 GiB-class memory each. The default solver configuration uses a single GPU (physical index 3). The software environment is Ubuntu 24.04.4 LTS, Python 3.12.3, JAX/jaxlib 0.6.2, and NumPy 2.4.4, with NVIDIA driver 580.126.20, JAX CUDA 12 plugin/PJRT 0.6.2, and the installed CUDA runtime package version 12.9.79. Solver computations use double precision.

C.2 Well-posedness and constant-mode correction

The periodic discrete Poisson operator has a constant nullspace. A rank-one correction removes this singularity while preserving the nonconstant Fourier modes. Define the constant-mode projector $\Pi_0^{(h)} = n^{-1}\mathbf{1}\mathbf{1}^\top$. For the Poisson1D data-generation discretization, the corrected operator is

$$\widetilde{A}_h = A_h + h^{-2}\Pi_0^{(h)}, \tag{50}$$

where A_h here denotes the uncorrected periodic discrete Poisson matrix. The correction replaces its zero eigenvalue by h^{-2} and leaves all nonconstant Fourier modes unchanged. Thus $\widetilde{A}_h$ is SPD and Fourier-diagonal, so the corrected discrete Poisson problem has a unique solution that depends continuously on the source at each fixed grid. Elsewhere in the experiments, A_h denotes the corrected matrix.

C.3 Benchmark setup and predictive performance

This section describes the data and training configurations for the four benchmarks in Section 3. Prediction errors characterize the learned solution maps; solver statistics are reported separately. Only parameter settings used in the main experiments are included.

Data-generation protocol. We construct source–solution pairs by sampling a source s_h and numerically solving the discretized PDE for its reference solution u_h. Poisson1D and Helmholtz2D use direct linear solves; nonlinear Darcy2D and Convec2D use Newton iterations. All random fields use data-generation seed 42. The two-dimensional sources have $(\gamma, \tau, \sigma_s) = (2, 1, 1000)$ and zero spatial mean. Periodic random-field sampling does not impose periodic solution boundary conditions. The reference-solution procedures below generate the datasets; the iterative solvers evaluated in the main experiments are described in Appendix B.

Shared model and training setup. All models take the source and spatial coordinates as inputs and predict a scalar solution. Each Fourier block combines spectral and pointwise transformations. Training uses Adam with shuffled batches, an initial learning rate of 0.001, and cosine decay; checkpoints are saved every 2000 epochs. Table 5 collects the architecture and training settings. The case descriptions below focus on data generation, model-specific details, and prediction results.

Table 5: FNO architecture and training settings for the four benchmarks.

Setting	Poisson1D	Helmholtz2D	Darcy2D	Convec2D
Model architecture				
Fourier blocks	2	2	3	3
Hidden channels	16	32	32	32
Modes per block	12	12×12	12×12	12×12
Input lifting	$2 \to 16$	$3 \to 32$	$3 \to 32$	$3 \to 32$
Output projection	$16 \to 32 \to 1$	$32 \to 64 \to 1$	$32 \to 64 \to 1$	$32 \to 64 \to 1$
Activation	None / GELU	GELU	GELU	GELU
Normalization	None	Spatial RMS	Spatial RMS	Spatial RMS
Padding type	None / zero / circular	Zero	Zero	Zero
Padding per side	0 / 1	1	1	1
Parameters	13,457	594,305	890,305	890,305
Training schedule				
Epochs	20,000	40,000	40,000	40,000
Batch size	200	100	100	100
Cosine-decay updates	80,000	320,000	320,000	320,000
Test interval (epochs)	100	1000	1000	1000

Padding is specified per spatial axis. Poisson1D variants are detailed below; their cosine decay to zero is inferred from the implementation default and logged learning rates, rather than an explicit saved schedule field.

C.3.1 POISSON1D

Equation and data. We discretize Eq. 17 on a periodic, uniform 128-point grid with $h = 1/128$. Sources are periodic Gaussian random fields with $(\gamma, \tau, \sigma_s) = (1, 1, 25)$; the constant Fourier mode is retained. The 1000 source–solution pairs are split into 800 training and 200 test samples.

Reference solution. For each sampled source s_h, we obtain u_h by directly solving the second-order discrete Poisson system $A_h u_h = s_h$. The matrix includes the constant-mode correction $h^{-2}\Pi_0^{(h)}$ described in Appendix C.2.

Model variants. The three Poisson1D variants use no activation or padding, GELU with zero padding, or GELU with circular padding (Figure 3). The padded variants add one entry at each end. All retain 12 modes and have 13,457 parameters.

Predictive performance. Table 6 reports the final-epoch results for the three configurations in the main comparison.

Table 6: Poisson1D prediction results after 20,000 epochs on 200 test samples. All three configurations retain 12 modes per Fourier block.

Activation / padding	Training loss ($\mathcal{L}_\theta$)	Test relative L^2 error
None / none	7.20×10^{-9}	9.36×10^{-4}
GELU / zero	1.38×10^{-8}	1.30×10^{-3}
GELU / circular	5.97×10^{-8}	2.77×10^{-3}

C.3.2 HELMHOLTZ2D

Equation and data. We discretize Eq. 18 on a 64×64 cell-centered grid with homogeneous Dirichlet boundary conditions. For each $k_0 \in \{8, 12, 16\}$, the coefficient field in Eq. 19 is fixed while the source varies. Each dataset contains 1000 source–solution pairs, split into 800 training and 200 test samples.

Reference solution. We obtain u_h by directly solving the five-point discrete linear system $A_h u_h = s_h$. The JAX implementation uses a dense solve with multiple right-hand sides and a relative residual acceptance tolerance of 10^{-8}.

Model details. A separate FNO is trained at each wavenumber. Spectral and pointwise transformations are added before spatial RMS normalization; GELU is applied between Fourier blocks and in the hidden output projection.

Predictive performance. Table 7 gives the final-epoch results. The relative prediction error increases across the three wavenumber settings, while the training MSE is not monotone.

Table 7: Helmholtz2D prediction results after $40{,}000$ epochs. Each wavenumber has a separately trained model and 200 test samples.

k_0	Training loss ($\mathcal{L}_\theta$)	Test relative L^2 error
8	2.42×10^{-5}	6.13×10^{-3}
12	1.25×10^{-5}	9.43×10^{-3}
16	8.03×10^{-4}	3.04×10^{-2}

C.3.3 NONLINEAR DARCY2D

Equation and data. We discretize Eq. 22 on a 64×64 cell-centered grid with homogeneous Dirichlet boundary conditions. For each $\sigma_g \in \{10, 20, 40\}$, one periodic, zero-mean log-permeability field is sampled with $(\gamma, \tau) = (2, 1)$ and scale σ_g; the resulting permeability is fixed while the source varies. Each dataset contains 1000 accepted source–solution pairs, split into 800 training and 200 test samples.

Reference solution. We obtain u_h by applying Newton iterations, initialized at zero, to the conservative second-order discrete nonlinear equation $\mathcal{N}_h(u_h) = s_h$. Each Newton correction is computed by GMRES with a fixed FFT preconditioner based on the Jacobian at zero. The JAX solver accepts solutions with original residual norm below 10^{-9}; sampling continues until 1000 pairs are accepted.

Model details. A separate FNO is trained at each log-permeability scale using the settings in Table 5. Permeability is fixed within each dataset and is not an input channel.

Predictive performance. Table 8 reports the final-epoch prediction results for the three log-permeability scales.

Table 8: Darcy2D prediction results after $40{,}000$ epochs. Each coefficient setting has a separately trained model and 200 test samples.

σ_g	Training loss ($\mathcal{L}_\theta$)	Test relative L^2 error
10	8.42×10^{-7}	2.72×10^{-2}
20	1.74×10^{-6}	3.03×10^{-2}
40	4.95×10^{-6}	4.63×10^{-2}

C.3.4 CONVEC2D

Equation and data. We discretize Eq. 24 on a periodic 64×64 cell-centered grid for $Re \in \{100, 250, 500\}$. For each setting, we sample 1000 candidate sources and retain only those with converged reference solutions. The first 800 retained pairs form the training set; The retained source sets can differ across Reynolds numbers.

Reference solution. We obtain u_h by Newton iterations with Armijo line search, using a five-point Laplacian, centered convection differences, and the constant-mode correction $h^{-2}\Pi_0^{(h)} u_h$. The JAX

solver uses an absolute residual tolerance of 10^{-10}, zero relative tolerance, and at most 200 Newton iterations.

Model details. A separate FNO is trained at each Reynolds number; Re is not an input channel. Network padding is zero padding even though the PDE is periodic.

Predictive performance. Table 9 reports the final-epoch results together with the actual train/test counts. Relative test error is not monotone in Re; these averages use the separately filtered test sets at each setting.

Table 9: Convec2D prediction results after $40{,}000$ epochs. Test-set sizes reflect convergence filtering during label generation.

Re	Train / test	Training loss ($\mathcal{L}_\theta$)	Test relative L^2 error
100	800 / 200	6.04×10^{-7}	7.41×10^{-2}
250	800 / 194	5.63×10^{-7}	9.76×10^{-2}
500	800 / 83	4.23×10^{-7}	8.56×10^{-2}

C.4 POISSON1D RESOLUTION TRANSFER

FNOs can be evaluated at resolutions different from their training grid (Li et al., 2021). We train at $n = 128$ and evaluate $J_{\mathcal{G}_\theta}(s_h)A_h$ and $J_{\mathrm{purify}}(s_h)A_h$ at $n \in \{128, 256, 512\}$ without retraining. Using the checkpoint at epoch $20{,}000$, we evaluate sample 817, drawn at random from the test set and then held fixed across resolutions. Table 10 reports spectral condition numbers and their reduction factors relative to the unpreconditioned matrix. For this sample, preconditioning with $J_{\mathrm{purify}}(s_h)$ reduces the condition number by more than $220\times$ at all three resolutions, whereas $J_{\mathcal{G}_\theta}(s_h)$ worsens conditioning at $n = 512$.

Table 10: Poisson1D resolution transfer for randomly selected test sample 817, using the epoch-$20{,}000$ checkpoint trained at $n = 128$. Each reduction factor is $\kappa_2(A_h)$ divided by the corresponding preconditioned condition number; a factor below one indicates worse conditioning.

n	$\kappa_2(A_h)$	$\kappa_2(J_{\mathcal{G}_\theta}(s_h)A_h)$	Reduction	$\kappa_2(J_{\mathrm{purify}}(s_h)A_h)$	Reduction
128	1660.38	493.65	$3.36\times$	7.51	$220.95\times$
256	6640.52	3704.87	$1.79\times$	29.28	$226.81\times$
512	26561.07	67302.47	$0.39\times$	116.29	$228.41\times$

C.5 CONVEC2D TIMING BREAKDOWN

Table 11 reports the recorded timing means for all three Reynolds numbers and both initializations. Stage means and T_{succ} use successful solves only. The recorded totals equal initialization plus assembly plus iteration; the separately logged GMRES time is part of iteration and is not added again. The nonzero recorded initialization cost for zero initialization is retained.

Table 11: Convec2D recorded timing means in seconds. Init., assembly, iteration, and T_{succ} condition on success. Dashes indicate no successful solves. Totals include all three recorded stages and are rounded independently to three decimals.

Re	Precond.	Success	Init.	Assembly	Iteration	T_{succ}
			Zero initialization			
100	None	39/50	0.000	0.000	0.304	0.304
	FPS	45/50	0.000	0.044	0.093	0.137
	Direct	0/50	–	–	–	–
	Purify	46/50	0.000	0.156	0.085	0.242
250	None	4/50	0.000	0.000	0.624	0.624
	FPS	6/50	0.000	0.044	0.143	0.188
	Direct	0/50	–	–	–	–
	Purify	6/50	0.000	0.156	0.098	0.254
500	None	0/50	–	–	–	–
	FPS	0/50	–	–	–	–
	Direct	0/50	–	–	–	–
	Purify	0/50	–	–	–	–
			Neural initialization			
100	None	50/50	<0.001	0.000	0.236	0.236
	FPS	50/50	<0.001	0.044	0.045	0.090
	Direct	0/50	–	–	–	–
	Purify	50/50	<0.001	0.156	0.055	0.211
250	None	50/50	<0.001	0.000	0.492	0.492
	FPS	49/50	<0.001	0.044	0.096	0.141
	Direct	0/50	–	–	–	–
	Purify	50/50	<0.001	0.156	0.083	0.239
500	None	34/50	<0.001	0.000	0.835	0.835
	FPS	47/50	<0.001	0.044	0.166	0.210
	Direct	0/50	–	–	–	–
	Purify	47/50	<0.001	0.156	0.102	0.259

At the tested problem sizes and iteration counts, Purify's iteration-time savings over FPS, where observed, do not offset its higher assembly cost. At $Re = 500$ with neural initialization, both methods successfully solve $47/50$ sources. Purify reduces the mean iteration time over successful solves from 0.166 to 0.102 seconds, but increases the mean assembly time from 0.044 to 0.156 seconds. Its mean total time is therefore higher: 0.259 versus 0.210 seconds for FPS. At higher resolutions and larger system sizes, iterative solves may become the dominant runtime bottleneck, potentially allowing the iteration-time savings to outweigh the assembly cost. Evaluating this trade-off at larger scales remains future work.