MEMORY NEEDS CONTROL: ADAPTIVE MEMORY GATE FOR SOLVING PDES

Anonymous authors
Paper under double-blind review

ABSTRACT

Neural operators have emerged as a powerful data-driven approach for solving time-dependent partial differential equations (PDEs). Among recent advances, memory-augmented neural operators explicitly incorporate past states and have achieved remarkable performance under low-resolution observation settings. However, existing approaches use a fixed memory weight, which sets how much of the past is mixed into the current state. We show that the benefit of memory depends on the resolution and the physical parameters. Empirically, on Kuramoto–Sivashinsky (KS), no single memory weight is optimal across resolutions and viscosities. Theoretically, the Bayes gain of history is bounded by the unresolved spectral energy. We propose AMGFNO, which controls the memory weight with an adaptive memory gate (AMG). The AMG is the product of a content gate and a frequency-aware gate. The gate is learned during training: It opens where much of the spectrum is unresolved and nearly closes where almost all of the spectrum is resolved. On 20 settings spanning 1D, 2D, and 3D PDEs (KS, Burgers', Navier–Stokes, and compressible CFD), AMGFNO is best or tied in 17. On KS at ν=0.10, f=32, AMGFNO reduces nRMSE by 30% (0.036 to 0.025) over the best fixed-memory baseline. The gate is not specific to S4. Across S4, S5, LRU, LSTM, and a causal Transformer memory, the gate lowers the error of the model without the gate in 38 of 45 settings.

1 INTRODUCTION

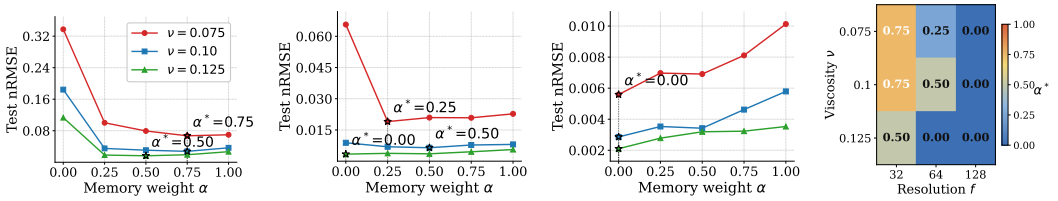
Neural operators have emerged as a powerful data-driven alternative to classical numerical solvers for PDEs (Li et al., 2021; Kovachki et al., 2023). Numerical methods such as finite difference, finite volume, and finite element methods (LeVeque, 2007; 2002; Zienkiewicz et al., 2005) suffer from high computational cost in high dimensions and sensitivity to initial conditions. Neural operators address these limitations by learning solution operators from data, enabling fast inference after a single training phase. Among the most practically important PDE families are time-dependent PDEs, which govern physical phenomena ranging from fluid dynamics to reaction-diffusion systems (Takamoto et al., 2022). For such PDEs, neural operators commonly adopt the Markovian assumption, predicting the next state only from the current state (Tran et al., 2023; Lippe et al., 2023).

Table 1: Memory: uses past states; Long: spans the full trajectory; Adaptive: adjusts memory weight per input. AMGFNO combines all three.

Method	Memory	Long	Adaptive
FFNO	✗	✗	✗
MI-FFNO	✓	✗	✗
S4FFNO	✓	✓	✗
AMGFNO	✓	✓	✓

The Markovian assumption breaks down when PDE solutions are observed at low resolution (Buitrago Ruiz et al., 2025). The observed state is a discretization on a finite grid, and a coarse grid does not carry the full state. As resolution decreases, fewer Fourier modes are observable, and by the Nyquist–Shannon sampling theorem (Shannon, 1949), high-frequency information is lost. Mori–Zwanzig theory formalizes this partial observation as inducing strongly non-Markovian behavior (Parish & Duraisamy, 2017; Mori, 1965; Chorin et al., 2000; 2002; Gouasmi et al., 2017). Accumulated information from past states, the memory term, then becomes decisive for accurate prediction. Past resolved states carry indirect evidence about the unresolved modes.

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.



(a) $f = 32$ (b) $f = 64$ (c) $f = 128$ (d) Optimal α^*

Figure 1: Effect of memory weight α on nRMSE across resolutions f and viscosity ν values on the KS equation. Panels (a-c) show nRMSE as a function of α for three viscosity values ($\nu \in \{0.075, 0.10, 0.125\}$) at a fixed resolution, with stars indicating optimal memory weight α^*. Panel (d) summarizes the optimal memory weight α^* across all settings.

Fixed memory weight cannot adapt to varying conditions. Explicitly modeling memory can therefore compensate for the information lost at low resolution. Existing memory-augmented neural operators use a fixed memory weight that controls how much memory is used, regardless of observation conditions (Buitrago Ruiz et al., 2025). However, the utility of memory varies with resolution and physical parameters. At a coarse grid, many high-frequency modes are unobserved and memory can supply what the current state lacks; at a fine grid, the current state already carries most of the information and memory can even hurt. Section 2.2 presents a preliminary study showing that the optimal memory weight varies with resolution and viscosity, and that no single fixed memory weight simultaneously optimizes performance across all settings (Figure 1). Section 2.3 gives the same conclusion from the other side: for an ideal predictor, the benefit of history is bounded by the spectral energy the grid cannot resolve.

Main idea. We propose Adaptive Memory Gate FNO (**AMGFNO**), a neural operator that controls the memory weight per input via an adaptive memory gate (AMG). The AMG is the product of content gate and frequency-aware gate. The content gate reads the current hidden state and the memory output. For each sample, channel, and spatial position, the content gate decides how much memory to admit. The frequency-aware gate supplies a condition-level prior from $\widehat{\omega}_f$, the fraction of spectral energy that the observation grid cannot resolve. As summarized in Table 1, FFNO uses no memory, Multi-Input FFNO (MI-FFNO) captures only short-range history, and S4FFNO (Tran et al., 2023; Buitrago Ruiz et al., 2025) introduces long-range memory but lacks adaptive control. AMGFNO is the first to combine long-range memory over the full trajectory with input-dependent control of the memory weight. Throughout this paper, we use α to denote the manually-tuned memory weight in our ablation study, α^* for the optimal memory weight that minimizes error per condition, $\bar{g}$ for the mean of AMGFNO's learned gate, reported as a summary statistic, and $\widehat{\omega}_f$ for the unresolved spectral-energy ratio of an observation condition.

The contributions of AMGFNO are summarized as follows:

- **Finding.** We show empirically and theoretically that memory needs control: the optimal memory weight ranges from 0 to 0.75 across resolutions and viscosities (Section 2.2), and we prove that the Bayes gain of history is bounded by the unresolved spectral energy and vanishes at high resolution (Section 2.3).
- **Method.** AMGFNO (Section 3) modulates memory weight with an adaptive memory gate combining a content gate and a frequency-aware gate. The content gate adapts the weight to each input, and the frequency-aware gate sets a prior from the unresolved spectral energy $\widehat{\omega}_f$. The gate adds 0.04M parameters (Appendix F).
- **Results.** On 20 settings spanning 1D KS and Burgers', 2D Navier–Stokes, and 2D/3D compressible CFD, AMGFNO is best or tied in 17 (Section 4.2). The gate opens where much of the spectrum is unresolved and nearly closes where almost all of it is resolved. On KS at ν=0.10, f=32, AMGFNO reduces nRMSE by 30% over S4FFNO (0.036 to 0.025). AMGFNO replaces training a separate memory or memory-free model for each setting.
- **Generality.** The gain is not specific to S4: with S5, LRU, LSTM, or a causal Transformer as the memory layer, the gate lowers nRMSE in 38 of 45 settings, including all 15 at f=32 (Section 4.4).

2 PRELIMINARIES AND MOTIVATION

2.1 TIME-DEPENDENT PDE PROBLEM DEFINITION

We consider a time-dependent partial differential equation defined on a spatial domain $\Omega \subset \mathbb{R}^d$ and a time interval $[0, T]$:

$$\frac{\partial u}{\partial t}(t, x) = \mathcal{L}_\rho[u](t, x), \qquad (t, x) \in [0, T] \times \Omega. \tag{1}$$

The PDE is equipped with an initial condition

$$u(0, x) = u_0(x), \qquad x \in \Omega, \tag{2}$$

and a boundary condition

$$\mathcal{B}[u](t) = 0, \qquad t \in [0, T]. \tag{3}$$

Here $u(t, \cdot) : \Omega \to \mathbb{R}^V$ is the PDE state at time t, x is the spatial coordinate, V is the number of physical state channels, $\mathcal{L}_\rho$ is a differential operator in x parameterized by physical parameters ρ, and $\mathcal{B}$ is the boundary-condition operator. For fixed physical parameters ρ, let Φ_t^ρ denote the solution map, so that

$$u(t, \cdot) = \Phi_t^\rho(u_0).$$

The learning problem is to approximate this solution map from discretized observations of the PDE state. Section 2.3 introduces the resolved-state and spectral-truncation notation.

2.2 PRELIMINARY STUDY: HOW MUCH MEMORY SHOULD BE USED?

To analyze when memory is beneficial, we conduct an ablation study that manually controls the memory weight in S4FFNO (Buitrago Ruiz et al., 2025). S4FFNO follows a FFNO–FFNO–S4–FFNO–FFNO layer structure, where the S4 memory layer is inserted at the middle position ($\ell^* = 3$). Let $\mathbf{h}_t^{(\ell^*-1)} \in \mathbb{R}^{N \times H}$ denote the hidden state at the output of the second FFNO layer, and let $\mathbf{z} = \mathcal{M}(\mathbf{u}_t) \in \mathbb{R}^{N \times H}$ denote the output of the S4 memory branch. We replace the original residual fusion with a convex combination that explicitly controls the relative weight of memory:

$$\tilde{\mathbf{h}}_t \leftarrow \alpha \cdot \mathbf{z}_t + (1 - \alpha) \cdot \mathbf{h}_t, \qquad \alpha \in [0, 1], \tag{4}$$

where $\tilde{\mathbf{h}}_t$ is passed to the third FFNO layer. Here, α is the memory weight: $\alpha = 0$ corresponds to a pure Markovian model that bypasses the memory branch, and $\alpha = 1$ fully replaces the hidden state with the memory output. Note that $\alpha = 1$ in Equation 4 discards the residual path and therefore differs from the original S4FFNO, which corresponds to the additive form of Appendix A at $\alpha = 1$. We sweep $\alpha \in \{0, 0.25, 0.5, 0.75, 1.0\}$ on the Kuramoto–Sivashinsky (KS) equation across three spatial resolutions ($f \in \{32, 64, 128\}$) and three viscosity values ($\nu \in \{0.075, 0.10, 0.125\}$). Results are shown in Figure 1. The same observations hold under an additive parameterization $\mathbf{h}_{t+1} \leftarrow \alpha \cdot \mathbf{z} + \mathbf{h}_t$; see Appendix A for details.

Observation 1: Optimal memory weight varies significantly across conditions. Figure 1(a-c) shows test nRMSE as a function of α for different resolutions and viscosities, with stars marking the optimal memory weight α^*. At low resolution ($f = 32$), the optimal $\alpha^* \in \{0.5, 0.75\}$ improves on pure memory at every viscosity, most clearly at $\nu = 0.125$, where nRMSE drops from 0.027 to 0.016. At high resolution ($f = 128$), $\alpha^* = 0$ for all viscosities. At intermediate resolution ($f = 64$), α^* ranges from 0 to 0.5 depending on viscosity. As the heatmap in Figure 1(d) shows, optimal α^* spans $\{0.0, 0.25, 0.5, 0.75\}$ across the 9 settings—no single fixed value is optimal, motivating adaptive memory gate. This trend holds under both convex and additive fusion (Appendix A).

Observation 2: Fixed α fails to generalize across setting. As the heatmap in Figure 1(d) shows, a fixed α optimized for one condition performs poorly in others. For instance, $\alpha = 0.75$ works well at $f = 32$ (high viscosity) but degrades performance at $f = 128$, while $\alpha = 0$ underexploits memory at low resolution. At intermediate resolution ($f = 64$, Figure 1(b)), optimal α^* varies widely—from 0 to 0.5 across viscosities—illustrating that fine-grained, condition-dependent control is necessary.

Motivation. These findings reveal that memory weight should adapt dynamically to the simulation condition. A fixed α treats memory as a static architectural choice, ignoring that optimal memory usage depends on the observation setting (e.g., resolution, viscosity). We propose an Adaptive Memory Gate Block in Section 3.3 that learns to modulate memory based on the current hidden state, enabling the model to automatically adjust across different observation settings without manual tuning.

2.3 Theoretical Motivation

We quantify the squared-loss Bayes value of history under noiseless Fourier observations and relate it to unresolved spectral content. The argument concerns information available to an ideal predictor, rather than the optimality of a particular memory architecture or gate.

Fix a physical condition ρ and any finite spatial dimension $d \geq 1$ on the periodic domain $\mathbb{T}^d$, using spatial L^2 norms. Here $\Delta t > 0$ is the observation interval, and the integer t indexes observations at physical times $t\Delta t$. For $u^\rho_{t+1} = \Phi^\rho_{\Delta t}(u^\rho_t)$, let P_f be the orthogonal Fourier projection at per-axis resolution scale f, and set $Q_f = I - P_f$. Write

$$u^\rho_{t,f} = P_f u^\rho_t, \qquad Y^{f,\rho}_t = P_f u^\rho_{t+1}, \qquad \mathcal{U}^{f,\rho}_t = (u^\rho_{0,f}, \ldots, u^\rho_{t,f}).$$

The projection residual

$$\delta_{t,f,\rho} := P_f \Phi^\rho_{\Delta t}(u^\rho_t) - P_f \Phi^\rho_{\Delta t}(P_f u^\rho_t) \tag{5}$$

measures the effect of removing unresolved modes before a one-step restart. Motivated by optimal prediction under partial information (Chorin et al., 2000; 2002) at each fixed t, define the *Bayes history gain* by

$$\mathcal{M}_{t,f,\rho} := \inf_{\varphi} \mathbb{E}\big\|Y^{f,\rho}_t - \varphi(u^\rho_{t,f})\big\|^2 - \inf_{\psi} \mathbb{E}\big\|Y^{f,\rho}_t - \psi(\mathcal{U}^{f,\rho}_t)\big\|^2, \tag{6}$$

where both infima range over all square-integrable measurable predictions of the indicated inputs at the same fixed time. Expectations are taken over the initial state $u^\rho_0 \sim \mu_\rho$. The initial-data law and finite prediction horizon remain fixed as f varies.

Proposition 1 (History gain and the projection residual). For square-integrable $Y^{f,\rho}_t$ and $\delta_{t,f,\rho}$ and a measurable solution map,

$$\mathcal{M}_{t,f,\rho} = \mathbb{E}\left\|\mathbb{E}[\delta_{t,f,\rho} \mid \mathcal{U}^{f,\rho}_t] - \mathbb{E}[\delta_{t,f,\rho} \mid u^\rho_{t,f}]\right\|^2. \tag{7}$$

Under the finite-horizon Sobolev regularity and solution-map Lipschitz conditions in Assumptions 1–2,

$$0 \leq \mathcal{M}_{t,f,\rho} \leq \mathbb{E}\|\delta_{t,f,\rho}\|^2 \leq \kappa_{\mathrm{tail},\rho} f^{-2\beta} \xrightarrow[f\to\infty]{} 0, \qquad \beta > 0. \tag{8}$$

History improves squared-loss Bayes prediction exactly when it changes the projection residual's conditional mean with positive probability. Under Assumptions 1–2, this gain vanishes in the high-resolution limit. Appendix B provides the proofs and a predictor-class comparison.

Implication for spectral calibration. Let $\omega_f(\phi) = \|Q_f\phi\|^2_{L^2}/\|\phi\|^2_{L^2}$, with $\omega_f(0) = 0$, and let $\omega^{\mathrm{pop}}_{f,\rho}$ and $\mathcal{M}_{f,\rho}$ be the finite-horizon averages of $\mathbb{E}[\omega_f(u^\rho_t)]$ and $\mathcal{M}_{t,f,\rho}$, respectively. The same assumptions give

$$\mathcal{M}_{f,\rho} \leq \kappa_{\omega,\rho}\big(\omega^{\mathrm{pop}}_{f,\rho}\big)^{\beta/(r+\beta)} \tag{9}$$

where $r \geq 0$ and $\beta > 0$ are the regularity indices in Assumptions 1–2, and $\kappa_{\omega,\rho}$ is independent of f. The bound follows from Appendix B.2. Small unresolved spectral-energy ratios therefore constrain the squared-loss Bayes benefit of history, motivating a condition-level spectral prior for memory modulation. Section 3.3 combines this prior with a content-dependent gate.

3 Proposed Method

AMGFNO builds on the FFNO backbone by inserting an Adaptive Memory Gate Block (AMG Block) at the middle layer, as illustrated in Figure 2. The AMG Block consists of a memory layer and an adaptive memory gate (AMG). The AMG consists of content gate and frequency-aware gate.

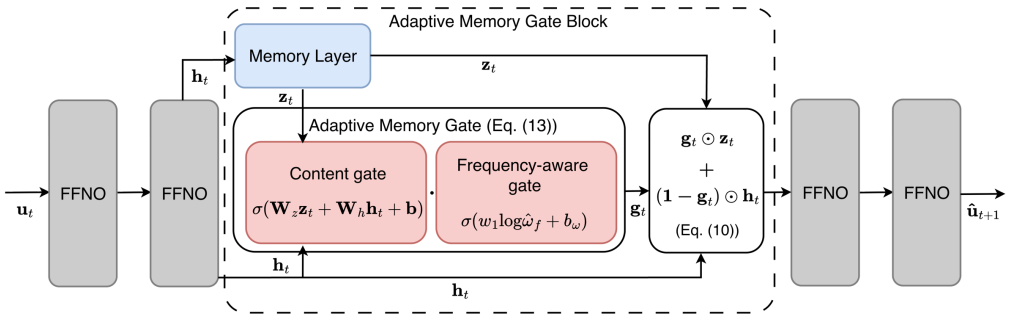


Figure 2: Overview of the AMGFNO architecture. The model consists of five layers with an Adaptive Memory Gate Block (AMG Block) at the middle layer ($\ell = 3$). The AMG Block combines the memory layer output $\mathbf{z}_t$ and the Markovian state $\mathbf{h}_t$ (output of the second FFNO layer, $\ell = 2$) using an adaptive memory gate $\mathbf{g}_t$ that consists of two components: a content gate $\sigma(\mathbf{W}_z\mathbf{z}_t + \mathbf{W}_h\mathbf{h}_t + \mathbf{b})$ for input-adaptive control and a frequency-aware gate $\sigma(w_1 \log \hat{\omega}_f + b_\omega)$ for conditional priors. The final output is computed as $\mathbf{g}_t \odot \mathbf{z}_t + (1 - \mathbf{g}_t) \odot \mathbf{h}_t$.

The input PDE state $\mathbf{u}_t \in \mathbb{R}^{S\times V}$ is first encoded into a latent representation $\mathbf{h}_t^{(0)} = \mathrm{Enc}(\mathbf{u}_t, \mathbf{s}) \in \mathbb{R}^{S\times H}$, where $\mathbf{s} \in \mathbb{R}^{S\times d}$ where S is the total number of spatial locations, $\mathbf{s} \in \mathbb{R}^{S\times d}$ contains their coordinates, and H is the hidden dimension. The latent state then passes through $L = 5$ layers in the order FFNO–FFNO–AMG–FFNO–FFNO:

$$\mathbf{h}_t^{(\ell)} = \begin{cases} \mathcal{F}_\ell\left(\mathbf{h}_t^{(\ell-1)}\right) + \mathbf{h}_t^{(\ell-1)}, & \ell \neq \ell^* \\ \mathbf{g}_t \odot \mathbf{z}_t + (1 - \mathbf{g}_t) \odot \mathbf{h}_t^{(\ell-1)}, & \ell = \ell^* \end{cases} \tag{10}$$

where $\ell^* = 3$ is the index of the Adaptive Memory Gate (AMG) Block and $\mathbf{z}_t = \mathcal{M}(\mathbf{h}_{1:t}^{(\ell^*-1)})$ is the memory layer output. At $\ell \neq \ell^*$, each layer applies an FFNO operation with a residual connection. At $\ell = \ell^*$, the AMG Block combines the memory output $\mathbf{z}_t$ and the Markovian hidden state $\mathbf{h}_t^{(\ell^*-1)}$ via the adaptive memory gate $\mathbf{g}_t \in [0, 1]^{S\times H}$, which is computed from the current feature state. The gate dynamically adjusts memory weight across different simulation conditions, allowing the model to modulate memory usage based on the learned representation. The final prediction is $\hat{\mathbf{u}}_{t+1} = \mathrm{Dec}(\mathbf{h}_t^{(L)})$.

3.1 Factorized Fourier Neural Operator (FFNO) Layer

Each FFNO layer $\mathcal{F}_\ell$ operates along the *spatial* axis and combines a spectral path with a local linear path:

$$\mathcal{F}_\ell(\mathbf{h}) = \mathcal{F}^{-1}\left(\mathbf{W}_\ell^{\mathrm{freq}} \cdot [\mathcal{F}(\mathbf{h})]_{1:K}\right) + \mathbf{W}_\ell^{\mathrm{loc}}\,\mathbf{h}, \tag{11}$$

where $\mathcal{F}$ is the discrete Fourier transform along the spatial dimension, K is the number of retained Fourier modes, and $\mathbf{W}_\ell^{\mathrm{freq}}$, $\mathbf{W}_\ell^{\mathrm{loc}}$ are learnable weight matrices.

3.2 Memory Layer

The memory output $\mathbf{z}_t = \mathcal{M}(\mathbf{h}_{1:t})$ is computed by applying S4 (Structured State Space Model) along the *temporal* axis. For each spatial position $p \in \{1, \ldots, S\}$ independently:

$$\mathbf{v}_t^{(p)} = \mathbf{A}\,\mathbf{v}_{t-1}^{(p)} + \mathbf{B}\,\mathbf{h}_t^{(p)}, \qquad \mathbf{z}_t^{(p)} = \mathbf{C}\,\mathbf{v}_t^{(p)}, \tag{12}$$

where $\mathbf{v}_t^{(p)} \in \mathbb{R}^N$ is the S4 internal state that accumulates temporal context up to time t, and $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$ are learnable parameters initialized with HiPPO (Gu et al., 2020). The spatial dimension is treated as a batch axis, so all positions share the same SSM parameters.

3.3 Adaptive Memory Gate

As shown in Figure 2, the adaptive memory gate $\mathbf{g}_t \in (0,1)^{S\times H}$ controls how much the memory output $\mathbf{z}_t$ contributes relative to the current hidden state $\mathbf{h}_t$. The gate consists of a content gate and a frequency-aware gate:

$$\mathbf{g}_t = \underbrace{\sigma(\mathbf{W}_z\,\mathbf{z}_t + \mathbf{W}_h\,\mathbf{h}_t + \mathbf{b})}_{\text{content gate} \in (0,1)^{S\times H}} \cdot \underbrace{\sigma(w_1 \log(\widehat{\omega}_{f,\rho} + \varepsilon_{\text{num}}) + b_\omega)}_{\text{frequency-aware gate} \in (0,1)}. \tag{13}$$

Content gate. The content gate $\sigma(\mathbf{W}_z\mathbf{z}_t + \mathbf{W}_h\mathbf{h}_t + \mathbf{b}) \in (0,1)^{S\times H}$ provides input-adaptive control that varies per sample, channel, and spatial location. It takes the memory output $\mathbf{z}_t$ and the current hidden state $\mathbf{h}_t$ as input and learns to modulate memory weight based on their learned representations. The parameters $\mathbf{W}_z$, $\mathbf{W}_h$, and $\mathbf{b}$ are initialized to zero, allowing the content gate to learn data-driven corrections during training while deferring to the frequency-aware prior at initialization.

Frequency-aware gate. The frequency-aware gate $\sigma(w_1 \log(\widehat{\omega}_{f,\rho} + \varepsilon_{\text{num}}) + b_\omega) \in (0,1)$ supplies a condition-level prior based on the observation condition (f,ρ). Here $\widehat{\omega}_{f,\rho}$ is the empirical spectral-energy ratio defined in Eq. (45), obtained by averaging the frame-wise spectral ratios over the training trajectories and calibration timesteps. This value is held fixed during both training and inference. It is the finite-reference empirical counterpart of the population spectral statistic used in Section 2.3. We found empirically that this fixed estimate consistently outperforms per-batch computation by 5–20% across settings (see Appendix D.4), suggesting that $\widehat{\omega}_{f,\rho}$ serves best as a calibration constant characterizing the observation condition rather than a dynamic per-sample feature.

4 Experiments

4.1 Experimental Setup

Benchmarks and observation settings. We evaluate on 1D Kuramoto–Sivashinsky (KS) and Burgers' equations, 2D incompressible Navier–Stokes, and 2D/3D compressible CFD, all with periodic boundary conditions. We vary spatial resolution and physical parameters to assess memory under different observation conditions. Dataset parameters and observation grids are detailed in Appendices C.1 and C.2.

Training objective and procedure. All models are trained with teacher forcing and a one-step relative ℓ_2 objective (nRMSE). At test time, models are rolled out autoregressively from the initial state. We report nRMSE averaged over test trajectories and benchmark-specific evaluation windows, with mean and standard deviation over three seeds. Loss definitions, rollout procedures, and training protocols are provided in Appendix C.3.

Baseline comparisons. We compare AMGFNO against Markovian operators—FactFormer (Li et al., 2023b), Galerkin Transformer (GKT) (Cao, 2021), U-Net (Gupta & Brandstetter, 2023), and FFNO (Tran et al., 2023)—and memory-augmented operators—Multi-Input FFNO (MI-FFNO) and S4FFNO (Buitrago Ruiz et al., 2025). All baselines use comparable model capacity and identical training hyperparameters within each benchmark. Architectural details are provided in Appendix C.4.

4.2 Main Results

AMGFNO is best or tied in 17 of the 20 settings in Table 2. In the remaining three it trails the best entry by at most 0.003. No baseline is uniformly best or uniformly reliable: each wins in some regime and falls far behind in another. AMGFNO is the only model that stays near the best entry across all four regimes.

The gate adapts to resolution and viscosity without manual tuning. On KS at $f{=}32$, where the grid misses more than half of the spectral energy, S4FFNO is far ahead of FFNO and AMGFNO opens its gate ($\bar{g}{\approx}0.7$) to surpass both. At $f{=}128$, where almost no energy is unresolved, fixed

Table 2: Rollout nRMSE on 1D, 2D, and 3D PDEs, mean±std over three seeds (seeds 155–157). AMGFNO is best or tied in 17 of the 20 settings. $\widehat{\omega}_f$ is the fraction of spectral energy the grid cannot resolve (Appendix C.5). $\bar{g}$ is the mean of AMGFNO's learned gate, reported only as a summary; the gate itself varies per input, channel, and position (Eq. 42). †One of three seeds diverges; the value averages the other two. "–" marks a setting we did not run: FactFormer at f=64 (an order of magnitude behind already at f=32).

					Markovian				Memory		
Dataset	Setting	f	$\widehat{\omega}_f$	$\bar{g}$	FactFormer	U-Net	GKT	FFNO	MI-FFNO	S4FFNO	**AMGFNO**
KS (1D)	ν=0.075	32	0.63	0.69	0.306±.006	0.400±.004	0.432±.011	0.347±.008	0.084±.005	0.086±.009	**0.067**±.006
		64	0.19	0.47	0.110±.007	0.124±.013	0.404±.020	0.066±.004	0.033±.004	0.021±.003	**0.020**±.002
		128	1.9e-3	0.003	0.076±.006	0.045±.001	0.025±.003	**0.006**±.000	0.016±.000	0.007±.001	**0.006**±.000
	ν=0.10	32	0.58	0.70	0.155±.008	0.258±.004	0.327±.001	0.193±.010	0.038±.001	0.036±.001	**0.025**±.002
		64	0.083	0.22	0.044±.003	0.040±.001	0.035±.001	0.010±.000	0.010±.001	0.007±.001	**0.006**±.001
		128	3e-4	0.001	0.030±.004	0.023±.001	0.010±.001	**0.003**±.000	0.008±.000	0.004±.000	**0.003**±.000
	ν=0.125	32	0.54	0.70	0.106±.010	0.187±.002	0.248±.004	0.131±.008	0.025±.001	0.018±.001	**0.016**±.002
		64	0.041	0.07	0.021±.001	0.022±.001	0.012±.000	**0.003**±.000	0.007±.002	**0.003**±.000	0.004±.000
		128	2e-4	<0.001	0.021±.003	0.015±.001	0.006±.001	**0.002**±.000	0.006±.000	0.003±.000	**0.002**±.000
Burgers' (1D)	ν=0.001	32	6.2e-3	0.027	0.167±.016	0.180±.006	0.256±.005	0.173±.004	0.088±.004	0.025±.001	**0.023**±.001
		64	1.2e-3	0.017	0.135±.008	0.142±.007	0.290±.058	0.123±.002	0.033±.002	0.019±.000	**0.016**±.001
		128	2e-4	0.002	0.096±.008	0.061±.003	0.225±.060	0.035±.002	0.015±.001	0.016±.001	**0.014**±.000
NS (2D)	$\nu=10^{-5}$	32	1.5e-4	0.002	0.677±.009	0.041±.000	0.240±.129	0.021±.002	**0.019**±.000	0.020±.001	0.020±.001
		64	8.3e-6	<0.001	–	0.022±.001	0.167±.009	**0.008**±.002	0.009±.001	**0.008**±.001	**0.008**±.001
	$\nu=10^{-4}$	32	2.4e-5	<0.001	0.685±.014	0.023±.001	0.181±.024	**0.010**±.002	**0.010**±.002	0.011±.003	**0.010**±.001
		64	4.7e-7	<0.001	–	0.020±.001	0.164±.015	0.007±.003	0.010±.000	**0.006**±.001	0.009±.003
	$\nu=10^{-3}$	32	4.0e-7	<0.001	0.736±.004	0.053±.056	0.160±.009	0.008±.001	0.008±.004	0.007±.003	**0.006**±.002
		64	2.6e-11	<0.001	–	0.040±.038	0.161±.012†	**0.008**±.003	**0.008**±.002	0.010±.002	**0.008**±.002
CFD (2D)	`Turb`	32	9.1e-3	0.10	0.531±.002†	0.708±.000	1.100±.139	0.451±.001†	0.453±.002†	0.438±.008	**0.433**±.003
CFD (3D)	`Turb`	16	1.9e-2	0.22	0.554±.009	0.772±.006	0.686±.005	0.502±.003	0.480±.005	0.465±.001	**0.464**±.004

memory hurts and FFNO overtakes S4FFNO; the gate now closes ($\bar{g}\leq 0.003$) and AMGFNO matches FFNO. At the intermediate resolution f=64, the gate separates the three viscosities ($\bar{g}$=0.47, 0.22, 0.07), tracking how much energy each leaves unresolved. A single trained model thus reproduces S4FFNO where memory helps and FFNO where it does not, choosing which from the data alone. $\bar{g}$ decreases monotonically with $\widehat{\omega}_f$ within every viscosity.

A small gate value can still recover most of memory's benefit. On Burgers', memory helps even though the grid resolves most of the spectrum. $\widehat{\omega}_f$ is small at every resolution, comparable to KS at f=128, yet S4FFNO is far ahead of FFNO. AMGFNO is best at all three resolutions. Its mean gate value is small but nonzero ($\bar{g} \leq 0.03$), and this is enough to recover most of S4FFNO's advantage over FFNO. On KS at f=128, a similarly small mean gate value coincides with memory hurting the model. Burgers' and KS at f=128 thus have similar $\widehat{\omega}_f$ but opposite effects of memory. The spectral bound in Equation 9 is consistent with this: it scales with $\kappa_{\omega,\rho}$, which depends on the PDE and its physical parameters, so the same $\widehat{\omega}_f$ can allow different memory gains for different PDEs.

Unhelpful memory leaves the gate closed. On 2D Navier–Stokes, $\widehat{\omega}_f \leq 1.5\times10^{-4}$ in every setting, and FFNO and S4FFNO perform almost identically. Memory itself offers nothing here. The gate closes accordingly ($\bar{g} \leq 0.002$ throughout), and AMGFNO stays in the same range as both, losing only two settings and only within one standard deviation. Where memory does not help, AMGFNO does not lose accuracy relative to the memory-free and fixed-memory models.

Unresolved energy opens the gate in 2D and 3D. On 2D and 3D compressible CFD, the unresolved spectral-energy ratio is $\widehat{\omega}_f = 9.1\times10^{-3}$ and 1.9×10^{-2}, and the mean gate value is $\bar{g}$=0.10 and 0.22. S4FFNO has lower error than FFNO in both settings (0.438 vs. 0.451 in 2D, 0.465 vs. 0.502 in 3D), so memory is useful here. AMGFNO has the lowest error in both settings. AMGFNO improves on S4FFNO in 2D (0.433 vs. 0.438) and is within 0.001 of S4FFNO in 3D (0.464 vs. 0.465). On 1D PDEs, a larger $\widehat{\omega}_f$ likewise goes with a larger mean gate value and a benefit from memory. The CFD results show the same pattern in 2D and 3D, with four and five physical fields instead of one.

AMGFNO is best or tied with the better of S4FFNO and FFNO in most settings, and it never falls far behind either. Neither fixed choice is safe. S4FFNO trails FFNO on KS at f=128, and FFNO trails S4FFNO on KS at f=32 and on Burgers'. The best amount of memory therefore depends on the PDE, its physical parameters, and the resolution. AMGFNO adjusts the amount of memory automatically during training.

4.3 Analysis of the Adaptive Memory Gate

This section examines two questions on the 1D benchmarks, where the sweep over resolution and viscosity gives the finest-grained view. First, how are the gate values learned during training? Second, does AMGFNO match the best fixed memory weight without a sweep?

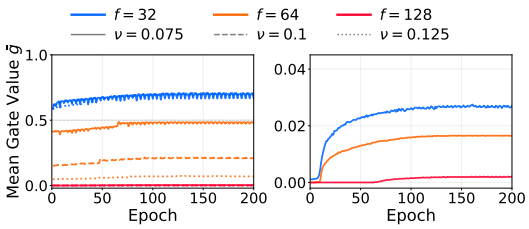


Figure 3: Mean gate value $\bar{g}$ during training on KS (left) and Burgers' (right), by resolution f (color) and, on KS, viscosity ν (line style). On KS the gate separates by f within the first few epochs, settling near $\bar{g}\approx 0.7$ at f=32 and $\bar{g}\leq 0.003$ at f=128. On Burgers', where $\widehat{\omega}_f$ is small throughout, curves start near zero and separate only later in training.

Learned gate values. Figure 3 shows $\bar{g}$ over training. $\bar{g}$ is the mean of AMGFNO's learned gate, reported only as a summary; the gate itself varies per input, channel, and position (Eq. 42). On KS (left), $\bar{g}$ is distinct by resolution and viscosity from epoch 0 and is only fine-tuned afterwards. It starts large at f=32 ($\bar{g} \approx 0.60$–0.70 for all ν), depends on viscosity at f=64 ($\approx 0.47, 0.21, 0.07$ for ν=0.075, 0.1, 0.125), and starts near zero at f=128 for all ν. On Burgers' (right, ν=0.001), $\bar{g}$ starts near zero at every resolution and then rises, substantially at low resolution and hardly at all at high resolution, so the gate values are learned during training rather than set by initialization. Appendix E.1 shows the mean gate and test nRMSE over training for all six KS and Burgers' settings.

Comparison with the best fixed weight. We compare AMGFNO with S4FFNO trained at the best tested fixed weight α^* from Section 2.2, selected separately for each of the nine KS settings. AMGFNO reaches comparable nRMSE in every setting: it is lower than this oracle in four of the nine and higher in five, with the largest gap at f=64, ν=0.075 (0.0208 vs. 0.0190, a 9.4% increase). AMGFNO does so with one training run per setting, without the five-point sweep that α^* requires. Appendix E.2 (Figure 6) shows all nine KS settings. Appendix D reports the gate ablations: removing each gate factor, replacing the gate with its mean, resetting the recurrent state, and computing $\widehat{\omega}_f$ per batch.

4.4 Gate Ablation across Memory Layers

Is the benefit of adaptive gating specific to S4? We replace the S4 memory layer with S5 (Smith et al., 2023), LRU (Orvieto et al., 2023), LSTM, and a causal Transformer, keep everything else fixed, and train each of the five memory layers with the original residual fusion (ungated) and with the adaptive memory gate (gated). Table 3 reports nRMSE on KS for all 45 settings (five memory layers $\times$ three resolutions $\times$ three viscosities). Adding the gate improves performance in most settings regardless of the memory layer: it lowers nRMSE in 38 of the 45 settings, including all 15 settings at f=32, where the reduction averaged over the three viscosities ranges from 10% (S5) to 48% (LSTM). At f=128, it also lowers the error of S4, S5, and LRU by 21–32% on average. The gate adds 0.03–0.04M parameters (under 1% of the model) (Appendix F).

The gate makes the model worse in six settings. All six are at $f \geq 64$, where memory contributes less (Section 4.2) and the errors are already small, and the gap is at most 0.004 nRMSE. Four of the six are LSTM, whose ungated version is already among the lowest-error models at these settings. A possible reason is that LSTM has its own internal gates. Finally, the best gated model at f=32 uses the Transformer memory rather than S4, with 32–39% lower nRMSE than gated S4. We nevertheless keep S4 as the default, because the Transformer memory requires $1.5\times$ the training time of S4 and scales quadratically with the trajectory length, and because S4 allows a direct comparison with S4FFNO (Buitrago Ruiz et al., 2025).

Table 3: nRMSE on KS with different memory layers, without (w/o) and with (w/) the adaptive memory gate (AMG), by resolution f and viscosity ν (sub-columns). The lower value of each pair is in bold; cells where the gate improves over the ungated variant are shaded.

		$f = 32$			$f = 64$			$f = 128$		
Memory	Gate	0.075	0.10	0.125	0.075	0.10	0.125	0.075	0.10	0.125
S4	w/o	0.0860	0.0362	0.0188	0.0201	0.0068	**0.0035**	0.0072	0.0034	0.0031
	w/	**0.0666**	**0.0254**	**0.0159**	**0.0198**	**0.0061**	**0.0035**	**0.0061**	**0.0029**	**0.0021**
S5	w/o	0.0684	0.0259	0.0170	0.0271	0.0102	0.0040	0.0066	0.0043	0.0026
	w/	**0.0569**	**0.0243**	**0.0158**	**0.0204**	**0.0061**	**0.0033**	**0.0054**	**0.0028**	**0.0020**
LRU	w/o	0.1118	0.0419	0.0220	**0.0328**	0.0078	0.0046	0.0079	0.0044	0.0030
	w/	**0.0741**	**0.0288**	**0.0173**	0.0367	**0.0076**	**0.0033**	**0.0058**	**0.0028**	**0.0020**
LSTM	w/o	0.2707	0.0777	0.0373	0.0324	**0.0060**	**0.0029**	0.0061	**0.0026**	**0.0020**
	w/	**0.1666**	**0.0377**	**0.0167**	**0.0312**	0.0064	0.0034	**0.0054**	0.0031	0.0032
Transformer	w/o	0.0526	0.0207	0.0135	0.0193	0.0075	0.0044	**0.0066**	0.0035	0.0024
	w/	**0.0404**	**0.0173**	**0.0103**	**0.0144**	**0.0063**	**0.0031**	0.0067	**0.0028**	**0.0022**

5 Related Work

The Fourier Neural Operator (FNO) (Li et al., 2021) parameterizes integral kernels in Fourier space, while FFNO (Tran et al., 2023) reduces their cost through separable spectral layers. Other architectures rely on multiscale U-Net features (Gupta & Brandstetter, 2023), Galerkin Transformer's linear attention (Cao, 2021), OFormer's self- and cross-attention (Li et al., 2023a), and FactFormer's axial factorization (Li et al., 2023b). Despite these spatial inductive biases, current-state-only prediction can be inadequate when observations do not fully resolve the underlying dynamics.

Memory-augmented operators address this by incorporating past states. Multi-Input FFNO (Buitrago Ruiz et al., 2025) concatenates a short history, whereas S4FFNO inserts an S4 layer (Gu et al., 2022b) between FFNO layers to encode temporal dependencies, with evidence that history helps under partial observation, particularly at low resolution. SS-NO (Koren & Lanthaler, 2026) extends state-space modeling to both space and time with bidirectional spatial state-space layers and a temporal memory module. Our preliminary study shows that the optimal weight for fusing memory with current features varies across observation conditions, whereas S4FFNO fuses them by residual addition without an input-dependent gate.

Input-dependent control of information retention is central to LSTM (Hochreiter & Schmidhuber, 1997), GRU (Cho et al., 2014), Highway Networks (Srivastava et al., 2015), and Mamba (Gu & Dao, 2024). In neural operators, TF-SNO (Zhou et al., 2026) uses current-state statistics to modulate spectral responses. AMGFNO instead applies adaptive control to temporal-memory fusion: a content gate conditions on current features and memory, and a frequency-aware gate incorporates a condition-level spectral prior, so that memory contributes according to both the input and the observation condition. Further details are in Appendix G.

6 Conclusion

We showed both empirically and theoretically that memory in neural operators for time-dependent PDEs should be controlled rather than fixed. On KS, the optimal fixed memory weight of S4FFNO depends on resolution and viscosity. For an ideal predictor, unresolved spectral energy bounds the Bayes gain of history from above, and this gain decays as the resolution increases. AMGFNO replaces the fixed residual fusion of S4FFNO with an adaptive memory gate, the product of a content gate and a frequency-aware gate. The gate opens where much of the spectrum is unresolved and nearly closes where almost all of the spectrum is resolved. On 20 settings spanning 1D, 2D, and 3D PDEs, AMGFNO is best or tied in 17. On KS at ν=0.10, f=32, AMGFNO reduces nRMSE by 30% (0.036 to 0.025) over S4FFNO. The gate also lowers the error of S5, LRU, LSTM, and Transformer memory layers in 38 of 45 settings.

Future work includes estimating $\widehat{\omega}_f$ without high-resolution references, extending the analysis from the ideal predictor to trained models, and pairing the gate with cheaper long-range memory layers (Appendix H).

AI USE STATEMENT

We used generative AI tools to assist with writing, improve the clarity and readability of the manuscript, and help identify related work. Also, AI tools were used to some extent to assist with code development and debugging. The authors reviewed all AI-assisted text and code, and take full responsibility for the final manuscript and implementation.

ETHICS STATEMENT

This work develops neural operators for scientific simulation. Our experiments use a combination of numerically generated PDE trajectories and publicly available benchmark datasets, including PDEBench, and involve no human participants or personal data. The original datasets, numerical methods, and baseline methods are appropriately cited in the paper. Applications in safety-critical scientific and engineering settings should validate prediction accuracy and rollout stability under the relevant physical and observation conditions.

REPRODUCIBILITY STATEMENT

We release the main code in an anonymous repository ([TODO: anonymous link]). The AMGFNO architecture and adaptive memory gate are described in Section 3, and the training objective and autoregressive evaluation metric are defined in Section 4.1. Appendix B states the assumptions and provides proofs for the theoretical results in Section 2.3. Appendix C documents the datasets, data-generation and preprocessing procedures, train/test splits, observation resolutions, spectral-ratio computation, baseline architectures, and training and evaluation protocols. Additional fusion and gate ablations are presented in Appendices A and D, with further gate analysis in Appendix E. Parameter counts and training-time measurements are reported in Appendix F. Random seeds and aggregation procedures are specified in the experimental descriptions and table captions.

REFERENCES

Johannes Brandstetter, Max Welling, and Daniel E Worrall. Lie point symmetry data augmentation for neural pde solvers. In *International Conference on Machine Learning*, pp. 2241–2256. PMLR, 2022.

Ricardo Buitrago Ruiz, Tanya Marwah, Albert Gu, and Andrej Risteski. On the benefits of memory for modeling time-dependent pdes. In *International Conference on Learning Representations*, volume 2025, pp. 54972–55002, 2025.

Shuhao Cao. Choose a transformer: Fourier or galerkin. *Advances in neural information processing systems*, 34:24924–24940, 2021.

Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1724–1734, 2014.

Alexandre J Chorin, Ole H Hald, and Raz Kupferman. Optimal prediction and the mori–zwanzig representation of irreversible processes. *Proceedings of the National Academy of Sciences*, 97(7): 2968–2973, 2000.

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

James W. Cooley, Peter A. W. Lewis, and Peter D. Welch. The finite fourier transform. *IEEE Transactions on Audio and Electroacoustics*, 17(2):77–85, 1969.

Ayoub Gouasmi, Eric J Parish, and Karthik Duraisamy. A priori estimation of memory effects in reduced-order models of nonlinear systems using the mori–zwanzig formalism. *Proceedings. Mathematical, Physical, and Engineering Sciences*, 473(2205):20170385, 2017.

Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. In *Conference on Language Modeling (COLM)*, 2024.

Albert Gu, Tri Dao, Stefano Ermon, Atri Rudra, and Christopher Ré. HiPPO: Recurrent memory with optimal polynomial projections. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

Albert Gu, Karan Goel, Ankit Gupta, and Christopher Ré. On the parameterization and initialization of diagonal state space models. *Advances in neural information processing systems*, 35:35971–35983, 2022a.

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

Jayesh K. Gupta and Johannes Brandstetter. Towards multi-spatiotemporal-scale generalized PDE modeling. *Transactions on Machine Learning Research*, 2023. ISSN 2835-8856.

Ernst Hairer and Gerhard Wanner. *Solving Ordinary Differential Equations II: Stiff and Differential-Algebraic Problems*. Springer-Verlag, 1996.

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

Nodens Koren and Samuel Lanthaler. Merging memory and space: A state space neural operator. *Transactions on Machine Learning Research*, 2026. ISSN 2835-8856.

Nikola Kovachki, Zongyi Li, Burigede Liu, Kamyar Azizzadenesheli, Kaushik Bhattacharya, Andrew 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.

Randall J LeVeque. *Finite volume methods for hyperbolic problems*, volume 31. Cambridge university press, 2002.

Randall J LeVeque. *Finite difference methods for ordinary and partial differential equations: steady-state and time-dependent problems*. SIAM, 2007.

Zijie Li, Kazem Meidani, and Amir Barati Farimani. Transformer for partial differential equations' operator learning. *Transactions on Machine Learning Research*, 2023a.

Zijie Li, Dule Shu, and Amir Barati Farimani. Scalable transformer for PDE surrogate modeling. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023b.

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

Kevin K Lin and Fei Lu. Data-driven model reduction, wiener projections, and the koopman-mori-zwanzig formalism. *Journal of Computational Physics*, 424:109864, 2021.

Phillip Lippe, Bas Veeling, Paris Perdikaris, Richard Turner, and Johannes Brandstetter. Pde-refiner: Achieving accurate long rollouts with neural pde solvers. *Advances in Neural Information Processing Systems*, 36:67398–67433, 2023.

Ilya Loshchilov and Frank Hutter. SGDR: Stochastic gradient descent with warm restarts. In *International Conference on Learning Representations*, 2017.

Hazime Mori. Transport, collective motion, and brownian motion. *Progress of theoretical physics*, 33(3):423–455, 1965.

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. PMLR, 2023.

Eric J Parish and Karthik Duraisamy. Non-markovian closure models for large eddy simulations using the mori-zwanzig formalism. *Physical Review Fluids*, 2(1):014604, 2017.

William E. Schiesser. *The Numerical Method of Lines*. Academic Press, 1991.

Claude E Shannon. Communication in the presence of noise. *Proceedings of the IRE*, 37(1):10–21, 1949.

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

Rupesh K. Srivastava, Klaus Greff, and Jürgen Schmidhuber. Training very deep networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 28, 2015.

Makoto Takamoto, Timothy Praditia, Raphael Leiteritz, Daniel MacKinlay, Francesco Alesiani, Dirk Pflüger, and Mathias Niepert. PDEBench: An extensive benchmark for scientific machine learning. In *Advances in Neural Information Processing Systems*, volume 35, 2022.

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

Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, et al. SciPy 1.0: Fundamental algorithms for scientific computing in python. *Nature Methods*, 17:261–272, 2020.

Yitian Zhou, Chaoning Zhang, Zhenzhen Huang, Haoxuan Yu, Jiaquan Zhang, Yiran Li, Fan Mo, Kuien Liu, Jie Zou, Caiyan Qin, et al. Tf-sno: Time-frequency gated spectral neural operators for learning non-stationary partial differential equations. In *Proceedings of the 32nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*, pp. 13089–13098, 2026.

O. C. Zienkiewicz, R. L. Taylor, and J. Z. Zhu. *The Finite Element Method: Its Basis and Fundamentals*. Elsevier Butterworth-Heinemann, 6 edition, 2005.

APPENDIX

A Preliminary Study under Additive Fusion **12**

B Supporting Analysis for the Theoretical Motivation **13**

C Experimental Details **16**

D Ablations of the Adaptive Memory Gate **22**

E Additional Analysis of the Learned Gate **24**

F Parameters and Training Time **25**

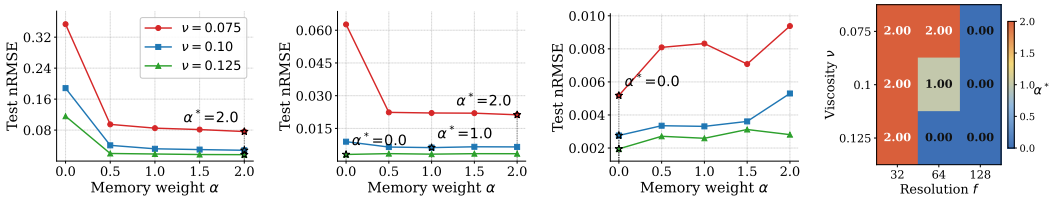
G Extended Related Work **25**

H Future Work **26**

A PRELIMINARY STUDY UNDER ADDITIVE FUSION

Section 2.2 parameterizes the memory weight by a convex blend, which has the same form as the adaptive memory gate. Here we repeat the ablation under an additive parameterization that mirrors the original residual fusion of S4FFNO (Buitrago Ruiz et al., 2025). With the notation of Section 2.2, let $\mathbf{h}_t \in \mathbb{R}^{N\times H}$ be the output of the second FFNO layer and $\mathbf{z}_t = \mathcal{M}(\mathbf{h}_{1:t}^{(\ell^*-1)})$ the output of the S4 memory branch. We replace the convex blend with

$$\mathbf{h}_{t+1} \leftarrow \alpha \cdot \mathbf{z}_t + \mathbf{h}_t, \tag{14}$$



(a) $f = 32$ (b) $f = 64$ (c) $f = 128$ (d) Optimal α^*

Figure 4: The fixed-weight limitation is robust to the fusion rule. Under additive fusion $\mathbf{h} + \alpha\mathbf{z}$, the best tested α is largest at $f = 32$, zero at $f = 128$, and viscosity dependent at $f = 64$. Panels (a–c) show the sweep; (d) summarizes α^*.

which keeps the residual path in full and scales the memory output by α. S4FFNO is recovered exactly at $\alpha = 1$, $\alpha = 0$ is again a pure Markovian model, and $\alpha > 1$ amplifies memory beyond its default contribution. We sweep $\alpha \in \{0.0, 0.5, 1.0, 1.5, 2.0\}$ on the same nine KS settings ($f \in \{32, 64, 128\}$, $\nu \in \{0.075, 0.10, 0.125\}$) (Figure 4).

Observation 1: More memory is beneficial at low resolution. At $f = 32$, $\alpha^* = 2.0$ for all three viscosities, and larger α consistently reduces nRMSE.

Observation 2: Memory is not useful at high resolution. At $f = 128$, $\alpha^* = 0.0$ for all three viscosities, and larger α monotonically degrades performance.

Observation 3: The optimal memory weight depends jointly on resolution and viscosity. At $f = 64$, $\alpha^* = 2.0$ at $\nu = 0.075$, $\alpha^* = 1.0$ at $\nu = 0.10$, and $\alpha^* = 0.0$ at $\nu = 0.125$. Higher viscosity reduces the high-frequency content of the solution and with it the benefit of memory.

Implications. The convex (Section 2.2) and additive parameterizations lead to the same three conclusions: memory helps most at low resolution, hurts at high resolution, and its optimal weight depends jointly on resolution and viscosity. The motivation that no single fixed memory weight fits all observation conditions therefore does not depend on the form of the fusion.

B Supporting Analysis for the Theoretical Motivation

This appendix provides the assumptions and proofs for Section 2.3, together with a predictor-class approximation bound.

B.1 Setting and Assumptions

Fix $d \geq 1$, a physical condition ρ, an initial-data law μ_ρ, an observation interval $\Delta t > 0$, and an integer prediction horizon $T_{\text{pred}} \geq 1$. Let $\mathcal{H} = L^2(\mathbb{T}^d; \mathbb{R}^V)$, with normalized Lebesgue measure on $\mathbb{T}^d = (\mathbb{R}/2\pi\mathbb{Z})^d$, and let $u_0^\rho \sim \mu_\rho$ and $u_{t+1}^\rho = \Phi_{\Delta t}^\rho(u_t^\rho)$. The solution map is measurable on its admissible domain. The index $t \in \{0, \dots, T_{\text{pred}}\}$ corresponds to physical time $t\,\Delta t$. One-step predictions are evaluated for $t = 0, \dots, T_{\text{pred}} - 1$. For the Fourier coefficients $a_k(\phi) \in \mathbb{C}^V$, set $m_f = \lfloor f/2 \rfloor$ and

$$P_f\phi = \sum_{\|k\|_\infty \leq m_f} a_k(\phi) e^{ik\cdot x}, \qquad \|\phi\|_{H^s}^2 = \sum_{k\in\mathbb{Z}^d} (1 + \|k\|_2^2)^s \|a_k(\phi)\|_2^2. \tag{15}$$

Here $s \geq 0$, and $\mathcal{V}_f = P_f\mathcal{H}$ inherits the spatial L^2 norm. The cutoff is rectangular, and f denotes a per-axis resolution scale. This ideal projection is distinct from interpolation or subsampling.

Use the resolved state, history, and target from Section 2.3. For an admissible resolved restart ϕ, define $\mathcal{T}_{f,\rho}(\phi) = P_f \Phi^{\rho}_{\Delta t}(\phi)$. Then

$$Y_t^{f,\rho} = \mathcal{T}_{f,\rho}(u^{\rho}_{t,f}) + \delta_{t,f,\rho}. \tag{16}$$

At each fixed t, expectation is over u_0^{ρ}, and both Bayes predictors have the same time information. We write $\mathbb{E}_{u_0,t}$ for the additional uniform average over $t = 0, \dots, T_{\mathrm{pred}} - 1$. The spectral quantity and the finite-horizon averages are

$$\omega_f(\phi) = \begin{cases} \|Q_f \phi\|^2_{\mathcal{H}} / \|\phi\|^2_{\mathcal{H}}, & \phi \neq 0, \\ 0, & \phi = 0, \end{cases} \tag{17}$$

$$\omega^{\mathrm{pop}}_{f,\rho} = \mathbb{E}_{u_0,t}[\omega_f(u_t^{\rho})], \qquad \mathcal{M}_{f,\rho} = \frac{1}{T_{\mathrm{pred}}} \sum_{t=0}^{T_{\mathrm{pred}}-1} \mathcal{M}_{t,f,\rho}. \tag{18}$$

The finite-reference estimate $\widehat{\omega}_{f,\rho}$ used in the experiments is defined in Appendix C.5.

Assumption 1 (Finite-horizon regularity and admissible restarts). The PDE has a unique solution on the prediction horizon for μ_ρ-almost every initial state. For some $r \geq 0$ and $\beta > 0$, there is a deterministic constant $0 < R_{\rho,\beta} < \infty$, independent of f, such that

$$\max_{t \in \{0,\dots,T_{\mathrm{pred}}\}} \|u_t^{\rho}\|_{H^{r+\beta}} \leq R_{\rho,\beta} \qquad \text{almost surely.} \tag{19}$$

Both u_t^{ρ} and $P_f u_t^{\rho}$ are admissible restarts for $\Phi^{\rho}_{\Delta t}$ at every prediction time and resolution considered.

The uniform trajectory bound in Assumption 1 does not directly cover the untruncated Gaussian initial-data law of the Navier–Stokes dataset in Appendix C.1.

Assumption 2 (Lipschitz continuity on bounded admissible sets). For every $R > 0$, there is $L_{\rho,R} < \infty$ such that

$$\|\Phi^{\rho}_{\Delta t}(\phi) - \Phi^{\rho}_{\Delta t}(\psi)\|_{L^2} \leq L_{\rho,R} \|\phi - \psi\|_{H^r} \tag{20}$$

for admissible ϕ, ψ with H^r norms at most R.

The constants may depend on the fixed dimension, physical condition, time step, horizon, and initial-data law, but are independent of f.

B.2 Residual Bounds

Lemma 2 (Residual bounds). Under Assumptions 1 and 2, let

$$\kappa_{\omega,\rho} = L^2_{\rho,R_{\rho,\beta}} R^2_{\rho,\beta}, \qquad \kappa_{\mathrm{tail},\rho} = 2^{2\beta} \kappa_{\omega,\rho}.$$

Then

$$\mathbb{E}_{u_0} \|\delta_{t,f,\rho}\|^2 \leq \kappa_{\mathrm{tail},\rho} f^{-2\beta}, \tag{21}$$

$$\mathbb{E}_{u_0,t} \|\delta_{t,f,\rho}\|^2 \leq \kappa_{\omega,\rho} \big(\omega^{\mathrm{pop}}_{f,\rho}\big)^{\beta/(r+\beta)}. \tag{22}$$

The first bound is uniform over the fixed finite prediction horizon.

Proof. Since P_f is a contraction in H^r, both restarts have H^r norms bounded by $R_{\rho,\beta}$. Its L^2 contraction property and Assumption 2 give

$$\|\delta_{t,f,\rho}\| \leq L_{\rho,R_{\rho,\beta}} \|Q_f u_t^{\rho}\|_{H^r}. \tag{23}$$

Every excluded index satisfies $\|k\|_2 > f/2$, so

$$\begin{aligned} \|Q_f u_t^{\rho}\|^2_{H^r} &= \sum_{\|k\|_\infty > m_f} (1 + \|k\|_2^2)^r \|a_k(u_t^{\rho})\|_2^2 \\ &\leq 2^{2\beta} f^{-2\beta} \|u_t^{\rho}\|^2_{H^{r+\beta}}. \end{aligned} \tag{24}$$

Squaring Equation 23, applying Equation 24 and Assumption 1, and taking expectation proves Equation 21.

For $r > 0$, Hölder's inequality on the Fourier sums yields

$$\begin{aligned}\|Q_f u_t^\rho\|_{H^r}^2 &\le \big(\|Q_f u_t^\rho\|_{L^2}^2\big)^{\beta/(r+\beta)} \big(\|Q_f u_t^\rho\|_{H^{r+\beta}}^2\big)^{r/(r+\beta)} \\ &\le R_{\rho,\beta}^2 \omega_f(u_t^\rho)^{\beta/(r+\beta)}.\end{aligned} \tag{25}$$

The last inequality uses $\|Q_f u_t^\rho\|_{L^2}^2 = \omega_f(u_t^\rho)\|u_t^\rho\|_{L^2}^2$ and Assumption 1; the zero state is handled by $\omega_f(0) = 0$. For $r = 0$, the same final bound follows directly from this L^2 identity. Squaring Equation 23, applying this bound, and averaging gives

$$\begin{aligned}\mathbb{E}_{u_0,t}\|\delta_{t,f,\rho}\|^2 &\le \kappa_{\omega,\rho}\, \mathbb{E}_{u_0,t}\left[\omega_f(u_t^\rho)^{\beta/(r+\beta)}\right] \\ &\le \kappa_{\omega,\rho}\big(\omega_{f,\rho}^{\mathrm{pop}}\big)^{\beta/(r+\beta)},\end{aligned}$$

where the last inequality is Jensen's inequality for the concave function $x \mapsto x^{\beta/(r+\beta)}$. This proves Equation 22. □

B.3 Proof of the Bayes History-Gain Proposition

Proof of Proposition 1. Fix t. Conditional expectations minimize squared loss. Since the current state is included in the history, L^2 orthogonality gives

$$\begin{aligned}&\mathbb{E}\big\|Y_t^{f,\rho} - \mathbb{E}[Y_t^{f,\rho} \mid u_{t,f}^\rho]\big\|^2 \\ &\quad = \mathbb{E}\big\|Y_t^{f,\rho} - \mathbb{E}[Y_t^{f,\rho} \mid \mathcal{U}_t^{f,\rho}]\big\|^2 + \mathbb{E}\big\|\mathbb{E}[Y_t^{f,\rho} \mid \mathcal{U}_t^{f,\rho}] - \mathbb{E}[Y_t^{f,\rho} \mid u_{t,f}^\rho]\big\|^2.\end{aligned} \tag{26}$$

Subtract the optimal history risk. The current-state-measurable term $\mathcal{T}_{f,\rho}(u_{t,f}^\rho)$ in Equation 16 cancels between the two conditional means, proving Equation 7. A nonnegative square has positive expectation exactly when it is nonzero with positive probability.

By the tower property, the conditional-mean difference in Equation 7 equals

$$\mathbb{E}\left[\delta_{t,f,\rho} - \mathbb{E}[\delta_{t,f,\rho} \mid u_{t,f}^\rho] \,\middle|\, \mathcal{U}_t^{f,\rho}\right].$$

Conditional Jensen and orthogonality therefore give

$$\mathcal{M}_{t,f,\rho} \le \mathbb{E}\big\|\delta_{t,f,\rho} - \mathbb{E}[\delta_{t,f,\rho} \mid u_{t,f}^\rho]\big\|^2 \le \mathbb{E}\|\delta_{t,f,\rho}\|^2. \tag{27}$$

Lemma 2 proves Equation 8. Averaging over t and applying Equation 22 proves Equation 9. □

B.4 Relation to Architecture Gains

Let $\mathcal{C}_{f,\rho}^F$ be a nonempty class of square-integrable current-state predictors. We view each $\psi \in \mathcal{C}_{f,\rho}^F$ as a function of histories of the form

$$\psi(\mathcal{U}_t^{f,\rho}) = \varphi_\psi(u_{t,f}^\rho),$$

where the same map φ_ψ is used at every prediction time t. At each fixed t, define the Bayes history predictor by

$$b_{t,f,\rho}(\mathcal{U}_t^{f,\rho}) := \mathbb{E}[Y_t^{f,\rho} \mid \mathcal{U}_t^{f,\rho}].$$

For a nonempty class $\mathcal{C}$ of square-integrable history predictors, define

$$\mathcal{E}_{f,\rho}^{\mathcal{C}} := \inf_{\psi \in \mathcal{C}} \mathbb{E}_{u_0,t}\big\|b_{t,f,\rho}(\mathcal{U}_t^{f,\rho}) - \psi(\mathcal{U}_t^{f,\rho})\big\|^2, \tag{28}$$

$$\mathcal{A}_{f,\rho}^F := \inf_{\psi \in \mathcal{C}_{f,\rho}^F} \mathbb{E}_{u_0,t}\big\|\mathcal{T}_{f,\rho}(u_{t,f}^\rho) - \psi(\mathcal{U}_t^{f,\rho})\big\|^2. \tag{29}$$

Here $\mathcal{E}_{f,\rho}^F$ denotes Equation 28 with $\mathcal{C} = \mathcal{C}_{f,\rho}^F$. Each infimum is taken over a single predictor shared across the finite prediction horizon.

Remark 3 (Conditional architecture comparison). Under Assumptions 1 and 2,

$$\mathcal{E}^F_{f,\rho} \le 2\mathcal{A}^F_{f,\rho} + 2\kappa_{\mathrm{tail},\rho} f^{-2\beta}. \tag{30}$$

If a memory class $\mathcal{C}$ contains $\mathcal{C}^F_{f,\rho}$, then

$$0 \le \mathcal{E}^F_{f,\rho} - \mathcal{E}^{\mathcal{C}}_{f,\rho} \le \mathcal{E}^F_{f,\rho}.$$

Consequently, if $\mathcal{A}^F_{f,\rho} \to 0$ as $f \to \infty$, the additional approximation gain of $\mathcal{C}$ also vanishes.

To obtain Equation 30, use

$$b_{t,f,\rho}(\mathcal{U}^{f,\rho}_t) = \mathcal{T}_{f,\rho}(u^{\rho}_{t,f}) + \mathbb{E}[\delta_{t,f,\rho} \mid \mathcal{U}^{f,\rho}_t].$$

For any $\psi \in \mathcal{C}^F_{f,\rho}$, the squared triangle inequality and conditional Jensen give

$$\begin{aligned}&\mathbb{E}_{u_0,t}\left\|b_{t,f,\rho}(\mathcal{U}^{f,\rho}_t) - \psi(\mathcal{U}^{f,\rho}_t)\right\|^2 \\ &\quad\le 2\mathbb{E}_{u_0,t}\left\|\mathcal{T}_{f,\rho}(u^{\rho}_{t,f}) - \psi(\mathcal{U}^{f,\rho}_t)\right\|^2 + 2\mathbb{E}_{u_0,t}\|\delta_{t,f,\rho}\|^2.\end{aligned}$$

Taking the infimum and applying Lemma 2 proves Equation 30. The gain inequalities follow from class inclusion and nonnegativity.

C Experimental Details

This appendix gives the details omitted from Section 4.1. Appendix C.1 describes the PDE datasets, Appendix C.2 describes the observation grids, and Appendix C.3 gives the training and evaluation protocols. , Appendix C.4 describes the compared models, and Appendix C.5 describes spectral-ratio estimation. Unless otherwise stated, the KS and Burgers' setups follow S4FFNO (Buitrago Ruiz et al., 2025).

C.1 PDE Datasets

Kuramoto–Sivashinsky equation. The KS equation is

$$u_t + uu_x + u_{xx} + \nu u_{xxxx} = 0, \qquad (t,x) \in [0,T] \times [0,L], \tag{31}$$

with periodic boundary conditions and $L = 64$. Following S4FFNO, we use the low-viscosity datasets $\nu \in \{0.075, 0.10, 0.125\}$, because low viscosity produces solutions with substantial high-frequency Fourier components; the generation pipeline is that of PDE-Refiner/LPSDA (Lippe et al., 2023; Brandstetter et al., 2022). Each trajectory is observed at 26 equispaced time points up to $T = 2.5$, giving 25 prediction steps with $\Delta t = 0.1$. Reference trajectories are generated at spatial resolution 512, and we use 2048 training and 256 held-out trajectories. The initial condition is a superposition of sinusoidal waves,

$$u_0(x) = \sum_{i=0}^{20} A_i \sin\left(\frac{2\pi k_i}{L}x + \phi_i\right), \tag{32}$$

with $A_i \sim \mathrm{Unif}[-0.5, 0.5]$, $k_i \sim \mathrm{Unif}\{1,\dots,8\}$, and $\phi_i \sim \mathrm{Unif}[0, 2\pi]$ drawn independently for each trajectory. The data are generated with the method of lines (Schiesser, 1991): spatial derivatives are computed pseudospectrally with fast Fourier transforms (Cooley et al., 1969), and the resulting ODE system is solved with an implicit Runge–Kutta method of the Radau IIA family of order 5 (Hairer & Wanner, 1996), implemented in SciPy (Virtanen et al., 2020).

Burgers' equation. The viscous Burgers' equation is

$$u_t + uu_x = \nu u_{xx}, \qquad (t,x) \in [0,T] \times [0,L], \tag{33}$$

with periodic boundary conditions. We use the public PDEBench dataset (Takamoto et al., 2022) with $\nu = 0.001$, whose highest spatial resolution is 1024. Following S4FFNO, we use 2048 samples for training and reserve 10% of the full dataset for testing. We evaluate the first 20 prediction steps (up to $T = 1.4$), because after this period the diffusion term attenuates most high-frequency components and the solution changes slowly. Table 4 summarizes the reference grids, training-set sizes, and rollout horizons for KS and Burgers'.

Table 4: Summary of the KS and Burgers' benchmarks. We report the physical parameter, high-resolution reference grid, number of training trajectories, and autoregressive rollout horizon.

Dataset	ν	Reference grid	Training trajectories	Rollout steps
KS	$\{0.075, 0.10, 0.125\}$	512	2048	25
Burgers'	0.001	1024	2048	20

Incompressible Navier–Stokes. We consider the vorticity form on the unit torus,

$$\frac{\partial w(x,t)}{\partial t} + u(x,t)\cdot\nabla w(x,t) = \nu\Delta w(x,t) + f(x), \qquad x\in(0,1)^2, \quad t\in(0,T], \tag{34}$$

$$\nabla\cdot u(x,t) = 0, \qquad x\in(0,1)^2, \quad t\in[0,T], \tag{35}$$

$$w(x,0) = w_0(x), \qquad x\in(0,1)^2, \tag{36}$$

where w is the vorticity, u the velocity field, ν the viscosity, and the forcing is $f(x_1,x_2) = 0.1\big[\sin\big(2\pi(x_1+x_2)\big)+\cos\big(2\pi(x_1+x_2)\big)\big]$. Data generation follows Li et al. (2021). Initial vorticities are sampled from the Gaussian random field $w_0 \sim \mathcal{N}\big(0, 7^{3/2}(-\Delta+49I)^{-2.5}\big)$ on the space of spatially mean-zero periodic functions, with the zero Fourier mode set to zero. At each numerical step, the Poisson equation for the stream function is solved in Fourier space to recover the velocity. Spatial derivatives are computed spectrally, and the nonlinear advection term is evaluated in physical space before the two-thirds dealiasing rule is applied. The viscous term is advanced with a Crank–Nicolson update, advection and forcing are treated explicitly, and the numerical time step is 10^{-4}. We generate trajectories at $\nu\in\{10^{-5},10^{-4},10^{-3}\}$ on a 512×512 grid with $T=3.2$, and record 32 snapshots at intervals of 0.1 ($t=0.1,\dots,3.2$). For each viscosity, we generate 1,280 trajectories and split them into 1,024 training and 256 held-out trajectories.

Compressible CFD. The 2D and 3D datasets are the compressible Navier–Stokes data of PDEBench (Takamoto et al., 2022) at Mach number $M=1.0$ with the turbulent (`Turb`) initial condition, shear viscosity $\eta=10^{-8}$, bulk viscosity $\zeta=10^{-8}$ and periodic boundaries. Each stores 21 snapshots at $\Delta t=0.05$ over $t\in[0,1]$, of which we use the first 20. The 2D data have four fields (density, pressure, and two velocity components) on a native 512^2 grid, with 1,000 trajectories split 80/20 into 800 training and 200 test trajectories. The 3D data have five fields (density, pressure, and three velocity components) on a native 64^3 grid, with 600 trajectories split 80/20 into 480 and 120. Because η and ζ are not the kinematic viscosity ν of the incompressible and 1D settings, the CFD rows of Table 2 are labelled by initial condition rather than by ν. Table 5 summarizes the native reference grids, field counts, train/test splits, and retained frame counts for Navier–Stokes and compressible CFD.

Table 5: Summary of the 2D and 3D datasets. The table reports physical parameters, native reference grids, field counts, train/test splits, and the number of frames used for Navier–Stokes and compressible CFD.

Dataset	Parameters	Reference grid	Fields	Train / test	Frames used
NS (2D)	$\nu\in\{10^{-5},10^{-4},10^{-3}\}$	512^2	1	1024 / 256	32
CFD (2D)	M=1.0, η=ζ=10^{-8}	512^2	4	800 / 200	20
CFD (3D)	M=1.0, η=ζ=10^{-8}	64^3	5	480 / 120	20

C.2 Resolution Construction

Kuramoto–Sivashinsky and Burgers'. For KS and Burgers', all models are trained and tested at $f\in\{32,64,128\}$. For a domain of length L, the equispaced grid at resolution f is

$$S_f := \left\{ x_r = \frac{rL}{f} : r = 0,\dots,f-1 \right\}. \tag{37}$$

Following S4FFNO, the observation at resolution f is obtained by cubic interpolation $\mathcal{I}_f$ of the highest-resolution trajectory onto S_f. For trajectory i and timestep j, the resolved state is

$$u_j^{f,(i)} := \mathcal{I}_f\left[u^{(i)}(t_j, \cdot)\right] \in \mathbb{R}^{f \times V}, \tag{38}$$

where V is the number of physical channels. All models use the same resolved states and the same grid encoding.

Navier–Stokes and compressible CFD. The observation grid S_f is the tensor-product extension of Equation 37 to each spatial axis. For these datasets, every native resolution is an exact multiple of its target, so we observe by uniform subsampling, which coincides with interpolating onto S_f. Navier–Stokes is observed at $f \in \{32, 64\}$ by subsampling the same 512^2 trajectories with strides 16 and 8, and the train/test split is shared across the two resolutions. 2D CFD is observed at $f = 32$ (stride 16) and 3D CFD at $f = 16$ (stride 4). The unresolved-energy statistic $\widehat{\omega}_f$ is computed from the native-grid training trajectories using the procedure in Appendix C.5.

Relation to the theory. Appendix B analyzes an ideal Fourier projection P_f in order to isolate the unresolved spectral-tail mechanism, whereas the experiments construct the low-resolution inputs by cubic interpolation for KS and Burgers' and by uniform subsampling for Navier–Stokes and compressible CFD. The empirical quantity $\widehat{\omega}_{f,\rho}$ is computed from Fourier coefficients of the high-resolution training trajectories, as described in Appendix C.5.

C.3 Training and Evaluation

Training objective and procedure. All models are trained with a one-step prediction objective. Given the current resolved state $u_j^{f,(i)}$ on spatial grid S_f at time t_j, the model predicts the next state $u_{j+1}^{f,(i)}$ at time t_{j+1}. During training, we employ teacher forcing: each step receives the ground-truth state as input rather than the model's own prediction from the previous step. For memory-augmented models, the internal memory state is updated using the same teacher-forced sequence. Using the per-step relative ℓ_2 error (nRMSE), the training loss is

$$\mathcal{L}_{\text{train}}(\theta) = \frac{1}{J} \sum_{i \in \mathcal{B}} \sum_{j=0}^{J-1} \frac{\left\| \mathcal{G}_\theta(u_j^{f,(i)}, g_f) - u_{j+1}^{f,(i)} \right\|_2}{\left\| u_{j+1}^{f,(i)} \right\|_2}, \tag{39}$$

where $\mathcal{G}_\theta$ is the neural operator, g_f is the grid encoding, $\mathcal{B}$ is the mini-batch, and J is the number of prediction steps per training chunk. All models within each benchmark use the same optimizer settings.

Evaluation protocol. At test time, models are evaluated via autoregressive rollout. Only the initial state $u_0^{f,(i)}$ is provided as ground truth; subsequent predictions are generated by repeatedly feeding the model's own output back as input:

$$\widetilde{u}_0^{f,(i)} = u_0^{f,(i)}, \qquad \widetilde{u}_{j+1}^{f,(i)} = \mathcal{G}_\theta(\widetilde{u}_j^{f,(i)}, g_f), \quad j = 0, \ldots, J_{\text{roll}} - 1, \tag{40}$$

where J_{roll} is the number of generated prediction steps. Training and evaluation use the same per-step error functional, but training is teacher-forced on ground-truth states, whereas evaluation rolls out from a single initial frame on the model's own predictions. Performance is measured by the normalized root mean squared error (nRMSE), averaged over all test trajectories and the benchmark-specific evaluation window:

$$\text{nRMSE}(f) = \frac{1}{N_{\text{test}} J_{\text{eval}}} \sum_{i=1}^{N_{\text{test}}} \sum_{j=1}^{J_{\text{eval}}} \frac{\left\| \widetilde{u}_j^{f,(i)} - u_j^{f,(i)} \right\|_2}{\left\| u_j^{f,(i)} \right\|_2}, \tag{41}$$

where N_{test} is the number of test trajectories and J_{eval} is the number of scored prediction steps. The evaluation window may be shorter than the generated rollout; the benchmark-specific horizons are given below. Lower nRMSE indicates more accurate trajectory prediction.

Kuramoto–Sivashinsky and Burgers'. For KS and Burgers', training follows FFNO and S4FFNO (Tran et al., 2023; Buitrago Ruiz et al., 2025), with a cosine-annealed learning rate (Loshchilov & Hutter, 2017). Table 6 lists the hyperparameters. The memory length is the full rollout, 25 steps for KS and 20 for Burgers'. Experiments run on NVIDIA A6000 GPUs; a KS or Burgers' run needs less than 10 GB of GPU memory and 1–2 hours, depending on the resolution.

Table 6: Training protocol for KS and Burgers'. Both use 200-epoch one-step teacher-forced training with cosine annealing; the listed rollout horizon is also the full memory length used by memory-based models.

Hyperparameter	KS	Burgers'
Epochs	200	200
Initial learning rate	10^{-3}	10^{-3}
Scheduler	cosine annealing	cosine annealing
Batch size	32	64
Training samples	2048	2048
Rollout steps	25	20

At test time, all models are rolled out autoregressively from the first observed state, and we report the rollout nRMSE of Equation 41, averaged over test trajectories and rollout steps; all values in Table 2 use this metric.

Navier–Stokes and compressible CFD. Navier–Stokes and compressible CFD models are trained using AdamW with an initial learning rate of 10^{-3} halved every 90 epochs and zero weight decay; Table 7 lists the hyperparameters. Navier–Stokes models are trained on 16-frame chunks: each 32-frame trajectory is divided into two non-overlapping chunks, which yields 15 next-step targets per chunk, and memory states are initialized separately for each chunk. CFD models are trained on the first 10 of the 20 frames. The batch size is 4 in 3D because folding the spatial axes of a 64^3, five-field state into the batch dimension of the memory layer makes larger batches exceed the 48 GB of GPU memory (measured peak 22.9 GB at batch 4). No trajectory-wise normalization is used. The CFD fields are normalized per channel with statistics computed once on the training split, whereas $\widehat{\omega}_f$ is measured on the unnormalized native fields, because Equation 44 sums the channel energies as written. No gradient clipping is used; clipping at 1.0 did not reduce divergence in these experiments.

Table 7: Training and rollout protocol for Navier–Stokes and compressible CFD. All runs use seeds 155–157; the training window specifies the recurrent memory length, while parentheses in the rollout row denote the in-loop evaluation window used for the reported nRMSE.

Hyperparameter	NS (2D)	CFD (2D)	CFD (3D)
Optimizer	AdamW	AdamW	AdamW
Epochs	300	300	300
Initial learning rate	10^{-3}	10^{-3}	10^{-3}
Scheduler	step ($\times 0.5$ / 90 ep.)	step ($\times 0.5$ / 90 ep.)	step ($\times 0.5$ / 90 ep.)
Weight decay	0	0	0
Batch size	32	32	4
Training samples	1024	800	480
Training window (memory length)	16 frames	10 frames	10 frames
Rollout steps (in-loop)	31 (16)	19 (10)	19 (10)

At test time the first frame is given as input and the model is rolled out autoregressively over all remaining steps as a single continuous sequence, with the recurrent state of S4FFNO and AMGFNO carried across the whole rollout rather than reset at the training-chunk boundary. We evaluate the checkpoint after the final epoch and report the rollout nRMSE of Equation 41, averaged over test trajectories and the steps covered by training—the first 16 for Navier–Stokes and the first 10 for CFD—which is what the *in-loop* label refers to throughout; mean and standard deviation are taken over the three seeds. The remaining steps are recorded separately as an extrapolation metric and are not used in any table here.

Mean gate value. For AMGFNO we additionally report the mean gate value of the trained model: the mean of AMGFNO's learned gate tensor $g \in (0,1)^{B\times T\times S\times H}$ over its batch, step, spatial and channel entries, taken on the last test batch after the final training epoch,

$$\bar{g}(f,\rho) = \frac{1}{B\,T\,S\,H} \sum_{b,j,s,h} g_{b,j,s,h}, \tag{42}$$

and averaged over the three seeds in Table 2.

C.4 Compared Models

The baseline implementations and hyperparameters follow S4FFNO whenever possible. All models share the same input and output processing: the encoder maps the concatenation of the PDE state and the normalized grid coordinate $g_r = x_r/L$ to the hidden dimension, and the decoder maps the final hidden representation back to the PDE state.

FFNO. The Factorized Fourier Neural Operator (Tran et al., 2023) builds on the Fourier Neural Operator (Li et al., 2021) and combines a spectral convolution path with a pointwise feed-forward path. The hidden dimension is 128, the expanded hidden dimension of the FFNO MLP is $4 \cdot 128$, and we keep all resolvable Fourier modes, $k_{\max} = \lfloor f/2 \rfloor$.

S4FFNO. S4FFNO instantiates the Memory Neural Operator framework by combining FFNO layers with an S4 memory layer (Buitrago Ruiz et al., 2025; Gu et al., 2022b) in the configuration FFNO−FFNO−S4−FFNO−FFNO. All layers except the memory layer are the same as in FFNO. The S4 layer uses state dimension 64 and the diagonal S4D kernel (Gu et al., 2022a), operates along the temporal history dimension, and is applied independently at each spatial location.

Multi-Input FFNO. Multi-Input FFNO takes the last $K = 4$ timesteps as input to predict the next one, following the multi-step input setting of FNO and S4FFNO (Li et al., 2021; Buitrago Ruiz et al., 2025). The number of layers and the hidden dimensions are the same as in FFNO; only the input lifting layer changes, which receives the concatenation of four states.

FactFormer. FactFormer is a transformer-based neural operator (Li et al., 2023b). We use the one-dimensional adaptation described in S4FFNO, with four linear attention layers over the spatial sequence, hidden dimension 64, four heads, and hidden dimension 128 in each attention layer.

Galerkin Transformer (GKT). Following Cao (2021), GKT uses four linear attention layers over the spatial sequence, concatenates positional information into the attention inputs, and uses two FNO layers as a spectral regressor. The hidden dimension is 32.

U-Net. The U-Net baseline follows the architecture used for PDE surrogate modeling (Gupta & Brandstetter, 2023): four downsampling convolution blocks, a middle block, and four upsampling blocks with skip connections, with channel multipliers $[1, 2, 2, 2]$ and no time embeddings.

AMGFNO. AMGFNO uses the same FFNO backbone and S4 memory branch as S4FFNO and replaces the fixed residual fusion with the adaptive memory gate of Section 3.3.

C.5 Computation of the Spectral Ratio

This subsection gives the finite-reference implementation of Equation 17 in one, two, and three dimensions. Let $\mathcal{I}^{(d)}_{\mathrm{ref}}$ denote the complete signed DFT index set of a d-dimensional reference grid, and let $a^{\mathrm{ref}}_{\mathbf{k}}(u) \in \mathbb{C}^V$ denote the spatial DFT coefficients of a reference frame u; time is not Fourier transformed. For an observation resolution f along each spatial axis, write $m_f = \lfloor f/2 \rfloor$. The unresolved-energy ratio is computed separately for each spatial dimension as follows.

$$\omega_f^{\mathrm{ref}}(u) = \frac{\displaystyle\sum_{\mathbf{k}\in\mathcal{I}_{\mathrm{ref}}^{(d)},\ \|\mathbf{k}\|_\infty > m_f} \|a_{\mathbf{k}}^{\mathrm{ref}}(u)\|_2^2}{\displaystyle\sum_{\mathbf{k}\in\mathcal{I}_{\mathrm{ref}}^{(d)}} \|a_{\mathbf{k}}^{\mathrm{ref}}(u)\|_2^2}, \tag{43}$$

1D: KS and Burgers'.

$$\omega_f^{\mathrm{ref}}(u) = \frac{\displaystyle\sum_{\substack{k\in\mathcal{I}_{\mathrm{ref}}^{(1)}\\ |k|>m_f}} \|a_k^{\mathrm{ref}}(u)\|_2^2}{\displaystyle\sum_{k\in\mathcal{I}_{\mathrm{ref}}^{(1)}} \|a_k^{\mathrm{ref}}(u)\|_2^2}. \tag{44a}$$

2D: Navier–Stokes and CFD.

$$\omega_f^{\mathrm{ref}}(u) = \frac{\displaystyle\sum_{\substack{(k_x,k_y)\in\mathcal{I}_{\mathrm{ref}}^{(2)}\\ |k_x|>m_f \text{ or } |k_y|>m_f}} \|a_{k_x,k_y}^{\mathrm{ref}}(u)\|_2^2}{\displaystyle\sum_{(k_x,k_y)\in\mathcal{I}_{\mathrm{ref}}^{(2)}} \|a_{k_x,k_y}^{\mathrm{ref}}(u)\|_2^2}. \tag{44b}$$

3D: CFD.

$$\omega_f^{\mathrm{ref}}(u) = \frac{\displaystyle\sum_{\substack{(k_x,k_y,k_z)\in\mathcal{I}_{\mathrm{ref}}^{(3)}\\ |k_x|>m_f \text{ or } |k_y|>m_f\\ \text{or } |k_z|>m_f}} \|a_{k_x,k_y,k_z}^{\mathrm{ref}}(u)\|_2^2}{\displaystyle\sum_{(k_x,k_y,k_z)\in\mathcal{I}_{\mathrm{ref}}^{(3)}} \|a_{k_x,k_y,k_z}^{\mathrm{ref}}(u)\|_2^2}. \tag{44c}$$

These expressions are defined for frames with positive total energy. A mode is counted as unresolved when *any* of its components exceeds the cutoff, so the cutoff is rectangular rather than radial, in agreement with P_f in Equation 15; modes exactly at the cutoff are retained, and for unequal per-axis resolutions the threshold on axis j is $\lfloor f_j/2 \rfloor$. The squared coefficient norm sums over all physical channels; for the multi-field CFD datasets it is computed on the unnormalised fields (Appendix C.3).

For each observation condition (f, ρ), the gate uses the training-set average

$$\widehat{\omega}_{f,\rho} := \frac{1}{N_{\mathrm{train}} T_{\mathrm{obs}}} \sum_{i=1}^{N_{\mathrm{train}}} \sum_{j=0}^{T_{\mathrm{obs}}-1} \omega_f^{\mathrm{ref}}\left(u_j^{\mathrm{high},(i)}\right), \tag{45}$$

where $u_j^{\mathrm{high},(i)}$ is the high-resolution training reference state. The ratio is computed for each frame separately and then averaged arithmetically; energies are not pooled across frames before the division. The estimate uses training trajectories only and is held fixed during training and evaluation, so no high-resolution test information is used. It is a finite-reference diagnostic and is distinct from the continuum high-resolution limit in Appendix B.

The frequency-aware gate of Section 3.3 takes $\log(\widehat{\omega}_{f,\rho} + \varepsilon_{\mathrm{num}})$ as input, where $\varepsilon_{\mathrm{num}} > 0$ is a small constant used only for finite-precision stability; the asymptotic theory uses the corresponding zero-tail convention. We initialize its parameters to $w_1 = 2.0$ and $b_\omega = 3.8$ for every dataset, and both are learned during training.

Note on observability. Because $\widehat{\omega}_{f,\rho}$ is computed from high-resolution training references, it is a calibration constant of the observation condition (f, ρ) and not a per-sample test-time feature. For deployment to new conditions without high-resolution training references, this scalar could be replaced by observable proxies, such as condition-level summaries of the resolved high-band spectrum, or by metadata-based gates conditioned on f and the available physical parameters ρ. We leave the systematic study of such proxies to future work.

D Ablations of the Adaptive Memory Gate

The gate is the product of a content factor, which reads the current state, and a frequency-aware factor, which reads the observation's unresolved spectral energy $\widehat{\omega}_f$:

$$g \;=\; \underbrace{\sigma(W_z z_t + W_h h_t + b)}_{\text{content gate}} \;\times\; \underbrace{\sigma(w_1 \log \widehat{\omega}_f + b_\omega)}_{\text{frequency-aware gate}}. \tag{46}$$

This section reports four ablations: removing each factor (§D.1), replacing the input-dependent gate with its mean at inference (§D.2), discarding the recurrent state at every step (§D.3), and computing $\widehat{\omega}_f$ per batch instead of per condition (§D.4). Every ablation keeps all other components and hyperparameters unchanged.

D.1 Removing Each Gate Factor: Content vs. Frequency-Aware

Table 8: Ablation of the two factors of the adaptive memory gate. Rollout nRMSE, mean±std over seeds 155–157. Shading marks the full gate; each row below removes one factor. Removing the content factor is the more damaging of the two on the coarsest grids. While the frequency-aware factor is a weaker effect. †One of three seeds diverges; the value averages the other two.

	Kuramoto–Sivashinsky									Burgers'		
	$f=32$			$f=64$			$f=128$					
Method	ν=.075	ν=.10	ν=.125	ν=.075	ν=.10	ν=.125	ν=.075	ν=.10	ν=.125	f=32	f=64	f=128
AMGFNO (full)	.067$_{\pm.006}$	**.025**$_{\pm.002}$	**.016**$_{\pm.002}$	**.020**$_{\pm.002}$	**.006**$_{\pm.001}$	.004$_{\pm.000}$	**.006**$_{\pm.000}$	**.003**$_{\pm.000}$	**.002**$_{\pm.000}$	**.023**$_{\pm.001}$	.016$_{\pm.001}$	.014$_{\pm.000}$
w/o content gate	.100$_{\pm.007}$	.037$_{\pm.001}$	.020$_{\pm.001}$	.021$_{\pm.001}$	.007$_{\pm.002}$	.003$_{\pm.000}$	.007$_{\pm.001}$	.005$_{\pm.001}$	.003$_{\pm.000}$	.024$_{\pm.001}$	.015$_{\pm.001}$	.012$_{\pm.000}$
w/o frequency-aware	.064$_{\pm.010}$	.026$_{\pm.002}$	.019$_{\pm.006}$	.020$_{\pm.002}$	.006$_{\pm.000}$	.003$_{\pm.000}$	.007$_{\pm.000}$	.004$_{\pm.000}$	.003$_{\pm.001}$	.026$_{\pm.001}$	.015$_{\pm.001}$	.012$_{\pm.000}$

	Navier–Stokes (2D)						CFD (2D)	CFD (3D)
	$f=32$			$f=64$			Turb	Turb
Method	ν=10^{-5}	ν=10^{-4}	ν=10^{-3}	ν=10^{-5}	ν=10^{-4}	ν=10^{-3}	f=32	f=16
AMGFNO (full)	**.020**$_{\pm.001}$	**.010**$_{\pm.001}$	**.006**$_{\pm.002}$	**.008**$_{\pm.001}$	**.009**$_{\pm.003}$	**.008**$_{\pm.002}$	**.433**$_{\pm.003}$	**.464**$_{\pm.004}$
w/o content gate	.029$_{\pm.008}$	.013$_{\pm.004}$	.011$_{\pm.002}$	.008$_{\pm.001}$	.010$_{\pm.000}^{\dagger}$	.012$_{\pm.005}^{\dagger}$	.443$_{\pm.002}$	.464$_{\pm.004}$
w/o frequency-aware	.021$_{\pm.002}$	.011$_{\pm.002}$	.008$_{\pm.001}$	.012$_{\pm.004}$	.012$_{\pm.002}$	.012$_{\pm.003}$	.435$_{\pm.000}$	.467$_{\pm.001}$

The content factor carries most of the gate's benefit, and the frequency-aware factor is a smaller, setting-dependent correction on top of it. Table 8 removes each factor from the full gate with everything else fixed. The full gate is best or tied in 16 of the 20 settings, and where it loses, the margin is at most .003 nRMSE.

Removing the content factor is the more damaging ablation, and the damage concentrates on the coarsest grids. At KS f=32 the nRMSE rises from .067 to .100 (ν=.075), .025 to .037 (ν=.10) and .016 to .020 (ν=.125); at NS f=32 from .020 to .029, .010 to .013 and .006 to .011 across the three viscosities; on 2D CFD from .433 to .443. On KS at f=64 and f=128 the gap is at most .002. This is the pattern expected if the content factor does what the gate was built for: on a coarse grid the current input is least reliable, and a gate that cannot read the input cannot decide when to lean on memory instead.

Removing the frequency-aware factor is a weaker and mixed effect. Its largest costs are at NS f=64 (.008→.012, .009→.012, .008→.013), KS ν=.125, f=32 (.016→.019) and Burgers' f=32 (.023→.026); on KS at f=64 and f=128 the two variants differ by at most .001, and at KS ν=.075, f=32 and Burgers' f=64, 128 the model without it is marginally better. We therefore treat the frequency prior as a second-order term; the input-dependent content factor is the component that should not be removed.

D.2 Replacing the Input-Dependent Gate with Its Mean Value

This control tests whether the input dependence of the gate matters, or whether only its average size does. The learned gate is a tensor whose entries differ across inputs, channels, and spatial positions. We take a trained AMGFNO and change only one thing at inference, without retraining: we replace

Table 9: Replacing the input-dependent gate with its scalar mean at inference. We report rollout nRMSE (↓) with the learned input-dependent gate (input-dep.) and with the gate replaced by its mean value $\bar{g}$ (mean fixed), without retraining; change is the relative increase. Single seed (155).

Setting	Variant	input-dep.	mean fixed	change	$\bar{g}$
NS (2D) ν=10^{-5}, f=32	AMGFNO	0.0209	0.0518	+148%	0.001
CFD (2D)	AMGFNO	0.4369	0.7254	+66%	0.091
CFD (3D)	AMGFNO	0.4615	3.9393	+754%	0.191
CFD (3D)	w/o frequency-aware	0.4658	0.9048	+94%	0.711

the gate tensor by its scalar mean $\bar{g}$ (Equation 42), computed for the same setting and seed. Every input, channel, and position then receives the same gate value. The memory branch, all weights, and the average amount of memory admitted stay unchanged, so any change in error comes from removing the input dependence. We run this control on the 2D and 3D settings (Table 9).

Replacing the gate by its mean raises nRMSE by 66–754% where the gate is open, so what matters is how the gate varies across inputs, not only its average size. This increase overstates the contribution of the variation, because the substituted model sees a gate distribution it was not trained on. It also shows that $\bar{g}$ is not a measure of memory usage: on 3D CFD, $\bar{g}$=0.19, yet fixing the gate at that value raises the error by 754%. We therefore report $\bar{g}$ only as a summary of the gate. This control leaves the memory branch intact; the complementary control, which removes the history while leaving the gate intact, is reported next.

D.3 Discarding the Recurrent State at Every Step

Table 10: Discarding the recurrent state at inference. The recurrent state is reset before every rollout step, without retraining. We report rollout nRMSE (↓) with the history kept (default) and with the state discarded; change is the relative increase, and $\bar{g}$ is the mean gate value. Single seed (155).

Setting	Variant	history kept	state discarded	change	$\bar{g}$
NS (2D) ν=10^{-5}, f=32	AMGFNO	0.0209	0.0280	+34%	0.001
CFD (2D)	AMGFNO	0.4369	0.4678	+7%	0.091
CFD (3D)	AMGFNO	0.4615	0.7584	+64%	0.191
CFD (3D)	w/o frequency-aware	0.4658	0.7749	+66%	0.711

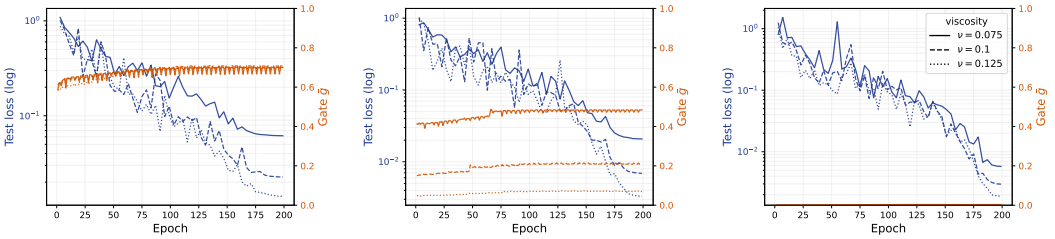
A non-zero gate does not by itself show that past frames are used, since the memory layer also reads the current frame. We therefore reset the recurrent state of the memory layer before every rollout step of a trained model, at inference and without retraining. Table 10 shows that resetting the recurrent state increases nRMSE by 7–66% in the reported NS and CFD runs, demonstrating the benefit of retaining recurrent history. Compared with Table 9, freezing the gate at its mean costs more than discarding the history (+148% versus +34% on NS; +66% versus +7% on 2D CFD).

D.4 Computing $\widehat{\omega}_f$ per Condition vs. per Batch

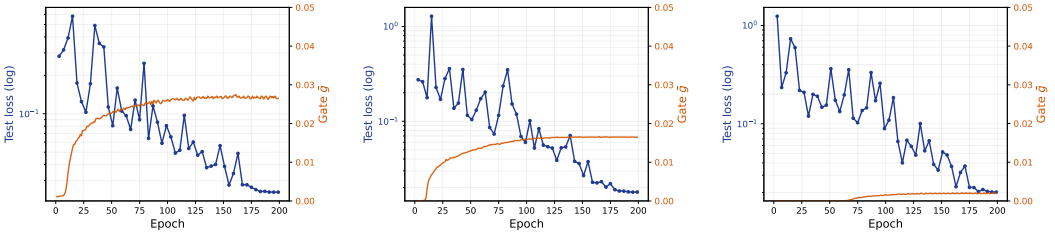
AMGFNO computes $\widehat{\omega}_f$ once per observation condition on the training set and holds it fixed. An alternative is to recompute it from each batch at run time, which would let the gate follow the spectral content of the current input. Table 11 compares the two on the nine KS settings with all other components unchanged. Per-batch recomputation is worse in every setting, by 5.8–20.1%. One plausible reason is that the per-batch estimate is noisier than the condition-level statistic while the observation condition does not change within a trajectory. We therefore use $\widehat{\omega}_f$ as a condition-level calibration constant rather than a run-time signal.

Table 11: Computing $\widehat{\omega}_f$ once per observation condition versus recomputing it per batch. Per-batch recomputation gives 5.8–20.1% higher nRMSE on all nine KS settings. Single seed (155).

f	ν	per condition	per batch	Δ%
32	0.075	0.0616	0.0651	+5.8
32	0.10	0.0226	0.0249	+10.1
32	0.125	0.0142	0.0168	+18.2
64	0.075	0.0208	0.0250	+20.1
64	0.10	0.0069	0.0077	+11.9
64	0.125	0.0033	0.0039	+17.2
128	0.075	0.0058	0.0068	+18.5
128	0.10	0.0030	0.0033	+11.1
128	0.125	0.0019	0.0020	+5.8



(a) KS, $f = 32$ (b) KS, $f = 64$ (c) KS, $f = 128$



(d) Burgers', $f = 32$ (e) Burgers', $f = 64$ (f) Burgers', $f = 128$

Figure 5: The gate typically stabilizes before the final error level is reached. Blue curves show test nRMSE (log scale) and orange curves show $\bar{g}$ during training for KS (a–c) and Burgers' (d–f).

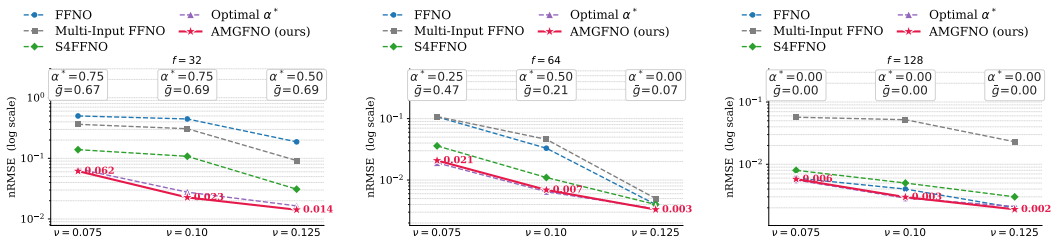
E Additional Analysis of the Learned Gate

E.1 Gate Value and Test Loss during Training

Figure 5 complements Figure 3 by showing the test loss together with $\bar{g}$ over the 200 training epochs. The test loss decreases monotonically while the gate settles at a stable level, and the gate typically converges before the loss does. Table 2 reports the final-epoch mean gate values averaged over the three seeds.

E.2 Comparison with the Oracle Memory Weight

Figure 6 compares AMGFNO with the oracle of Section 4.3: S4FFNO trained at the memory weight α^* that is selected separately for each (f, ν) setting by the five-point sweep of Section 2.2, i.e., the best fixed memory weight on that grid. AMGFNO reaches nRMSE comparable to this oracle in all nine settings while training a single model per setting, and it does so with the mean of AMGFNO's learned gate $\bar{g}$ of Table 2: large at $f = 32$, graded by viscosity at $f = 64$, and close to zero at $f = 128$. This behavior is consistent with the Mori–Zwanzig picture, in which memory matters when unobserved high-frequency modes are prevalent and the Markovian model suffices otherwise (Gouasmi et al., 2017; Lin & Lu, 2021).



(a) KS $f = 32$ (b) KS $f = 64$ (c) KS $f = 128$

Figure 6: AMGFNO tracks the best fixed memory weight without an α sweep. Across nine KS settings, its nRMSE is comparable to S4FFNO using the best tested fixed weight α^* selected separately for each (f, ν). Panels show $f = 32, 64, 128$; annotations report α^* and the learned $\bar{g}$.

F Parameters and Training Time

The gate adds negligible parameter and training overhead. Table 12 reports the number of parameters and the training time per iteration of every model in Sections 4.2 and 4.4. For each of the five memory layers, the gate adds 0.03–0.04M parameters (under 1% of the model) and 2.6–2.8 ms per iteration.

Table 12: Adaptive gating adds negligible parameter and training overhead. Across the five memory architectures, the gate adds 0.03–0.04M parameters and 2.6–2.8 ms per training iteration. Δ is the increase over the same architecture without the gate (S4FFNO for AMGFNO). Timings use KS at $f = 64$ on a single NVIDIA RTX A6000.

Architecture	Params (M)	Δ Params (M)	Time (ms)	Δ Time (ms)
GKT	0.28	–	22.0	–
U-Net	2.68	–	23.3	–
FactFormer (1D)	1.45	–	112.3	–
FFNO (4 layers)	4.89	–	31.5	–
FFNO (5 layers)	6.10	–	38.9	–
S4FFNO	4.94	–	39.5	–
AMGFNO w/o freq. gate	4.98	+0.04	41.7	+2.2
AMGFNO	4.98	+0.04	42.3	+2.8
S5-FFNO	4.90	–	38.8	–
+ gate	4.94	+0.04	41.4	+2.6
LRU-FFNO	4.92	–	42.1	–
+ gate	4.95	+0.03	44.9	+2.8
LSTM-FFNO	5.03	–	38.2	–
+ gate	5.07	+0.04	40.8	+2.6
T-FFNO	5.25	–	60.5	–
+ gate	5.28	+0.03	63.1	+2.6

Is the gain explained by capacity? The gain of AMGFNO does not come from added capacity. Table 13 compares AMGFNO on Burgers' with FFNO baselines of four and five layers. FFNO with five layers has 0.60M, 1.12M, and 2.17M more parameters than AMGFNO at f=32, 64, and 128, yet it does not close the gap at any resolution (0.126, 0.076, 0.024 versus 0.024, 0.018, 0.020).

G Extended Related Work

Neural operators for PDEs. Beyond FNO (Li et al., 2021) and FFNO (Tran et al., 2023), other Markovian operators explore alternative inductive biases: multiscale U-Net features (Gupta & Brandstetter, 2023), Galerkin Transformer's linear attention (Cao, 2021), and self-, cross-, and axial attention in OFormer (Li et al., 2023a) and FactFormer (Li et al., 2023b). These models predict the next state from the current state alone. Current-state-only prediction can be inadequate when observations do not fully resolve the underlying dynamics.

Table 13: The gain of AMGFNO is not explained by increased model capacity. Across Burgers' resolutions, increasing FFNO depth and parameter count does not close the accuracy gap to AMGFNO. The four-layer row is the AMGFNO architecture with the memory weight fixed at α=0, which bypasses the memory branch and leaves a purely Markovian model of the same width; the five-layer row adds one FFNO layer instead. Single seed (155).

	nRMSE			Params (M)		
Model	$f = 32$	$f = 64$	$f = 128$	$f = 32$	$f = 64$	$f = 128$
FFNO (4 layers)	0.125	0.084	0.029	2.85	4.94	9.14
FFNO (5 layers)	0.126	0.076	0.024	3.48	6.10	11.34
AMGFNO (ours)	**0.024**	**0.018**	**0.020**	2.88	4.98	9.17

Memory in PDE models. Multi-Input FFNO (Buitrago Ruiz et al., 2025) concatenates a short window of past states as input channels, which provides only short-range memory. S4FFNO (Buitrago Ruiz et al., 2025) inserts an S4 layer (Gu et al., 2022b) between FFNO layers to encode temporal dependencies and adds the result back through a residual connection. The State Space Neural Operator (SS-NO) (Koren & Lanthaler, 2026) combines bidirectional spatial state-space layers with a temporal memory module, extending state-space modeling to both space and time. In Multi-Input FFNO and S4FFNO, the memory weight is fixed regardless of the input. Our preliminary study (Section 2.2) shows that the optimal weight for fusing memory with current features varies across observation conditions.

Gating mechanisms. Input-dependent control of information retention and transmission is central to LSTM (Hochreiter & Schmidhuber, 1997), GRU (Cho et al., 2014), Highway Networks (Srivastava et al., 2015), and Mamba (Gu & Dao, 2024). In neural operators, TF-SNO (Zhou et al., 2026) uses current-state statistics to modulate spectral responses. S4FFNO fuses memory by a fixed, input-independent residual addition (Buitrago Ruiz et al., 2025).

Position of our work. AMGFNO keeps the temporal-memory architecture of S4FFNO and replaces its fixed fusion with an adaptive memory gate. Unlike prior memory-augmented neural operators, AMGFNO modulates memory weight through a learnable gate. The content gate conditions on the current features and the memory output, and the frequency-aware gate incorporates a condition-level spectral prior.

H Future Work

Frequency prior without high-resolution references. $\widehat{\omega}_f$ is computed once from high-resolution training trajectories (Appendix C.5) and is not used at test time. Using training-time references in this way is comparable to how ground-truth targets are used in training. In practice, however, high-resolution data may not always be available for a new observation condition. In such cases, $\widehat{\omega}_f$ could be replaced by a summary of the resolved high-band spectrum, or by a gate conditioned on f and the physical parameters ρ. We leave these options to future work.

Theory for trained models. Our bound concerns an ideal predictor. Extending it to trained models would help explain why fixed memory hurts on KS at f=128 and why memory remains useful on Burgers', where $\widehat{\omega}_f$ is small.

Cheaper memory layers. A causal Transformer memory gives the lowest error at f=32 (Section 4.4) but costs $1.5\times$ the training time of S4. Pairing the gate with cheaper long-range memory layers is a natural next step.