SDHilb: Schrödinger Dynamics-Guided Neural Network with Adaptive Multi-Scale Hilbert Transform for Time Series Forecasting

Caihua He **Weiyang Ding***
Institute of Science and Technology for Brain-Inspired Intelligence
Fudan University, Shanghai, China
Key Laboratory of Computational Neuroscience and Brain-Inspired Intelligence
(Fudan University), Ministry of Education, China
MOE Frontiers Center for Brain Science
Fudan University, Shanghai, China
hhcaihua1121@163.com dingwy@fudan.edu.cn

Abstract

Conventional real-valued neural networks struggle to explicitly capture phase-frequency coupling, while long-horizon time series forecasting requires models to characterize non-stationary amplitude variations, phase shifts, and frequency-dependent temporal evolution. This mismatch limits the stability and expressiveness of neural models on complex forecasting tasks. To address this challenge, we propose **SDHilb**, a forecasting framework inspired by Schrödinger dynamics that lifts real-valued observations into an adaptive complex state space and models temporal evolution by jointly incorporating amplitude and phase information. Through the Hilbert transform, phase-related structures are explicitly encoded, leading to a coupled evolution of real and quadrature components that motivates Schrödinger-Hamiltonian structured dynamics in the lifted complex space. Based on this formulation, SDHilb integrates Schrödinger-guided linear dynamics with an adaptive multi-scale Hilbert transform and introduces three key components: (1) a linear forecasting module with learned coupling of real and quadrature components, motivated by a symplectic discretization of Schrödinger dynamics, (2) an adaptive multi-scale Hilbert module that constructs data-dependent quadrature components through Hilbert-initialized filters and learned channel-wise scale weighting, and (3) a theoretically grounded parameter reuse mechanism derived from the closed-form recurrence structure. Experiments on eight real-world time series forecasting benchmarks demonstrate that SDHilb achieves competitive or superior performance on both short- and long-term forecasting tasks while reducing horizon-specific parameterization through recurrence-guided parameter reuse.

1 Introduction

Long-horizon time series forecasting is a central problem in sequence modeling, with broad applications in energy systems, transportation, weather forecasting, finance, healthcare, and industrial monitoring. A key challenge is that real-world multivariate time series are rarely governed by stationary temporal correlations alone. They often involve coupled amplitude variations, phase shifts, seasonal misalignment, abrupt changes, and frequency-dependent evolution. These properties make

*Corresponding author.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

long-horizon prediction difficult, especially when the model must preserve coherent temporal structure over many future steps.

Recent neural forecasting models have advanced this area through improved temporal mixing, decomposition, channel modeling, and frequency-domain representations, including recurrent neural networks such as LSTM [1], temporal convolutional networks such as TCN [2], Transformer-based architectures [3, 4, 5], as well as SCINet [6] and Crossformer [7].

However, most methods still operate primarily on real-valued observations and treat amplitude- and phase-related temporal patterns only implicitly. This limits their ability to represent non-stationary dynamics where amplitude changes and phase shifts interact.

Hilbert-based analytic representations offer a natural way to pair an observed real-valued signal with a quadrature component, thereby forming a complex-valued state in which amplitude and phase can be jointly represented. However, directly applying fixed Fourier or Hilbert operators is often not sufficiently adaptive for real-world forecasting tasks, since temporal patterns may be superimposed across datasets, variables, and prediction horizons [8, 9, 10]. This motivates an adaptive Hilbert-based construction that preserves amplitude-phase interpretability while learning data-dependent quadrature components.

Recent neurodynamical studies further suggest that Hilbert-induced latent components can be interpreted as conjugate variables and used to motivate Schrödinger-Hamiltonian complex dynamics [11]. Building on this perspective, we propose **SDHilb**, a Schrödinger dynamics-guided neural forecasting framework with an adaptive multi-scale Hilbert transform. SDHilb first constructs data-dependent quadrature components across variables and temporal scales, producing a lifted complex state for amplitude-phase modeling. It then evolves this state using a Schrödinger-Hamiltonian motivated dynamic. We use this dynamic as a structure-preserving inductive bias in the lifted complex state space; it does not assume that raw forecasting benchmarks are conservative physical systems. Instead, the conservation structure regularizes latent amplitude-phase evolution after Hilbert-based lifting.

The discretized dynamics further leads to a horizon-efficient parameter reuse strategy. By unrolling the symplectic recurrence, future real components can be represented through repeated applications of a shared transition structure and alternating contributions from the initial real and imaginary states. From this closed-form representation, we derive a parameter-reuse formulation in which future states at different steps are expressed through matrix polynomials of the same dynamics matrix applied to the initial real and quadrature components.

Our contributions are summarized as follows:

- We formulate the network dynamics using the general form of the Schrödinger equation: $\mathbf{i}\frac{dx}{dt} = Ax, \quad x = x_r + \mathbf{i}x_i \in \mathbb{C}^d$. This formulation provides a principled framework for modeling temporal evolution, where x_r and x_i correspond to generalized coordinates and momenta in Hamiltonian mechanics.
- We introduce a learnable multi-scale adaptive Hilbert module that constructs data-dependent quadrature components across frequency scales. The module uses Gumbel-Softmax weighted depthwise convolutions to adapt the effective Hilbert receptive field across channels and datasets.
- We derive a neurodynamic-inspired parameter reuse strategy from the closed-form recurrence of the discretized Schrödinger dynamics, allowing learned projection weights to be reused for longer forecasting horizons (Eq. 5).

The overall architecture of SDHilb is illustrated in Figure 1.

2 Related Work

2.1 Complex-Valued and Hilbert-Based Representations

Complex-valued neural networks (CVNNs) have attracted growing attention in recent years due to their intrinsic ability to jointly encode real and imaginary components. These networks employ complex-valued parameters and activation functions to learn from complex-domain data [12, 13, 14]. However, their practical deployment still faces several challenges. First, for time series forecasting,

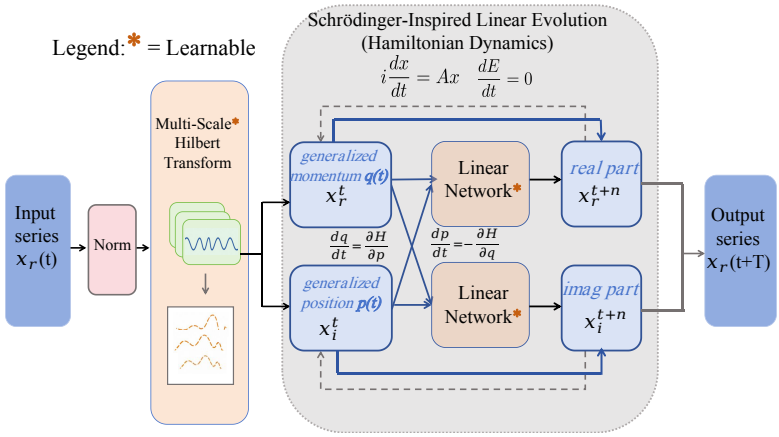


Figure 1: Architecture of SDHilb: a structure-preserving forecasting framework that combines an adaptive multi-scale Hilbert transform with Schrödinger-Hamiltonian motivated complex dynamics.

raw observations are usually real-valued signals. Therefore, the key challenge is not simply to introduce complex-valued parameters, but to construct and interpret meaningful complex-valued states. Classical spectral tools such as the Fourier transform and the Hilbert transform can reveal frequency, amplitude, and phase information of a single sequence [8, 9, 10]. Nevertheless, when phase patterns vary across variables, datasets, and forecasting horizons, these tools may be insufficient to capture the superposition of different state components. Second, there is still no consensus on training protocols for complex-valued networks, including the design of activation functions, parameter initialization, and optimization stability [15, 16, 17]. This makes CVNNs more prone to training instability and less generalizable across tasks [18]. Moreover, existing methods often directly convert neurons and input data into complex numbers, using complex values mainly as computational tools rather than as a way to model the underlying physical dynamics of complex systems [19].

Overall, existing complex-valued methods primarily use complex numbers as computational representations, while the relationship between complex-valued states and structured temporal evolution remains under-explored.

2.2 Neural Time Series Forecasting Models

Modern neural forecasting models improve long-horizon prediction from different perspectives. Transformer-based methods model temporal dependencies through decomposition, patching, or cross-variable attention mechanisms, including Autoformer [4], FEDformer [5], PatchTST [20], and Crossformer [7]. Structured state space models, such as S4 [21] and Mamba [22], achieve efficient sequence modeling through recurrent state-space formulations. Frequency-domain methods, including TimesNet [23], FITS [24], and FreDF [25], exploit periodic or spectral patterns, while lightweight models such as DLinear [26] and LightTS [27] improve efficiency through simple projection structures. Operator- or dynamics-inspired methods such as Koopa [28] further introduce structural assumptions on temporal evolution.

For a more detailed discussion, please refer to Appendix B. To address these challenges, we propose **SDHilb**, which constructs an analytic state through an adaptive Hilbert module and uses the Schrödinger-Hamiltonian form as an inductive bias for stable amplitude-phase evolution. In this way, amplitude-phase coupling is modeled as part of the forecasting dynamics rather than merely as a feature transformation, enabling a balance among interpretability, efficiency, and predictive performance in complex time series forecasting.

3 Methodology

3.1 Motivation: Why Hilbert and Complex-Valued States?

Hilbert-based complex representations provide a principled way to lift real-valued observations into a state space where amplitude, phase, and frequency-related information can be modeled jointly. In such a representation, the modulus reflects amplitude or energy intensity and can help localize abrupt changes such as surges and drops, while the phase angle describes periodicity and lead-lag relations such as seasonal shifts. The instantaneous frequency, defined as $\omega(t) = d\phi(t)/dt$, further characterizes non-stationary temporal evolution beyond purely global Fourier decomposition or fixed-basis wavelet analysis.

Classical Hilbert analysis provides a useful reference point for constructing this representation. For a real-valued signal $x(t)$, the analytic signal

$$z(t) = x(t) + \mathbf{i}\mathcal{H}[x](t) \tag{1}$$

pairs the observed signal with a quadrature component, where $|z(t)|$ and $\arg z(t)$ correspond to instantaneous amplitude and instantaneous phase. This explains why Hilbert-based complex representations are suitable for time series whose amplitude variations, phase shifts, and frequency-dependent patterns are coupled. The formal definition of the Hilbert transform and its frequency-domain interpretation are provided in Appendix I.

Building on this perspective, SDHilb generalizes the fixed Hilbert transform into an adaptive multi-scale module that learns data-dependent quadrature components across variables and horizons. This adaptive construction retains the interpretability of analytic signals while extending them to multivariate forecasting scenarios with heterogeneous temporal scales. The resulting lifted complex state provides a natural interface for structure-preserving temporal evolution. Recent neurodynamical studies suggest that Hilbert-induced latent components can be interpreted as conjugate variables and used to motivate Schrödinger-Hamiltonian-like complex dynamics [29, 11]. This connection motivates a structure-preserving evolution rule in the lifted complex state space, where the conservation structure regularizes latent amplitude-phase dynamics over long horizons. The conservation structure is imposed on the lifted state rather than assumed for the raw observed sequence.

3.2 Adaptive Multi-Scale Hilbert Transform

To capture different temporal scales, SDHilb constructs a bank of filters with widths $w_k \in \{15, 31, 63, 127\}$. The filter coefficients are initialized using Hann-windowed antisymmetric Hilbert kernels and can be updated through backpropagation to learn data-adaptive quadrature representations. At initialization,

$$h_k[n] = \begin{cases} \dfrac{\mathrm{Hann}(n)}{\pi(n - \lfloor w_k/2 \rfloor)}, & n \neq \lfloor w_k/2 \rfloor, \\ 0, & n = \lfloor w_k/2 \rfloor. \end{cases}$$

The initialization provides an antisymmetric Hilbert filter, while the window tapering reduces spectral artifacts in finite-length sequences. The Hilbert-transform definition, causality discussion, and additional kernel details are provided in Appendix I.

For each channel c, the model learns channel-specific multi-scale weights through a Gumbel-Softmax selector,

$$\alpha_{c,k} = \frac{\exp((W_{c,k} + g_{c,k})/\tau)}{\sum_{j=1}^{K} \exp((W_{c,j} + g_{c,j})/\tau)}. \tag{2}$$

where $g_{c,k}$ is sampled during training and set to zero at inference. The adaptive Hilbert component is then computed as

$$\mathcal{H}_{\mathrm{multi}}(X) = \sum_{k=1}^{K} \alpha_{c,k} \cdot \mathrm{DepthwiseConv1D}(X, h_k; \mathrm{groups} = C).$$

This produces data-adaptive quadrature components that extend analytic-signal modeling from a fixed single-scale transform to channel-specific multi-scale temporal representations. The resulting lifted complex state is

$$\mathbf{x} = \mathbf{x}_r + \mathbf{i}\mathbf{x}_i, \quad \mathbf{x}_r = X, \quad \mathbf{x}_i = \mathcal{H}_{\mathrm{multi}}(X).$$

Thus, the real component retains the observed temporal information, while the imaginary component provides an adaptive phase-related counterpart for subsequent complex-valued dynamics.

3.3 Discretized Schrödinger Dynamics Implementation

After obtaining the adaptive lifted state $\mathbf{x} = \mathbf{x}_r + \mathbf{i}\mathbf{x}_i$, SDHilb evolves it using a Schrödinger-type linear complex dynamic:

$$\mathbf{i}\frac{d\mathbf{x}}{dt} = \mathbf{A}\mathbf{x}, \quad \mathbf{A} = \mathbf{A}_r + \mathbf{i}\mathbf{A}_i. \tag{3}$$

Separating real and imaginary components yields

$$\frac{d\mathbf{x}_r}{dt} = \mathbf{A}_r\mathbf{x}_i + \mathbf{A}_i\mathbf{x}_r, \quad \frac{d\mathbf{x}_i}{dt} = -\mathbf{A}_r\mathbf{x}_r + \mathbf{A}_i\mathbf{x}_i.$$

Under the sufficient conditions $\mathbf{A}_r = \mathbf{A}_r^\top$ and $\mathbf{A}_i = 0$, the system reduces to

$$\frac{d\mathbf{x}_r}{dt} = \mathbf{A}_r\mathbf{x}_i, \quad \frac{d\mathbf{x}_i}{dt} = -\mathbf{A}_r\mathbf{x}_r,$$

which corresponds to a Hamiltonian system in the lifted complex state space with

$$H(\mathbf{x}_r, \mathbf{x}_i) = \frac{1}{2}\mathbf{x}_r^\top\mathbf{A}_r\mathbf{x}_r + \frac{1}{2}\mathbf{x}_i^\top\mathbf{A}_r\mathbf{x}_i.$$

Here, $\mathbf{x}_r$ and $\mathbf{x}_i$ can be interpreted as coordinate-like and momentum-like variables in the lifted space. Importantly, this conservation-related structure should be understood as a stability-oriented constraint on the lifted complex state, rather than as a strict conservation law imposed on the raw observed sequence. Thus, SDHilb uses Hamiltonian structure to regularize latent amplitude-phase evolution, without requiring the original real-valued signal to be a conservative physical system. The full derivation and energy-conservation conditions are given in Appendix H.

To discretize the continuous dynamics, we adopt a symplectic Euler update:

$$\begin{cases} \mathbf{x}_r^{(t+\Delta t)} = \mathbf{x}_r^{(t)} + \Delta t \cdot \mathbf{A}_r\mathbf{x}_i^{(t)}, \\ \mathbf{x}_i^{(t+\Delta t)} = \mathbf{x}_i^{(t)} - \Delta t \cdot \mathbf{A}_r\mathbf{x}_r^{(t+\Delta t)}. \end{cases} \tag{4}$$

The key difference from a standard explicit Euler step is the sequential update: the imaginary component is updated using the newly computed real component. This update preserves the symplectic structure of the idealized lifted Hamiltonian system and reduces numerical drift in long-horizon evolution.

Unrolling the symplectic recursion yields a Chebyshev-type polynomial representation for the future real component:

$$\mathbf{x}_r^{(t+n\Delta t)} = \mathcal{U}_n(\mathbf{A}_r)\,\mathbf{x}_r^{(t)} + \Delta t\,\mathbf{A}_r\,\mathcal{V}_n(\mathbf{A}_r)\,\mathbf{x}_i^{(t)}. \tag{5}$$

Let

$$\mathbf{D} = 2\mathbf{I} - (\Delta t)^2\mathbf{A}_r^2.$$

The matrix polynomials $\mathcal{U}_n(\cdot)$ and $\mathcal{V}_n(\cdot)$ are induced by the second-order recurrence

$$P_{n+1} = \mathbf{D}\,P_n - P_{n-1}, \tag{6}$$

with initial conditions

$$\mathcal{U}_0 = \mathcal{U}_1 = \mathbf{I}, \quad \mathcal{V}_0 = \mathbf{0}, \quad \mathcal{V}_1 = \mathbf{I}.$$

Equivalently, for $n \geq 1$,

$$\begin{aligned} \mathcal{V}_n(\mathbf{A}_r) &= \sum_{j=0}^{\lfloor (n-1)/2 \rfloor} (-1)^j \binom{n-1-j}{j} \mathbf{D}^{\,n-1-2j}, \\ \mathcal{U}_n(\mathbf{A}_r) &= \mathcal{V}_n(\mathbf{A}_r) - \mathcal{V}_{n-1}(\mathbf{A}_r), \end{aligned} \tag{7}$$

where $\mathcal{V}_0(\mathbf{A}_r) = \mathbf{0}$ and $\mathcal{U}_0(\mathbf{A}_r) = \mathbf{I}$. Both $\mathcal{U}_n$ and $\mathcal{V}_n$ are polynomials in $\mathbf{A}_r$ whose coefficients depend only on n and the fixed step size Δt. Therefore, future states are structured linear combinations of the initial real and imaginary components governed by a shared, n-indexed transition structure. This expression provides the mathematical basis for the parameter reuse strategy below, while the detailed induction proof and its connection to matrix trigonometric functions are deferred to Appendix J.

3.4 Horizon-Efficient Parameter Reuse

Forecasting over extended horizons often requires a large number of horizon-specific parameters, especially when each future step is predicted by an independent output head. In contrast, the unrolled Schrödinger-style recurrence reveals a reusable transition pattern shared across future steps. Each future real component can be represented as a structured combination of the initial real and imaginary components, with coefficients induced by the repeated symplectic transition.

Based on this observation, SDHilb learns two shared projection layers:

$$\begin{bmatrix} \mathbf{x}_r^{(t+\Delta t)} \\ \vdots \\ \mathbf{x}_r^{(t+n\Delta t)} \end{bmatrix} = \mathbf{W}_1^{\text{symp}}(n)\mathbf{x}_r^{(t)} + \mathbf{W}_2^{\text{symp}}(n)\mathbf{x}_i^{(t)}. \tag{8}$$

From the closed-form symplectic recurrence, we derive a parameter reuse strategy implemented through two directly learned projection maps. When extending the forecasting horizon, previously learned projection weights are retained and frozen, while trainable rows are appended for additional future steps. This enables cross-horizon weight reuse and reduces the number of projection weights requiring optimization during horizon extension. The symplectic properties are established for the underlying recurrence. The detailed connection between Eq. (4), the closed-form recurrence, and Eq. (8) is provided in Appendix J.

4 Experiment

Datasets and Baselines. We evaluate our proposed method on 8 widely used long-term forecasting benchmark datasets, including Weather, Traffic, Electricity, ILI, and 4 ETT variants (ETTh1/ETTh2/ETTm1/ETTm2). The statistical information of these datasets is summarized in Table 1; these datasets are from [4] and [23] for time series forecasting benchmarking. They cover diverse real-world scenarios, including meteorological measurements, traffic occupancy, electricity consumption, influenza-like illness records, and electricity transformer temperature, with different numbers of variables, sampling frequencies, seasonal patterns, and noise levels.

We compare our method with several representative time series forecasting models: DLinear (2023) [26] and FITS (2024) [24] as lightweight linear or spectral models; FEDformer (2022) [5] and FreDF (2025) [25] as frequency-domain or decomposition-based models; PatchTST (2023) [20] and TimesNet (2023) [23] as strong neural forecasting architectures; and Koopa (2023) [28] as a Koopman-inspired model. Additionally, we further compare with state-space models S4 (2021) [21], Mamba (2023) [22], CVNN(Transformer) [30], and recent frequency-domain methods *(results provided in Appendix D)*. All models are evaluated under the same data splits and metrics. When official implementations are available, we use the configurations recommended or reported by the original papers to ensure fair comparison.

Table 1: Dataset statistics.

Datasets	Weather	Traffic	Electricity	ILI	ETTh1(2)	ETTm1(2)
Features	21	862	321	7	7	7
Timesteps	52696	17544	26304	966	17420	69680

Implementation Details. To ensure fair comparison across forecasting architectures, we follow model-specific input-length settings based on prior findings and reported configurations. DLinear, FITS, Koopa, FreDF, PatchTST, and SDHilb use $L = 336$, while FEDformer and TimesNet retain their default $L = 96$ setting to reduce overfitting under long inputs; input-length ablations in Appendix E.1 further verify the fairness of this setting. All models are trained with the **Adam** optimizer [31] and evaluated using **MSE** and **MAE** with early stopping. Hyperparameters are selected on the validation set, and each main forecasting comparison is repeated three times with fixed random seeds. Additional training configurations, reproducibility protocols, and hardware details are provided in Appendix C.

Table 2: Performance comparison of different models on multiple datasets. Best results are **bold**, second-best are underlined.

Models	Ours		FITS		DLinear		PatchTST		TimesNet		Koopa		FEDFormer		FreDF	
Metric	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Etth1																
96	0.370	**0.393**	0.380	0.400	0.430	0.457	**0.368**	0.394	0.389	0.412	0.398	0.419	0.376	0.415	0.397	0.418
192	**0.405**	**0.413**	0.412	0.418	0.445	0.457	0.414	0.421	0.476	0.467	0.428	0.435	0.423	0.446	0.447	0.455
336	**0.429**	**0.431**	0.434	0.432	0.458	0.460	0.431	0.436	0.550	0.511	0.458	0.457	0.444	0.462	0.480	0.475
720	**0.428**	0.452	0.429	**0.449**	0.508	0.518	0.450	0.466	0.570	0.525	0.464	0.480	0.469	0.492	0.544	0.534
Avg	**0.408**	**0.422**	0.414	0.425	0.460	0.473	0.416	0.429	0.496	0.479	0.437	0.448	0.428	0.454	0.467	0.471
Etth2																
96	**0.272**	**0.335**	0.276	0.339	0.283	0.348	0.281	0.339	0.338	0.372	0.308	0.361	0.332	0.374	0.311	0.371
192	**0.327**	**0.374**	0.337	0.377	0.367	0.402	0.338	0.378	0.410	0.417	0.384	0.405	0.407	0.446	0.404	0.435
336	**0.335**	**0.387**	0.360	0.396	0.436	0.455	0.347	0.391	0.465	0.456	0.397	0.422	0.400	0.447	0.462	0.474
720	**0.380**	**0.420**	0.386	0.423	0.631	0.564	0.381	0.422	0.433	0.447	0.432	0.451	0.412	0.469	0.770	0.628
Avg	**0.329**	**0.379**	0.340	0.384	0.429	0.442	0.337	0.383	0.412	0.423	0.380	0.410	0.388	0.434	0.487	0.477
Ettm1																
96	0.290	**0.338**	0.303	0.345	0.313	0.362	**0.288**	0.340	0.348	0.381	0.308	0.360	0.326	0.390	0.307	0.353
192	0.331	**0.361**	0.337	0.365	0.344	0.379	**0.330**	0.369	0.407	0.414	0.343	0.381	0.365	0.415	0.347	0.377
336	0.371	**0.384**	0.372	0.385	0.376	0.393	**0.366**	0.390	0.464	0.441	0.371	0.399	0.392	0.425	0.381	0.398
720	0.428	0.417	0.428	**0.416**	0.440	0.439	**0.418**	0.423	0.541	0.481	0.428	0.430	0.446	0.458	0.454	0.449
Avg	0.355	**0.375**	0.360	0.378	0.368	0.393	**0.351**	0.381	0.440	0.429	0.363	0.393	0.382	0.422	0.372	0.394
Ettm2																
96	**0.161**	**0.250**	0.165	0.255	0.169	0.260	0.164	0.253	0.191	0.268	0.178	0.265	0.180	0.271	0.173	0.263
192	**0.217**	**0.289**	0.221	0.292	0.237	0.315	0.224	0.295	0.268	0.315	0.242	0.307	0.252	0.318	0.244	0.310
336	**0.271**	**0.325**	0.274	0.326	0.308	0.369	0.281	0.331	0.330	0.354	0.298	0.344	0.324	0.364	0.324	0.362
720	**0.366**	**0.382**	0.367	0.383	0.398	0.419	**0.366**	0.384	0.435	0.415	0.376	0.394	0.410	0.420	0.414	0.427
Avg	**0.254**	**0.311**	0.257	0.314	0.278	0.341	0.259	0.316	0.306	0.338	0.274	0.328	0.292	0.343	0.289	0.341
Weather																
96	**0.312**	**0.281**	0.391	0.322	0.359	0.329	0.328	0.288	0.340	0.294	0.319	0.283	0.388	0.356	0.329	0.298
192	**0.388**	0.333	0.432	0.355	0.425	0.371	0.402	0.336	0.445	0.352	0.392	**0.332**	0.489	0.405	0.435	0.371
336	**0.457**	**0.372**	0.503	0.389	0.492	0.413	0.469	0.376	0.540	0.398	0.467	0.378	0.580	0.438	0.510	0.415
720	**0.559**	**0.417**	0.595	0.429	0.594	0.514	0.567	0.420	0.658	0.444	0.569	0.425	0.676	0.471	0.592	0.464
Avg	**0.429**	**0.351**	0.480	0.374	0.467	0.407	0.442	0.355	0.496	0.372	0.437	0.354	0.533	0.417	0.467	0.387
Traffic																
96	0.410	**0.281**	0.457	0.330	0.410	0.282	**0.400**	0.284	0.593	0.321	0.406	0.296	0.576	0.359	0.409	0.286
192	0.422	**0.284**	0.469	0.334	0.423	0.287	**0.416**	0.289	0.617	0.336	0.431	0.304	0.610	0.380	0.436	0.291
336	0.435	**0.290**	0.479	0.338	0.435	0.295	**0.429**	0.297	0.629	0.336	0.458	0.338	0.608	0.375	0.436	0.299
720	0.464	**0.308**	0.502	0.349	0.468	0.315	**0.461**	0.315	0.640	0.350	0.472	0.398	0.621	0.375	0.473	0.318
Avg	0.433	**0.291**	0.477	0.338	0.434	0.295	**0.426**	0.296	0.620	0.336	0.442	0.334	0.604	0.372	0.439	0.299
Electricity																
96	**0.140**	**0.225**	0.147	0.234	0.146	0.236	0.142	0.229	0.264	0.337	0.146	0.241	0.186	0.302	**0.140**	0.231
192	**0.155**	**0.238**	0.162	0.246	0.161	0.249	0.162	0.247	0.253	0.331	0.166	0.257	0.197	0.311	0.158	0.248
336	**0.178**	**0.255**	0.185	0.262	0.183	0.268	0.185	0.265	0.279	0.350	0.189	0.275	0.213	0.328	0.184	0.270
720	**0.203**	**0.278**	0.212	0.286	0.209	0.297	0.207	0.282	0.302	0.365	0.224	0.305	0.233	0.344	0.217	0.300
Avg	**0.169**	**0.249**	0.176	0.257	0.175	0.263	0.174	0.256	0.275	0.346	0.181	0.270	0.207	0.321	0.175	0.262
ILI																
24	**1.462**	0.866	2.468	1.094	2.215	1.081	1.522	**0.814**	2.317	0.934	1.990	0.889	2.624	1.095	2.110	1.022
36	1.453	0.884	2.440	1.093	1.963	0.963	**1.430**	**0.834**	1.972	0.920	1.866	0.862	2.516	1.021	2.255	1.048
48	**1.516**	0.867	3.321	1.322	2.130	1.024	1.673	**0.854**	2.238	0.940	1.883	0.899	2.505	1.041	2.446	1.100
60	1.722	0.932	2.955	1.206	2.368	1.096	**1.529**	**0.862**	2.027	0.928	1.909	1.306	2.742	1.122	2.644	1.168
Avg	**1.538**	0.887	2.796	1.179	2.169	1.041	**1.538**	**0.841**	2.139	0.930	1.912	0.989	2.597	1.070	2.364	1.085

4.1 Main Results

Table 2 compares different models across eight benchmark datasets and multiple forecasting horizons. SDHilb achieves the lowest or highly competitive average MSE and MAE on most datasets, showing strong generalization across diverse forecasting scenarios. On Weather with a 720-step horizon, SDHilb achieves the lowest MSE (0.559) among all models compared in Table 2, demonstrating its effectiveness for long-horizon forecasting.

On the challenging ILI dataset, where sequences are shorter, noisier, and more sparsely observed, SDHilb matches PatchTST's best MSE and achieves the second-best MAE while maintaining clear

Table 3: Parameter and MAC comparison on Electricity.

Model	Params	MACs
FEDformer	18.44M	4.54G
FreDF	11.16M	3.56G
Koopa	9.13M	1.16G
PatchTST	2.21M	5.95G
TimesNet	1.44M	4.00G
DLinear	226.46K	72.59M
FITS	51.52K	32.87M
SDHilb(Ours)	303.50K	170.41M

Table 4: Core component ablation on ETTh1.

Case	Components				ETTh1	
	A	**B**	**C**	**D**	**MSE**	**MAE**
Ours	✓	✓	×	✓	**0.429**	**0.431**
Case 1	×	✓	×	×	0.436	0.440
Case 2	×	×	×	×	0.435	0.435
Case 3	✓	✓	×	×	0.442	0.447
Case 4	×	✓	✓	×	0.452	0.451
Case 5	✓	×	×	✓	0.447	0.448

A: Multi-Scale Hilbert B: Structured Complex Dynamics
C: FC replacement D: Hann window filter

advantages over both lightweight and heavier baselines. The strong performance reflects the benefit of combining a Hilbert-lifted complex representation with a Schrödinger-Hamiltonian motivated recurrence: the former encodes amplitude-phase information through real and learned quadrature components, while the latter provides a structure-guided inductive bias for stable long-horizon evolution. These results demonstrate the robustness of SDHilb across multivariate, high-frequency, noisy, and long-horizon forecasting settings. Additional studies on forecasting length sensitivity are reported in Appendix E.

4.2 Model Efficiency

In addition to strong forecasting accuracy, SDHilb is highly efficient in architecture-level computational cost. As shown in Table 3, on the Electricity dataset with look-back window $L = 336$ and prediction horizon $H = 336$, SDHilb uses only 303.5K parameters and 170.4M MACs, corresponding to *98.4% fewer parameters than FEDformer* and *86.3% fewer than PatchTST*, while requiring only *2.9% of PatchTST's MACs*. Although FITS and DLinear are lighter, SDHilb achieves a favorable balance between forecasting accuracy and computational efficiency.

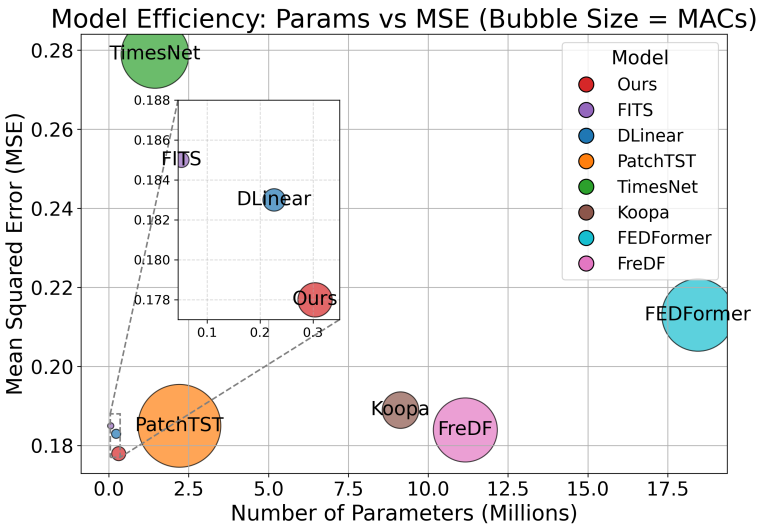


Figure 2: Comparison between mean squared error (MSE) and number of parameters on the Electricity dataset. Each bubble represents a distinct model, with bubble size indicating computational cost measured by MACs (multiply-accumulate operations). Larger bubbles denote more resource-intensive models. SDHilb achieves strong forecasting accuracy with significantly fewer parameters and lower computational cost.

4.3 Ablation Study

We conduct ablation studies to examine how the major design components of SDHilb contribute to forecasting performance. The main ablations focus on the adaptive Hilbert lifting, the lifted complex-valued representation, the structured recurrence, and the parameter reuse mechanism (complete results in Appendix E).

Lookback Window Ablation. We investigate the impact of different lookback window lengths on model performance.

Core Component Ablation. To evaluate the core designs in SDHilb, we ablate the adaptive multi-scale Hilbert module, structured complex dynamics, Hann-windowed filtering, and FC replacement of the complex transformation. Table 4 reports the ETTh1 results with a forecasting horizon of $H = 336$. As shown in Table 4, removing or replacing any major component degrades performance, showing that SDHilb benefits from the coupled design of Hilbert-based lifting, complex-state evolution, and structure-preserving recurrence.

The results further clarify each component's role: adaptive Hilbert lifting selects suitable time-frequency scales, structured complex dynamics models the coupling between real and quadrature components, and Hann-windowed filtering reduces boundary leakage. In this setting, replacing Hilbert-based imaginary-component construction with an FC mapping produces the largest increase in both MSE and MAE, supporting the contribution of Hilbert-based lifting. Removing Hann windowing also increases both errors. Full ablations in Appendix E.3 show consistent trends across datasets, and Appendix F.3 reports lifted-state energy drift as an approximate stability indicator in the Hilbert-lifted complex space.

Architecture Replacement Study. We further replace the Schrödinger-Hamiltonian motivated recurrence with common neural backends after the Hilbert lifting module, including MLP, CNN, LSTM, and Transformer variants. This experiment examines whether the lifted complex representation alone is sufficient, or whether its evolution benefits from the structured form derived from the proposed dynamics. The results show that applying a more expressive nonlinear backend after Hilbert lifting does not improve long-horizon forecasting. These replacements weaken the structured evolution in the lifted complex space, where the real and quadrature components are designed to evolve as coupled variables. The best-performing replacement remains the structured linear recurrence derived from our formulation. This agreement between derivation and empirical behavior supports using a structured linear transition for the lifted dynamics, rather than treating the Hilbert representation as a generic feature extractor followed by an arbitrary nonlinear predictor.

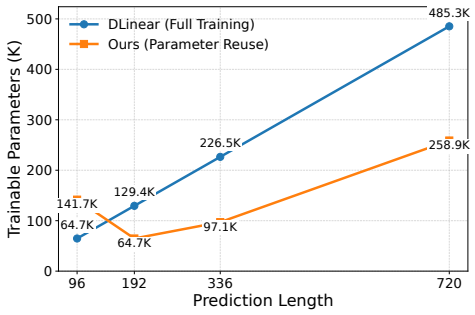
Adaptive Multi-Scale Hilbert Module vs. Standard Hilbert Transform. The Adaptive Multi-Scale Hilbert module extends traditional fixed-kernel Hilbert transform methods with a Gumbel-Softmax-based kernel fusion mechanism. This enables data-dependent quadrature construction, adaptive frequency-scale selection, and channel-wise kernel weighting, while Hann-windowed filtering reduces spectral leakage. Compared with the standard discrete Hilbert transform, the adaptive multi-scale design better captures heterogeneous temporal patterns across variables and datasets. In Appendix F, we provide a detailed comparison with the standard discrete Hilbert transform and analyze learned kernel-weight distributions.

4.4 Parameter Reuse Analysis: Theoretical Motivation and Empirical Efficiency

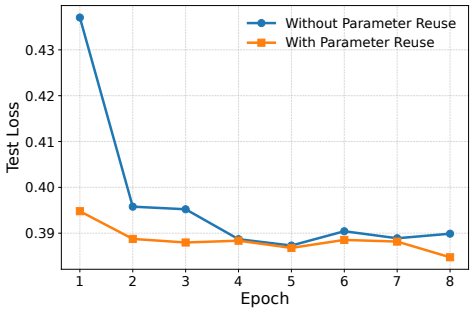
Based on the recurrence structure of Schrödinger-type dynamics (see Eq. 5 and Eq. 8), we propose a theory-driven parameter reuse strategy. The recurrence indicates that multi-step transitions in the lifted complex space are governed by shared coupled mappings over the real and quadrature components. SDHilb implements this coupled linear form through two directly learned projection maps, enabling learned projection weights to be reused when extending the forecasting horizon without explicitly computing higher-order matrix powers.

As shown in Figure 3a, parameter reuse reduces the number of trainable parameters during horizon extension. For the 720-step task, SDHilb requires optimizing only 258.9K parameters, compared with 485.3K for DLinear. Reuse-initialized models also converge faster and more stably than ran-

dom initialization, as shown in Figure 3b. These results show that parameter reuse is a practical approximation of the recurrence-guided transition structure rather than a simple compression trick.



(a) Trainable parameters vs. forecast length under parameter reuse.



(b) Convergence speed comparison. Reuse-initialized models converge faster and more stably than randomly initialized ones.

Figure 3: Benefits of the ODE-guided parameter reuse mechanism: reduced model size and improved training stability.

5 Conclusion

We propose **SDHilb**, a Schrödinger-Hamiltonian motivated forecasting framework that combines adaptive Hilbert lifting with structured complex-state dynamics. By constructing a Hilbert-lifted amplitude-phase representation and evolving it through a real-quadrature coupled transition, SDHilb captures non-stationary temporal patterns while promoting stable long-horizon evolution in the lifted complex space. The closed-form recurrence further yields a cross-horizon reuse strategy for learned projection weights, reducing the number of weights requiring optimization during horizon extension. Experiments on eight real-world benchmark datasets show that SDHilb achieves competitive or superior performance compared with representative linear, frequency-domain, state-space, and Transformer-based baselines, while maintaining favorable computational efficiency. These results support the effectiveness of structure-preserving complex dynamics for efficient and robust time series forecasting.

Acknowledgments and Disclosure of Funding

This work was partially supported by the National Natural Science Foundation of China (12471481 to W.D., U24A2001 to W.D.). The authors declare no competing interests.

References

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

[2] Shaojie Bai, J. Zico Kolter, and Vladlen Koltun. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv preprint arXiv:1803.01271*, 2018.

[3] Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 11106–11115, 2021.

[4] Haixu Wu, Jiehui Xu, Jianmin Wang, and Mingsheng Long. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. In *Advances in Neural Information Processing Systems*, volume 34, 2021.

[5] Tian Zhou, Ziqing Ma, Qingsong Wen, Xue Wang, Liang Sun, and Rong Jin. Fedformer: Frequency enhanced decomposed transformer for long-term series forecasting. In *Proceedings of the International Conference on Machine Learning*, 2022.

[6] Minhao Liu, Ailing Zeng, Muxi Chen, Zhijian Xu, Qiuxia Lai, Lingna Ma, and Qiang Xu. Scinet: Time series modeling and forecasting with sample convolution and interaction. In *Advances in Neural Information Processing Systems*, volume 35, 2022.

[7] Yunhao Zhang and Junchi Yan. Crossformer: Transformer utilizing cross-dimension dependency for multivariate time series forecasting. In *Proceedings of the International Conference on Learning Representations*, 2023.

[8] Norden E. Huang, Zheng Shen, Steven R. Long, Manli C. Wu, H. H. Shih, Quanan Zheng, Nai-Chyuan Yen, Chi Chao Tung, and Hsing Henry Liu. The empirical mode decomposition and the hilbert spectrum for nonlinear and non-stationary time series analysis. *Proceedings of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences*, 454(1971):903–995, 1998.

[9] Leon Cohen. *Time-Frequency Analysis*. Prentice-Hall, 1995.

[10] Ingrid Daubechies. The wavelet transform, time-frequency localization and signal analysis. *IEEE Transactions on Information Theory*, 36(5):961–1005, 1990.

[11] Jiangnan Zhang, Chengyuan Qian, Wenlian Lu, Gustavo Deco, Weiyang Ding, and Jianfeng Feng. Dark signals in the brain: Augment brain network dynamics to the complex-valued field. *arXiv preprint arXiv:2509.24715*, 2025.

[12] Joshua Bassey, Xiangfang Li, and Lijun Qian. A survey of complex-valued neural networks. *arXiv preprint arXiv:2101.12249*, 2021.

[13] Chiheb Trabelsi, Olexa Bilaniuk, Ying Zhang, Dmitriy Serdyuk, Sandeep Subramanian, João Felipe Santos, Soroush Mehri, Negar Rostamzadeh, Yoshua Bengio, and Christopher J. Pal. Deep complex networks. In *Proceedings of the International Conference on Learning Representations*, 2018.

[14] Akira Hirose, editor. *Complex-Valued Neural Networks: Advances and Applications*. Wiley-IEEE Press, 2013.

[15] Igor Aizenberg and Claudio Moraga. Multilayer feedforward neural network based on multi-valued neurons (mlmvn) and a backpropagation learning algorithm. *Soft Computing*, 11(2):169–183, 2007.

[16] Taehwan Kim and Tülay Adali. Fully complex multi-layer perceptron network for nonlinear signal processing. *Journal of VLSI Signal Processing Systems for Signal, Image and Video Technology*, 32:29–43, 2002.

[17] Huisheng Zhang and Danilo P. Mandic. Is a complex-valued stepsize advantageous in complex-valued gradient learning algorithms? *IEEE Transactions on Neural Networks and Learning Systems*, 27(12):2730–2735, 2016.

[18] Nils Mönning and Suresh Manandhar. Evaluation of complex-valued neural networks on real-valued classification tasks. *arXiv preprint arXiv:1811.12351*, 2018.

[19] Călin-Adrian Popa. Complex-valued convolutional neural networks for real-valued image classification. In *Proceedings of the International Joint Conference on Neural Networks*, pages 816–822, 2017.

[20] Yuqi Nie, Nam H. Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. A time series is worth 64 words: Long-term forecasting with transformers. In *Proceedings of the International Conference on Learning Representations*, 2023.

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

[22] Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. In *Proceedings of the Conference on Language Modeling*, 2024.

[23] Haixu Wu, Tengge Hu, Yong Liu, Hang Zhou, Jianmin Wang, and Mingsheng Long. Timesnet: Temporal 2d-variation modeling for general time series analysis. In *Proceedings of the International Conference on Learning Representations*, 2023.

[24] Zhijian Xu, Ailing Zeng, and Qiang Xu. Fits: Modeling time series with 10k parameters. In *Proceedings of the International Conference on Learning Representations*, 2024.

[25] Hao Wang, Licheng Pan, Yuan Shen, Zhichao Chen, Degui Yang, Yifei Yang, Sen Zhang, Xinggao Liu, Haoxuan Li, and Dacheng Tao. Fredf: Learning to forecast in the frequency domain. In *Proceedings of the International Conference on Learning Representations*, 2025.

[26] Ailing Zeng, Muxi Chen, Lei Zhang, and Qiang Xu. Are transformers effective for time series forecasting? In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 11121–11128, 2023.

[27] Tianping Zhang, Yizhuo Zhang, Wei Cao, Jiang Bian, Xiaohan Yi, Shun Zheng, and Jian Li. Less is more: Fast multivariate time series forecasting with light sampling-oriented mlp structures. *arXiv preprint*, arXiv:2207.01186, 2022.

[28] Yong Liu, Chenyu Li, Jianmin Wang, and Mingsheng Long. Koopa: Learning non-stationary time series dynamics with koopman predictors. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

[29] Gustavo Deco, Yonatan Sanz Perl, and Morten L. Kringelbach. Complex harmonics reveal low-dimensional manifolds of critical brain dynamics. *Physical Review E*, 111(1):014410, 2025.

[30] Muqiao Yang, Martin Q. Ma, Dongyu Li, Yao-Hung Hubert Tsai, and Ruslan Salakhutdinov. Complex transformer: A framework for modeling complex-valued sequence. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, 2020.

[31] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *Proceedings of the International Conference on Learning Representations*, 2015.

[32] Shiyang Li, Xiaoyong Jin, Yao Xuan, Xiyou Zhou, Wenhu Chen, Yu-Xiang Wang, and Xifeng Yan. Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

[33] Yong Liu, Guo Qin, Xiangdong Huang, Jianmin Wang, and Mingsheng Long. Timer-xl: Long-context transformers for unified time series forecasting. In *Proceedings of the International Conference on Learning Representations*, 2025.

[34] Xiaoming Shi, Shiyu Wang, Yuqi Nie, Dianqi Li, Zhou Ye, Qingsong Wen, and Ming Jin. Time-moe: Billion-scale time series foundation models with mixture of experts. In *Proceedings of the International Conference on Learning Representations*, 2025.

[35] Xiaowen Ma, Zhen-Liang Ni, Shuai Xiao, and Xinghao Chen. Timepro: Efficient multivariate long-term time series forecasting with variable- and time-aware hyper-state. In *Proceedings of the 42nd International Conference on Machine Learning (ICML)*, volume 267 of *Proceedings of Machine Learning Research*, pages 42096–42111. PMLR, 2025.

[36] Wenzhen Yue, Yong Liu, Xianghua Ying, Bowei Xing, Ruohao Guo, and Ji Shi. Freeformer: Frequency enhanced transformer for multivariate time series forecasting. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 3606–3614, 2025.

[37] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 32, 2019.

[38] Tian Zhou, Peisong Niu, Xue Wang, Liang Sun, and Rong Jin. One fits all: Power general time series analysis by pretrained lm. In *Advances in Neural Information Processing Systems*, volume 36, pages 43322–43355, 2023.

[39] Yao Qin, Dongjin Song, Haifeng Chen, Wei Cheng, Guofei Jiang, and Garrison Cottrell. A dual-stage attention-based recurrent neural network for time series prediction. In *Proceedings of the International Joint Conference on Artificial Intelligence*, 2017.

[40] Guokun Lai, Wei-Cheng Chang, Yiming Yang, and Hanxiao Liu. Modeling long- and short-term temporal patterns with deep neural networks. In *Proceedings of the 41st International ACM SIGIR Conference*, 2018.

A Reproducibility

Code repository: `https://github.com/huahua11211/SDHilb`.

B Overview of Representative Time Series Forecasting Models

This section summarizes representative time series forecasting models and clarifies their architectural characteristics, modeling assumptions, and limitations. The goal is to provide additional background for the baseline selection and to position SDHilb within the broader landscape of forecasting architectures.

Transformer-based forecasting models improve long-range dependency modeling through attention variants and task-specific sequence representations. LogTrans [32] introduces LogSparse self-attention with causal convolution to reduce memory complexity while preserving local structures. Informer [3] uses ProbSparse attention and attention distillation for efficient long-sequence forecasting. Autoformer [4] combines seasonal-trend decomposition with autocorrelation mechanisms to capture periodic components, while FEDformer [5] further introduces frequency-enhanced attention based on Fourier and wavelet transforms. PatchTST [20] segments time series patches and employs channel-independent Transformer blocks to improve long-context modeling. Crossformer [7] models cross-dimension dependencies through attention across variables and time. More recent large forecasting models, such as Timer-XL [33] and Time-MoE [34], further explore decoder-only architectures, long-context modeling, and sparse mixture-of-experts pretraining. Although these models have achieved notable progress, attention-based designs can still introduce substantial computational cost, and their temporal evolution is usually learned implicitly rather than modeled through explicit dynamical principles.

State-space models provide an efficient alternative for long-sequence modeling by parameterizing sequence evolution through structured latent dynamics. S4 [21] introduces structured state-space layers for long-range dependency modeling with favorable computational complexity. Mamba [22] extends this direction with selective state-space mechanisms that adaptively modulate sequence processing. TimePro [35] introduces variable- and time-aware hyper-states within a Mamba-based forecasting architecture. These models are efficient and scalable, but they do not explicitly construct an amplitude-phase representation or model phase-frequency coupling in a Hilbert-lifted complex space.

Frequency-domain and hybrid models exploit spectral representations to capture periodic and oscillatory patterns. TimesNet [23] maps one-dimensional time series into two-dimensional temporal variation representations to model multi-periodicity. FITS [24] performs forecasting through frequency interpolation and learns compact spectral representations. FreDF [25] introduces frequency-domain modeling with spectral alignment, and FreEformer [36] enhances Transformer forecasting in the frequency domain. These methods demonstrate the importance of spectral information, but many of them rely on global transforms or fixed spectral priors, which may be less adaptive to local time-varying phase and frequency patterns.

Linear and lightweight models focus on efficient inference and simple temporal mappings. DLinear [26] decomposes time series into trend and seasonal components and applies simple linear projections, achieving competitive performance with low computational cost. LightTS [27] uses sampling-oriented MLP structures for efficient forecasting. While these models are fast and robust, their simple structures may limit their ability to model coupled amplitude-phase evolution or complex multivariate dependencies.

Operator-inspired models incorporate dynamical-system perspectives into forecasting. Koopa [28] leverages Koopman operator theory to decompose temporal dynamics into structured components, providing an interpretable way to model temporal evolution. Such methods are closely related to SDHilb in their use of dynamical-system motivation, but they differ in representation construction and transition modeling.

SDHilb is complementary to the above model families. Unlike Transformer-based models, SDHilb does not rely on quadratic attention for long-range dependency modeling. Unlike standard frequency-domain methods, it constructs a Hilbert-lifted complex representation that explicitly encodes real and quadrature components. Unlike general state-space or operator-inspired models, SD-

Hilb uses Schrödinger-Hamiltonian motivated dynamics to model the evolution of the lifted complex state. This design enables SDHilb to combine amplitude-phase modeling, structure-preserving recurrence, and parameter reuse, leading to a favorable balance between forecasting accuracy and computational efficiency.

C Experimental Details

C.1 Implementation Details

All baseline models in this study were reproduced using the official open-source implementations whenever available. For each baseline, we followed the hyperparameter settings recommended by the original papers or official repositories, and verified reproducibility before including the results. Following our experimental protocol, DLinear, FITS, Koopa, FreDF, and SDHilb use an input length of $L = 336$, while FEDformer and TimesNet use $L = 96$ to avoid overfitting under longer input windows. PatchTST uses $L = 336$ following its reported setting.

All models were trained using the **Adam** optimizer [31] with MSE as the training objective. The learning rate was selected from $\{10^{-3}, 5 \times 10^{-4}, 10^{-4}\}$, and the batch size was selected from $\{16, 32, 64\}$ according to dataset scale and validation performance. The maximum number of training epochs was set to 50, and early stopping was applied if the validation performance did not improve for 5 consecutive epochs.

For SDHilb, the kernel size in the adaptive multi-scale Hilbert module was treated as a tunable hyperparameter. Depending on the characteristics of each dataset, we also selected whether the kernel parameters were learnable based on validation performance. All experiments were implemented in PyTorch [37] and conducted on a single NVIDIA RTX 3090 GPU with 24GB of memory. For the main forecasting comparisons, each experiment was repeated three times with fixed random seeds. The additional statistical analysis in Appendix E.4 uses five different random seeds.

C.2 Computational Efficiency

We compare six models on Electricity with an input length of 336, a prediction horizon of 720, and a batch size of 16, using the same NVIDIA RTX 3090 (24 GB). After warm-up, training time and forward latency are medians over three epochs and 100 synchronized calls, respectively; full-test time is the median of three complete evaluations. Training and full-test times include data loading and transfers, with CPU metric computation included in full-test time. Forward timing uses GPU-resident inputs. One-time setup is excluded.

Table 5: Computational efficiency on Electricity. Parameters are counted in millions of unique real scalar equivalents, with each complex element counted twice. Memory is peak allocated GPU memory in decimal MB; evaluation excludes gradients and optimizer state.

Model	Params. (M)	Train s/epoch	Full test s	Forward ms/batch	Train peak MB	Eval. peak MB
SDHilb	0.562	17.577	17.856	1.687	160.9	123.1
DLinear	0.485	14.714	15.419	0.681	118.5	84.1
PatchTST	4.276	254.175	37.060	78.321	8439.6	2784.5
FITS	0.162	16.211	13.471	3.543	184.9	168.8
Koopa	2.138	59.406	13.691	5.599	288.0	163.5
FEDformer	24.727	314.730	20.695	36.059	2176.3	590.9

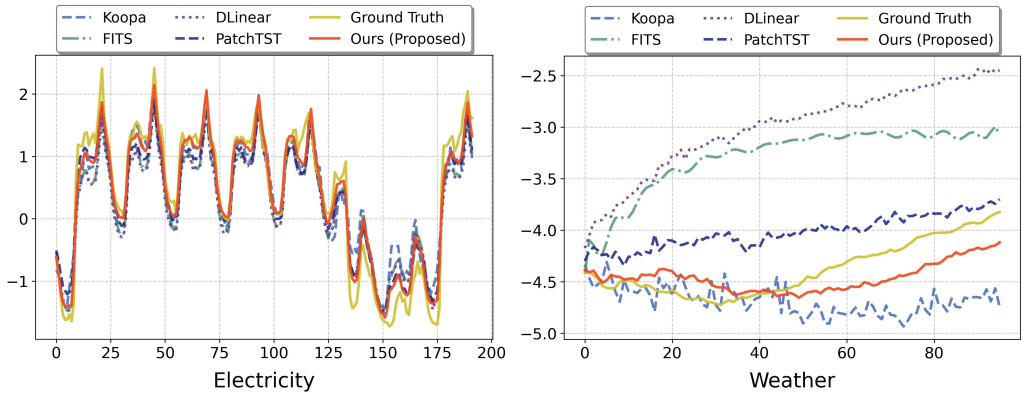
Under this protocol, SDHilb has training time close to DLinear and FITS, while achieving lower forward latency than FITS, Koopa, PatchTST, and FEDformer.

C.3 Visualization

To complement the quantitative results, we provide visual comparisons of long-term forecasting results in Figure 4. We consider 192-step forecasting on the Electricity dataset and 96-step fore-

casting on the Weather dataset, which represent multivariate load patterns and noisy meteorological dynamics, respectively.

As shown in Figure 4, SDHilb produces forecasts that closely follow the ground truth in both datasets. On Electricity, SDHilb captures the overall trend and local variations more accurately than the compared baselines. On Weather, SDHilb maintains a stable trajectory over the forecasting horizon, while some baseline predictions gradually deviate from the ground truth. These visual results are consistent with the quantitative MSE and MAE comparisons reported in the main experimental tables.



(a) Electricity, $H = 192$.

(b) Weather, $H = 96$.

Figure 4: Visualization of long-term forecasting results on the Electricity and Weather datasets. SDHilb follows the ground truth more closely than the compared baselines, especially in trend tracking and long-horizon stability.

C.4 Univariate Forecasting

To further evaluate the generality of SDHilb, we report univariate forecasting results on four ETT datasets in Table 6. The results show that SDHilb achieves competitive or better performance than the compared baselines across most settings, suggesting that the proposed Hilbert-lifted representation and structured recurrence are also effective for univariate long-horizon forecasting.

D Additional Baseline Comparisons with State-Space, Complex-Valued, and Frequency-Domain Methods

This appendix presents additional comparisons with representative baseline models that are not included in the main text due to space constraints. We evaluate SDHilb against three categories of forecasting approaches: state-space models, including S4 [21] and Mamba [22]; complex-valued neural networks, including CVNN (Transformer), a time-series-adapted variant of the Complex Transformer [30]; and frequency-enhanced Transformer models, including FreEformer [36]. These baselines complement the main-text comparisons by covering structured state-space dynamics, complex-valued representations, and frequency-domain Transformer architectures.

All experiments follow the same evaluation protocol as the main text, including eight standard benchmark datasets, multiple forecasting horizons, and the same MSE/MAE evaluation metrics. For each additional baseline, we use the recommended configurations from the original papers or official implementations whenever available.

The additional results further support the effectiveness of SDHilb under long-horizon forecasting settings. On the 720-step forecasting task, SDHilb achieves a favorable balance between prediction accuracy and computational efficiency. On the Weather dataset at a forecasting horizon of 720, SDHilb reduces MSE by 4.8% compared with Mamba [22]. On the noisy and sparse ILI dataset at a forecasting horizon of 60, SDHilb reduces MAE by 1.9% compared with CVNN (Transformer) [30] and by 0.5% compared with FreEformer [36]. These results indicate that the Hilbert-lifted repre-

Table 6: Forecasting results for univariate long-sequence time-series on the full ETT benchmark.

Models	Ours		FITS		DLinear		PatchTST		TimesNet		Koopa		FEDFormer		FreDF	
Metric	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Etth1																
96	**0.053**	**0.176**	0.055	0.179	0.056	0.180	0.055	0.179	0.060	0.188	0.068	0.198	0.079	0.215	0.093	0.233
192	**0.070**	**0.204**	0.072	0.207	0.071	**0.204**	0.071	0.205	0.073	0.206	0.092	0.234	0.104	0.245	0.093	0.228
336	**0.081**	0.226	0.087	0.234	0.098	0.244	**0.081**	0.225	**0.081**	**0.220**	0.109	0.263	0.119	0.270	0.108	0.254
720	**0.086**	**0.232**	0.104	0.233	0.189	0.359	0.087	**0.232**	0.089	0.237	0.105	0.256	0.142	0.299	0.195	0.368
Avg	**0.073**	**0.210**	0.080	0.213	0.104	0.247	0.074	**0.210**	0.076	0.213	0.094	0.238	0.111	0.257	0.122	0.271
Etth2																
96	0.129	0.276	0.132	0.282	0.131	0.279	0.129	0.282	0.137	0.287	0.131	0.276	**0.128**	**0.271**	0.139	0.288
192	0.169	**0.323**	0.173	0.329	0.176	0.329	**0.168**	0.328	0.187	0.338	0.171	0.329	0.185	0.330	0.185	0.333
336	0.195	0.357	0.202	0.364	0.209	0.367	**0.185**	**0.351**	0.221	0.372	0.221	0.373	0.231	0.378	0.230	0.378
720	0.230	0.386	0.247	0.401	0.276	0.426	**0.224**	0.383	0.246	**0.369**	0.234	0.374	0.278	0.420	0.319	0.461
Avg	0.181	**0.336**	0.189	0.344	0.198	0.350	**0.177**	**0.336**	0.198	0.342	0.189	0.338	0.206	0.350	0.218	0.365
Ettm1																
96	**0.026**	**0.121**	0.027	0.123	0.028	0.123	**0.026**	**0.121**	0.029	0.128	0.031	0.132	0.033	0.140	0.051	0.173
192	**0.039**	**0.149**	0.040	0.150	0.045	0.156	**0.039**	0.150	0.048	0.165	0.047	0.166	0.058	0.186	0.092	0.238
336	**0.052**	**0.172**	0.053	**0.172**	0.061	0.182	0.053	0.173	0.067	0.198	0.061	0.189	0.084	0.231	0.152	0.318
720	**0.073**	**0.206**	**0.073**	**0.206**	0.080	0.210	0.074	0.207	0.084	0.228	0.079	0.214	0.102	0.250	0.204	0.374
Avg	**0.048**	**0.162**	**0.048**	0.163	0.054	0.168	**0.048**	0.163	0.057	0.180	0.055	0.175	0.069	0.202	0.125	0.276
Ettm2																
96	**0.063**	**0.183**	0.064	0.185	**0.063**	**0.183**	0.065	0.186	0.066	0.186	0.073	0.198	0.067	0.198	0.071	0.195
192	**0.090**	**0.225**	0.091	0.226	0.092	0.227	0.094	0.231	0.112	0.249	0.115	0.252	0.102	0.245	0.107	0.246
336	**0.117**	**0.259**	0.118	0.260	0.119	0.261	0.120	0.265	0.133	0.277	0.149	0.291	0.130	0.279	0.145	0.295
720	**0.171**	**0.319**	0.172	0.322	0.175	0.320	**0.171**	0.322	0.182	0.332	0.208	0.354	0.178	0.325	0.201	0.351
Avg	**0.110**	**0.247**	0.111	0.248	0.112	0.248	0.113	0.251	0.123	0.261	0.136	0.274	0.119	0.262	0.131	0.272

sentation and Schrödinger-Hamiltonian motivated dynamics remain effective when compared with state-space, complex-valued, and frequency-domain baselines.

Beyond prediction accuracy, SDHilb also maintains strong computational efficiency. SDHilb uses only 303.5K parameters, corresponding to a 90% reduction compared with Mamba [22] (3.03M), a 99.7% reduction compared with CVNN (Transformer) [30] (92.3M), and a 98% reduction compared with FreEformer [36] (15.2M). On the Electricity dataset with look-back window 336 and forecasting horizon 336, SDHilb also requires fewer MACs than these additional neural baselines, as shown in Table 8.

Overall, the additional comparisons show that SDHilb remains competitive against representative state-space, complex-valued, and frequency-domain forecasting models. Together with the main experimental results, these findings further support the effectiveness of combining Hilbert-lifted amplitude-phase modeling, Schrödinger-Hamiltonian motivated dynamics, and parameter-efficient recurrence for long-horizon time series forecasting.

E Ablation Study

E.1 Varying Looking-back Window

A closer examination of Figure 5 reveals that different models exhibit varying levels of sensitivity to changes in the input sequence length L, reflecting both their architectural capacity for modeling long-term dependencies and their ability to learn from longer input horizons. Transformer-based models such as FEDFormer show a clear trend of overfitting as L increases. For example, on the *ETTh1* dataset (Figure 5(a)), its MSE rises significantly from $L = 336$ to $L = 720$, indicating that its frequency-enhanced attention mechanism suffers from limited generalization when processing long sequences and tends to overfit noise as the input length grows. TimesNet, which adopts an Inception-style temporal-frequency block design, exhibits more stable behavior. As shown in the figure, its performance plateaus beyond $L = 192$, suggesting that the benefit of further increasing the receptive field is limited by its local spectral representation.

In contrast, non-attention-based models such as DLinear and Koopa demonstrate stronger adaptability to longer inputs. DLinear applies independent linear projections to each channel, offering high

Table 7: Performance comparison of different models on multiple datasets. Best results are **bold**, second-best are underlined.

Horizon	Ours		S4		FreEformer		CVNN (Transformer)		Mamba	
	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1										
96	**0.370**	0.393	0.385	0.401	0.371	**0.392**	0.402	0.415	0.375	0.395
192	**0.405**	**0.413**	0.421	0.428	0.424	0.421	0.410	0.420	0.418	0.425
336	**0.429**	**0.431**	0.445	0.442	0.469	0.445	0.458	0.453	0.438	0.435
720	**0.428**	**0.452**	0.449	0.468	0.471	0.469	0.463	0.475	0.436	0.460
Avg	**0.408**	**0.422**	0.425	0.435	0.434	0.432	0.433	0.441	0.417	0.429
ETTh2										
96	**0.272**	0.335	0.278	0.338	0.284	**0.331**	0.295	0.346	0.280	0.340
192	**0.327**	**0.374**	0.335	0.382	0.362	0.381	0.349	0.391	0.338	0.378
336	**0.335**	**0.387**	0.342	0.390	0.413	0.419	0.357	0.402	0.345	0.395
720	**0.380**	**0.420**	0.391	0.432	0.423	0.437	0.405	0.445	0.398	0.428
Avg	**0.329**	**0.379**	0.337	0.386	0.370	0.392	0.352	0.396	0.340	0.385
ETTm1										
96	**0.290**	**0.338**	0.302	0.349	0.313	0.349	0.308	0.345	0.295	0.342
192	**0.331**	**0.361**	0.345	0.372	0.362	0.375	0.358	0.384	0.338	0.365
336	**0.371**	**0.384**	0.389	0.398	0.393	0.396	0.395	0.405	0.379	0.390
720	**0.428**	**0.417**	0.446	0.432	0.455	0.431	0.452	0.441	0.435	0.425
Avg	**0.355**	**0.375**	0.371	0.388	0.381	0.388	0.378	0.394	0.362	0.380
ETTm2										
96	**0.161**	**0.250**	0.168	0.258	0.169	0.255	0.165	0.253	0.172	0.261
192	**0.217**	**0.289**	0.225	0.298	0.233	0.291	0.221	0.302	0.229	0.305
336	**0.271**	**0.325**	0.283	0.336	0.292	0.329	0.278	0.332	0.287	0.340
720	**0.366**	**0.382**	0.379	0.395	0.393	0.389	0.372	0.390	0.385	0.401
Avg	**0.254**	**0.311**	0.264	0.322	0.272	0.316	0.259	0.319	0.268	0.327
Weather										
96	**0.312**	**0.281**	0.328	0.292	0.319	0.285	0.335	0.298	0.331	0.295
192	**0.388**	**0.333**	0.405	0.346	0.395	0.338	0.418	0.354	0.409	0.349
336	**0.457**	**0.372**	0.476	0.385	0.465	0.378	0.489	0.392	0.481	0.388
720	**0.559**	**0.417**	0.582	0.432	0.568	0.423	0.598	0.441	0.587	0.436
Avg	**0.429**	**0.351**	0.448	0.364	0.437	0.356	0.460	0.371	0.452	0.367
Traffic										
96	**0.410**	**0.281**	0.428	0.290	0.419	0.285	0.435	0.295	0.432	0.292
192	**0.422**	**0.284**	0.440	0.293	0.431	0.288	0.447	0.298	0.443	0.296
336	**0.435**	**0.290**	0.453	0.299	0.444	0.293	0.461	0.304	0.457	0.301
720	**0.464**	**0.308**	0.482	0.318	0.473	0.312	0.490	0.323	0.486	0.320
Avg	**0.433**	**0.291**	0.451	0.300	0.442	0.295	0.458	0.305	0.454	0.302
Electricity										
96	**0.140**	**0.225**	0.148	0.232	0.145	0.228	0.153	0.236	0.151	0.234
192	**0.155**	**0.238**	0.163	0.245	0.159	0.241	0.168	0.250	0.165	0.247
336	**0.178**	**0.255**	0.186	0.262	0.182	0.258	0.191	0.267	0.189	0.264
720	**0.203**	**0.278**	0.211	0.285	0.207	0.281	0.216	0.290	0.213	0.287
Avg	**0.169**	**0.249**	0.177	0.256	0.173	0.252	0.182	0.261	0.179	0.258
ILI										
24	**1.462**	**0.866**	1.485	0.875	1.473	0.872	1.502	0.883	1.491	0.878
36	**1.453**	**0.884**	1.471	0.892	1.465	0.888	1.489	0.901	1.478	0.895
48	**1.516**	**0.867**	1.538	0.876	1.527	0.873	1.554	0.885	1.545	0.880
60	**1.722**	**0.932**	1.745	0.941	1.736	0.937	1.762	0.950	1.753	0.945
Avg	**1.538**	**0.887**	1.560	0.896	1.550	0.893	1.577	0.905	1.567	0.900

computational efficiency and robustness, while Koopa integrates dynamic decomposition strategies and maintains stable performance across different input lengths. Another frequency-domain model, FreDF, also shows performance improvements at certain input lengths, though its trend is somewhat less consistent. It is worth noting that although FITS and PatchTST are not included in our figure, prior studies have systematically evaluated their performance under varying look-back lengths. FITS learns amplitude and phase trajectories in the complex domain, exhibiting steady improvements on longer sequences. PatchTST, leveraging patch tokenization, mitigates the overfitting issues commonly observed in Transformer-based models on long inputs.

For the main comparisons, we retain the model-specific input-length settings described in Section 4.

Table 8: Number of trainable parameters and computational cost measured by MACs under look-back window 336 and forecasting horizon 336 on the Electricity dataset. All models are evaluated under the same input-output setting.

Model	Parameters	MACs
SDHilb (Ours)	303.50K	170.41M
S4	2.77M	940.3M
Mamba	3.03M	939.4M
CVNN (Transformer)	92.3M	3.52G
FreEformer	15.2M	450.7M

Table 9 shows that **SDHilb** benefits from longer look-back windows in many settings, although the best input length depends on the dataset and forecasting horizon.

Table 9: Multivariate forecasting results under varying look-back lengths and prediction horizons. Parenthesized look-back lengths apply to ILI; unparenthesized lengths apply to the other datasets.

Dataset	Horizon	Look-back Window									
		48(36)		96(46)		192(60)		336(104)		720(144)	
		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	96	0.395	0.397	0.386	0.393	0.385	0.398	0.370	0.393	0.388	0.410
	192	0.451	0.430	0.439	0.423	0.426	0.419	0.405	0.413	0.420	0.430
	336	0.504	0.456	0.482	0.444	0.458	0.437	0.429	0.431	0.422	0.441
	720	0.510	0.481	0.482	0.464	0.459	0.458	0.428	0.452	0.431	0.462
ETTh2	96	0.301	0.431	0.289	0.336	0.284	0.337	0.272	0.335	0.279	0.341
	192	0.392	0.394	0.375	0.388	0.358	0.385	0.327	0.374	0.331	0.374
	336	0.442	0.434	0.414	0.425	0.385	0.414	0.335	0.387	0.339	0.388
	720	0.438	0.444	0.419	0.438	0.407	0.433	0.380	0.420	0.388	0.430
ETTm1	96	0.505	0.439	0.342	0.369	0.303	0.345	0.290	0.338	0.321	0.363
	192	0.559	0.465	0.385	0.390	0.344	0.369	0.331	0.361	0.355	0.382
	336	0.610	0.494	0.419	0.411	0.382	0.389	0.371	0.384	0.370	0.386
	720	0.666	0.525	0.486	0.449	0.440	0.423	0.428	0.417	0.416	0.412
ETTm2	96	0.189	0.269	0.177	0.257	0.169	0.252	0.161	0.250	0.162	0.252
	192	0.261	0.315	0.244	0.301	0.232	0.396	0.217	0.289	0.220	0.292
	336	0.328	0.356	0.306	0.341	0.284	0.330	0.271	0.325	0.272	0.329
	720	0.429	0.412	0.406	0.397	0.380	0.386	0.366	0.382	0.348	0.377
Traffic	96	0.878	0.478	0.642	0.385	0.450	0.298	0.410	0.281	0.387	0.271
	192	0.758	0.425	0.595	0.362	0.457	0.301	0.422	0.284	0.397	0.274
	336	0.785	0.436	0.602	0.364	0.471	0.305	0.435	0.290	0.410	0.278
	720	0.819	0.450	0.641	0.385	0.499	0.322	0.464	0.308	0.451	0.303
Weather	96	0.366	0.308	0.337	0.293	0.320	0.284	0.312	0.281	0.295	0.278
	192	0.453	0.358	0.426	0.348	0.408	0.340	0.388	0.333	0.374	0.333
	336	0.564	0.412	0.527	0.397	0.494	0.384	0.457	0.372	0.434	0.366
	720	0.477	0.459	0.635	0.442	0.596	0.428	0.559	0.417	0.523	0.411
Electricity	96	0.261	0.305	0.199	0.268	0.145	0.230	0.140	0.225	0.136	0.223
	192	0.248	0.297	0.199	0.269	0.164	0.245	0.155	0.238	0.152	0.237
	336	0.273	0.316	0.222	0.286	0.188	0.264	0.178	0.255	0.174	0.254
	720	0.302	0.337	0.252	0.309	0.219	0.290	0.203	0.278	0.199	0.274
ILI	24	2.641	1.075	2.151	0.946	2.166	0.975	1.462	0.866	2.110	1.040
	36	2.540	1.040	2.077	0.915	2.198	0.937	1.453	0.884	1.829	0.941
	48	2.352	1.002	2.094	0.933	2.035	0.928	1.516	0.867	1.989	0.995
	60	2.668	1.089	2.183	0.974	2.150	0.979	1.722	0.932	2.081	0.987

Table 10: Ablation study on ETTh1 across multiple forecasting horizons and model configurations.

Architecture	Configuration	Params (M)	MACs	MSE				MAE			
				96	192	336	720	96	192	336	720
Linear	1xWidth	0.23	2.10M	**0.370**	**0.405**	**0.429**	**0.428**	**0.393**	**0.413**	**0.431**	**0.452**
MLP	1L-H64	0.09	1.73M	0.381	0.420	0.452	0.475	0.405	0.427	0.447	0.477
	2L-H128	0.21	1.95M	0.383	0.424	0.439	0.459	0.405	0.431	0.441	0.471
	3L-H256	0.61	4.76M	0.379	0.420	0.434	0.476	0.403	0.427	0.439	0.481
CNN	3L-C16-K5	0.24	4.44M	0.399	0.448	0.474	0.524	0.421	0.443	0.471	0.503
	5L-C32-K7	0.29	23.26M	0.424	0.474	0.487	0.723	0.442	0.473	0.479	0.596
	7L-C64-K9	0.69	156.50M	0.455	0.472	0.615	0.715	0.462	0.473	0.551	0.606
LSTM	2L-H128	0.83	6.34M	0.396	0.441	0.462	0.498	0.419	0.446	0.461	0.497
	3L-H256	3.50	25.06M	0.411	0.438	0.462	0.495	0.430	0.445	0.460	0.494
	4L-H512	12.65	115.78M	0.416	0.452	0.481	0.551	0.435	0.458	0.473	0.520
Transformer	4H-2L-D64	0.32	47.43M	0.430	0.469	0.603	0.591	0.445	0.465	0.539	0.541
	8H-4L-D128	1.70	358.38M	0.451	0.594	0.544	0.783	0.466	0.525	0.515	0.600
	16H-6L-D256	9.59	1.023G	0.513	0.575	0.588	0.882	0.487	0.524	0.530	0.614

E.2 Architectural Ablation Study

We conducted a comprehensive ablation study on ETTh1, as shown in Table 10, to investigate whether a linear structure is necessary after applying the multi-scale Hilbert transform. To ensure fair and consistent comparison, we adopted representative configurations for each architecture type. In Table 10, L indicates the number of layers, H the hidden size or attention heads, C the number of channels, and K the kernel size. These configurations are chosen based on widely adopted settings in prior work for MLP [38], CNN [2], LSTM [39, 40], and Transformer-based models [3, 4], covering various model complexities while ensuring fair comparisons.

Across all forecasting horizons (from 96 to 720 steps), the linear structure consistently achieved the lowest MSE and MAE scores, outperforming all more complex architectures, including MLP, CNN, LSTM, and Transformer. This result validates our hypothesis that the linear transformation, as derived from the Schrödinger equation, is better suited for preserving the phase-frequency relationships encoded by the Hilbert transform. This finding is not only supported by empirical evidence but also aligns with the structural properties of the time-dependent Schrödinger equation.

We further observed that the performance gap widened significantly with increasing prediction length. For example, in the 720-step forecasting task, the errors of Transformer and CNN models were significantly higherup to twice as large as those of the linear model. This suggests that complex architectures struggle to maintain the consistency of frequency-phase structures in long-term sequence modeling, whereas linear models possess a natural advantage in this regard.

In addition to superior performance, the linear model also demonstrated significantly lower parameter count (only 0.23M) and computational cost (2.10M MACs) than most tested alternatives, highlighting its computational efficiency in practical applications.

This supports the notion that simple structures are not only effective, but also more consistent with the physical modeling principles of the problem.

The Hilbert transform converts real-valued time series into complex-valued signals, making their phase and frequency relationships explicit. This complex representation closely resembles the wavefunction formalism in quantum mechanics. In the Schrödinger equation, the system evolves linearly in Hilbert space. These results are consistent with the motivation for preserving coupled amplitudephase modeling through a linear prediction structure. Therefore, a linear structure provides a minimally invasive mapping that respects the physical nature of the transformed signal.

Meanwhile, recent years have witnessed the success of linear models in time series forecasting tasks, and the dynamics of the Schrödinger equation provide a theoretical explanation for this empirical trend from a physics-informed perspective.

E.3 Core Component Ablation Analysis

Table 11: Core component ablation results of SDHilb across eight benchmark datasets and multiple forecasting horizons.

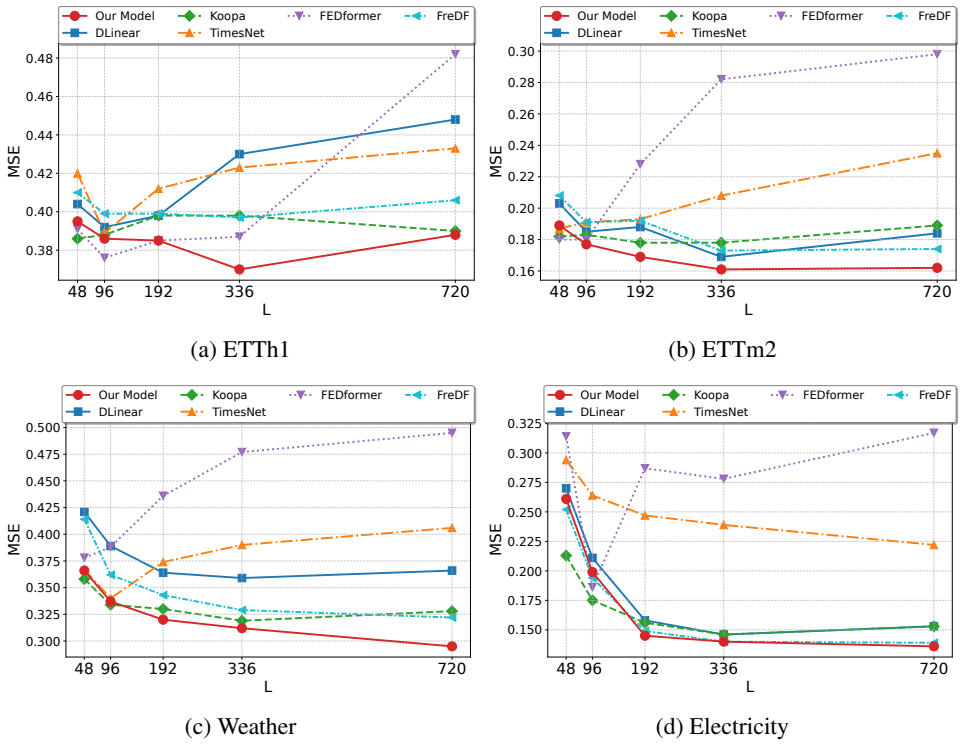
Dataset	Horizon	Cases											
		Ours(Full)		Case1		Case2		Case3		Case4		Case5	
		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	96	**0.370**	**0.393**	0.373	0.396	0.372	0.395	0.373	0.397	0.380	0.403	0.373	0.398
	192	**0.405**	**0.413**	0.408	0.415	0.407	0.415	0.410	0.421	0.418	0.427	0.412	0.422
	336	**0.429**	**0.431**	0.436	0.440	0.435	0.435	0.442	0.447	0.452	0.451	0.447	0.448
	720	**0.428**	**0.452**	0.441	0.459	0.436	0.455	0.484	0.499	0.475	0.493	0.441	0.457
ETTh2	96	**0.272**	**0.335**	0.278	0.341	0.276	0.340	0.301	0.362	0.314	0.370	0.284	0.343
	192	**0.327**	**0.374**	0.338	0.378	0.339	0.377	0.375	0.412	0.390	0.423	0.345	0.381
	336	**0.335**	**0.387**	0.351	0.395	0.348	0.393	0.430	0.453	0.420	0.448	0.352	0.395
	720	**0.380**	**0.420**	0.386	0.424	0.392	0.428	0.692	0.590	0.742	0.613	0.392	0.429
ETTm1	96	**0.290**	**0.338**	0.305	0.347	0.304	0.345	0.304	0.348	0.303	0.351	0.303	0.345
	192	**0.331**	**0.361**	0.339	0.367	0.338	0.365	0.337	0.369	0.343	0.377	0.340	0.368
	336	**0.371**	**0.384**	0.381	0.391	0.377	0.389	0.378	0.396	0.375	0.390	0.379	0.392
	720	**0.428**	**0.417**	0.436	0.424	0.440	0.426	0.435	0.431	0.436	0.435	0.440	0.427
ETTm2	96	**0.161**	**0.250**	0.167	0.255	0.165	0.253	0.169	0.262	0.168	0.263	0.165	0.254
	192	**0.217**	**0.289**	0.225	0.296	0.225	0.295	0.231	0.311	0.237	0.317	0.228	0.299
	336	**0.271**	**0.325**	0.277	0.330	0.278	0.330	0.280	0.339	0.298	0.357	0.280	0.329
	720	**0.366**	**0.382**	0.373	0.388	0.374	0.388	0.431	0.447	0.403	0.425	0.386	0.403
Weather	96	**0.312**	**0.281**	0.358	0.312	0.359	0.313	0.358	0.333	0.358	0.328	0.364	0.316
	192	**0.388**	**0.333**	0.431	0.353	0.434	0.354	0.424	0.376	0.423	0.369	0.398	0.344
	336	**0.457**	**0.372**	0.503	0.389	0.502	0.388	0.511	0.437	0.497	0.419	0.483	0.399
	720	**0.559**	**0.417**	0.598	0.430	0.597	0.430	0.579	0.468	0.591	0.479	0.578	0.463
Traffic	96	**0.410**	**0.281**	0.418	0.291	0.416	0.286	0.420	0.292	0.416	0.289	0.417	0.289
	192	**0.422**	**0.284**	0.434	0.299	0.432	0.294	0.432	0.299	0.431	0.297	0.433	0.296
	336	**0.435**	**0.290**	0.448	0.307	0.445	0.297	0.448	0.310	0.446	0.306	0.447	0.305
	720	**0.464**	**0.308**	0.474	0.318	0.475	0.320	0.477	0.323	0.476	0.325	0.486	0.331
Electricity	96	**0.140**	**0.225**	0.146	0.234	0.146	0.233	0.145	0.236	0.146	0.236	0.148	0.236
	192	**0.155**	**0.238**	0.164	0.245	0.163	0.245	0.167	0.250	0.166	0.250	0.164	0.346
	336	**0.178**	**0.255**	0.185	0.263	0.184	0.262	0.182	0.268	0.184	0.271	0.189	0.270
	720	**0.203**	**0.278**	0.211	0.284	0.211	0.284	0.211	0.299	0.209	0.297	0.212	0.286
ILI	24	**1.462**	**0.866**	1.998	1.046	2.251	1.006	2.611	1.166	2.804	1.237	2.037	0.986
	36	**1.453**	**0.884**	1.761	0.959	1.944	0.947	2.608	1.187	2.856	1.146	2.003	0.950
	48	**1.516**	**0.867**	1.798	0.931	1.957	0.952	2.653	1.146	2.528	1.116	1.929	0.941
	60	**1.722**	**0.932**	1.932	1.018	2.064	0.998	2.830	1.192	2.901	1.257	2.089	0.974

We conduct component ablation studies across eight forecasting datasets: ETTh1, ETTh2, ETTm1, ETTm2, Weather, Traffic, Electricity, and ILI. Table 11 reports MSE and MAE under different forecasting horizons. The comparisons examine the Hilbert lifting strategy, the use of a lifted complex representation, Hann windowing, imaginary-component construction, and the forecasting module.

Case1 replaces adaptive multi-scale Hilbert lifting with the standard Hilbert transform, while Case2 provides a real-valued forecasting baseline without Hilbert lifting. Case3 removes Hann windowing while retaining the adaptive multi-scale Hilbert module. Case4 replaces Hilbert-based imaginary-component construction with an FC mapping, while retaining the predictor applied to the concatenated real and quadrature components. Case5 retains adaptive multi-scale Hilbert lifting and Hann windowing while removing this prediction module (component B in Table 4).

In the ETTh1 comparison at $H = 336$ in Table 4, the full model achieves the lowest MSE and MAE among the reported variants. Replacing Hilbert-based imaginary-component construction with an FC mapping produces the largest increase in both errors in this setting, supporting the contribution of Hilbert-based lifting. Using the standard Hilbert transform, relying only on real-valued inputs, or removing Hann windowing also increases forecasting errors. These comparisons support the com-

bined design of adaptive Hilbert lifting, windowed quadrature construction, and joint real–imaginary prediction. The full model also achieves the lowest MSE and MAE among the variants reported in Table 11.



(a) ETTh1 (b) ETTm2

(c) Weather (d) Electricity

Figure 5: Model performance under varying look-back window lengths across representative datasets.

E.4 Statistical Significance Analysis

To evaluate the statistical significance of the experimental results of our proposed method, **SDHilb**, we conducted comparative experiments on three representative non-stationary time series forecasting datasets: **Weather**, **ETTh2**, and **Traffic**. These datasets are widely used benchmarks in long-term forecasting tasks, characterized by varying degrees of seasonality, trend, and volatility, making them suitable for comprehensively validating model performance.

Specifically, for each forecast length (96, 192, 336, 720), we trained SDHilb with **five different random seeds** and reported the mean and standard deviation of **MSE** and **MAE** on the test sets, as summarized in Table 12. All experiments were conducted with the same data splits to ensure comparability across runs.

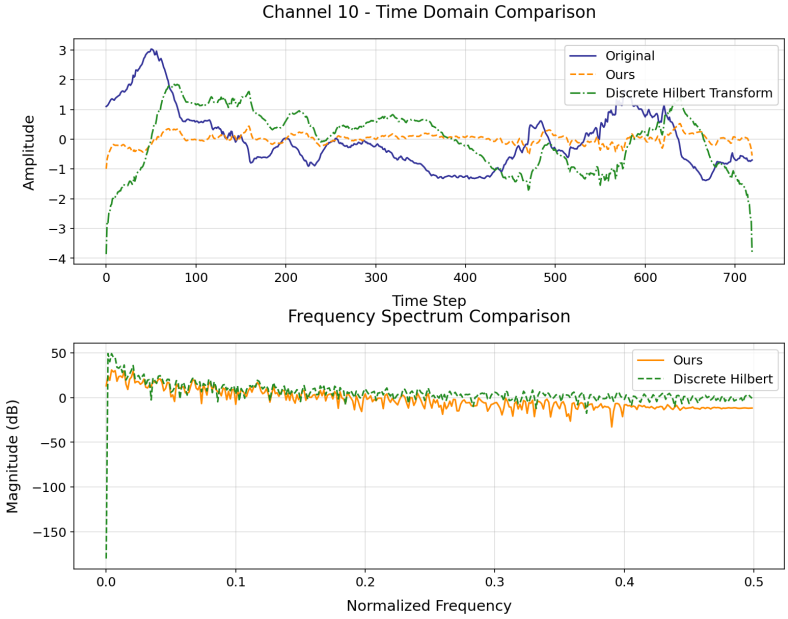
Although repeated experiments are often omitted in large-scale foundation models due to resource constraints, we deliberately performed multiple training runs to quantify the effect of initialization randomness on model performance. As shown in the results, the standard deviations across all datasets and forecast lengths remain small, indicating that SDHilb demonstrates strong **stability and consistency** under different training conditions.

Table 12: Mean and standard deviation of MSE and MAE for **SDHilb** over 5 random seeds across four forecast lengths.

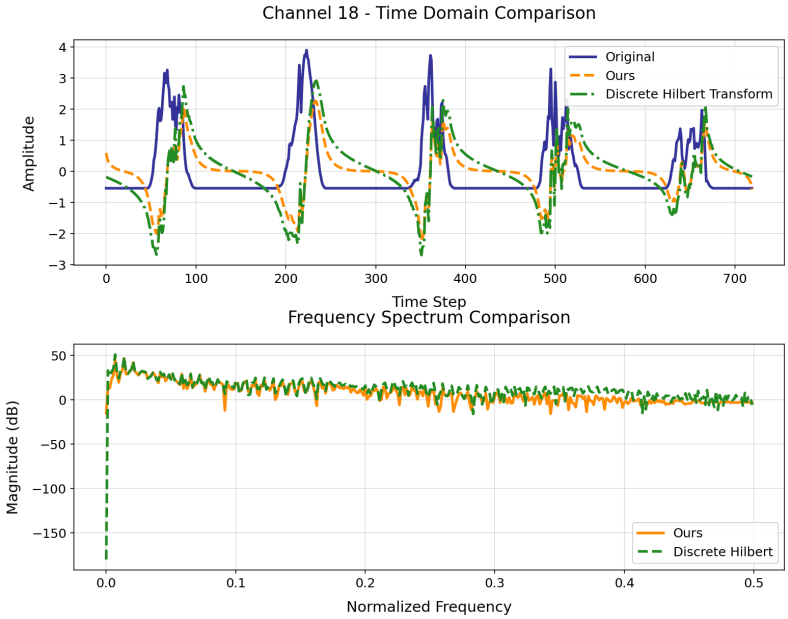
Dataset	**96**		**192**		**336**		**720**	
	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh2	0.273 ± 0.001	0.334 ± 0.003	0.328 ± 0.002	0.373 ± 0.004	0.335 ± 0.002	0.385 ± 0.005	0.380 ± 0.003	0.419 ± 0.003
Weather	0.312 ± 0.003	0.282 ± 0.001	0.387 ± 0.003	0.333 ± 0.002	0.456 ± 0.004	0.373 ± 0.002	0.560 ± 0.005	0.415 ± 0.004
Traffic	0.408 ± 0.006	0.280 ± 0.002	0.423 ± 0.004	0.285 ± 0.003	0.436 ± 0.003	0.289 ± 0.001	0.466 ± 0.006	0.309 ± 0.003

F Adaptive Multi-Scale Hilbert Module

F.1 Comparison with the Discrete Hilbert Transform



(a) Channel 10 comparison between the adaptive multi-scale Hilbert module and the standard discrete Hilbert transform. The adaptive module shows a sharper local response around the abrupt temperature drop.



(b) Channel 18 comparison between the adaptive multi-scale Hilbert module and the standard discrete Hilbert transform. The adaptive module exhibits smoother behavior in the steady-state region.

Figure 6: Comparison between the adaptive multi-scale Hilbert module and the standard discrete Hilbert transform on representative channels.

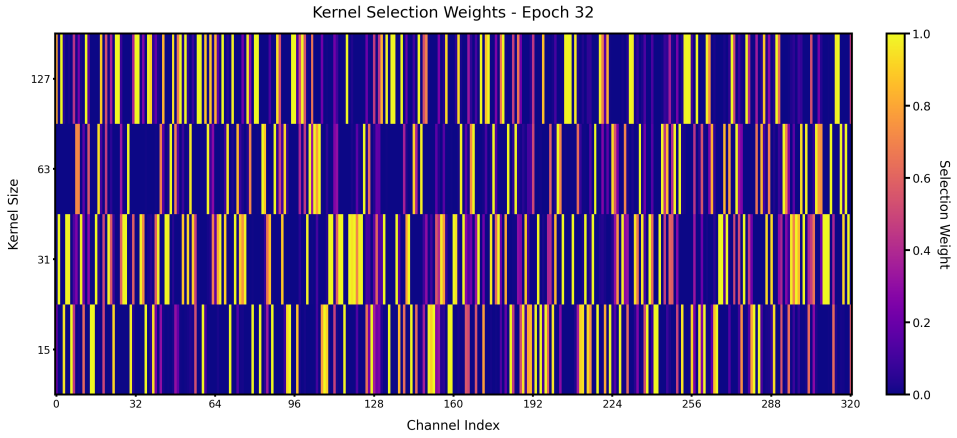
In the time-domain comparison, the proposed adaptive multi-scale Hilbert module produces a quadrature component that is broadly consistent with the standard discrete Hilbert transform, while

showing improved behavior in finite-length forecasting windows. As illustrated in Figure 6a and Figure 6b, the adaptive module better follows local variations around abrupt changes and exhibits smoother behavior near boundary regions. This is useful for time series forecasting, where the input window is finite and boundary artifacts can affect the quality of the lifted complex representation.

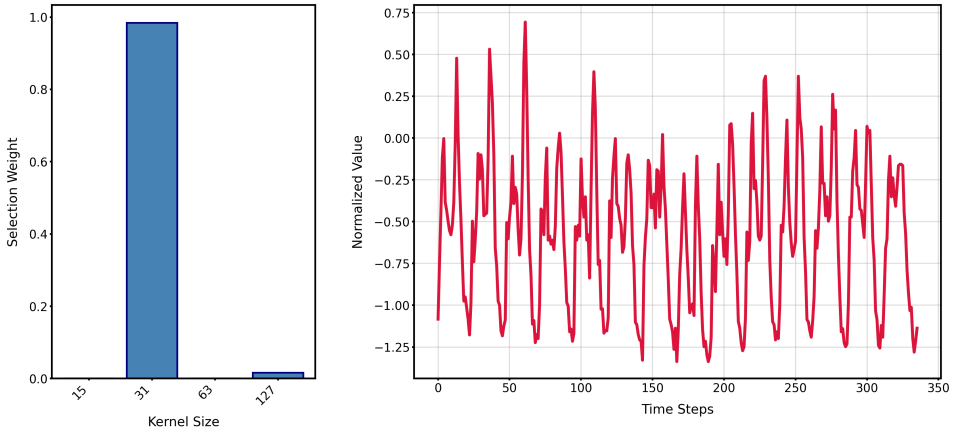
In the frequency-domain comparison, the Hann-windowed multi-scale design mainly aims to reduce spectral leakage caused by finite-window observations. Compared with a rectangular window, the Hann window suppresses side lobes more effectively, which reduces leakage from dominant frequency components to neighboring bands. Rather than claiming universally higher frequency resolution, we use the multi-scale design to balance locality and smoothness across different temporal patterns. This provides a cleaner quadrature representation for downstream forecasting and reduces the sensitivity of the lifted complex state to high-frequency artifacts.

Taken together, the time-domain visualization, frequency-domain interpretation, learned kernel-weight distributions, and component ablations suggest that the adaptive multi-scale Hilbert module improves the quality of the lifted complex representation. It helps capture local variations, reduces boundary and leakage artifacts, and provides a more stable quadrature component for the subsequent structured dynamics.

F.2 Multi-scale Kernel Selection Analysis



(a) Channel-wise kernel selection weights.



(b) Channel 5 input sequence and learned kernel selection weights.

Figure 7: Analysis of multi-scale Hilbert kernel selection. (a) Channel-wise kernel selection weights on the Electricity dataset. (b) Selected kernel weights for Channel 5 and its corresponding input series.

We further analyze the learned multi-scale kernel weights to examine whether the adaptive Hilbert module captures heterogeneous temporal patterns across channels. As shown in Figure 7(a), the Electricity dataset exhibits diverse channel-wise kernel preferences. Channels with rapidly varying local patterns tend to assign larger weights to shorter kernels, while smoother or longer-period channels tend to use larger kernels. This indicates that the model does not rely on a fixed Hilbert kernel for all variables, but instead adapts the effective receptive field according to channel-specific temporal characteristics.

Figure 7(b) provides a representative example for Channel 5. The learned weights assign the largest mass to the medium-scale kernel with size $k = 31$. The corresponding input sequence contains local oscillatory patterns at a similar temporal scale, suggesting that the selected kernel provides a suitable receptive field for this channel. Compared with very short kernels, the medium-scale kernel is less sensitive to local noise; compared with very large kernels, it avoids excessive smoothing of short-term variations.

To avoid relying only on a single-channel example, we also examine aggregate kernel-size preferences across datasets. As summarized in Table 13, the learned distributions are clearly dataset-dependent. Weather assigns more weight to the largest kernel $k = 127$, which is consistent with slow-varying meteorological cycles. ILI tends to favor the medium-to-large kernel $k = 63$, reflecting longer seasonal patterns. In contrast, Electricity and ETTh1 show more distributed preferences across multiple kernel sizes, consistent with their heterogeneous multi-scale temporal structures. These aggregate statistics indicate that the Gumbel-Softmax fusion does not collapse to a fixed kernel, but instead learns dataset- and channel-dependent scale selection.

Table 13: Aggregate kernel-size preference patterns observed from the adaptive multi-scale Hilbert module. The dataset-dependent distributions suggest that the scale-selection mechanism adapts to different temporal characteristics instead of collapsing to a fixed kernel size.

Dataset	Dominant / Typical Kernel Preference	Interpretation
Weather	$k = 127$	Slow-varying meteorological cycles
ILI	$k = 63$	Longer seasonal or epidemic patterns
Electricity	Distributed across scales	Multi-scale heterogeneous load patterns
ETTh1	Distributed across scales	Mixed temporal dynamics across variables

In summary, the learned kernel weights provide empirical evidence that the adaptive multi-scale Hilbert module adjusts its receptive field across heterogeneous channels and datasets. The aggregate kernel-size statistics show dataset-dependent scale preferences: Weather concentrates more on $k = 127$, ILI favors $k = 63$, while Electricity and ETTh1 distribute weights more uniformly across multiple scales. Together with the ablation results, where replacing the adaptive multi-scale Hilbert module with a standard Hilbert transform degrades ETTh1 $H = 720$ MSE from 0.428 to 0.441 and leads to a larger gap on ILI, these results support the use of adaptive multi-scale Hilbert lifting instead of a fixed-kernel design.

F.3 Connection to Lifted-State Stability

The structured dynamics in SDHilb is connected to the stability of the Hilbert-lifted complex state. We emphasize that SDHilb does not assume the original real-valued time series to be physically conservative. Instead, the stability-oriented inductive bias is imposed in the lifted complex analytic-signal space constructed by the Hilbert representation. Specifically, we monitor the lifted-state energy

$$E(t) = \|\mathbf{x}_r(t)\|^2 + \|\mathbf{x}_i(t)\|^2,$$

where $\mathbf{x}_r(t)$ and $\mathbf{x}_i(t)$ denote the real and quadrature components of the lifted complex state. This quantity is used as an approximate stability diagnostic in the lifted complex space, rather than as exact physical energy conservation of the raw time series.

As summarized in Table 14, SDHilb achieves the lowest lifted-state energy drift among the compared variants. On ETTh1, SDHilb obtains 9.66% drift at $H = 96$ and 12.10% drift at $H = 720$, while the Standard Euler ODE and FC replacement variants exhibit larger drift, especially at the long horizon. This trend is consistent with the forecasting results: variants with larger lifted-state energy drift also produce higher MSE. These results suggest that the structured recurrence provides

useful regularization for long-horizon evolution in the Hilbert-lifted complex space, even though the implemented model only approximates the ideal energy-conserving coefficient structure.

Table 14: Lifted-state energy drift and forecasting accuracy on ETTh1. Energy drift is measured as the mean relative change of the lifted-state energy $E(t) = \|\mathbf{x}_r(t)\|^2 + \|\mathbf{x}_i(t)\|^2$ between the input and output states across test samples. Lower energy drift and lower MSE are better.

Model	$H = 96$ E-drift	$H = 720$ E-drift	$H = 96$ MSE	$H = 720$ MSE
SDHilb	9.66%	12.10%	0.370	0.428
Standard Euler ODE	10.26%	14.28%	0.372	0.443
FC replacement	11.24%	15.93%	0.373	0.452

These results clarify the role of the structured dynamics. The Hilbert module constructs the lifted amplitude-phase state, and the Schrödinger-Hamiltonian motivated dynamics regularizes its long-horizon evolution in the complex state space. Therefore, the conservation-related argument should be understood as a stability-oriented inductive bias in the lifted representation space, not as a literal physical conservation law for arbitrary real-world time series.

G Phase Behavior under Normalization in the Hilbert-Lifted Space

We analyze the behavior of the Hilbert-lifted phase under input normalization. This analysis shows that the amplitude-phase representation used by SDHilb remains stable after normalization, which supports subsequent complex-state evolution in the lifted space.

Lemma 1 *The Hilbert transform $\mathcal{H}$ is linear. For any $f, g \in L^p(\mathbb{R})$ with $1 < p < \infty$ and scalars $a, b \in \mathbb{C}$,*

$$\mathcal{H}[af + bg](t) = a\mathcal{H}[f](t) + b\mathcal{H}[g](t).$$

Let the Hilbert-lifted analytic representation of a real-valued signal $x(t)$ be

$$z(t) = x(t) + i\mathcal{H}[x](t),$$

where $x(t)$ is the real component and $\mathcal{H}[x](t)$ is the quadrature component. The instantaneous phase is given by

$$\phi(t) = \arg z(t) = \arg\left(x(t) + i\mathcal{H}[x](t)\right).$$

For a normalized signal

$$\tilde{x}(t) = \frac{x(t) - \mu}{\sigma}, \qquad \sigma > 0,$$

the linearity of the Hilbert transform gives

$$\mathcal{H}[\tilde{x}](t) = \frac{1}{\sigma}\mathcal{H}[x](t),$$

since the Hilbert transform of a constant offset is zero. Therefore, the normalized analytic representation becomes

$$\tilde{z}(t) = \frac{x(t) - \mu}{\sigma} + i\frac{\mathcal{H}[x](t)}{\sigma}.$$

The positive scaling factor $1/\sigma$ is applied consistently to both the real and quadrature components, so it preserves the phase structure of the analytic signal. In the centered case where $\mu = 0$, the normalized analytic representation is

$$\tilde{z}(t) = \frac{1}{\sigma}z(t),$$

and the instantaneous phase is exactly preserved:

$$\tilde{\phi}(t) = \phi(t).$$

When $\mu \neq 0$, the centering term acts on the real component and introduces a controlled phase perturbation. The normalized phase can be written as

$$\tilde{\phi}(t) = \arg\left((x(t) - \mu) + i\mathcal{H}[x](t)\right).$$

For signal regions where the mean offset is small relative to the local signal magnitude, i.e., $|\mu| \ll |x(t)|$, the phase variation is bounded by a first-order perturbation:

$$|\tilde{\phi}(t) - \phi(t)| = \mathcal{O}\left(\frac{|\mu|}{|x(t)|}\right).$$

Thus, positive scaling preserves phase exactly for any fixed linear filter, while the centering conclusion applies under the stated small-offset condition and, for a finite filter F, additionally requires $F\mathbf{1} = 0$.

H Energy Conservation and Hamiltonian Structure in the Hilbert-Lifted Complex Space

The following analysis establishes the conservation and Hamiltonian structure of the continuous dynamics that motivate SDHilb's real–quadrature coupling. Let

$$x(t) = x_r(t) + \mathbf{i}x_i(t) \in \mathbb{C}^d$$

denote the lifted complex state, where $x_r(t)$ and $x_i(t)$ are the real and quadrature components produced by the Hilbert lifting. We define the complex-state energy as the squared norm of the lifted state:

$$E(t) = \|x(t)\|_{\mathbb{C}^d}^2 = x_r(t)^\top x_r(t) + x_i(t)^\top x_i(t).$$

This energy characterizes the amplitude structure of the analytic-signal state in the Hilbert-lifted complex space and captures the joint evolution of the real and quadrature components.

We consider the general linear complex dynamics

$$\mathbf{i}\frac{d}{dt}x(t) = (A_r + \mathbf{i}A_i)x(t), \tag{9}$$

where $A_r, A_i \in \mathbb{R}^{d\times d}$. Decomposing Eq. (9) into real and imaginary parts gives

$$\begin{cases} \dfrac{dx_r}{dt} = A_r x_i + A_i x_r, \\ \dfrac{dx_i}{dt} = -A_r x_r + A_i x_i. \end{cases}$$

The time derivative of the complex-state energy is

$$\begin{aligned} \frac{dE}{dt} &= \frac{d}{dt}\left(x_r^\top x_r + x_i^\top x_i\right) \\ &= 2x_r^\top \frac{dx_r}{dt} + 2x_i^\top \frac{dx_i}{dt} \\ &= 2x_r^\top A_r x_i + 2x_r^\top A_i x_r - 2x_i^\top A_r x_r + 2x_i^\top A_i x_i. \end{aligned} \tag{10}$$

If $A_r = A_r^\top$, then $x_i^\top A_r x_r = x_r^\top A_r x_i$, so the two cross terms cancel. If $A_i = 0$, the remaining terms vanish. Therefore, under the structural conditions

$$A_r = A_r^\top, \qquad A_i = 0, \tag{11}$$

the complex-state energy satisfies

$$\frac{dE}{dt} = 0.$$

Under these conditions, the dynamics reduce to

$$\begin{cases} \dfrac{dx_r}{dt} = A_r x_i, \\ \dfrac{dx_i}{dt} = -A_r x_r. \end{cases}$$

Let $z = [x_r^\top, x_i^\top]^\top$. The system can be written as

$$\frac{d}{dt}\begin{bmatrix} x_r \\ x_i \end{bmatrix} = \begin{bmatrix} 0 & A_r \\ -A_r & 0 \end{bmatrix}\begin{bmatrix} x_r \\ x_i \end{bmatrix}. \tag{12}$$

Equivalently, it admits a Hamiltonian form

$$\dot{z} = J\nabla H(z), \tag{13}$$

where

$$J = \begin{bmatrix} 0 & I \\ -I & 0 \end{bmatrix}, \qquad J^\top = -J,$$

and

$$H(z) = \frac{1}{2} x_r^\top A_r x_r + \frac{1}{2} x_i^\top A_r x_i. \tag{14}$$

This formulation shows that the real and quadrature components form a coupled Hamiltonian state, with the antisymmetric structure governing their joint evolution.

Theorem 2 (Energy conservation in the Hilbert-lifted complex space) *Let* $x(t) = x_r(t) + \mathbf{i}x_i(t) \in \mathbb{C}^d$ *evolve according to*

$$\mathbf{i}\frac{d}{dt}x(t) = A_r x(t),$$

where $A_r \in \mathbb{R}^{d\times d}$ *is symmetric. Then the complex-state energy*

$$E(t) = x_r(t)^\top x_r(t) + x_i(t)^\top x_i(t)$$

is conserved along the continuous-time trajectory.

Proof. For $\mathbf{i}\dot{x} = A_r x$, we have $A_i = 0$ in Eq. (9). Since $A_r = A_r^\top$, the cross terms in Eq. (10) cancel and the remaining A_i terms vanish. Thus $dE/dt = 0$, which proves that $E(t)$ is conserved. □

This derivation motivates the coupled real–quadrature prediction form used in SDHilb. Its closed-form recurrence provides the basis for the projection-based parameter reuse described in Section 3.4.

I Mathematical Details of Hilbert Kernel Construction

This section provides the mathematical details of the adaptive multi-scale Hilbert kernel construction used in SDHilb. The goal is to construct a quadrature component for finite-length forecasting windows while allowing the effective Hilbert receptive field to adapt across channels and datasets.

I.1 Hann-Windowed Hilbert Kernels

For a real-valued signal x, the ideal Hilbert transform and its Fourier-domain relation are

$$\mathcal{H}[x](t) = \frac{1}{\pi}\,\mathrm{p.\,v.}\int_{\mathbb{R}} \frac{x(s)}{t-s}\,ds, \qquad \widehat{\mathcal{H}x}(\omega) = -\mathbf{i}\,\mathrm{sgn}(\omega)\widehat{x}(\omega),$$

where $\mathrm{p.\,v.}$ denotes the Cauchy principal value and $\mathrm{sgn}(0) = 0$. The ideal transform is two-sided. SDHilb applies its finite filters only to the observed look-back window X available at the forecast origin. Centered filtering can use neighboring positions on both sides within X, without accessing samples from the forecast interval.

For each candidate kernel size $w_k \in \mathcal{K}$, we first ensure an odd kernel length for center-aligned convolution:

$$w_k \leftarrow \begin{cases} w_k + 1, & \text{if } w_k \text{ is even,} \\ w_k, & \text{otherwise.} \end{cases}$$

Let $m_k = \lfloor w_k/2 \rfloor$ denote the center index. The kernel is initialized by sampling the continuous Hilbert kernel and applying a Hann window:

$$h_k[n] = \begin{cases} \dfrac{\mathrm{Hann}_k[n]}{\pi(n-m_k)}, & n \neq m_k, \\ 0, & n = m_k, \end{cases} \qquad n = 0, \ldots, w_k - 1,$$

where

$$\mathrm{Hann}_k[n] = 0.5\left(1 - \cos\left(\frac{2\pi n}{w_k - 1}\right)\right).$$

The Hann window smoothly decays to zero at the kernel boundaries, reducing boundary-induced artifacts caused by finite-window convolution. This design provides a stable quadrature approximation for the Hilbert-lifted representation.

Algorithm 1 Adaptive Hann-Windowed Hilbert Kernel Construction

Require: Input representation $X \in \mathbb{R}^{B\times L\times C}$, candidate kernel sizes $\mathcal{K}$
Ensure: Analytic signal $Z = X + \mathbf{i}H_{\text{multi}}(X)$
1: **for** each kernel size $w_k \in \mathcal{K}$ **do**
2: Set $w_k \leftarrow w_k + 1$ if w_k is even
3: Let $m_k \leftarrow \lfloor w_k/2 \rfloor$
4: Construct the Hann window $\text{Hann}_k[n]$
5: Construct $h_k[n] = \text{Hann}_k[n]/[\pi(n - m_k)]$ for $n \neq m_k$, and set $h_k[m_k] = 0$
6: **end for**
7: Compute channel-wise weights $\alpha_{c,k}$ by Gumbel-Softmax
8: Compute $H_{\text{multi}}(X)_c \leftarrow \sum_k \alpha_{c,k} \cdot \text{DepthwiseConv1D}(X_c, h_k)$
9: Construct $Z \leftarrow X + \mathbf{i}H_{\text{multi}}(X)$

I.2 Channel-Wise Multi-Scale Kernel Fusion

To adapt the Hilbert receptive field to heterogeneous temporal patterns, SDHilb learns channel-wise weights over multiple kernel sizes. For channel c and kernel size k, the scale-selection weight is computed by Gumbel-Softmax:

$$\alpha_{c,k} = \frac{\exp((W_{c,k} + g_{c,k})/\tau)}{\sum_{j=1}^{K} \exp((W_{c,j} + g_{c,j})/\tau)}, \qquad g_{c,k} \sim \text{Gumbel}(0, 1),$$

where $W_{c,k}$ is a learnable scale logit and τ is the temperature. Given the input representation X, the multi-scale Hilbert component is computed as

$$H_{\text{multi}}(X)_c = \sum_{k=1}^{K} \alpha_{c,k} \cdot \text{DepthwiseConv1D}(X_c, h_k).$$

The analytic signal used by SDHilb is then constructed as

$$Z = X + \mathbf{i}H_{\text{multi}}(X).$$

This channel-wise fusion allows each variable to select an effective Hilbert scale according to its temporal characteristics, rather than relying on a fixed Hilbert kernel for all channels.

I.3 Spectral Leakage Reduction

The Hann-windowed construction reduces spectral leakage introduced by finite-window observations. Compared with a rectangular truncation, the Hann window suppresses side lobes more effectively and reduces leakage from dominant frequency components into neighboring bands. This is useful for constructing a cleaner quadrature component in non-stationary forecasting windows, where abrupt local changes and boundary effects can otherwise introduce high-frequency artifacts.

The multi-scale design further balances locality and smoothness. Shorter kernels provide stronger local responsiveness, while longer kernels capture smoother and longer-period temporal patterns. By learning channel-wise kernel weights, SDHilb adapts this trade-off across heterogeneous variables and constructs a more stable Hilbert-lifted complex representation for downstream structured dynamics.

J Derivation of the Recurrence Formula

For the ideal Hamiltonian system, a symplectic Euler discretization preserves the underlying symplectic structure and supports stable long-horizon evolution:

$$\begin{cases} \mathbf{x}_r^{(t+\Delta t)} = \mathbf{x}_r^{(t)} + \Delta t \cdot \mathbf{A}_r \mathbf{x}_i^{(t)}, \\ \mathbf{x}_i^{(t+\Delta t)} = \mathbf{x}_i^{(t)} - \Delta t \cdot \mathbf{A}_r \mathbf{x}_r^{(t+\Delta t)}. \end{cases} \tag{15}$$

The key property is the sequential update: the imaginary component is updated using the newly computed real component. This sequential structure preserves the symplectic geometry of the lifted

Hamiltonian system and, under standard stability conditions on the step size and the spectrum of $\mathbf{A}_r$, supports bounded long-horizon energy behavior. The structural conditions $\mathbf{A}_r = \mathbf{A}_r^\top$ and $\mathbf{A}_i = \mathbf{0}$ derived in Appendix H guarantee energy conservation for the underlying continuous system.

For notational simplicity, write $\mathbf{x}_r^{(n)} := \mathbf{x}_r^{(t+n\Delta t)}$ and $\mathbf{x}_i^{(n)} := \mathbf{x}_i^{(t+n\Delta t)}$, and let

$$\mathbf{D} \;:=\; 2\mathbf{I} - (\Delta t)^2 \mathbf{A}_r^2.$$

Reduction to a second-order recurrence. Eliminating $\mathbf{x}_i$ from Eq. (15) yields a closed second-order recurrence for $\mathbf{x}_r^{(n)}$ alone. Using the first equation at step n and the second equation at step $n-1$,

$$\mathbf{x}_r^{(n+1)} - \mathbf{x}_r^{(n)} \;=\; \Delta t \cdot \mathbf{A}_r \mathbf{x}_i^{(n)} \;=\; \Delta t \cdot \mathbf{A}_r \Big(\mathbf{x}_i^{(n-1)} - \Delta t \cdot \mathbf{A}_r \mathbf{x}_r^{(n)} \Big),$$

and substituting $\Delta t \cdot \mathbf{A}_r \mathbf{x}_i^{(n-1)} = \mathbf{x}_r^{(n)} - \mathbf{x}_r^{(n-1)}$ gives

$$\boxed{\mathbf{x}_r^{(n+1)} \;=\; \mathbf{D}\,\mathbf{x}_r^{(n)} \;-\; \mathbf{x}_r^{(n-1)}.} \tag{16}$$

This is a Chebyshev-type linear recurrence with characteristic operator $\mathbf{D}$.

Closed form via Chebyshev-type matrix polynomials. Define two sequences of matrix polynomials in $\mathbf{A}_r$ (equivalently, in $\mathbf{D}$) by the same recurrence

$$P_{n+1} \;=\; \mathbf{D}\,P_n \;-\; P_{n-1}, \tag{17}$$

with initial conditions

$$\mathcal{U}_0 = \mathcal{U}_1 = \mathbf{I}, \qquad \mathcal{V}_0 = \mathbf{0}, \;\; \mathcal{V}_1 = \mathbf{I}.$$

We claim

$$\mathbf{x}_r^{(n)} \;=\; \mathcal{U}_n(\mathbf{A}_r)\,\mathbf{x}_r^{(0)} \;+\; \Delta t \cdot \mathbf{A}_r\,\mathcal{V}_n(\mathbf{A}_r)\,\mathbf{x}_i^{(0)}. \tag{18}$$

Proof by induction on n.

Base cases $n = 0, 1$. For $n = 0$ both sides equal $\mathbf{x}_r^{(0)}$. For $n = 1$, $\mathcal{U}_1 = \mathbf{I}$ and $\mathcal{V}_1 = \mathbf{I}$, so the right-hand side equals $\mathbf{x}_r^{(0)} + \Delta t \cdot \mathbf{A}_r \mathbf{x}_i^{(0)}$, which matches Eq. (15).

Inductive step. Assume the claim holds for $n-1$ and n. Since $\mathbf{D}$, $\mathcal{U}_n$, and $\mathcal{V}_n$ are all polynomials in $\mathbf{A}_r$, they commute with $\mathbf{A}_r$. Using Eq. (16) together with Eq. (17),

$$\begin{aligned}
\mathbf{x}_r^{(n+1)} &= \mathbf{D}\,\mathbf{x}_r^{(n)} - \mathbf{x}_r^{(n-1)} \\
&= \mathbf{D}\Big(\mathcal{U}_n \mathbf{x}_r^{(0)} + \Delta t \cdot \mathbf{A}_r \mathcal{V}_n \mathbf{x}_i^{(0)}\Big) - \Big(\mathcal{U}_{n-1} \mathbf{x}_r^{(0)} + \Delta t \cdot \mathbf{A}_r \mathcal{V}_{n-1} \mathbf{x}_i^{(0)}\Big) \\
&= \big(\mathbf{D}\mathcal{U}_n - \mathcal{U}_{n-1}\big)\mathbf{x}_r^{(0)} + \Delta t \cdot \mathbf{A}_r \big(\mathbf{D}\mathcal{V}_n - \mathcal{V}_{n-1}\big)\mathbf{x}_i^{(0)} \\
&= \mathcal{U}_{n+1}(\mathbf{A}_r)\,\mathbf{x}_r^{(0)} + \Delta t \cdot \mathbf{A}_r\,\mathcal{V}_{n+1}(\mathbf{A}_r)\,\mathbf{x}_i^{(0)},
\end{aligned}$$

which completes the induction.

Closed form for $\mathbf{x}_i^{(n)}$. Iterating the second line of Eq. (15) gives

$$\mathbf{x}_i^{(n)} \;=\; \mathbf{x}_i^{(0)} \;-\; \Delta t \cdot \mathbf{A}_r \sum_{k=1}^{n} \mathbf{x}_r^{(k)}.$$

Substituting Eq. (18) into this identity yields

$$\mathbf{x}_i^{(n)} \;=\; -\,\Delta t \cdot \mathbf{A}_r \sum_{k=1}^{n} \mathcal{U}_k(\mathbf{A}_r)\,\mathbf{x}_r^{(0)} \;+\; \Big[\mathbf{I} - (\Delta t)^2 \mathbf{A}_r^2 \sum_{k=1}^{n} \mathcal{V}_k(\mathbf{A}_r)\Big]\mathbf{x}_i^{(0)}. \tag{19}$$

Equations (18) and (19) together give the complete closed form of the symplectic recursion.

Explicit finite-sum representation. The recurrence (17) is the standard second-order Chebyshev recurrence with characteristic operator $\mathbf{D}$, so $\mathcal{V}_n$ admits the closed expansion

$$\mathcal{V}_n(\mathbf{A}_r) \;=\; \sum_{j=0}^{\lfloor (n-1)/2 \rfloor} (-1)^j \binom{n-1-j}{j} \mathbf{D}^{\,n-1-2j}, \qquad n \geq 1, \tag{20}$$

and $\mathcal{U}_n$ follows from the linear relation

$$\mathcal{U}_n(\mathbf{A}_r) \;=\; \mathcal{V}_n(\mathbf{A}_r) - \mathcal{V}_{n-1}(\mathbf{A}_r), \qquad \mathcal{V}_0(\mathbf{A}_r) := \mathbf{0},\;\; \mathcal{U}_0(\mathbf{A}_r) := \mathbf{I}. \tag{21}$$

Both Eqs. (20) and (21) can be verified directly against the recurrence (17) by standard binomial identities.

The first few polynomials read

$$\begin{aligned} &\mathcal{U}_1 = \mathbf{I}, \quad \mathcal{U}_2 = \mathbf{I} - (\Delta t)^2 \mathbf{A}_r^2, \quad \mathcal{U}_3 = \mathbf{I} - 3(\Delta t)^2 \mathbf{A}_r^2 + (\Delta t)^4 \mathbf{A}_r^4, \\ &\mathcal{V}_1 = \mathbf{I}, \quad \mathcal{V}_2 = 2\mathbf{I} - (\Delta t)^2 \mathbf{A}_r^2, \quad \mathcal{V}_3 = 3\mathbf{I} - 4(\Delta t)^2 \mathbf{A}_r^2 + (\Delta t)^4 \mathbf{A}_r^4, \end{aligned}$$

and higher orders follow from Eq. (17).

Connection to matrix trigonometric functions. Let

$$\mathbf{C} \;:=\; \frac{1}{2}\mathbf{D} \;=\; \mathbf{I} - \frac{1}{2}(\Delta t)^2 \mathbf{A}_r^2.$$

The recurrence (17) is the standard second-order Chebyshev recurrence; hence, for $n \geq 1$,

$$\mathcal{V}_n(\mathbf{A}_r) \;=\; U_{n-1}^{\mathrm{cheb}}(\mathbf{C}), \qquad \mathcal{U}_n(\mathbf{A}_r) \;=\; U_{n-1}^{\mathrm{cheb}}(\mathbf{C}) - U_{n-2}^{\mathrm{cheb}}(\mathbf{C}),$$

where U_n^{cheb} denotes the Chebyshev polynomial of the second kind, with the convention $U_{-1}^{\mathrm{cheb}} := \mathbf{0}$, and matrix functions are interpreted spectrally.

When the spectrum of $\mathbf{C}$ lies in $[-1, 1]$ (equivalently, under $\mathbf{A}_r = \mathbf{A}_r^\top$, $\|(\Delta t)\mathbf{A}_r\|_2 \leq 2$), we may write $\mathbf{C} = \cos \mathbf{\Theta}$ with

$$\mathbf{\Theta} \;:=\; \arccos(\mathbf{C}) \;=\; \arccos\Big(\mathbf{I} - \tfrac{1}{2}(\Delta t)^2 \mathbf{A}_r^2\Big).$$

The quotients below are interpreted spectrally through their continuous extensions at eigenvalues where $\sin \mathbf{\Theta}$ vanishes. Then

$$\mathcal{V}_n(\mathbf{A}_r) = \frac{\sin(n\mathbf{\Theta})}{\sin \mathbf{\Theta}}, \qquad \mathcal{U}_n(\mathbf{A}_r) = \frac{\sin(n\mathbf{\Theta}) - \sin\big((n-1)\mathbf{\Theta}\big)}{\sin \mathbf{\Theta}}.$$

so that

$$\mathbf{x}_r^{(n)} \;=\; \frac{\sin(n\mathbf{\Theta}) - \sin\big((n-1)\mathbf{\Theta}\big)}{\sin \mathbf{\Theta}}\, \mathbf{x}_r^{(0)} \;+\; \Delta t \cdot \mathbf{A}_r \, \frac{\sin(n\mathbf{\Theta})}{\sin \mathbf{\Theta}}\, \mathbf{x}_i^{(0)}.$$

For fixed $\mathbf{A}_r = \mathbf{A}_r^\top$ and fixed initial conditions, as $\Delta t \to 0$ with the elapsed evolution time $n\Delta t$ held fixed, the discrete solution converges to the matrix-trigonometric solution of the underlying continuous Hamiltonian flow:

$$\begin{aligned} \mathbf{x}_r^{(n)} \;&\approx\; \cos\big(n\Delta t\, \mathbf{A}_r\big)\mathbf{x}_r^{(0)} \;+\; \sin\big(n\Delta t\, \mathbf{A}_r\big)\mathbf{x}_i^{(0)}, \\ \mathbf{x}_i^{(n)} \;&\approx\; -\sin\big(n\Delta t\, \mathbf{A}_r\big)\mathbf{x}_r^{(0)} \;+\; \cos\big(n\Delta t\, \mathbf{A}_r\big)\mathbf{x}_i^{(0)}. \end{aligned} \tag{22}$$

This recovers the oscillatory, energy-preserving structure of the continuous Schrödinger flow $\mathbf{i}\, d\mathbf{x}/dt = \mathbf{A}_r \mathbf{x}$ and provides the principled foundation for translating continuous Hamiltonian flows into the discrete recurrence used in SDHilb. The identification of Eq. (18) with shared, n-indexed Chebyshev-type transitions is precisely what justifies the parameter-reuse strategy of Eq. (8) in the main text.

K Limitations and Future Work

Despite the promising results achieved by SDHilb, several limitations remain and merit further investigation. First, although the adaptive multi-scale Hilbert module mitigates spectral leakage to some extent, its performance on signals with highly localized transient events (such as sharp spikes or abrupt discontinuities) has not been thoroughly examined. Second, the current implementation is derived from the Schrödinger equation and has only been explored in time series forecasting tasks. Future work could incorporate brain-inspired neural structures to develop more complex models applicable to a broader range of tasks. Third, the conservation-related analysis is derived under idealized continuous-time Hamiltonian conditions and should be interpreted as a stability-oriented inductive bias in the Hilbert-lifted complex space rather than as an unconditional guarantee for arbitrary learned dynamics. Further analysis of stability under large step sizes or poorly conditioned transition matrices remains an important direction for future work.

To address these limitations, we propose several future research directions. First, the Hilbert module could be extended by introducing a richer variety of kernel shapes to improve adaptability to diverse signal morphologies. Second, integrating nonlinear Schrödinger dynamics or coupling with neural operators may better capture complex temporal dependencies. Third, exploring higher-order symplectic integrators, adaptive step-size schemes, or implicit discretization methods could further strengthen long-horizon stability in the Hilbert-lifted complex space. Finally, applying SDHilb to emerging domains such as neuromorphic signal processing, quantum sensor data, or cross-modal forecasting (e.g., video-text-time series fusion) could not only broaden its practical applicability but also serve to validate its generalization capabilities. These efforts may further bridge the gap between physics-inspired modeling and the real-world demands of time series analysis.

L Broader Societal Impact

This study proposes an improved method for time series forecasting, aiming to enhance prediction accuracy and parameter efficiency for general-purpose forecasting tasks. While the research is foundational and not tailored to any specific application domain, time series forecasting is crucial in socially relevant areas such as energy demand, financial risk, climate modeling, and infrastructure planning. Advances in forecasting models can aid decision-making in these domains, supporting efficient resource allocation, public health responsiveness, and economic stability.

This study involves no human-subject experiments or automated deployment. In downstream applications, inaccurate forecasts may lead to inefficient resource allocation, motivating application-specific validation before deployment.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction state the main contributions of SDHilb, including the adaptive multi-scale Hilbert representation, the Schrödinger-Hamiltonian motivated structured recurrence, and the resulting accuracy–efficiency trade-off. The claims are supported by the theoretical formulation, ablation studies, and experiments on eight long-term forecasting benchmarks.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The paper discusses limitations related to the modeling assumptions, dataset scope, and empirical evaluation setting. It also reports architecture-level computational efficiency using parameter counts and MACs to clarify the scalability of the proposed method.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: The paper provides the assumptions, derivations, and recurrence formulations underlying the Schrödinger-Hamiltonian motivated dynamics. Formal derivations and additional explanations are provided in the main text and appendix, with equations and related results cross-referenced.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: The experimental setup and training procedure are described in Section 4 and Appendix C.1.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: Code repository: `https://github.com/huahua11211/SDHilb`.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Section 4 and Appendix C.1 describe the experimental setup and hyperparameter selection procedure.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: The relevant details are provided in Appendix E.4.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.

- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: The paper reports the experimental hardware, parameter counts, and MACs, with implementation details provided in the appendix. Appendix C.2 and Table 5 additionally document the timing protocol, training time per epoch, full-test time, forward latency, and peak GPU memory for six models under a common Electricity setting.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The work uses public benchmark time-series datasets and does not involve private personal data, human-subject experiments, or high-risk deployment. The research is conducted in accordance with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [Yes]

Justification: Potential benefits and risks are discussed in Appendix L.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

Answer: [N/A]

Justification: The paper does not release a pre-trained language model, image generator, scraped dataset, or other asset with high misuse or dual-use risk. The work focuses on time-series forecasting with public benchmark datasets.

Guidelines:

- The answer [N/A] means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: The paper cites the original sources of the benchmark datasets and baseline methods used in the experiments. Existing assets are used for research benchmarking purposes, and their corresponding sources and terms are respected.

Guidelines:

- The answer [N/A] means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: The paper does not introduce a new dataset or benchmark asset. The main contribution is a new forecasting architecture evaluated on existing public benchmark datasets.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing experiments or research with human subjects. All experiments are conducted on existing public time-series benchmark datasets.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [N/A]

Justification: The paper does not involve crowdsourcing or human-subject research, and therefore IRB approval or equivalent review is not applicable.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: LLMs are not used as an important, original, or non-standard component of the proposed method, theoretical development, or experimental evaluation. Any use of writing or editing assistance does not affect the core methodology, scientific rigor, or originality of the research.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.