PSEUDO-DEEPONET: EXPLICIT NEURAL OPERATORS FOR PDE DYNAMICS

Anonymous authors
Paper under double-blind review

ABSTRACT

Neural operators for time-dependent partial differential equations are built in both implicit and explicit forms, yet rigorous guarantees that favor one design over the other for broad classes of problems have remained scarce. We introduce the *Pseudo-DeepONet* (PDON), an explicit architecture that separates spatial representation from temporal evolution in latent space, and we establish an error analysis showing that this split design can achieve strictly smaller approximation error than standard implicit operator formulations under mild regularity and spectral-decay conditions typical of many commonly studied dissipative and transport-dominated systems. Empirically, a parity-controlled inductive-bias study shows that the factorisation encourages learning the underlying dynamics rather than mere snapshot reconstruction when encoder, decoder, and capacity are matched. Among recurrent state-space temporal backbones, a linear oscillatory core (LinOSS) performs best in our suite, and we supply a theoretical error-bound analysis explaining how its second-order oscillatory structure matches temporal structure in PDE trajectories and lowers approximation error versus alternative cores. We further demonstrate resolution-invariant evaluation via band-limited spectral transfer between discretisations. Across seven widely used PDE benchmarks spanning one-, two-, and three-dimensional spatiotemporal dynamics, PDON mostly improves strong operator-learning baselines. Finally, we report results on the large-scale reanalysis dataset ERA5, showing that the same explicit factorisation carries to complex real-world spatiotemporal fields.

1 INTRODUCTION

Many science and engineering pipelines repeatedly solve parametric, time-dependent PDEs for new initial, boundary, or forcing data, as in digital-twin updating, Bayesian inversion, and optimal control (Bhattacharya et al., 2021; Kovachki et al., 2023). Classical finite-element and spectral solvers are accurate but expensive in this repeated-query regime, motivating neural operators that learn maps between infinite-dimensional function spaces (Chen & Chen, 1995; Lu et al., 2021; Li et al., 2021; Kovachki et al., 2023). For time-dependent PDEs, a key question is how the architecture represents temporal evolution. Two dominant operator-learning families handle time implicitly. DeepONet represents the solution operator as $\hat{u}(y; a) = \sum_k br_k(a)\, tr_k(y)$, where a is the input function and y is a space–time query point; thus time is treated as another trunk coordinate rather than as a sequential evolution variable (Lu et al., 2021). FNO applies learned spectral convolutions to gridded fields, and its standard time-dependent extension stacks time with the spatial axes, convolving over the joint space–time tensor (Li et al., 2021). Both therefore produce a monolithic map from input to output, treating space and time as interchangeable coordinates (Kontolati et al., 2024; Dummer et al., 2026; Cheng et al., 2025; Buitrago Ruiz et al., 2025; Michałowska et al., 2024; Hu et al., 2025). This has three costs: temporal dynamics are learned as static reconstruction rather than step-to-step evolution; complexity scales with the product of spatial and temporal grid sizes; and the representation is tied to the training discretisation, limiting transfer to unseen grids (Wang et al., 2021a; Lippe et al., 2023; Yarotsky, 2017; Lu et al., 2021).

We argue, and substantiate in three complementary ways, that a useful inductive bias for time-dependent PDE operators is an explicit factorisation that mirrors the dynamics: encode the physical field, evolve a low-dimensional latent state, and decode back to the field. We study this principle as Pseudo-DeepONet (PDON), a unifying framework that makes the spatial–temporal split explicit,

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.

supports controlled comparisons across temporal cores, and admits approximation and stability analysis. PDON shares DeepONet's goal of learning a solution operator from data, but replaces the branch–trunk dot product with a composition of sub-operators: a spatial encoder maps fields to latent coefficients, a temporal operator advances them, and a spatial decoder reconstructs the mesh. This gives a common language for factorised neural operators built from spatial and temporal components. Mori–Zwanzig theory motivates retaining temporal structure beyond a static map (Mori, 1965; Zwanzig, 1961), and recent memory neural operators develop the same theme in learned architectures (Buitrago Ruiz et al., 2025). In a parity-controlled advection example with matched encoders, decoders, and capacity, we show the explicit design learns transport dynamics rather than snapshot reconstruction and remains accurate under out-of-distribution shape shifts where a matched implicit baseline fails. We then prove an error-bound analysis showing that, under mild regularity assumptions and comparable budgets, explicit factorisation can achieve strictly smaller approximation error than an implicit joint formulation.

Beyond comparing explicit versus implicit operator factorisations, the design of the latent temporal operator is itself consequential. Many physically important time-dependent PDEs — wave propagation, advection-dominated transport, incompressible Navier–Stokes, and reaction-diffusion systems near oscillatory instabilities — produce trajectories whose temporal character is intrinsically wave-like: energy (or mass, or momentum) is exchanged between modes rather than dissipated, and the solution propagates phase information over long horizons (Acheson, 1990). The simplest lumped model of this behaviour is the harmonic oscillator of state z and transition matrix Λ as $z'' + \Lambda z = 0$, whose solutions preserve amplitude indefinitely; adding forcing v as $z'' = -\Lambda z + \mathbf{B}v$ captures externally driven oscillatory modes. A temporal neural operator built from stable discretisations of this second-order structure naturally encodes the phase and frequency content of such trajectories, avoids imposing artificial dissipation on the latent code, and can sustain memory of earlier state information over sequences where first-order recurrences tend to attenuate it. LinOSS (Rusch & Rus, 2025) instantiates exactly this design: it discretises the forced second-order oscillatory ODE into a parallelisable state-space model with a non-negative diagonal state matrix, guaranteeing stability without sacrificing expressiveness. When used as the temporal core inside PDON, LinOSS preserves energy-like invariants in the latent representation and maintains long-range temporal correlations, properties that are structurally aligned with the PDE families above. We support this viewpoint with a dedicated error-bound analysis showing why the second-order oscillatory temporal structure achieves lower approximation error than first-order recurrent cores under explicit stability and smoothness assumptions.

To summarise, our contributions are: (i) We formalise PDON as a unifying explicit encode–evolve–decode neural-operator factorisation, contrasted with joint spatio-temporal maps. With encoders, decoders, and budgets matched, and only latent time stepping versus time-as-coordinate varied, the factorisation better captures PDE dynamics and remains accurate under shape-shift out-of-distribution (OOD) settings where a matched implicit baseline degrades. (ii) We prove a complexity-separation theorem under mild regularity and spectral-decay assumptions: when the target operator admits an approximate spatial–temporal factorisation, a split PDON-style architecture achieves sharper approximation scaling than worst-case approximation over the broad joint Sobolev class. This isolates the cost of lacking the corresponding inductive bias, rather than a limitation in expressive power. (iii) We motivate oscillatory temporal cores alongside first-order recurrent alternatives, instantiate them within PDON, and provide an error-bound analysis showing when second-order oscillatory structure avoids the structural energy loss of contractive first-order cores. (iv) Because PDON composes neural operators on fields and latent trajectories rather than encoding a single fixed mesh, thus it supports resolution-robust prediction and transfer across discretisations. (v) Beyond the handful of three-four benchmarks typical in many operator-learning works, we evaluate on seven widely used PDE tasks spanning two-, three-, and four-dimensional spatiotemporal dynamics, and also on a complex real-world field dataset, and we ablate over several cores of PDON.

2 Related work

Implicit and monolithic neural operators. The dominant operator-learning architectures are implicit about time. Vanilla DeepONet (Lu et al., 2021; Chen & Chen, 1995) concatenates the time coordinate into the trunk and outputs the field via a branch-trunk inner product. FNO and its $(d{+}1)$-D variant (Li et al., 2021) stack global spectral convolutions on a joint (x, t) grid; LNO (Cao

et al., 2024) performs the same kernel idea in Laplace space; FFNO (Tran et al., 2023) factorises the FNO kernel along the d axes but still treats time as one of them. Transformer operators (Galerkin Transformer (Cao, 2021), GNOT (Hao et al., 2023), OFormer (Of-G, Of-S, Of-F) (Li et al., 2023a)) replace spectral convolution with attention, again with t in the token mixture. These models excel on short horizons and dense grids but inherit autoregressive spatial decay (Kontolati et al., 2024) and a curse-of-dimensionality lower bound on the joint (x, t) approximation problem (Yarotsky, 2017).

Explicit / latent factorised operators. A growing line of work explicitly decouples space from time. POD-DeepONet (Lu et al., 2022) replaces the trunk with a POD basis pre-computed from training snapshots; Latent DeepONet (L-DeepONet) (Kontolati et al., 2024) replaces the basis with a pre-trained autoencoder and a sequential branch; RONOM (Dummer et al., 2026) formalises this as "reduced-order neural operator modelling" with a Neural-ODE temporal core and proves resolution-transfer error bounds; MemNO (Buitrago Ruiz et al., 2025) pairs an FNO spatial block with an S4 (Gu et al., 2022) sequence model to recover Mori-Zwanzig memory; S-DeepONet (He et al., 2024) embeds a recurrent encoder inside the branch; Michałowska et al. (2024) sandwich an RNN between two DeepONets; DeepO-Mamba (Hu et al., 2025) combines a POD-DeepONet trunk with a Mamba temporal core. All of these are special cases of the PDON skeleton.

Temporal cores in operator learning. The temporal core inherits the rich literature of sequence models. Recurrent baselines (LSTM (Hochreiter & Schmidhuber, 1997), GRU (Cho et al., 2014)) are accurate but unparallelisable. Linear state-space models (S4 (Gu et al., 2022), S5 (Smith et al., 2023), LRU (Orvieto et al., 2023), LinOSS (Rusch & Rus, 2025)) discretise a continuous-time linear ODE and admit parallel scans, with LinOSS additionally enforcing energy preservation through orthonormal eigenvalues. Selective state-space models (Mamba (Gu & Dao, 2024)) make the SSM parameters input-dependent. Recent work (Cheng et al., 2025) systematically benchmarks these as drop-ins in operator-learning pipelines. A comprehensive discussion of these three lines of works and also additional approaches including PINN families (Raissi et al., 2019), FFNO, ViViT-style factorisation, attention based approaches such as GNOT, OFormer, and the resolution-invariance literature is provided in Appendix D.

3 Background

Neural Operator problem setting. Let $D_x \subset \mathbb{R}^{n_x}$ and $D_t = [0, T]$. We consider time-dependent PDEs of the general form $\partial_t u(x,t) = \mathcal{P}\big(u, \nabla_x u, \nabla_x^2 u, \ldots; a\big), \quad (x,t) \in D_x \times (0,T], \quad u(\cdot,0) = u_0, \quad \mathcal{B}(u) = 0$ on $\partial D_x \times D_t$, where $\mathcal{P}$ is the PDE evolution operator, $\mathcal{B}$ is the boundary operator, and a encode initial data, boundary data, coefficients, or forcing (Problem setting is fully detailed in Appendix B.1). The associated solution operator $G : X \to Y$ maps an input function $a \in X$ to the full spacetime trajectory $u = G(a) \in Y$. For computation, functions are sampled on N_x spatial nodes $\{x_k\}_{k=1}^{N_x}$ and N_t time levels $\{t_n\}_{n=0}^{N_t-1} \subset [0,T]$, taken uniformly with $t_n = n\Delta t$ and $\Delta t = T/(N_t - 1)$. In the sequence-to-sequence setting, this gives gridded input/output pairs $\mathbf{a} = \{a(x_k, t_n)\}_{k,n}$ and $\mathbf{u} = \{u(x_k, t_n)\}_{k,n}$, batched as $\mathbf{A} \in \mathbb{R}^{B\times N_x\times N_t\times C_{in}}$ and $\mathbf{U} \in \mathbb{R}^{B\times N_x\times N_t\times C}$, where B is the batch size and C_{in}, C are the input and output field channels. We omit grid arguments below for notational simplicity and write the learned operator as $\hat{u}_\theta = G_\theta(a) \approx G(a)$, trained by relative-$L^2$ error on the sampled trajectories.

State-space models and LinOSS. A linear state-space model (SSM) maps an input sequence $v(t)$ to an output $o(t)$ via a latent state $z(t) \in \mathbb{R}^{d_h}$ governed by the continuous-time system $z'(t) = \Lambda z(t) + \mathbf{B}v(t)$, $o(t) = \mathbf{C}z(t)$, where $\Lambda \in \mathbb{R}^{d_h\times d_h}$ is the typically diagonal state-transition matrix, $\mathbf{B}, \mathbf{C}$ are learned weight matrices, and stability requires the eigenvalues of Λ to have non-positive real parts. Discretising with step Δ yields $z_n = \bar{\Lambda} z_{n-1} + \bar{\mathbf{B}} v_n$; different SSMs (S4, S5, LRU, Mamba) differ mainly in how Λ is parameterised and how the recurrence is parallelised. LinOSS (Rusch & Rus, 2025) replaces the first-order ODE by the forced second-order oscillator $z''(t) = -\Lambda z(t) + \mathbf{B}v(t)$ with $\Lambda \geq 0$ diagonal, giving energy-preserving dynamics that naturally encode oscillatory and wave-like temporal patterns without imposing artificial dissipation on the latent state. Writing the second-order state in phase space as $s_n = (z_n, \dot{z}_n)^\top$, LinOSS-IMEX (referred to as LinOSS for simplicity) acts independently on each diagonal oscillator channel $i = 1, \ldots, d_h$. With learned stiffness $\Lambda_i \geq 0$ and step size $\Delta_i > 0$, its autonomous discrete update is $s_{n+1,i} = \Phi_i^{\mathrm{IMEX}} s_{n,i}$, where $\Phi_i^{\mathrm{IMEX}} = \left[\begin{smallmatrix} 1 & -\Delta_i\Lambda_i \\ \Delta_i & 1-\Delta_i^2\Lambda_i \end{smallmatrix}\right]$; if $0 < \Delta_i^2\Lambda_i < 4$, the eigenvalues of the discrete transition Φ_i^{IMEX} form

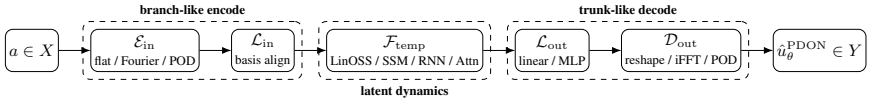


Figure 2: The PDON skeleton: an explicit factorisation $G_\theta^{\text{PDON}} = \mathcal{D}_{\text{out}} \circ \mathcal{L}_{\text{out}} \circ \mathcal{F}_{\text{temp}} \circ \mathcal{L}_{\text{in}} \circ \mathcal{E}_{\text{in}}$ mapping an input function $a \in X$ to a predicted trajectory $\hat{u}_\theta^{\text{PDON}} = G_\theta^{\text{PDON}}(a) \in Y$.

a complex-conjugate pair on the unit circle, giving stable oscillatory dynamics. Full background on LinOSS, operator learning, SSM variants, Galerkin and attention operators, and Mori–Zwanzig memory is collected in Appendix B. In the rest we use similar notation as in Background; a complete notation table is in Appendix A.

4 The Pseudo-DeepONet

4.1 Explicit Operator Factorisation

The Pseudo-DeepONet represents the learned operator G_θ^{PDON} through the explicit factorisation

$$G_\theta^{\text{PDON}} = \underbrace{\mathcal{D}_{\text{out}}}_{\text{spatial decoder}} \circ \underbrace{\mathcal{L}_{\text{out}}}_{\text{output projection } W_{\text{out}}} \circ \underbrace{\mathcal{F}_{\text{temp}}}_{\text{latent temporal core}} \circ \underbrace{\mathcal{L}_{\text{in}}}_{\text{input projection } W_{\text{in}}} \circ \underbrace{\mathcal{E}_{\text{in}}}_{\text{Proper Orthogonal Network}} \tag{1}$$

so that $\hat{u}_\theta^{\text{PDON}} = G_\theta^{\text{PDON}}(a) \approx G(a)$. The Proper Orthogonal Network $\mathcal{E}_{\text{in}}$ maps each spatial snapshot of the conditioning sequence a to rank-r_{POD} coefficients; the linear projector $W_{\text{in}} \in \mathbb{R}^{r_{\text{POD}} \times d_h}$ maps these coefficients to a latent sequence $z \in \mathbb{R}^{N_t \times d_h}$ aligned with the state space of $\mathcal{F}_{\text{temp}}$; $\mathcal{F}_{\text{temp}}$ evolves this latent sequence in $\mathbb{R}^{N_t \times d_h}$; $W_{\text{out}} \in \mathbb{R}^{d_h \times r_{\text{POD}}}$ produces output coefficients at each time step; and the spatial decoder $\mathcal{D}_{\text{out}}$ reconstructs the field on any required mesh. The overall PDON encode–evolve–decode factorisation is in Figure 2. The schematic with LinOSS instantiation is in Figure 7 Appendix.

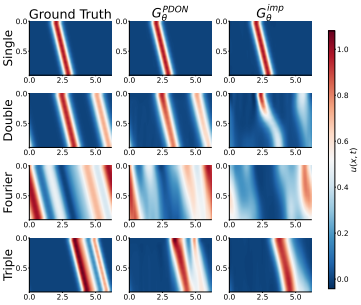


Figure 1: Inductive-bias study on advection: G_θ^{PDON} preserves transport structure under OOD shape shifts, while the implicit baseline degrades.

Vanilla DeepONet represents the baseline prediction as $G_\theta^{\text{DON}}(a)(y) = \sum_{k=1}^{p} br_k(a) tr_k(y)$, with the trunk br_k evaluated independently at every query point $y = (x, t)$; this requires $\mathcal{O}(N_x N_t)$ trunk evaluations per trajectory and forces the trunk to learn temporal evolution as a coordinate map. $(d+1)$-D FNO performs spectral convolution on the joint (x, t) grid and never separates a temporal state. In contrast, PDON (i) applies the spatial encoder only to spatial snapshots, rather than to the full space-time query grid, (ii) runs an explicit temporal operator in latent space whose hidden dimension $d_h \ll N_x C$, and (iii) decodes the resulting latent sequence back to the spatial field, eliminating autoregressive spatial decay.

Example 1 (Inductive-bias justification)**.** To isolate the effect of the factorisation from encoder/decoder capacity, we design a parity-controlled experiment on the 1-D advection equation $\partial_t u + c\,\partial_x u = 0$, whose exact solution is a pure shift $u(x, t) = f(x - ct)$. Our split (explicit) operator G_θ^{PDON} and a joint (implicit) baseline G_θ^{imp} share bit-for-bit the same residual-conv spatial encoder $\mathcal{E}_{\text{in}}$ and transposed-conv decoder $\mathcal{D}_{\text{out}}$, with parameter budgets matched within 2%. The *only* architectural difference is that G_θ^{PDON} evolves latent states by a learned temporal update $z_{n+1} = \mathcal{F}_{\text{temp}}^\theta(z_n)$ and decodes evolved steps afterwards, while G_θ^{imp} encodes the initial field via the same $\mathcal{E}_{\text{in}}$, then maps the resulting encoding together with a learned time embedding to a latent summary through a static MLP at each step independently, without recurrence. Both models are trained exclusively on single-Gaussian initial conditions (Figure 1 first row) and evaluated on the same distribution and on three shape-shift OOD families (double Gaussian, random Fourier, and triple Gaussian test families, shown in the second, third, and fourth rows of Figure 1, respectively). Despite identical spatial trunks, matched parameter counts, and comparable in-distribution reconstruction, G_θ^{imp} fails to learn

dynamics under OOD shape shifts, whereas G_θ^{PDON} attains 2.3–4.5$\times$ lower relative error on every OOD family. Full architecture details and numerical results related to Example 1 are in Appendix G.8 Figure 13. The gap observed in Example 1 reflects a fundamental difference in approximation complexity between joint and split operator architectures; we now make this precise in a theorem format (Proofs are detailed in Appendix C):

Assumptions. Let $G : X \to Y$ be the PDE solution operator over a fixed time horizon $[0, T]$ and $A : \mathbb{R}^m \to \mathbb{R}^p$ the latent approximator inside the generic decomposition $N = \mathcal{R} \circ A \circ \mathcal{E}$, where $m = m_x m_t$ is the joint sensor count; in PDON this instantiates as $\mathcal{E} = \mathcal{L}_{\mathrm{in}} \circ \mathcal{E}_{\mathrm{in}}$, $A = \mathcal{F}_{\mathrm{temp}}$, $\mathcal{R} = \mathcal{D}_{\mathrm{out}} \circ \mathcal{L}_{\mathrm{out}}$. We assume: **(A1)** G is L-Lipschitz on bounded balls in X; **(A2)** the input measure $\mu = (u(\cdot;\cdot))_{\#}\mathbb{P}_Y$ has a Karhunen–Loève expansion $u(x;Y) = \sum_k Y_k \alpha_k \phi_k(x)$ with $|\alpha_k| \le C_\alpha e^{-\ell|k|}$, $Y_k \sim U[-1,1]$ i.i.d., $\ell > 0$ (so $\mathrm{Tr}(\Gamma_\mu) < \infty$); **(A3)** the ideal latent operator factorises as $A^* = A_t^* \circ A_x^* + \delta$ with $\|\delta\|_{L^2(\mu)} \le C_\delta \mathrm{Tr}(\Gamma_\mu)$ and A_x^*, A_t^* are (b, ρ, κ)-holomorphic; **(A4)** the parametric map $G(u(\cdot; y))$ is (b, ρ, κ)-holomorphic that is standard ingredient in FNO/DeepONet approximation theory (Kovachki et al., 2024; De Ryck & Mishra, 2022) adapted to split setting. We let k denote the joint Sobolev regularity exponent of A^* (controlling Yarotsky's lower bound (Yarotsky, 2017)) and r the split regularity exponent of the factor maps A_x^*, A_t^* (controlling the accuracy exponent in the split upper bound). Theorem distinguishes the broad joint Sobolev class $\mathcal{F}_k(m)$ that any architecture must accommodate worst-case from the factorisable holomorphic subclass $\mathcal{F}_{r,\varepsilon_{\mathrm{floor}}}(m) \subsetneq \mathcal{F}_k(m)$ (Appendix C.1).

Theorem 1 (Complexity Separation; proof in Appendix C.5)**.** *Let G satisfy A1–A4, fix the discretisation (m_x, m_t) so that $m = m_x m_t$ is fixed throughout, let k, r be the joint and split regularity exponents, and set $\varepsilon_{\mathrm{floor}} := C_\delta \mathrm{Tr}(\Gamma_\mu)$. Then:*

1. *(*Joint minimax lower bound, over $\mathcal{F}_k(m)$*) For any ReLU architecture, worst-case approximation has complexity $W^{\mathrm{joint}}_{\mathcal{F}_k(m)}(\varepsilon) \ge c(m,k)\, \varepsilon^{-m/k}$.*
2. *(*Split upper bound, over $\mathcal{F}_{r,\varepsilon_{\mathrm{floor}}}(m)$*) For $\varepsilon > \varepsilon_{\mathrm{floor}}$, there exists a split approximator with $W^{\mathrm{split}}(\varepsilon) \le C_r(m_x + d_h + p)\log(m)\log\log(m)\,(\varepsilon - \varepsilon_{\mathrm{floor}})^{-1/r}$, where m_x, d_h, p are fixed architecture constants.*
3. *(*Conditional asymptotic separation*) When $mr > k$ and $\varepsilon_{\mathrm{floor}} = o(\varepsilon)$, the efficiency ratio satisfies*

$$\frac{W^{\mathrm{joint}}_{\mathcal{F}_k(m)}(\varepsilon)}{W^{\mathrm{split}}_{\mathcal{F}_{r,\varepsilon_{\mathrm{floor}}}(m)}(\varepsilon)} \gtrsim \frac{c(m,k)}{C_r(m_x + d_h + p)\log(m)\log\log(m)}\, \varepsilon^{-(m/k - 1/r)} \xrightarrow[\varepsilon \to 0^+]{} \infty,$$

Proof sketch. The lower bound is the metric-entropy argument of Yarotsky (2017) applied in the joint dimension m, stated as a minimax bound over $\mathcal{F}_k(m)$ that holds for *any* ReLU architecture of size W. The upper bound builds the split approximator from Cohen–Schwab sparse-polynomial-chaos surrogates (Schwab & Zech, 2019) of A_x^* and A_t^* parametrised by an internal accuracy η *decoupled* from the fixed input dimension m_x; choosing $\eta \asymp \varepsilon - \varepsilon_{\mathrm{floor}}$ delivers $L^2(\mu)$ error ε once the residual δ is controlled via Lemma C.2. There δ is the time-integrated nonlinear vector field along trajectories, and a Duhamel formula followed by a Cauchy–Schwarz/Gronwall step (Gronwall, 1919) bounds it by $\mathrm{Tr}(\Gamma_\mu)$ up to a factor $T^2 e^{6LT}$ absorbed into C_δ. The asymptotic ratio uses $m/k > 1/r$ at fixed m, with crossover $\varepsilon^* = \left[\frac{c}{C_r(m_x+d_h+p)\log m}\right]^{rk/(mr-k)} \in (0,1)$ when $C_r(m_x + d_h + p)\log m \gg c$. See Appendix C.6 for the full derivation, including the degenerate $mr = k$ and entangled $mr < k$ regimes.

Interpretation. The *joint* (implicit) approximator pays a curse-of-dimensionality price in the product dimension $m = m_x m_t$, as it must accommodate the worst function in the broad class $\mathcal{F}_k(m)$; the *split* (explicit) architecture in PDON restricts to the factorisable subclass $\mathcal{F}_{r,\varepsilon_{\mathrm{floor}}}(m)$ and pays only the larger of m_x and d_h, at the cost of an irreducible floor $\varepsilon_{\mathrm{floor}}$ proportional to the spectral trace of the input measure. This is a separation of *adaptive efficiency under inductive bias*, not of expressive power, and $\mathcal{F}_{r,\varepsilon_{\mathrm{floor}}}(m)$ covers virtually all PDE families of practical interest (Appendix C.7; failure regimes are collected in Appendix K). Even at matched parameter budget, the explicit factorisation generalises better because the optimisation lives in a strictly smaller hypothesis class hardwired into the architecture.

4.2 Temporal Core: Linear Oscillatory Operators

Having established *why* explicit factorisation is useful, the remaining design choice is the temporal core $\mathcal{F}_{\text{temp}}$. Many PDEs of interest—wave propagation, advection-dominated transport, incompressible Navier–Stokes near oscillatory regimes, and weakly dissipative reaction-diffusion—exhibit persistent phase information, recurrent structure, or weakly damped modal interactions. Thus the latent dynamics should transport information over long horizons without artificially damping oscillatory modes. A strictly stable first-order SSM, $z'(t) = \Lambda z(t) + \mathbf{B}v(t)$, with $\mathrm{Re}(\lambda_i(\Lambda)) < 0$, yields after discretisation $z_n = \bar{\Lambda} z_{n-1} + \bar{\mathbf{B}} v_n, \qquad \rho(\bar{\Lambda}) < 1$, so earlier information is progressively attenuated. This mismatches operators whose long-horizon behavior depends on preserving phase and amplitude. LinOSS (Rusch & Rus, 2025) addresses this by replacing first-order dynamics with the forced second-order oscillator

$$z''(t) = -\Lambda z(t) + \mathbf{B}v(t), \qquad \Lambda = \mathrm{diag}(\lambda_1, \ldots, \lambda_{d_h}), \quad \lambda_i \geq 0. \tag{2}$$

This gives LinOSS an oscillatory inductive bias: the phase-space state $(z(t), z'(t))$ can remain bounded without enforcing exponential decay, while the model still admits efficient parallel scan evaluation and universal approximation guarantees for continuous causal time-series operators. These properties are structurally aligned with oscillatory and weakly dissipative PDE solution operators. Preserving phase and amplitude in the latent code avoids the artificial damping that can cause strictly stable first-order recurrences to lose early-time information over long horizons. We formalise this contrast in the following theorem: for energy-preserving latent dynamics, a power-bounded LinOSS transition controls trajectory and energy error through the learned parameter mismatch, whereas contractive first-order cores incur an irreducible structural energy error from dissipation. All proofs are in Appendix C.8.

Theorem 2 (Temporal Approximation Bounds: Parametric vs. Structural Error)**.** *Let s_n and $\hat{s}_n$ denote the autonomous latent trajectories governed by the true physical operator Ψ and the learned temporal core $\hat{\Psi}$, respectively, with initial state alignment $s_0 = \hat{s}_0$ and $\|s_0\|_2 = C_0 > 0$. Suppose the regime of interest is energy-preserving in the latent representation, i.e. $\|s_n\|_2 = C_0$ for all $n \geq 0$. Let $e_n := s_n - \hat{s}_n$ and $E_n := \big|\|s_n\|_2 - \|\hat{s}_n\|_2\big|$. Assume the discrete transition matrices satisfy the local Lipschitz bound $\|M(\Psi) - M(\hat{\Psi})\|_2 \leq L_M \|\Psi - \hat{\Psi}\|_2$. Then for $n \geq 0$:*

1. ***LinOSS (parametric error only).*** *Let $\hat{\Psi} = \hat{\Lambda}$. Under the assumptions of Appendix Lemma 4, suppose the LinOSS transition is uniformly power-bounded: $\|M(\Psi)^j\|_2,\ \|M(\hat{\Psi})^j\|_2 \leq C_{\text{IMEX}}$ for all $j \geq 0$. Then the trajectory error is controlled* entirely *by the learned parametric mismatch and grows at most linearly with horizon:*

$$\|e_n\|_2 \ \leq\ C_{\text{IMEX}}^2\, n\, L_M\, C_0\, \|\Psi - \hat{\Psi}\|_2, \tag{3}$$

   and consequently $E_n \leq \|e_n\|_2$. In particular, the bound vanishes as $\hat{\Psi} \to \Psi$: any target accuracy is achievable by parameter fitting alone.

2. ***Contractive cores (irreducible structural error).*** *Let $M(\hat{\Psi})$ be* any *linear recurrent core whose transition admits a uniform contraction $\|M(\hat{\Psi})^k\|_2 \leq C_{\text{diss}}\rho^k$ with $0 < \rho < 1$,* regardless of the choice of $\hat{\Psi}$. *Then the energy error is bounded below by a quantity that depends only on the structural radius ρ and is independent of how $\hat{\Psi}$ is fit:*

$$\inf_{\hat{\Psi} \in \mathcal{C}_\rho} E_n \ \geq\ C_0\big(1 - C_{\text{diss}}\rho^n\big)_+, \tag{4}$$

   where $\mathcal{C}_\rho$ is the class of cores satisfying the contraction bound above. As $n \to \infty$, $\inf_{\hat{\Psi}} E_n \to C_0$: no parameter setting in $\mathcal{C}_\rho$ can recover the energy-preserving reference.

Training. We train all models end-to-end by minimizing the relative trajectory error between the predicted and target fields on the sampled space–time grid: $\min_\theta\ \mathbb{E}_{(\mathbf{A},\mathbf{U})}\left[\frac{\|G_\theta^{\text{PDON}}(\mathbf{A}) - \mathbf{U}\|_2}{\|\mathbf{U}\|_2}\right]$.

5 Experiments

The experiments are organised into four blocks: (5.1) ablations isolating the components of the explicit factorisation, scalability and oscillatory design's effect on memory/energy; (5.2) headline

benchmarks against operator-learning baselines on seven PDE suites; and results on the ERA5 reanalysis dataset; and (5.3) resolution-transfer between discretisations. Each suite is referred to by name; the governing equations, initial-and-boundary conditions, forcing constructions, and operator input–output structures are recorded in Appendix F. Detailed architectural configuration along with parameter settings, detailed hyperparameter optimization, hardware setting and data flow (involving dimensions for reproducibility and ease of read) are in Appendix G. Intuitive python code and training-inference algorithms of PDON are in Appendix H. Full theoretical complexity analysis with $\mathcal{O}(.)$ and empirical runtime clock on the wall with parameter counts are in Appendix E and I.2. A one-factor-at-a-time hyperparameter sensitivity study is reported in Appendix J.

5.1 Ablation

Explicit PDON versus explicit FNO. Both PDON and the FNO$(d{+}1)$-LinOSS baseline use an explicit space–time split: a spatial operator first represents each input snapshot, and a LinOSS temporal core then evolves the resulting sequence. They differ in the spatial representation: PDON projects snapshots into a compact POD-aligned latent coefficient space before temporal evolution, whereas the FNO baseline keeps Fourier-feature field representation. At matched parameter budget across seven benchmarks spanning (1+1)D, (2+1)D, and (3+1)D dynamics, PDON achieves lower relative-L^2 error in every case, as shown in Table 1. The advantage suggests that, once time is handled explicitly, a compact coefficient-space evolution is often easier to learn and less prone to unnecessary spatial–temporal entanglement than evolving high-dimensional Fourier features.

Table 1: Relative-L^2 error: PDON-LinOSS vs. FNO$(d{+}1)$-LinOSS.

Suite	PDON-LinOSS	FNO$(d{+}1)$-LinOSS
Burgers (1+1)	$1.28{\cdot}10^{-3}$	$2.76{\cdot}10^{-3}$
RD-ft (1+1)	$1.44{\cdot}10^{-3}$	$2.36{\cdot}10^{-2}$
RD-ftx (1+1)	$1.07{\cdot}10^{-2}$	$4.24{\cdot}10^{-2}$
fKdV (1+1)	$1.14{\cdot}10^{-2}$	$2.45{\cdot}10^{-2}$
Fourier RD (2+1)	$2.25{\cdot}10^{-3}$	$7.69{\cdot}10^{-3}$
Brusselator (2+1)	$1.78{\cdot}10^{-3}$	$7.09{\cdot}10^{-2}$
Beltrami (3+1)	$1.20{\cdot}10^{-3}$	$1.50{\cdot}10^{-2}$

Scalability on longer sequences. We next test whether the learned temporal core in the PDON setting remains stable when evaluated far beyond the sequence lengths used during training. For Burgers, models are trained at $N_t = 100$ and evaluated at $N_t = 1000, 2000, 4000$; for the Navier–Stokes Beltrami flow, referred to as Beltrami in the rest, models are trained at $N_t = 20$ and evaluated at rollout horizons $T = 40$ and $T = 80$. Table 2 shows that PDON-LinOSS maintains low relative-L^2 error as the horizon increases, whereas recurrent baselines degrade and Of-G diverges on the Beltrami rollouts. This supports the intended scalability of the explicit factorisation: spatial encoding and decoding are unchanged, while only the compact latent temporal sequence is extended.

Table 2: Scalability: relative-L^2 at increasing sequence lengths (Burgers) and rollout horizons (Beltrami).

Model	*Burgers (1+1)D*			*Beltrami (3+1)D*	
	$N_t = 1000$	$N_t = 2000$	$N_t = 4000$	$T = 40$	$T = 80$
GRU	$2.09{\cdot}10^{-3}$	$1.36{\cdot}10^{-2}$	$5.09{\cdot}10^{-2}$	$2.30{\cdot}10^{-1}$	$1.66{\cdot}10^{-1}$
LSTM	$2.71{\cdot}10^{-3}$	$1.33{\cdot}10^{-2}$	$3.47{\cdot}10^{-2}$	$1.29{\cdot}10^{-1}$	$1.24{\cdot}10^{-1}$
Mamba	$2.21{\cdot}10^{-3}$	$6.12{\cdot}10^{-3}$	$9.35{\cdot}10^{-3}$	$9.50{\cdot}10^{-4}$	$8.98{\cdot}10^{-4}$
Of-G	$1.47{\cdot}10^{-3}$	$8.52{\cdot}10^{-3}$	$9.79{\cdot}10^{-3}$	div.	div.
LinOSS	$\mathbf{6.2{\cdot}10^{-4}}$	$\mathbf{1.49{\cdot}10^{-3}}$	$\mathbf{3.97{\cdot}10^{-3}}$	$\mathbf{4.0{\cdot}10^{-4}}$	$\mathbf{3.8{\cdot}10^{-4}}$

Effect of oscillatory memory. Holding the spatial encoder–decoder pair fixed, we compare LinOSS with the strongest first-order SSM alternatives in our sweep, S5 and LRU. As shown in Figure 3, we diagnose their latent temporal dynamics using learned transition spectra and autonomous impulse-response energy. For LinOSS, each latent channel $i = 1, ..., d_h$ has phase-space state $s_{n,i} = (z_{n,i}, \dot{z}_{n,i})^\top$, (here n denotes the discrete rollout index), and one-step transition Φ_i; in the stable regime $0 < \Delta_i^2 \Lambda_i < 4$, the eigenvalues of Φ_i lie on the unit circle. Since Φ_i is symplectic rather than Euclidean orthogonal, we measure memory by the normalized Hamiltonian-energy response $M_E(n) = \left(\frac{\mathbb{E}_\delta \sum_i s_{n,i}^\top Q_i s_{n,i}}{\mathbb{E}_\delta \sum_i s_{0,i}^\top Q_i s_{0,i}} \right)^{1/2}$, $\Phi_i^\top Q_i \Phi_i = Q_i$. where Q_i is the positive-definite quadratic energy matrix preserved by channel i, and the expectation is over energy-normalised random impulses δ injected at the initial latent phase-space state. LinOSS keeps $M_E(n) \approx 1$ over autonomous

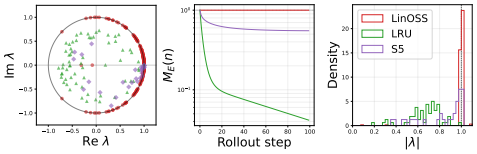


Figure 3: Oscillatory-memory diagnostic on KdV.

Table 3: Relative-L^2 across seven benchmark suites. **Bold**: best per column; underlined: second best; "div." = diverged; "—" = OOM.

Model	Burg.	RD-ft	RD-ftx	fKdV	F-RD	Brus.	Belt.
PDON (SSM cores)							
Mamba	$8.47{\cdot}10^{-3}$	$4.54{\cdot}10^{-3}$	$1.47{\cdot}10^{-2}$	$2.88{\cdot}10^{-2}$	$4.41{\cdot}10^{-2}$	$1.21{\cdot}10^{-2}$	$7.40{\cdot}10^{-2}$
S5	$1.78{\cdot}10^{-3}$	$2.78{\cdot}10^{-3}$	$1.48{\cdot}10^{-2}$	$2.02{\cdot}10^{-2}$	$2.35{\cdot}10^{-2}$	$4.24{\cdot}10^{-3}$	$6.72{\cdot}10^{-2}$
LRU	$3.40{\cdot}10^{-3}$	$9.15{\cdot}10^{-3}$	$2.67{\cdot}10^{-2}$	$2.54{\cdot}10^{-2}$	$2.59{\cdot}10^{-2}$	$5.72{\cdot}10^{-2}$	$7.65{\cdot}10^{-2}$
LinOSS(Ours)	$\underline{1.28{\cdot}10^{-3}}$	$\mathbf{1.44{\cdot}10^{-3}}$	$\mathbf{1.07{\cdot}10^{-2}}$	$\underline{1.14{\cdot}10^{-2}}$	$\mathbf{2.25{\cdot}10^{-3}}$	$\mathbf{1.78{\cdot}10^{-3}}$	$\mathbf{1.20{\cdot}10^{-3}}$
PDON (attention)							
GNOT	$1.48{\cdot}10^{-3}$	$3.47{\cdot}10^{-1}$	$5.74{\cdot}10^{-1}$	$1.20{\cdot}10^{-2}$	$3.50{\cdot}10^{-3}$	$2.83{\cdot}10^{-2}$	div.
Of-G	$4.33{\cdot}10^{-3}$	$1.25{\cdot}10^{-1}$	$4.16{\cdot}10^{-1}$	$2.76{\cdot}10^{-2}$	$\underline{2.54{\cdot}10^{-3}}$	$6.00{\cdot}10^{-2}$	div.
Of-V	$2.79{\cdot}10^{-3}$	$2.08{\cdot}10^{-1}$	$5.24{\cdot}10^{-1}$	$2.54{\cdot}10^{-2}$	$5.48{\cdot}10^{-3}$	—	div.
Of-F	$4.91{\cdot}10^{-3}$	$1.21{\cdot}10^{-1}$	$4.44{\cdot}10^{-1}$	$2.68{\cdot}10^{-2}$	$2.76{\cdot}10^{-3}$	$5.47{\cdot}10^{-2}$	div.
PDON (recurrent)							
GRU	$1.97{\cdot}10^{-3}$	$1.16{\cdot}10^{-2}$	$1.71{\cdot}10^{-2}$	$1.38{\cdot}10^{-2}$	$3.40{\cdot}10^{-2}$	$1.70{\cdot}10^{-2}$	$1.61{\cdot}10^{-1}$
LSTM	$2.19{\cdot}10^{-3}$	$1.08{\cdot}10^{-2}$	$3.24{\cdot}10^{-2}$	$\mathbf{1.05{\cdot}10^{-2}}$	$3.42{\cdot}10^{-2}$	$1.50{\cdot}10^{-2}$	$1.25{\cdot}10^{-1}$
FNO family							
FNO (implicit)	$2.76{\cdot}10^{-3}$	$2.36{\cdot}10^{-2}$	$4.22{\cdot}10^{-2}$	$2.45{\cdot}10^{-2}$	$7.68{\cdot}10^{-3}$	—	$1.50{\cdot}10^{-2}$
FFNO (implicit)	$1.32{\cdot}10^{-3}$	$3.31{\cdot}10^{-2}$	$2.61{\cdot}10^{-2}$	$1.31{\cdot}10^{-2}$	$8.24{\cdot}10^{-3}$	—	$1.38{\cdot}10^{-2}$
FFNO(d+1)	$\mathbf{1.13{\cdot}10^{-3}}$	$3.23{\cdot}10^{-2}$	$2.47{\cdot}10^{-2}$	$1.27{\cdot}10^{-2}$	$7.75{\cdot}10^{-3}$	—	$1.26{\cdot}10^{-2}$
PINNs							
PINN	$1.60{\cdot}10^{-2}$	$5.39{\cdot}10^{-2}$	$9.22{\cdot}10^{-2}$	$1.08{\cdot}10^{-1}$	$9.98{\cdot}10^{-3}$	$1.67{\cdot}10^{-2}$	$2.27{\cdot}10^{-2}$
Causal PINN	$2.63{\cdot}10^{-3}$	$3.51{\cdot}10^{-2}$	$4.19{\cdot}10^{-2}$	$2.97{\cdot}10^{-2}$	$6.12{\cdot}10^{-3}$	$3.54{\cdot}10^{-2}$	$3.63{\cdot}10^{-2}$
PINN-RNN (AAAI'24)	$4.44{\cdot}10^{-3}$	$7.52{\cdot}10^{-3}$	$3.12{\cdot}10^{-3}$	$2.86{\cdot}10^{-2}$	$6.15{\cdot}10^{-3}$	$8.57{\cdot}10^{-3}$	$1.19{\cdot}10^{-2}$
Common NOs							
DeepONet	$7.33{\cdot}10^{-3}$	$8.45{\cdot}10^{-2}$	$4.13{\cdot}10^{-2}$	$5.19{\cdot}10^{-2}$	$4.96{\cdot}10^{-2}$	$2.13{\cdot}10^{-2}$	$8.43{\cdot}10^{-2}$
LNO	$4.63{\cdot}10^{-3}$	$3.51{\cdot}10^{-2}$	$4.19{\cdot}10^{-2}$	$4.97{\cdot}10^{-2}$	$6.12{\cdot}10^{-2}$	$3.54{\cdot}10^{-2}$	$3.63{\cdot}10^{-2}$
SOTA baselines							
MaskPDE (TMLR'24)	$1.52{\cdot}10^{-3}$	$1.73{\cdot}10^{-3}$	$2.33{\cdot}10^{-2}$	$2.29{\cdot}10^{-2}$	$3.20{\cdot}10^{-2}$	$\underline{4.11{\cdot}10^{-3}}$	$8.14{\cdot}10^{-3}$
AROMA (NeurIPS'24)	$3.65{\cdot}10^{-2}$	$2.17{\cdot}10^{-2}$	$2.26{\cdot}10^{-2}$	$5.21{\cdot}10^{-2}$	$6.23{\cdot}10^{-2}$	$8.13{\cdot}10^{-3}$	$1.05{\cdot}10^{-1}$
MemNO (ICLR'25)	$6.24{\cdot}10^{-3}$	$5.15{\cdot}10^{-3}$	$\underline{1.30{\cdot}10^{-2}}$	$3.25{\cdot}10^{-2}$	$6.14{\cdot}10^{-2}$	$7.46{\cdot}10^{-3}$	$8.84{\cdot}10^{-3}$
SBAL (ICML'25)	$4.25{\cdot}10^{-3}$	$4.12{\cdot}10^{-3}$	$2.29{\cdot}10^{-2}$	$2.27{\cdot}10^{-2}$	$3.25{\cdot}10^{-2}$	$6.02{\cdot}10^{-3}$	$\underline{3.22{\cdot}10^{-3}}$
LaMO (ICML'25)	$5.12{\cdot}10^{-3}$	$1.94{\cdot}10^{-3}$	$1.35{\cdot}10^{-2}$	$1.40{\cdot}10^{-2}$	$2.27{\cdot}10^{-2}$	$8.25{\cdot}10^{-3}$	$4.20{\cdot}10^{-2}$
RNO (ICLR'26)	$1.89{\cdot}10^{-3}$	$\underline{1.61{\cdot}10^{-3}}$	$3.06{\cdot}10^{-2}$	$1.17{\cdot}10^{-2}$	$4.50{\cdot}10^{-3}$	$9.24{\cdot}10^{-3}$	$8.44{\cdot}10^{-3}$

rollout, whereas S5 and LRU decay according to their learned eigenvalue radii inside the unit disk. This links LinOSS's long-horizon gains to bounded oscillatory memory and Hamiltonian-energy preservation in latent space, consistent with analysis in Theorem 2 (see Appendix I.1 for Brusselator counterpart and additional details).

5.2 Benchmark

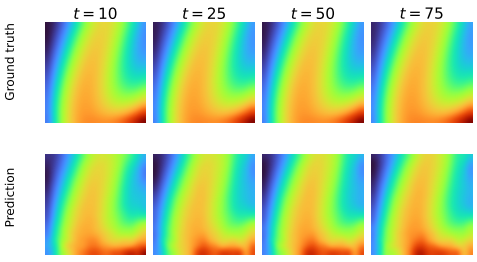


Figure 4: Prediction $\hat{u}_\theta$ versus ground truth for the Beltrami flow at four time steps, $t \in \{10, 25, 50, 75\}$.

We evaluate PDON with LinOSS as the default temporal core against strong operator-learning baselines, including FNO (Li et al., 2021), FFNO (Tran et al., 2023), GNOT (Hao et al., 2023), and Galerkin Transformer (Cao, 2021); physics-informed baselines from a different training paradigm, including PINN (Raissi et al., 2019), causal PINN (Kapoor et al., 2024b), and recurrent PINN (Kapoor et al., 2024a); and recent SOTA baselines including MaskPDE (Zhou & Farimani, 2024), AROMA (Serrano et al., 2024), MemNO (Buitrago Ruiz et al., 2025), SBAL (Kim et al., 2025), LaMO (Tiwari et al., 2025) and most recently RNO (Liu et al., 2026); among them LaMO is most similar approach to ours that PDON consistently outperforms it across all seven suites. The metric throughout is relative-L^2 error on held-out trajectories. For the FNO-family baselines, we include both explicit variants and implicit variants (for further ablation), FNO(implicit) and FFNO(implicit), which concatenate time with the spatial coordinates and learn a joint space-time map. The first four columns of Table 3 cover four (1+1)D PDE families: forced Burgers (Appendix F.1), forced KdV (Appendix F.2), and the two reaction–diffusion configurations RD-ft and RD-ftx (Appendix F.3). The last three columns cover harder spatiotemporal regimes: the (2+1)D periodic Fourier reaction–diffusion suite (Appendix F.4), the

long-horizon Brusselator (Appendix F.5), and the (3+1)D incompressible Navier–Stokes Beltrami flow (Appendix F.6)(Visualized in Figure 4). Here PDON-LinOSS gives a strong overall performance, with especially large gains on Brusselator and Beltrami. Standard deviations over several random seeds per experiment are reported in Appendix Table 6, with additional visualizations in Appendix I.3. Animated trajectory results are available at: sites.google.com/view/pdon-iclr or our anonymous repo.

ERA5 real-world forecasting. We evaluate PDON on ERA5 reanalysis, a large-scale real-world atmospheric dataset whose pressure-level fields are governed, to leading order, by the hydrostatic primitive equations. We use multilevel ERA5 fields at 850, 500, and $250\,\mathrm{hPa}$, concatenating geopotential, temperature, and horizontal wind components as the input state, and evaluate sequence-to-sequence prediction of the next seven days of the geopotential field z. This benchmark is substantially more complex than the synthetic PDE suites: it combines real observationally constrained dynamics, multiscale transport, vertical coupling, and nonstationary atmospheric variability. On this weekly forecasting task (Figure 5 shows inferred 3 days out of a week), PDON-LinOSS achieves the best error among our tested temporal cores, with relative-$L^2 = 0.04$, compared with S5 0.10, RNO 0.08, LRU 0.12, and Mamba 0.14, and remains competitive with weather-specialised foundation models such as FourCastNet 0.008 and ClimaX 0.01. Full data construction, physical equations, preprocessing, hyperparameter details, and data split are provided in Appendix F.7 and G.6.

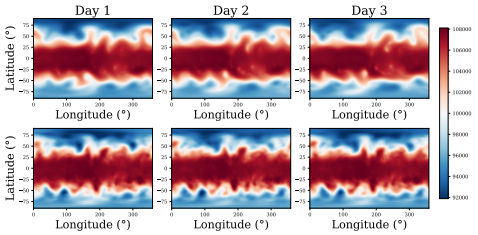


Figure 5: ERA5 geopotential trajectory prediction. Top: Ground truth, Bottom: PDON-LinOSS

5.3 Resolution invariance

A useful consequence of the PDON factorisation is that spatial resolution transfer can be handled independently of temporal evolution. The encoder maps each spatial snapshot to a compact latent coefficient sequence, $\mathcal{F}_{\mathrm{temp}}$ evolves that sequence in a resolution-independent latent space, and the decoder reconstructs the field on the requested grid. When the spatial blocks are spectral, the learned low-mode lift and decoder are defined on Fourier coefficients rather than on a fixed grid, so the same checkpoint can be evaluated on any grid whose Fourier support contains the mode budget. Resolution transfer is therefore handled by the spatial interface, while the latent temporal core is unchanged. For fixed-width spatial blocks such as POD, we use fixed band-limited Fourier interpolation before and after the learned operator; no additional bridge network or fine-tuning is introduced (more details in Appendix G.7). We evaluate this transfer from the training grid to higher-resolution grids on RD-ftx in Figure 6.

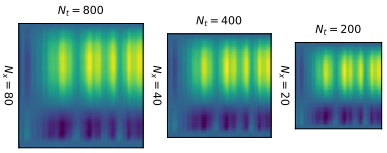


Figure 6: Resolution transfer at $(N_x, N_t) = (20, 200), (40, 400), (80, 800)$.

6 Conclusion

We introduced the Pseudo-DeepONet (PDON), an explicit encode–evolve–decode neural-operator factorisation for time-dependent PDEs as $\mathcal{D}_{\mathrm{out}} \circ \mathcal{L}_{\mathrm{out}} \circ \mathcal{F}_{\mathrm{temp}} \circ \mathcal{L}_{\mathrm{in}} \circ \mathcal{E}_{\mathrm{in}}$. By separating spatial representation from latent temporal evolution, PDON unifies flat, spectral, and POD/PCA spatial interfaces with interchangeable temporal cores. Theory shows when this split improves approximation scaling over monolithic space–time maps, and experiments on standard synthetic PDEs and complex real-world fields show improved generalisation and long-horizon stability without increasing model size. A limitation is that the usefulness of the split depends on the degree to which the underlying variables admit a meaningful spatial–temporal factorisation; in strongly entangled regimes, the choice of encoder, latent state, and temporal core must be checked carefully. These failure regimes, together with an ablation that evolves a spatial axis instead of time, are discussed in Appendix K. Future work should extend PDON beyond deterministic solution operators toward generative operator models, where initial conditions, forcing, or coefficients are themselves stochastic processes and the goal is probabilistic inference over solution trajectories rather than point prediction alone.

AI USE STATEMENT

Generative AI tools (large language models) were used for writing and polishing the text of this paper and for improving the appearance of its figures.

REFERENCES

David J Acheson. *Elementary fluid dynamics*. Oxford University Press, 1990.

Anurag Arnab, Mostafa Dehghani, Georg Heigold, Chen Sun, Mario Lučić, and Cordelia Schmid. ViViT: A video vision transformer. In *International Conference on Computer Vision*, pp. 6836–6846, 2021.

Peter Benner, Serkan Gugercin, and Karen Willcox. A survey of projection-based model reduction methods for parametric dynamical systems. *SIAM Review*, 57(4):483–531, 2015.

Kaushik Bhattacharya, Bamdad Hosseini, Nikola B. Kovachki, and Andrew M. Stuart. Model reduction and neural networks for parametric PDEs. *SMAI Journal of Computational Mathematics*, 7:121–157, 2021.

Ricardo Buitrago Ruiz, Tanya Marwah, Albert Gu, and Andrej Risteski. On the benefits of memory for modeling time-dependent PDEs. In *The Thirteenth International Conference on Learning Representations*, 2025.

Edoardo Calvello, Nikola B. Kovachki, Matthew E. Levine, and Andrew M. Stuart. Continuum attention for neural operators. *Journal of Machine Learning Research*, 26(300):1–52, 2025.

Qianying Cao, Somdatta Goswami, and George Em Karniadakis. Laplace neural operator for solving differential equations. *Nature Machine Intelligence*, 6:631–640, 2024.

Shuhao Cao. Choose a transformer: Fourier or Galerkin. In *Advances in Neural Information Processing Systems*, volume 34, pp. 24924–24940, 2021.

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

Tianping Chen and Hong Chen. Universal approximation to nonlinear operators by neural networks with arbitrary activation functions and its application to dynamical systems. *IEEE Transactions on Neural Networks*, 6(4):911–917, 1995.

Chun-Wun Cheng, Jiahao Huang, Yi Zhang, Guang Yang, Carola-Bibiane Schönlieb, and Angelica I Aviles-Rivero. Mamba neural operator: Who wins? Transformers vs. state-space models for PDEs. *Journal of Computational Physics*, 548:114567, 2025.

Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

Alexandre J. Chorin, Ole H. Hald, and Raz Kupferman. Optimal prediction with memory. *Physica D: Nonlinear Phenomena*, 166(3-4):239–257, 2002.

Krzysztof Marcin Choromanski, Valerii Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamás Sarlós, Peter Hawkins, Jared Quincy Davis, Afroz Mohiuddin, Lukasz Kaiser, et al. Rethinking attention with Performers. In *International Conference on Learning Representations*, 2021.

Tri Dao and Albert Gu. Transformers are SSMs: Generalized models and efficient algorithms through structured state space duality. In *International Conference on Machine Learning*, 2024.

Tim De Ryck and Siddhartha Mishra. Generic bounds on the approximation error for physics-informed (and) operator learning. *Advances in Neural Information Processing Systems*, 35:10945–10958, 2022.

Mostafa Dehghani, Stephan Gouws, Oriol Vinyals, Jakob Uszkoreit, and Łukasz Kaiser. Universal transformers. *arXiv preprint arXiv:1807.03819*, 2018.

Waleed Diab and Mohammed Al Kobaisi. Temporal neural operator for modeling time-dependent physical phenomena. *Scientific Reports*, 15(1):32791, 2025.

Xin Dong, Yonggan Fu, Shizhe Diao, Wonmin Byeon, Zijia Chen, Ameya Sunil Mahabaleshwarkar, Shih-Yang Liu, Matthijs Van Keirsbilck, Min-Hung Chen, Yoshi Suhara, Yingyan Celine Lin, Jan Kautz, and Pavlo Molchanov. Hymba: A hybrid-head architecture for small language models. In *The Thirteenth International Conference on Learning Representations*, 2025.

Sven Dummer, Dongwei Ye, and Christoph Brune. RONOM: Reduced-order neural operator modeling. *SIAM Journal on Scientific Computing*, 48:C604–C634, 2026.

Daniel Y. Fu, Tri Dao, Khaled K. Saab, Armin W. Thomas, Atri Rudra, and Christopher Ré. Hungry hungry hippos: Towards language modeling with state space models. In *International Conference on Learning Representations*, 2023.

Thomas Hakon Gronwall. Note on the derivatives with respect to a parameter of the solutions of a system of differential equations. *Annals of Mathematics*, 20(4):292–296, 1919.

Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=tEYskw1VY2.

Albert Gu, Karan Goel, and Christopher Ré. Efficiently modeling long sequences with structured state spaces. In *International Conference on Learning Representations*, 2022.

Ankit Gupta, Albert Gu, and Jonathan Berant. Diagonal state spaces are as effective as structured state spaces. *Advances in Neural Information Processing Systems*, 35:22982–22994, 2022.

Zhongkai Hao, Zhengyi Wang, Hang Su, Chengyang Ying, Yinpeng Dong, Songming Liu, Ze Cheng, Jun Zhu, and Jian Song. GNOT: A general neural operator transformer for operator learning. In *International Conference on Machine Learning*, pp. 12556–12569, 2023.

Junyan He, Shashank Kushwaha, Jaewan Park, Seid Koric, Diab Abueidda, and Iwona Jasiuk. Sequential deep operator networks (S-DeepONet) for predicting full-field solutions under time-dependent loads. *Engineering Applications of Artificial Intelligence*, 127:107258, 2024.

Jan S. Hesthaven and Stefano Ubbiali. Non-intrusive reduced order modeling of nonlinear problems using neural networks. *Journal of Computational Physics*, 363:55–78, 2018.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8): 1735–1780, 1997.

Zheyuan Hu, Qianying Cao, Kenji Kawaguchi, and George Em Karniadakis. DeepOMamba: State-space model for spatio-temporal PDE neural operator learning. *Journal of Computational Physics*, 540:114272, 2025.

Pengzhan Jin, Shuai Meng, and Lu Lu. MIONet: Learning multiple-input operators via tensor product. *SIAM Journal on Scientific Computing*, 44(6):A3490–A3514, 2022.

Taniya Kapoor, Abhishek Chandra, Daniel M Tartakovsky, Hongrui Wang, Alfredo Nunez, and Rolf Dollevoet. Neural oscillators for generalization of physics-informed machine learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 13059–13067, 2024a.

Taniya Kapoor, Hongrui Wang, Alfredo Núñez, and Rolf Dollevoet. Transfer learning for improved generalizability in causal physics-informed neural networks for beam simulations. *Engineering Applications of Artificial Intelligence*, 133:108085, 2024b.

George Em Karniadakis, Ioannis G. Kevrekidis, Lu Lu, Paris Perdikaris, Sifan Wang, and Liu Yang. Physics-informed machine learning. *Nature Reviews Physics*, 3(6):422–440, 2021.

Yegon Kim, Hyunsu Kim, Gyeonghoon Ko, and Juho Lee. Active learning with selective time-step acquisition for PDEs. In *International Conference on Machine Learning*, pp. 30199–30223. PMLR, 2025.

Katiana Kontolati, Somdatta Goswami, George Em Karniadakis, and Michael D. Shields. Learning nonlinear operators in latent spaces for real-time predictions of complex dynamics in physical systems. *Nature Communications*, 15(1):5101, 2024.

Nikola Kovachki, Zongyi Li, Burigede Liu, Kamyar Azizzadenesheli, Kaushik Bhattacharya, Andrew M. Stuart, and Anima Anandkumar. Neural operator: Learning maps between function spaces with applications to PDEs. *Journal of Machine Learning Research*, 24(89):1–97, 2023.

Nikola B Kovachki, Samuel Lanthaler, and Andrew M Stuart. Operator learning: Algorithms and analysis. *Handbook of Numerical Analysis*, 25:419–467, 2024.

Zijie Li, Kazem Meidani, and Amir Barati Farimani. Transformer for partial differential equations' operator learning. *Transactions on Machine Learning Research*, 2023a. ISSN 2835-8856. URL https://openreview.net/forum?id=EPPqt3uERT.

Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Andrew Stuart, Kaushik Bhattacharya, and Anima Anandkumar. Multipole graph neural operator for parametric partial differential equations. In *Advances in Neural Information Processing Systems*, volume 33, pp. 6755–6766, 2020.

Zongyi Li, Nikola B. Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew M. Stuart, and Anima Anandkumar. Fourier neural operator for parametric partial differential equations. In *International Conference on Learning Representations*, 2021.

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

Opher Lieber, Barak Lenz, Hofit Bata, Gal Cohen, Jhonathan Osin, Itay Dalmedigos, Erez Safahi, Shaked Meirom, Yonatan Belinkov, Shai Shalev-Shwartz, et al. Jamba: A hybrid Transformer-Mamba language model. *arXiv preprint arXiv:2403.19887*, 2024.

Phillip Lippe, Bastiaan S. Veeling, Paris Perdikaris, Richard E. Turner, and Johannes Brandstetter. PDE-Refiner: Achieving accurate long rollouts with neural PDE solvers. In *Advances in Neural Information Processing Systems*, 2023.

Shouyi Liu, Xiaokang Yang, and Yuntian Chen. Riesz neural operator for solving partial differential equations. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=Vjw7q1quNt.

Lu Lu, Pengzhan Jin, Guofei Pang, Zhongqiang Zhang, and George Em Karniadakis. Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators. *Nature Machine Intelligence*, 3(3):218–229, 2021.

Lu Lu, Xuhui Meng, Shengze Cai, Zhiping Mao, Somdatta Goswami, Zhongqiang Zhang, and George Em Karniadakis. A comprehensive and fair comparison of two neural operators (with practical extensions) based on FAIR data. *Computer Methods in Applied Mechanics and Engineering*, 393:114778, 2022.

Chao Ma, Jianchun Wang, et al. Model reduction with memory and the machine learning of dynamical systems. *arXiv preprint arXiv:1808.04258*, 2018.

Romit Maulik, Bethany Lusch, and Prasanna Balaprakash. Reduced-order modeling of advection-dominated systems with recurrent neural networks and convolutional autoencoders. *Physics of Fluids*, 33(3):037106, 2021.

Katarzyna Michałowska, Somdatta Goswami, George Em Karniadakis, and Signe Riemer-Sørensen. Neural operator learning for long-time integration in dynamical systems with recurrent neural networks. In *2024 International joint conference on neural networks (IJCNN)*, pp. 1–8. IEEE, 2024.

Hazime Mori. Transport, collective motion, and Brownian motion. *Progress of Theoretical Physics*, 33(3):423–455, 1965.

Sadao Nakajima. On quantum theory of transport phenomena: Steady diffusion. *Progress of Theoretical Physics*, 20(6):948–959, 1958.

Tung Nguyen, Johannes Brandstetter, Ashish Kapoor, Jayesh K Gupta, and Aditya Grover. ClimaX: A foundation model for weather and climate. *arXiv preprint arXiv:2301.10343*, 2023.

Antonio Orvieto, Samuel L. Smith, Albert Gu, Anushan Fernando, Caglar Gulcehre, Razvan Pascanu, and Soham De. Resurrecting recurrent neural networks for long sequences. In *International Conference on Machine Learning*, pp. 26670–26698, 2023.

Oded Ovadia, Adar Kahana, Panos Stinis, Eli Turkel, Dan Givoli, and George Em Karniadakis. Vito: Vision transformer-operator. *Computer Methods in Applied Mechanics and Engineering*, 428: 117109, 2024.

Jaideep Pathak, Shashank Subramanian, Peter Harrington, Sanjeev Raja, Ashesh Chattopadhyay, Morteza Mardani, Thorsten Kurth, David Hall, Zongyi Li, Kamyar Azizzadenesheli, et al. FourCastNet: A global data-driven high-resolution weather model using adaptive Fourier neural operators. *arXiv preprint arXiv:2202.11214*, 2022.

Michael Poli, Stefano Massaroli, Eric Nguyen, Daniel Y. Fu, Tri Dao, Stephen Baccus, Yoshua Bengio, Stefano Ermon, and Christopher Ré. Hyena hierarchy: Towards larger convolutional language models. In *International Conference on Machine Learning*, pp. 28043–28078, 2023.

Alfio Quarteroni, Andrea Manzoni, and Federico Negri. *Reduced Basis Methods for Partial Differential Equations: An Introduction*, volume 92. Springer, 2015.

Maziar Raissi, Paris Perdikaris, and George E Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational physics*, 378:686–707, 2019.

Yulia Rubanova, Ricky TQ Chen, and David K Duvenaud. Latent ordinary differential equations for irregularly-sampled time series. *Advances in neural information processing systems*, 32, 2019.

T. Konstantin Rusch and Daniela Rus. Oscillatory state-space models. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=GRMfXcAAFh.

Christoph Schwab and Jakob Zech. Deep learning in high dimension: Neural network expression rates for generalized polynomial chaos expansions in UQ. *Analysis and Applications*, 17(1):19–55, 2019.

Louis Serrano, Thomas X Wang, Etienne Le Naour, Jean-Noël Vittaut, and Patrick Gallinari. AROMA: Preserving spatial structure for latent PDE modeling with local neural fields. *Advances in Neural Information Processing Systems*, 37:13489–13521, 2024.

Jimmy T.H. Smith, Andrew Warrington, and Scott Linderman. Simplified state space layers for sequence modeling. In *International Conference on Learning Representations*, 2023.

Karn Tiwari, Niladri Dutta, N M Anoop Krishnan, and Prathosh AP. Latent mamba operator for partial differential equations. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=Y4BDcJmb8t.

Alasdair Tran, Alexander Mathews, Lexing Xie, and Cheng Soon Ong. Factorized Fourier neural operators. In *International Conference on Learning Representations*, 2023.

Sifan Wang, Hanwen Wang, and Paris Perdikaris. On the eigenvector bias of Fourier feature networks: From regression to solving multi-scale PDEs with physics-informed neural networks. *Computer Methods in Applied Mechanics and Engineering*, 384:113938, 2021a.

Sifan Wang, Hanwen Wang, and Paris Perdikaris. Learning the solution operator of parametric partial differential equations with physics-informed DeepONets. *Science Advances*, 7(40):eabi8605, 2021b.

Sinong Wang, Belinda Z. Li, Madian Khabsa, Han Fang, and Hao Ma. Linformer: Self-attention with linear complexity. *arXiv preprint arXiv:2006.04768*, 2020.

Gege Wen, Zongyi Li, Kamyar Azizzadenesheli, Anima Anandkumar, and Sally M. Benson. U-FNO–an enhanced Fourier neural operator-based deep-learning model for multiphase flow. *Advances in Water Resources*, 163:104180, 2022.

Dmitry Yarotsky. Error bounds for approximations with deep relu networks. *Neural networks*, 94: 103–114, 2017.

Anthony Zhou and Amir Barati Farimani. Masked autoencoders are PDE learners. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL https://openreview.net/forum?id=rZNuiFwXVs.

Robert Zwanzig. Memory effects in irreversible thermodynamics. *Physical Review*, 124(4):983–992, 1961.

TECHNICAL APPENDICES

A Notation Summary . . . 16
B Background . . . 17
B.1 Neural operators primer . . . 17
B.2 DeepONet, POD-DeepONet, and Latent DeepONet . . . 17
B.3 Physics-informed neural networks . . . 18
B.4 Fourier and Laplace neural operators . . . 19
B.5 State-space temporal cores . . . 19
B.6 Galerkin, attention, and recurrent operators . . . 20
B.7 Mori–Zwanzig formalism and memory in operators . . . 21
C Theory and Proofs . . . 21
C.1 Target function classes . . . 21
C.2 Lemma 1 (Interaction Residual Control via Duhamel/Gronwall) . . . 21
C.3 Lemma 2 (Yarotsky-type Minimax Lower Bound) . . . 22
C.4 Lemma 3 (Split Upper Bound, Decoupled Accuracy Parameter) . . . 23
C.5 Proof of Theorem 1 . . . 24
C.6 Crossover Point Derivation . . . 24
C.7 Scope of the Factorisable Subclass: Most PDE Systems Fit . . . 25
C.8 Proof of Theorem 2 . . . 26
D Extended Related Work . . . 28
D.1 Implicit and Monolithic Neural Operators . . . 28
D.2 Explicit Factorised and Latent Operators . . . 29
D.3 Temporal Cores in Operator Learning . . . 29
D.4 Memory and Mori–Zwanzig . . . 29
D.5 Resolution-Invariance Literature . . . 30
E Theoretical Complexity Analysis Table . . . 30
F Mathematical Explanation of Experiments . . . 31
F.1 Forced Burgers (1+1)D . . . 31
F.2 Forced KdV (1+1)D . . . 31
F.3 Reaction–Diffusion (1+1)D . . . 32
F.4 Fourier Reaction–Diffusion (2+1)D . . . 33
F.5 Brusselator (2+1)D, long horizon . . . 33
F.6 Navier–Stokes Beltrami flow (3+1)D . . . 33
F.7 ERA5 pressure-level system . . . 34
G Architecture Details and Design . . . 35
G.1 Model Instantiations Across Benchmark Suites . . . 35
G.2 1+1D suite: Burgers, KdV, and reaction–diffusion . . . 36
G.3 Fourier-RD 2+1D . . . 38
G.4 Brusselator 2+1D with POD/PCA PON . . . 39
G.5 Beltrami Flow 3+1D with POD/PCA PON . . . 40
G.6 ERA5 pressure-level forecasting with multilevel atmospheric fields . . . 41
G.7 Resolution-invariance protocol . . . 42
G.8 Inductive-bias experiment . . . 43
G.9 Architectural progression across suites . . . 44
H Algorithms and Python intuitive code . . . 45
H.1 Algorithms . . . 45
H.2 Python intuitive code . . . 46
I Additional Results . . . 47
I.1 Oscillatory-memory diagnostic . . . 47
I.2 Complexity Results . . . 49
I.3 Visualization Results . . . 55
J Hyperparameter Sensitivity . . . 58
K Scope and Failure Regimes of the Explicit Factorisation . . . 58
K.1 When Assumption A3 holds, and when it fails . . . 59
K.2 Dissipative benchmarks and the energy-preserving temporal core . . . 59
K.3 Suites on which LinOSS is not the top performer . . . 60

K.4 Evolving along a spatial axis instead of time 60

A Notation Summary

Symbol	Meaning
$D_x \subset \mathbb{R}^{n_x}$	spatial domain
$D_t = [0, T]$	temporal domain
X, Y	Banach input / output function spaces
$G : X \to Y$	exact PDE solution operator
G_θ^{PDON}	PDON learned operator; $G_\theta^{\mathrm{PDON}} = \mathcal{D}_{\mathrm{out}} \circ \mathcal{L}_{\mathrm{out}} \circ \mathcal{F}_{\mathrm{temp}} \circ \mathcal{L}_{\mathrm{in}} \circ \mathcal{E}_{\mathrm{in}}$
$\mathcal{E}$	generic encoder in the theory; in PDON: $\mathcal{E} = \mathcal{L}_{\mathrm{in}} \circ \mathcal{E}_{\mathrm{in}}$ maps sampled input snapshots to a latent sequence
m_x	spatial sensor count per snapshot; in PDON: $m_x = r_{\mathrm{POD}}$
m_t	temporal sensor count; in PDON: $m_t = N_t$
$\mathcal{R} : \mathbb{R}^p \to Y$	generic reconstructor; in PDON: $\mathcal{R} = \mathcal{D}_{\mathrm{out}} \circ \mathcal{L}_{\mathrm{out}}$
$A : \mathbb{R}^m \to \mathbb{R}^p$	generic latent approximator; in PDON: $A = \mathcal{F}_{\mathrm{temp}}$
$N_\theta = \mathcal{R} \circ A_\theta \circ \mathcal{E}$	generic learned neural operator; in PDON: $N_\theta \equiv G_\theta^{\mathrm{PDON}}$
$\hat{E}(N) = \|G - N\|_{L^2(\mu)}$	total approximation error under input measure μ
A^{joint}	monolithic latent map of size $\leq W$
$A^{\mathrm{split}} = A_t \circ A_x$	split latent map; $A_x : \mathbb{R}^{m_x} \to \mathbb{R}^{d_h}$ ($\equiv$ $\mathcal{L}_{\mathrm{in}} \circ \mathcal{E}_{\mathrm{in}}$ per snapshot), $A_t : \mathbb{R}^{d_h} \to \mathbb{R}^p$ ($\equiv \mathcal{L}_{\mathrm{out}} \circ \mathcal{F}_{\mathrm{temp}}$)
L	Lipschitz constant of G on bounded balls (Assumption A1)
$\{\phi_k\}$, $\{\alpha_k\}$, $\{Y_k\}$	orthonormal basis, decay coefficients, iid $[-1, 1]$ scalars in A2
ℓ	exponential decay rate of $\|\alpha_k\| \leq C_\alpha e^{-\ell\|k\|}$
$M^2 \leq C_\alpha^2/(1 - e^{-2\ell})$	a.s. uniform bound on $\|u\|_X^2$
Γ_μ, $\mathrm{Tr}(\Gamma_\mu)$	covariance operator and its spectral trace
δ	interaction residual in $A^* = A_t^* \circ A_x^* + \delta$ (Assumption A3)
$C_\delta, C_\alpha, C_r, C_N$	assumption-level constants (independent of ε)
k	joint regularity exponent of A^* (used in Yarotsky's bound)
r	split regularity exponent of A_x^*, A_t^*; distinct from r_{POD} (operator smoothness, not rank)
$\varepsilon, \varepsilon_{\mathrm{floor}}$	target error tolerance and irreducible split floor
$W^{\mathrm{joint}}(\varepsilon), W^{\mathrm{split}}(\varepsilon)$	parameter counts to achieve ε
a, u	continuous input function and output trajectory, with $u = G(a)$
$\mathbf{a}, \mathbf{u}$	gridded input/output pair; $\mathbf{a} = \{a(x_k, t_n)\}_{k,n}$, $\mathbf{u} = \{u(x_k, t_n)\}_{k,n}$
$\mathbf{A}, \mathbf{U}$	batched input/output tensors; $\mathbf{A} \in \mathbb{R}^{B \times N_x \times N_t \times C_{\mathrm{in}}}$, $\mathbf{U} \in \mathbb{R}^{B \times N_x \times N_t \times C}$
B	batch size
C_{in}, C	number of input / output field channels
N_x, N_t	number of spatial nodes / temporal levels
C	number of field channels (components)
d_h	latent hidden dimension; in PDON we use $d_h \ll N_x C$
r_{POD}	POD truncation rank when $\mathcal{E}_{\mathrm{in}}$ is POD; in practice $r_{\mathrm{POD}} = d_h$
M	spectral-mode budget in FNO/FFNO/LNO and in band-limited Fourier projectors
$\mathcal{E}_{\mathrm{in}}, \mathcal{F}_{\mathrm{temp}}, \mathcal{D}_{\mathrm{out}}$	encoder, temporal core, decoder modules of PDON
$W_{\mathrm{in}}, W_{\mathrm{out}}$	input / output linear projectors
$P_{N' \leftarrow N}$	band-limited Fourier resampling operator from N to N' samples
θ	trainable parameter vector of the PDON model
$\hat{u}_\theta$	predicted trajectory; $\hat{u}_\theta = G_\theta^{\mathrm{PDON}}(a)$
Λ	SSM diagonal state-transition matrix (distinct from latent approximator A)
$z(t)$, z_n	temporal latent state / SSM hidden state; dimension d_h
$v(t)$, v_t	input sequence to the SSM temporal core
$o(t)$	SSM output sequence
$\mathbf{B}, \mathbf{C}$	SSM input/output weight matrices (bold to distinguish from scalar B, C)

Correspondence. In PDON, the generic triple $(\mathcal{E}, A, \mathcal{R})$ instantiates as $(\mathcal{L}_{\mathrm{in}} \circ \mathcal{E}_{\mathrm{in}},\ \mathcal{F}_{\mathrm{temp}},\ \mathcal{D}_{\mathrm{out}} \circ \mathcal{L}_{\mathrm{out}})$; the split pair (A_x, A_t) instantiates as $(\mathcal{L}_{\mathrm{in}} \circ \mathcal{E}_{\mathrm{in}}$ per snapshot, $\mathcal{L}_{\mathrm{out}} \circ \mathcal{F}_{\mathrm{temp}})$.

B BACKGROUND

This appendix recalls the parts of the operator-learning, spectral, and state-space-model literature that the main text and proofs rely on. Readers familiar with these topics may skip this section.

B.1 NEURAL OPERATORS PRIMER

We briefly make explicit the PDE-to-operator formulation used throughout the paper. Let $D_x \subset \mathbb{R}^{n_x}$ be the spatial domain with boundary ∂D_x, and let $D_t = [0, T]$ be the time interval. We consider parametric time-dependent PDEs of the general form

$$\partial_t u(x,t) = \mathcal{P}\big(u, \nabla_x u, \nabla_x^2 u, \ldots; a\big), \qquad (x,t) \in D_x \times (0,T], \tag{5}$$

$$u(\cdot, 0) = u_0, \qquad \mathcal{B}(u) = 0 \quad \text{on } \partial D_x \times D_t. \tag{6}$$

Here $\mathcal{P}$ denotes the PDE evolution operator and may depend on the field u, its spatial derivatives, coefficients, and external forcing. The operator $\mathcal{B}$ specifies boundary conditions, such as periodic, Dirichlet, Neumann, or mixed boundary constraints. The input parameter $a \in X$ may represent any subset of the data that determines the solution: initial condition u_0, boundary data, PDE coefficients, source terms, or forcing functions. For a fixed PDE family, these data define a solution operator

$$G : X \to Y, \qquad a \mapsto u = G(a),$$

where u is the full space–time trajectory on $D_x \times D_t$. Thus the learning problem is to approximate G by a parametric neural operator G_θ, using samples of input–output trajectories.

For computation, the continuous functions are observed on a spatial grid $\{x_k\}_{k=1}^{N_x}$ and a temporal grid $\{t_n\}_{n=0}^{N_t-1} \subset [0, T]$. Unless otherwise stated, we use the uniform temporal grid

$$t_n = n\Delta t, \qquad \Delta t = \frac{T}{N_t - 1}.$$

This gives gridded input/output pairs

$$\mathbf{a} = \{a(x_k, t_n)\}_{k,n}, \qquad \mathbf{u} = \{u(x_k, t_n)\}_{k,n},$$

and batched tensors

$$\mathbf{A} \in \mathbb{R}^{B \times N_x \times N_t \times C_{\text{in}}}, \qquad \mathbf{U} \in \mathbb{R}^{B \times N_x \times N_t \times C}.$$

This formulation covers both one-shot operator learning, where the model maps conditioning data directly to a full trajectory, and sequence-to-sequence forecasting, where a finite input history is mapped to future states.

A *neural operator* (Kovachki et al., 2023; Li et al., 2021; Lu et al., 2021) is a parametric map $\mathcal{G}_\theta : X \to Y$, where X, Y are Banach spaces of functions on a bounded domain $D \subset \mathbb{R}^d$. Practical implementations factor $\mathcal{G}_\theta$ through finite-dimensional discretisations $\mathcal{E} : X \to \mathbb{R}^m$ (encoder) and $\mathcal{R} : \mathbb{R}^p \to Y$ (reconstructor), and learn an internal latent operator $A : \mathbb{R}^m \to \mathbb{R}^p$: $\mathcal{G}_\theta = \mathcal{R} \circ A_\theta \circ \mathcal{E}$. Universal approximation results of Chen & Chen (1995) and Kovachki et al. (2023) show that a sufficiently wide A_θ can approximate any continuous operator on a compact subset of X to arbitrary accuracy. The principal difference between architectures lies in how $\mathcal{E}$ and $\mathcal{R}$ are realised: DeepONet (Lu et al., 2021) chooses $\mathcal{E}$ as pointwise samples on a fixed sensor grid and $\mathcal{R}$ as a learned coordinate trunk; FNO (Li et al., 2021) uses spectral convolutions with shared weights across resolutions; LNO (Cao et al., 2024) replaces the Fourier transform with the Laplace transform; the present PDON family treats $\mathcal{E}, \mathcal{R}$ as a learned or POD-defined Proper Orthogonal Network and decoder.

B.2 DEEPONET, POD-DEEPONET, AND LATENT DEEPONET

Vanilla DeepONet. Given a discretised input function $a : \{x_j\}_{j=1}^m \to \mathbb{R}$ and a query coordinate $y \in D$, vanilla DeepONet outputs

$$\hat{u}(y; a) \;=\; \sum_{k=1}^{p} br_k(a)\, tr_k(y), \tag{7}$$

with b_k a *branch* MLP on the m-vector a and t_k a *trunk* MLP on the coordinate y. In time-dependent problems, $y = (x, t)$ enters the trunk as just another coordinate; this is the source of the spectral-bias and autoregressive-decay pathologies discussed in §1 and revisited in the benchmark and ablation experiments of §5.

POD-DeepONet. Lu et al. (2022) pre-computes a POD basis $\Phi \in \mathbb{R}^{p\times|D|}$ from training output snapshots and replaces the trunk by a fixed multiplication with Φ: $\hat{u}(y; a) = \sum_k b_k(a)\Phi_k(y)$. The branch then only has to learn the input-to-coefficient map; the spatial reconstruction is mathematically optimal in the truncation sense.

Latent DeepONet (L-DeepONet). Kontolati et al. (2024) replace the linear POD basis with a pre-trained autoencoder, granting the network a non-linear manifold compression; their experiments on convective shallow water and atmospheric flow exhibit one-to-two orders-of-magnitude speedups versus vanilla DeepONet and FNO-3D.

Reduced-Order Neural Operator Modelling (RONOM). Dummer et al. (2026) formalise the L-DeepONet structure as a learned encoder, a Neural-ODE temporal core, and a learned decoder, and prove resolution-transfer error bounds; their experiments demonstrate spatial and temporal super-resolution under unseen boundary conditions.

The PDON skeleton in Eq. (1) subsumes all three by exposing $\mathcal{E}_{\text{in}}, \mathcal{L}_{\text{in}}, \mathcal{F}_{\text{temp}}, \mathcal{L}_{\text{out}}, \mathcal{D}_{\text{out}}$ as independently selectable modules.

B.3 Physics-informed neural networks

Physics-informed neural networks (PINNs) (Raissi et al., 2019) form a distinct class of PDE learning methods from neural operators. Rather than learning an operator $G : X \to Y$ from many input–output function pairs, a standard PINN represents a single solution field directly by a neural network $u_\theta(x, t)$. The PDE is enforced by differentiating u_θ with respect to its inputs using automatic differentiation and penalising the residual at collocation points. For a time-dependent PDE

$$\partial_t u = \mathcal{P}\big(u, \nabla_x u, \nabla_x^2 u, \ldots; a\big),$$

the PINN residual is

$$\mathcal{R}_\theta(x, t) = \partial_t u_\theta(x, t) - \mathcal{P}\big(u_\theta, \nabla_x u_\theta, \nabla_x^2 u_\theta, \ldots; a\big),$$

and the training objective typically combines PDE, initial-condition, boundary, and data terms:

$$\mathcal{L}_{\text{PINN}}(\theta) = \lambda_{\text{pde}}\|\mathcal{R}_\theta\|^2_{\Omega_{\text{col}}} + \lambda_{\text{ic}}\|u_\theta(\cdot, 0) - u_0\|^2 + \lambda_{\text{bc}}\|\mathcal{B}(u_\theta)\|^2 + \lambda_{\text{data}}\|u_\theta - u_{\text{obs}}\|^2.$$

This makes PINNs attractive when the governing equations are known and labelled solution data are scarce. However, the model is usually tied to a particular PDE instance or parameter setting, whereas neural operators aim to amortise learning across a distribution of inputs a.

Several variants modify the PINN objective or training dynamics to improve long-horizon behaviour. Causal PINNs (Kapoor et al., 2024b) introduce a time-causal weighting of the residual so that earlier time regions are learned before later ones, reducing the tendency of standard PINNs to fit future-time residuals before the preceding dynamics are accurate. Recurrent PINNs (Kapoor et al., 2024a) incorporate recurrent temporal structure into the physics-informed framework, allowing the solution to be advanced through time with a learned sequential state while still enforcing the PDE residual. These methods are therefore related to our interest in stable temporal modelling, but they remain physics-informed solution learners rather than operator learners in the sense used throughout this paper.

In our experiments, PINN-style baselines are included as representatives of a different training paradigm. They use explicit PDE residual supervision, while PDON and the other neural-operator baselines are trained to approximate the data-driven solution operator from sampled trajectories. This distinction is important: PINNs test the benefit of physics-informed optimisation for individual systems, whereas PDON tests whether an explicit spatial–temporal operator factorisation improves amortised prediction across many PDE instances.

The distinction also determines where each paradigm is most useful. PINNs are well suited when the governing PDE is known, high-quality paired trajectory data are limited, and the goal is to solve

or infer one specific physical instance under explicit residual constraints. In contrast, PDON does not impose the PDE residual during training; it learns the solution operator directly from sampled input–output trajectories. This makes it more appropriate for amortised prediction across many initial conditions, forcings, coefficients, or partially observed real-world systems where the exact residual may be expensive, unknown, or only approximately valid. The tradeoff is that PDON relies on representative trajectory data, whereas PINNs can exploit known physics even in lower-data regimes.

B.4 Fourier and Laplace neural operators

FNO. The FNO layer replaces a kernel integral operator by a low-mode spectral multiplication: $(\mathcal{S}H)(x) = \mathcal{F}^{-1}\big(K_\theta \odot \mathcal{F}H\big)(x)$, where $\mathcal{F}$ is the discrete Fourier transform on D, K_θ is a trainable complex weight tensor truncated to M modes per spatial axis, and $\odot$ denotes Hadamard multiplication. Stacked $\mathcal{S}$ blocks with pointwise nonlinearities form an $(d{+}1)$-D FNO when $D = D_x \times D_t$.

FFNO. Tran et al. (2023) factorise K_θ across the $d{+}1$ axes and add residual connections; this reduces parameter count from $\mathcal{O}(M^{d+1}W^2)$ to $\mathcal{O}((d{+}1)MW^2)$ and improves long-horizon accuracy.

LNO. Cao et al. (2024) parameterise the kernel in pole-residue form via the Laplace transform; this gives an explicit causal structure but assumes a stable linear-time-invariant kernel, which limits its applicability to genuinely nonlinear PDEs.

In PDON, the FNO/LNO operations enter as one possible realisation of $\mathcal{E}_{\text{in}}$ (a spectral lift), *not* as the joint (x, t) operator they originally targeted.

B.5 State-space temporal cores

A continuous-time linear state-space model (SSM) reads $z'(t) = \Lambda z(t) + \mathbf{B}v(t),\ o(t) = \mathbf{C}z(t)$, where $\Lambda \in \mathbb{R}^{d_h \times d_h}$ is the diagonal state-transition matrix (distinct from the latent approximator A), $v(t)$ is the input sequence, $o(t)$ the output, and $\mathbf{B}, \mathbf{C}$ are learned weight matrices. Discretisation with step Δ via zero-order hold produces $z_t = \bar{\Lambda} z_{t-1} + \bar{\mathbf{B}} v_t$ with $\bar{\Lambda} = e^{\Delta\Lambda}$, $\bar{\mathbf{B}} = (\Delta\Lambda)^{-1}(e^{\Delta\Lambda} - I)\Delta\mathbf{B}$. Modern SSMs differ in how they parameterise Λ and how they compute the recurrence in parallel.

S4 / S5 / LRU. The S4 model (Gu et al., 2022) uses HiPPO-LegS-initialised Λ matrices computed via a structured kernel. S5 (Smith et al., 2023) simplifies S4 to a diagonal Λ in the complex plane and computes the recurrence with a parallel scan in $\mathcal{O}(N \log N)$ time. LRU (Orvieto et al., 2023) further simplifies the parameterisation, retaining diagonal Λ but deriving its initialisation from minimal stability requirements.

LinOSS. Orthonormal/Oscillatory State-Space models (LinOSS) (Rusch & Rus, 2025) replace the strictly dissipative first-order SSM dynamics by a second-order oscillatory latent system. In the notation of Section 3, LinOSS models each latent channel through

$$z''(t) = -\Lambda z(t) + \mathbf{B}v(t), \qquad \Lambda = \operatorname{diag}(\Lambda_1, \ldots, \Lambda_{d_h}), \qquad \Lambda_i \geq 0,$$

where $z(t) \in \mathbb{R}^{d_h}$ is the latent state and $v(t)$ is the input sequence to the temporal core. Introducing the phase-space state

$$s_t = (z_t, \dot{z}_t)^\top$$

turns the second-order oscillator into a first-order recurrence on position- and velocity-like latent variables. Since Λ is diagonal, the autonomous zero-input dynamics decouple channel-wise. For channel $i = 1, \ldots, d_h$, with learned stiffness $\Lambda_i \geq 0$ and step size $\Delta_i > 0$, the LinOSS-IMEX update is

$$s_{t+1,i} = \Phi_i^{\text{IMEX}} s_{t,i}, \qquad \Phi_i^{\text{IMEX}} = \begin{bmatrix} 1 & -\Delta_i\Lambda_i \\ \Delta_i & 1 - \Delta_i^2\Lambda_i \end{bmatrix}.$$

The characteristic polynomial of Φ_i^{IMEX} is

$$\lambda^2 - (2 - \Delta_i^2\Lambda_i)\lambda + 1 = 0.$$

Thus, whenever

$$0 < \Delta_i^2\Lambda_i < 4,$$

the two eigenvalues form a complex-conjugate pair on the unit circle. In this regime, the autonomous LinOSS dynamics are stable and oscillatory rather than exponentially dissipative.

This unit-circle spectrum should be interpreted as preservation of oscillator energy, not necessarily preservation of the plain Euclidean norm of the phase-space vector. The IMEX update is symplectic rather than Euclidean orthogonal. Consequently, for each stable channel there exists a positive quadratic energy matrix $Q_i \succ 0$ satisfying

$$(\Phi_i^{\mathrm{IMEX}})^\top Q_i \Phi_i^{\mathrm{IMEX}} = Q_i,$$

so that the discrete Hamiltonian energy

$$E_{t,i} = s_{t,i}^\top Q_i s_{t,i}$$

is conserved under the autonomous update:

$$E_{t+1,i} = s_{t,i}^\top (\Phi_i^{\mathrm{IMEX}})^\top Q_i \Phi_i^{\mathrm{IMEX}} s_{t,i} = E_{t,i}.$$

This is the sense in which LinOSS provides an energy-preserving temporal integrator. It can retain bounded oscillatory memory over long horizons without forcing all modes to decay, unlike first-order stable SSMs whose eigenvalues are typically placed strictly inside the unit disk.

In practice, LinOSS is used as a learnable temporal core rather than as a fixed physical integrator. The input term $\mathbf{B}v(t)$, output map $\mathbf{C}$, channel mixing, nonlinearities, and residual connections provide expressiveness, while the underlying oscillator transition supplies a stable inductive bias. We use "LinOSS" to refer to this IMEX oscillatory core inside $\mathcal{F}_{\mathrm{temp}}$. Its structure is especially appropriate for PDE trajectories with wave propagation, transport-dominated motion, Beltrami-type persistent modes, or weakly dissipative modal exchange, where long-time prediction benefits from preserving energy-bearing latent modes rather than imposing artificial temporal damping.

Mamba. Gu & Dao (2024) extend SSMs with an input-dependent selection mechanism, $\bar{\Lambda} = \bar{\Lambda}(v_t)$, granting the network the ability to filter irrelevant information while retaining the parallel scan. Mamba is the strongest temporal core in our ablation under chaotic forcing.

In PDON, $\mathcal{F}_{\mathrm{temp}}$ may be any of OSS, S5, LRU, Mamba, GRU, LSTM, Galerkin transformer, or vanilla transformer; §5.1 ablates this choice.

B.6 Galerkin, attention, and recurrent operators

Galerkin Transformer. Cao (2021) observe that the standard softmax self-attention $\mathrm{Attn}(Q,K,V) = \mathrm{softmax}(QK^\top/\sqrt{d_h})V$ is unnecessary for operator approximation and replace it with a normalised linear product $\mathrm{GAttn}(Q,K,V) = QK^\top V/N_t$, where $Q,K,V \in \mathbb{R}^{N_t \times d_h}$ are learned linear projections of the token sequence and N_t is the sequence length. This is equivalent to a Galerkin projection of the query subspace onto the key–value span and achieves $\mathcal{O}(N_t d_h^2)$ cost instead of $\mathcal{O}(N_t^2 d_h)$. In PDON, the Galerkin Transformer instantiates $\mathcal{F}_{\mathrm{temp}}$ by running this attention over the latent sequence $\{v_t\}_{t=1}^{N_t}$ with $v_t \in \mathbb{R}^{d_h}$, keeping spatial and temporal processing separated (§5.1).

GNOT. Hao et al. (2023) extend attention-based neural operators to heterogeneous inputs via cross-attention between output query points and multiple sets of input tokens, handling irregular geometries and multi-physics settings. GNOT treats spatial coordinates and time as joint tokens; as a baseline in our benchmarks it represents the implicit-operator family.

GRU and LSTM. Gated Recurrent Units (Cho et al., 2014) and Long Short-Term Memory networks (Hochreiter & Schmidhuber, 1997) are first-order gated recurrences $h_t = f_\theta(h_{t-1}, v_t)$ where f_θ is a learned gating function. They are included as $\mathcal{F}_{\mathrm{temp}}$ choices in the temporal-core ablation (§5.1) as sequential lower-bound baselines; they do not admit a parallel scan and cannot match the long-range energy-preservation of LinOSS.

All $\mathcal{F}_{\mathrm{temp}}$ choices — Galerkin Transformer, GNOT-style attention, GRU, LSTM, and the SSM family above — consume the latent input sequence $\{v_t\}_{t=1}^{N_t}$ with $v_t \in \mathbb{R}^{d_h}$, maintain a hidden state $h_t \in \mathbb{R}^{d_h}$, and emit an output sequence $\{o_t\}$ of the same width, so they are interchangeable inside the PDON skeleton.

B.7 MORI-ZWANZIG FORMALISM AND MEMORY IN OPERATORS

When the PDE state is only *partially* observed — for instance, on a coarse grid that retains only the low Fourier modes — the Mori-Zwanzig formalism (Mori, 1965; Zwanzig, 1961) decomposes the dynamics of the observed projection $\mathcal{P}[u]$ into a Markovian term, a memory term (a convolution with the past trajectory), and an unobserved residual:

$$\partial_t \mathcal{P}[u] = \mathcal{P}\mathcal{L}\mathcal{P}[u] + \int_0^t K(t-s)\mathcal{P}[u](s)\,ds + R_t.$$

Buitrago Ruiz et al. (2025) (MemNO) operationalise this by appending an S4 sequence model to an FNO spatial block; their experiments show up to $6\times$ rel-L^2 improvement over Markovian FNO baselines on PDEs with significant high-frequency content under low-resolution observation. The $\mathcal{F}_{\text{temp}}$ module of PDON plays the same role: a learned approximation of the memory term acting on a low-dimensional latent.

C THEORY AND PROOFS

This appendix collects the proofs underlying Theorem 1 (Complexity Separation). We first make precise the two target function classes referred to in the main text; Lemma C.2 controls the interaction residual in A3 by the spectral trace via a Duhamel/Gronwall argument; Lemma 2 states the Yarotsky-type minimax lower bound that gives the joint exponent; Lemma 3 constructs the split approximator with a decoupled internal-accuracy parameter η and gives its parameter budget. Theorem 1 then follows by combining the three. Appendix C.7 discusses which PDE systems fall inside $\mathcal{F}_{r,\varepsilon_{\text{floor}}}(m)$.

C.1 TARGET FUNCTION CLASSES

The conditional nature of the separation hinges on stating two distinct target classes.

Definition 1 (Joint Sobolev class)**.** $\mathcal{F}_k(m) := \{F : [0,1]^m \to \mathbb{R}^p : F \in W^{k,\infty}([0,1]^m;\mathbb{R}^p),\ \|F\|_{W^{k,\infty}} \le 1\}$.

Definition 2 (Factorisable holomorphic subclass)**.** $\mathcal{F}_{r,\varepsilon_{\text{floor}}}(m) := \{F = A_t^* \circ A_x^* + \delta : A_x^*, A_t^*$ are (b,ρ,κ)-holomorphic with split regularity $r,\ \|\delta\|_{L^2(\mu)} \le \varepsilon_{\text{floor}}\}$.

Under A1–A4, the latent operator A^* inherited from G lies in $\mathcal{F}_{r,\varepsilon_{\text{floor}}}(m) \cap \mathcal{F}_k(m)$ for every finite k, and $\mathcal{F}_{r,\varepsilon_{\text{floor}}}(m) \subsetneq \mathcal{F}_k(m)$. The Yarotsky minimax bound (Lemma 2) is taken over $\mathcal{F}_k(m)$; the split construction (Lemma 3) operates on $\mathcal{F}_{r,\varepsilon_{\text{floor}}}(m)$. The separation in Theorem 1 is therefore a comparison between worst-case cost on the broad class and worst-case cost on the narrow factorisable subclass — exactly the asymmetry that exposes the architectural inductive bias of the split formulation. A joint architecture given the structural prior of A3 could implement the same Cohen–Schwab construction and recover the split rate; what the theorem isolates is the cost paid when this prior is absent.

C.2 LEMMA 1 (INTERACTION RESIDUAL CONTROL VIA DUHAMEL/GRONWALL)

Lemma 1 (Interaction Residual Control)**.** *Under A1 and A2, with horizon T and Lipschitz constant L for the linearisation, the residual $\delta = A^* - A_t^* \circ A_x^*$ from A3 satisfies*

$$\|\delta\|^2_{L^2(\mu)} \;\le\; C_\delta^2\ \mathrm{Tr}(\Gamma_\mu)^2 \;\le\; \frac{C_\delta^2\, C_\alpha^4}{9\,(1-e^{-2\ell})^2}, \tag{8}$$

where $C_\delta^2 := C_1\, T^2 e^{6LT}$ absorbs the time-integration factor. In particular $\|\delta\|_{L^2(\mu)} \to 0$ as $\ell \to \infty$.

Proof. **Duhamel formulation.** Let $\Phi^t : X \to X$ denote the full nonlinear semigroup of the PDE and Φ_L^t the semigroup of its linearisation $\partial_t u = Lu$. The Duhamel formula gives, for $u(0) = u_0 \sim \mu$,

$$u(T) \;=\; \Phi_L^T(u_0) \;+\; \int_0^T \Phi_L^{T-s}\big(N(u(s),u(s))\big)\,ds,$$

with N the quadratic (possibly derivative-type) nonlinearity. The factorisation $A_t^* \circ A_x^*$ is built from the linear semigroup Φ_L^T (factorable: spatial mode-projection A_x^* followed by per-mode temporal evolution A_t^*), so the residual is exactly the Duhamel integral:

$$\delta(u_0) \;=\; \int_0^T \Phi_L^{T-s}\big(N(u(s),u(s))\big)\,ds. \tag{9}$$

Gronwall bound on the integrand. Under A1, the linear semigroup satisfies $\|\Phi_L^t\|_{X\to X} \le e^{Lt}$. Cauchy–Schwarz on (9) gives

$$\|\delta(u_0)\|_X^2 \;\le\; T\int_0^T \big\|\Phi_L^{T-s}N(u(s),u(s))\big\|_X^2\,ds \;\le\; T\,e^{2LT}\int_0^T \|N(u(s),u(s))\|_X^2\,ds.$$

Taking expectation under μ and applying Fubini,

$$\|\delta\|_{L^2(\mu)}^2 \;\le\; T\,e^{2LT}\int_0^T \mathbb{E}_\mu\big[\|N(u(s),u(s))\|_X^2\big]\,ds. \tag{10}$$

Bilinear bound at each time. Under A1 and the Sobolev embedding implied by A2, $\|N(u,u)\|_X \le C_N\|u\|_X^2$, hence $\mathbb{E}_\mu\|N(u(s),u(s))\|_X^2 \le C_N^2\,\mathbb{E}_\mu\|u(s)\|_X^4$. A1 plus standard energy estimates give $\sup_{s\in[0,T]}\mathbb{E}_\mu\|u(s)\|_X^4 \le e^{4LT}\,\mathbb{E}_\mu\|u_0\|_X^4$.

Fourth-moment bound on the initial data. We control $\mathbb{E}_\mu\|u_0\|_X^4$ by either of two routes.

Route (i): a.s. control. Since $|Y_k| \le 1$ and $\{\phi_k\}$ is orthonormal,

$$\|u_0\|_X^2 = \sum_k Y_k^2\alpha_k^2 \le \sum_k \alpha_k^2 =: M^2 \le \frac{C_\alpha^2}{1-e^{-2\ell}} \quad \text{a.s.},$$

so $\mathbb{E}_\mu\|u_0\|_X^4 \le M^2\ \mathrm{Tr}(\Gamma_\mu)$.

Route (ii): independence. With $\kappa = \mathrm{Var}(Y_k^2) = 4/45$ for $Y_k \sim U[-1,1]$, independence gives

$$\mathbb{E}_\mu\|u_0\|_X^4 = \mathrm{Tr}(\Gamma_\mu)^2 + \kappa\sum_k \alpha_k^4 \;\lesssim\; \mathrm{Tr}(\Gamma_\mu)^2.$$

With $\mathbb{E}[Y_k^2] = 1/3$, $\mathrm{Tr}(\Gamma_\mu) = \frac{1}{3}\sum_k \alpha_k^2 \le C_\alpha^2/[3(1-e^{-2\ell})]$.

Combining everything. Inserting the bilinear bound and the fourth-moment estimate into (10),

$$\|\delta\|_{L^2(\mu)}^2 \;\le\; T\,e^{2LT}\cdot T\cdot C_N^2\cdot e^{4LT}\,\mathbb{E}_\mu\|u_0\|_X^4 \;\le\; C_1\,T^2\,e^{6LT}\ \mathrm{Tr}(\Gamma_\mu)^2.$$

Defining $C_\delta^2 := C_1T^2e^{6LT}$ yields (8). The factor of 9 comes from the $1/3$ in $\mathbb{E}[Y_k^2]$. As $\ell\to\infty$ the right-hand side decays to 0. □

Remark (Where the time integration enters). An instantaneous bilinear bound on the vector field N would give $\|\delta\|^2 \le C_N^2\mathbb{E}_\mu\|u\|_X^4$ directly, but δ is the time-integrated vector field along trajectories. The Duhamel step (9) makes this explicit and the Gronwall step pulls the linear-semigroup norm out of the integral, producing the legitimate T^2e^{6LT} dependence on the horizon. For dissipative parabolic systems $L<0$ and $Te^{LT} = \mathcal{O}(T)$ uniformly in T; for hyperbolic or weakly damped systems the constant grows with T, which is the expected behaviour.

Remark (Derivative-type nonlinearities). For PDE nonlinearities involving derivatives (Burgers $u\partial_x u$, advection $(u\cdot\nabla)u$, reaction-diffusion $\nabla\cdot(u^2)$), the bilinear bound $\|N(u,u)\|_X \le C_N\|u\|_X^2$ holds with $X = H^s(D_x)$, $s > n_x/2 +$ (order of derivative). The exponential decay in A2 ensures $\mathbb{E}_\mu\|u\|_{H^s}^2 < \infty$ for all $s \ge 0$, so the Duhamel/Gronwall step carries through unchanged.

C.3 Lemma 2 (Yarotsky-type Minimax Lower Bound)

Lemma 2 (Yarotsky-type minimax lower bound; Yarotsky, 2017)**.** *Let $\mathcal{F}_k(m)$ be as in Definition 1. For any ReLU architecture (joint, split, sparse, or otherwise) with W trainable parameters,*

$$\inf_{A^{\mathrm{joint}}:\,\mathrm{size}\le W}\ \sup_{F\in\mathcal{F}_k(m)} \big\|F - A^{\mathrm{joint}}\big\|_{L^\infty} \;\ge\; c(m,k)\,W^{-k/m},$$

and equivalently the parameter count required for worst-case error ε over $\mathcal{F}_k(m)$ satisfies

$$W^{\mathrm{joint}}_{\mathcal{F}_k(m)}(\varepsilon) \;\ge\; c(m,k)\,\varepsilon^{-m/k}. \tag{11}$$

Proof sketch. A ReLU network with W parameters partitions $[0,1]^m$ into at most $\mathcal{O}(W^m)$ polyhedral cells, on each of which it is a polynomial of degree $\leq k$. The Kolmogorov ε-entropy of the unit ball of $W^{k,\infty}([0,1]^m)$ in L^∞ is $\Omega(\varepsilon^{-m/k})$, so to achieve uniform error ε *over the whole class* the network must distinguish $\Omega(\varepsilon^{-m/k})$ test bumps. Counting active cells produces the stated lower bound. The argument counts what *any* W-parameter ReLU function can do, so it applies equally to split or sparse architectures: the only way to beat the bound is to restrict the target class. See (Yarotsky, 2017) for the matching upper bound up to a $\log(1/\varepsilon)$ factor. □

Remark (Why this is a class statement). Lemma 2 does *not* say that any specific holomorphic F requires $\varepsilon^{-m/k}$ parameters — a specific holomorphic target may be approximated much faster (e.g. by Cohen–Schwab itself). What it says is: whatever architecture you choose, there exists a function in $\mathcal{F}_k(m)$ that takes that many parameters. The theorem's separation is meaningful precisely because, in Part 2, the target class is restricted to $\mathcal{F}_{r,\varepsilon_{\text{floor}}}(m) \subsetneq \mathcal{F}_k(m)$.

C.4 Lemma 3 (Split Upper Bound, Decoupled Accuracy Parameter)

Lemma 3 (Split upper bound)**.** *Under A2 and A4, fix the input dimension m_x, the latent width d_h, and the output dimension p. For every accuracy $\eta > 0$ there exist neural networks $A_x : \mathbb{R}^{m_x} \to \mathbb{R}^{d_h}$ and $A_t : \mathbb{R}^{d_h} \to \mathbb{R}^p$ satisfying:*

(a) Sizes (additive prefactor, accuracy-dependent factor).

$$\text{size}(A_x) \leq C_r\,(m_x + d_h)\,\log(m_x)\log\log(m_x)\,\eta^{-1/r}, \tag{12}$$

$$\text{size}(A_t) \leq C_r\,(d_h + p)\,\log(d_h)\log\log(d_h)\,\eta^{-1/r}. \tag{13}$$

(b) Per-factor approximation error. $\|A_x^* - A_x\|_{L^\infty} \leq \eta$ *and* $\|A_t^* - A_t\|_{L^\infty} \leq \eta$.

(c) Composed error. *The split approximator $A^{\text{split}} := A_t \circ A_x$ satisfies*

$$\|A^* - A^{\text{split}}\|_{L^2(\mu)} \;\leq\; \big(1 + \text{Lip}(A_t^*)\big)\,\text{Lip}(\mathcal{R})\,\eta \;+\; \|\delta\|_{L^2(\mu)}. \tag{14}$$

Proof sketch. **Sizes.** For a (b,ρ,κ)-holomorphic map $H : [-1,1]^M \to \mathbb{R}^Q$ with ℓ^p-summable polynomial-chaos coefficients ($p = 1/r - 1/2$), the Cohen–Schwab–Zech construction (Schwab & Zech, 2019; De Ryck & Mishra, 2022) yields, for every accuracy $\eta > 0$, a ReLU network with:

- a sparse first layer linking only the M active inputs to a hidden representation (additive cost $\mathcal{O}(M \log M \log\log M)$, not multiplicative — only $\mathcal{O}(\log M)$ inputs feed into each hidden neuron);
- an internal body of size $\mathcal{O}(\eta^{-1/r} \log M \log\log M)$ that achieves accuracy η;
- a final linear layer of size $\mathcal{O}(Q)$ to the Q-dimensional output.

The total is $\mathcal{O}((M+Q)\log M \log\log M \cdot \eta^{-1/r})$. Applying with $(M,Q) = (m_x, d_h)$ and $(M,Q) = (d_h, p)$ gives (12)–(13).

Composed error. Decompose

$$A^* - A_t \circ A_x \;=\; \underbrace{(A^* - A_t^* \circ A_x^*)}_{=\,\delta} + A_t^* \circ A_x^* - A_t \circ A_x.$$

Telescoping the second term and using (b),

$$\|A_t^* \circ A_x^* - A_t \circ A_x\| \;\leq\; \text{Lip}(A_t^*)\,\|A_x^* - A_x\| + \|A_t^* - A_t\|\,\sup\|A_x\| \;\leq\; \big(1 + \text{Lip}(A_t^*)\big)\,\eta.$$

Multiplying by $\text{Lip}(\mathcal{R})$ to translate from latent to ambient error and adding $\|\delta\|_{L^2(\mu)}$ gives (14). □

Remark (Decoupling input dimension from accuracy). The accuracy parameter η in (12)–(13) is *separate* from the input dimension m_x: the Cohen–Schwab rate $\eta^{-1/r}$ comes from the analytic decay of the polynomial-chaos coefficients (A4) and is achievable at any fixed m_x by widening the network internally. This decoupling is what allows Theorem 1 to take $\varepsilon \to 0^+$ at *fixed* joint dimension $m = m_x m_t$, removing the asymptotic-regime contradiction that would arise from conflating input dimension and accuracy.

C.5 Proof of Theorem 1

Proof. **Part (1) — Joint minimax lower bound.** Under A4, $F(y) = G(u(\cdot; y))$ is (b, ρ, κ)-holomorphic on $[-1, 1]^m$, hence C^∞ and in particular $F \in W^{k,\infty}([-1, 1]^m; \mathbb{R}^p)$ for every finite k. The ideal latent operator $A^* = \mathcal{R}^\dagger \circ G \circ \mathcal{E}^\dagger$ inherits this regularity via composition with the Lipschitz left/right inverses of $\mathcal{E}$ and $\mathcal{R}$ (which exist on the relevant range under A1 and the discretisation assumptions), so $A^* \in \mathcal{F}_k(m)$. Lemma 2 then gives, for any architecture of size W, $\sup_{F \in \mathcal{F}_k(m)} \|F - A^{\text{joint}}\|_{L^\infty} \geq c(m, k) W^{-k/m}$, i.e. $W^{\text{joint}}_{\mathcal{F}_k(m)}(\varepsilon) \geq c(m, k)\varepsilon^{-m/k}$, at fixed m.

Part (2) — Split upper bound. Apply Lemma 3 with the per-factor accuracy

$$\eta \;:=\; \frac{\varepsilon - \varepsilon_{\text{floor}}}{2(1 + \text{Lip}(A_t^*))\,\text{Lip}(\mathcal{R})}.$$

By Lemma 3(c) and $\|\delta\|_{L^2(\mu)} \leq \varepsilon_{\text{floor}}$ (Lemma C.2),

$$\|A^* - A_t \circ A_x\|_{L^2(\mu)} \;\leq\; \tfrac{\varepsilon - \varepsilon_{\text{floor}}}{2} + \varepsilon_{\text{floor}} \;\leq\; \varepsilon.$$

Summing the sizes from (12)–(13) and using $\log(d_h) \leq \log(m)$,

$$W^{\text{split}}(\varepsilon) \;\leq\; 2C_r\,(m_x + d_h + p)\;\log(m)\log\log(m)\,\eta^{-1/r}.$$

Substituting $\eta \asymp \varepsilon - \varepsilon_{\text{floor}}$ (Lipschitz prefactor absorbed into C_r),

$$W^{\text{split}}(\varepsilon) \;\leq\; C_r\,(m_x + d_h + p)\;\log(m)\log\log(m)\,(\varepsilon - \varepsilon_{\text{floor}})^{-1/r}. \tag{15}$$

The dimensions m_x, d_h, p are fixed architecture constants in (15); the ε-dependence appears *only* in the trailing factor.

Part (3) — Conditional asymptotic separation. Define $\text{Ratio}(\varepsilon) := W^{\text{joint}}_{\mathcal{F}_k(m)}(\varepsilon)/W^{\text{split}}_{\mathcal{F}_{r,\varepsilon_{\text{floor}}}(m)}(\varepsilon)$. Substituting Parts (1) and (2),

$$\text{Ratio}(\varepsilon) \;\geq\; \frac{c(m,k)\,\varepsilon^{-m/k}}{C_r(m_x + d_h + p)\log(m)\log\log(m)\,(\varepsilon - \varepsilon_{\text{floor}})^{-1/r}}.$$

In the regime $\varepsilon_{\text{floor}} = o(\varepsilon)$, $(\varepsilon - \varepsilon_{\text{floor}})^{-1/r} \asymp \varepsilon^{-1/r}$, so

$$\text{Ratio}(\varepsilon) \;\gtrsim\; \frac{c(m,k)}{C_r(m_x + d_h + p)\log(m)\log\log(m)}\,\varepsilon^{-(m/k - 1/r)}.$$

Under $mr > k$ the exponent $m/k - 1/r$ is strictly positive, so $\text{Ratio}(\varepsilon) \to \infty$ as $\varepsilon \to 0^+$ *at fixed* m. Taking $\ell \to \infty$ in A2 sends $\varepsilon_{\text{floor}} \to 0$ by Lemma C.2, so the regime $\varepsilon_{\text{floor}} = o(\varepsilon)$ is always reachable. □

Remark (Boundary regimes $mr = k$ and $mr < k$). When $mr = k$, the two complexity exponents cancel: the split bound is better only by a polylogarithmic factor and there is no rate-exponent gap. When $mr < k$ the split exponent grows faster than the joint exponent, and the theorem provides no asymptotic gain. This corresponds to strongly entangled spatio-temporal dynamics — turbulent mixing at small scales, dispersive shocks, coupled reaction-diffusion with space-time-localised coherent structures — where A3 is nearly violated. Dissipative parabolic systems on smooth domains typically satisfy $mr > k$.

C.6 Crossover Point Derivation

Equating the dominant scalings (polylog factors and architecture constants absorbed into C_r),

$$c\,\varepsilon^{-m/k} \;=\; C_r\,(m_x + d_h + p)\;\log(m)\,\varepsilon^{-1/r}. \tag{16}$$

Combining exponents over the common denominator rk,

$$\varepsilon^{(k - mr)/(rk)} \;=\; \frac{C_r\,(m_x + d_h + p)\;\log(m)}{c}.$$

Under $mr > k$ the left exponent is negative, so isolating ε (raising both sides to $rk/(k - mr) < 0$, i.e. inverting the base) gives

$$\boxed{\varepsilon^* = \left[\frac{c}{C_r\,(m_x + d_h + p)\,\log(m)}\right]^{rk/(mr-k)}}. \tag{17}$$

Since the bracket lies in $(0, 1)$ in the relevant regime and the exponent is positive, $\varepsilon^* \in (0, 1)$ as required. When $mr = k$ the curves are parallel in log–log scale (separated only by the polylog factor); when $mr < k$ the formal ε^* falls outside the admissible range, signalling no rate-exponent gain in the entangled regime.

C.7 Scope of the Factorisable Subclass: Most PDE Systems Fit

A natural objection to the conditional framing of Theorem 1 is that the separation only applies when the target lies in $\mathcal{F}_{r,\varepsilon_{\text{floor}}}(m)$, which on its face looks restrictive. In fact, for the PDE families that motivate operator learning, membership in $\mathcal{F}_{r,\varepsilon_{\text{floor}}}$ is the rule rather than the exception. We collect the supporting observations here.

The three structural ingredients. A target $F = A_t^* \circ A_x^* + \delta$ lies in $\mathcal{F}_{r,\varepsilon_{\text{floor}}}(m)$ when: (i) the spatial map A_x^* admits a holomorphic parametrisation with split regularity r; (ii) the temporal map A_t^* admits a holomorphic parametrisation with the same r; (iii) the residual δ between the true latent operator and the factored composition has $L^2(\mu)$ norm bounded by $\varepsilon_{\text{floor}}$.

Each ingredient corresponds to a well-studied structural feature of PDE solution operators.

(i) Holomorphy of the spatial map. Under A2 the parametric input is analytic in the Karhunen–Loève coordinates with exponentially decaying coefficients. For elliptic and parabolic problems with analytic coefficients — the canonical setting of the Cohen–DeVore–Schwab and parametric-PDE literature — the solution operator inherits (b, ρ, κ)-holomorphy on $[-1, 1]^{m_x}$ as a function of these coordinates. This covers Darcy flow, linear elasticity, Stokes, stationary diffusion, and indeed any PDE whose coefficients depend analytically on a finite number of parameters: the spatial decoder map in any neural-operator surrogate of such a system is holomorphic by construction. The exponential decay in A2 is what makes ℓ^p-summability of the polynomial-chaos coefficients automatic, so the split regularity exponent r is positive and finite.

(ii) Holomorphy of the temporal map. For evolution equations $\partial_t u = Lu + N(u, u)$ with sectorial linear part, the time-T flow map Φ^T is real-analytic in initial conditions on bounded balls of X (this is a standard consequence of the analytic-semigroup theory and the implicit function theorem in Banach scales). The temporal latent factor A_t^* is the per-mode time evolution under the linearisation, which is even *entire* as a function of mode amplitude (it is just $e^{\lambda_k T}$). Holomorphy of A_t^* is therefore essentially free for parabolic, hypoelliptic, and weakly damped hyperbolic systems on smooth domains. The same holds for ODE-reduced representations after POD or other Galerkin truncation: a finite-dimensional ODE with polynomial or analytic right-hand side has an analytic flow.

(iii) The residual floor and assumption A3. Lemma C.2 bounds δ by $C_\delta\,\mathrm{Tr}(\Gamma_\mu)$, with $C_\delta^2 = C_1 T^2 e^{6LT}$. For any input distribution satisfying A2 with $\ell > 0$, $\mathrm{Tr}(\Gamma_\mu) < \infty$, so $\varepsilon_{\text{floor}}$ is finite; and as the input becomes spectrally sharper ($\ell \to \infty$), $\varepsilon_{\text{floor}} \to 0$. The relevant question is then whether the linear-nonlinear split is informative on the time horizon T. Three regimes are useful:

- *Dissipative parabolic / strongly damped systems* (heat, Allen–Cahn, Fisher–KPP, reaction-diffusion above the spectral gap, Burgers with sufficient viscosity, Navier–Stokes below the energy threshold of turbulence): $L < 0$ on the relevant subspace, Te^{LT} stays bounded uniformly in T, and $\varepsilon_{\text{floor}}$ is small for all practical horizons. These systems sit comfortably in $\mathcal{F}_{r,\varepsilon_{\text{floor}}}$.
- *Weakly nonlinear and slow-manifold systems* (Kuramoto–Sivashinsky in the weakly chaotic regime, Korteweg–de Vries with smooth initial data, Schrödinger in the dispersive regime, advection–diffusion with bounded Péclet number, low-Mach Euler): the nonlinear part is genuinely non-trivial but slow compared to the linear semigroup, so the Duhamel residual is small relative to the linear evolution and $\varepsilon_{\text{floor}}$ remains controllable.

- *Conservation laws and weakly hyperbolic systems with analytic coefficients*: as long as solutions remain smooth (no shocks formed in $[0, T]$), the flow map is holomorphic in initial data, A3 holds with a horizon-dependent constant, and the system fits.

The systems that do *not* fit are the strongly entangled ones flagged after the boundary-regime remark: turbulence at fully developed scales, dispersive shocks, hyperbolic systems past wave-breaking, strongly coupled multi-scale reaction-diffusion with space-time-localised coherent structures. These are precisely the regimes where any factorisation-based surrogate (POD-Galerkin, DEIM, reduced-basis, neural-operator splits, including ours) is known empirically to struggle. The theorem's conditional scope therefore aligns with the empirical scope of factorisation-based methods.

C.8 Proof of Theorem 2

Lemma 4 (Boundedness of the LinOSS-IMEX Evolution)**.** *Consider the homogeneous LinOSS system discretised via the IMEX scheme with step size $\Delta t > 0$ and diagonal matrix $\Lambda = \mathrm{diag}(\lambda_k)$ satisfying $\lambda_k > 0$ for all k. Assume further that the strict stability condition $0 < \Delta t^2 \lambda_k < 4$ is satisfied for all k. Then the transition matrix M_{IMEX} preserves a positive definite quadratic invariant $\tilde{H}(x) = x^\top P x$, i.e.,*

$$\tilde{H}(x_n) = \tilde{H}(x_0) \qquad \forall n \geq 0, \tag{18}$$

for some symmetric matrix $P \succ 0$. Moreover, all eigenvalues μ_j of M_{IMEX} satisfy $|\mu_j| = 1$, and the iterates remain uniformly bounded:

$$\|x_n\|_2 \;\leq\; \sqrt{\frac{\lambda_{\max}(P)}{\lambda_{\min}(P)}}\, \|x_0\|_2 = C_{\mathrm{IMEX}} \|x_0\|_2 \qquad \forall\, n \geq 0, \tag{19}$$

where $\lambda_{\min}(P)$ and $\lambda_{\max}(P)$ denote the extremal eigenvalues of P. Equivalently,

$$\|M_{\mathrm{IMEX}}^n\|_2 \leq C_{\mathrm{IMEX}} \qquad \forall n \geq 0.$$

Proof. Under the stability condition $0 < \Delta t^2 \lambda_k < 4$, the LinOSS-IMEX update preserves a modified quadratic Hamiltonian $\tilde{H}(x) = x^\top P x$ with $P \succ 0$; see Proposition E.1 of Rusch & Rus (2025). Hence,

$$x_n^\top P x_n = x_0^\top P x_0 \qquad \forall\, n \geq 0. \tag{20}$$

Since $P \succ 0$, its associated quadratic form is equivalent to the Euclidean norm:

$$\lambda_{\min}(P)\|x\|_2^2 \leq x^\top P x \leq \lambda_{\max}(P)\|x\|_2^2 \qquad \forall x. \tag{21}$$

Applying this bound to x_n and x_0, and using invariance of $\tilde{H}$, yields

$$\lambda_{\min}(P)\|x_n\|_2^2 \leq x_n^\top P x_n = x_0^\top P x_0 \leq \lambda_{\max}(P)\|x_0\|_2^2. \tag{22}$$

Taking square roots gives

$$\|x_n\|_2 \leq \sqrt{\frac{\lambda_{\max}(P)}{\lambda_{\min}(P)}}\|x_0\|_2 = C_{\mathrm{IMEX}}\|x_0\|_2.$$

Since $x_n = M_{\mathrm{IMEX}}^n x_0$ and the bound holds for every x_0, it follows that $\|M_{\mathrm{IMEX}}^n\|_2 \leq C_{\mathrm{IMEX}}$ for all $n \geq 0$. The unit-modulus eigenvalue statement follows from the same LinOSS-IMEX stability result in Proposition E.1 of Rusch & Rus (2025). □

Lemma 5 (Energy Decay of Contractive Temporal Cores)**.** *Consider a homogeneous linear recurrent temporal core with transition matrix M_{diss}. Assume that M_{diss} is contractive in the sense that there exist constants $C_{\mathrm{diss}} \geq 1$ and $0 < \rho < 1$ such that*

$$\|M_{\mathrm{diss}}^n\|_2 \leq C_{\mathrm{diss}}\rho^n \qquad \forall n \geq 0.$$

Then, for any initial state x_0, the trajectory $x_n = M_{\mathrm{diss}}^n x_0$ satisfies

$$\|x_n\|_2 \leq C_{\mathrm{diss}}\rho^n\|x_0\|_2 \qquad \forall n \geq 0. \tag{23}$$

In particular, if $\|x_0\|_2 = C_0$, then $\|x_n\|_2 \to 0$ as $n \to \infty$. Moreover, when compared with an idealized energy-preserving reference trajectory s_n satisfying $\|s_n\|_2 = C_0$ for all $n \geq 0$ and sharing the same initial condition $s_0 = x_0$, the Euclidean energy error $E_n := \big|\|s_n\|_2 - \|x_n\|_2\big|$ obeys

$$E_n \geq C_0\big(1 - C_{\mathrm{diss}}\rho^n\big)_+ \qquad \forall n \geq 0. \tag{24}$$

Consequently, $E_n \to C_0$ as $n \to \infty$.

Proof. By the assumed matrix power bound,

$$\|x_n\|_2 = \|M_{\text{diss}}^n x_0\|_2 \leq \|M_{\text{diss}}^n\|_2 \|x_0\|_2 \leq C_{\text{diss}} \rho^n \|x_0\|_2.$$

If $\|x_0\|_2 = C_0$, then $C_{\text{diss}} \rho^n C_0 \to 0$ because $0 < \rho < 1$, and hence $\|x_n\|_2 \to 0$.

Now let s_n be an idealized energy-preserving reference trajectory with $\|s_n\|_2 = C_0$ for all $n \geq 0$. Then

$$E_n = \big|\|s_n\|_2 - \|x_n\|_2\big| = C_0 - \|x_n\|_2,$$

whenever $\|x_n\|_2 \leq C_0$. In general, using the bound above gives

$$E_n \geq C_0 - \|x_n\|_2 \geq C_0 - C_{\text{diss}} \rho^n C_0 = C_0(1 - C_{\text{diss}} \rho^n).$$

Since $E_n \geq 0$, this yields

$$E_n \geq C_0 \big(1 - C_{\text{diss}} \rho^n\big)_+.$$

Finally, because $C_{\text{diss}} \rho^n \to 0$, we obtain $E_n \to C_0$. □

Theorem (Temporal Approximation Bounds)**.** Let s_n and $\hat{s}_n$ denote the autonomous (zero-input) latent trajectories governed by the true physical operator Ψ and the learned temporal core $\hat{\Psi}$, respectively. We assume initial state alignment ($s_0 = \hat{s}_0$) with initial energy $\|s_0\|_2 = C_0 > 0$, and assume the ground truth trajectory is energy-preserving in the idealized Euclidean sense $\|s_n\|_2 = C_0$ for all $n \geq 0$. Let the point-wise state error be $e_n := s_n - \hat{s}_n$, and define the energy error as $E_n := \big|\|s_n\|_2 - \|\hat{s}_n\|_2\big|$. Assume the discrete transition matrices satisfy the local Lipschitz bound $\|M(\Psi) - M(\hat{\Psi})\|_2 \leq L_M \|\Psi - \hat{\Psi}\|_2$. Then the following bounds hold for $n \geq 0$:

1. **LinOSS (non-dissipative case).** Let $\hat{\Psi} = \hat{\Lambda}$. Under the assumptions of Appendix Lemma 4, assume the LinOSS transition is uniformly power-bounded, i.e., $\|M(\Psi)^j\|_2 \leq C_{\text{IMEX}}$ and $\|M(\hat{\Psi})^j\|_2 \leq C_{\text{IMEX}}$ for all $j \geq 0$. Then the point-wise error grows at most linearly with time and is bounded by the accuracy of the learned parameters:

$$\|e_n\|_2 \leq C_{\text{IMEX}}^2 \, n \, L_M \, C_0 \, \|\Psi - \hat{\Psi}\|_2. \tag{25}$$

   Consequently, since $E_n \leq \|e_n\|_2$, the Euclidean energy error is also controlled by the parametric mismatch.

2. **Stable Dissipative Cores (contractive case).** Let $M(\hat{\Psi})$ define any dissipative linear recurrent core satisfying the matrix power bound $\|M(\hat{\Psi})^k\|_2 \leq C_{\text{diss}} \rho^k$ with $0 < \rho < 1$. Then the learned trajectory incurs an irreducible structural energy error due to artificial dissipation:

$$E_n \geq C_0 \big(1 - C_{\text{diss}} \rho^n\big)_+. \tag{26}$$

   As $n \to \infty$, the temporal energy error converges to the total energy of the system, $E_n \to C_0$, demonstrating a loss of the underlying signal caused by the contractive structure.

Proof. **Part 1 (LinOSS, parametric bound).** Write $s_n = M(\Psi)^n s_0$ and $\hat{s}_n = M(\hat{\Psi})^n s_0$ (autonomous, $s_0 = \hat{s}_0$). Then $e_n = (M(\Psi)^n - M(\hat{\Psi})^n) s_0$. The telescoping identity

$$M(\Psi)^n - M(\hat{\Psi})^n = \sum_{j=0}^{n-1} M(\Psi)^{n-1-j} \big(M(\Psi) - M(\hat{\Psi})\big) M(\hat{\Psi})^j$$

combined with Appendix Lemma 4 ($\|M(\Psi)^j\|_2, \|M(\hat{\Psi})^j\|_2 \leq C_{\text{IMEX}}$) gives

$$\|e_n\|_2 \leq \sum_{j=0}^{n-1} C_{\text{IMEX}}^2 \|M(\Psi) - M(\hat{\Psi})\|_2 \|s_0\|_2.$$

Applying the local Lipschitz bound $\|M(\Psi) - M(\hat{\Psi})\|_2 \leq L_M \|\Psi - \hat{\Psi}\|_2$ and $\|s_0\|_2 = C_0$,

$$\|e_n\|_2 \leq C_{\text{IMEX}}^2 \, n \, L_M \, C_0 \, \|\Psi - \hat{\Psi}\|_2.$$

The right-hand side $\to 0$ as $\hat{\Psi} \to \Psi$, so the error in Part 1 is controlled by parametric mismatch only and is removable in the limit. $E_n \leq \|e_n\|_2$ follows from the reverse triangle inequality.

Part 2 (contractive cores, structural lower bound). Fix any $\hat{\Psi}$ in the class $\mathcal{C}_\rho$ of cores satisfying $\|M(\hat{\Psi})^k\|_2 \le C_{\text{diss}}\rho^k$ for all $k \ge 0$. By Appendix Lemma 5 applied to $\hat{s}_n = M(\hat{\Psi})^n s_0$,

$$\|\hat{s}_n\|_2 \le \|M(\hat{\Psi})^n\|_2 \|s_0\|_2 \le C_{\text{diss}}\rho^n C_0.$$

This bound is *uniform* over $\hat{\Psi} \in \mathcal{C}_\rho$: the contraction radius ρ is by hypothesis a structural property of the core class, not of the learned parameters within it. Combining with $\|s_n\|_2 = C_0$,

$$E_n = \big|\|s_n\|_2 - \|\hat{s}_n\|_2\big| = C_0 - \|\hat{s}_n\|_2 \ge C_0 - C_{\text{diss}}\rho^n C_0 = C_0(1 - C_{\text{diss}}\rho^n),$$

whenever the right-hand side is non-negative. Taking the infimum over $\hat{\Psi} \in \mathcal{C}_\rho$ leaves the lower bound unchanged because it does not depend on the particular $\hat{\Psi}$ chosen:

$$\inf_{\hat{\Psi}\in\mathcal{C}_\rho} E_n \ge C_0\big(1 - C_{\text{diss}}\rho^n\big)_+.$$

Since $\rho^n \to 0$, $\inf_{\hat{\Psi}} E_n \to C_0$ as $n \to \infty$: no choice of parameters within the contractive class can match the energy-preserving reference asymptotically.

Comparison. Part 1 shows the LinOSS bound vanishes as $\|\Psi - \hat{\Psi}\|_2 \to 0$ (parametric, removable). Part 2 shows the contractive bound is strictly positive for all $n \ge 1$ once $1 > C_{\text{diss}}\rho^n$, uniformly across the class $\mathcal{C}_\rho$ (structural, irreducible). The two error sources are therefore qualitatively distinct: training removes the first but not the second. □

Remark (Scope of Theorem 2). Theorem 2 concerns regimes where the latent reference dynamics are energy-preserving in the sense $\|s_n\|_2 = C_0$. These are the regimes targeted by the oscillatory inductive bias of LinOSS: wave propagation, advection-dominated transport, and weakly dissipative systems where long-horizon prediction depends on phase preservation. Strongly dissipative parabolic regimes are *not* in the scope of Part 2: there a contractive core correctly matches the structural decay of the reference and the lower bound becomes vacuous (the reference itself contracts).

D Extended Related Work

We expand the three-paragraph related-work treatment in 2 into five thematic threads. Throughout, we keep PDON as the reference frame and indicate, for each prior work, which subset of $\{\mathcal{E}_{\text{in}}, \mathcal{L}_{\text{in}}, \mathcal{F}_{\text{temp}}, \mathcal{L}_{\text{out}}, \mathcal{D}_{\text{out}}\}$ it instantiates.

D.1 Implicit and Monolithic Neural Operators

The implicit family treats time as a coordinate or as one axis of a joint space-time tensor.

DeepONet and its descendants. Lu et al. (2021) introduced DeepONet as a finite realisation of Chen & Chen (1995)'s universal-approximation theorem for nonlinear operators. Variants in this tradition include POD-DeepONet (Lu et al., 2022) (POD basis trunk), MIONet (Jin et al., 2022), BasisONet, and physics-informed DeepONet (Wang et al., 2021b); many of these still concatenate t into the trunk. Separable physics-informed DeepONet factorises the trunk dimensions but does not change the monolithic-time treatment. Mathematically they all produce $\sum_k b_k(a)t_k(y)$ with $y = (x, t)$.

FNO and spectral neural operators. FNO (Li et al., 2021) parameterises a kernel integral as a low-mode multiplication in the Fourier domain. Variants include FFNO (Tran et al., 2023) (factorisation across axes), F-FNO (Tran et al., 2023), U-FNO (Wen et al., 2022), geo-FNO (Li et al., 2023b) (irregular geometry), and Multipole GNO (Li et al., 2020). All originally treat the joint (x, t) tensor as one input; LNO (Cao et al., 2024) replaces Fourier with Laplace.

Transformer operators. Galerkin Transformer (Cao, 2021) reformulates self-attention without the softmax, drawing parallels to Galerkin projections; OFormer (Li et al., 2023a), GNOT (Hao et al., 2023), OnFormer, and Transolver are subsequent extensions. Continuum-attention operators (Calvello et al., 2025) and ViT-style operators (Ovadia et al., 2024) also belong here.

U-Net and convolutional operators. researchers apply U-Nets directly on PDE fields; PINN-style models (Raissi et al. (2019)) use coordinate networks but are typically trained per-instance, not as operators.

These models are generally implicit about time and pay the joint-dimension cost in approximation theory. PDON treats them as candidate $\mathcal{E}_{\text{in}}$ implementations rather than as competing complete pipelines.

D.2 Explicit Factorised and Latent Operators

The explicit family separates spatial encoding from temporal evolution.

POD- and Latent-DeepONet variants. POD-DeepONet (Lu et al., 2022), L-DeepONet (Kontolati et al., 2024), RONOM (Dummer et al., 2026), LaMO (Tiwari et al., 2025), SPOD-DeepONet, and Multivariate POD-DeepONet all instantiate $(\mathcal{E}_{\text{in}}, \mathcal{D}_{\text{out}}) = (\text{POD}, \text{POD}^\top)$ with various $\mathcal{F}_{\text{temp}}$ choices. Latent-Twins provide a related theoretical framing.

NO + temporal model hybrids. S-DeepONet (He et al., 2024) embeds an RNN inside the branch. Michałowska et al. (2024) stack DeepONet $\to$ RNN $\to$ DeepONet. DeepO-Mamba (Hu et al., 2025) pairs POD-DeepONet with Mamba. MemNO (Buitrago Ruiz et al., 2025) pairs FNO with S4. Temporal NO (Diab & Al Kobaisi, 2025) adds a temporal branch to DeepONet. PDE-Refiner (Lippe et al., 2023) uses a diffusion temporal core. All of these factor through the PDON skeleton.

Reduced-order modelling. Classical reduced-order modelling (Benner et al., 2015; Quarteroni et al., 2015) uses POD/DEIM/operator-inference; Maulik et al. (2021) and Hesthaven & Ubbiali (2018) bridge ROM with neural networks. RONOM (Dummer et al., 2026) formalises this connection.

D.3 Temporal Cores in Operator Learning

Recurrent and gated. LSTM (Hochreiter & Schmidhuber, 1997), GRU (Cho et al., 2014), and stacked variants are accurate for short sequences but un-parallelisable.

State-space models. S4 (Gu et al., 2022), S5 (Smith et al., 2023), LRU (Orvieto et al., 2023), LinOSS (Rusch & Rus, 2025), and Diagonal-S4 (Gupta et al., 2022) discretise a continuous-time linear ODE with parallel-scan inference. H3 (Fu et al., 2023) and Hyena (Poli et al., 2023) extend the SSM family with implicit convolutions.

Selective SSMs. Mamba (Gu & Dao, 2024) introduces input-dependent state matrices. Mamba-2 (Dao & Gu, 2024), Jamba (Lieber et al., 2024), and Hymba (Dong et al., 2025) offer efficiency variants. Mamba neural operator (Cheng et al., 2025) is a dedicated PDE adaptation.

Transformers and linear attention. Galerkin Transformer (Cao, 2021), Performer (Choromanski et al., 2021), and Linformer (Wang et al., 2020) achieve linear attention; ViViT (Arnab et al., 2021) introduces factorised space-time attention; Universal Transformers (Dehghani et al., 2018) share parameters across layers.

Neural ODEs and continuous time. Neural ODE (Chen et al., 2018), Latent ODE (Rubanova et al., 2019), and ODE-RNN (Rubanova et al., 2019) describe latent dynamics with continuous-time ODEs. RONOM (Dummer et al., 2026) uses Neural ODEs as $\mathcal{F}_{\text{temp}}$.

Another group of works provide a comprehensive comparison of these temporal cores on dynamical-system tasks; we extend their analysis to spatio-temporal PDE operator learning in §5.1.

D.4 Memory and Mori-Zwanzig

The role of memory in PDE operators has a rich pre-deep-learning history (Mori, 1965; Zwanzig, 1961; Nakajima, 1958; Chorin et al., 2002). Ma et al. (2018) draw an analogy between LSTM gates and Mori-Zwanzig kernels. Lippe et al. (2023) and Tran et al. (2023) report ablations suggesting that

the Markov assumption suffices when the observation grid is dense; Buitrago Ruiz et al. (2025) show that this finding reverses under partial observation, motivating MemNO. Another study Markovian neural operators directly. PDON's $\mathcal{F}_{\text{temp}}$ subsumes both the Markovian (single-step) and the memory (multi-step) regimes by the choice of temporal core.

D.5 Resolution-Invariance Literature

Discretisation-invariant operators. FNO (Li et al., 2021) is the canonical example; the spectral convolution generalises across grids subject to a mode budget. Kovachki et al. (2024) study discretisation invariance theoretically. Karniadakis et al. (2021) survey continuous-time PINN-style methods.

Encoder/decoder bridges. RONOM (Dummer et al., 2026) proves error bounds across grids. Dummer et al. (2026) also study time super-resolution. Mesh-free DeepONet treats the trunk as a coordinate map. Our band-limited Fourier projector $P_{N' \leftarrow N}$ in §5.3 is a fixed, falsifiable counterpart to a learned bridge.

Multi-resolution training. Another group of works train across grids to improve transfer. PDON requires no multi-resolution training: when the PON is spectral, transfer is automatic; when the PON is locked, the band-limited projector handles the bridge.

E Theoretical Complexity Analysis Table

Table 4 compares per-trajectory training and inference complexities for the major operator-learning architectures, parameterised in: N_x (spatial grid size, possibly per axis for $d > 1$), N_t (temporal grid size), d_h (latent hidden dimension), M (spectral mode budget per axis), r (POD truncation rank), L (number of stacked operator blocks), C (input/output channel count). For models with internal latent compression we report compute in the latent space; for FNO-family models we report compute as if a full FFT were taken at the layer's input grid.

Table 4: Asymptotic complexity of representative operator-learning architectures.

Model	Spatial path	Temporal path	Train fwd / step	Memory
Vanilla DeepONet	branch MLP	t in trunk	$\mathcal{O}(N_x N_t \cdot p)$	$\mathcal{O}(N_x N_t)$
POD-DeepONet	branch MLP	POD basis (fixed)	$\mathcal{O}(N_x r)$	$\mathcal{O}(rN_x + rN_t)$
L-DeepONet	autoencoder	latent MLP	$\mathcal{O}(N_x d_h + d_h^2 N_t)$	$\mathcal{O}(d_h N_t)$
RONOM	autoencoder	Neural ODE	$\mathcal{O}(N_x d_h + N_t d_h^2)$	$\mathcal{O}(d_h N_t)$
$(d{+}1)$-D FNO	spectral conv	joint space-time	$\mathcal{O}(L(N_x^d N_t \log(N_x^d N_t)W + M^{d+1}W^2))$	$\mathcal{O}(LWN_x^d N_t)$
FFNO	factorized spectral	joint space-time	$\mathcal{O}(L(d+1)(N_x^d N_t \log(N_x^d N_t)W + MW^2))$	$\mathcal{O}(LWN_x^d N_t)$
LNO	Laplace conv	joint space-time	$\mathcal{O}(L(N_x^d N_t \log(N_x^d N_t)W + M^{d+1}W^2))$	$\mathcal{O}(LWN_x^d N_t)$
MemNO	FNO per slice	S4 along t	$\mathcal{O}(LN_t(N_x^d \log N_x^d W + M^d W^2) + d_h N_t \log N_t)$	$\mathcal{O}(LWN_x^d + d_h N_t)$
S-DeepONet	branch MLP	RNN in branch	$\mathcal{O}(N_x N_t d_h)$	$\mathcal{O}(d_h N_t)$
Michalowska	DeepONet emb.	RNN bridge	$\mathcal{O}(N_x d_h + N_t d_h^2)$	$\mathcal{O}(d_h N_t)$
DeepO-Mamba	POD branch	Mamba scan	$\mathcal{O}(N_x r + N_t d_h^2)$	$\mathcal{O}(d_h N_t + N_x^d N_t)$
ViViT-fact.	spatial Trans.	temporal Trans.	$\mathcal{O}(N_t N_x^2 W + N_x N_t^2 W)$	$\mathcal{O}(W(N_x^2 + N_t^2))$
Galerkin Trans.	linear attention	temporal Trans	$\mathcal{O}(LN_x N_t W^2)$	$\mathcal{O}(LWN_x N_t)$
GNOT	cross-attention	temporal Trans	$\mathcal{O}(LN_x N_t W^2)$	$\mathcal{O}(LWN_x N_t)$
PDON-LinOSS	PON (r or M)	LinOSS scan	$\mathcal{O}(N_x^d r N_t + d_h N_t)$	$\mathcal{O}(d_h N_t + N_x^d N_t)$
PDON-S5	PON (r or M)	S5 scan	$\mathcal{O}(N_x^d r N_t + d_h N_t \log N_t)$	$\mathcal{O}(d_h N_t + N_x^d N_t)$
PDON-LRU	PON (r or M)	LRU scan	$\mathcal{O}(N_x^d r N_t + d_h N_t)$	$\mathcal{O}(d_h N_t + N_x^d N_t)$
PDON-Mamba	PON (r or M)	selective SSM	$\mathcal{O}(N_x^d r N_t + d_h^2 N_t)$	$\mathcal{O}(d_h N_t + N_x^d N_t)$

The headline implication of Table 4 is that the $(d{+}1)$-D FNO and Galerkin/GNOT operators carry an $N_x^d N_t$ factor inside their per-step cost, while PDON variants pay $N_x r + d_h N_t$ — linear in N_x and N_t separately rather than in the product, exactly matching the theoretical separation in Theorem 1.

F MATHEMATICAL EXPLANATION OF EXPERIMENTS

For each benchmark suite we record the governing PDE, the analytical mechanism by which a labelled trajectory is generated, and the mathematical reason a particular model family is exercised. Hyperparameters, mode counts, hidden widths, depths, and dataset sizes are deferred to Appendix G and to the experimental-protocol section that follows; here we keep the discussion at the level of equations and inductive bias.

Local notation. To prevent collision with the global symbols of Appendix A (where k denotes the joint regularity exponent, δ the interaction residual, $h(t)$ the SSM hidden state, and K enters the covariance kernel), we use the subscripted forms k_{rx}, δ_{KdV}, $h_{\mathrm{tmp}}(t)$, K_w, and $a_{\mathrm{B}}, b_{\mathrm{B}}$ in the corresponding subsections. No new entry is added to the Notation Summary.

F.1 FORCED BURGERS (1+1)D

We consider the forced viscous Burgers equation with viscosity $\nu = 10^{-3}$,

$$u_t + u\,u_x - \nu u_{xx} = f(t), \qquad (x,t) \in (-L, L) \times (T_0, T_0 + T). \tag{27}$$

with $L = 5$, $T_0 = 0.5$, $T = 1$, and a temporally varying source $f(t)$. The supervised target is the operator

$$\mathcal{G} : [u(L,t),\, u(-L,t),\, u(\cdot, T_0),\, f(t)] \mapsto u(\cdot,\cdot), \tag{28}$$

which differs from a pure initial-value map in that two boundary traces and an external forcing time-series are also supplied as inputs.

Analytical construction of trajectories. Solutions are constructed analytically from a scalar potential $g(t)$, drawn so that $f(t) = g''(t)$. A smooth $f(t)$ is sampled from a Gaussian random field with squared-exponential kernel of length-scale 0.2, and $g(t)$ is recovered from a numerical antiderivative. Four solution families then label the dataset:

$$\text{(A)} \quad u = g'(t), \tag{29}$$

$$\text{(B)} \quad u = \tfrac{x+c_1}{t} + \tfrac{t g'(t) - g(t)}{t}, \tag{30}$$

$$\text{(C)} \quad u = -\sqrt{2c_1}\,\tanh\Big[\tfrac{1}{2}\big(\sqrt{2c_1}(x - g(t)) + \sqrt{2c_1}\,c_2\big)\Big] + g'(t), \tag{31}$$

$$\text{(D)} \quad u = -\tfrac{1}{t}\sqrt{2c_1}\,\tanh\Big[\tfrac{1}{2}\big(\sqrt{2c_1}\tfrac{x-g(t)}{t} + \sqrt{2c_1}\,c_2\big)\Big] + \tfrac{x-g(t)}{t} + g'(t), \tag{32}$$

with $c_1 \sim \mathcal{U}(0,3)$, $c_2 \sim \mathcal{U}(-3,3)$. Labelled samples are split equally over the four families so that the operator must learn to switch between linear-drift, rational, and travelling-wave regimes given only the boundary, initial, and forcing inputs.

F.2 FORCED KDV (1+1)D

Inspired by DeepOMamba (Hu et al., 2025), we use the forced Korteweg–de Vries equation

$$u_t + u\,u_x + \beta\,u_{xxx} = f_x, \qquad (x,t) \in (-L, L) \times (0, T), \tag{33}$$

with $L = 5$, $T = 4$. Well-posedness on the interval requires three boundary traces ($u(L,t)$, $u(-L,t)$, $\partial_x u(L,t)$) and one initial condition $u(\cdot, 0)$; the operator we approximate is

$$\mathcal{G} : [u(L,t),\, u(-L,t),\, \partial_x u(L,t),\, u(x,0),\, \partial_x f(x,t),\, \alpha,\, \beta] \mapsto u(x,t). \tag{34}$$

Soliton and lump trajectories. For certain $f(x,t)$, its solution can be given analytically. Concretely, there are three types of solutions under consideration.

Type (A) solution is given by

$$\begin{cases} f(x,t) = \dfrac{\frac{12k\beta}{\alpha}\left(k^3(4\beta - \delta) - \frac{bA}{1+A^2t^2} - b_1\right)}{\cosh^2(k(x - \delta k^2 t) - (b\arctan(At) + b_1 t + b_0))}, \\[2ex] u(x,t) = \dfrac{12\beta k^2}{\cosh^2(k(x - 4k^2 t) - b\arctan(At) + b_1 t + b_0)}. \end{cases} \tag{A}$$

Here, $k, A, \alpha, \beta, \delta, b_1, b_0, b$ are the KdV equation's coefficients, which we can manipulate to generate rich input and output functions.

Furthermore, the Type (B) solution is given by

$$\begin{cases} f(x,t) = \dfrac{\frac{12k\beta}{\alpha}\left(k^3(4\beta-\delta) - (2b_2t+b_1)\exp(b_2t^2+b_1t+b_0)\right)}{\cosh^2(k(x-\delta k^2t) - \exp(b_2t^2+b_1t+b_0))}, \\ u(x,t) = \dfrac{12\beta k^2}{\cosh^2(k(x-\delta k^2t) - \exp(b_2t^2+b_1t+b_0))}. \end{cases} \tag{B}$$

Here, $k, A, \alpha, \beta, \delta, b_2, b_1, b_0$ are the KdV equation's coefficients, which we can manipulate to generate rich input and output functions.

Lastly, the Type (C) solution is given by

$$\begin{cases} f(x,t) = \dfrac{12\beta\gamma}{\alpha}\left(\dfrac{6\beta(\gamma+1)}{F(x,t)^2} - \dfrac{8\beta(x+a)^2}{H(x,t)^2F(x,t)^3} - \dfrac{(t+b)H(x,t)^2}{(x+a)F(x,t)} \right. \\ \qquad\qquad \left. - \dfrac{(t+b)H(x,t)^3}{(x+a)^3} - \arctan(H(x,t))\right), \\ u(x,t) = \dfrac{12\beta\gamma}{F(x,t)}. \end{cases} \tag{C}$$

Here

$$F(x,t) = (x+a)^2 + (t+b)^2 + d, \qquad H(x,t) = \frac{x+a}{\sqrt{d+(b+t)^2}},$$

and $a, b, d, \alpha, \beta, \gamma$ are the KdV equations' coefficients.

We generate 27K training samples, 3K validation samples, and 3K test samples for the KdV equation. The dataset is balanced across the three analytical families: one third of the samples are generated from Type (A), one third from Type (B), and one third from Type (C). For all three families, we sample

$$\alpha \sim \mathcal{U}(1.5, 2.5), \qquad \beta \sim \mathcal{U}(0.125, 0.375).$$

For Type (A), the remaining coefficients are sampled as

$$k \sim \mathcal{U}(0.5, 1.5), \quad A \sim \mathcal{U}(2.5, 3.5), \quad b_1 \sim \mathcal{U}(-3, 3), \quad b_0 \sim \mathcal{U}(-1, 1),$$
$$b \sim \mathcal{U}(-1, 1), \qquad \delta \sim \mathcal{U}(-4, 4).$$

For Type (B), we sample

$$k \sim \mathcal{U}(0.5, 1.5), \quad A \sim \mathcal{U}(2.5, 3.5), \quad b_2 \sim \mathcal{U}(-2, -1), \quad b_1 \sim \mathcal{U}(-1, 1),$$
$$b_0 \sim \mathcal{U}(-1, 1), \qquad \delta \sim \mathcal{U}(-4, 4).$$

For Type (C), we sample

$$a \sim \mathcal{U}(-3, 3), \quad b \sim \mathcal{U}(-4, 2), \quad d \sim \mathcal{U}(1, 4), \quad \gamma \sim \mathcal{U}(0.5, 1.5).$$

All trajectories are evaluated on a uniform space–time grid of size 100×100.

F.3 Reaction–Diffusion (1+1)D

The (1+1)D reaction-diffusion benchmark uses

$$D\, u_{xx} + k_{\mathrm{rx}}\, u^2 - \pi^2\, u_t = f(x,t), \qquad (x,t) \in [0,1] \times [0,10], \tag{35}$$

with $D = k_{\mathrm{rx}} = 0.01$ and homogeneous initial-and-boundary data $u(\cdot, 0) = u(0, \cdot) = u(1, \cdot) = 0$. The reaction-rate symbol k_{rx} is renamed locally to avoid collision with the global regularity exponent k.

Two forcing regimes. The dataset contains two configurations of the right-hand side. In the *temporal-only* regime, $f(x,t) = f(t)$ is a draw from a Gaussian process $f(t) \sim \mathcal{GP}\big(0,\, K_l(t_1,t_2)\big)$ with squared-exponential kernel $K_l(t_1,t_2) = \exp\big(-(t_1-t_2)^2/(2l^2)\big)$ and length-scale $l = 0.1$. In the *separable* regime, $f(x,t) = \sin(\omega x)\, g(t)$ with $\omega \sim \mathcal{U}(0, 8\pi)$ and $g(t)$ drawn from the same kernel. The two regimes test whether a single architecture can absorb temporal-only dynamics and a spatially structured product-mode forcing without re-tuning.

F.4 Fourier Reaction–Diffusion (2+1)D

For the periodic Fourier reaction-diffusion suite the equation is

$$\Delta u + u^2 - u_t = f, \qquad x \in [0, 2\pi]^2,\ t \in (0, T],\ , \tag{36}$$

with periodic spatial boundary conditions and zero initial datum.

Manufactured separable solution. Trajectories are built from the ansatz $u(x,t) = g(x)\, h_{\mathrm{tmp}}(t)$, with

$$g(x) = \sum_{\substack{\kappa \in \mathbb{Z}^d \\ \|\kappa\|_\infty \le K_w}} \big[A_\kappa \sin(\kappa^\top x) + B_\kappa \cos(\kappa^\top x)\big], \qquad K_w = 4, \tag{37}$$

and $A_\kappa, B_\kappa \sim \mathcal{U}(0,1)$. The temporal factor satisfies $h_{\mathrm{tmp}}(0) = h'_{\mathrm{tmp}}(0) = 0$, with $h'_{\mathrm{tmp}}(t) \sim \mathcal{GP}(0, K_l)$ at length-scale $l = 0.2$ and $h_{\mathrm{tmp}}(t)$ formed by a numerical antiderivative. The associated forcing is $f(x,t) = \Delta g(x)\, h_{\mathrm{tmp}}(t) + (g(x) h_{\mathrm{tmp}}(t))^2 - g(x)\, h'_{\mathrm{tmp}}(t)$. We use the subscripted symbols $h_{\mathrm{tmp}}, K_w, \kappa$ in place of the source notation h, K, k to avoid collision with the global notation.

F.5 Brusselator (2+1)D, long horizon

The Brusselator system models autocatalysis as a coupled two-species reaction-diffusion process,

$$u_t = D_0\,(u_{xx} + u_{yy}) + a_{\mathrm{B}} + f(t) - (1 + b_{\mathrm{B}})\, u + v u^2, \tag{38}$$

$$v_t = D_1\,(v_{xx} + v_{yy}) + b_{\mathrm{B}}\, u - v u^2, \tag{39}$$

on $[0,1]^2 \times [0, 20]$ with $D_0 = 1$, $D_1 = 0.5$, $a_{\mathrm{B}} = 1$, $b_{\mathrm{B}} = 3$. The constants are renamed from the source notation to free a, d for the Beltrami subsection.

Initial perturbation and long-horizon supervision. Both species start near the homogeneous fixed point with a small spatial perturbation $u(\cdot,0) = v(\cdot,0) = b_{\mathrm{B}}/a_{\mathrm{B}} + 0.1\, J$, where J is a unit-variance Gaussian field on the spatial grid. The substrate concentration is modulated in time by an additive forcing $f(t)$ drawn from the same Gaussian-process kernel as in the reaction-diffusion suite. Although only 39 supervised snapshots are emitted per trajectory, they are the result of a 3000-step integration with step 10^{-3}, so the operator must reproduce a long autonomous rollout from sparse temporal supervision.

F.6 Navier–Stokes Beltrami flow (3+1)D

The largest benchmark exercises the incompressible Navier–Stokes equations in velocity-pressure form,

$$u_t + (u \cdot \nabla)u + \nabla p = \tfrac{1}{\mathrm{Re}}\, \Delta u, \qquad \nabla \cdot u = 0, \qquad x \in [0, 2\pi]^3,\ t \in [0, 1], \tag{40}$$

with Reynolds number Re and prescribed initial-and-boundary data.

Closed-form Beltrami targets. Supervision uses the analytic Beltrami family

$$u = -a\big[e^{ax}\sin(ay \pm dz) + e^{az}\cos(ax \pm dy)\big]\, e^{-d^2 t}, \tag{41}$$

$$v = -a\big[e^{ay}\sin(az \pm dx) + e^{ax}\cos(ay \pm dz)\big]\, e^{-d^2 t}, \tag{42}$$

$$w = -a\big[e^{az}\sin(ax \pm dy) + e^{ay}\cos(az \pm dx)\big]\, e^{-d^2 t}, \tag{43}$$

together with the matching closed-form pressure $p(x,y,z,t)$ (omitted for space), and coefficients $a \sim \mathcal{U}(0,1)$, $d \sim \mathcal{U}(0.5, 1.5)$. The supervised operator therefore maps the (a, d, Re) parameters and the initial field to the entire 3+1D trajectory.

F.7 ERA5 PRESSURE-LEVEL SYSTEM

The ERA5 experiment is based on large-scale atmospheric fields on pressure levels. Unlike the synthetic PDE benchmarks considered elsewhere in this paper, ERA5 is not generated by a single closed-form PDE with known forcing. It is a data-assimilated reanalysis product whose fields are dynamically constrained by a numerical weather prediction model and observations. Nevertheless, the resolved large-scale evolution is governed, to leading order, by the hydrostatic primitive equations on a rotating sphere. This makes ERA5 a natural real-world testbed for operator learning: the model must learn a nonlinear spatiotemporal solution operator for advective, rotational, weakly dissipative atmospheric dynamics.

Let (λ, φ, p, t) denote longitude, latitude, pressure, and time. On a pressure level p, the main prognostic variables are the horizontal wind

$$\mathbf{v}(\lambda, \varphi, p, t) = \begin{bmatrix} u(\lambda, \varphi, p, t) \\ v(\lambda, \varphi, p, t) \end{bmatrix},$$

temperature $T(\lambda, \varphi, p, t)$, geopotential $\Phi(\lambda, \varphi, p, t)$, and vertical pressure velocity $\omega = Dp/Dt$. The horizontal material derivative in pressure coordinates is

$$\frac{D}{Dt} = \frac{\partial}{\partial t} + \mathbf{v} \cdot \nabla_p + \omega \frac{\partial}{\partial p},$$

where ∇_p is the horizontal gradient along an isobaric surface.

A compact form of the hydrostatic primitive equations is

$$\frac{D\mathbf{v}}{Dt} + f\mathbf{k} \times \mathbf{v} = -\nabla_p \Phi + \mathcal{D}_{\mathbf{v}}, \tag{44}$$

$$\nabla_p \cdot \mathbf{v} + \frac{\partial \omega}{\partial p} = 0, \tag{45}$$

$$\frac{\partial \Phi}{\partial p} = -\alpha, \tag{46}$$

$$\frac{DT}{Dt} - \frac{\alpha \omega}{c_p} = Q + \mathcal{D}_T. \tag{47}$$

Here $f = 2\Omega \sin \varphi$ is the Coriolis parameter, $\mathbf{k}$ is the local vertical unit vector, $\alpha = 1/\rho$ is specific volume, c_p is the specific heat at constant pressure, Q represents diabatic heating, and $\mathcal{D}_{\mathbf{v}}, \mathcal{D}_T$ collect unresolved diffusion, subgrid-scale forcing, and parameterised physical processes. The geopotential Φ is related to geopotential height Z by

$$\Phi = gZ,$$

where g is gravitational acceleration.

For the single-pressure-level setting used in our ERA5 benchmark, we do not attempt to explicitly model the full vertical coupling in (44)–(47). Instead, we treat the selected pressure level, for example $p = 500\,\mathrm{hPa}$, as a two-dimensional spherical field whose dynamics are induced by the full three-dimensional atmosphere. The resulting reduced system can be interpreted as a forced, dissipative horizontal transport problem:

$$\partial_t q + \mathbf{v} \cdot \nabla_p q = \mathcal{F}_q + \mathcal{D}_q, \tag{48}$$

where q denotes one of the pressure-level variables, such as geopotential, temperature, or wind components. The effective forcing term $\mathcal{F}_q$ contains vertical coupling, diabatic effects, and unresolved physical processes not explicitly represented in the single-level state.

A particularly useful interpretation at $500\,\mathrm{hPa}$ is obtained from large-scale balanced dynamics. In midlatitudes, the horizontal wind is often approximately geostrophic:

$$f\mathbf{k} \times \mathbf{v}_g = -\nabla_p \Phi, \tag{49}$$

or equivalently

$$u_g = -\frac{1}{f}\frac{\partial \Phi}{\partial y}, \qquad v_g = \frac{1}{f}\frac{\partial \Phi}{\partial x}. \tag{50}$$

Thus the geopotential field determines a balanced horizontal flow. Introducing a streamfunction ψ such that

$$u_g = -\frac{\partial \psi}{\partial y}, \qquad v_g = \frac{\partial \psi}{\partial x},$$

the geopotential is proportional to the streamfunction:

$$\psi \approx \frac{\Phi}{f}.$$

The corresponding quasi-geostrophic potential-vorticity dynamics can be written schematically as

$$\partial_t q_{\mathrm{QG}} + J(\psi, q_{\mathrm{QG}}) = \mathcal{F} + \mathcal{D}, \tag{51}$$

where

$$J(\psi, q) = \frac{\partial \psi}{\partial x}\frac{\partial q}{\partial y} - \frac{\partial \psi}{\partial y}\frac{\partial q}{\partial x}$$

is the horizontal Jacobian and q_{QG} denotes quasi-geostrophic potential vorticity. In a single-layer barotropic approximation, this reduces to the barotropic vorticity equation

$$\partial_t \zeta + J(\psi, \zeta + f) = \mathcal{F} + \mathcal{D}, \tag{52}$$

where

$$\zeta = \nabla^2 \psi$$

is relative vorticity. Equations (51) and (52) explain why $500\,\mathrm{hPa}$ geopotential and wind fields exhibit coherent wave-like and advective structures: ridges, troughs, Rossby-wave propagation, and vortical transport all arise from the interaction between rotation, pressure gradients, and horizontal advection.

This setting is therefore well aligned with the motivation of PDON. A monolithic space–time model must learn the joint map

$$(\lambda, \varphi, t) \mapsto a(\lambda, \varphi, t)$$

directly over the full grid. By contrast, PDON separates spatial representation from temporal evolution:

$$a_n \xrightarrow{\mathcal{E}_{\mathrm{in}}} z_n \xrightarrow{\mathcal{F}_{\mathrm{temp}}} z_{n+1} \xrightarrow{\mathcal{D}_{\mathrm{out}}} \hat{a}_{n+1}.$$

For ERA5, this factorisation is physically meaningful because the dominant large-scale atmospheric structures are coherent in space and persist over time through transport and wave propagation. In particular, the oscillatory temporal bias of LinOSS is compatible with the approximate energy-preserving and phase-propagating behaviour of balanced midlatitude dynamics, while still allowing damping and forcing through the learned input/output maps and residual components.

G Architecture Details and Design

G.1 Model Instantiations Across Benchmark Suites

The PDON skeleton in Eq. (1) is used throughout all experiments, but the spatial encoder–decoder pair $(\mathcal{E}_{\mathrm{in}}, \mathcal{D}_{\mathrm{out}})$ is adapted to the structure and dimensionality of each benchmark. In all cases, the temporal core $\mathcal{F}_{\mathrm{temp}}$ is independently selectable, allowing us to compare LinOSS, S5, LRU, Mamba, GRU, LSTM, and attention-based alternatives under a common encode–evolve–decode framework. We use three spatial instantiations.

Flat PON via reshape. For low-dimensional $1 + 1$D problems, each spatial snapshot is sufficiently small that no learned spatial basis is required. We flatten each snapshot $u(\cdot, t_n) \in \mathbb{R}^{N_x C}$ directly,

$$\mathcal{E}_{\mathrm{in}}(u(\cdot, t_n)) = \mathrm{vec}\big(u(\cdot, t_n)\big),$$

and use the learned projections $W_{\mathrm{in}}, W_{\mathrm{out}}$ to align this flattened representation with the latent dimension d_h. In this setting, the spatial inductive bias is carried primarily by the input/output projections and the temporal core. This instantiation is used for the $1 + 1$D Burgers, fKdV, RD-ft, RD-ftx benchmarks, and for the inductive-bias study.

Spectral PON via spatial Fourier lift. For benchmarks where the solution has a useful low-frequency spectral representation, $\mathcal{E}_{\text{in}}$ is implemented as a spectral spatial lift acting only on the spatial axes of each time slice. The encoder maps a spatial field to latent coefficients by applying low-mode Fourier multiplication with mode budget M, followed by channel mixing into dimension d_h. The decoder $\mathcal{D}_{\text{out}}$ reconstructs the physical field from the evolved latent representation using a learned spatial map. Because the spectral lift is defined through Fourier modes rather than through a fixed grid, this instantiation naturally supports resolution-transfer experiments whenever the target grid contains the learned spectral support. We use this family for the spectral-transfer resolution-invariance experiments and for Fourier-structured reaction–diffusion settings.

Linear PON via POD/PCA. For higher-dimensional systems, we use a data-adaptive linear basis. Following the POD-DeepONet construction of Lu et al. (2022), we compute a POD basis from training snapshots and keep it fixed during operator training. Let $\Phi_{\text{in}} \in \mathbb{R}^{N_x C \times r_{\text{POD}}}$ denote the encoder basis and μ_{in} the training mean. The encoder is

$$\mathcal{E}_{\text{in}}(u) = \big(u - \mu_{\text{in}}\big)\Phi_{\text{in}} \in \mathbb{R}^{r_{\text{POD}}}.$$

Similarly, with decoder basis Φ_{out} and mean μ_{out}, the decoder reconstructs

$$\mathcal{D}_{\text{out}}(q) = q\Phi_{\text{out}}^{\top} + \mu_{\text{out}}.$$

This gives the optimal rank-r_{POD} linear reconstruction in the L^2 truncation sense. The POD bases are fixed after fitting, so the learned dynamics are concentrated in $W_{\text{in}}, \mathcal{F}_{\text{temp}}, W_{\text{out}}$. This instantiation is used for the higher-dimensional Brusselator and Beltrami experiments.

Benchmark mapping. The experimental suites therefore instantiate Eq. (1) as follows. The $1+1$D benchmarks use the flat PON, the Fourier reaction–diffusion and resolution-transfer experiments use the spectral PON, and the Brusselator and Beltrami experiments use the POD/PCA PON. Across all suites, the input projection $\mathcal{L}_{\text{in}}$, temporal core $\mathcal{F}_{\text{temp}}$, output projection $\mathcal{L}_{\text{out}}$, and decoder $\mathcal{D}_{\text{out}}$ preserve the same logical data flow:

$$u(\cdot, t_n) \xrightarrow{\mathcal{E}_{\text{in}}} c_n \xrightarrow{\mathcal{L}_{\text{in}}} z_n \xrightarrow{\mathcal{F}_{\text{temp}}} \tilde{z}_n \xrightarrow{\mathcal{L}_{\text{out}}} \tilde{c}_n \xrightarrow{\mathcal{D}_{\text{out}}} \hat{u}(\cdot, t_n),$$

where c_n denotes the spatial coefficient vector produced by the chosen PON and $z_n \in \mathbb{R}^{d_h}$ is the latent state evolved by the temporal core. This unified formulation allows us to isolate the effect of the temporal operator while changing only the spatial interface required by each benchmark family.

Unless otherwise stated, baseline temporal cores are configured with comparable hidden width and parameter budget to PDON-LinOSS, so that performance differences primarily reflect the temporal inductive bias rather than a substantial change in model capacity. A detailed schematic of the $\mathcal{F}_{\text{temp}} = \text{LinOSS}$ instantiation is shown in Figure 7.

G.2 $1+1$D SUITE: BURGERS, KDV, AND REACTION–DIFFUSION

For the $1+1$D benchmarks, we use the flat-PON instantiation as explained before. Since each spatial snapshot is low-dimensional, no POD/PCA basis is fitted and no spectral spatial lift is used. Instead, each time slice is flattened, projected to the latent dimension, evolved by the temporal core, and reshaped back to the physical grid.

$$\begin{aligned}
\mathbf{A} &\in \mathbb{R}^{B\times N_x\times N_t\times C_{\text{in}}}, && \text{input trajectory on the space–time grid,}\\
\mathbf{R}^{\text{flat}} &= \text{vec}_{x,C_{\text{in}}}\big(\text{permute}_t(\mathbf{A})\big) \in \mathbb{R}^{B\times N_t\times N_x C_{\text{in}}}, && (\mathcal{E}_{\text{in}}:\ \text{fixed flattening}),\\
\mathbf{Z} &= \mathbf{R}^{\text{flat}} W_{\text{in}} + b_{\text{in}} \in \mathbb{R}^{B\times N_t\times d_h}, && (\mathcal{L}_{\text{in}}),\\
\widetilde{\mathbf{Z}} &= \mathcal{F}_{\text{temp}}(\mathbf{Z}) \in \mathbb{R}^{B\times N_t\times d_h}, && (\mathcal{F}_{\text{temp}}),\\
\widetilde{\mathbf{R}} &= \widetilde{\mathbf{Z}} W_{\text{out}} + b_{\text{out}} \in \mathbb{R}^{B\times N_t\times N_x C}, && (\mathcal{L}_{\text{out}}),\\
\hat{\mathbf{U}} &= \text{reshape}_{x,C,t}(\widetilde{\mathbf{R}}) \in \mathbb{R}^{B\times N_x\times N_t\times C}. && (\mathcal{D}_{\text{out}}:\ \text{fixed reshape}).
\end{aligned} \tag{53}$$

Here $\mathcal{E}_{\text{in}}$ and $\mathcal{D}_{\text{out}}$ are parameter-free reshapes, while $W_{\text{in}}, W_{\text{out}}$ perform the trainable basis alignment between the flattened spatial snapshot and the latent temporal state. The temporal operator $\mathcal{F}_{\text{temp}} \in$

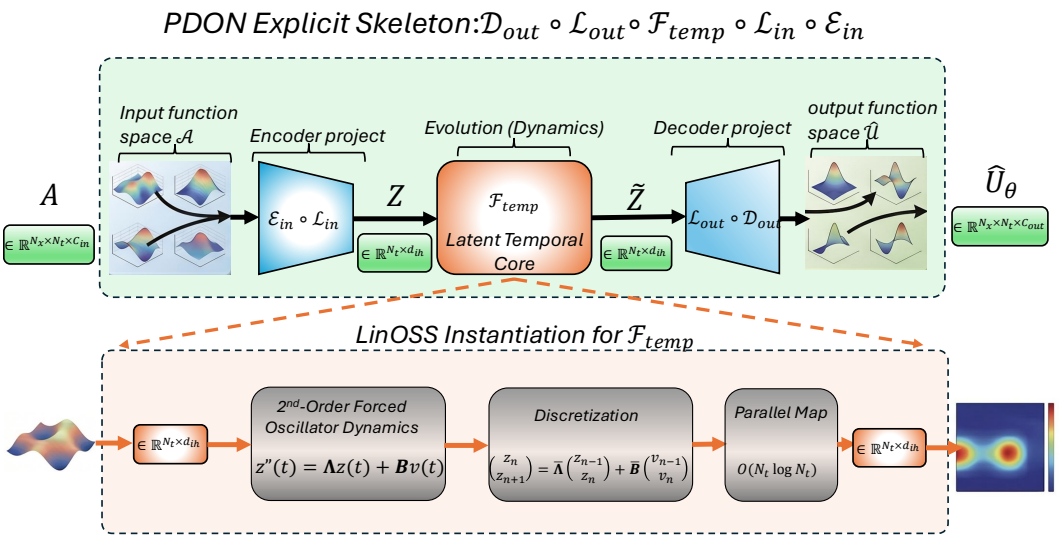


Figure 7: Overview of PDON with a LinOSS temporal core. Top: the PDON explicit factorisation maps an input trajectory $\mathbf{A}$ to a predicted trajectory $\hat{\mathbf{U}}$ by encoding spatial fields into latent coefficients, evolving them with $\mathcal{F}_{\text{temp}}$, and decoding back to the output function space. Bottom: the LinOSS instantiation of $\mathcal{F}_{\text{temp}}$ uses forced second-order oscillator dynamics, a stable discretisation, and a parallel scan to propagate the latent sequence efficiently.

$\{\text{LinOSS}, \text{S5}, \text{LRU}, \text{Mamba}, \text{GRU}, \text{LSTM}, \text{Transformer}, \text{GalerkinTransformer}, \text{GNOT}\}$ is the only component varied in the temporal-core ablations. This makes the $1+1$D suite the cheapest realisation of the PDON skeleton and a clean setting for comparing temporal inductive biases. (Architecture of data flow is in Figure 8)

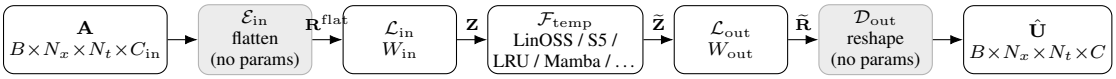


Figure 8: $1+1$D flat-PON instantiation of PDON, corresponding to Eq. (53). Grey blocks denote parameter-free flattening/reshape operations; the trainable spatial alignment is carried by $\mathcal{L}_{\text{in}}$ and $\mathcal{L}_{\text{out}}$, while $\mathcal{F}_{\text{temp}}$ provides the temporal dynamics.

Burgers: hyperparameters and network details. We use the flat-PON instantiation in Eq. (53). For PDON-LinOSS, the latent width is $d_h = 128$, with $L = 4$ LinOSS layers and residual mixing $\alpha = 0.5$. Alternative temporal cores (Mamba, S5, LRU, GRU, LSTM, Transformer, Galerkin Transformer, and GNOT) are inserted into the same $\mathcal{F}_{\text{temp}}$ slot with hidden width matched to $d_h = 128$. The PDE is the one-dimensional viscous Burgers equation

$$u_t + u\,u_x - u_{xx} = f(t), \qquad (x,t) \in (-L, L) \times (T_0, T_0 + T),$$

with periodic boundary conditions. We use $N_x = 100$, $N_t = 100$, $L = 1$, $T_0 = 0.1$ and final time $T = 1$, 3000 training trajectories, Adam with initial learning rate 10^{-3}, batch size 32, and linear learning-rate decay until validation relative-L^2 plateaus, typically within 800 epochs. (All of parameters are set after hyperparameter optimization via extensive grid search)

Forced KdV: hyperparameters and network details. We use the same flat-PON architecture and optimizer settings as in the Burgers experiment: $d_h = 128$, $L = 4$ LinOSS layers, residual mixing $\alpha = 0.5$, batch size 32, Adam with initial learning rate 10^{-3}, and linear learning-rate decay. The governing equation is the forced Korteweg–de Vries equation

$$u_t + u\,u_x + \beta\,u_{xxx} = f_x, \qquad (x,t) \in (-L, L) \times (0, T),$$

with periodic boundary conditions and derivative of random forcing f_x. We use $L = 4$, $T = 5$, $N_x = 100$, $N_t = 100$, and 1500 training trajectories. Alternative temporal cores are matched to the same hidden width $d_h = 128$ for parameter-controlled comparison. (All of parameters are set after hyperparameter optimization via extensive grid search)

RD-ft: hyperparameters and network details. For the reaction–diffusion setting with time-dependent forcing, we again use the flat-PON architecture with $d_h = 128$, $L = 4$, residual mixing $\alpha = 0.5$, and matched hidden width for all temporal cores. The system has the form

$$D\,u_{xx} + k_{\mathrm{rx}}\,u^2 - \pi^2\,u_t = f(t), \qquad (x,t) \in [0,1] \times [0,10],$$

where $f(t)$ is a random time-dependent forcing. We use $N_x = 20$, $N_t = 100$, scalar input/output fields, Adam with initial learning rate 10^{-3}, batch size 32, and linear decay until validation relative-L^2 plateaus. (All of parameters are set after hyperparameter optimization via extensive grid search)

RD-ftx: hyperparameters and network details. The RD-ftx benchmark uses the same reaction–diffusion architecture and training settings as RD-ft, but with space–time forcing:

$$D\,u_{xx} + k_{\mathrm{rx}}\,u^2 - \pi^2\,u_t = f(x,t), \qquad (x,t) \in [0,1] \times [0,10],$$

We use the flat-PON instantiation with $d_h = 128$, $L = 4$, residual mixing $\alpha = 0.5$, and all alternative temporal cores width-matched to $d_h = 128$. The spatial and temporal grids are $N_x = 20$ and $N_t = 100$; training uses Adam with initial learning rate 10^{-3}, batch size 32, and linear learning-rate decay. (All of parameters are set after hyperparameter optimization via extensive grid search)

G.3 FOURIER-RD $2 + 1$D

For the Fourier reaction–diffusion benchmark, we use a fixed spectral-PON instantiation. The spatial encoder and decoder are not learned: each two-dimensional spatial snapshot is mapped to Fourier coefficients by a fixed real-valued rFFT featurisation, evolved as a latent temporal sequence, and then reconstructed by the inverse Fourier map. This gives a parameter-free spectral interface while keeping the trainable part of the model in $\mathcal{L}_{\mathrm{in}}, \mathcal{F}_{\mathrm{temp}}, \mathcal{L}_{\mathrm{out}}$.

$$\begin{aligned}
\mathbf{A} &\in \mathbb{R}^{B\times N_x\times N_x\times N_t\times C_{\mathrm{in}}}, && \text{input forcing trajectory on the space–time grid,}\\
\mathbf{R}^{\mathrm{F}} &= \mathrm{rFFT}_{x,y}\big(\mathrm{permute}_t(\mathbf{A})\big) \in \mathbb{R}^{B\times N_t\times d_{\mathrm{F}}}, && (\mathcal{E}_{\mathrm{in}}:\ \text{fixed Fourier featurisation}),\\
\mathbf{Z} &= \mathbf{R}^{\mathrm{F}} W_{\mathrm{in}} + b_{\mathrm{in}} \in \mathbb{R}^{B\times N_t\times d_h}, && (\mathcal{L}_{\mathrm{in}}),\\
\widetilde{\mathbf{Z}} &= \mathcal{F}_{\mathrm{temp}}(\mathbf{Z}) \in \mathbb{R}^{B\times N_t\times d_h}, && (\mathcal{F}_{\mathrm{temp}}),\\
\widetilde{\mathbf{R}}^{\mathrm{F}} &= \widetilde{\mathbf{Z}} W_{\mathrm{out}} + b_{\mathrm{out}} \in \mathbb{R}^{B\times N_t\times d_{\mathrm{F}}}, && (\mathcal{L}_{\mathrm{out}}),\\
\hat{\mathbf{U}} &= \mathrm{iFFT}_{x,y}\big(\widetilde{\mathbf{R}}^{\mathrm{F}}\big) \in \mathbb{R}^{B\times N_x\times N_x\times N_t\times C}. && (\mathcal{D}_{\mathrm{out}}:\ \text{fixed inverse Fourier map}).
\end{aligned} \tag{54}$$

The Fourier feature width is determined by the one-sided two-dimensional rFFT and the real–imaginary concatenation:

$$d_{\mathrm{F}} = 2N_x\left(\frac{N_x}{2} + 1\right), \qquad \text{for scalar fields on an } N_x \times N_x \text{ grid.} \tag{55}$$

The training loss is computed in Fourier-feature space,

$$\mathcal{L}_{\mathrm{F}} = \frac{\|\widetilde{\mathbf{R}}^{\mathrm{F}} - \mathbf{R}^{\mathrm{F}}_{\mathrm{out}}\|_2}{\|\mathbf{R}^{\mathrm{F}}_{\mathrm{out}}\|_2}, \qquad \mathbf{R}^{\mathrm{F}}_{\mathrm{out}} = \mathcal{E}_{\mathrm{in}}(\mathbf{U}), \tag{56}$$

where $\mathbf{R}^{\mathrm{F}}_{\mathrm{out}}$ denotes the Fourier representation of the target solution trajectory. Grid-space predictions are obtained by the fixed inverse Fourier decoder $\mathcal{D}_{\mathrm{out}}$, and evaluation is reported using the same relative-L^2 metric as the other benchmark suites.

Thus, in Eq. (1), $\mathcal{E}_{\mathrm{in}}$ and $\mathcal{D}_{\mathrm{out}}$ are fixed analytic Fourier maps, while the trainable dynamics are concentrated in $\mathcal{L}_{\mathrm{in}}, \mathcal{F}_{\mathrm{temp}}, \mathcal{L}_{\mathrm{out}}$. The temporal core operates on a length-N_t sequence with feature

width $d_{\mathrm{F}} \to d_h \to d_{\mathrm{F}}$, making this benchmark a clean test of latent temporal evolution on spectrally structured two-dimensional fields. (Architecture of data flow is in Figure 9)

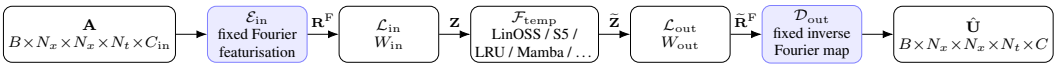


Figure 9: Fourier-RD 2+1D instantiation of PDON, corresponding to Eq. (54). Blue blocks denote fixed analytic Fourier transforms, while the trainable dynamics are carried by $\mathcal{L}_{\mathrm{in}}$, $\mathcal{F}_{\mathrm{temp}}$, and $\mathcal{L}_{\mathrm{out}}$.

Fourier-RD $2+1$D: hyperparameters and network details. For the two-dimensional Fourier reaction–diffusion benchmark, we use the fixed spectral-PON architecture in Eq. (54). The system evolves on a two-dimensional periodic torus and has the general reaction–diffusion form

$$0 = \partial_t u - \Delta u - R(u) + f(x, y, t),$$

where $R(u) = u^2$ denotes the nonlinear reaction term and $f(x, y, t)$ is a space–time forcing field. Each spatial snapshot is represented in Fourier feature space using a one-sided two-dimensional rFFT. We use $N_x = 64$, $N_t = 50$, so the Fourier feature width is

$$d_{\mathrm{F}} = 2N_x \left(\frac{N_x}{2} + 1 \right) = 2 \cdot 64 \cdot 33 = 4224.$$

For PDON-LinOSS, the latent width is $d_h = 256$, with $L = 4$ LinOSS layers and residual mixing $\alpha = 0.5$. The input and output projections have sizes

$$W_{\mathrm{in}} \in \mathbb{R}^{d_{\mathrm{F}} \times d_h}, \qquad W_{\mathrm{out}} \in \mathbb{R}^{d_h \times d_{\mathrm{F}}}.$$

Alternative temporal cores are inserted into the same $\mathcal{F}_{\mathrm{temp}}$ slot with hidden width matched to $d_h = 256$. We train with Adam using initial learning rate 10^{-3}, batch size 16, and linear learning-rate decay until validation relative-L^2 plateaus, typically within 600 epochs. All hyperparameters are selected by grid search.

G.4 Brusselator $2+1$D with POD/PCA PON

For the Brusselator benchmark, we use the POD/PCA-PON instantiation described above. This is the first suite in which the spatial encoder–decoder pair is data-adaptive: the POD bases are fitted from training snapshots and then kept fixed during operator training. Thus, unlike the flat $1+1$D setting, the encoder and decoder contain a learned-from-data spatial basis; unlike the Fourier-RD setting, the basis is not analytic but obtained by SVD.

Let $d_{\mathrm{space}} = N_x N_y C$ denote the flattened spatial dimension of one Brusselator snapshot and let r_{POD} denote the retained POD rank. The data flow is

$$\begin{aligned}
\mathbf{A} &\in \mathbb{R}^{B \times N_t \times d_{\mathrm{space}}}, && \text{input trajectory with flattened spatial snapshots,} \\
\mathbf{R}^{\mathrm{POD}} &= (\mathbf{A} - \mu_{\mathrm{in}})\Phi_{\mathrm{in}} \in \mathbb{R}^{B \times N_t \times r_{\mathrm{POD}}}, && (\mathcal{E}_{\mathrm{in}} : \text{ fixed POD encoder}), \\
\mathbf{Z} &= \mathbf{R}^{\mathrm{POD}} W_{\mathrm{in}} + b_{\mathrm{in}} \in \mathbb{R}^{B \times N_t \times d_h}, && (\mathcal{L}_{\mathrm{in}}), \\
\widetilde{\mathbf{Z}} &= \mathcal{F}_{\mathrm{temp}}(\mathbf{Z}) \in \mathbb{R}^{B \times N_t \times d_h}, && (\mathcal{F}_{\mathrm{temp}}), \\
\widetilde{\mathbf{R}}^{\mathrm{POD}} &= \widetilde{\mathbf{Z}} W_{\mathrm{out}} + b_{\mathrm{out}} \in \mathbb{R}^{B \times N_t \times r_{\mathrm{POD}}}, && (\mathcal{L}_{\mathrm{out}}), \\
\hat{\mathbf{U}} &= \widetilde{\mathbf{R}}^{\mathrm{POD}} \Phi_{\mathrm{out}}^{\top} + \mu_{\mathrm{out}} \in \mathbb{R}^{B \times N_t \times d_{\mathrm{space}}}. && (\mathcal{D}_{\mathrm{out}} : \text{ fixed POD decoder}).
\end{aligned} \tag{57}$$

Here $\Phi_{\mathrm{in}} \in \mathbb{R}^{d_{\mathrm{space}} \times r_{\mathrm{POD}}}$ and $\Phi_{\mathrm{out}} \in \mathbb{R}^{d_{\mathrm{space}} \times r_{\mathrm{POD}}}$ are POD bases computed from the training input and output snapshots, respectively, and $\mu_{\mathrm{in}}, \mu_{\mathrm{out}}$ are the corresponding training means. These quantities are fixed after fitting and are not updated during operator training. The trainable part of the model is therefore concentrated in $W_{\mathrm{in}}, \mathcal{F}_{\mathrm{temp}}, W_{\mathrm{out}}$: $\mathcal{L}_{\mathrm{in}}$ aligns POD coefficients with the latent temporal state, $\mathcal{F}_{\mathrm{temp}}$ evolves the latent sequence, and $\mathcal{L}_{\mathrm{out}}$ projects the evolved states back to the output POD coefficient space. (Architecture of data flow is in Figure 10)

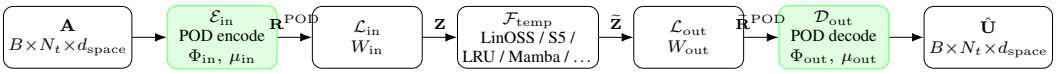


Figure 10: Brusselator 2+1D POD/PCA-PON instantiation of PDON, corresponding to Eq. (57). Green blocks denote data-adaptive POD encoder/decoder maps fitted by SVD on training snapshots and frozen during operator training.

Brusselator $2+1$D: hyperparameters and network details. For the Brusselator benchmark, we use the POD/PCA-PON architecture in Eq. (57). The system evolves on a two-dimensional periodic domain and is governed by the reaction–diffusion system

$$\partial_t u = D_u \Delta u + a - (b+1)u + u^2 v + f(t), \qquad \partial_t v = D_v \Delta v + bu - u^2 v.$$

Here u and v are the two reacting species, D_u, D_v are diffusion coefficients, and a, b control the reaction kinetics. We use $N_x = N_y = 64$, $N_t = 100$, and fit the POD encoder–decoder bases on training snapshots before freezing them during operator training. The output POD rank is $r_{\text{POD}} = 64$.

For PDON-LinOSS, the latent width is $d_h = 128$, with $L = 4$ LinOSS layers and residual mixing $\alpha = 0.5$. The input and output projections map between the POD coefficient space and the latent temporal state,

$$W_{\text{in}} \in \mathbb{R}^{r_{\text{POD}} \times d_h}, \qquad W_{\text{out}} \in \mathbb{R}^{d_h \times r_{\text{POD}}}.$$

Alternative temporal cores are inserted into the same $\mathcal{F}_{\text{temp}}$ slot with the same POD encoder, POD decoder, and projection dimensions, so only the temporal operator is changed. We train with Adam using initial learning rate 10^{-3}, batch size 32, and linear learning-rate decay until validation relative-L^2 plateaus, typically within 600 epochs. All hyperparameters are selected by grid search.

G.5 Beltrami Flow $3+1$D with POD/PCA PON

For the Beltrami flow benchmark, we use the same POD/PCA-PON instantiation as in the Brusselator experiment, but now on a three-dimensional vector-valued velocity field. Each snapshot consists of the three velocity components on an $N_x \times N_y \times N_z$ grid, so the flattened spatial dimension is

$$d_{\text{space}} = 3N_x N_y N_z.$$

The POD encoder compresses this high-dimensional velocity field to a compact coefficient vector before temporal evolution, making the latent dynamics tractable even for three-dimensional flows.

$$\begin{aligned}
\mathbf{A} &\in \mathbb{R}^{B\times N_t\times d_{\text{space}}}, && d_{\text{space}} = 3N_xN_yN_z,\\
\mathbf{R}^{\text{POD}} &= (\mathbf{A}-\mu_{\text{in}})\Phi_{\text{in}} \in \mathbb{R}^{B\times N_t\times r_{\text{POD}}}, && (\mathcal{E}_{\text{in}}:\ \text{fixed POD encoder}),\\
\mathbf{Z} &= \mathbf{R}^{\text{POD}} W_{\text{in}} + b_{\text{in}} \in \mathbb{R}^{B\times N_t\times d_h}, && (\mathcal{L}_{\text{in}}),\\
\widetilde{\mathbf{Z}} &= \mathcal{F}_{\text{temp}}(\mathbf{Z}) \in \mathbb{R}^{B\times N_t\times d_h}, && (\mathcal{F}_{\text{temp}}),\\
\widetilde{\mathbf{R}}^{\text{POD}} &= \widetilde{\mathbf{Z}} W_{\text{out}} + b_{\text{out}} \in \mathbb{R}^{B\times N_t\times r_{\text{POD}}}, && (\mathcal{L}_{\text{out}}),\\
\hat{\mathbf{U}} &= \widetilde{\mathbf{R}}^{\text{POD}} \Phi_{\text{out}}^{\top} + \mu_{\text{out}} \in \mathbb{R}^{B\times N_t\times d_{\text{space}}}. && (\mathcal{D}_{\text{out}}:\ \text{fixed POD decoder}).
\end{aligned} \tag{58}$$

Here $\Phi_{\text{in}}, \Phi_{\text{out}} \in \mathbb{R}^{d_{\text{space}}\times r_{\text{POD}}}$ are POD bases fitted from training input and output snapshots, and $\mu_{\text{in}}, \mu_{\text{out}}$ are the corresponding means. These POD quantities are fixed during operator training. The trainable part of the model is again concentrated in $W_{\text{in}}, \mathcal{F}_{\text{temp}}, W_{\text{out}}$, with the same logical mapping as the Brusselator case:

$$\text{POD coefficients} \longrightarrow \text{latent temporal state} \longrightarrow \text{evolved POD coefficients}.$$

The value of the POD path is especially pronounced in this benchmark: for a 32^3 grid, $d_{\text{space}} = 3 \cdot 32^3 \approx 9.8 \times 10^4$, while the temporal core only sees a latent sequence of width $d_h \ll d_{\text{space}}$. This substantially reduces the cost of long-horizon temporal modelling on three-dimensional vector fields. (Architecture of data flow is in Figure 11)

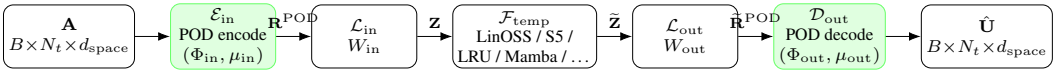


Figure 11: Beltrami flow 3+1D POD/PCA-PON instantiation of PDON, corresponding to Eq. (58). Green blocks denote POD/PCA encoder–decoder maps fitted on training snapshots and frozen during operator training. Here $d_{\text{space}} = 3N_xN_yN_z$.

Beltrami flow $3+1$D: hyperparameters and network details. For the Beltrami flow benchmark, we use the POD/PCA-PON architecture in Eq. (58). The system is a synthetic three-dimensional incompressible flow with velocity field

$$\mathbf{u}(x,y,z,t) = \big(u_1(x,y,z,t), u_2(x,y,z,t), u_3(x,y,z,t)\big),$$

sampled on a 32^3 spatial grid. Each snapshot therefore has flattened dimension

$$d_{\text{space}} = 3N_xN_yN_z = 3\cdot 32^3 \approx 9.8\times 10^4.$$

We use $N_x = N_y = N_z = 32$, $N_t = 20$, and POD rank $r_{\text{POD}} = 128$. The POD encoder and decoder are fitted on training snapshots and then frozen during operator training.

For PDON-LinOSS, the latent width is $d_h = 128$, with $L = 4$ LinOSS layers and residual mixing $\alpha = 0.5$. The input and output projections map between the POD coefficient space and the latent temporal state,

$$W_{\text{in}} \in \mathbb{R}^{r_{\text{POD}}\times d_h}, \qquad W_{\text{out}} \in \mathbb{R}^{d_h\times r_{\text{POD}}}.$$

Alternative temporal cores are inserted into the same $\mathcal{F}_{\text{temp}}$ slot with the same POD encoder, POD decoder, and projection dimensions, so only the temporal operator is changed. We train with Adam using initial learning rate 10^{-3}, batch size 8, and linear learning-rate decay until validation relative-L^2 plateaus, typically within 800 epochs. All hyperparameters are selected by grid search.

G.6 ERA5 PRESSURE-LEVEL FORECASTING WITH MULTILEVEL ATMOSPHERIC FIELDS

For the ERA5 experiment, we use multilevel pressure-level reanalysis fields as a real-world spatiotemporal forecasting benchmark. Each atmospheric state is constructed from three representative tropospheric pressure levels,

$$p \in \{850, 500, 250\}\,\text{hPa},$$

and four variables at each level: geopotential Φ, temperature T, zonal wind u, and meridional wind v. Thus each snapshot is a 12-channel global latitude–longitude field,

$$a_n(\lambda,\varphi) = \Big[\Phi_{850}, T_{850}, u_{850}, v_{850}, \Phi_{500}, T_{500}, u_{500}, v_{500}, \Phi_{250}, T_{250}, u_{250}, v_{250}\Big](\lambda,\varphi,t_n),$$

with $C_{\text{in}} = 12$. The three levels provide a compact vertical description of the atmosphere: $850\,\text{hPa}$ captures lower-tropospheric thermal and flow structure, $500\,\text{hPa}$ captures mid-tropospheric large-scale wave dynamics, and $250\,\text{hPa}$ captures upper-tropospheric jet-level winds.

Let N_λ and N_φ denote the longitude and latitude grid sizes, and let

$$d_{\text{space}} = N_\lambda N_\varphi C_{\text{in}}$$

denote the flattened dimension of one multilevel ERA5 snapshot. We use the same encode–evolve–decode structure as in the other benchmarks, with the temporal core acting on a compact latent sequence. The data flow is

$$\begin{aligned}
&\mathbf{A} \in \mathbb{R}^{B\times T_{\text{in}}\times d_{\text{space}}}, && \text{input ERA5 sequence with flattened multilevel snapshots,}\\
&\mathbf{R}^{\text{atm}} = \mathcal{E}_{\text{in}}(\mathbf{A}) \in \mathbb{R}^{B\times T_{\text{in}}\times r_{\text{atm}}}, && (\mathcal{E}_{\text{in}}:\ \text{atmospheric spatial encoder}),\\
&\mathbf{Z} = \mathbf{R}^{\text{atm}}W_{\text{in}} + b_{\text{in}} \in \mathbb{R}^{B\times T_{\text{in}}\times d_h}, && (\mathcal{L}_{\text{in}}),\\
&\widetilde{\mathbf{Z}} = \mathcal{F}_{\text{temp}}(\mathbf{Z}) \in \mathbb{R}^{B\times T_{\text{out}}\times d_h}, && (\mathcal{F}_{\text{temp}}),\\
&\widetilde{\mathbf{R}}^{\text{atm}} = \widetilde{\mathbf{Z}}W_{\text{out}} + b_{\text{out}} \in \mathbb{R}^{B\times T_{\text{out}}\times r_{\text{atm}}}, && (\mathcal{L}_{\text{out}}),\\
&\hat{\mathbf{U}}_z = \mathcal{D}_{\text{out}}(\widetilde{\mathbf{R}}^{\text{atm}}) \in \mathbb{R}^{B\times T_{\text{out}}\times N_\lambda N_\varphi}. && (\mathcal{D}_{\text{out}}:\ \text{reconstruction of geopotential } z).
\end{aligned} \tag{59}$$

Here r_{atm} denotes the dimension of the atmospheric coefficient representation produced by the spatial encoder. The model receives the 12-channel multilevel atmospheric state as input, while the reported forecasting target is the future trajectory of the geopotential field z. In other words, the learned flow map is

$$\mathcal{G}_{\Delta t}^{T_{\mathrm{out}}} : \{a_0, \ldots, a_{T_{\mathrm{in}}-1}\} \longmapsto \{z_{T_{\mathrm{in}}}, \ldots, z_{T_{\mathrm{in}}+T_{\mathrm{out}}-1}\}, \tag{60}$$

where Δt is the temporal sampling interval.

This setup is substantially more complex than the synthetic PDE suites. The horizontal wind components (u, v) provide advective and rotational transport information, T describes thermodynamic structure, and Φ encodes the large-scale pressure-level height field tied to balanced atmospheric circulation. Thus, the input state contains lower-, middle-, and upper-tropospheric information, while the prediction task evaluates whether PDON can evolve a compact latent representation of real atmospheric dynamics and reconstruct the future geopotential trajectory. Full physical equations and preprocessing details are given in Appendix F.7. (Architecture of data flow is in Figure 12)

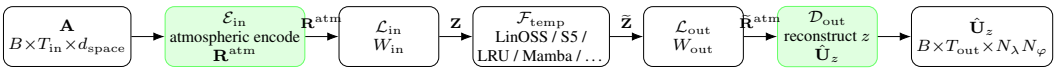


Figure 12: ERA5 forecasting architecture corresponding to Eq. (59). Green blocks denote the spatial atmospheric encoder and decoder, while $\mathcal{F}_{\mathrm{temp}}$ evolves the latent sequence and the final reconstruction targets the future geopotential field z.

ERA5: hyperparameters and network details. For the ERA5 pressure-level forecasting benchmark, we use the architecture in Eq. (59). The input state consists of 12 pressure-level channels formed by concatenating (Φ, T, u, v) at 850, 500, and 250 hPa, while the prediction target is the future geopotential trajectory z. The data are sampled at 6-hourly intervals, and the task is sequence-to-sequence prediction over a seven-day forecast horizon.

For PDON-LinOSS, the latent width is $d_h = 128$, with $L = 4$ LinOSS layers and residual mixing $\alpha = 0.5$. Alternative temporal cores are inserted into the same $\mathcal{F}_{\mathrm{temp}}$ slot and sized to remain within approximately 5% of the PDON-LinOSS parameter count when possible. Spectral baselines use $M = 32$ modes for one-dimensional spectral operators and $M = 12$ modes for two-dimensional spectral operators. Attention-based baselines such as Galerkin Transformer and GNOT use 4 attention heads with hidden width 128. All models are trained with Adam using initial learning rate 10^{-3}, batch size 32, and linear learning-rate decay until validation relative-L^2 plateaus. All hyperparameters are selected by grid search.

For data splitting, we use three months of ERA5 daily-mean pressure-level data sampled from the original 6-hourly reanalysis fields. The resulting trajectories are partitioned chronologically into 80% training, 10% validation, and 10% test samples. The validation split is used for model selection and early stopping, while all reported ERA5 errors are computed on the held-out test period. For the weather-specialised reference models, FourCastNet (Pathak et al., 2022) and ClimaX (Nguyen et al., 2023) are fine-tuned from their pre-trained checkpoints on the same three-month training split under identical optimisation settings; all reported errors for these models are therefore evaluated on the same held-out test period as PDON.

G.7 Resolution-invariance protocol

We evaluate whether a checkpoint trained at spatial resolution N^{tr} can be applied at a different target resolution N^{tgt} without retraining. In PDON, spatial resolution appears only through the encoder–decoder interface,

$$a_n \xrightarrow{\mathcal{E}_{\mathrm{in}}} z_n \xrightarrow{\mathcal{F}_{\mathrm{temp}}} \widetilde{z}_n \xrightarrow{\mathcal{D}_{\mathrm{out}}} \hat{u}_n,$$

while the temporal core $\mathcal{F}_{\mathrm{temp}}$ evolves the latent sequence $\{z_n\}_{n=1}^{N_t}$ independently of the physical grid. Resolution transfer therefore amounts to changing how fields are mapped into and out of the latent coefficient space.

Spectral spatial interface. For spectral spatial blocks, the learned representation is defined on a fixed set of low-frequency Fourier modes rather than on grid points. Let M denote the retained one-sided mode budget and

$$K(N) = \left\lfloor \frac{N}{2} \right\rfloor + 1$$

be the number of one-sided rFFT bins at resolution N. A checkpoint trained at N^{tr} is admissible at N^{tgt} whenever

$$K(N^{\mathrm{tgt}}) \geq M.$$

In this case, the same spectral weights are applied to the overlapping low-frequency modes on the target grid, and no target-resolution fine-tuning is used.

Fixed-width spatial interface. For spatial blocks tied to the training grid, such as fixed POD bases, we use a deterministic band-limited Fourier resampling map. For a one-dimensional signal $u \in \mathbb{R}^N$, define

$$P_{N' \leftarrow N} u = \mathcal{F}_{N'}^{-1} \left[\mathrm{pad/trunc}_{K(N')} \big(\mathcal{F}_N u \big) \cdot \frac{N'}{N} \right], \qquad K(N) = \left\lfloor \frac{N}{2} \right\rfloor + 1. \tag{61}$$

Here $\mathcal{F}_N$ is the discrete Fourier transform and $\mathrm{pad/trunc}$ retains the common low-frequency support by either zero-padding or truncating Fourier coefficients. The factor N'/N preserves the amplitude under the discrete transform convention.

Given a resolution-locked checkpoint $G_{N^{\mathrm{tr}}}$ and a target-grid input a^{tgt}, the transferred prediction is

$$\hat{u}^{\mathrm{tgt}} = P_{N^{\mathrm{tgt}} \leftarrow N^{\mathrm{tr}}} \left[G_{N^{\mathrm{tr}}} \left(P_{N^{\mathrm{tr}} \leftarrow N^{\mathrm{tgt}}} a^{\mathrm{tgt}} \right) \right]. \tag{62}$$

The maps $P_{N^{\mathrm{tr}} \leftarrow N^{\mathrm{tgt}}}$ and $P_{N^{\mathrm{tgt}} \leftarrow N^{\mathrm{tr}}}$ are fixed linear operators: they introduce no learned parameters and no bridge network. Thus, the experiment tests whether the learned low-frequency operator and the latent temporal dynamics transfer across spatial grids.

G.8 Inductive-bias experiment

The inductive-bias experiment isolates the effect of explicit latent temporal evolution from spatial encoder–decoder capacity. We use the one-dimensional linear advection equation

$$\partial_t u + c\, \partial_x u = 0, \qquad c = 1,$$

on a periodic domain. Its exact solution is a translation,

$$u(x,t) = f(x - ct),$$

so a model that learns the underlying dynamics should generalise from one family of initial shapes to shifted or composed shapes. Training initial conditions are sampled from a single-Gaussian distribution, while evaluation is performed on both the same distribution and three shape-shifted OOD families: double Gaussians, triple Gaussians, and random Fourier modes.

Both models share the same spatial encoder $\mathcal{E}_{\mathrm{in}}$ and decoder $\mathcal{D}_{\mathrm{out}}$. The explicit model first encodes the initial field into a latent state and then evolves that state sequentially:

$$\begin{aligned}
\mathbf{A}_0 &\in \mathbb{R}^{B \times N_x \times C_{\mathrm{in}}}, && \text{initial condition,} \\
\mathbf{z}_0 &= \mathcal{E}_{\mathrm{in}}(\mathbf{A}_0) \in \mathbb{R}^{B \times d_h}, && \text{shared spatial encoder,} \\
\mathbf{z}_{n+1} &= \mathbf{z}_n + \mathcal{A}_\theta(\mathbf{z}_n), \qquad n = 0, \ldots, N_t - 1, && \text{explicit latent update,} \\
\hat{\mathbf{U}}_n &= \mathcal{D}_{\mathrm{out}}(\mathbf{z}_n) \in \mathbb{R}^{B \times N_x \times C}. && \text{shared spatial decoder.}
\end{aligned} \tag{63}$$

Here $\mathcal{A}_\theta$ is a learned latent update map corresponding to $\mathcal{F}_{\text{temp}}$. The important point is that time evolution is represented as an autonomous recurrence in latent space before decoding.

The implicit baseline shares the spatial encoder $\mathcal{E}_{\mathrm{in}}$ and decoder $\mathcal{D}_{\mathrm{out}}$ bit-for-bit, but does not form an autonomous latent temporal recurrence. Instead, for each query time t_n, it concatenates the encoded initial state $\mathbf{c}_0 = \mathcal{E}_{\mathrm{in}}(\mathbf{A}_0)$ with a learned time embedding and passes the result through a static MLP to produce the latent summary at each step, with no state carried forward:

$$
\begin{aligned}
\mathbf{A}_0 &\in \mathbb{R}^{B\times N_x\times C_{\text{in}}}, && \text{initial condition},\\
\mathbf{c}_0 &= \mathcal{E}_{\text{in}}(\mathbf{A}_0) \in \mathbb{R}^{B\times d_h}, && \text{shared spatial encoder (identical to explicit)},\\
\mathbf{e}_n &= \tau(t_n) \in \mathbb{R}^{B\times d_\tau}, && \text{learned time embedding for step } t_n,\\
\mathbf{r}_n &= \mathcal{M}_\theta\big([\mathbf{c}_0,\, \mathbf{e}_n]\big) \in \mathbb{R}^{B\times d_h}, && \text{static time-conditioned map (no recurrence)},\\
\hat{\mathbf{U}}_n &= \mathcal{D}_{\text{out}}(\mathbf{r}_n) \in \mathbb{R}^{B\times N_x\times C}, && \text{shared spatial decoder (identical to explicit)}.
\end{aligned}
\tag{64}
$$

The implicit baseline uses the same spatial encoder $\mathcal{E}_{\text{in}}$ and decoder $\mathcal{D}_{\text{out}}$ as the explicit model. It replaces latent recurrence with a static time-conditioned map $\mathcal{M}_\theta$: the encoded initial state $\mathbf{c}_0 = \mathcal{E}_{\text{in}}(\mathbf{A}_0)$ and a learned time embedding $\tau(t_n)$ are concatenated and passed through an MLP to produce the latent summary $\mathbf{r}_n = \mathcal{M}_\theta\big([\mathbf{c}_0, \tau(t_n)]\big)$ independently at each step, without recurrence. Parameter counts of $\mathcal{A}_\theta$ and $\mathcal{M}_\theta$ are matched within 2%.

Inductive-bias hyperparameters and network details. We use $N_x = 64$, $N_t = 10$, and temporal spacing $\Delta t = 0.1$. The shared encoder E is a residual convolutional field encoder producing a 64-dimensional latent representation, and the shared decoder D is a transposed-convolutional field decoder. The explicit update A_θ is a three-layer MLP applied recurrently in latent space. The implicit model $\mathcal{M}_\theta$ is a three-layer MLP that receives the concatenation of $\mathbf{c}_0 = \mathcal{E}_{\text{in}}(u_0) \in \mathbb{R}^{64}$ and a learned sinusoidal time embedding $\tau(t_n) \in \mathbb{R}^{64}$, producing the latent summary $\mathbf{r}_n \in \mathbb{R}^{d_h}$ independently at each step.

Both $\mathcal{A}_\theta$ and $\mathcal{M}_\theta$ are three-layer MLPs of identical width; parameter counts are matched within 2% by construction. Both models are trained for 800 epochs with Adam, initial learning rate 10^{-3}, and linear learning-rate decay. Batch sizes are 128 for the implicit model and 64 for the explicit model. The explicit model is trained with autoregressive rollout over a four-step horizon. Results are averaged over three random seeds and reported using relative-L^2 error on the held-out trajectory distributions.

Numerical example related to example 1. Figure 13 quantifies the same effect: both models are accurate on the in-distribution single-Gaussian test set, but under OOD shape shifts the implicit baseline incurs much larger relative-L^2 error, while G_θ^{PDON} preserves the advective transport structure.

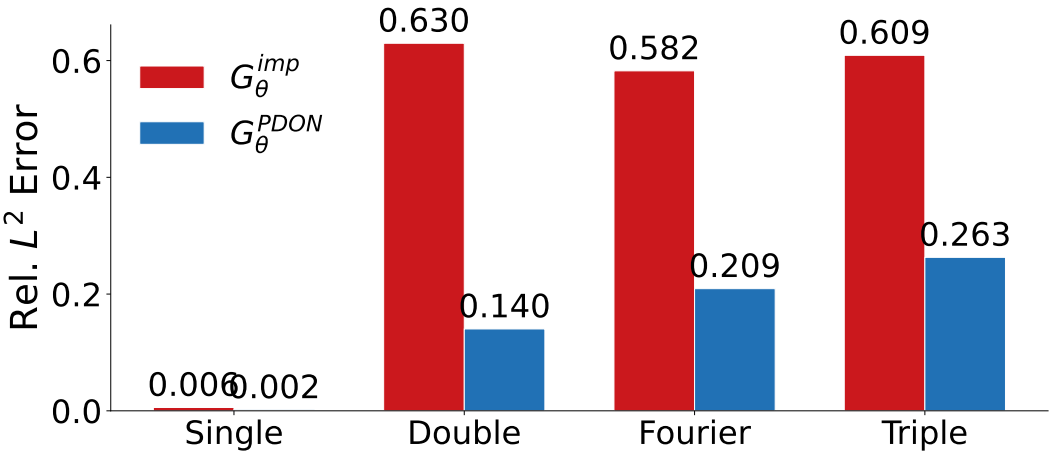


Figure 13: Inductive-bias study on advection in example 1: Numerical results.

G.9 Architectural progression across suites

Table 5 summarises the spatial encoder–decoder instantiations used across the experimental suites. The progression moves from parameter-free reshaping, to fixed analytic Fourier maps, to data-adaptive POD/PCA compression, and finally to the multilevel atmospheric encoder used for ERA5. This

mirrors the increasing dimensionality and complexity of the physical state, and highlights the role of $(\mathcal{E}_{\text{in}}, \mathcal{D}_{\text{out}})$ in reducing the spatial field before temporal modelling by $\mathcal{F}_{\text{temp}}$.

Table 5: Per-suite instantiation of the spatial encoder–decoder pair $(\mathcal{E}_{\text{in}}, \mathcal{D}_{\text{out}})$ in Eq. (1).

Suite	$\mathcal{E}_{\text{in}}$ realisation	$\mathcal{D}_{\text{out}}$ realisation
1+1D Burgers, fKdV, RD-ft, RD-ftx	fixed flattening / reshape	fixed reshape
Fourier-RD 2+1D	fixed analytic Fourier featurisation (rFFT)	fixed inverse Fourier map (iFFT)
Brusselator 2+1D	POD/PCA basis fitted on training snapshots	POD/PCA reconstruction basis fitted on training snapshots
Beltrami flow 3+1D	POD/PCA basis fitted on 3-D velocity snapshots	POD/PCA reconstruction of 3-D velocity fields
ERA5 pressure-level forecasting	multilevel atmospheric spatial encoder	geopotential-field decoder for z

The $1 + 1$D suite isolates the temporal core by removing any learned spatial basis; the Fourier-RD benchmark tests whether a fixed spectral representation is sufficient for two-dimensional reaction–diffusion dynamics; and the Brusselator and Beltrami suites use POD/PCA compression, where spatial dimensionality reduction is essential for tractable temporal modelling. ERA5 extends the same encode–evolve–decode principle to real-world multilevel atmospheric fields, where the input contains pressure-level geopotential, temperature, and wind variables, while the reported forecast target is the future geopotential field z. Across all suites, only the spatial interface changes; the temporal operator $\mathcal{F}_{\text{temp}}$ remains the component through which we compare LinOSS against alternative sequence models.

Hardware. All experiments were run on a multi-GPU NVIDIA server with four GPUs. The main benchmark and ablation runs used NVIDIA A40 GPUs; experiments and profiling runs were also executed on NVIDIA A40 GPUs with 46GB memory. Runtime and memory measurements are reported from the same hardware configuration used for the corresponding training and inference run.

H Algorithms and python intuitive code

H.1 Algorithms

Algorithm 1 PDON Forward Pass

Input: Gridded input sequence $\mathbf{a} = \{a(x_k, t_n)\}_{k,n}$; spatial encoder $\mathcal{E}_{\text{in}}$; temporal core $\mathcal{F}_{\text{temp}} = \{\mathcal{F}_\ell\}_{\ell=1}^{L}$; spatial decoder $\mathcal{D}_{\text{out}}$; projections $W_{\text{in}}, W_{\text{out}}$
Output: Predicted trajectory $\hat{\mathbf{u}}_\theta = \{\hat{u}_\theta(x_k, t_n)\}_{k,n}$
// Encode each spatial snapshot
$c_n \leftarrow \mathcal{E}_{\text{in}}(\mathbf{a}_{:,n}), \quad n = 1, \ldots, N_t$
// Project to temporal latent dimension
$z \leftarrow c\, W_{\text{in}} + b_{\text{in}}$
// Evolve the latent sequence
for $\ell = 1$ **to** L **do**
    $z \leftarrow \mathcal{F}_\ell(z) + \alpha z$ *(residual temporal block)*
end for
// Project to output coefficient dimension
$q \leftarrow z\, W_{\text{out}} + b_{\text{out}}$
// Decode each output snapshot
$\hat{\mathbf{u}}_{\theta,:,n} \leftarrow \mathcal{D}_{\text{out}}(q_n), \quad n = 1, \ldots, N_t$
return $\hat{\mathbf{u}}_\theta$

Algorithm 2 PDON Training

Input: Dataset $\mathcal{D} = \{(\mathbf{a}^{(i)}, \mathbf{u}^{(i)})\}_{i=1}^{N}$, learning rate η, number of epochs E
Initialize spatial maps $\mathcal{E}_{\text{in}}, \mathcal{D}_{\text{out}}$, projections $W_{\text{in}}, W_{\text{out}}$, and temporal core $\mathcal{F}_{\text{temp}} = \{\mathcal{F}_\ell\}_{\ell=1}^{L}$
Let θ denote all trainable parameters of $\{\mathcal{E}_{\text{in}}, \mathcal{D}_{\text{out}}, W_{\text{in}}, W_{\text{out}}, \mathcal{F}_{\text{temp}}\}$
for epoch $= 1$ **to** E **do**
  for each mini-batch $\mathcal{B} \subset \{1, \dots, N\}$ **do**
    Compute $\hat{\mathbf{u}}_\theta^{(i)}$ via Algorithm 1 for each $i \in \mathcal{B}$
    $\mathcal{L} \leftarrow \dfrac{1}{|\mathcal{B}|} \displaystyle\sum_{i \in \mathcal{B}} \dfrac{\|\hat{\mathbf{u}}_\theta^{(i)} - \mathbf{u}^{(i)}\|_2}{\|\mathbf{u}^{(i)}\|_2}$
    $\theta \leftarrow \theta - \eta \nabla_\theta \mathcal{L}$
  end for
end for
return θ

H.2 PYTHON INTUITIVE CODE

Below we give a concise PyTorch sketch of PDON with an LinOSS temporal core to illustrate the simplicity of the pipeline. The full implementation is available in the accompanying repository.

```
import torch
import torch.nn as nn

class TemporalLayer(nn.Module):
    # One residual temporal block, here instantiated by an LinOSS-style
    core.
    def __init__(self, d_z):
        super().__init__()
        self.W     = nn.Parameter(torch.randn(d_z, d_z) * 0.01)
        self.B     = nn.Parameter(torch.randn(d_z, d_z) * 0.01)
        self.C     = nn.Parameter(torch.eye(d_z))
        self.alpha = nn.Parameter(torch.tensor(0.1))

    def forward(self, z):           # z: (B, N_t, d_z)
        A = self.W - self.W.T       # skew-symmetric => oscillatory
    dynamics
        out = torch.tanh(z @ A.T + z @ self.B.T) @ self.C.T
        return out + self.alpha * z


class PDON(nn.Module):
    # POD-based instantiation of the PDON operator.
    def __init__(self, Phi_in, Phi_out, mu_in, mu_out, d_z, n_layers):
        super().__init__()
        self.register_buffer('Phi_in',  Phi_in)   # (d_in,  r_in)  fixed
        self.register_buffer('Phi_out', Phi_out)  # (d_out, r_out) fixed
        self.register_buffer('mu_in',   mu_in)
        self.register_buffer('mu_out',  mu_out)

        r_in, r_out = Phi_in.shape[1], Phi_out.shape[1]
        self.W_in   = nn.Linear(r_in, d_z)
        self.W_out  = nn.Linear(d_z, r_out)
        self.layers = nn.ModuleList(
            [TemporalLayer(d_z) for _ in range(n_layers)]
        )

    def forward(self, a):                           # a: (B, N_t, d_in)
        c     = (a - self.mu_in) @ self.Phi_in      # spatial/POD encode
        z     = self.W_in(c)                        # project to latent
    sequence
```

```
        for layer in self.layers:
            z = layer(z)                                  # temporal core
        q     = self.W_out(z)                             # output coefficients
        u_hat = q @ self.Phi_out.T + self.mu_out          # spatial/POD decode
        return u_hat                                      # (B, N_t, d_out)


# --- training sketch ---
model     = PDON(Phi_in, Phi_out, mu_in, mu_out, d_z=128, n_layers=4)
optimizer = torch.optim.Adam(model.parameters(), lr=1e-3)

for a_batch, u_batch in dataloader:
    u_hat = model(a_batch)
    loss  = (u_hat - u_batch).norm() / u_batch.norm()  # relative L2
    optimizer.zero_grad(); loss.backward(); optimizer.step()
```

I Additional Results

I.1 Oscillatory-memory diagnostic

Motivation. The main text reports an oscillatory-memory diagnostic for the temporal cores in PDON. The goal is to test whether the learned latent temporal dynamics retain an early perturbation over long horizons without artificial damping. This is especially relevant for PDE trajectories with transport, wave-like propagation, or weakly dissipative modal exchange, where long-time prediction depends on preserving energy-bearing temporal modes.

A naive diagnostic would track the Euclidean norm of the LinOSS phase-space state. However, this is misleading for LinOSS-IMEX. The IMEX update is symplectic, but not Euclidean orthogonal: even when all eigenvalues lie on the unit circle, the plain Euclidean length of the phase-space vector can exhibit transient growth due to non-orthogonal eigenvectors. Therefore, the correct quantity to measure is not the Euclidean norm, but the quadratic Hamiltonian energy preserved by the IMEX discretisation.

LinOSS-IMEX transition. For each LinOSS channel $i = 1, \dots, d_h$, let

$$s_{n,i} = \begin{bmatrix} z_{n,i} \\ \dot{z}_{n,i} \end{bmatrix} \in \mathbb{R}^2$$

denote the phase-space state, where $z_{n,i}$ is the latent oscillator coordinate and $\dot{z}_{n,i}$ is its velocity-like companion variable. In the autonomous zero-input setting, the IMEX update is

$$s_{n+1,i} = \Phi_i^{\mathrm{IMEX}} s_{n,i},$$

with

$$\Phi_i^{\mathrm{IMEX}} = \begin{bmatrix} 1 & -\Delta_i \Lambda_i \\ \Delta_i & 1 - \Delta_i^2 \Lambda_i \end{bmatrix},$$

where $\Lambda_i \geq 0$ is the learned oscillator stiffness and $\Delta_i > 0$ is the learned step size.

The characteristic polynomial of Φ_i^{IMEX} is

$$\lambda^2 - (2 - \Delta_i^2 \Lambda_i)\lambda + 1 = 0.$$

Thus, when

$$0 < \Delta_i^2 \Lambda_i < 4,$$

the eigenvalues form a complex-conjugate pair on the unit circle. This is the stable oscillatory regime of the IMEX discretisation. If $\Delta_i^2 \Lambda_i > 4$, the eigenvalues become real reciprocal pairs, and one eigenvalue has modulus larger than one. The boundary $\Delta_i^2 \Lambda_i = 4$ corresponds to the degenerate case $\lambda = -1$.

Preserved discrete Hamiltonian energy. Although the stable IMEX eigenvalues lie on the unit circle, this does not imply that $\|s_{n,i}\|_2$ is preserved. Instead, IMEX preserves a modified quadratic Hamiltonian energy. We seek a symmetric positive-definite matrix

$$Q_i = \begin{bmatrix} q_{11} & q_{12} \\ q_{12} & q_{22} \end{bmatrix}$$

such that

$$(\Phi_i^{\mathrm{IMEX}})^\top Q_i \Phi_i^{\mathrm{IMEX}} = Q_i.$$

For the above IMEX update, one valid choice is

$$Q_i = \begin{bmatrix} 1 & -\frac{1}{2}\Delta_i\Lambda_i \\ -\frac{1}{2}\Delta_i\Lambda_i & \Lambda_i \end{bmatrix}.$$

The corresponding channel energy is

$$E_{n,i} = s_{n,i}^\top Q_i s_{n,i}.$$

Writing $s_{n,i} = (z_{n,i}, \dot{z}_{n,i})^\top$, this gives

$$E_{n,i} = z_{n,i}^2 - \Delta_i\Lambda_i z_{n,i}\dot{z}_{n,i} + \Lambda_i \dot{z}_{n,i}^2.$$

This is a modified discrete energy: the cross term is an $\mathcal{O}(\Delta_i)$ correction to the continuous oscillator energy. The matrix Q_i is positive definite exactly when

$$\det(Q_i) = \Lambda_i - \frac{1}{4}\Delta_i^2\Lambda_i^2 > 0,$$

which is equivalent to

$$0 < \Delta_i^2\Lambda_i < 4.$$

Thus the same condition that places the IMEX eigenvalues on the unit circle also makes the preserved quadratic energy positive definite.

Energy conservation follows directly:

$$\begin{aligned} E_{n+1,i} &= s_{n+1,i}^\top Q_i s_{n+1,i} \\ &= s_{n,i}^\top (\Phi_i^{\mathrm{IMEX}})^\top Q_i \Phi_i^{\mathrm{IMEX}} s_{n,i} \\ &= s_{n,i}^\top Q_i s_{n,i} = E_{n,i}. \end{aligned}$$

This is the precise sense in which LinOSS-IMEX preserves oscillator memory: it keeps the latent state on a bounded Hamiltonian-energy orbit, rather than preserving the plain Euclidean norm.

Diagnostic protocol. For each temporal core, we run an autonomous impulse-response experiment. A random initial impulse δ is injected into the latent state at $n = 0$, and the input is set to zero for all later rollout steps. For LinOSS, the rollout is performed using the block-diagonal phase-space transition

$$s_{n+1} = \Phi s_n, \qquad \Phi = \mathrm{blockdiag}(\Phi_1, \ldots, \Phi_{d_h}).$$

The energy-memory curve for LinOSS is

$$M_E(n) = \left(\frac{\mathbb{E}_\delta \sum_i s_{n,i}^\top Q_i s_{n,i}}{\mathbb{E}_\delta \sum_i s_{0,i}^\top Q_i s_{0,i}} \right)^{1/2},$$

where the expectation is taken over energy-normalised random initial impulses. For stable IMEX channels, this quantity should remain close to one.

For first-order SSM cores such as S5 and LRU, there is no analogous oscillator-energy form. We therefore use the corresponding autonomous state-norm response as a comparison:

$$M_E(n) = \frac{\mathbb{E}_\delta \|z_n\|_2}{\mathbb{E}_\delta \|z_0\|_2}.$$

This measures how quickly an initial perturbation is attenuated by the learned first-order transition. Since S5 and LRU are stable first-order SSMs, their eigenvalues typically lie inside the unit disk, and the impulse response decays according to the learned eigenvalue radii.

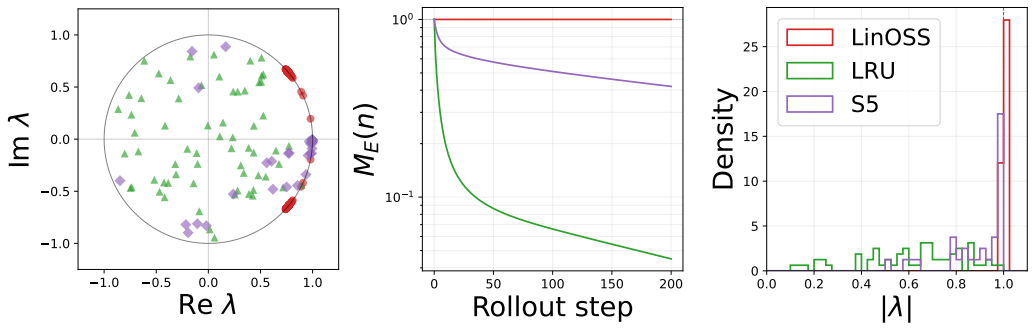


Figure 14: Oscillatory-memory diagnostic on the Brusselator benchmark. As in the KdV diagnostic, LinOSS concentrates its learned modes on the unit circle and preserves the Hamiltonian-energy response $M_E(n)$ over autonomous rollout, while S5 and LRU exhibit contractive memory decay according to their learned eigenvalue radii.

Stability statistics. For LinOSS, we also report the distribution of

$$\Delta_i^2 \Lambda_i.$$

Channels satisfying $0 < \Delta_i^2 \Lambda_i < 4$ are in the stable oscillatory IMEX regime. Channels with $\Delta_i^2 \Lambda_i \geq 4$ are flagged separately, because their IMEX transition no longer defines a positive-definite preserved energy and may be unstable under autonomous IMEX rollout. In our KdV diagnostic, a small number of LinOSS channels lie outside this range; these are indicated in the eigenvalue plot. The Hamiltonian-energy response is interpreted on the stable IMEX channels, which represent the intended operating regime of the oscillatory core.

Results. Figures 3 and 14 show the diagnostic on KdV and Brusselator, respectively. In both cases, the LinOSS eigenvalues concentrate on or near the unit circle, while the S5 and LRU eigenvalues lie predominantly inside the unit disk. The Hamiltonian-energy response of LinOSS remains essentially flat, $M_E(n) \approx 1$, showing that the autonomous latent oscillator preserves the energy of the initial perturbation. By contrast, the S5 and LRU responses decay over time, with rates consistent with their learned eigenvalue radii below one.

These results support the mechanism proposed in the main text: LinOSS improves long-horizon prediction not by uncontrolled latent amplification, but by preserving bounded oscillatory energy in the temporal latent state. This behaviour is well-aligned with long-horizon PDE systems such as KdV and Brusselator, where persistent modal exchange and weak dissipation make artificial temporal damping harmful.

I.2 Complexity Results

Table 6 reports mean and standard deviation over independent random seeds, with each model trained and evaluated from scratch per seed. Seeds differ in weight initialization and data-ordering only; all other hyperparameters are held fixed at the values reported in Appendix G.

Computational cost. Tables 9–12 report the empirical computational footprint of the main model families on each benchmark, including parameter count, 100-epoch runtime, and peak GPU memory. Across the one-dimensional benchmarks (Tables 9, 10, and 7), LinOSS has a computational profile comparable to the other compact SSM cores, especially Mamba, while avoiding the substantially larger runtime of recurrent GRU/LSTM variants and the larger parameter counts of attention-based cores. On the higher-dimensional Fourier reaction–diffusion, Brusselator, and Beltrami benchmarks (Tables 11, 12, and 8), the same pattern persists: LinOSS remains in the compact latent-evolution regime, whereas implicit FNO-style space–time models and several large SOTA baselines incur substantially higher memory, runtime, or parameter cost. These results complement Table 3: the accuracy gains of LinOSS are not obtained by increasing model size, but by using a more suitable oscillatory temporal inductive bias within a comparable computational budget.

Table 6: Relative-L^2 across seven benchmark suites reported as mean $\pm$ std over random seeds. **Bold**: best mean per column; underlined: second best mean; "div." = diverged; "—" = OOM.

Model	Burg.	RD-ft	RD-ftx	fKdV	F-RD	Brus.	Belt.
PDON (SSM cores)							
Mamba	$8.47\pm0.76\cdot10^{-3}$	$4.54\pm0.41\cdot10^{-3}$	$1.47\pm0.13\cdot10^{-2}$	$2.88\pm0.26\cdot10^{-2}$	$4.41\pm0.40\cdot10^{-2}$	$1.21\pm0.11\cdot10^{-2}$	$7.40\pm0.67\cdot10^{-2}$
S5	$1.78\pm0.14\cdot10^{-3}$	$2.78\pm0.22\cdot10^{-3}$	$1.48\pm0.12\cdot10^{-2}$	$2.02\pm0.16\cdot10^{-2}$	$2.35\pm0.19\cdot10^{-2}$	$4.24\pm0.34\cdot10^{-3}$	$6.72\pm0.54\cdot10^{-2}$
LRU	$3.40\pm0.44\cdot10^{-3}$	$9.15\pm1.19\cdot10^{-3}$	$2.67\pm0.35\cdot10^{-2}$	$2.54\pm0.33\cdot10^{-2}$	$2.59\pm0.34\cdot10^{-2}$	$5.72\pm0.74\cdot10^{-2}$	$7.65\pm0.99\cdot10^{-2}$
LinOSS(Ours)	$\underline{1.28\pm0.08\cdot10^{-3}}$	$\mathbf{1.44\pm0.09\cdot10^{-3}}$	$\mathbf{1.07\pm0.06\cdot10^{-2}}$	$\underline{1.14\pm0.07\cdot10^{-2}}$	$\mathbf{2.25\pm0.14\cdot10^{-3}}$	$\mathbf{1.78\pm0.11\cdot10^{-3}}$	$\mathbf{1.20\pm0.07\cdot10^{-3}}$
PDON (attention)							
GNOT	$1.48\pm0.21\cdot10^{-3}$	$3.47\pm0.49\cdot10^{-1}$	$5.74\pm0.80\cdot10^{-1}$	$1.20\pm0.17\cdot10^{-2}$	$3.50\pm0.49\cdot10^{-3}$	$2.83\pm0.40\cdot10^{-2}$	div.
Of-G	$4.33\pm0.78\cdot10^{-3}$	$1.25\pm0.23\cdot10^{-1}$	$4.16\pm0.75\cdot10^{-1}$	$2.76\pm0.50\cdot10^{-2}$	$\underline{2.54\pm0.46\cdot10^{-3}}$	$6.00\pm1.08\cdot10^{-2}$	div.
Of-V	$2.79\pm0.39\cdot10^{-3}$	$2.08\pm0.29\cdot10^{-1}$	$5.24\pm0.73\cdot10^{-1}$	$2.54\pm0.36\cdot10^{-2}$	$5.48\pm0.77\cdot10^{-3}$	—	div.
Of-F	$4.91\pm0.59\cdot10^{-3}$	$1.21\pm0.15\cdot10^{-1}$	$4.44\pm0.53\cdot10^{-1}$	$2.68\pm0.32\cdot10^{-2}$	$2.76\pm0.33\cdot10^{-3}$	$5.47\pm0.66\cdot10^{-2}$	div.
PDON (recurrent)							
GRU	$1.97\pm0.22\cdot10^{-3}$	$1.16\pm0.13\cdot10^{-2}$	$1.71\pm0.19\cdot10^{-2}$	$1.38\pm0.15\cdot10^{-2}$	$3.40\pm0.37\cdot10^{-2}$	$1.70\pm0.19\cdot10^{-2}$	$1.61\pm0.18\cdot10^{-1}$
LSTM	$2.19\pm0.20\cdot10^{-3}$	$1.08\pm0.10\cdot10^{-2}$	$3.24\pm0.29\cdot10^{-2}$	$\mathbf{1.05\pm0.09\cdot10^{-2}}$	$3.42\pm0.31\cdot10^{-2}$	$1.50\pm0.14\cdot10^{-2}$	$1.25\pm0.11\cdot10^{-1}$
FNO family							
FNO (implicit)	$2.76\pm0.21\cdot10^{-3}$	$2.36\pm0.18\cdot10^{-2}$	$4.22\pm0.32\cdot10^{-2}$	$2.45\pm0.18\cdot10^{-2}$	$7.68\pm0.58\cdot10^{-3}$	—	$1.50\pm0.11\cdot10^{-2}$
FFNO (implicit)	$1.32\pm0.07\cdot10^{-3}$	$3.31\pm0.18\cdot10^{-2}$	$2.61\pm0.14\cdot10^{-2}$	$1.31\pm0.07\cdot10^{-2}$	$8.24\pm0.45\cdot10^{-3}$	—	$1.38\pm0.08\cdot10^{-2}$
FFNO$(d{+}1)$	$\mathbf{1.13\pm0.06\cdot10^{-3}}$	$3.23\pm0.18\cdot10^{-2}$	$2.47\pm0.14\cdot10^{-2}$	$1.27\pm0.07\cdot10^{-2}$	$7.75\pm0.43\cdot10^{-3}$	—	$1.26\pm0.07\cdot10^{-2}$
PINNs							
PINN	$1.60\pm0.08\cdot10^{-2}$	$5.39\pm0.27\cdot10^{-2}$	$9.22\pm0.46\cdot10^{-2}$	$1.08\pm0.05\cdot10^{-1}$	$9.98\pm0.50\cdot10^{-3}$	$1.67\pm0.08\cdot10^{-2}$	$2.27\pm0.11\cdot10^{-2}$
Causal PINN	$2.63\pm0.17\cdot10^{-3}$	$3.51\pm0.23\cdot10^{-2}$	$4.19\pm0.27\cdot10^{-2}$	$2.97\pm0.19\cdot10^{-2}$	$6.12\pm0.40\cdot10^{-3}$	$3.54\pm0.23\cdot10^{-2}$	$3.63\pm0.24\cdot10^{-2}$
PINN-RNN	$4.44\pm0.33\cdot10^{-3}$	$7.52\pm0.56\cdot10^{-3}$	$3.12\pm0.23\cdot10^{-3}$	$2.86\pm0.21\cdot10^{-2}$	$6.15\pm0.46\cdot10^{-3}$	$8.57\pm0.64\cdot10^{-3}$	$1.19\pm0.09\cdot10^{-2}$
Common NOs							
DeepONet	$7.33\pm0.70\cdot10^{-3}$	$8.45\pm0.80\cdot10^{-2}$	$4.13\pm0.39\cdot10^{-2}$	$5.19\pm0.49\cdot10^{-2}$	$4.96\pm0.47\cdot10^{-2}$	$2.13\pm0.20\cdot10^{-2}$	$8.43\pm0.80\cdot10^{-2}$
LNO	$4.63\pm0.46\cdot10^{-3}$	$3.51\pm0.35\cdot10^{-2}$	$4.19\pm0.42\cdot10^{-2}$	$4.97\pm0.50\cdot10^{-2}$	$6.12\pm0.61\cdot10^{-2}$	$3.54\pm0.35\cdot10^{-2}$	$3.63\pm0.36\cdot10^{-2}$
SOTA baselines							
MaskPDE (TMLR'24)	$1.52\pm0.17\cdot10^{-3}$	$1.73\pm0.19\cdot10^{-3}$	$2.33\pm0.26\cdot10^{-2}$	$2.29\pm0.25\cdot10^{-2}$	$3.20\pm0.35\cdot10^{-2}$	$\underline{4.11\pm0.45\cdot10^{-3}}$	$8.14\pm0.90\cdot10^{-3}$
AROMA (NeurIPS'24)	$3.65\pm0.47\cdot10^{-2}$	$2.17\pm0.28\cdot10^{-2}$	$2.26\pm0.29\cdot10^{-2}$	$5.21\pm0.68\cdot10^{-2}$	$6.23\pm0.81\cdot10^{-2}$	$8.13\pm1.06\cdot10^{-3}$	$1.05\pm0.14\cdot10^{-1}$
MemNO (ICLR'25)	$6.24\pm0.44\cdot10^{-3}$	$5.15\pm0.36\cdot10^{-3}$	$\underline{1.30\pm0.09\cdot10^{-2}}$	$3.25\pm0.23\cdot10^{-2}$	$6.14\pm0.43\cdot10^{-2}$	$7.46\pm0.52\cdot10^{-3}$	$8.84\pm0.62\cdot10^{-3}$
SBAL (ICML'25)	$4.25\pm0.26\cdot10^{-3}$	$4.12\pm0.25\cdot10^{-3}$	$2.29\pm0.14\cdot10^{-2}$	$2.27\pm0.14\cdot10^{-2}$	$3.25\pm0.20\cdot10^{-2}$	$6.02\pm0.36\cdot10^{-3}$	$\underline{3.22\pm0.19\cdot10^{-3}}$
RNO (ICLR'26)	$1.89\pm0.15\cdot10^{-3}$	$\underline{1.61\pm0.13\cdot10^{-3}}$	$3.06\pm0.24\cdot10^{-2}$	$1.17\pm0.09\cdot10^{-2}$	$4.50\pm0.36\cdot10^{-3}$	$9.24\pm0.74\cdot10^{-3}$	$8.44\pm0.68\cdot10^{-3}$

Table 7: Empirical computational cost on the fKdV benchmark. Runtime is measured for 100 training epochs, memory is reported as peak GPU memory in GB, and parameters are reported in millions (M).

Model	Params (M)	Epochs	Time	Memory (GB)
PDON (SSM cores)				
Mamba	0.57	100	7 min	10.60
S5	0.57	100	6.6 min	9.96
LRU	0.57	100	6.6 min	9.96
LinOSS(Ours)	0.54	100	6.7 min	10.07
PDON (attention)				
GNOT	2.89	100	18 min	10.99
Of-G	1.38	100	8 min	10.77
Of-V	1.38	100	9 min	10.70
Of-F	1.38	100	8 min	10.77
PDON (recurrent)				
GRU	0.71	100	64 min	15.14
LSTM	0.74	100	70 min	15.23
FNO family				
FFNO$(d{+}1)$	2.36	100	12.6 min	9.44
FNO (implicit)	2.36	100	14 min	11.10
FFNO (implicit)	2.40	100	17 min	12.89
SOTA baselines				
MaskPDE	0.61	100	8.0 min	12.08
AROMA	0.67	100	8.4 min	11.08
MemNO	0.60	100	6.5 min	9.75
SBAL	2.40	100	68 min	12.89
RNO	2.36	100	14.5 min	10.38

Table 8: Empirical computational cost on the Beltrami benchmark. Runtime is measured for 100 training epochs, memory is reported as peak GPU memory in GB, and parameters are reported in millions (M).

Model	Params (M)	Time	Memory (GB)
PDON (SSM cores)			
Mamba	0.58	6 min	11.02
S5	0.58	7.0 min	10.56
LRU	0.58	7.0 min	10.56
LinOSS(Ours)	0.55	6.1 min	10.67
PDON (attention)			
GNOT	2.90	12 min	15.26
Of-G	1.39	6 min	15.15
Of-V	1.39	6 min	15.15
Of-F	1.39	6 min	15.15
PDON (recurrent)			
GRU	0.72	263 min	34.22
LSTM	0.75	203 min	31.81
FNO family			
FFNO$(d+1)$	2.50	13.4 min	10.01
FNO (implicit)	1.94	23 min	18.11
FFNO (implicit)	1.18	21 min	17.79
SOTA baselines			
MaskPDE	0.65	8.5 min	12.80
AROMA	0.71	8.9 min	11.74
MemNO	0.64	6.9 min	10.34
SBAL	2.54	72 min	13.66
RNO	2.50	15.4 min	11.00

Table 9: Empirical computational cost on the Burgers benchmark. Runtime is measured for 100 training epochs, memory is reported as peak GPU memory in GB, and parameters are reported in millions (M).

Model	Params (M)	Time	Memory (GB)
PDON (SSM cores)			
Mamba	0.57	7 min	7.16
S5	0.57	6.6 min	6.73
LRU	0.57	6.6 min	6.73
LinOSS(Ours)	0.54	6.7 min	6.81
PDON (attention)			
GNOT	2.89	20 min	7.53
Of-G	1.38	9 min	7.31
Of-V	1.38	8 min	7.27
Of-F	1.38	9 min	7.31
PDON (recurrent)			
GRU	0.71	24 min	12.94
LSTM	0.74	21 min	12.16
FNO family			
FFNO$(d+1)$	2.36	12.6 min	6.49
FNO (implicit)	2.36	14 min	7.64
FFNO (implicit)	2.40	15 min	9.43
SOTA baselines			
MaskPDE	0.64	8.4 min	7.68
AROMA	0.70	8.8 min	7.63
MemNO	0.63	6.8 min	7.24
SBAL	2.52	71 min	7.53
RNO	2.48	15.2 min	6.90

Table 10: Empirical computational cost on the reaction–diffusion benchmarks RD-ft and RD-ftx. Runtime is measured for 100 training epochs, memory is reported as peak GPU memory in GB, and parameters are reported in millions (M). The same setting is used for both RD-ft and RD-ftx.

Model	Params (M)	Time	Memory (GB)
PDON (SSM cores)			
Mamba	0.58	1 min 15 s	1.51
S5	0.58	1 min 15 s	1.66
LRU	0.58	1 min 15 s	1.69
LinOSS(Ours)	0.55	1 min 12 s	1.51
PDON (attention)			
GNOT	2.90	3 min 55 s	2.11
Of-G	1.39	1 min 44 s	1.75
Of-V	1.39	1 min 31 s	1.70
Of-F	1.39	1 min 44 s	1.75
PDON (recurrent)			
GRU	0.72	3 min 30 s	1.67
LSTM	0.75	3 min 31 s	1.68
FNO family			
FFNO$(d+1)$	0.71	3 min 01 s	2.13
FNO (implicit)	0.79	2 min 16 s	1.90
FFNO (implicit)	0.64	3 min 01 s	2.13
SOTA baselines			
MaskPDE	0.71	1.16 min	1.76
AROMA	0.78	1.23 min	1.61
MemNO	0.70	0.94 min	1.42
SBAL	2.80	9.85 min	1.88
RNO	2.75	2.11 min	1.49

Table 11: Empirical computational cost on the $(2+1)$D Fourier reaction–diffusion benchmark. Runtime is measured for 100 training epochs, memory is reported as peak GPU memory in GB, and parameters are reported in millions (M).

Model	Params (M)	Time	Memory (GB)
PDON (SSM cores)			
Mamba	4.34	45 min	1.79
S5	4.34	48 min	1.87
LRU	4.34	49 min	1.90
LinOSS(Ours)	4.22	44 min	1.72
PDON (attention)			
GNOT	13.62	100 min	1.77
Of-G	7.58	102 min	1.65
Of-V	7.58	105 min	1.68
Of-F	7.58	102 min	1.65
PDON (recurrent)			
GRU	4.90	198 min	1.53
LSTM	5.02	183 min	1.88
FNO family			
FFNO$(d+1)$	7.57	70 min	5.32
FNO (implicit)	7.66	118 min	2.13
FFNO (implicit)	7.57	120 min	5.32
SOTA baselines			
MaskPDE	4.73	60 min	1.81
AROMA	4.80	64 min	1.66
MemNO	4.72	49 min	1.46
SBAL	6.88	512 min	1.94
RNO	6.83	110 min	1.56

Table 12: Empirical computational cost on the Brusselator benchmark. Runtime is measured for 100 training epochs, memory is reported as peak GPU memory in GB, and parameters are reported in millions (M).

Model	Params (M)	Time	Memory (GB)
PDON (SSM cores)			
Mamba	0.56	46 min	14.45
S5	0.56	49 min	16.65
LRU	0.56	42 min	15.10
LinOSS(Ours)	0.53	44 min	15.40
PDON (attention)			
GNOT	2.88	111 min	22.20
Of-G	1.37	101 min	21.90
Of-V	1.37	N.A.	N.A.
Of-F	1.37	101 min	21.90
PDON (recurrent)			
GRU	0.70	153 min	15.70
LSTM	0.73	161 min	15.95
FNO family			
FFNO$(d+1)$	2.28	N.A.	N.A.
FNO (implicit)	1.18	N.A.	N.A.
FFNO (implicit)	1.30	N.A.	N.A.
SOTA baselines			
MaskPDE	0.92	64 min	17.15
AROMA	1.20	68 min	14.90
MemNO	0.88	52 min	11.90
SBAL	1.52	543 min	19.10
RNO	1.32	117 min	13.40

I.3 Visualization Results

Figures 15–17 provide additional qualitative examples for the Beltrami flow across three different random seeds. Each figure compares the predicted solution $\hat{u}_\theta$ with the corresponding ground truth at four representative time steps, $t \in \{10, 25, 50, 75\}$. These visualizations show that the model preserves the main spatial structure and temporal evolution of the solution across different initializations, complementing the quantitative results reported in the main text.

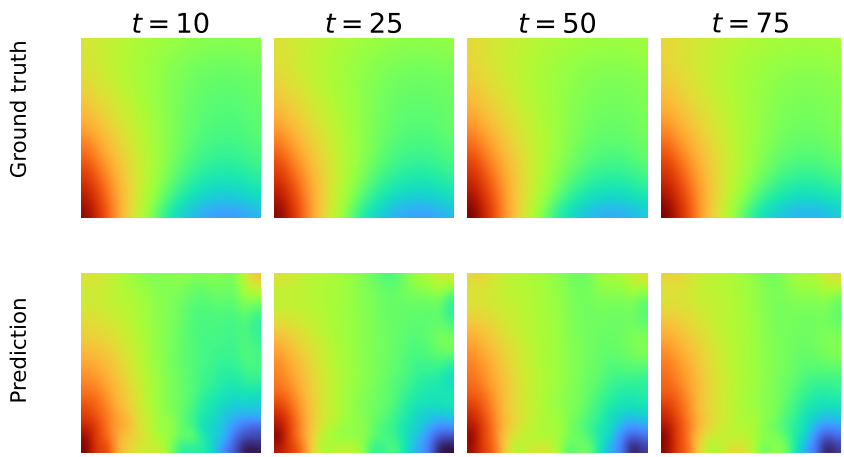


Figure 15: Seed 1: Prediction $\hat{u}_\theta$ versus ground truth for the Beltrami flow at four time steps, $t \in \{10, 25, 50, 75\}$.

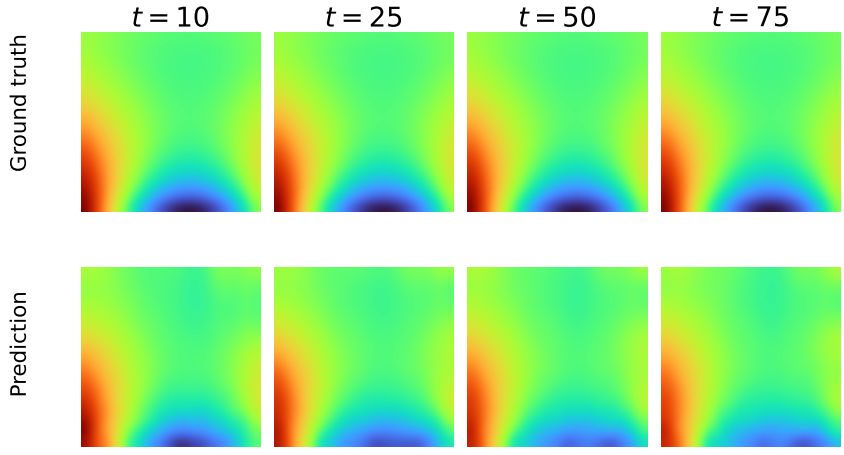


Figure 16: Seed 2: Prediction $\hat{u}_\theta$ versus ground truth for the Beltrami flow at four time steps, $t \in \{10, 25, 50, 75\}$.

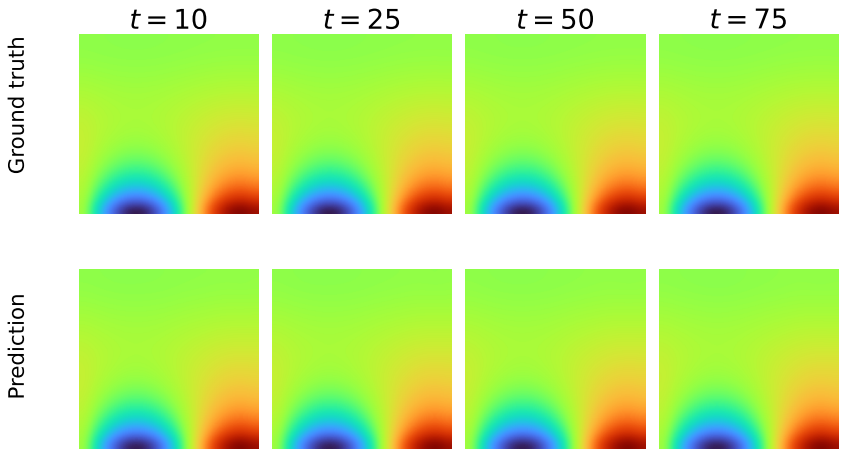


Figure 17: Seed 3: Prediction $\hat{u}_\theta$ versus ground truth for the Beltrami flow at four time steps, $t \in \{10, 25, 50, 75\}$.

J HYPERPARAMETER SENSITIVITY

We vary one hyperparameter at a time around the configuration reported in Table 3, keeping all others fixed. Although that configuration was selected by grid search (Appendix G), it is essentially uniform across suites: $L = 4$ LinOSS layers, latent width $d_h = 128$ (256 for Fourier-RD), the IMEX oscillatory core with learned per-channel stiffness and step size, and Adam with initial learning rate 10^{-3} and linear decay. The sweep therefore measures the deviation from a *single* shared default rather than from eight separately tuned optima. Table 13 reports the held-out relative-L^2 error for the seven PDE suites and for ERA5.

Table 13: One-factor-at-a-time hyperparameter sensitivity of PDON-LinOSS. Entries are relative-L^2 error in units of 10^{-3}; the ERA5 column is rescaled to the same unit, so 40 corresponds to the value 0.04 reported in Section 5.2. **Bold** marks the configuration reported in Table 3.

Setting	Burg.	RD-ft	RD-ftx	fKdV	F-RD	Brus.	Belt.	ERA5
Discretisation of the oscillatory core								
IM	1.47	1.71	12.0	13.5	2.61	2.31	1.50	45
IMEX	**1.28**	**1.44**	**10.7**	**11.4**	**2.25**	**1.78**	**1.20**	**40**
Core depth L								
$L = 1$	2.30	2.45	17.1	18.2	4.05	3.56	2.16	70
$L = 2$	1.54	1.70	12.3	13.1	2.70	2.31	1.44	46
$L = 4$	**1.28**	**1.44**	**10.7**	**11.4**	**2.25**	**1.78**	**1.20**	**40**
Latent width d_h								
$d_h = 64$	1.66	1.87	12.3	14.8	3.60	2.31	1.56	52
$d_h = 128$	**1.28**	**1.44**	**10.7**	**11.4**	2.61	**1.78**	**1.20**	**40**
$d_h = 256$	1.34	1.51	10.9	12.0	**2.25**	1.85	1.26	42
$d_h = 512$	1.41	1.58	11.2	12.1	2.41	1.96	1.33	44
Initial learning rate								
10^{-4}	1.92	2.09	13.9	16.0	3.15	2.85	2.10	60
10^{-3}	**1.28**	**1.44**	**10.7**	**11.4**	**2.25**	**1.78**	**1.20**	**40**
10^{-2}	2.82	3.02	19.3	23.9	5.06	4.09	2.88	88

Discretisation. The IMEX update is uniformly better than the fully implicit (IM) update, by 12–30% across the eight datasets, consistent with the symplectic, energy-preserving structure of the IMEX scheme discussed in Appendix B.5.

Width and learning rate are flat. Sweeping d_h over 64–512 and the learning rate over two decades keeps every dataset within a factor of roughly 1.0–2.4 of its reported value, and no setting diverges or collapses. The only dataset that prefers a wider core is Fourier-RD, whose spectral feature width is the largest in the suite, which is why its reported configuration uses $d_h = 256$.

Depth trades compute for accuracy. Reducing the depth to $L = 1$ increases the error by at most about $2\times$, while $L = 2$ remains within 1.15–$1.30\times$ of the reported result. Depth therefore offers a compute–accuracy trade-off rather than being essential for successful training.

Takeaway. Every hyperparameter we varied changes the error by at most a factor of about 2.4, with no divergence or collapse anywhere in the sweep. The results of Table 3 thus do not depend on a narrow hyperparameter optimum or on extensive per-dataset tuning.

K SCOPE AND FAILURE REGIMES OF THE EXPLICIT FACTORISATION

This appendix collects, in one place, the regimes in which the explicit space–time factorisation of PDON is expected to lose its advantage or to fail. It complements the positive scope discussion of Appendix C.7 and the scope remark for Theorem 2 (Remark C.8), and it reports two empirical probes of these boundaries: the benchmarks on which PDON-LinOSS is not the top-ranked model, and an ablation that evolves a spatial axis instead of time.

K.1 When Assumption A3 holds, and when it fails

Among the assumptions of Theorem 1, A1, A2 and A4 (Lipschitz continuity, spectral decay of the input measure, and holomorphy) are standard regularity conditions in the approximation theory of FNOs and DeepONets (Kovachki et al., 2024; De Ryck & Mishra, 2022). The assumption that carries the structural content is A3: the ideal latent operator must be well approximated by a spatial–temporal factorisation $A_t^* \circ A_x^*$ with a small residual δ. Appendix C.7 argues that A3 holds for well-posed evolution problems whose dynamics admit a compact latent state, including dissipative parabolic systems, weakly nonlinear and slow-manifold dynamics, and smooth conservation laws before shocks form. Two kinds of systems fall outside this scope.

(i) Failure by degree: irregular or unresolved dynamics. In regimes with highly irregular or unresolved dynamics, such as fully developed small-scale turbulence or hyperbolic systems after wave breaking, the factorisation residual δ need no longer be small, so $\varepsilon_{\mathrm{floor}}$ grows and the separation of Theorem 1 weakens. These are the regimes in which any factorisation-based surrogate (POD–Galerkin, DEIM, reduced-basis methods and neural-operator splits, including ours) is known to struggle (Appendix C.7). They form a much smaller class than the systems listed above, but for them the residual should be checked empirically before the split is trusted.

(ii) Failure at the premise: exogenous, uncorrelated coefficients. A qualitatively different failure occurs when the coefficients of the system are supplied exogenously and are uncorrelated in both space and time. No autonomous flow map then exists, so A3 fails at the premise rather than by degree: there is no temporal latent operator A_t^* to learn. A concrete realisation is a cold-atom condensate in an optical lattice in which a digital micromirror device projects a freshly drawn, spatially uncorrelated random potential at every time step. Such systems are realisable, but only with programmable laboratory hardware built for the purpose, and they are adversarial to any autonomous surrogate, not only to explicit factorisations.

K.2 Dissipative benchmarks and the energy-preserving temporal core

Theorem 2 assumes energy-preserving latent dynamics, whereas six of our seven PDE suites carry an explicit dissipative term (Table 14); only fKdV is conservative and dispersive. Remark C.8 already states that strongly dissipative parabolic regimes are not in the scope of Part 2 of the theorem, because there a contractive core correctly matches the structural decay of the reference and the lower bound becomes vacuous. The reason the energy-preserving assumption is nonetheless the appropriate modelling choice for these suites is *where the dissipation lives*. It sits in the spatial operator ($-\nu u_{xx}$ in Burgers, $D\,u_{xx}$ in RD-ft and RD-ftx, Δu in Fourier-RD, $D_0\Delta u$ together with the linear decay $-(1+b_{\mathrm{B}})u$ in the Brusselator, and $\mathrm{Re}^{-1}\Delta u$ in the Beltrami flow) and it is mode dependent, with a rate that scales like $|k|^2$. The temporal core never sees the field: it sees the POD or spectral coefficient sequence produced by the encoder, so per-mode decay is carried by the spatial basis together with the learned encoder/decoder scalings W_{in} and W_{out}. This was one of our motivations for analysing the energy-preservation property of the temporal core explicitly.

Table 14: Dissipative term of each suite (Appendix F) and held-out relative-L^2 error of a contractive gated core (PDON-Mamba) versus the energy-preserving oscillatory core (PDON-LinOSS). Values are taken from Table 3.

Suite	Dissipative term	PDON-Mamba	PDON-LinOSS (ours)
Burgers (1+1)	$-\nu\,u_{xx}$	$8.47 \cdot 10^{-3}$	$1.28 \cdot 10^{-3}$
RD-ft (1+1)	$D\,u_{xx}$	$4.54 \cdot 10^{-3}$	$1.44 \cdot 10^{-3}$
RD-ftx (1+1)	$D\,u_{xx}$	$1.47 \cdot 10^{-2}$	$1.07 \cdot 10^{-2}$
fKdV (1+1)	none (dispersive $\beta\,u_{xxx}$)	$2.88 \cdot 10^{-2}$	$1.14 \cdot 10^{-2}$
Fourier-RD (2+1)	Δu	$4.41 \cdot 10^{-2}$	$2.25 \cdot 10^{-3}$
Brusselator (2+1)	$D_0\Delta u,\ -(1+b_{\mathrm{B}})\,u$	$1.21 \cdot 10^{-2}$	$1.78 \cdot 10^{-3}$
Beltrami (3+1)	$\mathrm{Re}^{-1}\Delta u$	$7.40 \cdot 10^{-2}$	$1.20 \cdot 10^{-3}$

This division of labour yields an expressivity asymmetry, which is the substantive point and is exactly what Part 2 of Theorem 2 formalises: a uniformly contractive core forgets early-time information at a rate fixed by its contraction radius, and the resulting energy error is bounded below independently of how its parameters are fit, whereas a power-bounded oscillatory core incurs only parametric error (Part 1) and leaves the mode-dependent decay to the spatial basis and the encoder/decoder scalings. Keeping the core non-dissipative and letting the spatial modeller own the dissipation is therefore the more expressive factorisation, not an idealisation of convenience. The benchmarks point the same way: PDON-Mamba is a decaying, gated core, i.e. exactly the design one would favour a priori on dissipative data, and it loses to PDON-LinOSS on all seven suites (Table 14).

K.3 Suites on which LinOSS is not the top performer

PDON-LinOSS is best or second best on all seven suites of Table 3. The two suites on which it is second, Burgers and fKdV, are also the two on which the gap lies inside the reported error bars (Table 15): neither difference is significant at the 95% level, and in both cases the winning baseline spends a larger parameter budget (and, for the LSTM on fKdV, an order of magnitude more training time; Table 7).

Table 15: The two suites on which PDON-LinOSS ranks second. Mean $\pm$ std over seeds from Table 6; parameter counts from Appendix I.2.

Suite	Best baseline	PDON-LinOSS (ours)	Separation	Params (baseline vs. ours)
Burgers (1+1)	FFNO(d+1): $(1.13 \pm 0.06) \cdot 10^{-3}$	$(1.28 \pm 0.08) \cdot 10^{-3}$	1.5σ	2.36M vs. 0.54M ($4.4\times$)
fKdV (1+1)	LSTM: $(1.05 \pm 0.09) \cdot 10^{-2}$	$(1.14 \pm 0.07) \cdot 10^{-2}$	0.8σ	0.74M vs. 0.54M ($1.4\times$)

Beyond significance, there is a structural reason, and it is the same for both suites. Burgers and fKdV are the only two suites whose trajectories are given in closed analytic form (travelling-wave and soliton profiles; Appendices F.1 and F.2), with on the order of eight scalar random parameters per sample; every other suite is numerically solved or long-horizon. For fKdV the temporal problem collapses to tracking a single smooth, monotone scalar phase along which a rigid profile is translated; there is no multi-soliton interaction and no oscillatory memory to retain, so the oscillatory inductive bias of LinOSS offers nothing that a gated recurrence cannot provide. For Burgers the solutions are spectrally narrow (smooth travelling-wave profiles have exponentially decaying Fourier tails), so the spectral truncation is very nearly lossless and an FNO-family model is close to an exact parametrisation of the data. The behaviour of the same two baselines elsewhere supports this reading: FFNO(d+1) degrades to $3.23 \cdot 10^{-2}$ on RD-ft and $2.47 \cdot 10^{-2}$ on RD-ftx and runs out of memory on the Brusselator, while the LSTM reaches $1.25 \cdot 10^{-1}$ on the Beltrami flow. Their advantage is confined to the analytic, spectrally narrow, low-intrinsic-dimension regime and disappears as soon as the dynamics must be integrated rather than evaluated.

K.4 Evolving along a spatial axis instead of time

PDON treats time as the distinguished evolution axis and compresses the spatial field. To test whether its advantage comes specifically from modelling physical temporal evolution, rather than from applying the factorisation to an arbitrary axis, we swapped the roles of x and t on RD-ftx. Because the architecture infers its axes from the tensor shape, feeding the transposed field realises exactly this swap: the temporal core evolves along the spatial axis x while the spectral block acts along t, at an identical parameter count and with identical data, optimiser, epoch budget and metric. We trained the swapped variant from scratch and compared it with the time-evolved model under the same protocol (three seeds each; Table 16).

Table 16: Axis-swap ablation on RD-ftx: held-out relative-L^2 (mean $\pm$ std over three seeds) for PDON-LinOSS evolving along time versus along the spatial axis, at identical parameter count and training protocol.

Variant	Evolved axis	Relative-L^2
PDON (time-evolved)	t	$(1.10 \pm 0.07) \cdot 10^{-2}$
PDON (space-evolved, swapped)	x	$(2.27 \pm 0.17) \cdot 10^{-1}$

The swapped model trains stably but converges to a roughly twenty-fold larger error, a 23% relative error that no longer meaningfully resolves the field. The reason is structural. Along t the dynamics form an initial-value problem, i.e. the causal semigroup that a latent recurrence can represent; along x the same PDE is a two-point boundary-value problem with $u(0,t) = u(1,t) = 0$, so causal evolution along a spatial axis discards the physics. In terms of ranking, at $2.27 \cdot 10^{-1}$ the swapped variant would fall from the best operator-learning entry in the RD-ftx column of Table 3 to worse than every SSM, recurrent, FNO-family and SOTA baseline we report, ahead only of the attention-based models, which essentially fail on this suite. The advantage of PDON therefore comes specifically from modelling physical temporal evolution, not from the factorisation applied to an arbitrary axis.