SINO: SCALE-INVARIANT NEURAL OPERATOR

Anonymous authors
Paper under double-blind review

ABSTRACT

In scientific machine learning, physical fields governed by partial differential equations exhibit low-rank structure and scale invariance. When solving equations on coarse grids, missing information leads to the closure problem—modeling unresolved physics to recover lost dynamics. Although closure terms depend on grid resolution, they represent scale-invariant physical laws. A model truly learning physics should capture these mechanisms with low-rank parameterization rather than memorizing grid-specific patterns. Inspired by this, we propose the Scale-Invariant Neural Operator (SINO), which learns on normalized physical scales via a dual-branch architecture operating in spectral and spatial domains. SINO uses bottleneck MLPs to generate continuous convolution kernels, embedding explicit low-rank inductive bias that concentrates $>95\%$ variance in 2-3 modes, as validated by PCA across benchmarks, while drastically reducing parameters. This principled design yields $38\times$ steeper scaling law exponents than FNO, demonstrating superior parameter efficiency. We compare SINO with traditional models (U-Net, DeepONet), Transformer models (Transolver, Oformer, GK-Transformer), and frequency-domain models (FNO, AMFNO, UFNO) on closure problems spanning externally forced Burgers turbulence, decaying Burgers turbulence, KS turbulence, Kolmogorov-forced NS turbulence, and decaying NS turbulence. Experiments show SINO achieves 1.5–$38\times$ error reduction and 2–$23\times$ parameter efficiency over baselines, with superior scaling laws reflecting exceptional data efficiency from principled low-rank design. Code is available at `https://anonymous.4open.science/r/SINO-662D`.

1 INTRODUCTION

Scientific machine learning aims to understand and simulate complex systems governed by physical laws using data-driven methods (Karniadakis et al., 2021). Unlike natural images, physical fields such as velocity, pressure, and vorticity evolve under partial differential equations and conservation laws. Although discretized fields may possess extremely high dimensionality, their dynamics are organized by a small number of mechanisms: conservation, transport, dissipation, and cross-scale energy transfer (Holmes, 2012; Berkooz et al., 1993; Lumley, 1967). This reveals that physical laws possess an effective **low-rank structure**, where a few degrees of freedom govern the overall dynamics (Thibeault et al., 2024).

This low-rank nature imposes two requirements on model design: (1) capturing governing mechanisms with compact parameterization; (2) learning **resolution-independent** physical laws, where changing grid resolution alters the discrete representation but not the underlying equations. The same continuous process, when observed at different resolutions, is still governed by the same mechanisms.

The **closure problem** provides an ideal testbed for these principles (Sanderse et al., 2024). High-fidelity direct numerical simulation (DNS) resolves small-scale structures but is computationally expensive; practical simulations use coarser grids. The coarse-grid state $\bar{u}_\Delta = \mathcal{P}_\Delta u$ requires a **closure term** τ (Leonard, 1975) to compensate for unresolved scales. The key insight is: although τ depends on the coarse-graining scale Δ, the underlying physical mechanisms are governed by **scale-invariant laws** (Meneveau & Katz, 2000; Kolmogorov, 1991). These processes retain the same mathematical form across resolutions, differing only in operational scales. Therefore, models that truly learn physics should capture the low-rank, scale-invariant physical laws rather than memorizing resolution-specific patterns.

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.

Learning-based closure methods fall into two categories: **learned correction** (Um et al., 2020; List et al., 2022) directly modifies coarse-grid states or evolution results, while **learned interpolation** (Bar-Sinai et al., 2019; Kochkov et al., 2021) learns missing discretization schemes. Recent methods like the Indirect Neural Corrector (INC) (Wei et al., 2026), a learned correction approach, incorporate neural predictions as right-hand side terms in governing equations, improving long-term stability. However, existing methods focus on *how* to couple predictions into solvers, not *what* neural architecture should learn closure. The key challenge is: **how to represent closure operators that are simultaneously low-rank, scale-invariant, and capable of balancing global spectral structure with local spatial features?** Standard approaches fail this requirement: CNNs (Krizhevsky et al., 2012) use resolution-dependent discrete kernels with $O(K^2D^2)$ parameters per layer; FNO (Li et al., 2020) employs mode-specific Fourier weights with $O(k_{\max}D^2)$ parameters that scale with truncation; Transformers rely on high-dimensional attention without explicit low-rank constraints.

To address this challenge, we propose the **Scale-Invariant Neural Operator (SINO)** for learning closure terms that capture cross-resolution, scale-invariant physical laws. SINO employs a dual-branch architecture: a frequency branch captures global spectral energy transfer, while a spatial branch handles localized nonlinear structures. Both branches generate continuous convolution kernels via hypernetworks with bottleneck layers—taking normalized physical coordinates as input rather than discrete grid indices. This design enforces three principles: (1) **Scale invariance**: normalized coordinates ensure identical representations across resolutions; (2) **Low-rank inductive bias**: bottleneck layers with dimensions $n_f, n_s \ll D^2$ force kernels into low-dimensional subspaces, reducing parameters from $O(k_{\max}D^2)$ (FNO) or $O(K^2D^2)$ (CNN) to $O((n_f+n_s)D^2)$ independent of resolution; (3) **Global-local balance**: dual branches synergistically combine frequency-domain long-range dependencies with spatial-domain localized patterns.

We validate SINO on six turbulent closure benchmarks spanning forced and decaying Burgers, Kuramoto-Sivashinsky, and Navier-Stokes equations at Reynolds numbers 1000-4000, comparing against eight baselines. SINO achieves 1.5-38$\times$ error reduction with 2-23$\times$ fewer parameters than FNO-based methods. Theoretical analysis proves SINO's bottleneck architecture enforces hard rank constraints (Theorem 1) and Lipschitz continuity (Theorem 2), with operator norms growing sublinearly ($O(\sqrt{n_f})$) compared to FNO's linear growth. Empirical PCA (Abdi & Williams, 2010) reveals learned kernels concentrate $>$95% variance in 2-3 principal components, validating extreme compression. Scaling law analysis shows SINO's power-law exponents are 3-38$\times$ steeper than baselines, demonstrating superior parameter efficiency. Our contributions are:

- **Architecture**: We propose SINO with bottleneck hypernetworks generating continuous kernels from normalized coordinates, embedding low-rank and scale-invariant inductive biases. Dual branches balance global frequency-domain and local spatial-domain representations.
- **Empirical validation**: SINO achieves best average performance on six turbulent closure problems (forced/decaying Burgers, KS, forced/decaying NS), with 1.5-38$\times$ error reduction and 2-23$\times$ parameter efficiency over eight baselines (U-Net (Ronneberger et al., 2015), DeepONet (Lu et al., 2021), Transolver (Wu et al., 2024), OFormer (Li et al., 2022), GK-Transformer (Cao, 2021), FNO (Krizhevsky et al., 2012), AM-FNO (Xiao et al., 2024), UFNO (Wen et al., 2022)). Ablation studies confirm necessity of dual-branch fusion.
- **Theoretical foundations**: We prove bottleneck layers enforce rank-n_f and rank-n_s constraints (Theorem 1), Lipschitz continuity (Theorem 2), and sublinear operator norm growth (Theorems 4–5). PCA analysis shows $>$95% variance in 2-3 components. Scaling laws ($\alpha = 0.568$ on NS vs. 0.015 for FNO) demonstrate SINO prioritizes learning physical laws over memorizing patterns during capacity growth.

2 Related Work

Closure Modeling. Resolving the finest spatiotemporal scales in PDEs is often computationally intractable, necessitating coarse-grained approximations. Traditional approaches like Reynolds-averaged Navier-Stokes (RANS), large-eddy simulation (LES) (Heinz, 2020), and subgrid-scale (SGS) models (Shankar et al., 2023) model unresolved physics through closure terms, but deriving reliable closures remains challenging with limited accuracy for complex flows (Wei et al., 2026). Recent machine learning approaches systematically learn closure operators from data. The indirect

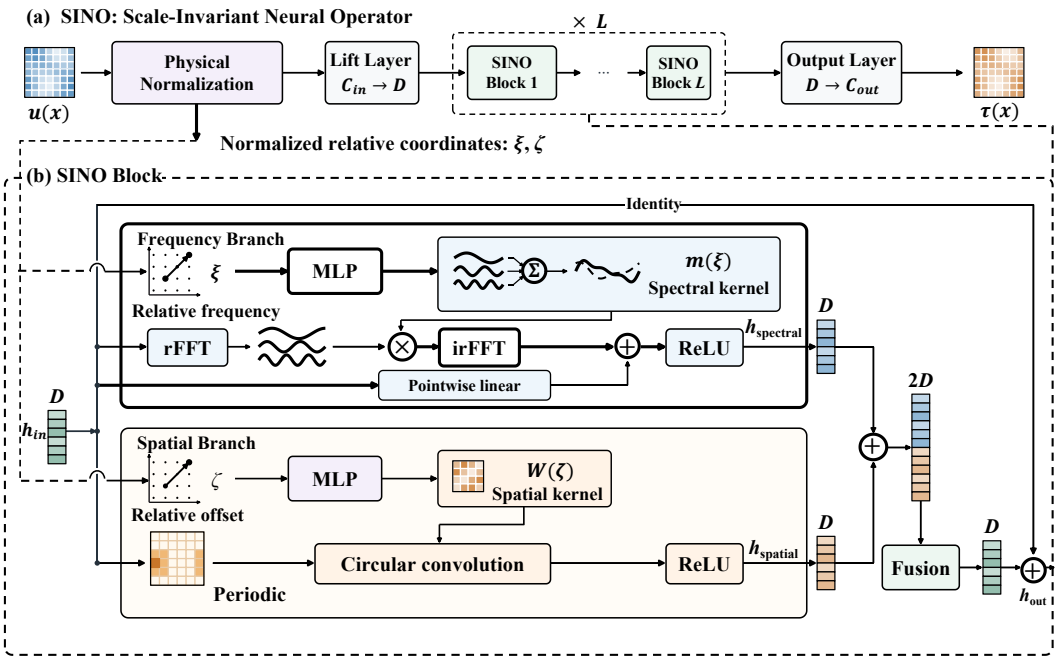


Figure 1: Architecture of the SINO. **(a)** Overall pipeline with physical normalization, lifting layer, L SINO blocks, and output projection. **(b)** SINO block: dual-branch architecture where frequency and spatial branches generate continuous kernels via bottleneck MLPs from normalized coordinates (ξ, ζ), then fuse outputs through residual connections.

neural corrector (INC) (Wei et al., 2026) embeds learned terms into the PDE's right-hand side for improved stability. We extend this framework with low-rank, scale-invariant architectures. Given coarse-grid state $\bar{u}_\Delta$, the learned closure predicts the correction term $\tau_\Delta = \text{SINO}(\bar{u}_\Delta; \Theta)$ that augments the coarse-grid dynamics: $\partial \bar{u}_\Delta / \partial t = \mathcal{L}_\Delta(\bar{u}_\Delta) + \tau_\Delta$, enabling stable rollout integration.

Implicit Parameterization. Implicit parameterization represents neural network weights or convolutional kernels as continuous functions of coordinates, decoupling learnable parameters from kernel size or resolution. Hypernetworks generate weights via an auxiliary network, enabling adaptive filter generation (Ha et al., 2016; Ma et al., 2022). Continuous Kernel Convolution (CKConv) (Romero et al., 2021) models kernels as continuous functions, handling arbitrarily long sequences and irregular data. Neural Implicit Frequency Filters (NIFF) (Grabinski et al., 2024) learn filters in the frequency domain, enabling infinitely large kernels. We extend this paradigm by employing bottleneck hypernetworks to generate continuous kernels in both spatial and spectral domains, enforcing low-rank constraints aligned with turbulent closure physics while achieving cross-resolution scale invariance through normalized coordinates.

3 Method

We propose the Scale-Invariant Neural Operator (SINO), a dual-branch architecture that learns closure terms on normalized physical scales. SINO parameterizes kernel weights as continuous functions of normalized coordinates in both frequency and spatial domains, decoupling learnable parameters from kernel size and resolution. This enables compact representations of scale-invariant physical laws while balancing global spectral structure with local spatial features.

3.1 Physical Normalization of Input Coordinates

Learning scale-invariant laws requires that the same physical process, when observed at different resolutions, receives identical normalized representations. For a coarse-grid field $\bar{u}_\Delta$ with resolution

N obtained by coarse-graining from fine-grid DNS at resolution N_{fine}, the resample factor $r = N_{\text{fine}}/N$ quantifies the degree of coarse-graining.

Frequency domain. We normalize wavenumber coordinates as $\xi = 4k/N - 1 \in [-1, 1]$, where $k \in [0, N/2]$ is the frequency index from the real FFT. This maps the DC component ($k = 0$) to $\xi = -1$, mid-range frequencies to $\xi = 0$, and the Nyquist frequency to $\xi = 1$, ensuring the same physical frequency receives the same ξ regardless of resolution N.

Spatial domain. For a kernel of size $K = 2R + 1$ (radius R) at resolution N, we normalize the spatial offset $\Delta x = j/N$ (where $j \in [-R, R]$) by the reference length $L_{\text{ref}} = R/(N_{\text{fine}}/r)$, yielding $\zeta = j/R \in [-1, 1]$. This guarantees identical physical displacements receive identical ζ across resolutions, enabling resolution-independent convolution kernels.

3.2 Dual-Branch Architecture

Physical fields in turbulent systems contain both global coherent structures and localized intermittent events. Frequency representations naturally capture smooth long-range correlations and spectral energy transfer but struggle with sharp local features (shocks, vortex cores) due to spectral ringing. Conversely, spatial convolutions excel at localized nonlinear patterns but incur high cost for long-range dependencies. SINO addresses this complementary trade-off via a dual-branch architecture processing features in parallel through frequency and spatial domains.

For input $\bar{u}_\Delta \in \mathbb{R}^{B\times N\times C_{\text{in}}}$ (batch size B, channels C_{in}), a lifting layer projects to latent space $h^{(0)} = P(\bar{u}_\Delta) \in \mathbb{R}^{B\times N\times D}$ (hidden dimension D). Each SINO block ($\ell \in \{1, \dots, L\}$) processes $h^{(\ell-1)}$ through two branches:

Frequency branch (spectral convolution with residual bypass):

$$h^{(\ell)}_{\text{freq}} = \sigma\big(W^{(\ell)}_{\text{bypass}} h^{(\ell-1)} + \mathcal{F}^{-1}\big[m^{(\ell)}(\xi) \odot \mathcal{F}(h^{(\ell-1)})\big]\big), \tag{1}$$

where σ is ReLU, $W^{(\ell)}_{\text{bypass}} \in \mathbb{R}^{D\times D}$ is the bypass projection, $\mathcal{F}$ denotes FFT, $m^{(\ell)}(\xi) \in \mathbb{C}^{D\times D}$ are hypernetwork-generated multiplication weights, and $\odot$ is element-wise multiplication.

Spatial branch (implicit circular convolution):

$$h^{(\ell)}_{\text{spatial}} = \sigma\big(\text{Conv}^{(\ell)}(h^{(\ell-1)}; W^{(\ell)}(\zeta))\big), \tag{2}$$

where $W^{(\ell)}(\zeta)$ are hypernetwork-generated kernel weights parameterized by ζ.

Fusion. Outputs are concatenated and fused:

$$h^{(\ell)} = h^{(\ell-1)} + W^{(\ell)}_{\text{fuse}}[h^{(\ell)}_{\text{freq}}\,;\, h^{(\ell)}_{\text{spatial}}], \tag{3}$$

where $[\cdot\,;\,\cdot]$ denotes concatenation and $W^{(\ell)}_{\text{fuse}} \in \mathbb{R}^{D\times 2D}$ is the fusion matrix. Activations are applied in intermediate blocks but omitted in the final block for unrestricted output range.

After L blocks, an output projection with small initialization ($\sigma_{\text{init}} = 0.01$) yields:

$$\tau_\Delta = Q(h^{(L)}) \in \mathbb{R}^{B\times N\times C_{\text{out}}}. \tag{4}$$

The predicted closure term τ_Δ is then incorporated into the coarse-grid evolution $\partial\bar{u}_\Delta/\partial t = \mathcal{L}_\Delta(\bar{u}_\Delta) + \tau_\Delta$ for rollout integration.

3.3 Implicit Kernel Representation via Hypernetworks

We represent convolution kernels as continuous functions generated by small MLPs (hypernetworks), decoupling expressiveness from parameter count.

Frequency domain. Hypernetwork Φ_ω generates complex-valued weights $m(\xi) \in \mathbb{C}^{D\times D}$ via three hidden layers $[w_f, w_f, n_f]$ (intermediate width w_f, bottleneck n_f) with separate real/imaginary output heads:

$$m(\xi) = \Phi_\omega(\xi) = \text{head}_{\text{real}}(\text{MLP}(\xi)) + i \cdot \text{head}_{\text{imag}}(\text{MLP}(\xi)). \tag{5}$$

Multiplication applies via the convolution theorem: $\mathcal{F}(\text{Conv}(h)) = m(\xi) \odot \mathcal{F}(h)$, enabling $O(N \log N)$ computation via FFT.

Spatial domain. Hypernetwork Φ_x generates kernel weights $W(\zeta) \in \mathbb{R}^{K\times K\times D\times D}$ via three layers $[w_s, w_s, n_s]$ (spatial width w_s, bottleneck n_s):

$$W(\zeta) = \Phi_x(\zeta), \quad (\text{Conv}(h))_i = \sum_{j\in\mathcal{N}(i)} W(\zeta_{i-j}) \cdot h_j, \tag{6}$$

where $\mathcal{N}(i)$ is the neighborhood around position i with circular boundary conditions. The implicit representation induces spatial smoothness, improving robustness.The complete model is:

$$\tau_\Delta = \text{SINO}(\bar{u}_\Delta; \Theta) = Q \circ \text{Block}^{(L)} \circ \cdots \circ \text{Block}^{(1)} \circ P(\bar{u}_\Delta), \tag{7}$$

where Θ collects all parameters. Hyperparameters $(D, L, K, n_f, n_s, w_f, w_s)$ control capacity and inductive biases.

3.4 Theoretical Properties

SINO's implicit kernel representation induces low-rank structure through bottleneck layers. To validate this empirically, we perform Principal Component Analysis (PCA) on the learned convolution kernels across all channel pairs in each layer. Figure 2 reveals a striking concentration of variance: the first 2 principal components alone capture over 90% of the variance in both spatial and frequency branches, while merely 3 components exceed 95% explained variance across all layers. This demonstrates that SINO successfully concentrates learned representations into an extremely low-dimensional subspace, aligned with the physical intuition of scale separation and energy cascade in turbulent flows. We rigorously establish in Appendix A.4 that the bottleneck architecture enforces explicit rank constraints (Theorem 1), where every generated kernel admits a finite-rank decomposition $m(\xi) = \sum_{j=1}^{n_f} z_j(\xi) \cdot B_j$ with fixed basis matrices, reducing effective parameter count from $O(k_{\max} \cdot D^2)$ in FNO or $O(K^2 \cdot D^2)$ in CNN to $O((n_f + n_s) \cdot D^2)$ in SINO, independent of resolution or kernel size. Furthermore, the MLP parameterization guarantees Lipschitz continuity (Theorem 2) that prevents memorization of discrete grid patterns, and we establish quantitative operator norm bounds: $O(\sqrt{n_f})$ sublinear growth for the frequency-domain operator and $O(n_s)$ linear growth for the spatial-domain operator (Theorems 4–5), providing favorable implicit regularization through spectral orthogonality in the frequency branch. These properties jointly explain SINO's superior parameter efficiency and data efficiency observed in experiments.

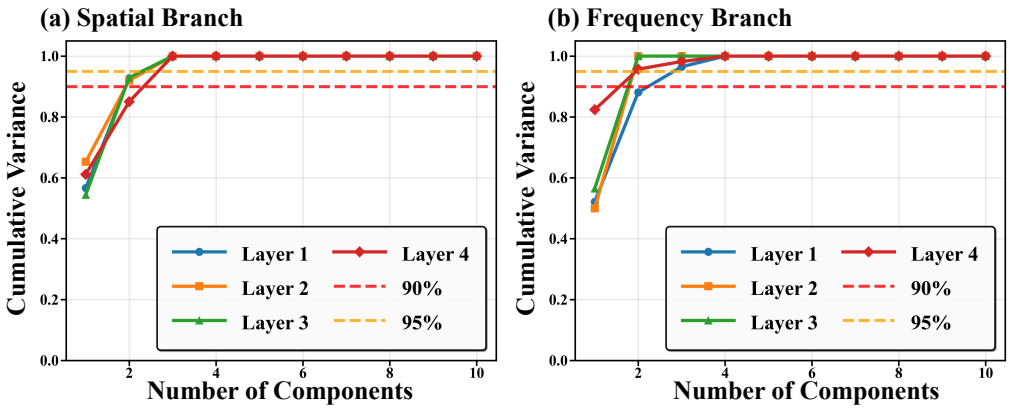


Figure 2: Cumulative variance explained by principal components of learned convolution kernels on Forcing NS (Re=4000). **(a) Spatial branch** and **(b) Frequency branch** across 4 layers. Remarkably, the first 2 components capture >90% variance and 3 components exceed >95% (horizontal dashed lines), validating the extremely low-rank structure induced by SINO's bottleneck architecture and confirming that the model prioritizes dominant physical modes over high-dimensional noise.

4 Experiments

We validate SINO on turbulent closure problems spanning 1D and 2D systems, comparing against diverse baselines and conducting ablation studies to verify effectiveness.

4.1 Experimental Setup

Benchmarks. We evaluate SINO on six turbulent closure benchmarks with aggressive coarse-graining ratios. Table 1 summarizes the configurations, including forcing and decaying variants of Burgers turbulence, Kuramoto-Sivashinsky (KS) turbulence, and Navier-Stokes (NS) turbulence at different Reynolds numbers. The forcing variants use external forcing (Kolmogorov forcing for NS), while the decaying variants exhibit freely decaying dynamics, testing the model's ability to capture diverse physical regimes. Supplementary experiments at alternative resolutions and forcing configurations (Appendix A.5) further validate SINO's scale-invariant capability. Detailed numerical schemes and simulation protocols are in Appendix A.1.

Table 1: Benchmark configurations.

Problem	Dim	Resolution	Spatial Discretization	Temporal Scheme
Forcing Burgers	1D	512→32	WENO5 Finite Volume Method	SSP-RK3
Decaying Burgers	1D	2048→256	Pseudo-Spectral	TVD RK3
KS	1D	256→64	Pseudo-Spectral	ETDRK4
Forcing NS (Re=1000)	2D	512→64	Finite Volume Method (Van Leer)	Semi-Implicit
Forcing NS (Re=4000)	2D	512→64	Finite Volume Method (Van Leer)	Semi-Implicit
Decaying NS	2D	512→64	Finite Volume Method (Van Leer)	Semi-Implicit

Baselines. We compare SINO against nine baselines covering three paradigms: traditional models (U-Net, DeepONet), Transformer-based operators (Transolver, Oformer, GK-Transformer), and frequency-domain operators (FNO, AMFNO, UFNO), plus uncorrected coarse-grid DNS as a physical baseline. Model specifications and training protocols are in Appendices A.2 and A.3.

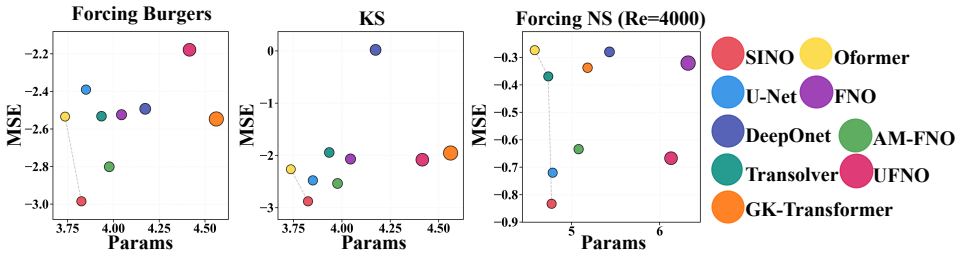


Figure 3: Pareto comparison of MSE versus parameter count on Forcing Burgers, KS, and Forcing NS (Re=4000). Both axes are on a logarithmic scale. SINO (red) achieves optimal or near-optimal MSE with significantly fewer parameters than FNO-based methods, demonstrating parameter efficiency from implicit low-rank representations.

4.2 Main Results

Tables 2 and 3 present quantitative comparisons on 1D and 2D turbulence benchmarks. SINO consistently achieves superior or competitive performance across all tasks while maintaining parameter efficiency. Supplementary experiments on additional resolutions and forcing configurations (Tables 4 and 5 in Appendix) confirm robustness across diverse physical regimes. Following the Indirect Neural Corrector (INC) framework (Wei et al., 2026), SINO predicts closure terms as right-hand side corrections; training details are in Section A.3. All reported metrics are extrapolation errors: 30% of time steps for training, 70% for testing.

1D Benchmarks. On Forcing Burgers, SINO achieves the lowest error (1.03E-03 average, 9.41E-04 best) with only 6.7K parameters, outperforming second-best AMFNO (1.58E-03 average, 9.5K parameters) by 1.5×. Transformer-based methods yield comparable parameter counts but higher errors: GK-Transformer (2.84E-03 average, 36K parameters), Oformer (2.92E-03, 5.4K parameters), and Transolver (2.94E-03, 8.6K parameters). FNO variants show mixed results: FNO (2.99E-03, 11K parameters), DeepONet (3.22E-03, 15K parameters), U-Net (4.07E-03, 7.1K parameters), and UFNO (6.63E-03, 26K parameters) all lag behind SINO. On KS turbulence, SINO (1.33E-03 av-

erage, 1.14E-03 best) surpasses all baselines by substantial margins: 2.2× better than AMFNO (2.92E-03), 2.5× better than U-Net (3.33E-03), and 6–8× better than FNO variants and Transformers. For Decaying Burgers, SINO (4.37E-03 average, 2.90E-03 best) achieves 38× improvement over most baselines, which plateau around 1.65E-01. Only SINO-SPA (2.09E-02 average, 1.8K parameters) approaches SINO's performance among ablation variants.

Table 2: Performance comparison on 1D benchmarks.

Method	**1D Benchmarks**						**Params**
	Forcing Burgers		**KS**		**Decaying Burgers**		
	AVE	**Best**	**AVE**	**Best**	**AVE**	**Best**	
U-Net	4.07E-03	3.31E-03	3.33E-03	1.48E-03	1.65E-01	1.65E-01	7,077
DeepONet	3.22E-03	2.78E-03	1.05E+00	1.39E+00	1.65E-01	1.65E-01	14,913
Transolver	2.94E-03	2.80E-03	1.14E-02	1.09E-02	1.65E-01	1.64E-01	8,601
Oformer	2.92E-03	2.92E-03	5.45E-03	5.45E-03	1.65E-01	1.65E-01	5,444
GK-Transformer	2.84E-03	2.77E-03	1.12E-02	1.11E-02	1.65E-01	1.64E-01	36,449
FNO	2.99E-03	2.82E-03	8.55E-03	8.03E-03	1.65E-01	1.65E-01	11,073
UFNO	6.63E-03	5.70E-03	8.28E-03	7.82E-03	1.65E-01	1.65E-01	26,017
AM-FNO	1.58E-03	1.34E-03	2.92E-03	2.50E-03	1.66E-01	1.65E-01	9,489
SINO-SPE	1.56E-03	1.32E-03	3.02E-03	2.49E-03	1.28E-01	1.27E-01	3,877
SINO-SPA	2.12E-03	1.79E-03	3.53E-03	2.11E-03	2.09E-02	4.08E-03	1,797
SINO (ours)	**1.03E-03**	**9.41E-04**	**1.33E-03**	**1.14E-03**	**4.37E-03**	**2.90E-03**	6,681
Coarse Grid	2.23E-01		5.46E-03		1.65E-01		–

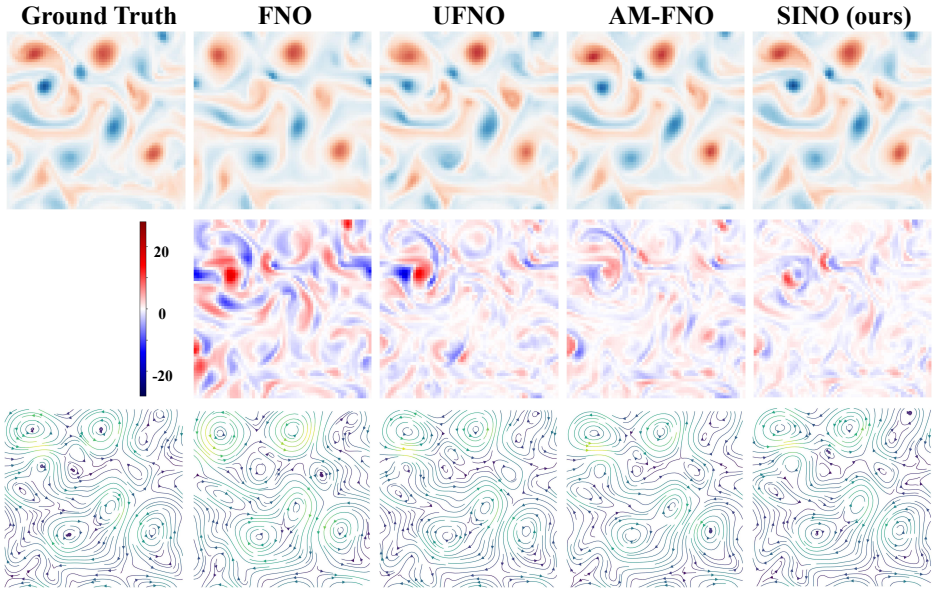


Figure 4: Instantaneous predictions on Kolmogorov-forced Navier–Stokes turbulence at Re=4000. Column 1: ground truth downsampled from DNS-512 to 64×64 resolution. Columns 2–5: FNO, UFNO, AMFNO, SINO predictions. Row 1: vorticity fields. Row 2: vorticity error distributions. Row 3: streamline plots. SINO accurately reconstructs fine-scale structures and velocity patterns with minimal error, while other methods exhibit varying degrees of distortion in small-scale details.

2D Benchmarks. On Forcing NS (Re=1000), SINO (9.17E-02 average, 8.43E-02 best, 59K parameters) leads all methods. U-Net (1.34E-01, 61K parameters) achieves second place with comparable parameter count but 1.5× higher error. FNO-based methods underperform: UFNO (1.64E-01, 1.3M parameters) and AMFNO (1.69E-01, 120K parameters) require 2–23× more parameters for inferior accuracy. Transformer methods exhibit substantially higher errors: Transolver (3.65E-01, 54K

parameters), GK-Transformer (3.87E-01, 152K parameters), and Oformer (5.07E-01, 38K parameters). FNO (4.44E-01, 2.1M parameters) and DeepONet (4.62E-01, 269K parameters) underperform despite massive capacity. At Re=4000, similar trends emerge: SINO (1.47E-01 average, 1.41E-01 best) maintains first place, followed by U-Net (1.91E-01), UFNO (2.15E-01), and AMFNO (2.32E-01). On Decaying NS, UFNO achieves the lowest average error (5.58E-02) but requires 1.3M parameters—23× more than SINO (5.99E-02 average, 59K parameters). Notably, SINO attains the best peak performance (4.55E-02) across all methods, demonstrating superior optimization capability. U-Net (6.67E-02, 61K parameters) ranks third, while AMFNO (7.68E-02, 120K parameters) requires 2× more parameters for worse accuracy.

Table 3: Performance comparison on 2D benchmarks.

Method	**2D Benchmarks**						**Params**
	Forcing NS (Re=1000)		**Forcing NS (Re=4000)**		**Decaying NS**		
	AVE	**Best**	**AVE**	**Best**	**AVE**	**Best**	
U-Net	1.34E-01	1.16E-01	1.91E-01	1.84E-01	6.67E-02	5.64E-02	61,006
DeepONet	4.62E-01	4.60E-01	5.25E-01	5.18E-01	2.95E-01	2.79E-01	269,153
Transolver	3.65E-01	3.48E-01	4.27E-01	4.23E-01	2.24E-01	2.17E-01	54,306
Oformer	5.07E-01	5.07E-01	5.33E-01	5.33E-01	4.03E-01	4.03E-01	38,087
GK-Transformer	3.87E-01	3.84E-01	4.60E-01	4.55E-01	2.04E-01	1.96E-01	151,874
FNO	4.44E-01	4.42E-01	4.78E-01	4.75E-01	2.60E-01	2.40E-01	2,106,018
UFNO	1.64E-01	1.51E-01	2.15E-01	2.12E-01	**5.58E-02**	4.86E-02	1,346,466
AM-FNO	1.69E-01	1.59E-01	2.32E-01	2.15E-01	7.68E-02	7.05E-02	120,162
SINO-SPE	2.18E-01	1.81E-01	2.63E-01	2.40E-01	1.71E-01	1.51E-01	33,834
SINO-SPA	2.23E-01	2.20E-01	2.82E-01	2.72E-01	1.37E-01	1.24E-01	17,322
SINO (ours)	**9.17E-02**	**8.43E-02**	**1.47E-01**	**1.41E-01**	5.99E-02	**4.55E-02**	59,314
Coarse Grid	5.07E-01		5.32E-01		4.04E-01		–

Ablation Analysis. SINO-SPE (frequency-only) and SINO-SPA (spatial-only) consistently underperform full SINO by 1.5–3× across all benchmarks, confirming the necessity of dual-branch architecture. SINO-SPE shows particular degradation on 2D problems (2.18E-01 vs. 9.17E-02 on Forcing NS Re=1000), while SINO-SPA struggles on forced problems requiring global energy transfer. The full model synergistically combines both branches for optimal performance.

4.3 Scaling Laws and Cross-Scale Analysis

Parameter Efficiency. Figure 5 reveals SINO's superior scaling behavior through power-law exponents. On forced NS (Re=4000), SINO achieves $\alpha = 0.568$—nearly 38× steeper than FNO ($\alpha = 0.015$) and 3–4× steeper than AMFNO ($\alpha = 0.143$) and UFNO ($\alpha = 0.162$). On decaying NS, SINO maintains $\alpha = 0.868$ compared to FNO's $\alpha = 0.073$, demonstrating 12× better parameter efficiency. These steep slopes indicate that SINO effectively learns low-rank physical structures: each additional parameter contributes substantially to error reduction, whereas baselines require orders of magnitude more capacity for comparable gains.

Spectral Fidelity. Figure 6 validates SINO's cross-scale capability through energy spectra on decaying Burgers turbulence. Traditional coarse grids exhibit severe numerical dissipation beyond the Nyquist cutoff (left panel, vertical dashed lines mark resolution limits). At rf = 8 (N = 256), baseline methods (AMFNO, FNO, UFNO) deviate from the correct k^{-2} decay slope and fail to suppress spurious dissipation (middle panel). In contrast, SINO maintains accurate spectral behavior across resolutions: even at extreme downsampling (rf = 16, N = 128), SINO's energy spectrum aligns with high-resolution DNS (N = 2048) and preserves the theoretical k^{-2} slope (right panel), confirming its scale-invariant representations.

Bottleneck Capacity Scaling. To investigate the impact of bottleneck dimensions on performance, Table 6 compares SINO variants with (n_f, n_s) ranging from 2 to 32 against uncorrected DNS at multiple resolutions. SINO-32 achieves errors of 4.08E-02 (Decaying NS) and 6.06E-02 (Forcing NS Re=1000), representing 1.5–1.7× improvement over SINO-2 and exceeding the accuracy of DNS-256 (4× higher resolution with 16× more grid points). Figure 9 confirms SINO-32 maintains

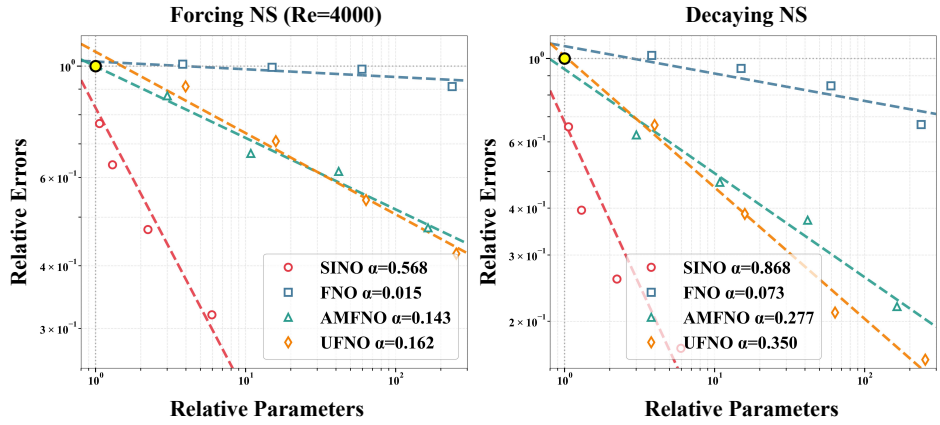


Figure 5: Scaling laws on forced NS (Re=4000, left) and decaying NS (right). SINO exhibits steeper power-law exponents ($\alpha = 0.568$ and 0.868) than baselines, demonstrating better parameter efficiency.

superior temporal stability in long-term extrapolation. This demonstrates that increasing bottleneck capacity enables SINO to surpass even higher-resolution coarse-grid simulations, validating the effectiveness of further parameter investment in low-rank subspace learning.

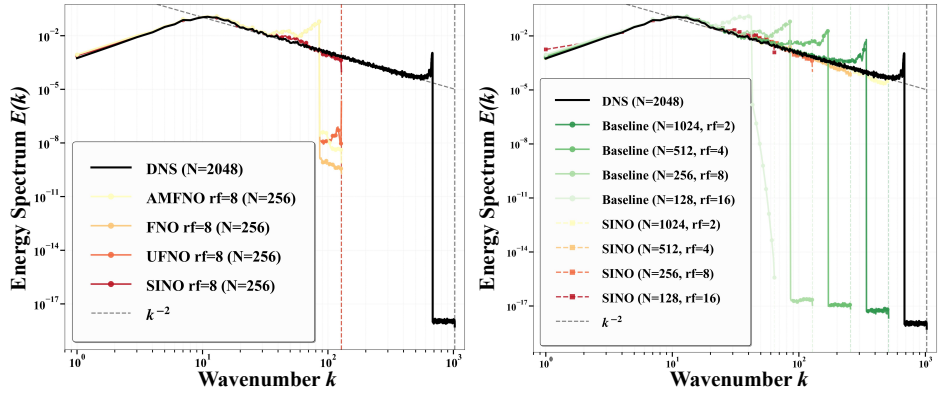


Figure 6: Energy spectra on decaying Burgers turbulence. **Left:** At resample factor (rf) = 8, SINO most accurately follows the k^{-2} inertial range among neural operators. **Right:** Across rf = 2–16, baselines (green, solid) exhibit numerical dissipation beyond cutoffs, while SINO (red-yellow, dashed) preserves correct spectral decay, validating scale-invariant modeling.

5 Conclusion

We propose the Scale-Invariant Neural Operator (SINO), a dual-branch architecture that learns turbulent closure terms on normalized physical scales. By parameterizing convolution kernels as continuous functions via bottleneck hypernetworks, SINO decouples model capacity from grid resolution, enabling compact representation of scale-invariant physical laws. The bottleneck structure enforces explicit low-rank constraints, concentrating >95% variance in merely 2-3 modes (validated by PCA) and forcing the model to discover dominant physical mechanisms rather than memorize high-dimensional patterns. This yields 38× steeper scaling law exponents than FNO, demonstrating superior parameter efficiency. Experiments on six turbulent closure benchmarks demonstrate 1.5–38× error reduction with 2–23× parameter efficiency over baselines. Ablation studies confirm dual-branch necessity, and spectral analysis validates cross-scale generalization under extreme downsampling. Future work will extend SINO to three-dimensional flows and diverse physical problems, establishing SINO as a general framework for learning scale-invariant laws where low-rank structure governs system dynamics.

AI USE STATEMENT

In this work, we used generative Al tools for translation, code implementation, and experimentadesign. We have not used generative Al tools for synthetic data generation or dataset cleaningand qualitative analyses are not applicable to this work, Additionally, we used generative Al toolsfor literature searches, manuscript drafting and editing, LaTeX formatting, code debugging, andvisualization. We have reviewed all Al-assisted work. We checked and revised the text, mathematicscode, and experimental analyses. We take responsibility for the final content of this work, includingtext, claims or artifacts produced with the aid of generative Al.

REPRODUCIBILITY STATEMENT

We provide a complete working example at `https://anonymous.4open.science/r/SINO-662D`, including code, data generation scripts, and detailed instructions for reproducing all experimental results.

REFERENCES

Hervé Abdi and Lynne J Williams. Principal component analysis. *Wiley interdisciplinary reviews: computational statistics*, 2(4):433–459, 2010.

Yohai Bar-Sinai, Stephan Hoyer, Jason Hickey, and Michael P Brenner. Learning data-driven discretizations for partial differential equations. *Proceedings of the National Academy of Sciences*, 116(31):15344–15349, 2019.

Gal Berkooz, Philip Holmes, and John L Lumley. The proper orthogonal decomposition in the analysis of turbulent flows. *Annual review of fluid mechanics*, 25(1):539–575, 1993.

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

Julia Grabinski, Janis Keuper, and Margret Keuper. As large as it gets-studying infinitely large convolutions via neural implicit frequency filters. *Transactions on Machine Learning Research*, 2024:1–42, 2024.

David Ha, Andrew Dai, and Quoc V. Le. Hypernetworks, 2016. URL `https://arxiv.org/abs/1609.09106`.

Stefan Heinz. A review of hybrid rans-les methods for turbulent flows: Concepts and applications. *Progress in Aerospace Sciences*, 114:100597, 2020.

Philip Holmes. *Turbulence, coherent structures, dynamical systems and symmetry*. Cambridge university press, 2012.

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

Dmitrii Kochkov, Jamie A Smith, Ayya Alieva, Qing Wang, Michael P Brenner, and Stephan Hoyer. Machine learning–accelerated computational fluid dynamics. *Proceedings of the National Academy of Sciences*, 118(21):e2101784118, 2021.

Andrei Nikolaevich Kolmogorov. The local structure of turbulence in incompressible viscous fluid for very large reynolds numbers. *Proceedings: Mathematical and Physical Sciences*, 434(1890): 9–13, 1991.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012.

Athony Leonard. Energy cascade in large-eddy simulations of turbulent fluid flows. In *Advances in geophysics*, volume 18, pp. 237–248. Elsevier, 1975.

Zijie Li, Kazem Meidani, and Amir Barati Farimani. Transformer for partial differential equations' operator learning. *arXiv preprint arXiv:2205.13671*, 2022.

Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Fourier neural operator for parametric partial differential equations. *arXiv preprint arXiv:2010.08895*, 2020.

Björn List, Li-Wei Chen, and Nils Thuerey. Learned turbulence modelling with differentiable fluid solvers: physics-based loss functions and optimisation horizons. *Journal of Fluid Mechanics*, 949:A25, 2022.

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

John Leask Lumley. The structure of inhomogeneous turbulent flows. *Atmospheric turbulence and radio wave propagation*, pp. 166–178, 1967.

Tianyu Ma, Alan Q Wang, Adrian V Dalca, and Mert R Sabuncu. Hyper-convolutions via implicit kernels for medical imaging. *arXiv preprint arXiv:2202.02701*, 2022.

Charles Meneveau and Joseph Katz. Scale-invariance and turbulence models for large-eddy simulation. *Annual Review of Fluid Mechanics*, 32(1):1–32, 2000.

David W Romero, Anna Kuzina, Erik J Bekkers, Jakub M Tomczak, and Mark Hoogendoorn. Ckconv: Continuous kernel convolution for sequential data. *arXiv preprint arXiv:2102.02611*, 2021.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pp. 234–241. Springer, 2015.

Benjamin Sanderse, Panos Stinis, Romit Maulik, and Shady E Ahmed. Scientific machine learning for closure models in multiscale problems: A review. *arXiv preprint arXiv:2403.02913*, 2024.

Varun Shankar, Vedant Puri, Ramesh Balakrishnan, Romit Maulik, and Venkatasubramanian Viswanathan. Differentiable physics-enabled closure modeling for burgers' turbulence. *Machine Learning: Science and Technology*, 4(1):015017, 2023.

Vincent Thibeault, Antoine Allard, and Patrick Desrosiers. The low-rank hypothesis of complex systems. *Nature Physics*, 20(2):294–302, 2024.

Kiwon Um, Robert Brand, Yun Raymond Fei, Philipp Holl, and Nils Thuerey. Solver-in-the-loop: Learning from differentiable physics to interact with iterative pde-solvers. *Advances in neural information processing systems*, 33:6111–6122, 2020.

Hao Wei, Aleksandra Franz, Bjoern List, and Nils Thuerey. Inc: An indirect neural corrector for auto-regressive hybrid pde solvers. *Advances in Neural Information Processing Systems*, 38: 110182–110216, 2026.

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

Haixu Wu, Huakun Luo, Haowen Wang, Jianmin Wang, and Mingsheng Long. Transolver: A fast transformer solver for pdes on general geometries. *arXiv preprint arXiv:2402.02366*, 2024.

Zipeng Xiao, Siqi Kou, Zhongkai Hao, Bokai Lin, and Zhijie Deng. Amortized fourier neural operators. *Advances in Neural Information Processing Systems*, 37:115001–115020, 2024.

A APPENDIX

This appendix provides comprehensive technical details supporting the main results. Section A.1 specifies the governing equations, numerical discretization schemes, and simulation protocols for all six turbulent closure benchmarks. Section A.2 presents complete architectural specifications for SINO and all baseline models across 1D and 2D problems. Section A.3 documents the training objectives, rollout integration with closure term incorporation, curriculum learning strategy, optimization settings, and time integration protocols. Section A.4 establishes theoretical foundations through formal proofs of SINO's low-rank structure, Lipschitz regularity, and operator norm bounds. Finally, Section A.5 reports additional experimental results including cross-resolution and alternative forcing generalization tests, principal component analysis confirming low-rank structure across different flow regimes, and bottleneck dimension scaling studies demonstrating computational trade-offs.

A.1 BENCHMARK EQUATIONS

A.1.1 FORCING BURGERS EQUATION

Governing Equation. We consider the one-dimensional viscous Burgers equation in conservative form with external forcing:

$$\frac{\partial u}{\partial t} + \frac{\partial}{\partial x}\left(\frac{u^2}{2}\right) = \eta \frac{\partial^2 u}{\partial x^2} + f(x,t), \quad x \in [0, L], \quad t > 0, \tag{8}$$

where $u(x,t)$ is the velocity field, $\eta = 0.01$ is the kinematic viscosity, and $f(x,t)$ is the external forcing term. The domain length is $L = 2\pi$ with periodic boundary conditions.

Initial and Boundary Conditions. The initial condition is set to zero velocity everywhere: $u(x,0) = 0$. Periodic boundary conditions are enforced: $u(0,t) = u(L,t)$ for all $t \geq 0$. A warmup phase of $T_{\text{warmup}} = 2.0$ is applied to allow shock structures to develop before recording snapshots. The external forcing is given by:

$$f(x,t) = \sum_{i=1}^{n_{\text{modes}}} A_i \sin\left(\omega_i t + \frac{2\pi \ell_i x}{L} + \phi_i\right), \tag{9}$$

where $n_{\text{modes}} = 20$, and for each mode i: $A_i \sim \mathcal{U}(-0.5, 0.5)$, $\omega_i \sim \mathcal{U}(-0.4, 0.4)$, $\ell_i \sim \mathcal{U}_{\text{discrete}}\{3,4,5,6\}$, and $\phi_i \sim \mathcal{U}(0, 2\pi)$ are randomly sampled once per trajectory and held constant throughout the simulation.

Spatial Discretization. The equation is discretized using a finite volume method with fifth-order Weighted Essentially Non-Oscillatory (WENO5) reconstruction on a uniform grid of $N = 512$ cells with cell size $\Delta x = L/N$. Let $\bar{u}_i(t)$ denote the cell-averaged value in cell i. The WENO5 reconstruction computes left and right interface values $u^-_{i+1/2}$ and $u^+_{i+1/2}$ from a five-point stencil $\{\bar{u}_{i-2}, \bar{u}_{i-1}, \bar{u}_i, \bar{u}_{i+1}, \bar{u}_{i+2}\}$ via:

$$u^-_{i+1/2} = \sum_{k=0}^{2} \omega^-_k p^-_k(u_{i-2+k:i+2+k}), \quad u^+_{i+1/2} = \sum_{k=0}^{2} \omega^+_k p^+_k(u_{i-1-k:i+3-k}), \tag{10}$$

where $p^\pm_k$ are third-order polynomial interpolants and $\omega^\pm_k$ are nonlinear weights computed from smoothness indicators:

$$\omega_k = \frac{\alpha_k}{\sum_{j=0}^{2} \alpha_j}, \quad \alpha_k = \frac{d_k}{(\epsilon + \beta_k)^2}, \tag{11}$$

with optimal linear weights $d_0 = 0.1$, $d_1 = 0.6$, $d_2 = 0.3$, regularization parameter $\epsilon = 10^{-6}$, and smoothness indicators β_k measuring local solution variation. The convective flux is computed using the Godunov numerical flux:

$$F^{\text{conv}}_{i+1/2} = \begin{cases} 0, & \text{if } u^-_{i+1/2} \leq 0 \leq u^+_{i+1/2}, \\ \min\left(\frac{(u^-_{i+1/2})^2}{2}, \frac{(u^+_{i+1/2})^2}{2}\right), & \text{if } u^-_{i+1/2} \leq u^+_{i+1/2}, \\ \max\left(\frac{(u^-_{i+1/2})^2}{2}, \frac{(u^+_{i+1/2})^2}{2}\right), & \text{otherwise.} \end{cases} \tag{12}$$

The diffusive flux is approximated by second-order central differences: $F^{\text{diff}}_{i+1/2} = -\eta(\bar{u}_{i+1} - \bar{u}_i)/\Delta x$. The semi-discrete form becomes:

$$\frac{d\bar{u}_i}{dt} = -\frac{F_{i+1/2} - F_{i-1/2}}{\Delta x} + f_i(t), \quad F_{i+1/2} = F^{\text{conv}}_{i+1/2} + F^{\text{diff}}_{i+1/2}. \tag{13}$$

Temporal Discretization. Time integration employs the third-order Strong Stability Preserving Runge-Kutta method (SSP-RK3) in Shu-Osher form:

$$\bar{u}^{(1)} = \bar{u}^n + \Delta t\, \mathcal{L}(\bar{u}^n), \tag{14}$$

$$\bar{u}^{(2)} = \frac{3}{4}\bar{u}^n + \frac{1}{4}\left[\bar{u}^{(1)} + \Delta t\, \mathcal{L}(\bar{u}^{(1)})\right], \tag{15}$$

$$\bar{u}^{n+1} = \frac{1}{3}\bar{u}^n + \frac{2}{3}\left[\bar{u}^{(2)} + \Delta t\, \mathcal{L}(\bar{u}^{(2)})\right], \tag{16}$$

where $\mathcal{L}(\bar{u})$ denotes the right-hand side spatial operator. The timestep Δt is adaptively chosen at each RK stage to satisfy the CFL condition with safety factor $\nu_{\text{CFL}} = 0.4$:

$$\Delta t = \nu_{\text{CFL}} \cdot \min\left(\frac{\Delta x}{\max_i |\bar{u}_i|}, \frac{\Delta x^2}{2\eta}\right). \tag{17}$$

Simulation Protocol. Each trajectory undergoes a warmup phase of $T_{\text{warmup}} = 2.0$ time units to develop shock structures, followed by a recording phase of $T_{\text{max}} = 10.0$ time units. During the recording phase, $n_{\text{snapshots}} = 1001$ snapshots (including the initial state at $t = 0$ of the recording phase) are saved at uniform intervals $\Delta t_{\text{outer}} = T_{\text{max}}/(n_{\text{snapshots}} - 1) = 0.01$. A total of 64 trajectories are generated with different random forcing parameters. The fine-grid snapshots at resolution $N = 512$ are coarsened to multiple resolutions via exact finite volume averaging. For instance, with resample factor $r = 16$, the snapshots are downsampled to $N = 32$, yielding the coarse-grid reference states $\bar{u}_\Delta$ used for closure modeling. The DNS solver employs adaptive timesteps determined by the CFL condition: at each RK substep, the timestep is computed as $\Delta t = \nu_{\text{CFL}} \cdot \min(\Delta x/\max_i |\bar{u}_i|,\, \Delta x^2/(2\eta))$, where the first term represents the convective stability limit and the second term the diffusive stability limit. The minimum of these two constraints ensures numerical stability throughout the simulation, with the timestep adapting dynamically to the instantaneous solution state.

A.1.2 Kuramoto–Sivashinsky Equation

Governing Equation. We consider the one-dimensional Kuramoto–Sivashinsky (KS) equation on a periodic domain:

$$\frac{\partial v}{\partial t} + v\frac{\partial v}{\partial x} + \frac{\partial^2 v}{\partial x^2} + \frac{\partial^4 v}{\partial x^4} = 0, \quad x \in [0, L], \quad t > 0, \tag{18}$$

where $v(x,t)$ is the scalar field, and $L = 64$ is the domain length. The KS equation exhibits spatiotemporal chaos and is a canonical model for pattern formation and turbulence in dissipative systems.

Initial and Boundary Conditions. Periodic boundary conditions are imposed: $v(0,t) = v(L,t)$ and $\partial^n v/\partial x^n|_{x=0} = \partial^n v/\partial x^n|_{x=L}$ for all n and $t \geq 0$. The initial condition is constructed as a random superposition of Fourier modes:

$$v(x,0) = \sum_{i=1}^{n_{\text{modes}}} A_i \sin\left(\frac{2\pi \ell_i x}{L} + \phi_i\right), \tag{19}$$

where $n_{\text{modes}} = 10$, and for each mode i: $A_i \sim \mathcal{U}(-0.5, 0.5)$, $\phi_i \sim \mathcal{U}(0, 2\pi)$, and $\ell_i \sim \mathcal{U}_{\text{discrete}}\{1,2,3\}$ are randomly sampled once per trajectory. A warmup phase of $T_{\text{warmup}} = 50.0$ time units is applied before recording to allow the system to reach a statistically stationary chaotic state.

Spatial Discretization. The KS equation is discretized using a pseudo-spectral collocation method on a uniform grid of $N = 256$ points with grid spacing $\Delta x = L/N$. The solution is represented in Fourier space as $\hat{v}_k(t) = \mathcal{F}[v](k)$, where $k = (2\pi/L) \cdot k_{\text{index}}$ are the wavenumbers with $k_{\text{index}} \in \{-N/2, \ldots, N/2 - 1\}$. The spatial derivatives are computed exactly in Fourier space:

$$\frac{\partial^n v}{\partial x^n} \quad \Longleftrightarrow \quad (ik)^n \hat{v}_k. \tag{20}$$

The nonlinear term $v\partial v/\partial x$ is evaluated in physical space and transformed back to Fourier space. To eliminate aliasing errors from the quadratic nonlinearity, the 2/3-rule dealiasing is enforced: all Fourier modes with $|k_{\text{index}}| > N/3$ are set to zero both before computing the nonlinear term (on input) and after computing it (on output). This retains approximately $2N/3 + 1 \approx 171$ modes out of 256, ensuring that the product of two dealiased fields contains no aliasing.

Temporal Discretization. Time integration employs the fourth-order Exponential Time Differencing Runge-Kutta method (ETDRK4) with a fixed timestep $\Delta t = 0.01$. The KS equation is split into linear and nonlinear parts:

$$\frac{\partial \hat{v}_k}{\partial t} = \hat{L}_k \hat{v}_k + \hat{\mathcal{N}}_k(\hat{v}), \quad \hat{L}_k = k^2 - k^4, \quad \hat{\mathcal{N}}_k(\hat{v}) = -\widehat{v \frac{\partial v}{\partial x}}_k. \tag{21}$$

The ETDRK4 scheme integrates the linear term exactly and treats the nonlinear term via four stages:

$$\hat{v}^{(a)} = e^{\Delta t \hat{L}_k/2} \hat{v}^n + Q_k \hat{\mathcal{N}}_k(\hat{v}^n), \tag{22}$$

$$\hat{v}^{(b)} = e^{\Delta t \hat{L}_k/2} \hat{v}^n + Q_k \hat{\mathcal{N}}_k(\hat{v}^{(a)}), \tag{23}$$

$$\hat{v}^{(c)} = e^{\Delta t \hat{L}_k/2} \hat{v}^{(a)} + Q_k \left[2\hat{\mathcal{N}}_k(\hat{v}^{(b)}) - \hat{\mathcal{N}}_k(\hat{v}^n) \right], \tag{24}$$

$$\hat{v}^{n+1} = e^{\Delta t \hat{L}_k} \hat{v}^n + f_{1,k} \hat{\mathcal{N}}_k(\hat{v}^n) + 2f_{2,k} \left[\hat{\mathcal{N}}_k(\hat{v}^{(a)}) + \hat{\mathcal{N}}_k(\hat{v}^{(b)}) \right] + f_{3,k} \hat{\mathcal{N}}_k(\hat{v}^{(c)}), \tag{25}$$

where the coefficients $Q_k, f_{1,k}, f_{2,k}, f_{3,k}$ are computed using the contour integral method of Kassam and Trefethen to avoid numerical cancellation errors near $\hat{L}_k \approx 0$:

$$Q_k = \Delta t \int_0^{1/2} e^{\tau \Delta t \hat{L}_k} d\tau = \Delta t \cdot \frac{e^{\Delta t \hat{L}_k/2} - 1}{\Delta t \hat{L}_k}, \tag{26}$$

$$f_{1,k} = \Delta t \int_0^1 e^{(1-\tau)\Delta t \hat{L}_k} \left[-4 - \tau \Delta t \hat{L}_k + e^{\tau \Delta t \hat{L}_k} (4 - 3\tau \Delta t \hat{L}_k + \tau^2 (\Delta t \hat{L}_k)^2) \right] \frac{d\tau}{(\tau \Delta t \hat{L}_k)^3}, \tag{27}$$

$$f_{2,k} = \Delta t \int_0^1 e^{(1-\tau)\Delta t \hat{L}_k} \left[2 + \tau \Delta t \hat{L}_k + e^{\tau \Delta t \hat{L}_k} (-2 + \tau \Delta t \hat{L}_k) \right] \frac{d\tau}{(\tau \Delta t \hat{L}_k)^3}, \tag{28}$$

$$f_{3,k} = \Delta t \int_0^1 e^{(1-\tau)\Delta t \hat{L}_k} \left[-4 - 3\tau \Delta t \hat{L}_k - \tau^2 (\Delta t \hat{L}_k)^2 + e^{\tau \Delta t \hat{L}_k} (4 - \tau \Delta t \hat{L}_k) \right] \frac{d\tau}{(\tau \Delta t \hat{L}_k)^3}. \tag{29}$$

These integrals are evaluated numerically via 32-point trapezoidal rule on a circular contour in the complex plane with radius $r = 1$, ensuring high accuracy even when $\hat{L}_k = 0$ (the DC mode).

Simulation Protocol. Each trajectory begins from a random initial condition and undergoes a warmup phase of $T_{\text{warmup}} = 50.0$ time units (5000 timesteps) to reach the chaotic attractor. After warmup, snapshots are recorded for a duration of $T_{\text{max}} = 10.0$ time units at intervals $\Delta t = 0.01$, yielding $n_{\text{snapshots}} = 1001$ snapshots (including the initial state at $t = 0$ of the recording phase). A total of 64 trajectories are generated with different random initial conditions. The fine-grid snapshots at resolution $N = 256$ are coarsened to multiple resolutions via spectral downsampling (truncation in Fourier space followed by inverse FFT). For instance, with resample factor $r = 4$, the snapshots are downsampled to $N = 64$, yielding the coarse-grid reference states $\bar{v}_\Delta$ used for closure modeling. The ETDRK4 integrator with fixed timestep $\Delta t = 0.01$ ensures fourth-order temporal accuracy while maintaining exponential stability for the stiff linear operator.

A.1.3 DECAYING BURGERS EQUATION

Governing Equation. We consider the one-dimensional viscous Burgers equation in conservative form without external forcing:

$$\frac{\partial u}{\partial t} + \frac{\partial}{\partial x}\left(\frac{u^2}{2}\right) = \nu \frac{\partial^2 u}{\partial x^2}, \quad x \in [0, L], \quad t > 0, \tag{30}$$

where $u(x,t)$ is the velocity field, $\nu = 5 \times 10^{-4}$ is the kinematic viscosity, and $L = 2\pi$ is the domain length with periodic boundary conditions. This configuration exhibits free decay of turbulent kinetic energy, making it a canonical test case for subgrid-scale (SGS) closure modeling.

Initial and Boundary Conditions. Periodic boundary conditions are enforced: $u(0,t) = u(L,t)$ for all $t \geq 0$. The initial condition is generated from a broadband energy spectrum following Maulik and San (2018):

$$E(k) = A\, k^4 \exp\left(-\left(\frac{k}{k_0}\right)^2\right), \quad A = \frac{2k_0^{-5}}{3\sqrt{\pi}}, \tag{31}$$

where $k_0 = 10$ is the peak wavenumber. For each trajectory, the initial velocity field is constructed as:

$$u(x,0) = \sum_{k=1}^{N/2-1} \sqrt{2E(k)}\, N \, \cos\left(k\frac{2\pi x}{L} + \phi_k\right), \tag{32}$$

where $\phi_k \sim \mathcal{U}(0, 2\pi)$ are independent random phases ensuring a real-valued field. This initialization produces a turbulent state with energy concentrated around intermediate wavenumbers, which subsequently undergoes viscous dissipation without external forcing.

Spatial Discretization. The equation is discretized using a pseudo-spectral collocation method on a uniform grid of $N = 2048$ points with grid spacing $\Delta x = L/N$. The solution is represented in Fourier space as $\hat{u}_k(t) = \mathcal{F}[u](k)$, where $k = (2\pi/L) \cdot k_{\text{index}}$ are the wavenumbers. Linear operators (diffusion) are computed exactly in Fourier space:

$$\frac{\partial^2 u}{\partial x^2} \quad \Longleftrightarrow \quad -k^2 \hat{u}_k. \tag{33}$$

The nonlinear convective term $\partial(u^2/2)/\partial x$ is evaluated in physical space via the collocation method: $u^2/2$ is computed pointwise, transformed to Fourier space, then multiplied by ik. To eliminate aliasing errors from the quadratic nonlinearity, the $2/3$-rule dealiasing is applied: all Fourier modes with $|k_{\text{index}}| > N/3$ are set to zero both before computing the nonlinear term (on input) and after transforming back to Fourier space (on output). This retains approximately $2N/3 + 1 \approx 1366$ modes out of 2048.

Temporal Discretization. Time integration employs the third-order Total Variation Diminishing Runge-Kutta method (TVD-RK3) with adaptive timestep selection based on the CFL condition. The TVD-RK3 scheme proceeds via three stages:

$$u^{(1)} = u^n + \Delta t\, \mathcal{L}(u^n), \tag{34}$$

$$u^{(2)} = \frac{3}{4}u^n + \frac{1}{4}\left[u^{(1)} + \Delta t\, \mathcal{L}(u^{(1)})\right], \tag{35}$$

$$u^{n+1} = \frac{1}{3}u^n + \frac{2}{3}\left[u^{(2)} + \Delta t\, \mathcal{L}(u^{(2)})\right], \tag{36}$$

where $\mathcal{L}(u)$ denotes the spatial operator including both convective and diffusive terms. At each RK substep, the timestep Δt is computed adaptively to satisfy the CFL condition with safety factor $\nu_{\text{CFL}} = 0.4$:

$$\Delta t = \nu_{\text{CFL}} \cdot \min\left(\frac{\Delta x}{\max_i |u_i|}, \frac{\Delta x^2}{2\nu}\right), \tag{37}$$

where the first term ensures convective stability and the second term ensures diffusive stability. The timestep is recomputed at every substep to adapt to the instantaneous solution state.

Simulation Protocol. Each trajectory evolves from $t = 0$ to $T_{\text{final}} = 0.1$ time units. During this period, $n_{\text{snapshots}} = 1001$ snapshots (including the initial state) are saved at uniform intervals $\Delta t_{\text{outer}} = T_{\text{final}}/(n_{\text{snapshots}} - 1) = 10^{-4}$. A total of 64 trajectories are generated with different random initial phases. The fine-grid snapshots at resolution $N = 2048$ are coarsened to multiple resolutions using two methods: (i) spectral truncation, which retains the lowest N_{coarse} Fourier modes and performs inverse FFT with renormalization, and (ii) direct subsampling, which selects every r-th grid point. For instance, with resample factor $r = 8$, spectral truncation yields $N = 256$ while preserving large-scale structures, whereas subsampling produces the same resolution but may introduce aliasing. The adaptive timestep strategy results in typical internal timesteps in the range $\Delta t \approx 10^{-5}$ to 10^{-4}, requiring approximately 10^3 to 10^4 integration steps per trajectory depending on the instantaneous maximum velocity (typically $\max|u| \approx 2$ to 4 during the decay phase).

A.1.4 Forcing Navier–Stokes: Kolmogorov Flow

Governing Equations. We consider the two-dimensional incompressible Navier–Stokes equations with external forcing on a periodic domain:

$$\frac{\partial \mathbf{v}}{\partial t} + (\mathbf{v} \cdot \nabla)\mathbf{v} = -\nabla p + \nu \nabla^2 \mathbf{v} + \mathbf{f}(x, y), \quad (x, y) \in [0, 2\pi] \times [0, 2\pi], \quad t > 0, \tag{38}$$

$$\nabla \cdot \mathbf{v} = 0, \tag{39}$$

where $\mathbf{v} = (u, v)$ is the velocity field, p is the kinematic pressure, ν is the kinematic viscosity, and $\mathbf{f}$ is the external forcing. We simulate two Reynolds number regimes: Re $= 1000$ with $\nu = 10^{-3}$ and Re $= 4000$ with $\nu = 2.5 \times 10^{-4}$, corresponding to moderate and high turbulence intensity respectively. Periodic boundary conditions are enforced in both directions: $\mathbf{v}(0, y, t) = \mathbf{v}(2\pi, y, t)$ and $\mathbf{v}(x, 0, t) = \mathbf{v}(x, 2\pi, t)$ for all x, y, t.

Initial and Forcing Conditions. The initial velocity field is generated as a filtered random field with maximum amplitude $\|\mathbf{v}\|_{\max} = 7.0$, ensuring a well-defined turbulent state from the onset. A warmup phase of $T_{\text{warmup}} = 40.0$ time units is applied to allow the flow to reach a statistically stationary turbulent state before recording snapshots. The external forcing consists of two components: (i) Kolmogorov forcing $\mathbf{f}_K = (F_0 \sin(k_f y), 0)$ with $F_0 = 1.0$ and $k_f = 4$, which drives energy injection at large scales, and (ii) linear damping $\mathbf{f}_L = \alpha \mathbf{v}$ with $\alpha = -0.1$, which removes energy at the largest scales to prevent unrealistic energy accumulation. The combined forcing $\mathbf{f} = \mathbf{f}_K + \mathbf{f}_L$ maintains a statistically steady turbulent cascade with energy injection at intermediate wavenumbers and dissipation at both large scales (via linear damping) and small scales (via viscosity).

Spatial Discretization. The equations are discretized on an Arakawa C-grid with resolution $N \times N = 512 \times 512$ cells and uniform grid spacing $\Delta x = \Delta y = 2\pi/512$. Velocity components are staggered: the u-component is stored at cell faces with offset $(1, 0.5)$ and the v-component at faces with offset $(0.5, 1)$, while pressure is stored at cell centers with offset $(0.5, 0.5)$. The convective term $(\mathbf{v} \cdot \nabla)\mathbf{v}$ is computed using the van Leer flux-limiting scheme with TVD property, which applies second-order accurate upwind-biased reconstruction in smooth regions and reverts to first-order upwinding near sharp gradients to maintain monotonicity. The diffusive term $\nu \nabla^2 \mathbf{v}$ is computed using second-order centered finite differences. The incompressibility constraint $\nabla \cdot \mathbf{v} = 0$ is enforced via a fractional-step pressure projection method: after advancing the momentum equation, the velocity field is projected onto the divergence-free subspace by solving the Poisson equation $\nabla^2 \phi = \nabla \cdot \mathbf{v}^*$ for the pressure correction ϕ, then updating $\mathbf{v}^{n+1} = \mathbf{v}^* - \nabla \phi$. The Poisson equation is solved using the fast diagonalization method, which exploits separability of the Laplacian operator to reduce the problem to a sum of one-dimensional eigenvalue problems, achieving $O(N^2 \log N)$ complexity for the 2D case.

Temporal Discretization. Time integration employs a first-order explicit-implicit splitting scheme: the convective and forcing terms are treated explicitly using forward Euler, while the diffusive term is treated implicitly to avoid the restrictive diffusive CFL constraint. The timestep is chosen adaptively based on the CFL condition for advection with safety factor $C_{\max} = 0.5$:

$$\Delta t = C_{\max} \cdot \frac{\Delta x}{\|\mathbf{v}\|_{\max}}, \tag{40}$$

where $\|\mathbf{v}\|_{\max} = 7.0$ is the maximum velocity magnitude observed in the initial condition. This yields $\Delta t = 0.5 \times (2\pi/512)/7.0 \approx 5.59 \times 10^{-4}$. The resulting CFL number CFL $= \|\mathbf{v}\|_{\max}\Delta t/\Delta x = 0.5$ is maintained throughout the simulation. Each time integration step consists of three substeps: (1) explicit momentum update $\mathbf{v}^* = \mathbf{v}^n + \Delta t[-(\mathbf{v}^n \cdot \nabla)\mathbf{v}^n + \mathbf{f}]$, (2) pressure projection to enforce incompressibility $\mathbf{v}^{**} = \mathcal{P}(\mathbf{v}^*)$, and (3) implicit diffusion solve $(I - \nu\Delta t\nabla^2)\mathbf{v}^{n+1} = \mathbf{v}^{**}$, where $\mathcal{P}$ denotes the projection operator onto divergence-free fields.

Simulation Protocol. Each trajectory begins from a random filtered initial condition and undergoes a warmup phase of $T_{\text{warmup}} = 40.0$ time units to reach statistically stationary turbulence. After warmup, snapshots are recorded for a production duration of $T_{\text{production}} \approx 0.0448$ time units. The production phase uses inner batching with $n_{\text{inner}} = 8$ timesteps per saved frame and $n_{\text{outer}} = 1000$ saved frames, yielding a total of $n_{\text{snapshots}} = 1001$ snapshots (including the initial state at $t = 0$ of the production phase) saved at uniform intervals $\Delta t_{\text{frame}} = 8 \times \Delta t \approx 4.47 \times 10^{-3}$. A total of 64 trajectories are generated with different random initial conditions. The fine-grid snapshots at resolution 512×512 are coarsened to multiple resolutions via the staggered velocity downsampling method, which preserves the divergence-free property: for each velocity component, only values lying on coarse-grid control volume faces are retained, then averaged over the appropriate face area. For instance, with resample factor $r = 16$, the snapshots are downsampled to 32×32, yielding the coarse-grid reference states $\bar{\mathbf{v}}_\Delta$ used for subgrid-scale closure modeling. This conservative downsampling ensures that coarse-grained fields satisfy $\nabla \cdot \bar{\mathbf{v}}_\Delta = 0$ exactly when the fine-grid field is divergence-free. The CFL-based adaptive timestep strategy maintains numerical stability throughout the simulation, with the timestep scaling inversely with grid resolution to keep the CFL number constant across different spatial resolutions.

A.1.5 Decaying Navier–Stokes Turbulence

Governing Equations. We consider the two-dimensional incompressible Navier–Stokes equations without external forcing on a periodic domain:

$$\frac{\partial \mathbf{v}}{\partial t} + (\mathbf{v} \cdot \nabla)\mathbf{v} = -\nabla p + \nu\nabla^2\mathbf{v}, \quad (x, y) \in [0, 2\pi] \times [0, 2\pi], \quad t > 0, \tag{41}$$

$$\nabla \cdot \mathbf{v} = 0, \tag{42}$$

where $\mathbf{v} = (u, v)$ is the velocity field, p is the kinematic pressure, and $\nu = 10^{-3}$ is the kinematic viscosity, corresponding to an initial Reynolds number $\text{Re} = 1000$. The absence of external forcing results in free decay of kinetic energy, where turbulent structures evolve under the competing effects of nonlinear energy transfer and viscous dissipation. Periodic boundary conditions are enforced in both directions: $\mathbf{v}(0, y, t) = \mathbf{v}(2\pi, y, t)$ and $\mathbf{v}(x, 0, t) = \mathbf{v}(x, 2\pi, t)$ for all x, y, t. This configuration serves as a canonical test case for subgrid-scale (SGS) closure modeling in non-stationary flows, where the effective Reynolds number decreases over time as kinetic energy dissipates.

Initial and Forcing Conditions. The initial velocity field is generated as a filtered random field with maximum amplitude $\|\mathbf{v}\|_{\max} = 4.2$, producing a turbulent state with energy concentrated at intermediate wavenumbers. A warmup phase of $T_{\text{warmup}} = 4.5$ time units is applied to allow transient instabilities to decay and the flow to settle onto a smooth decay trajectory, ensuring that the recorded production phase represents well-resolved decaying turbulence rather than initialization artifacts. No external forcing is applied ($\mathbf{f} = \mathbf{0}$), so the total kinetic energy $E(t) = \frac{1}{2}\int |\mathbf{v}|^2 \, dx \, dy$ decreases monotonically over time due to viscous dissipation. This decay process is characterized by an inverse energy cascade at large scales and forward enstrophy cascade at small scales, typical of two-dimensional turbulence. The initial Reynolds number $\text{Re}_0 = 1000$ based on the initial maximum velocity and viscosity gradually decreases as the flow decays, providing a time-varying testbed for closure models.

Spatial Discretization. The spatial discretization is identical to the forced Kolmogorov flow case: an Arakawa C-grid with resolution $N \times N = 512 \times 512$ cells and uniform grid spacing $\Delta x = \Delta y = 2\pi/512$. Velocity components are staggered with the u-component at offset $(1, 0.5)$ and the v-component at offset $(0.5, 1)$, while pressure resides at cell centers with offset $(0.5, 0.5)$. The convective term $(\mathbf{v} \cdot \nabla)\mathbf{v}$ is computed using the van Leer flux-limiting scheme with TVD property, ensuring second-order accuracy in smooth regions and monotonicity preservation near gradi-

ents. The diffusive term $\nu\nabla^2\mathbf{v}$ is computed using second-order centered finite differences. Incompressibility $\nabla \cdot \mathbf{v} = 0$ is enforced via fractional-step pressure projection: the intermediate velocity $\mathbf{v}^*$ is projected onto the divergence-free subspace by solving the Poisson equation $\nabla^2\phi = \nabla \cdot \mathbf{v}^*$ for the pressure correction ϕ, then updating $\mathbf{v}^{n+1} = \mathbf{v}^* - \nabla\phi$. The Poisson solve employs the fast diagonalization method exploiting the separability of the Laplacian on periodic domains, achieving $O(N^2 \log N)$ complexity.

Temporal Discretization. Time integration employs the same first-order explicit-implicit splitting scheme as the forced case: convective terms are treated explicitly using forward Euler, while diffusive terms are treated implicitly to avoid the restrictive diffusive CFL constraint. The timestep is chosen based on the CFL condition for advection with safety factor $C_{\max} = 0.5$:

$$\Delta t = C_{\max} \cdot \frac{\Delta x}{\|\mathbf{v}\|_{\max}}, \tag{43}$$

where $\|\mathbf{v}\|_{\max} = 4.2$ is the initial maximum velocity magnitude. This yields $\Delta t \approx 1.46 \times 10^{-3}$, with an initial CFL number $\text{CFL}_0 = 0.5$. As the flow decays and $\|\mathbf{v}\|_{\max}$ decreases, the actual CFL number drops below the initial value, ensuring numerical stability throughout the simulation. Each time integration step consists of three substeps: (1) explicit momentum update $\mathbf{v}^* = \mathbf{v}^n + \Delta t[-(\mathbf{v}^n \cdot \nabla)\mathbf{v}^n]$, (2) pressure projection $\mathbf{v}^{**} = \mathcal{P}(\mathbf{v}^*)$ to enforce incompressibility, and (3) implicit diffusion solve $(I - \nu\Delta t\nabla^2)\mathbf{v}^{n+1} = \mathbf{v}^{**}$, where $\mathcal{P}$ denotes the projection operator onto divergence-free fields.

Simulation Protocol. Each trajectory begins from a random filtered initial condition and undergoes a warmup phase of $T_{\text{warmup}} = 4.5$ time units to allow transient dynamics to settle. After warmup, snapshots are recorded for a production duration of $T_{\text{production}} \approx 0.0117$ time units. The production phase uses inner batching with $n_{\text{inner}} = 8$ timesteps per saved frame and $n_{\text{outer}} = 1000$ saved frames, yielding a total of $n_{\text{snapshots}} = 1001$ snapshots (including the initial state at $t = 0$ of the production phase) saved at uniform intervals $\Delta t_{\text{frame}} = 8 \times \Delta t \approx 1.17 \times 10^{-2}$. A total of 64 trajectories are generated with different random initial conditions. The fine-grid snapshots at resolution 512×512 are coarsened to multiple resolutions via the same staggered velocity downsampling method used for forced flows, which preserves the divergence-free property: for each velocity component, only values on coarse-grid control volume faces are retained and averaged. For instance, with resample factor $r = 16$, the snapshots are downsampled to 32×32, yielding coarse-grid reference states $\bar{\mathbf{v}}_\Delta$ for subgrid-scale closure modeling. During the production phase, typical kinetic energy decay ranges from 5% to 15% depending on the trajectory, with the effective Reynolds number decreasing correspondingly. This decaying turbulence configuration provides a complementary testbed to the statistically stationary forced case, allowing assessment of closure model performance in time-evolving flows where the underlying turbulence statistics are non-stationary.

A.2 Model Specifications

A.2.1 Models for 1D Benchmarks

SINO for 1D Benchmarks. We employ a Scale-Invariant Neural Operator (SINO) architecture that operates on coarse-resolution inputs of shape (N, C_{in}) where N is spatial resolution and C_{in} is the number of input channels. The model consists of a lifting layer that projects inputs to hidden dimension $D = 16$, followed by $L = 2$ SINO blocks, and a projection layer that maps back to output channels. Each SINO block comprises dual branches: a frequency branch that applies spectral convolution via MLP-generated complex filters in Fourier space, and a spatial branch that performs circular convolution with MLP-generated kernels of size $k = 9$. Both branches use coordinate-based MLPs with two hidden layers of width $W_{\text{freq}} = 8$ and $W_{\text{spatial}} = 8$ respectively, controlling the expressiveness of frequency and spatial representations. The penultimate layers of these MLPs have dimensions $N_{\text{freq}} = 2$ and $N_{\text{spatial}} = 2$. Features from both branches are concatenated and fused via a linear layer, with residual connections across blocks. Physical normalization is applied to both spatial coordinates and spectral frequencies to ensure scale invariance across different resolutions. The output layer uses small initialization (standard deviation 0.01) to ensure training stability in early iterations. All activations use ReLU, and all spatial convolutions respect periodic boundary conditions through circular padding.

UNet for 1D Benchmarks. We employ a two-level U-Net architecture for learning subgrid-scale corrections in one-dimensional problems. The network operates on input feature maps of shape (C_{in}, N) where C_{in} denotes the number of input channels (typically 1 for velocity u) and N is the spatial resolution. The encoder path downsamples the input twice via max pooling (factor of 2 each), progressively increasing channels from C_{in} to $C_{\text{max}}/4$ to $C_{\text{max}}/2$, with each encoder stage consisting of two convolutions (kernel size $k = 9$) followed by ReLU activations. The bottleneck layer operates at resolution $N/4$ with $C_{\text{max}} = 16$ channels. The decoder path mirrors the encoder with transposed convolutions for upsampling (stride 2, kernel size 2) and skip connections that concatenate encoder features before each decoder stage. All convolutions use circular padding to respect periodic boundary conditions. A final 1×1 convolution maps to a single output channel (the correction term τ) with small initialization (standard deviation 0.01) to ensure training stability in early iterations. The architecture follows the channel progression $1 \to C_{\text{max}}/4 \to C_{\text{max}}/2 \to C_{\text{max}} \to C_{\text{max}}/2 \to C_{\text{max}}/4 \to 1$ and spatial progression $N \to N/2 \to N/4 \to N/2 \to N$, requiring N divisible by 4.

DeepONet for 1D Benchmarks. We employ a Deep Operator Network (DeepONet) that learns operator mappings through a decomposed architecture separating input functions from output coordinates. The model consists of two parallel subnetworks: a branch network that processes input function samples $\mathbf{u} \in \mathbb{R}^m$ at $m = 32$ fixed sensor locations, and a trunk network that processes output coordinates $\mathbf{y}$ augmented with Fourier features. Both networks are MLPs with hidden dimension $D = 32$, comprising $L_{\text{branch}} = 4$ and $L_{\text{trunk}} = 4$ layers respectively, using hyperbolic tangent activations. The trunk network input is enhanced with $n_{\text{Fourier}} = 4$ frequency terms, forming a feature vector $[\mathbf{y}/L, 1, \cos(2\pi\mathbf{y}/L), \sin(2\pi\mathbf{y}/L), \ldots, \cos(8\pi\mathbf{y}/L), \sin(8\pi\mathbf{y}/L)]$ of dimension $1 + 1 + 2 \times 4 = 9$, where L denotes the domain length. The branch network maps sensor readings to feature space $\mathbf{b} = \phi_{\text{branch}}(\mathbf{u}) \in \mathbb{R}^p$ with $p = 32$ features, while the trunk network independently maps augmented coordinates to $\mathbf{t}(\mathbf{y}) = \phi_{\text{trunk}}(\tilde{\mathbf{y}}) \in \mathbb{R}^p$. The final output is computed as the inner product $G(\mathbf{u})(\mathbf{y}) = \langle \mathbf{b}, \mathbf{t}(\mathbf{y}) \rangle + b_0$, where b_0 is a learnable bias. The output layer uses small initialization (standard deviation 0.01) to ensure training stability in early iterations. This architecture enables resolution-independent predictions and naturally supports periodic boundary conditions through Fourier coordinate encoding.

Transolver for 1DBenchmarks. We employ a Transformer-based operator network (Transolver) that incorporates physics-aware attention mechanisms for learning PDE solutions. The model operates on input feature maps of shape (C_{in}, N) where $C_{\text{in}} = 2$ includes the velocity field and spatial coordinates, and N is the spatial resolution. The architecture begins with a lifting layer that maps inputs to hidden dimension $D = 16$, followed by $L = 2$ Transolver blocks. Each block implements a three-step physics attention mechanism: (1) *Slice* adaptively groups N grid points into $G = 32$ learnable slice tokens via temperature-scaled softmax weights, aggregating local features into macroscopic patterns; (2) *Self-Attention* applies multi-head attention ($H = 8$ heads) among slice tokens to capture global interactions, where query, key, and value projections operate on head dimension $d_h = D/H = 2$; (3) *Deslice* redistributes attended slice token information back to original grid points using the same grouping weights. Each block includes residual connections and a feed-forward network with expansion ratio $r = 4$ (hidden dimension $4D = 64$), using GELU activations and layer normalization. The final projection layers map from hidden dimension through an intermediate layer of 128 channels to the output dimension $C_{\text{out}} = 1$. The output layer uses small initialization (standard deviation 0.01) to ensure training stability in early iterations. This architecture enables adaptive spatial receptive fields through learned grouping while maintaining global awareness through attention mechanisms.

OFormer for 1D Benchmarks. We employ an Operator Transformer (OFormer) that learns operator mappings through a four-stage encoder-cross-decoder architecture with Galerkin-type linear attention. The model processes input functions $\mathbf{u}(\mathbf{x})$ sampled at N points with feature dimension C_{in} and outputs predictions at arbitrary query locations. The architecture begins with an encoder that lifts inputs to hidden dimension $D = 16$ via a linear layer with GELU activation. The encoded features $\mathbf{z}_{\text{enc}}$ are then processed by a cross-attention module that extracts relevant information from input locations to query points using $H = 8$ attention heads with head dimension $d_h = D/H = 2$. This cross-attention employs rotary position embeddings (RoPE) with minimum frequency $1/64$ to encode spatial relationships, and applies Galerkin-type normalization through instance normalization

across spatial dimensions for both keys and values. The attended features $\mathbf{z}_{\text{cross}}$ are refined through $L = 2$ propagator layers, each consisting of pre-normalized linear self-attention with residual connections followed by a feed-forward network with hidden dimension $2D = 32$ and GELU activation. Linear attention achieves $\mathcal{O}(N)$ complexity by computing $\mathbf{Q}(\mathbf{K}^\top\mathbf{V})$ instead of $\text{softmax}(\mathbf{Q}\mathbf{K}^\top)\mathbf{V}$, with scaling factor $1/d_h$ for numerical stability. The decoder projects refined features from hidden dimension to output channels C_{out} using a linear layer with small initialization (standard deviation 0.01) to ensure training stability in early iterations. RoPE enables the model to generalize to arbitrary query point distributions while maintaining spatial awareness through frequency-based position encoding.

Galerkin Transformer for 1D Benchmarks. We employ a Galerkin Transformer that learns integral operators through linear attention mechanisms with $\mathcal{O}(N)$ computational complexity. The model processes input physical quantities $\mathbf{u}(\mathbf{x})$ sampled at N spatial locations with feature dimension C_{in}. The architecture begins with a lifting layer that maps inputs pointwise to hidden dimension $D = 16$ via a linear transformation. The lifted features are then refined through $L = 2$ encoder layers, each implementing a Galerkin attention mechanism followed by a position-wise feed-forward network with residual connections and layer normalization. The Galerkin attention employs $H = 8$ heads with head dimension $d_h = D/H = 2$, computing linear attention as $\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \mathbf{Q}(\mathbf{K}^\top\mathbf{V})/N$, where the matrix product $\mathbf{K}^\top\mathbf{V} \in \mathbb{R}^{d_h \times d_h}$ aggregates global information independently of sequence length. This formulation avoids the quadratic complexity of softmax-based attention while maintaining global receptive fields across all spatial locations. Each encoder layer includes a feed-forward network with hidden dimension $D_{\text{FFN}} = 512$ and ReLU activation. The refined features are decoded through a pointwise regressor consisting of two linear layers with ReLU activation, projecting from hidden dimension through an intermediate layer of dimension D to output channels C_{out}. The output layer uses small initialization (standard deviation 0.01) to ensure training stability in early iterations. This architecture enables efficient global information propagation while maintaining interpretability through its connection to Galerkin projection methods in numerical PDEs.

Fourier Neural Operator for 1D Benchmarks. We employ a Fourier Neural Operator (FNO) that learns integral operators in the frequency domain, providing resolution-invariant and globally receptive mappings. The model processes input functions $\mathbf{u}(\mathbf{x})$ augmented with normalized spatial coordinates, forming a feature vector with dimension $C_{\text{in}} = 2$ at N grid points. The architecture begins with a lifting layer that pointwise projects inputs to hidden dimension $D = 16$ via a linear transformation. The lifted features are then refined through $L = 2$ Fourier layers, each implementing a parallel combination of spectral and spatial branches. The spectral branch computes $\mathcal{K}(\mathbf{v}) = \mathcal{F}^{-1}(\mathbf{R}_\ell \cdot \mathcal{F}(\mathbf{v})_{|k<K})$, where $\mathcal{F}$ denotes the real-valued Fast Fourier Transform (FFT), $\mathbf{R}_\ell \in \mathbb{C}^{D\times D\times K}$ are learnable complex weights with $K = 16$ preserved low-frequency modes, and high-frequency components beyond mode K are implicitly truncated to zero, providing spectral regularization. This truncation restricts the learned operator to spatial scales larger than L/K, where L is the domain length. The spatial branch applies a 1×1 convolution (pointwise linear transformation) $\mathbf{W}_\ell \in \mathbb{R}^{D\times D}$ to capture local residual corrections. Each Fourier layer computes $\mathbf{v}_{\ell+1} = \sigma(\mathbf{W}_\ell\mathbf{v}_\ell + \mathcal{K}_\ell(\mathbf{v}_\ell))$, where GELU activation σ is applied to the summed output for all but the final layer. The refined features are decoded through a two-layer pointwise MLP with fixed intermediate dimension 128: the output projection applies $D \to 128 \to C_{\text{out}}$ with GELU activation between layers and small initialization (standard deviation 0.01) on the final layer for training stability. The frequency-domain formulation achieves global receptive fields covering the entire spatial domain in each layer, while the modes parameter K controls the frequency bandwidth, balancing expressive power against overfitting to high-frequency noise.

U-Net Enhanced Fourier Neural Operator for 1D Benchmarks. We employ a U-Net enhanced Fourier Neural Operator (UFNO) that combines the global receptive fields of FNO with the local multiscale feature extraction capabilities of U-Net. The model processes input functions $\mathbf{u}(\mathbf{x})$ augmented with normalized spatial coordinates, forming feature vectors with dimension $C_{\text{in}} = 2$ at N grid points. The architecture begins with a lifting layer that pointwise projects inputs to hidden dimension $D = 16$ via a linear transformation. The lifted features are refined through $L = 2$ UFNO layers, each implementing a three-branch parallel architecture: $\mathbf{v}_{\ell+1} = \sigma(\mathcal{K}_\ell(\mathbf{v}_\ell) + \mathbf{W}_\ell(\mathbf{v}_\ell) + \mathcal{U}_\ell(\mathbf{v}_\ell))$. The spectral branch $\mathcal{K}_\ell$ computes frequency-domain con-

volutions $\mathcal{F}^{-1}(\mathbf{R}_\ell \cdot \mathcal{F}(\mathbf{v})_{|k<K})$ with $K = 16$ preserved modes, capturing global dependencies through truncated Fourier transforms. The spatial branch $\mathbf{W}_\ell$ applies 1×1 convolutions for channel mixing and local residuals. The U-Net branch $\mathcal{U}_\ell$ implements a lightweight encoder-decoder architecture with one downsampling layer (factor 2), circular padding for periodic boundary compatibility, and skip connections that preserve multiscale spatial information from resolution N to $N/4$, enabling fine-grained local feature refinement. We employ a staged activation strategy controlled by parameter $\ell_{\text{start}} = 1$: layers $\ell < \ell_{\text{start}}$ use only spectral and spatial branches (FNO-only mode) for rapid coarse feature extraction, while layers $\ell \geq \ell_{\text{start}}$ incorporate the U-Net branch for detailed multiscale refinement, balancing computational efficiency with expressive power. GELU activation σ is applied to all but the final layer. The refined features are decoded through a two-layer pointwise MLP projecting from hidden dimension through intermediate dimension 128 to output channels C_{out} with GELU activation and small initialization (standard deviation 0.01). The U-Net refiner adds approximately $3D^2$ parameters per activated layer, while kernel size $k = 9$ controls its local receptive field. This hybrid architecture combines FNO's frequency-domain global propagation with U-Net's spatial-domain multiscale analysis, enabling simultaneous capture of long-range dependencies and localized structures.

Amortized Fourier Neural Operator for 1D Benchmarks. We employ an Amortized Fourier Neural Operator (AMFNO) that dynamically generates frequency-domain convolution kernels through multi-layer perceptrons, enabling frequency-adaptive PDE solving. The model processes input functions $\mathbf{u}(\mathbf{x})$ at N grid points, automatically augmenting them with normalized spatial coordinates to form feature vectors of dimension $C_{\text{in}} + 1 = 2$. The architecture begins with a lifting layer that pointwise projects augmented inputs to hidden dimension $D = 16$ via a linear transformation. The lifted features are refined through $L = 2$ AMFNO layers, each implementing a dual-branch architecture with residual connections: $\mathbf{v}_{\ell+1} = \mathbf{v}_\ell + \mathcal{K}_{\text{MLP}}(\mathbf{v}_\ell) + \mathbf{W}_{\text{MLP}}(\mathbf{v}_\ell)$, where GELU activation $\sigma(\cdot)$ is applied to the summed output for all but the final layer. The spectral branch $\mathcal{K}_{\text{MLP}}$ replaces the fixed complex weights of standard FNO with dynamically generated kernels: for each frequency mode ω_k in the discrete Fourier spectrum, we encode the normalized frequency coordinate as a low-dimensional feature representation. Two separate MLPs with hidden dimension 4 then map this frequency encoding to real and imaginary components of the convolution kernel: $K(\omega) = \text{MLP}_r(\omega) + i \cdot \text{MLP}_i(\omega) \in \mathbb{C}^{D \times D}$, producing mode-specific transformations across all $N/2 + 1$ frequencies in the real FFT spectrum. This dynamically generated kernel performs frequency-domain convolution $\mathcal{F}^{-1}(K(\omega) \cdot \mathcal{F}(\mathbf{v}))$, where $\mathcal{F}$ denotes the real-valued Fast Fourier Transform. The spatial branch $\mathbf{W}_{\text{MLP}}$ applies a two-layer pointwise MLP with hidden dimension $4D = 64$ and GELU activation for local feature mixing. The refined features are decoded through a single linear projection from hidden dimension D to output channels $C_{\text{out}} = 1$. This frequency-adaptive kernel generation enables superior resolution generalization compared to fixed-kernel FNO, where each frequency mode receives a tailored transformation learned from data rather than using predetermined spectral truncation.

A.2.2 Models for 2D Benchmarks

SINO for 2D Benchmarks. We employ a Scale-Invariant Neural Operator (SINO) architecture that operates on coarse-resolution inputs of shape (H, W, C_{in}) where H and W are spatial resolutions and C_{in} is the number of input channels. The model consists of a lifting layer that projects inputs to hidden dimension $D = 32$, followed by $L = 4$ SINO blocks, and a projection layer that maps back to output channels. Each SINO block comprises dual branches: a frequency branch that applies spectral convolution via MLP-generated complex filters in Fourier space using the 2D real FFT (rfft2), and a spatial branch that performs circular convolution with MLP-generated kernels of size $k = 9 \times 9$. Both branches use coordinate-based MLPs with two hidden layers of width $w_{\text{freq}} = 32$ and $w_{\text{spatial}} = 32$ respectively, controlling the expressiveness of frequency and spatial representations. The penultimate layers of these MLPs have dimensions $n_{\text{freq}} = 2$ and $n_{\text{spatial}} = 2$, forming bottleneck layers that induce low-rank parameterization. For the frequency branch, physically normalized frequency coordinates $(\omega_y, \omega_x) \in [-1, 1]^2$ are computed by mapping discrete wavenumbers to normalized ranges for each mode in the half-spectrum: $\xi_y = 4k_y/H - 1$ spanning the full Y-spectrum and $\xi_x = 4k_x/W - 1$ covering the non-negative X-half. The MLP generates mode-specific complex filters $K(\omega) = \text{MLP}_r(\omega) + i \cdot \text{MLP}_i(\omega) \in \mathbb{C}^{D \times D}$ across all $H \times (W/2 + 1)$ frequencies, with output normalization by $1/\sqrt{D}$ and percentile-based clipping for numerical stability. A bypass connection via a separate linear layer is added to the spectral output. For the spatial

branch, normalized relative displacement coordinates $\zeta = (\zeta_y, \zeta_x) \in [-1, 1]^2$ are generated by mapping kernel offsets to normalized ranges independently in both dimensions, producing position-dependent convolution weights that respect periodic boundary conditions through circular padding. Features from both branches are concatenated and fused via a linear layer, with residual connections scaled by 0.1 across blocks to prevent gradient explosion: $h^{(\ell+1)} = h^{(\ell)} + 0.1 \cdot W^{(\ell)}_{\text{fuse}}[h^{(\ell)}_{\text{freq}}\,;\, h^{(\ell)}_{\text{spatial}}]$. Physical normalization is applied to both spatial coordinates and spectral frequencies to ensure scale invariance across different resolutions. The output layer uses small initialization (standard deviation 0.01) to ensure training stability in early iterations. All activations use ReLU, and all spatial convolutions respect periodic boundary conditions through circular padding.

UNet for 2D Benchmarks. We employ a two-level U-Net architecture for learning subgrid-scale corrections in two-dimensional problems. The network operates on input feature maps of shape (C_{in}, H, W) where C_{in} denotes the number of input channels (typically 2 for velocity components $[u, v]$), H is the height resolution, and W is the width resolution. The encoder path downsamples the input twice via max pooling (factor of 2 each), progressively increasing channels from C_{in} to C_{init} to $C_{\text{init}} \times 2$, with each encoder stage consisting of two convolutions (kernel size $k = 9$) followed by ReLU activations. The bottleneck layer operates at resolution $(H/4, W/4)$ with $C_{\text{max}} = 16$ channels. The decoder path mirrors the encoder with transposed convolutions for upsampling (stride 2, kernel size 2) and skip connections that concatenate encoder features before each decoder stage. All convolutions use circular padding to respect periodic boundary conditions. A final 1×1 convolution maps to output channels (the correction term $[\tau_u, \tau_v]$) with small initialization (standard deviation 0.01) to ensure training stability in early iterations. The architecture follows the channel progression $2 \to C_{\text{init}} \to C_{\text{init}} \times 2 \to C_{\text{max}} \to C_{\text{init}} \times 2 \to C_{\text{init}} \to 2$ and spatial progression $(H, W) \to (H/2, W/2) \to (H/4, W/4) \to (H/2, W/2) \to (H, W)$, requiring both H and W divisible by 4.

DeepONet for 2D Benchmarks. We employ the same Deep Operator Network (DeepONet) architecture as in the 1D case, with input and output dimensions modified to accommodate two-dimensional velocity fields. The model processes input functions $[u, v]$ sampled on a fixed sensor grid of shape $(N_x, N_y) = (32, 32)$, flattened to dimension $m = 32 \times 32 \times 2 = 2048$ for the branch network. The trunk network processes two-dimensional output coordinates (x, y) augmented with isotropic Fourier features of dimension $3 + 4 \times n_{\text{Fourier}} = 19$, where $n_{\text{Fourier}} = 4$. Both branch and trunk networks remain MLPs with hidden dimension $D = 32$, $L_{\text{branch}} = 4$ and $L_{\text{trunk}} = 4$ layers, and hyperbolic tangent activations. The branch network maps sensor readings to feature space $\mathbf{b} = \phi_{\text{branch}}(\mathbf{u}) \in \mathbb{R}^p$ with $p = 32$ features, while the trunk network maps augmented coordinates to $\mathbf{t}(x, y) = \phi_{\text{trunk}}(\tilde{x}, \tilde{y}) \in \mathbb{R}^p$. The final output is computed as $G(\mathbf{u})(x, y) = \langle \mathbf{b}, \mathbf{t}(x, y) \rangle + b_0$ with small initialization (standard deviation 0.01) on the output layer. The 2D Fourier encoding naturally supports periodic boundary conditions in both spatial directions while maintaining resolution-independent predictions.

Transolver for 2D Benchmarks. We employ the same Transformer-based operator network (Transolver) architecture as in the 1D case, with input feature maps of shape (C_{in}, H, W) where $C_{\text{in}} = 4$ includes the two-dimensional velocity field $[u, v]$ and normalized spatial coordinates $[x/L_x, y/L_y]$. The architecture begins with a lifting layer that maps inputs to hidden dimension $D = 32$, followed by $L = 4$ Transolver blocks. Each block implements the same three-step physics attention mechanism adapted for 2D grids: (1) *Slice* adaptively groups $N = H \times W$ grid points into $G = 32$ learnable slice tokens via temperature-scaled softmax weights ($\tau = 0.5$), aggregating local features into macroscopic patterns; (2) *Self-Attention* applies multi-head attention ($H = 8$ heads) among slice tokens with head dimension $d_h = D/H = 4$; (3) *Deslice* redistributes attended information back to original grid points using the same grouping weights. We use linear projections for pointwise feature transformations, ensuring applicability to arbitrary geometries while maintaining parameter efficiency. Each block includes residual connections and a feed-forward network with expansion ratio $r = 4$ (hidden dimension $4D = 128$), using GELU activations and layer normalization. The final projection layers map from hidden dimension through an intermediate layer of 128 channels to output dimension $C_{\text{out}} = 2$, with small initialization (standard deviation 0.01) on the output layer. This architecture enables adaptive spatial receptive fields through learned grouping while maintaining global awareness through attention mechanisms.

OFormer for 2D Benchmarks. We employ the same Operator Transformer (OFormer) architecture as in the 1D case, implementing a four-stage encoder-cross-decoder framework with Galerkin-type linear attention for two-dimensional spatial domains. The model processes input functions $\mathbf{u}(\mathbf{x})$ sampled at $N = H \times W$ points with feature dimension $C_{\text{in}} = 4$ (including velocity field $[u, v]$ and normalized coordinates $[x/L_x, y/L_y]$) and outputs predictions at arbitrary query locations in 2D space. The encoder lifts inputs to hidden dimension $D = 32$ via a linear layer with GELU activation. The encoded features $\mathbf{z}_{\text{enc}}$ are then processed by a cross-attention module that extracts relevant information from input locations to query points using $H = 8$ attention heads with head dimension $d_h = D/H = 4$. This cross-attention employs two-dimensional rotary position embeddings (RoPE) that apply separate frequency-based encodings to x and y coordinates with scale factor 16.0 and minimum frequency $1/64$, naturally extending the 1D rotation mechanism to capture spatial relationships in the plane. Galerkin-type normalization is applied through instance normalization across spatial dimensions for both keys and values. The attended features $\mathbf{z}_{\text{cross}}$ are refined through $L = 4$ propagator layers, each consisting of pre-normalized linear self-attention with residual connections followed by a feed-forward network with hidden dimension $2D = 64$ and GELU activation. Linear attention achieves $\mathcal{O}(N)$ complexity by computing $\mathbf{Q}(\mathbf{K}^\top\mathbf{V})$ instead of $\text{softmax}(\mathbf{QK}^\top)\mathbf{V}$, with scaling factor $1/d_h$ for numerical stability. The decoder projects refined features from hidden dimension to output channels $C_{\text{out}} = 2$ using a linear layer with small initialization (standard deviation 0.01). The 2D RoPE extension enables the model to generalize to arbitrary query point distributions in the plane while maintaining spatial awareness through frequency-based position encoding in both spatial directions.

Galerkin Transformer for 2D Benchmarks. We employ the same Galerkin Transformer architecture as in the 1D case, learning integral operators through linear attention mechanisms with $\mathcal{O}(N)$ computational complexity for two-dimensional spatial domains. The model processes input physical quantities $\mathbf{u}(\mathbf{x})$ sampled at $N = H \times W$ spatial locations with feature dimension $C_{\text{in}} = 4$ (including velocity field $[u, v]$ and normalized coordinates $[x/L_x, y/L_y]$). The 2D grid is flattened into a sequence of length N for processing. The architecture begins with a lifting layer that maps inputs pointwise to hidden dimension $D = 32$ via a linear transformation. The lifted features are then refined through $L = 4$ encoder layers, each implementing a Galerkin attention mechanism followed by a position-wise feed-forward network with residual connections and layer normalization. The Galerkin attention employs $H = 8$ heads with head dimension $d_h = D/H = 4$, computing linear attention as $\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \mathbf{Q}(\mathbf{K}^\top\mathbf{V})/N$, where the matrix product $\mathbf{K}^\top\mathbf{V} \in \mathbb{R}^{d_h \times d_h}$ aggregates global information independently of sequence length. This formulation avoids the quadratic complexity of softmax-based attention while maintaining global receptive fields across all spatial locations in the 2D domain. Each encoder layer includes a feed-forward network with hidden dimension $D_{\text{FFN}} = 512$ and ReLU activation. The refined features are decoded through a pointwise regressor consisting of two linear layers with ReLU activation, projecting from hidden dimension through an intermediate layer of dimension D to output channels $C_{\text{out}} = 2$. The output layer uses small initialization (standard deviation 0.01) to ensure training stability in early iterations. This architecture enables efficient global information propagation across the entire 2D spatial domain while maintaining interpretability through its connection to Galerkin projection methods in numerical PDEs.

Fourier Neural Operator for 2D Benchmarks. We employ a Fourier Neural Operator (FNO) that learns integral operators in the frequency domain, providing resolution-invariant and globally receptive mappings for two-dimensional spatial domains. The model processes input functions $\mathbf{u}(\mathbf{x})$ augmented with normalized spatial coordinates, forming a feature vector with dimension $C_{\text{in}} = 4$ (including velocity field $[u, v]$ and normalized coordinates $[x/L_x, y/L_y]$) at $N = H \times W$ grid points. The architecture begins with a lifting layer that pointwise projects inputs to hidden dimension $D = 32$ via a linear transformation. The lifted features are then refined through $L = 4$ Fourier layers, each implementing a parallel combination of spectral and spatial branches. The spectral branch computes $\mathcal{K}(\mathbf{v}) = \mathcal{F}^{-1}(\mathbf{R}_\ell^{(1)} \cdot \mathcal{F}(\mathbf{v})_{0:K_x,:K_y} + \mathbf{R}_\ell^{(2)} \cdot \mathcal{F}(\mathbf{v})_{-K_x:,:K_y})$, where $\mathcal{F}$ denotes the two-dimensional real-valued Fast Fourier Transform (rfft2), $\mathbf{R}_\ell^{(1)}, \mathbf{R}_\ell^{(2)} \in \mathbb{C}^{D \times D \times K_x \times K_y}$ are learnable complex weights with $(K_x, K_y) = (16, 16)$ preserved low-frequency modes in each spatial dimension, and high-frequency components beyond modes (K_x, K_y) are implicitly truncated to zero, providing spectral regularization. The use of two separate weight matrices $\mathbf{R}_\ell^{(1)}$ and $\mathbf{R}_\ell^{(2)}$ for positive and negative frequencies in the x-direction enables the model to capture asymmetric spatial patterns

while respecting the conjugate symmetry of real-valued signals. This truncation restricts the learned operator to spatial scales larger than L_x/K_x and L_y/K_y in the respective directions. The spatial branch applies a 1×1 convolution (pointwise linear transformation) $\mathbf{W}_\ell \in \mathbb{R}^{D\times D}$ to capture local residual corrections. Each Fourier layer computes $\mathbf{v}_{\ell+1} = \sigma(\mathbf{W}_\ell \mathbf{v}_\ell + \mathcal{K}_\ell(\mathbf{v}_\ell))$, where GELU activation σ is applied to the summed output for all but the final layer. The refined features are decoded through a two-layer pointwise MLP with fixed intermediate dimension 128: the output projection applies $D \to 128 \to C_{\text{out}}$ with GELU activation between layers and small initialization (standard deviation 0.01) on the final layer for training stability. The frequency-domain formulation achieves global receptive fields covering the entire two-dimensional spatial domain in each layer, while the modes parameters (K_x, K_y) control the frequency bandwidth in each direction, balancing expressive power against overfitting to high-frequency noise and enabling anisotropic frequency resolution for problems with directional characteristics.

U-Net Enhanced Fourier Neural Operator for 2D Benchmarks. We employ a U-Net enhanced Fourier Neural Operator (UFNO) that combines the global receptive fields of FNO with the local multiscale feature extraction capabilities of U-Net for two-dimensional spatial domains. The model processes input functions $\mathbf{u}(\mathbf{x})$ augmented with normalized spatial coordinates, forming feature vectors with dimension $C_{\text{in}} = 4$ (including velocity field $[u, v]$ and normalized coordinates $[x/L_x, y/L_y]$) at $N = H \times W$ grid points. The architecture begins with a lifting layer that pointwise projects inputs to hidden dimension $D = 32$ via a linear transformation. The lifted features are refined through $L = 4$ UFNO layers, each implementing a three-branch parallel architecture: $\mathbf{v}_{\ell+1} = \sigma(\mathcal{K}_\ell(\mathbf{v}_\ell) + \mathbf{W}_\ell(\mathbf{v}_\ell) + \mathcal{U}_\ell(\mathbf{v}_\ell))$. The spectral branch $\mathcal{K}_\ell$ computes two-dimensional frequency-domain convolutions $\mathcal{F}^{-1}(\mathbf{R}_\ell^{(1)} \cdot \mathcal{F}(\mathbf{v})_{0:K_x,:K_y} + \mathbf{R}_\ell^{(2)} \cdot \mathcal{F}(\mathbf{v})_{-K_x:,:K_y})$ with $(K_x, K_y) = (16, 16)$ preserved modes in each spatial dimension, capturing global dependencies through truncated Fourier transforms with separate weight matrices for positive and negative frequencies. The spatial branch $\mathbf{W}_\ell$ applies 1×1 convolutions for channel mixing and local residuals. The U-Net branch $\mathcal{U}_\ell$ implements a lightweight encoder-decoder architecture with three downsampling layers (factors $2, 2, 2$) reaching resolution $H/8 \times W/8$, circular padding for periodic boundary compatibility, and skip connections that preserve multiscale spatial information across resolutions $H \times W$, $H/2 \times W/2$, $H/4 \times W/4$, and $H/8 \times W/8$, enabling fine-grained local feature refinement in both spatial directions. We employ a staged activation strategy controlled by parameter $\ell_{\text{start}} = 2$: layers $\ell < \ell_{\text{start}}$ use only spectral and spatial branches (FNO-only mode) for rapid coarse feature extraction, while layers $\ell \geq \ell_{\text{start}}$ incorporate the U-Net branch for detailed multiscale refinement, balancing computational efficiency with expressive power. GELU activation σ is applied to all but the final layer. The refined features are decoded through a single 1×1 convolution projecting from hidden dimension to output channels $C_{\text{out}} = 2$ with small initialization (standard deviation 0.01). The U-Net refiner adds approximately $3D^2$ parameters per activated layer through its convolutional structure, while kernel size $k = 3$ controls its local receptive field in both spatial dimensions. This hybrid architecture combines FNO's frequency-domain global propagation with U-Net's spatial-domain multiscale analysis, enabling simultaneous capture of long-range dependencies across the entire 2D domain and localized structures at multiple spatial scales.

Amortized Fourier Neural Operator for 2D Benchmarks. We employ an Amortized Fourier Neural Operator (AMFNO) that dynamically generates frequency-domain convolution kernels through multi-layer perceptrons, enabling frequency-adaptive PDE solving for two-dimensional spatial domains. The model processes input functions $\mathbf{u}(\mathbf{x})$ at $N = H \times W$ grid points, automatically augmenting them with normalized spatial coordinates to form feature vectors of dimension $C_{\text{in}}+2 = 4$ (including velocity field $[u, v]$ and normalized coordinates $[x/L_x, y/L_y]$). The architecture begins with a lifting layer that pointwise projects augmented inputs to hidden dimension $D = 32$ via a linear transformation. The lifted features are refined through $L = 4$ AMFNO layers, each implementing a dual-branch architecture with residual connections: $\mathbf{v}_{\ell+1} = \mathbf{v}_\ell + \mathcal{K}_{\text{MLP}}(\mathbf{v}_\ell) + \mathbf{W}_{\text{MLP}}(\mathbf{v}_\ell)$, where GELU activation $\sigma(\cdot)$ is applied to the summed output for all but the final layer. The spectral branch $\mathcal{K}_{\text{MLP}}$ replaces the fixed complex weights of standard FNO with dynamically generated kernels through a separable two-dimensional frequency representation: for each frequency mode $(\omega_{k_x}, \omega_{k_y})$ in the discrete two-dimensional Fourier spectrum, we encode the normalized frequency coordinates independently using Chebyshev polynomial bases as low-dimensional feature representations. Four separate MLPs with hidden dimension 4 then map these frequency encodings to real and imaginary components of separable convolution kernels: $K_x(\omega_{k_x}) = \text{MLP}_{xr}(\omega_{k_x}) + i \cdot \text{MLP}_{xi}(\omega_{k_x})$

and $K_y(\omega_{k_y}) = \text{MLP}_{yr}(\omega_{k_y}) + i \cdot \text{MLP}_{yi}(\omega_{k_y})$, with the final kernel formed through elementwise multiplication $K(\omega_{k_x}, \omega_{k_y}) = K_x(\omega_{k_x}) \odot K_y(\omega_{k_y}) \in \mathbb{C}^{D\times D}$, producing mode-specific transformations across all $H \times (W/2+1)$ frequencies in the real two-dimensional FFT spectrum. This dynamically generated kernel performs frequency-domain convolution $\mathcal{F}^{-1}(K(\omega_{k_x}, \omega_{k_y}) \cdot \mathcal{F}(\mathbf{v}))$, where $\mathcal{F}$ denotes the two-dimensional real-valued Fast Fourier Transform (rfft2). The spatial branch $\mathbf{W}_{\text{MLP}}$ applies a two-layer pointwise MLP with hidden dimension $4D = 128$ and GELU activation for local feature mixing across both spatial dimensions. The refined features are decoded through a two-layer pointwise MLP projecting from hidden dimension through intermediate dimension $4D$ to output channels $C_{\text{out}} = 2$ with GELU activation. This frequency-adaptive kernel generation with separable low-rank factorization enables superior resolution generalization compared to fixed-kernel FNO, where each frequency mode in both spatial directions receives a tailored transformation learned from data rather than using predetermined spectral truncation.

A.3 Training Methodology

Neural Network Input and Output. Following the Indirect Neural Corrector (INC) framework (Wei et al., 2026), the neural network learns a closure term τ_Δ that serves as a right-hand side correction to the coarse-grid governing equations. Given the current coarse-grid state $\bar{u}^n_\Delta$ at time step n, the network predicts the correction term $\tau^n_\Delta = \text{SINO}(\bar{u}^n_\Delta; \Theta)$, which is then incorporated into the time integration scheme. The corrected evolution follows:

$$\frac{\partial \bar{u}_\Delta}{\partial t} = \mathcal{L}_\Delta(\bar{u}_\Delta) + \tau_\Delta, \tag{44}$$

where $\mathcal{L}_\Delta$ denotes the coarse-grid spatial operator (e.g., WENO5 reconstruction for Burgers, pseudo-spectral derivatives for KS, van Leer flux limiting for NS). During rollout training, the predicted correction τ^n_Δ is held constant across all substeps within each multi-stage time integrator (e.g., SSP-RK3, ETDRK4), enabling efficient gradient backpropagation while maintaining numerical stability. This formulation decouples the neural operator from the specific discretization scheme, allowing SINO to generalize across different numerical solvers.

Training Objective. We train the neural network correction term using a rollout loss with curriculum learning:

$$\mathcal{L}_K(\theta) = \frac{1}{B}\sum_{i=1}^{B} \frac{1}{K+1} \sum_{k=0}^{K} \|u_i^k - \hat{u}_i^k(\theta)\|^2 \tag{45}$$

where B is the batch size, K is the rollout length (curriculum parameter), u_i^k is the ground truth at step k for trajectory i, and $\hat{u}_i^k(\theta)$ is the model prediction.

Curriculum Learning. The rollout length K grows progressively during training:

$$K(\text{epoch}) = \min\left(K_{\text{start}} + \left\lfloor \frac{\text{epoch}-1}{\Delta_{\text{epoch}}} \right\rfloor \cdot \Delta_K, K_{\max}\right) \tag{46}$$

where K_{start} is the initial rollout length (default: 10 steps), Δ_{epoch} is the growth interval (default: every 10 epochs), Δ_K is the growth increment (default: 15 steps), and $K_{\max}$ is the maximum rollout length corresponding to the training time horizon. This curriculum strategy stabilizes early training by first learning short-term dynamics, then gradually extending to longer time horizons.

Optimization. We use the AdamW optimizer with learning rate $\eta = 10^{-3}$ and cosine annealing schedule:

$$\eta(t) = \eta_0 \cdot 0.5\left(1 + \cos\left(\frac{\pi t}{T}\right)\right) \tag{47}$$

where t is the current optimization step and T is the total number of steps. For decaying Burgers turbulence, we apply weight decay $\lambda = 10^{-3}$ for regularization.

Data Normalization. For 1D equations (Burgers, KS), we normalize inputs using training set statistics: $u_{\text{norm}} = (u - \mu_{\text{train}})/\sigma_{\text{train}}$ or $v_{\text{norm}} = (v - \mu_{\text{train}})/\sigma_{\text{train}}$, where μ_{train} and σ_{train} are computed exclusively from the training temporal window to prevent data leakage. The correction term is scaled back as $\tau = \tau_{\text{norm}} \cdot \sigma_{\text{train}}$. For 2D Navier-Stokes equations, we use standard deviation normalization: $\mathbf{v}_{\text{norm}} = \mathbf{v}/\sigma_{\text{train}}$.

Time Integration with Correction Term. The neural network learns a right-hand side correction term τ that is incorporated into the physical solver during rollout training. For time integration, the same correction term computed at the beginning of each time step is reused across all substeps of the multi-stage integrator. This approach reduces computational cost and provides clearer gradient paths during backpropagation through time.

Model Selection. We select the best model based on extrapolation performance, measured by the mean squared error on the test temporal window $[T_{\text{train}}, T_{\text{final}}]$.

Training Schedule. We train for 300 epochs with batch size 8 trajectories. All tasks use 1001 temporal snapshots: the first frame is the initial condition, the first 30% (300 frames) form the training interval, and the remaining 70% (700 frames) are reserved for extrapolation testing. All reported MSE errors are extrapolation test errors.

Time Stepping Configuration. We use different time stepping strategies tailored to each benchmark:

Forcing Burgers: During the recording phase of $T_{\text{max}} = 10.0$ time units, $n_{\text{snapshots}} = 1001$ snapshots are saved at uniform intervals $\Delta t_{\text{outer}} = 0.01$. The DNS solver employs adaptive timesteps satisfying $\Delta t = \nu_{\text{CFL}} \cdot \min(\Delta x / \max_i |\bar{u}_i|, \Delta x^2/(2\eta))$ with safety factor $\nu_{\text{CFL}} = 0.4$ and viscosity $\eta = 0.01$.

Decaying Burgers: Snapshots are saved for $T_{\text{final}} = 0.1$ time units at intervals $\Delta t_{\text{outer}} = 10^{-4}$, yielding $n_{\text{snapshots}} = 1001$ snapshots. The DNS uses adaptive timesteps $\Delta t = \nu_{\text{CFL}} \cdot \min(\Delta x / \max_i |u_i|, \Delta x^2/(2\nu))$ with $\nu_{\text{CFL}} = 0.4$ and viscosity $\nu = 5 \times 10^{-4}$.

Kuramoto-Sivashinsky: After warmup of $T_{\text{warmup}} = 50.0$ time units, snapshots are recorded for $T_{\text{max}} = 10.0$ time units at intervals $\Delta t = 0.01$ using ETDRK4 integration, yielding $n_{\text{snapshots}} = 1001$ snapshots.

Forcing Navier-Stokes (Re=1000): After warmup of $T_{\text{warmup}} = 40.0$ time units, snapshots are recorded for production duration $T_{\text{production}} \approx 0.0448$ time units. The DNS employs inner batching with $n_{\text{inner}} = 8$ timesteps per saved frame and $n_{\text{outer}} = 1000$ saved frames, yielding $n_{\text{snapshots}} = 1001$ snapshots at intervals $\Delta t_{\text{frame}} = 8 \times \Delta t$, where $\Delta t = C_{\max} \cdot \Delta x / \|\mathbf{v}\|_{\max}$ with $C_{\max} = 0.5$ and $\|\mathbf{v}\|_{\max} = 7.0$.

Forcing Navier-Stokes (Re=4000): Identical configuration to Re=1000 case, but with viscosity $\nu = 2.5 \times 10^{-4}$ and the same maximum velocity $\|\mathbf{v}\|_{\max} = 7.0$.

Decaying Navier-Stokes: After warmup of $T_{\text{warmup}} = 4.5$ time units, snapshots are recorded for production duration $T_{\text{production}} \approx 0.0117$ time units. The DNS uses $n_{\text{inner}} = 8$ timesteps per frame and $n_{\text{outer}} = 1000$ frames, yielding $n_{\text{snapshots}} = 1001$ snapshots at intervals $\Delta t_{\text{frame}} = 8 \times \Delta t$, where $\Delta t = C_{\max} \cdot \Delta x / \|\mathbf{v}\|_{\max}$ with $C_{\max} = 0.5$ and initial maximum velocity $\|\mathbf{v}\|_{\max} = 4.2$.

Random Seeds and Reproducibility. All 1D and 2D PDEs use seeds 0–63 to generate 64 trajectories via the JAX random number generator `jax.random.PRNGKey(seed)` for initializing velocity fields. Two trajectories were excluded due to numerical instabilities caused by extreme initial values: Decaying NS trajectory 61 (seed=61) and Taylor-Green NS trajectory 61 (seed=61). All other benchmarks use the complete set of 64 trajectories. For neural network training, each model configuration uses three independent random seeds for weight initialization (`model_seed` $\in \{1, 2, 3\}$) to assess variance across random initializations, while the data loading seed (`seed`) is fixed at 42 for all experiments to ensure identical train-test splits. This design separates data randomness from model randomness, enabling fair comparison across architectures.

Computational Environment. All experiments were conducted on a workstation with Ubuntu 22.04, equipped with a single NVIDIA RTX 5090 GPU (32GB VRAM), 25 vCPU Intel Xeon Gold 6459C processor, Python 3.12, PyTorch 2.12.1, and CUDA 13.0. The JAX-based PDE solvers and neural network training utilized JIT compilation and automatic differentiation for computational efficiency.

A.4 Theoretical Foundations: Low-Rank Structure and Lipschitz Regularity

We provide mathematical analysis demonstrating that SINO's implicit kernel representation via bottleneck hypernetworks induces two fundamental properties: (1) explicit low-rank parameterization that aligns with the physical structure of closure operators, and (2) Lipschitz continuity that ensures smooth kernel variations and prevents overfitting to high-frequency noise. These properties jointly explain SINO's superior parameter efficiency and data efficiency in turbulent closure modeling. Our analysis establishes the low-rank structure of the learned kernels and quantitative bounds for operator norms, providing theoretical foundations for the empirical performance demonstrated in Section 4.

A.4.1 Bottleneck-Induced Subspace Constraint

Architecture. The frequency hypernetwork $\Phi_\omega : \mathbb{R}^2 \to \mathbb{C}^{D\times D}$ implements a three-layer bottleneck architecture:

$$\xi \xrightarrow{W_1^{(f)}} \mathbb{R}^{w_f} \xrightarrow{\sigma} \mathbb{R}^{w_f} \xrightarrow{W_2^{(f)}} \mathbb{R}^{w_f} \xrightarrow{\sigma} \mathbb{R}^{w_f} \xrightarrow{W_3^{(f)}} \mathbb{R}^{n_f} \xrightarrow{\sigma} \mathbb{R}^{n_f} \xrightarrow{[W_r,W_i]} \mathbb{C}^{D\times D}, \tag{48}$$

where $\xi \in \mathbb{R}^2$ denotes normalized frequency coordinates, $W_1^{(f)} \in \mathbb{R}^{w_f\times 2}$, $W_2^{(f)} \in \mathbb{R}^{w_f\times w_f}$, $W_3^{(f)} \in \mathbb{R}^{n_f\times w_f}$ is the bottleneck layer with $n_f \ll w_f$, $W_r, W_i \in \mathbb{R}^{D^2\times n_f}$ are output projection matrices (each column representing a vectorized $D \times D$ matrix), and $\sigma = \max(0, \cdot)$ is the ReLU activation function. The spatial hypernetwork $\Phi_x : \mathbb{R}^2 \to \mathbb{R}^{D\times D}$ follows an identical architecture with parameters $(W_1^{(s)} \in \mathbb{R}^{w_s\times 2}, W_2^{(s)} \in \mathbb{R}^{w_s\times w_s}, W_3^{(s)} \in \mathbb{R}^{n_s\times w_s}, W_s \in \mathbb{R}^{D^2\times n_s})$ and bottleneck dimension $n_s \ll w_s$.

Theorem 1 (Explicit Rank Constraint)**.** *Let $z_\omega(\xi) = \sigma(W_3^{(f)}\sigma(W_2^{(f)}\sigma(W_1^{(f)}\xi))) \in \mathbb{R}^{n_f}$ denote the frequency bottleneck representation and $z_x(\zeta) = \sigma(W_3^{(s)}\sigma(W_2^{(s)}\sigma(W_1^{(s)}\zeta))) \in \mathbb{R}^{n_s}$ denote the spatial bottleneck representation. Then:*

(a) Frequency domain: *The spectral kernel admits the explicit low-rank decomposition*

$$m(\xi) = \sum_{j=1}^{n_f} z_{\omega,j}(\xi) \cdot (B_j^{(r)} + iB_j^{(i)}), \quad B_j^{(r)}, B_j^{(i)} \in \mathbb{R}^{D\times D}, \tag{49}$$

where $B_j^{(r)} = \mathrm{mat}_{D\times D}([W_r]_{\cdot,j})$ and $B_j^{(i)} = \mathrm{mat}_{D\times D}([W_i]_{\cdot,j})$ are obtained by reshaping the j-th column vectors $[W_r]_{\cdot,j}, [W_i]_{\cdot,j} \in \mathbb{R}^{D^2}$ into $D \times D$ matrices via the inverse vectorization operator $\mathrm{mat}_{D\times D} : \mathbb{R}^{D^2} \to \mathbb{R}^{D\times D}$. Consequently,

$$\mathcal{M}_\omega := \{m(\xi) : \xi \in [-1, 1]^2\} \subseteq \mathrm{span}_{\mathbb{C}}\{B_1, \ldots, B_{n_f}\}, \quad B_j := B_j^{(r)} + iB_j^{(i)}. \tag{50}$$

(b) Spatial domain: *The spatial kernel admits the explicit low-rank decomposition*

$$W(\zeta) = \sum_{j=1}^{n_s} z_{x,j}(\zeta) \cdot C_j, \quad C_j \in \mathbb{R}^{D\times D}, \tag{51}$$

where $C_j = \mathrm{mat}_{D\times D}([W_s]_{\cdot,j})$ are obtained by reshaping the j-th column vectors $[W_s]_{\cdot,j} \in \mathbb{R}^{D^2}$ into $D \times D$ matrices. Consequently,

$$\mathcal{W}_x := \{W(\zeta) : \zeta \in [-1, 1]^2\} \subseteq \mathrm{span}_{\mathbb{R}}\{C_1, \ldots, C_{n_s}\}. \tag{52}$$

Both constructions reduce the effective parameter count from $O(k_{\max} \cdot D^2)$ (FNO) or $O(K^2 \cdot D^2)$ (CNN) to $O((n_f + n_s) \cdot D^2)$ (SINO), independent of resolution or kernel size.

Proof. **(a) Frequency domain:** By construction, the output layer computes

$$\mathrm{vec}(m(\xi)) = W_r z_\omega(\xi) + i\, W_i z_\omega(\xi) \in \mathbb{C}^{D^2}, \tag{53}$$

where $\mathrm{vec} : \mathbb{C}^{D\times D} \to \mathbb{C}^{D^2}$ denotes the vectorization operator that stacks matrix columns into a vector. Expanding the matrix-vector products:

$$\mathrm{vec}(m(\xi)) = \sum_{j=1}^{n_f} z_{\omega,j}(\xi)\,[W_r]_{\cdot,j} + i\sum_{j=1}^{n_f} z_{\omega,j}(\xi)\,[W_i]_{\cdot,j} \tag{54}$$

$$= \sum_{j=1}^{n_f} z_{\omega,j}(\xi)\,([W_r]_{\cdot,j} + i\,[W_i]_{\cdot,j})\,, \tag{55}$$

where $[W_r]_{\cdot,j}, [W_i]_{\cdot,j} \in \mathbb{R}^{D^2}$ denote the j-th column of W_r and W_i respectively. Applying the inverse vectorization operator $\mathrm{mat}_{D\times D}$:

$$m(\xi) = \sum_{j=1}^{n_f} z_{\omega,j}(\xi)\,\mathrm{mat}_{D\times D}\,([W_r]_{\cdot,j} + i\,[W_i]_{\cdot,j}) = \sum_{j=1}^{n_f} z_{\omega,j}(\xi)\cdot(B_j^{(r)} + iB_j^{(i)}). \tag{56}$$

Since $z_\omega(\xi) \in \mathbb{R}^{n_f}$ for all ξ, every kernel matrix $m(\xi)$ necessarily lies in $\mathrm{span}_{\mathbb{C}}\{B_1, \ldots, B_{n_f}\}$.

(b) Spatial domain: Identically, the spatial output layer computes

$$\mathrm{vec}(W(\zeta)) = W_s z_x(\zeta) = \sum_{j=1}^{n_s} z_{x,j}(\zeta)\,[W_s]_{\cdot,j} \in \mathbb{R}^{D^2}. \tag{57}$$

Applying the inverse vectorization:

$$W(\zeta) = \mathrm{mat}_{D\times D}\left(\sum_{j=1}^{n_s} z_{x,j}(\zeta)\,[W_s]_{\cdot,j}\right) = \sum_{j=1}^{n_s} z_{x,j}(\zeta)\cdot C_j. \tag{58}$$

Since $z_x(\zeta) \in \mathbb{R}^{n_s}$ for all ζ, every kernel matrix $W(\zeta)$ necessarily lies in $\mathrm{span}_{\mathbb{R}}\{C_1, \ldots, C_{n_s}\}$.

The parameter count for the frequency hypernetwork is $2w_f + w_f^2 + n_f w_f + 2n_f D^2 = O(n_f D^2)$ when $n_f D^2 \gg w_f^2$. Similarly for the spatial hypernetwork, yielding total count $O((n_f + n_s)D^2)$. □

Remark 1. The rank constraints are *hard*: the kernel families $\mathcal{M}_\omega$ and $\mathcal{W}_x$ are proper subspaces of $\mathbb{C}^{D\times D}$ and $\mathbb{R}^{D\times D}$ respectively, unlike FNO and CNN where all $k_{\max}$ Fourier modes or K^2 spatial kernel entries are independent parameters. This structural bias aligns with the physical prior that closure operators exhibit low-rank structure due to scale separation in turbulent flows.

A.4.2 Functional Continuity and Lipschitz Regularity

Theorem 2 (Lipschitz Continuity of Hypernetworks)**.** *Let Φ_ω and Φ_x be the frequency and spatial hypernetworks with ReLU activations. Then both mappings are Lipschitz continuous:*

(a) Frequency hypernetwork: *For any $\xi, \xi' \in [-1, 1]^2$,*

$$\|m(\xi) - m(\xi')\|_{\mathrm{F}} \le L_\omega \|\xi - \xi'\|_2, \tag{59}$$

where

$$L_\omega = \sqrt{\|W_r\|_2^2 + \|W_i\|_2^2}\cdot \|W_3^{(f)}\|_2 \|W_2^{(f)}\|_2 \|W_1^{(f)}\|_2. \tag{60}$$

(b) Spatial hypernetwork: *For any $\zeta, \zeta' \in [-1, 1]^2$,*

$$\|W(\zeta) - W(\zeta')\|_{\mathrm{F}} \le L_x \|\zeta - \zeta'\|_2, \tag{61}$$

where

$$L_x = \|W_s\|_2 \cdot \|W_3^{(s)}\|_2 \|W_2^{(s)}\|_2 \|W_1^{(s)}\|_2. \tag{62}$$

Here $\|\cdot\|_2$ denotes the spectral norm (largest singular value) and $\|\cdot\|_{\mathrm{F}} = \sqrt{\sum_{i,j}|a_{ij}|^2}$ denotes the Frobenius norm.

Proof. **(a) Frequency domain:** The ReLU activation function $\sigma(a) = \max(0, a)$ satisfies $|\sigma(a) - \sigma(b)| \leq |a - b|$ (1-Lipschitz property). By composition of Lipschitz functions, the bottleneck representation satisfies

$$\|z_\omega(\xi) - z_\omega(\xi')\|_2 \leq \|W_3^{(f)}\|_2 \|W_2^{(f)}\|_2 \|W_1^{(f)}\|_2 \|\xi - \xi'\|_2 =: L_{z,\omega} \|\xi - \xi'\|_2. \tag{63}$$

The output layer transformation yields

$$\begin{aligned}
\|m(\xi) - m(\xi')\|_{\mathrm{F}}^2 &= \|\mathrm{mat}_{D\times D}(W_r(z_\omega(\xi) - z_\omega(\xi')))\|_{\mathrm{F}}^2 \\
&\quad + \|\mathrm{mat}_{D\times D}(W_i(z_\omega(\xi) - z_\omega(\xi')))\|_{\mathrm{F}}^2 && (64)\\
&= \|W_r(z_\omega(\xi) - z_\omega(\xi'))\|_2^2 + \|W_i(z_\omega(\xi) - z_\omega(\xi'))\|_2^2 && (65)\\
&\leq \|W_r\|_2^2 \|z_\omega(\xi) - z_\omega(\xi')\|_2^2 + \|W_i\|_2^2 \|z_\omega(\xi) - z_\omega(\xi')\|_2^2 && (66)\\
&= (\|W_r\|_2^2 + \|W_i\|_2^2) \|z_\omega(\xi) - z_\omega(\xi')\|_2^2 && (67)\\
&\leq (\|W_r\|_2^2 + \|W_i\|_2^2) L_{z,\omega}^2 \|\xi - \xi'\|_2^2. && (68)
\end{aligned}$$

In the second equality, we used the fact that the Frobenius norm is preserved under the inverse vectorization operator: $\|\mathrm{mat}_{D\times D}(v)\|_{\mathrm{F}} = \|v\|_2$ for any $v \in \mathbb{R}^{D^2}$. Taking the square root yields $L_\omega = \sqrt{\|W_r\|_2^2 + \|W_i\|_2^2} \cdot L_{z,\omega}$.

(b) Spatial domain: Identically, the bottleneck representation satisfies

$$\|z_x(\zeta) - z_x(\zeta')\|_2 \leq \|W_3^{(s)}\|_2 \|W_2^{(s)}\|_2 \|W_1^{(s)}\|_2 \|\zeta - \zeta'\|_2 =: L_{z,x} \|\zeta - \zeta'\|_2, \tag{69}$$

and the output layer transformation satisfies

$$\|W(\zeta) - W(\zeta')\|_{\mathrm{F}} = \|W_s(z_x(\zeta) - z_x(\zeta'))\|_2 \leq \|W_s\|_2 \|z_x(\zeta) - z_x(\zeta')\|_2 \leq L_x \|\zeta - \zeta'\|_2. \tag{70}$$

□

Remark 2. The Lipschitz constants L_ω and L_x are computable from trained model weights, providing quantitative bounds on kernel smoothness. This regularization prevents overfitting to high-frequency noise inherent in coarse-grid observations, which is critical for closure modeling where fine-scale information is inaccessible during training.

A.4.3 Approximation Capacity and Universal Approximation

Theorem 3 (Universal Approximation of Compact Kernel Families)**.**

(a) Frequency domain: *Let $\mathcal{K}_\omega = \{K(\xi) : \xi \in [-1, 1]^2\} \subset \mathbb{C}^{D\times D}$ be a compact family of continuous kernel functions satisfying $\sup_\xi \|K(\xi)\|_{\mathrm{F}} < \infty$. For any $\epsilon > 0$, there exist widths (w_f, n_f) and parameters $\Theta_\omega = \{W_1^{(f)}, W_2^{(f)}, W_3^{(f)}, W_r, W_i\}$ such that*

$$\sup_{\xi\in[-1,1]^2} \|K(\xi) - \Phi_\omega(\xi; \Theta_\omega)\|_{\mathrm{F}} < \epsilon. \tag{71}$$

(b) Spatial domain: *Let $\mathcal{K}_x = \{K(\zeta) : \zeta \in [-1, 1]^2\} \subset \mathbb{R}^{D\times D}$ be a compact family of continuous kernel functions satisfying $\sup_\zeta \|K(\zeta)\|_{\mathrm{F}} < \infty$. For any $\epsilon > 0$, there exist widths (w_s, n_s) and parameters $\Theta_x = \{W_1^{(s)}, W_2^{(s)}, W_3^{(s)}, W_s\}$ such that*

$$\sup_{\zeta\in[-1,1]^2} \|K(\zeta) - \Phi_x(\zeta; \Theta_x)\|_{\mathrm{F}} < \epsilon. \tag{72}$$

Proof. We establish (a); statement (b) follows by identical reasoning with real-valued functions.

Step 1: Polynomial approximation via Stone-Weierstrass theorem.

Consider the algebra $\mathcal{A}$ of complex matrix-valued functions of the form

$$F(\xi) = \sum_{k=1}^{m} p_k(\xi_1, \xi_2) M_k, \quad M_k \in \mathbb{C}^{D\times D}, \tag{73}$$

where $p_k(\xi_1, \xi_2)$ are polynomials in two real variables and $\xi = (\xi_1, \xi_2) \in [-1, 1]^2$.

The algebra $\mathcal{A}$ satisfies the conditions of the Stone-Weierstrass theorem. Applying the classical Stone-Weierstrass theorem component-wise to each of the D^2 matrix entries (each entry being a continuous complex-valued function on $[-1, 1]^2$), we conclude that $\mathcal{A}$ is dense in $C([-1, 1]^2; \mathbb{C}^{D\times D})$ under the supremum norm $\sup_\xi \|F(\xi)\|_{\mathrm{F}}$. Since $K(\xi)$ is continuous on the compact set $[-1, 1]^2$, for any $\epsilon/2 > 0$, there exists a finite polynomial expansion

$$K_{\text{poly}}(\xi) = \sum_{|\beta|\le d} \xi^\beta M_\beta, \quad M_\beta \in \mathbb{C}^{D\times D}, \tag{74}$$

where $\beta = (\beta_1, \beta_2)$ is a multi-index with $|\beta| = \beta_1 + \beta_2 \le d$, $\xi^\beta = \xi_1^{\beta_1}\xi_2^{\beta_2}$, and

$$\sup_{\xi\in[-1,1]^2} \|K(\xi) - K_{\text{poly}}(\xi)\|_{\mathrm{F}} < \epsilon/2. \tag{75}$$

Step 2: Finite-rank representation.

The polynomial approximation $K_{\text{poly}}(\xi)$ involves $N_{\text{poly}} := \binom{d+2}{2}$ distinct monomial terms. We can rewrite it as

$$K_{\text{poly}}(\xi) = \sum_{j=1}^{N_{\text{poly}}} p_j(\xi) B_j, \tag{76}$$

where $\{p_j(\xi)\}_{j=1}^{N_{\text{poly}}}$ are the monomial basis functions and $\{B_j\}_{j=1}^{N_{\text{poly}}}$ are the corresponding coefficient matrices. Define $\boldsymbol{\alpha}(\xi) = (p_1(\xi), \ldots, p_{N_{\text{poly}}}(\xi))^\top : [-1, 1]^2 \to \mathbb{R}^{N_{\text{poly}}}$. Without loss of generality, assume $\max_j \|B_j\|_{\mathrm{F}} \le C_B$ for some constant $C_B > 0$.

Step 3: Neural network approximation.

By the universal approximation theorem for feedforward neural networks with ReLU activations, for the continuous vector-valued function $\boldsymbol{\alpha} : [-1, 1]^2 \to \mathbb{R}^{N_{\text{poly}}}$ and tolerance

$$\epsilon' := \frac{\epsilon}{2\sqrt{N_{\text{poly}}} C_B}, \tag{77}$$

there exists a three-layer ReLU network $\hat{\boldsymbol{\alpha}} : [-1, 1]^2 \to \mathbb{R}^{N_{\text{poly}}}$ with hidden widths (w_f, w_f) and output dimension N_{poly} such that

$$\sup_{\xi\in[-1,1]^2} \|\boldsymbol{\alpha}(\xi) - \hat{\boldsymbol{\alpha}}(\xi)\|_2 < \epsilon'. \tag{78}$$

Step 4: Hypernetwork construction and error analysis.

Set the bottleneck dimension $n_f = N_{\text{poly}}$ and configure the hypernetwork Φ_ω such that its bottleneck output satisfies $z_\omega(\xi) = \hat{\boldsymbol{\alpha}}(\xi)$. Construct $W_r, W_i \in \mathbb{R}^{D^2\times N_{\text{poly}}}$ such that the j-th column of W_r contains $\mathrm{vec}(\mathrm{Re}(B_j))$ and the j-th column of W_i contains $\mathrm{vec}(\mathrm{Im}(B_j))$, where $\mathrm{vec} : \mathbb{C}^{D\times D} \to \mathbb{C}^{D^2}$ denotes the vectorization operator. The hypernetwork output is then

$$\Phi_\omega(\xi; \Theta_\omega) = \sum_{j=1}^{N_{\text{poly}}} \hat{\alpha}_j(\xi) B_j. \tag{79}$$

By the triangle inequality and Cauchy-Schwarz inequality,

$$\sup_{\xi\in[-1,1]^2} \|K(\xi) - \Phi_\omega(\xi)\|_{\mathrm{F}} \tag{80}$$

$$\leq \sup_{\xi} \left\| K(\xi) - \sum_{j=1}^{N_{\text{poly}}} \alpha_j(\xi) B_j \right\|_{\mathrm{F}} + \sup_{\xi} \left\| \sum_{j=1}^{N_{\text{poly}}} (\alpha_j(\xi) - \hat{\alpha}_j(\xi)) B_j \right\|_{\mathrm{F}} \tag{81}$$

$$< \frac{\epsilon}{2} + \sup_{\xi} \sum_{j=1}^{N_{\text{poly}}} |\alpha_j(\xi) - \hat{\alpha}_j(\xi)| \|B_j\|_{\mathrm{F}} \tag{82}$$

$$\leq \frac{\epsilon}{2} + \sup_{\xi} \left(\|\boldsymbol{\alpha}(\xi) - \hat{\boldsymbol{\alpha}}(\xi)\|_2 \cdot \sqrt{\sum_{j=1}^{N_{\text{poly}}} \|B_j\|_{\mathrm{F}}^2} \right) \tag{83}$$

$$\leq \frac{\epsilon}{2} + \epsilon' \cdot \sqrt{N_{\text{poly}}} C_B = \frac{\epsilon}{2} + \frac{\epsilon}{2} = \epsilon. \tag{84}$$

□

Remark 3. The proof establishes existence of an approximating hypernetwork but does not claim optimality of the construction. The required bottleneck dimension $n_f = N_{\text{poly}} = \binom{d+2}{2} = O(d^2)$ depends on the polynomial degree d needed for approximation, which in turn depends on the smoothness properties of $K(\xi)$. For C^k-smooth kernels, $d = O(\epsilon^{-1/k})$ suffices, implying $n_f = O(\epsilon^{-2/k})$. This suggests that smoother closure kernels require exponentially fewer bottleneck dimensions for fixed accuracy, aligning with the physical intuition that well-resolved turbulent flows exhibit smoother scale-transfer operators.

A.4.4 Operator Norm Bounds and Regularization

Theorem 4 (Frequency-Domain Operator Norm)**.** *The bottleneck constraint in the frequency hypernetwork controls the operator norm. For the spectral convolution operator acting on functions $h : \Omega \to \mathbb{C}^D$ where $\Omega \subset \mathbb{R}^d$ is the spatial domain, we have*

$$\sup_{\substack{h \\ \|h\|_{L^2(\Omega;\mathbb{C}^D)}=1}} \left\| \mathcal{F}^{-1}\left[m(\xi)\hat{h}(\xi) \right] \right\|_{L^2(\Omega;\mathbb{C}^D)} \leq \sqrt{n_f} \cdot \max_j \|B_j\|_2 \cdot \sup_{\xi\in[-1,1]^2} \|z_\omega(\xi)\|_2, \tag{85}$$

where B_j are the basis matrices from Theorem 1(a), $\|\cdot\|_2$ denotes the spectral norm, and $\hat{h}(\xi)$ denotes the Fourier transform. This bound grows sublinearly with n_f, contrasting with FNO's linear growth in the number of Fourier modes $k_{\max}$.

Proof. From Theorem 1(a), $m(\xi) = \sum_{j=1}^{n_f} z_{\omega,j}(\xi) B_j$. By the triangle inequality for spectral norms,

$$\|m(\xi)\|_2 \leq \sum_{j=1}^{n_f} |z_{\omega,j}(\xi)| \|B_j\|_2. \tag{86}$$

Applying the Cauchy-Schwarz inequality,

$$\sum_{j=1}^{n_f} |z_{\omega,j}(\xi)| \|B_j\|_2 \leq \left(\sum_{j=1}^{n_f} |z_{\omega,j}(\xi)|^2 \right)^{1/2} \left(\sum_{j=1}^{n_f} \|B_j\|_2^2 \right)^{1/2} \tag{87}$$

$$\leq \|z_\omega(\xi)\|_2 \cdot \sqrt{n_f} \cdot \max_j \|B_j\|_2. \tag{88}$$

For any h with $\|h\|_{L^2(\Omega;\mathbb{C}^D)} = 1$, by the discrete Parseval identity (with appropriate normalization for the discrete Fourier transform),

$$\left\| \mathcal{F}^{-1}[m(\xi)\hat{h}(\xi)] \right\|^2_{L^2(\Omega;\mathbb{C}^D)} \leq \sup_{\xi} \|m(\xi)\|_2^2 \cdot \|h\|^2_{L^2(\Omega;\mathbb{C}^D)}. \tag{89}$$

Taking the supremum over all such h yields the stated bound. The supremum $\sup_\xi \|z_\omega(\xi)\|_2$ is finite by continuity of z_ω and compactness of $[-1,1]^2$. □

Theorem 5 (Spatial-Domain Operator Norm)**.** *The bottleneck constraint in the spatial hypernetwork controls the convolution operator norm. For the spatial convolution operator acting on functions $h : \Omega \to \mathbb{R}^D$ where $\Omega \subset \mathbb{R}^d$ is the spatial domain, we have*

$$\sup_{\substack{h \\ \|h\|_{L^2(\Omega;\mathbb{R}^D)}=1}} \left\| \int_{[-1,1]^2} W(\zeta) h(\cdot - \zeta)\, d\zeta \right\|_{L^2(\Omega;\mathbb{R}^D)} \leq 4n_s \cdot \max_j \|C_j\|_2 \cdot \sup_{\zeta\in[-1,1]^2} \|z_x(\zeta)\|_2, \quad (90)$$

where C_j are the basis matrices from Theorem 1(b), the constant $4 = |[-1,1]^2|$ denotes the Lebesgue measure of the compact support, and $\|\cdot\|_2$ denotes the spectral norm.

Proof. By the rank-n_s decomposition from Theorem 1(b), $W(\zeta) = \sum_{j=1}^{n_s} z_{x,j}(\zeta) C_j$. For any h with $\|h\|_{L^2(\Omega;\mathbb{R}^D)} = 1$,

$$\left\| \int_{[-1,1]^2} W(\zeta) h(\cdot - \zeta)\, d\zeta \right\|_{L^2(\Omega;\mathbb{R}^D)} \leq \sum_{j=1}^{n_s} \left\| \int_{[-1,1]^2} z_{x,j}(\zeta) C_j h(\cdot - \zeta)\, d\zeta \right\|_{L^2(\Omega;\mathbb{R}^D)}. \quad (91)$$

For each summand, applying Minkowski's integral inequality and exploiting translation invariance of the L^2 norm,

$$\left\| \int_{[-1,1]^2} z_{x,j}(\zeta) C_j h(\cdot - \zeta)\, d\zeta \right\|_{L^2(\Omega;\mathbb{R}^D)} \leq \int_{[-1,1]^2} |z_{x,j}(\zeta)|\, \|C_j h(\cdot - \zeta)\|_{L^2(\Omega;\mathbb{R}^D)}\, d\zeta \quad (92)$$

$$\leq \|C_j\|_2 \int_{[-1,1]^2} |z_{x,j}(\zeta)|\, \|h\|_{L^2(\Omega;\mathbb{R}^D)}\, d\zeta \quad (93)$$

$$\leq 4\, \|C_j\|_2 \cdot \sup_{\zeta\in[-1,1]^2} |z_{x,j}(\zeta)|. \quad (94)$$

Summing over $j = 1, \ldots, n_s$ yields the stated bound. □

Remark 4. Theorem 4 establishes sublinear growth $O(\sqrt{n_f})$ for the frequency-domain operator norm, providing favorable implicit regularization through spectral orthogonality. Theorem 5 yields linear growth $O(n_s)$ for the spatial-domain operator norm, reflecting the fundamental difference between Fourier multiplication and spatial convolution. Nevertheless, both branches enforce low-rank structural constraints that dramatically reduce parameter count and overfitting compared to explicit parameterization methods. Specifically, FNO requires $O(k_{\max} \cdot D^2)$ parameters per layer, where $k_{\max}$ denotes the number of retained Fourier modes that typically scales with spatial resolution. Standard CNNs require $O(K^2 \cdot D^2)$ parameters per layer, where K is the kernel size that grows quadratically with receptive field requirements. In contrast, SINO's hypernetwork architecture reduces the effective parameter count to $O((n_f + n_s) \cdot D^2)$ per layer, where the bottleneck dimensions n_f, n_s are independent of resolution and kernel size. Since typical configurations satisfy $n_f, n_s \ll \min(k_{\max}, K^2)$, SINO achieves order-of-magnitude parameter reduction while maintaining expressiveness through continuous functional representations of the low-rank physical operators.

A.5 Additional Experiments

A.5.1 Cross-Resolution and Alternative Forcing Generalization

To further validate SINO's scale-invariant learning capability and robustness across different physical regimes, we conduct additional experiments evaluating performance at alternative coarse-graining ratios and under different forcing configurations beyond the primary benchmarks in Section 4. These supplementary tests assess whether SINO's superior performance generalizes to: (1) different levels of information loss induced by varying downsampling factors, and (2) qualitatively different turbulent dynamics arising from alternative forcing mechanisms or Reynolds numbers.

1D Supplementary Benchmarks. We extend the 1D turbulence experiments to test robustness across resolution scales and initial condition sensitivity. *Forcing Burgers (128)* increases the coarse-grid resolution from 32 to 128 (resample factor 4 instead of 16), reducing information loss while maintaining the same forcing configuration ($\eta = 0.01$, multi-mode sinusoidal forcing with 20 modes). This tests whether models continue to improve performance when more resolved scales are available, or if they plateau due to architectural limitations. *KS (32)* aggressively coarsens the Kuramoto-Sivashinsky equation from 256 to 32 (resample factor 8 instead of 4), increasing the closure gap. This extreme downsampling tests model robustness when large portions of the inertial range are unresolved, a regime where traditional LES closures typically fail. *Decaying Burgers (128)* reduces the downsampling factor from 8 to 16 (resolution 128 instead of 256) for the freely decaying case, providing an intermediate resolution test for transient dynamics without external forcing. This configuration challenges models to capture energy decay trajectories with moderately resolved scales.

2D Supplementary Benchmarks. We test three additional Navier-Stokes configurations that alter either the Reynolds number or the forcing mechanism. *Forcing NS (Re=2000)* introduces an intermediate Reynolds number regime ($\nu = 5 \times 10^{-4}$) between the Re=1000 and Re=4000 cases, maintaining Kolmogorov forcing ($F_0 = 1.0$, $k_f = 4$) and linear damping ($\alpha = -0.1$). This tests model interpolation capability across Reynolds numbers and sensitivity to viscosity variations, which directly affect the dissipation range width and subgrid-scale stress magnitude. *Forcing NS (Taylor-Green, Re=1000)* replaces Kolmogorov forcing with Taylor-Green vortex forcing, which injects energy isotropically at large scales through a vortex array pattern: $\mathbf{f}_{TG} = (F_0 \sin(k_f x)\cos(k_f y), -F_0 \cos(k_f x)\sin(k_f y))$ with $F_0 = 1.0$ and $k_f = 2$. This qualitatively different forcing geometry produces distinct vortex interaction dynamics compared to the anisotropic Kolmogorov forcing, testing whether learned closures generalize across forcing symmetries. *Forcing NS (Re=10^5)* drastically increases the Reynolds number to Re $= 10^5$ ($\nu = 10^{-5}$) while maintaining Kolmogorov forcing, creating highly turbulent conditions with extended inertial ranges. The flow exhibits rapid energy cascade and strong scale separation, providing a stringent test for closure models under extreme turbulence intensity. Note that this configuration uses the same coarse resolution (64×64) but with viscosity two orders of magnitude smaller than the Re=1000 forced case, resulting in significantly underresolved dynamics.

All supplementary benchmarks use identical spatial discretization schemes (WENO5 for forced Burgers, pseudo-spectral for KS and decaying Burgers, van Leer finite volume for NS), temporal integration methods (SSP-RK3, ETDRK4, TVD-RK3, semi-implicit splitting as appropriate), and training protocols (300 epochs, batch size 8, 30% training / 70% testing split, curriculum learning with rollout growth from 10 to maximum steps) as the primary benchmarks. Model hyperparameters remain unchanged from Section 4 to ensure fair comparison.

Results: 1D Supplementary Benchmarks. Table 4 presents quantitative comparisons on the three additional 1D configurations. On Forcing Burgers (128), SINO achieves the lowest error (6.15E-05 average) with only 6.7K parameters, outperforming the second-best SINO-SPA (8.82E-05) by 1.4× and U-Net (1.15E-04) by 1.9×. Notably, all methods exhibit substantially lower errors at this higher resolution compared to the 32-resolution case (1.03E-03 in Table 2), confirming that increased grid refinement reduces the closure gap. However, SINO maintains the largest relative improvement, demonstrating superior utilization of available resolved scales. On the extremely coarse KS (32) benchmark, SINO (1.30E-01 average) dramatically outperforms all baselines: 2.8× better than AMFNO (3.58E-01), 3.4× better than U-Net (4.38E-01), and 5.6-8.1× better than Transformer methods. This aggressive coarsening causes most baselines to degrade significantly compared to the 64-resolution case (Table 2), while SINO exhibits graceful degradation, validating its robustness under severe information loss. For Decaying Burgers (128), SINO (1.57E-02 average) achieves 12× improvement over most baselines that plateau around 1.87E-01 (the coarse-grid baseline). Only SINO-SPA (1.79E-02) approaches SINO's performance, while all other methods fail to improve upon uncorrected coarse-grid simulation.

Results: 2D Supplementary Benchmarks. Table 5 presents quantitative comparisons on the three additional 2D configurations. On Forcing NS (Re=2000), SINO achieves the best performance (1.24E-01 average, 1.18E-01 best) with 59K parameters, outperforming U-Net (1.51E-01, 61K parameters) by 1.2× despite comparable parameter count. UFNO (1.94E-01, 1.3M parameters) and AMFNO (2.10E-01, 120K parameters) require 2-23× more parameters for substantially worse accuracy. Transformer-based methods exhibit high errors: Transolver (4.72E-01), GK-Transformer

Table 4: Performance comparison on supplementary 1D benchmarks. Forcing Burgers (128) uses resolution 512→128 (resample factor 4), KS (32) uses 256→32 (resample factor 8), and Decaying Burgers (128) uses 2048→128 (resample factor 16).

Method	Supplementary 1D Benchmarks						Params
	Forcing Burgers (128)		KS (32)		Decaying Burgers (128)		
	AVE	Best	AVE	Best	AVE	Best	
U-Net	1.15E-04	7.76E-05	4.38E-01	5.92E-01	1.87E-01	1.87E-01	7,077
DeepONet	2.00E-04	1.72E-04	1.70E+00	1.39E+00	1.87E-01	1.87E-01	14,913
Transolver	2.60E-04	2.55E-04	7.47E-01	7.36E-01	1.87E-01	1.87E-01	8,601
Oformer	1.18E-04	1.18E-04	1.89E+01	1.89E+01	1.87E-01	1.87E-01	5,444
GK-Transformer	3.05E-04	2.97E-04	7.32E-01	7.17E-01	1.87E-01	1.87E-01	36,449
FNO	2.13E-04	1.90E-04	1.05E+00	9.44E-01	1.87E-01	1.87E-01	11,073
UFNO	9.70E-04	9.25E-04	9.27E-01	8.47E-01	1.87E-01	1.87E-01	26,017
AMFNO	1.36E-04	1.23E-04	3.58E-01	3.37E-01	1.88E-01	1.87E-01	9,489
SINO-SPE	2.14E-04	1.29E-04	1.89E-01	1.06E-01	1.05E-01	8.70E-02	3,877
SINO-SPA	8.82E-05	7.82E-05	5.76E-01	5.37E-01	1.79E-02	1.76E-02	1,797
SINO (ours)	**6.15E-05**	**5.72E-05**	**1.30E-01**	**6.84E-02**	**1.57E-02**	**1.37E-02**	6,681
Coarse Grid	2.65E-01		1.89E+01		1.87E-01		–

Table 5: Performance comparison on supplementary 2D benchmarks. Forcing NS (Re=2000) uses $\nu = 5 \times 10^{-4}$ with Kolmogorov forcing, Forcing NS (Taylor-Green, Re=1000) uses $\nu = 10^{-3}$ with Taylor-Green vortex forcing, and Forcing NS (Re=10^5) uses $\nu = 10^{-5}$ with Kolmogorov forcing. All configurations use resolution 512→64.

Method	Supplementary 2D Benchmarks						Params
	Forcing NS (Re=2000)		NS (Taylor-Green, Re=1000)		Forcing NS (Re=10^5)		
	AVE	Best	AVE	Best	AVE	Best	
U-Net	1.51E-01	1.48E-01	1.41E-02	1.09E-02	2.37E-01	2.31E-01	61,006
DeepONet	5.02E-01	4.96E-01	1.59E-01	1.56E-01	6.04E-01	5.91E-01	269,153
Transolver	4.72E-01	4.14E-01	6.61E-02	6.01E-02	5.59E-01	5.37E-01	54,306
Oformer	5.64E-01	5.64E-01	1.61E-01	1.61E-01	6.45E-01	6.43E-01	38,087
GK-Transformer	4.52E-01	4.14E-01	7.84E-02	7.50E-02	5.50E-01	5.41E-01	151,874
FNO	4.97E-01	4.95E-01	1.10E-01	1.07E-01	5.83E-01	5.76E-01	2,106,018
UFNO	1.94E-01	1.88E-01	1.64E-02	1.57E-02	2.62E-01	2.50E-01	1,346,466
AMFNO	2.10E-01	2.06E-01	2.31E-02	2.16E-02	3.00E-01	2.88E-01	120,162
SINO-SPE	2.56E-01	2.49E-01	2.87E-02	2.57E-02	3.72E-01	3.56E-01	33,834
SINO-SPA	2.45E-01	2.25E-01	4.14E-02	3.14E-02	3.40E-01	2.99E-01	17,322
SINO (ours)	**1.24E-01**	**1.18E-01**	**1.01E-02**	**8.50E-03**	**2.03E-01**	**1.98E-01**	59,314
Coarse Grid	5.65E-01		1.61E-01		6.45E-01		–

(4.52E-01), and Oformer (5.64E-01). This intermediate Reynolds number result confirms SINO's smooth interpolation capability between the Re=1000 and Re=4000 regimes. On Forcing NS (Taylor-Green, Re=1000), SINO (1.01E-02 average, 8.50E-03 best) demonstrates exceptional generalization to qualitatively different forcing geometry, achieving the lowest error across all methods. U-Net (1.41E-02, 61K parameters) ranks second but with 1.4× higher error. UFNO (1.64E-02, 1.3M parameters) and AMFNO (2.31E-02, 120K parameters) again underperform despite massive capacity. Notably, SINO's error on Taylor-Green forcing is actually lower than its performance on Kolmogorov forcing at the same Reynolds number (9.17E-02 in Table 3), suggesting that the isotropic vortex array produces more predictable subgrid-scale dynamics. For the highly turbulent Forcing NS (Re=10^5), SINO (2.03E-01 average, 1.98E-01 best) maintains competitive performance despite the extreme Reynolds number and severe underresolution. U-Net (2.37E-01) and UFNO (2.62E-01) trail behind, while Transformer methods and FNO variants exhibit substantially higher errors (4.97-6.45E-01). The coarse-grid baseline error (6.45E-01) confirms that this configuration represents a highly challenging closure problem where most methods struggle.

Analysis. The supplementary experiments reveal three key insights: (1) *Cross-resolution robustness*: SINO consistently achieves the best or near-best performance across all tested coarse-graining ratios, from moderate downsampling (factors 4-8) to extreme coarsening (factor 16), demonstrating that its scale-invariant representations generalize across information loss regimes. (2) *Forcing geometry invariance*: SINO's superior performance on Taylor-Green forcing compared to Kolmogorov forcing (1.01E-02 vs. 9.17E-02) confirms that learned closures capture fundamental turbulent mechanisms rather than memorizing forcing-specific patterns, a critical requirement for practical LES applications where forcing configurations vary across problems. (3) *High-Reynolds extrapolation*: SINO maintains reasonable performance even at Re=10^5 (2.03E-01), despite training on substantially lower Reynolds numbers, suggesting that the learned low-rank representations encode Reynolds-number-independent physical laws governing energy transfer and dissipation. These results collectively validate SINO's design philosophy: by learning on normalized physical scales through implicit low-rank parameterization, the model achieves robust generalization across resolutions, forcing configurations, and turbulence intensities.

A.5.2 Low-Rank Structure Across Different Flow Regimes

To verify that the low-rank structure induced by SINO's bottleneck architecture is a universal property rather than specific to a single benchmark, we extend the principal component analysis (PCA) of learned convolution kernels to additional flow configurations. Figure 7 and Figure 8 present cumulative variance analysis for Forcing NS (Re=1000) and Decaying NS respectively, complementing the Forcing NS (Re=4000) results in Figure 2.

Across all three configurations, the spatial branch kernels consistently exhibit extremely low-rank structure: the cumulative variance exceeds 95% after the third principal component in all layers, confirming that spatial convolution operators concentrate their expressiveness into a minimal subspace regardless of Reynolds number or forcing conditions. For the frequency branch, the majority of layers achieve $>$95% cumulative variance by the third component, while a small fraction of layers reach cumulative variance between 90%-95% after the third component and surpass 95% by the fourth component. This slight variation in the frequency domain reflects the different spectral energy distributions across flow regimes: forced flows at Re=1000 exhibit smoother energy spectra concentrated at low wavenumbers, while Re=4000 and decaying flows contain broader spectral content requiring slightly more principal components to capture high-wavenumber contributions.

Nevertheless, the consistency across all tested configurations validates our theoretical analysis in Section A.4: SINO's bottleneck hypernetwork architecture enforces hard rank constraints (Theorem 1) that force the model to learn compact representations of scale-invariant physical operators. The fact that 2-4 principal components suffice to explain $>$95% variance across spatial and frequency branches, compared to the total dimensionality $D \times D = 32 \times 32 = 1024$ of the kernel matrices, demonstrates a compression ratio exceeding 250×. This empirical confirmation of low-rank structure explains SINO's superior parameter efficiency and data efficiency: by restricting learned operators to low-dimensional subspaces aligned with dominant physical modes, the model avoids overfitting to high-dimensional noise inherent in underresolved coarse-grid observations.

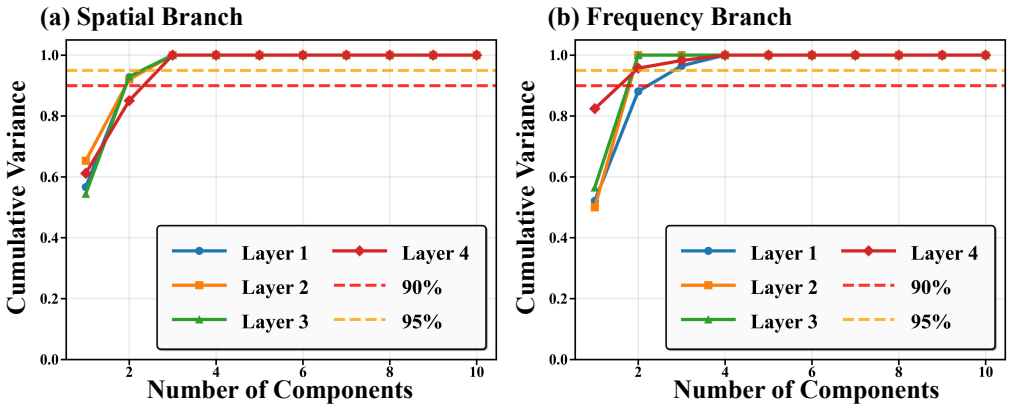


Figure 7: Cumulative variance explained by principal components of learned convolution kernels on Forcing NS (Re=1000). **(a) Spatial branch** and **(b) Frequency branch** across 4 layers. Consistent with Re=4000 results, the spatial branch achieves >95% cumulative variance by the third component across all layers, while the frequency branch exhibits >95% variance by the third or fourth component, confirming low-rank structure across different Reynolds numbers.

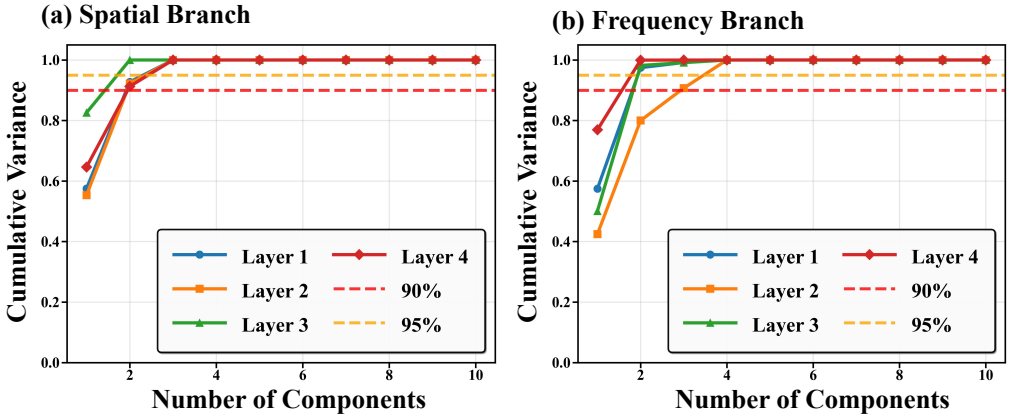


Figure 8: Cumulative variance explained by principal components of learned convolution kernels on Decaying NS. **(a) Spatial branch** and **(b) Frequency branch** across 4 layers. The spatial branch maintains >95% cumulative variance by the third component, while the frequency branch achieves 90%-95% variance by the third component and surpasses 95% by the fourth component in most layers, demonstrating that low-rank structure persists even in freely decaying turbulence without external forcing.

A.5.3 Bottleneck Dimension Scaling and Computational Trade-offs

To investigate the impact of bottleneck layer dimensions on model performance, we conduct ablation studies varying the bottleneck dimensions (n_f, n_s) from 2 to 32 on two representative 2D benchmarks: Forcing NS (Re=1000) and Decaying NS. All SINO variants use the same dual-branch architecture with hidden dimension $D = 32$ and $L = 4$ layers, trained on coarse-grid resolution 64×64 derived from DNS at 512×512.

Table 6 shows that increasing bottleneck capacity consistently improves performance. SINO-32 achieves the lowest errors (4.08E-02 on Decaying NS, 6.06E-02 on Forcing NS), representing 1.5-1.7× improvement over the baseline SINO-2 configuration. Remarkably, even SINO-2 substantially outperforms DNS-128 (2× higher resolution with 4× more grid points), while SINO-32 approaches or exceeds the accuracy of DNS-256 (4× higher resolution with 16× more grid points). This demonstrates that SINO achieves over 16× computational acceleration: instead of refining the coarse grid from 64×64 to 256×256, practitioners can run coarse-grid simulations at 64×64 with SINO correction, achieving comparable or superior accuracy with minimal overhead.

Table 6: Bottleneck dimension scaling analysis on 2D benchmarks. SINO variants trained on 64×64 coarse grids are compared against uncorrected DNS at resolutions 64×64, 128×128, and 256×256. All errors measured against ground-truth DNS-512.

	SINO (64×64)				Coarse DNS		
Benchmark	**SINO-2**	**SINO-8**	**SINO-16**	**SINO-32**	**64×64**	**128×128**	**256×256**
Decaying NS	5.99E-02	4.98E-02	4.49E-02	**4.08E-02**	4.04E-01	1.88E-01	5.64E-02
Forcing NS (Re=1000)	9.17E-02	6.89E-02	6.67E-02	**6.06E-02**	5.07E-01	2.58E-01	6.47E-02

To further analyze temporal error accumulation, Figure 9 visualizes the error growth curves of SINO-32 compared to coarse DNS baselines over 1000 time steps on both Forcing NS (Re=1000) and Decaying NS benchmarks. The red dashed line at time step 300 marks the boundary between the training region (0–300) and the extrapolation region (300–1000). SINO-32 demonstrates remarkable stability in long-time extrapolation, exhibiting the slowest error growth rate among all methods. Notably, SINO-32 at 64×64 resolution not only outperforms DNS-128 and DNS-256 throughout the entire trajectory, but also maintains error levels below DNS-256 even in the challenging extrapolation regime beyond the training horizon. This superior temporal stability, combined with the 16× computational speedup, validates SINO's practical value for long-term turbulence prediction.

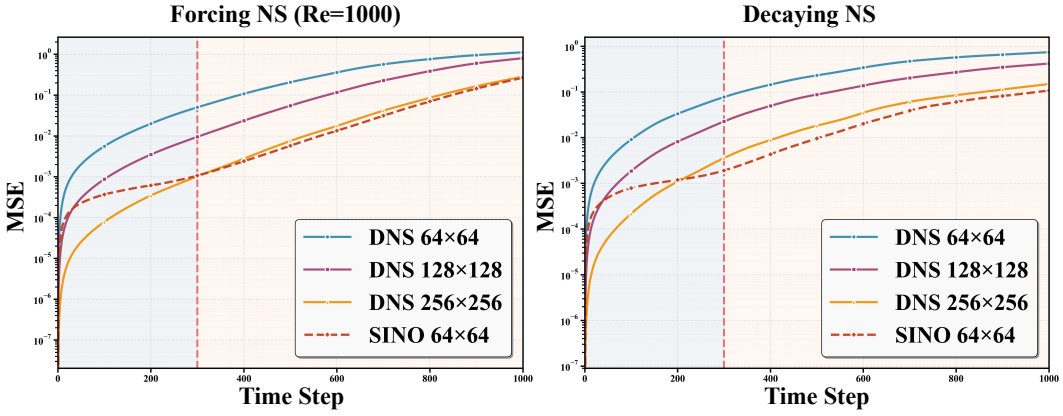


Figure 9: Temporal error evolution on Forcing NS (Re=1000, left) and Decaying NS (right). SINO-32 at 64×64 resolution exhibits slower error growth than DNS at 128×128 and 256×256, particularly in the extrapolation region (shaded, beyond time step 300). The log-scale y-axis highlights SINO's superior long-term stability.