LEARNING SPECTRAL PRECONDITIONER FOR LINEAR SYSTEM SOLVER WITH APPLICATION TO FOUR-DIMENSIONAL VARIATIONAL DATA ASSIMILATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Four-dimensional variational data assimilation (4DVar) requires solving large-scale, ill-conditioned linear systems within its inner loop, where convergence depends on the spectral properties of the system matrix. Traditional limited-memory preconditioners (LMPs) reuse historical Lanczos/Ritz vectors but may lose effectiveness as the relevant subspace rotates between assimilation windows. We propose a neural spectral preconditioner that predicts a current-window subspace from historical features using a Fourier neural operator. The network output is orthonormalized before the preconditioner is constructed. Under the control-variable-transformed model, this constructed preconditioner is symmetric positive definite. Experiments on the strongly nonlinear Lorenz-96 system demonstrate that the proposed method reduces the number of conjugate gradient iterations by 54.16% compared with unpreconditioned CG and by 39.6% compared with LMP. On the quasi-geostrophic (QG) system, the method reduces the iteration count by 22.5% relative to LMP and improves fixed-iteration solution accuracy in the reported tests. The method retains the PCG solution framework and yields consistent reductions in iteration count across the evaluated dynamical regimes.

1 INTRODUCTION

Four-dimensional variational data assimilation (4DVar) is the standard approach for correcting initial conditions in modern operational numerical weather prediction, addressing the inherent sensitivity to chaotic dynamics (Bannister, 2017). To reduce computational cost, operational systems adopt an incremental analysis scheme with an outer-inner loop structure. The outer loop computes observation increments and reference trajectories using a high-resolution nonlinear model. The inner loop constructs a quadratic objective functional using linear approximations and aims to solve a large-scale symmetric positive-definite linear system $A\delta x = b$. The efficiency of solving this system is critical for data assimilation.

High-dimensional linear systems are typically solved using the preconditioned conjugate gradient (PCG) method, whose convergence rate depends on the condition number of A (Courtier et al., 1994; Fisher, 1998). In 4DVar, the condition number is on the order of 10^4 to 10^5, causing classical algorithms to converge extremely slowly (Lin & Yongzhu, 2017). The control variable transform (CVT) serves as a first-level preconditioner (Lorenc, 1981), but additional preconditioning is still required (Haben et al., 2011). When the system matrix is explicitly available, methods like incomplete Cholesky and SOR exist, but they are not applicable to the matrix-free setting of 4DVar. In practice, mainstream systems adopt the limited-memory preconditioner (LMP) (Tshimanga et al., 2008), which reuses historical Ritz pairs from previous assimilation cycles. However, this strategy degrades when the subspace varies significantly across windows, especially in strongly nonlinear systems.

Recent advances in deep learning provide new opportunities. Existing approaches follow two directions: constructing surrogate models to directly predict analysis fields (Rozet & Louppe, 2023; Kochkov et al., 2024; Arcucci et al., 2021) and using neural networks to assist with preconditioning (Trappler & Vidard, 2025; Moazzami et al., 2025). However, these methods suffer from limited interpretability or risk replacing traditional solvers entirely.

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.

In this paper, we propose a deep learning-based method to learn a preconditioner that surpasses LMP using accumulated historical data. The learned preconditioner is integrated with CG to accelerate 4DVar linear systems. In contrast to field-to-field prediction approaches, our method accelerates iteration while preserving correctness in the worst case; solution quality is not degraded. The combination of deep learning and traditional methods provides a reliable and efficient framework.

Our contributions are as follows:

- We propose an FNO-based model that uses historical information to predict a preconditioner constructed from estimated Lanczos vectors.
- We employ a weighted Grassmann metric to measure subspace distance, accounting for potential rotation.
- We orthonormalize the predicted basis at inference and specify the conditions under which the resulting trace-scaled preconditioner is symmetric positive definite.
- We evaluate PCG iteration counts and fixed-iteration errors on the Lorenz-96 model and a quasi-geostrophic (QG) system, including tests across assimilation windows and dynamical regimes.

2 PRELIMINARIES

2.1 VARIATIONAL DATA ASSIMILATION PROBLEM

In 4DVar, the estimation of the initial condition x_0 is formulated as a nonlinear least-squares optimization problem. The objective functional $J(x_0)$ is defined as

$$J(x_0) = \frac{1}{2}\|x_0 - x_b\|^2_{B^{-1}} + \frac{1}{2}\sum_{k=0}^{K}\|y_k - \mathcal{H}_k(\mathcal{M}_{t_0\to t_k}(x_0))\|^2_{R_k^{-1}}. \tag{1}$$

Here, the assimilation window is $[t_0, t_k]$. The variable $x_b \in \mathbb{R}^n$ denotes the background field, and $B \in \mathbb{R}^{n\times n}$ is the background error covariance matrix. The observation at time t_k is denoted by $y_k \in \mathbb{R}^m$, with corresponding observation error covariance matrix $R_k \in \mathbb{R}^{m\times m}$. The operator $\mathcal{M}_{t_0\to t_k} : \mathbb{R}^n \to \mathbb{R}^n$ represents the nonlinear evolution model, and $\mathcal{H}_k : \mathbb{R}^n \to \mathbb{R}^m$ denotes the observation operator. In practical 4DVar applications, we generally have $m < n$.

This nonlinear optimization problem is typically solved by finding the stationary point satisfying $\nabla J(x_0) = 0$ using the incremental method. The process of computing the optimal increment $\delta x = x_0 - x_b$ is equivalent to solving a large-scale linear system $A\delta x = b$ within the Gauss-Newton framework. Specifically, the system matrix takes the form

$$A = B^{-1} + \sum_{k=0}^{K} M_k^T H_k^T R_k^{-1} H_k M_k. \tag{2}$$

Here, M_k denotes the tangent linear model operator from t_0 to t_k, and H_k is the linearized observation operator at t_k. The first term B^{-1} originates from the background field regularization; see details in the Appendix. The second summation term incorporates observational constraints accumulated over the entire assimilation window. When the observations are substantially more accurate than the background field, the summation term dominates the system matrix. The cumulative contribution of these terms causes A to become highly ill-conditioned.

2.2 FIRST-LEVEL PRECONDITIONING: CONTROL VARIABLE TRANSFORM

In 4DVar, there exists a natural preconditioning strategy known as the control variable transform (CVT). The transformation is defined as $\delta z = B^{-1/2}\delta x$. Under this transformation, the original system is converted into an equivalent linear system $S\delta z = \tilde{b}$, where S and $\tilde{b}$ are defined by

$$S = I + \sum_{k=0}^{K}(B^{1/2} M_k^T H_k^T R_k^{-1} H_k M_k B^{1/2}), \quad \tilde{b} = B^{1/2} b. \tag{3}$$

If each observation-error covariance R_k is positive definite, every term added to I in Eq. (3) is positive semidefinite. Thus $S \succeq I$ and $\lambda_{\min}(S) \geq 1$ in exact arithmetic. This lower bound does not imply that S is well conditioned: its largest eigenvalues may still be much greater than one. The control variable transform (CVT) supplies the first level of preconditioning, while the second level targets the remaining spectral spread.

3 Acceleration via Neural Spectral Preconditioning

3.1 Spectral Preconditioning Mechanism

Assume that the matrix S admits the eigendecomposition

$$S = U\Sigma U^T.$$

Motivated by the limited-memory preconditioner (LMP) (Tshimanga et al., 2008), we consider the following spectral preconditioner:

$$P = \alpha(I - QQ^T) + QT^{-1}Q^T, \tag{4}$$

where $Q \in \mathbb{R}^{n\times k}$ has orthonormal columns ($Q^TQ = I_k$), $\alpha > 0$, and

$$T = Q^TSQ$$

is the projection of S onto the subspace spanned by Q. When $\alpha = 1$ and $Q = U$ contains the complete eigenbasis of S, Eq. (4) gives $P = S^{-1}$. Obtaining this complete eigenbasis is infeasible in large-scale 4DVar. LMP instead uses a limited number of historical Ritz or Lanczos directions. For an arbitrary approximate subspace, Eq. (4) is symmetric positive definite under the conditions stated above, but a smaller condition number or fewer PCG iterations is not guaranteed. These depend on how well the subspace captures the difficult directions of the current system.

Operational weather forecasting systems impose extremely strict online timeliness requirements on the assimilation procedure. Explicitly constructing an accurate invariant subspace matrix V (aka eigenvectors)for the current system is computationally comparable to solving the original problem itself. Therefore, existing LMP approaches reuse the subspace matrix V from the previous assimilation window. However, directly reusing the complete subspace from the previous timestep introduces geometric misalignment between eigensubspaces across different assimilation windows. As the assimilation window advances, the residual projection error accumulates rapidly. Alternatively, computing only a very small number of basis vectors in the current window provides limited low-dimensional spectral information. Such information is insufficient to capture enough stiff modes in highly ill-conditioned systems whose effective rank significantly exceeds the number of computed vectors. Consequently, the resulting preconditioning effect remains limited. See Appendix for further discussion.

To address this issue, we propose using Lanczos vectors predicted by a deep learning model at the current assimilation timestep to construct the preconditioner. We choose Lanczos vectors instead of Ritz vectors because they exhibit similar empirical performance while avoiding the additional small-scale singular value decomposition required for Ritz vector computation.

3.2 Neural Spectral Preconditioning Acceleration Framework

3.2.1 Subspace Prediction from Low-Dimensional Historical Features

As discussed above, directly reusing historical spectral information introduces nonlinear subspace rotation errors, which may significantly degrade the effectiveness of the preconditioner. To address this issue, we propose training a deep learning model to predict a subspace that is better aligned with the spectral structure of the current assimilation timestep. Motivated by the successful application of FNOs in PDE solution prediction, we adopt an off-the-shelf FNO architecture as the backbone of our prediction framework. The inputs to the network consist of the historical Lanczos basis vectors V_{pre} together with the corresponding physical increment solution δx_{pre}, both of which are available from previous assimilation windows and can be recorded during offline data preparation.

The network concatenates the historical basis vectors and increment solutions as input feature channels. Using the historical sequence, the model predicts an output matrix V_{net} intended to span the target current-window Lanczos subspace V_{GT}. Historical features are available from preceding assimilation cycles. At inference, the predicted output is orthonormalized and projected onto the current-window system operator to construct the preconditioner.

3.2.2 Weighted Geometric Loss

Let V_{net} denote the raw network output and V_{GT} the ground-truth Lanczos basis. To enable supervised learning, we design a weighted geometric loss for their respective subspaces.

Traditional mean squared error evaluates basis vectors element-wise and is sensitive to rotations within the same subspace. We instead use a subspace projection loss motivated by Grassmann geometry (Edelman et al., 1998; Luo et al., 2024). During training, differentiable QR decomposition produces an orthonormal basis $Q_{\mathrm{train}} = \mathrm{qr}(V_{\mathrm{net}})$. The projection residual is

$$R = V_{\mathrm{GT}} - Q_{\mathrm{train}}(Q_{\mathrm{train}}^T V_{\mathrm{GT}}). \tag{5}$$

This residual measures the pure geometric discrepancy between the true and predicted subspaces. The formulation is rotation-invariant and effectively eliminates the ambiguity caused by internal coordinate rotations of the basis vectors.

The convergence speed of PCG is primarily influenced by dominant spectral modes associated with large eigenvalues. To emphasize these physically important directions, we further incorporate spectral-energy weighting into the geometric projection loss. The tridiagonal matrix

$$T_{\mathrm{GT}} = V_{\mathrm{GT}}^T S V_{\mathrm{GT}}$$

associated with the target subspace is used as a surrogate spectral-energy weight. The loss function in Eq. (6) is applied uniformly across our experiments. For L96, the network directly predicts V_{net}. For QG, it predicts a residual correction ΔV added to the historical basis (i.e., $V_{\mathrm{net}} = V_{\mathrm{pre}} + \Delta V$), with the same loss formulation.

In both settings, the loss computes the projection residual in Eq. (5), weighted by T_{GT}. The final energy-weighted geometric loss is

$$\mathcal{L} = \frac{1}{B}\sum_{i=1}^{B}\left(\|RT_{\mathrm{GT}}\|_F^2 + \gamma\|V_{\mathrm{net}}^T V_{\mathrm{net}} - I_k\|_F^2\right). \tag{6}$$

Here, B denotes the batch size. The weighted residual emphasizes spectral directions represented in T_{GT}; the second term encourages orthogonality during training. This penalty alone does not ensure orthogonality at inference, so we apply QR again when constructing the preconditioner.

3.2.3 Orthonormalization and Positive Definiteness

At inference, we orthonormalize the network output before using it in Eq. (4):

$$Q = \mathrm{qr}(V_{\mathrm{net}}), \qquad Q^T Q = I_k. \tag{7}$$

Here qr denotes the reduced QR factorization, returning the k orthonormal columns. The same Q is used in the projector, in $T = Q^T S Q$, and in the coarse correction. With $S \succeq I$, every $y \in \mathbb{R}^k$ satisfies

$$y^T T y = (Qy)^T S (Qy) \geq \|Qy\|_2^2 = \|y\|_2^2, \quad \text{hence} \quad T \succeq I_k. \tag{8}$$

We choose $\alpha = k/\operatorname{tr}(T) > 0$. For any nonzero $x \in \mathbb{R}^n$, Eq. (4) then gives

$$x^T P x = \alpha\|(I - QQ^T)x\|_2^2 + (Q^T x)^T T^{-1}(Q^T x) > 0. \tag{9}$$

This argument establishes positive definiteness under the stated assumptions. The effect on PCG convergence is a separate question. In particular, α scales directions orthogonal to Q; if it is too small relative to the remaining spectrum, some preconditioned eigenvalues may become small and convergence may deteriorate. The implementation retains an inverse-eigenvalue guard for exceptional numerical cases: for an eigenvalue of T below ϵ, the corresponding inverse is replaced by one. Because Eq. (8) gives $\lambda_{\min}(T) \geq 1$ in exact arithmetic, this guard is not the mechanism that controls the effect of α on the complementary subspace.

4 EXPERIMENTS

In this section, we evaluate the proposed preconditioner on two dynamical systems: the canonical Lorenz-96 system (L96) and a two-layer quasi-geostrophic (QG) model. These two systems serve as standard benchmarks for evaluating the performance of data assimilation algorithms, representing idealized chaotic dynamics and more realistic high-dimensional geophysical flows, respectively. See the details of the two governing equations in Appendix.

4.1 EXPERIMENTAL SETUP AND BASELINE METHODS

For both dynamical systems, we first describe the methodology for constructing the training datasets. The input to the prediction model consists of Lanczos vectors corresponding to the 4DVar linear systems at the previous timestep, while the target labels represent the oracle Lanczos vectors for the current timestep. The neural model is trained to predict the preconditioner for the current timestep using only historical information from previous assimilation cycles. To construct these paired datasets, we obtain the linear system matrix A via automatic differentiation of the 4DVar objective function, which depends on the governing equations as well as the background and observation error standard deviations. For both systems, these parameters are set to $\sigma_b = 1.0$ and $\sigma_o = 0.1$, respectively. Additional hyperparameter settings are detailed below:

1. **Lorenz-96 system**: the forcing term F is set to 8.0. The assimilation window length is 10 time steps, and the correlation length scale L is set to 4.0. We also consider state dimensions of $40, 80, 160, 1000$, with corresponding prediction subspace dimensions K set to $20, 30, 40, 100$, respectively. When testing generalization to unseen dynamical regimes, we consider forcing values of $F = 6.0, 10.0, 12.0$. To generate statistically independent test samples, the system is first spun up for 200 steps to reach the chaotic attractor. Each test sample is then obtained by randomly advancing 5 to 15 steps from the previous anchor to decorrelate temporal dependencies, followed by a fixed 10-step assimilation window to reach the target timestep. This random-gap protocol evaluates robustness against irregular historical state availability.
2. **QG system**: the system is discretized on a 32×32 (or $48 \times 48, 64 \times 64$) horizontal grid with two vertical layers, yielding a state dimension $D = 2048$ (or $4608, 8192$). The time step is $\Delta t = 7200$ seconds. The assimilation window spans 4 days, with observations sampled every 24 hours at 50% of the spatial grid points to mimic realistic satellite coverage gaps. The subspace dimension K is set to $45, 60, 80$ for the respective grid resolutions. The model undergoes a 20,000-step spin-up to reach statistical equilibrium, and each test pair is separated by a fixed gap of $\Delta T_{\text{gap}} = 72$ steps to ensure sufficient temporal decorrelation.

Algorithmic parameters The loss weight is $\gamma = 10^{-4}$. For Ours and Gold, the subspace scaling factor is $\alpha = k/\operatorname{tr}(T)$, where k is the subspace dimension; the evaluated LMP baseline uses $\alpha = 1$. The numerical inverse-eigenvalue guard uses $\epsilon = 10^{-5}$, independently of the PCG relative residual tolerance 10^{-5}.

Baseline methods We compare four preconditioning strategies across both systems. **None** denotes CG without preconditioning. **LMP** uses a historical limited-memory subspace (Tshimanga et al., 2008). **Ours** uses the predicted current-window subspace after QR. **Gold** uses current-window Lanczos vectors computed from the test system as an oracle reference; it is not a general lower bound on iteration counts. All experiments use 30 test samples for each system.

Table 1: Statistical performance comparison.

System	Method	Mean Iters	Std Iters	Cond. Num.
L96	None	17.17	1.07	3.95×10^2
	LMP	13.03	1.14	1.59×10^1
	Ours	7.87	0.91	3.86×10^1
	Gold	3.07	0.25	1.08×10^0
QG	None	461.90	7.46	2.90×10^4
	LMP	346.47	14.63	1.83×10^4
	Ours	268.48	5.61	5.34×10^3
	Gold	218.00	5.51	5.31×10^3

Evaluation protocol We use two evaluation protocols. First, we fix the relative residual tolerance for PCG at 10^{-5} and compare iteration counts. Second, we fix the number of PCG iterations and compare relative solution errors against a reference solve. Both protocols evaluate convergence of the PCG iterations on the same test systems.

4.2 Preconditioner Performance

We first evaluate performance using the total iteration counts required to reach a predefined tolerance for the 40-dimensional Lorenz-96 model and the 2048-dimensional QG model. We then compare performance under fixed total iteration budgets, examining how the methods scale as we increase the problem dimension.

Iteration count with fixed tolerance Table 1 reports the mean and standard deviation of iteration counts and a filtered spectral diagnostic over 30 tests. The code forms PS for preconditioned runs (and S for unpreconditioned runs), computes its eigenvalues, and reports the ratio of the largest to smallest retained real parts above 10^{-7}. In exact arithmetic, PS is similar to the symmetric SPD matrix $P^{1/2}SP^{1/2}$ when P and S are SPD. The filtering makes the reported ratio a diagnostic, not a certified condition number or a worst-case convergence guarantee: an eigenvalue excluded by the threshold would need separate investigation. Iteration counts also depend on the right-hand side.

For L96, Ours requires the fewest iterations among the practical methods tested. The average of 7.87 iterations is a 54.16% reduction relative to None and a 39.6% reduction relative to LMP. Its filtered spectral ratio is 3.86×10^1, compared with 3.95×10^2 for None and 1.59×10^1 for LMP. In particular, the filtered ratio is larger than LMP's even though the observed iteration count is lower; this diagnostic alone does not determine convergence for the tested right-hand sides.

For QG, Ours reduces the average iteration count from 461.90 (None) and 346.47 (LMP) to 268.48, reductions of 41.9% and 22.5%, respectively. The filtered spectral ratio decreases from 2.90×10^4 for None to 5.34×10^3, close to the Gold reference of 5.31×10^3. These results show fewer PCG iterations on the two evaluated systems.

Relative solution error with fixed iteration count In this evaluation, all methods receive the same PCG iteration budget within each setting. This comparison isolates the effect of the preconditioner on convergence at a prescribed iteration count. Table 2 and 3 report the mean relative error (MRE) and standard deviation under these budgets.

Both tables show iteration-count benefits in the tested settings. Within each reported fixed iteration budget, Ours achieves the lowest solution error among the practical methods compared. Figure 1 presents residual decay curves for the Lorenz-96 and QG models across dimensions; the vertical dashed line marks the iteration budget used in Table 2 and 3. The profiles agree with the corresponding iteration-count comparisons.

Table 2: Mean relative error (MRE) and standard deviation under fixed iteration (FI) budgets for L96 systems.

D	Method	K	FI	MRE	Std
80	None	–	10	4.59×10^{-2}	1.50×10^{-2}
	LMP	30	10	4.89×10^{-3}	6.53×10^{-3}
	Ours	30	10	5.07×10^{-6}	5.48×10^{-6}
	Gold	30	10	2.15×10^{-6}	2.66×10^{-6}
160	None	–	15	8.62×10^{-3}	3.42×10^{-3}
	LMP	40	15	2.63×10^{-3}	5.26×10^{-3}
	Ours	40	15	4.17×10^{-6}	2.36×10^{-6}
	Gold	40	15	3.16×10^{-6}	2.14×10^{-6}
1000	None	–	18	3.71×10^{-4}	2.18×10^{-4}
	LMP	100	18	2.44×10^{-3}	2.60×10^{-3}
	Ours	100	18	9.10×10^{-6}	1.19×10^{-5}
	Gold	100	18	3.60×10^{-6}	1.31×10^{-6}

Table 3: Mean relative error (MRE) and standard deviation (Std.) under fixed iteration (FI) budgets for QG systems.

D	Method	K	FI	MRE	Std.
2048	None	-	250	5.41×10^{-4}	6.03×10^{-5}
	LMP	45	250	2.30×10^{-5}	1.04×10^{-5}
	Ours	45	250	4.84×10^{-6}	9.05×10^{-7}
	Gold	45	250	2.73×10^{-7}	7.53×10^{-8}
4608	None	-	260	8.34×10^{-4}	9.77×10^{-5}
	LMP	60	260	1.10×10^{-4}	5.74×10^{-5}
	Ours	60	260	3.18×10^{-6}	4.54×10^{-7}
	Gold	60	260	3.25×10^{-7}	5.80×10^{-8}
8192	None	-	280	6.50×10^{-4}	1.46×10^{-4}
	LMP	80	280	2.00×10^{-4}	1.15×10^{-4}
	Ours	80	280	3.83×10^{-6}	1.30×10^{-6}
	Gold	80	280	1.28×10^{-7}	4.94×10^{-8}

4.3 Generalization and Robustness

4.3.1 Variable Assimilation Window

We evaluate the convergence behavior under different assimilation window lengths for both systems. Table 4 reports the average iteration counts for L96 with window lengths ranging from 5 to 18 steps, and for QG with window lengths ranging from 12 to 60 steps.

For L96, LMP provides acceleration only in short quasi-linear windows; its performance collapses as nonlinearity strengthens with longer windows, with iterations growing from 7.70 to 20.27—eventually matching unpreconditioned CG. In contrast, Ours remains robust, increasing only modestly from 6.80 to 9.97 across the same range. For QG, LMP degrades substantially from 81.77 to 461.53 iterations as the window length extends, while Ours maintains approximately 25% acceleration over LMP across all settings, increasing from 75.00 to 351.47. These results confirm that the proposed framework effectively handles both strongly nonlinear dynamics and long assimilation windows where traditional subspace reuse fails.

4.3.2 Cross-Dynamical State Generalization

We conduct two out-of-distribution generalization tests without retraining. For L96, the model trained under $F = 8.0$ is tested on unseen dynamical regimes with $F = 6.0, 10.0, 12.0$. As shown in Figure 2(a), Ours exhibits only a mild increase from 6.30 to 9.30 iterations across these regimes, while LMP degrades severely from 7.53 to 16.93 iterations. For QG, the model trained with a fixed

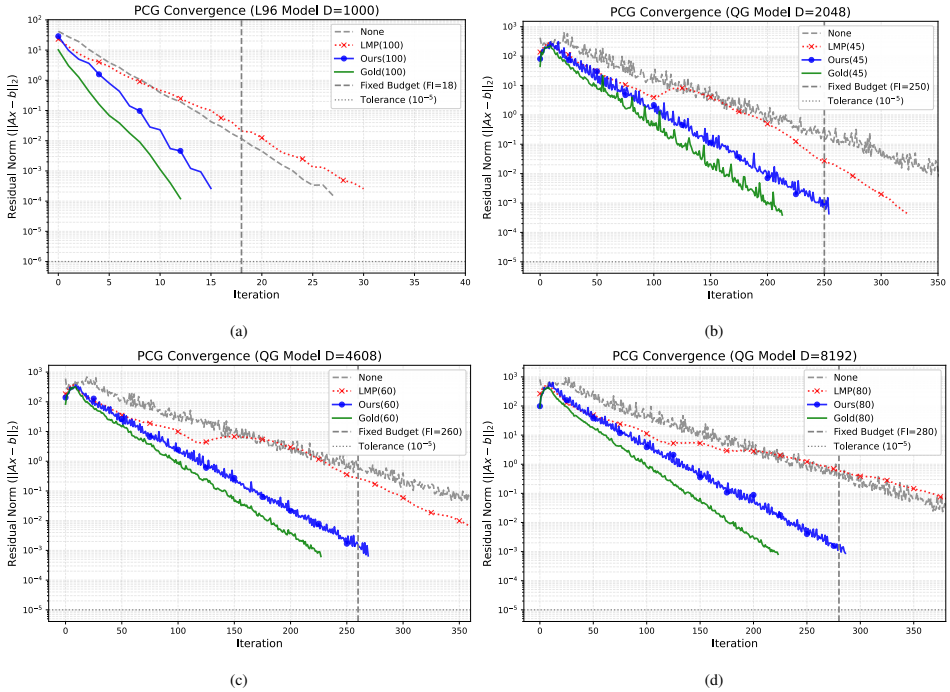

Figure 1: Convergence comparison of different methods for 4DVar systems. (a) Lorenz-96 model at $D = 1000$; (b) QG system at $D = 2048$; (c) QG system at $D = 4608$; (d) QG system at $D = 8192$.

Table 4: Iterations under different assimilation window lengths (wl).

L96				
wl	None	LMP	Ours	Gold
5	14.93	7.70	6.80	3.10
8	16.33	11.27	7.20	3.00
10	17.17	13.03	7.87	3.07
12	17.63	14.47	8.47	3.03
18	20.03	20.27	9.97	3.00

QG				
wl	None	LMP	Ours	Gold
12	110.77	81.77	75.00	61.93
24	216.63	155.07	138.53	111.40
36	336.83	240.60	205.57	162.67
48	461.90	346.47	268.47	218.00
60	594.07	461.53	351.47	276.37

72-step gap is tested on gaps ranging from 24 to 96 steps. As shown in Figure 2(b), LMP iterations rise from 279.37 to 378.47 as the gap increases, whereas Ours remains nearly constant between 266.63 and 269.27. This stability indicates that the network has learned the global manifold structure rather than memorizing fixed-step transition patterns.

4.4 Ablation Study

Table 5 reports ablation results where **w/o T-mat** removes spectral-energy weighting, **w/o dz** excludes historical physical increments, **w/o alpha** drops trace-based scaling, **CNN** replaces FNO with standard CNN, and **w/o res** removes residual connections (QG only).

For L96, removing T-mat increases iterations to 12.50, confirming that spectral-energy weighting is critical for prioritizing dominant modes. Excluding dz (11.80) shows that local curvature information helps mitigate nonlinear rotation errors. Dropping alpha (9.10) remains effective, suggesting a secondary role in this setting. The CNN variant fails catastrophically at 17.50—worse than unpreconditioned CG—demonstrating that conventional architectures lack sufficient global receptive fields.

For QG, the results reveal stronger dependencies on certain components. The w/o alpha variant causes severe numerical instability with iterations exploding to 695.90, indicating that trace-based

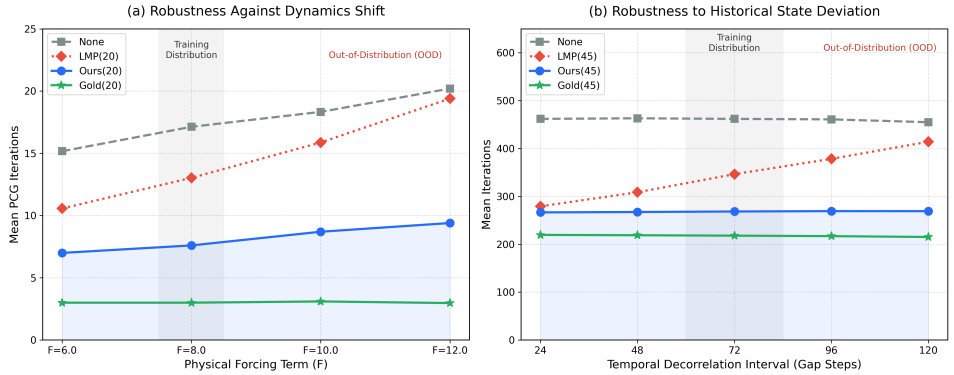

Figure 2: Robustness analysis under out-of-distribution conditions. (a) L96 iteration counts for varying forcing parameter F (model trained at $F = 8.0$); (b) QG iteration counts for varying temporal gaps between samples (model trained with 72-step gap).

Table 5: Ablation study results for L96 and QG systems.

Method	L96		QG	
	Mean Iters	Std Iters	Mean Iters	Std Iters
Ours full	7.87	0.91	268.47	5.61
w/o T-mat	12.50	1.57	273.00	4.94
w/o dz	11.80	0.83	269.57	5.43
w/o alpha	9.10	0.91	695.90	36.41
CNN	17.50	1.31	272.07	5.72
w/o res	–	–	286.80	5.17

scaling is essential for high-dimensional systems. Removing residual learning (w/o res, 286.80) confirms that incremental prediction effectively handles complex QG dynamics, while T-mat and dz show only mild degradation.

4.5 Limitations and Operational Potential

This study is an initial investigation of the advantages of spectral preconditioning. Constructing $T = Q^T SQ$ involves k applications of the current-window system operator, network inference, QR, and a small-matrix solve. At larger state dimensions, the storage and communication requirements of the $n \times k$ basis are relevant to implementation design. This study should be regarded as an idealized proof-of-concept investigation. The proposed framework has not yet been deployed within full-scale operational numerical weather prediction systems.

5 Conclusion

In this paper, we propose a Neural Spectral Preconditioning framework for accelerating the inner-loop optimization of variational data assimilation. By implicitly learning the evolution of the spectral subspace through a Fourier Neural Operator, the proposed method predicts the spectral structure at the current assimilation timestep using only offline historical by-products. Extensive experiments on the strongly nonlinear Lorenz-96 system demonstrate that the proposed framework reduces conjugate gradient iterations by 54.16% compared with unpreconditioned CG and by 39.6% compared with LMP, while maintaining strict numerical accuracy and stability. Validation on the high-dimensional QG system further confirms its effectiveness, achieving a 22.5% reduction over LMP and up to two orders of magnitude improvement in fixed-budget solution accuracy under more realistic dynamical conditions. As a proof-of-concept study in AI for Science, this work establishes a bridge between deep learning-based subspace prediction and mathematically rigorous iterative numerical solvers, providing a scalable, reliable, and computationally efficient acceleration framework for high-dimensional dynamical system state estimation.

AI USE STATEMENT

We used generative AI tools to assist with code implementation and debugging, interpretation and presentation of results, and translation. We also used them for brainstorming, literature discovery, and editing the manuscript for clarity. The authors reviewed AI-assisted text and code, checked references against their original sources, and ran and verified the reported experiments. The authors take full responsibility for the paper's content and claims.

ETHICS STATEMENT

This study uses synthetic data from numerical dynamical systems and involves no human participants, personal information, or sensitive datasets. We are not aware of any specific ethical concerns arising from the experiments.

REPRODUCIBILITY STATEMENT

Section 3 describes the method, and Section 4 provides the experimental settings, baselines, and evaluation protocol. The appendix specifies the background error covariance models and governing equations used in the experiments. The complete code, including data-generation and evaluation scripts, will be made available on GitHub upon acceptance.

REFERENCES

Rossella Arcucci, Jiangcheng Zhu, Shuang Hu, and Yi-Ke Guo. Deep data assimilation: integrating deep learning with data assimilation. *Applied Sciences*, 11(3):1114, 2021.

Ross N Bannister. A review of operational methods of variational and ensemble-variational data assimilation. *Quarterly Journal of the Royal Meteorological Society*, 143(703):607–633, 2017.

PHILIPPE Courtier, J-N Thépaut, and Anthony Hollingsworth. A strategy for operational implementation of 4d-var, using an incremental approach. *Quarterly Journal of the Royal Meteorological Society*, 120(519):1367–1387, 1994.

Alan Edelman, Tomás A Arias, and Steven T Smith. The geometry of algorithms with orthogonality constraints. *SIAM journal on Matrix Analysis and Applications*, 20(2):303–353, 1998.

M Fisher. Minimization algorithms for variational data assimilation. In *Proc. of ECMWF Seminar on Recent Development in Numerical Methods for Atmospheric Modelling*, 1998.

Stephen A Haben, Amos S Lawless, and Nancy K Nichols. Conditioning of incremental variational data assimilation, with application to the met office system. *Tellus A: Dynamic Meteorology and Oceanography*, 63(4):782–792, 2011.

Eugenia Kalnay. *Atmospheric modeling, data assimilation and predictability*. 2003.

Dmitrii Kochkov, Janni Yuval, Ian Langmore, Peter Norgaard, Jamie Smith, Griffin Mooers, Milan Klöwer, James Lottes, Stephan Rasp, Peter Düben, et al. Neural general circulation models for weather and climate. *Nature*, 632(8027):1060–1066, 2024.

Zhang Lin and Liu Yongzhu. The preconditioning of minimization algorithm in grapes global four-dimensional variational data assimilation system. *Journal of Applied Meteorological Science*, 28 (2):168–176, 2017.

AC Lorenc. A global three-dimensional multivariate statistical interpolation scheme. *Monthly Weather Review*, 109(4):701–721, 1981.

Jian Luo, Jie Wang, Hong Wang, Huanshuo Dong, Zijie Geng, Hanzhu Chen, and Yufei Kuang. Neural krylov iteration for accelerating linear system solving. *Advances in Neural Information Processing Systems*, 37:128636–128667, 2024.

Matthew J Martin, Ibrahim Hoteit, Laurent Bertino, and Andrew M Moore. Data assimilation schemes for ocean forecasting: state of the art. *State of the Planet Discussions*, 2024:1–16, 2024.

Hamidreza Moazzami, Asma Jamali, Nicholas Kevlahan, and Rodrigo A Vargas-Hernández. Meta-learning fourier neural operators for hessian inversion and enhanced variational data assimilation. *arXiv preprint arXiv:2509.22949*, 2025.

François Rozet and Gilles Louppe. Score-based data assimilation. *Advances in Neural Information Processing Systems*, 36:40521–40541, 2023.

Victor Trappler and Arthur Vidard. State-dependent preconditioning for the inner-loop in variational data assimilation using machine learning. *arXiv preprint arXiv:2501.04369*, 2025.

J Tshimanga, S Gratton, Anthony T Weaver, and A Sartenaer. Limited-memory preconditioners, with application to incremental four-dimensional variational data assimilation. *Quarterly Journal of the Royal Meteorological Society: A journal of the atmospheric sciences, applied meteorology and physical oceanography*, 134(632):751–769, 2008.

Xuguang Wang, Thomas M Hamill, Jeffrey S Whitaker, and Craig H Bishop. A comparison of hybrid ensemble transform kalman filter–optimum interpolation and ensemble square root filter analysis schemes. *Monthly weather review*, 135(3):1055–1076, 2007.