PrismFlow: Residual Dynamics for Flow Matching in Time-Series Generation

Junru Zhang[1] **Lang Feng**[2] **Jinbo Wang**[1] **Xu Guo**[2] **Yucheng Wang**[3,4]
Han Yu[2] **Min Wu**[3,4*] **Yabo Dong**[1*] **Duanqing Xu**[1]
[1]Zhejiang University, China
[2]Nanyang Technological University, Singapore
[3]Institute of Advanced Intelligence and Computing (IAIC), A*STAR, Singapore
[4]AI Centre of Excellence for Manufacturing (AIMfg), A*STAR, Singapore

Abstract

Generating high-quality time-series data is challenging because real-world signals often exhibit multimodal patterns and multiscale dynamics, including oscillations and high-frequency variations. Flow Matching (FM) offers an efficient alternative to diffusion models, but practical implementations typically rely on a single finite-capacity global vector-field estimator. In such heterogeneous temporal distributions, distinct regimes may pass through nearby flow states while requiring incompatible conditional velocities. A monolithic estimator trained with the standard ℓ_2 velocity-matching objective may therefore learn an overly smoothed approximation of the local transport field. This estimator-level smoothing can attenuate branch-specific dynamics, leading to spectral distortion and poor mode coverage. To address this, we propose PrismFlow, a new FM method with Koopman-inspired dynamical experts. Each expert learns residual corrections in a latent space where local nonlinear temporal evolution can be approximated by linear transitions. We further propose a confidence-aware Winner-Take-All (WTA) objective that updates only the expert best aligned with each sample while masking gradients to the others, encouraging mode-specific specialization. During sampling, the selected expert adds a residual dynamical correction to the global transport field, preserving FM stability while recovering fine-grained and high-frequency temporal structures. Across various benchmarks, PrismFlow effectively mitigates the spectral contraction in standard FM and achieves state-of-the-art performance, with a 15.6% gain in Context-FID and a 38.6% improvement in Discriminative Score, while remaining robust in low-data settings and effective for forecasting and imputation. Our code is available at `https://github.com/jrzhang33/PrismFlow`.

1 Introduction

Time-series data underpin decision-making in domains such as healthcare [1, 2], finance [3–5], and environmental monitoring [6–8]. However, the acquisition of high-fidelity signals is frequently restricted by stringent privacy regulations and prohibitive costs [9, 10]. This scarcity necessitates generative models capable of synthesizing sequences that are not only statistically consistent but also preserve the underlying temporal evolution of real-world phenomena.

Recent advances in time-series generation have shifted from adversarial frameworks [11, 12] toward simulation-free continuous transport methods, most notably Flow Matching (FM) [13, 14]. While FM offers superior stability and sampling efficiency compared to diffusion models [15, 16], its practical implementation typically relies on a *single global vector-field estimator* to approximate the

*Corresponding authors.

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

transport dynamics [17]. Although the exact FM transport field is well-defined in principle, learning it from finite data with a monolithic estimator introduces statistical and representational challenges. In multi-modal temporal scenarios, such as those with varying frequencies or transient responses, heterogeneous temporal patterns may occupy nearby regions of the flow state space while requiring distinct sequence evolutions. Under the standard ℓ_2 velocity-matching objective, the estimator is therefore encouraged to predict the conditional mean of incompatible target velocities, which tends to smooth out mode-specific time-series patterns. As a result, rich temporal structure can collapse into an overly smooth learned flow with constrained spectral diversity, leading to *mode collapse* [18–22]. This degrades local trajectory fidelity and restricts the expressive capacity of flow-based generators by limiting their ability to reproduce diverse dynamical patterns.

To mitigate this mode collapse, we propose PrismFlow, a new FM method that corrects the global transport field using a bank of Koopman-inspired dynamical experts. Drawing on Koopman operator theory [23, 24], our method maps nonlinear temporal evolutions into a latent space where they can be modeled as local linear transitions. Rather than forcing a single learned estimator to capture all dynamics, we treat generation as a *routing* problem: at each intermediate state, the model dynamically identifies the local temporal pattern and assigns responsibility to the expert best suited to model that specific evolution. To ensure these experts learn distinct modes, we introduce a Winner-Take-All (WTA) training objective with competitive selection. Unlike standard mixtures with soft gating [25, 26], which can reintroduce averaging effects, our approach enforces clearer expert specialization. For a given state, all experts predict candidate residual velocities, but only the expert with the best confidence-aware WTA score receives the specialization update, while gradients to the others are masked. This mechanism reduces the averaging of incompatible modes, which is the primary cause of regression-to-the-mean behavior in the learned monolithic estimator. During sampling, PrismFlow integrates the selected expert as a *residual dynamical correction* term added to the global transport field, thus preserving the stability of the global flow while allowing the model to recover diverse spectral components across multiple scales.

Our main contributions are summarized as follows:

- We characterize velocity averaging in learned FM models as a key bottleneck for time-series generation. Our analysis shows that single-field estimators can induce conditional-mean behavior, leading to spectral distortion and mode collapse in generated time series.
- We propose PrismFlow, a novel method that mitigates mode collapse in time-series generation. By integrating Koopman-inspired experts as residual dynamical correction terms to the global flow, we capture multi-scale dynamics that standard monolithic estimators often fail to represent.
- We propose training these experts with a WTA competitive selection rule. By assigning each flow state to a single expert and masking gradients to the others, this objective reduces regression-to-the-mean behavior under the standard ℓ_2 loss and promotes clear mode-specific specialization.
- Empirical evaluations of PrismFlow demonstrate strong performance across diverse time-series generation tasks. PrismFlow effectively recovers diverse modes, improving Context-FID by 15.6% and Discriminative Score by 38.6%, while remaining robust in low-data settings and high-fidelity for forecasting and imputation.

2 Related Work

Time-Series Generation. Time-series generation has progressed from early adversarial [27, 28, 12, 29, 30] and latent-variable [31, 32] models toward iterative refinement methods that deliver higher fidelity. Representative baselines such as TimeGAN [12] and TimeVAE [31] capture global temporal dependencies, but they often suffer from training instability and tend to blur local, phase-sensitive dynamics. More recently, diffusion-based models [33–36], including Diffwave [35] and Diffusion-TS [36], have achieved strong synthesis quality by casting generation as progressive denoising. SDformer [37] further explores discrete sequence modeling with a large-parameter diffusion model. While diffusion models provide impressive fidelity and distributional alignment, their many-step inference remains computationally expensive, limiting real-time and large-scale deployment.

Flow Matching. Flow Matching (FM) is a simulation-free framework for training continuous normalizing flows, combining stable objectives with efficient ordinary differential equation (ODE)-based sampling [13, 14]. By regressing time-dependent velocity fields along predefined probability paths, FM avoids expensive trajectory simulations during training and enables deterministic generation

via standard solvers. This efficiency has yielded strong results in image synthesis [38], video generation [39], stable neural ODE dynamics [40], time-series foundation modeling [41, 42], and probabilistic forecasting [43, 44]. Recently, TimeMCL [45] introduced a multiple-choice learning approach for forecasting diverse futures. Complementary to such output-level diversity methods, our work focuses on estimator-level averaging in monolithic FM implementations. When heterogeneous temporal modes pass through nearby flow states, the standard ℓ_2 objective may drive a single estimator toward the conditional mean of incompatible velocities, yielding over-smoothed trajectories and reduced spectral diversity. Rather than treating this as a flaw of the exact FM transport field, we address the practical limitation of its finite-sample approximation by augmenting the global estimator with dynamically routed Koopman-inspired residual experts.

3 Preliminaries

Problem Setting. Let $x \in \mathcal{X} = \mathbb{R}^{S \times D}$ denote a multivariate time series with S temporal steps and D channels. Each sequence x is drawn from an unknown data distribution $q(x)$ over $\mathcal{X}$. The goal of generative modeling is to learn a parameterized distribution p_θ that approximates q, enabling the synthesis of sequences $\hat{x}$ that preserve both the statistical properties and temporal dynamics of real-world data.

Flow Matching. Flow Matching (FM) [13] is a simulation-free framework for training Continuous Normalizing Flows. It generates samples by transporting a simple source distribution p_0 to the data distribution through a time-dependent vector field $v_t^\theta : [0,1] \times \mathcal{X} \to \mathcal{X}$. The transformation is governed by the ODE:

$$\frac{\mathrm{d}}{\mathrm{d}t} x_t = v_t^\theta(x_t, t), \qquad x_{t=0} = x_0, \tag{1}$$

which induces a probability path $\{p_t\}_{t\in[0,1]}$ with $x_t \sim p_t$ for $x_0 \sim p_0$. We parameterize $v_t^\theta(x_t, t)$ with an encoder-decoder network whose parameters are denoted by $\theta = (\phi_\eta, \phi_\zeta)$. Following standard Conditional Flow Matching (CFM), we adopt the linear interpolation between $x_0 \sim p_0 = \mathcal{N}(\mathbf{0}, \mathbf{I})$ and $x_1 \sim q$: $x_t = (1-t)x_0 + tx_1$, whose target velocity is constant:

$$\frac{\mathrm{d}}{\mathrm{d}t} x_t = x_1 - x_0. \tag{2}$$

The model is trained by minimizing:

$$\mathcal{L}_{\mathrm{CFM}}(\theta) = \mathbb{E}_{t,x_0,x_1}\left[\left\| v_t^\theta(x_t, t) - (x_1 - x_0)\right\|_2^2\right], \tag{3}$$

where $t \sim \mathcal{U}[0,1]$. To generate a sample, we draw $x_0 \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ and integrate the learned ODE from $t = 0$ to $t = 1$:

$$x_{t+\Delta t} = x_t + v_t^\theta(x_t, t)\Delta t, \tag{4}$$

using a numerical solver to obtain $\hat{x} = x_{t=1}$.

Mode Collapse. While the ℓ_2 objective in Eq. (3) is effective, practical single-field estimators can become mean-seeking in multi-modal temporal settings. In standard FM, when trajectories from different temporal regimes pass through nearby flow states x_t, the local target velocity distribution $q(u_t \mid x_t)$ can become highly heterogeneous, where $u_t = x_1 - x_0$ denotes the CFM target velocity. Under the ℓ_2 loss, a finite-capacity estimator may learn a smoothed approximation around the conditional-average trend, $v_t^*(x_t, t) = \mathbb{E}[u_t \mid x_t]$, rather than effectively preserving branch-specific velocity directions. When nearby samples correspond to incompatible temporal regimes, this estimator-level averaging can reduce the effective velocity energy, $\|\mathbb{E}[u_t \mid x_t]\|_2^2 \le \mathbb{E}[\|u_t\|_2^2 \mid x_t]$, and may attenuate transient branches and high-frequency components after ODE integration. From a Dynamic Mode Decomposition (DMD) perspective [46], this practical smoothing manifests as spectral contraction, where energy concentrates into a few slowly varying modes while faster or weaker modes vanish. We study this specific form of mode collapse [19, 20] in time-series generation, which is further validated by a Gaussian-mixture diagnostic in Appendix D.

4 Methodology

We introduce PrismFlow, a new flow matching method designed to mitigate the estimator-level mean seeking. As shown in Fig. 1, PrismFlow preserves the global transport process of FM while

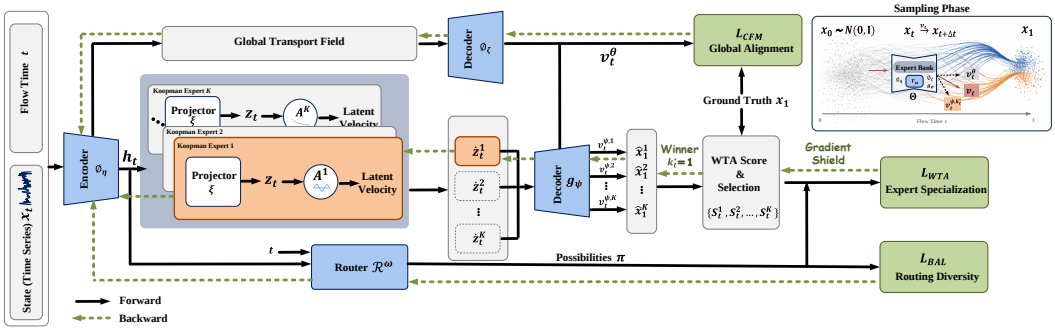


Figure 1: Overall architecture of PrismFlow. Given a flow state x_t, a flow matching backbone predicts a global transport velocity v_t^θ. In parallel, a shared encoder and projector map x_t into a latent Koopman space, where a bank of linear experts models different temporal modes. Each expert evolves its latent state through a linear velocity, which is decoded by g_ψ into a candidate residual velocity $\{v_t^{\psi,k}\}_{k=1}^K$. A router produces expert probabilities π_t, and a WTA score selects the dominant expert k_t^*. During training, $\mathcal{L}_{\text{WTA}}$ assigns the specialization signal to the selected expert, $\mathcal{L}_{\text{CFM}}$ aligns the global transport with the target distribution, and $\mathcal{L}_{\text{BAL}}$ encourages balanced expert usage. During sampling, the selected expert provides a residual correction to the global transport field.

augmenting it with residual dynamical corrections. The method contains two core components: (i) *Mode-Specific Experts*, a bank of Koopman-inspired experts that captures distinct temporal regimes and allows local dynamics to deviate from the global average; and (ii) *Residual Routing*, a Winner-Take-All (WTA) mechanism that assigns each flow state to a dominant expert. By learning residual velocities that complement the global estimator, PrismFlow recovers multi-scale temporal structures while retaining the simulation-free efficiency of standard FM.

In the following, we detail the components of PrismFlow by (**1**) defining Koopman-based linear experts that provide mode-specific residual dynamics (Sec. 4.1); (**2**) introducing the WTA routing policy for hard expert selection (Sec. 4.2); (**3**) describing the multi-objective training that enforces gradient-masked specialization while balancing expert utilization (Sec. 4.3); and (**4**) presenting the sampling procedure where expert-informed residuals complement the global transport estimator to preserve spectral diversity (Sec. 4.4).

4.1 Koopman Experts for Structured Mode-Seeking

To mitigate estimator-level averaging across incompatible temporal regimes, such as oscillations and sharp transients, we introduce a bank of K experts that capture distinct evolution patterns. When multiple temporal modes can explain nearby flow states, different experts should take responsibility for different regimes rather than forcing a single estimator to absorb them through a mean-seeking prediction. To model such structured dynamics efficiently, we adopt Koopman theory [23], which suggests that nonlinear temporal evolution in the original state space can admit an approximately linear description in a latent representation. Formally, for $x^{(s)} \to x^{(s+1)}$, there exist an observable map f and a linear operator A such that $f(x^{(s+1)}) = A \circ f(x^{(s)})$. We therefore learn a latent mapping with a shared encoder and parameterize each expert as a finite-dimensional linear generator.

Latent linear experts. We construct a finite-dimensional Koopman embedding using a shared encoder ϕ_η and a projector ξ. Given a flow state x_t, we compute $h_t = \phi_\eta(x_t)$ and project it into the Koopman latent space as $z_t = \xi(h_t)$. PrismFlow maintains K experts, each parameterized by a learnable matrix $A^k \in \mathbb{R}^{d_z \times d_z}$. Conditioned on expert k, the latent velocity field is:

$$\frac{\mathrm{d}z_t}{\mathrm{d}t} = A^k z_t. \tag{5}$$

Each A^k defines a coherent latent transport mode along the flow path. Its spectrum controls the rotational and contractive components of the latent velocity over flow time, allowing different experts to specialize in distinct local transport regimes while retaining linear latent evolution.

Expert velocity decoding. To correct the data-space flow, we project the latent velocity $A^k z_t$ back to $\mathcal{X}$ using an expert velocity decoder g_ψ:

$$v_t^{\psi,k}(x_t, t) = g_\psi\big(z_t,\, A^k z_t\big)\,. \tag{6}$$

The decoder maps structured linear dynamics in the Koopman space into a nonlinear data-space residual velocity.

Spectral constraints and plasticity. Unconstrained learning of A^k can lead to unstable dynamics and unbounded time-series data. To encourage stable trajectories while preserving expressive temporal structure, we enforce dissipativity by requiring $\frac{1}{2}\big(A^k + (A^k)^\top\big) \preceq -\delta I, \delta \geq 0$. We impose this constraint by parameterizing:

$$A^k = \big(S^k - (S^k)^\top\big) - (R^k)^\top R^k - \delta I, \tag{7}$$

where $S^k \in \mathbb{R}^{d_z \times d_z}$ generates the skew-symmetric component that supports oscillatory modes through latent rotations, and $R^k \in \mathbb{R}^{d_z \times d_z}$ parameterizes the dissipative term that damps transient dynamics. The margin $-\delta I$ ensures spectral dissipativity. This construction keeps S^k and R^k unconstrained during optimization while guaranteeing stable latent dynamics and preserving broad spectral diversity.

4.2 WTA Competitive Routing for Flow Matching

While Koopman experts provide a structured way to model distinct temporal modes, PrismFlow still needs to associate each intermediate flow state with an appropriate expert. In multi-modal regions, several experts may induce plausible local corrections for the same x_t, while soft mixing can average their velocities and reintroduce regression-to-the-mean behavior. We therefore introduce a Winner-Take-All (WTA) routing rule that assigns each flow state to a dominant expert and encourages mode-specific residual dynamics along the trajectory.

Flow routing policy. Along the flow-matching probability path, we use a learnable routing network $\mathcal{R}^\omega$ to assign flow states to experts. Given flow time t and state x_t, the router outputs a categorical distribution over K experts:

$$\pi_t^k = \mathcal{R}^\omega(t, \phi_\eta(x_t)), \quad \text{s.t. } \pi_t^k \geq 0,\ \sum\nolimits_{k=1}^{K} \pi_t^k = 1. \tag{8}$$

Here, π_t^k denotes the probability of selecting expert k at flow time t. The dependence on t allows routing decisions to adapt to different noise-to-data stages of the transport process.

Confidence-aware WTA selection. During training, the target endpoint x_1 provides a natural signal for expert assignment. For each expert k, we estimate a candidate endpoint $\hat{x}_1^k$ using the expert-informed velocity from the current state toward the endpoint. We then define a confidence-aware score:

$$\mathcal{S}_t^k(t, x_t, x_1) = \|\hat{x}_1^k - x_1\|_2^2 - \beta \log\big(\pi_t^k + \varepsilon\big), \tag{9}$$

where β controls the confidence regularization and $\varepsilon > 0$ ensures numerical stability. The winning expert is selected by a hard WTA rule:

$$k_t^* = \arg\min_{k \in \{1,\dots,K\}} \mathcal{S}_t^k(t, x_t, x_1). \tag{10}$$

So, at each flow time t, a single dominant expert provides the primary residual correction for the current state. This hard assignment avoids soft velocity averaging and promotes specialization across temporal regimes. The score in Eq. (9) also stabilizes routing by combining endpoint error with router confidence, reducing noisy early assignments and allowing specialization to develop more smoothly.

4.3 Learning Mode-Specific Dynamical Experts

In this part, we describe the training pipeline of PrismFlow, which learns mode-specific experts to complement the global FM transport in multi-modal regimes. The global transport is supervised by the CFM loss $\mathcal{L}_{\text{CFM}}$ in Eq. (3), which anchors the overall noise-to-data mapping and preserves distributional alignment. To mitigate estimator-level mean seeking and the resulting mode collapse,

Table 1: Results of all methods on all datasets.

Metric	Methods	Sines	Stocks	ETTh	MuJoCo	Energy	fMRI
Context-FID Score ↓	Ours	**0.003±.000**	0.027±.008	**0.015±.001**	**0.006±.005**	**0.022±.002**	0.155±.002
	SMF [44]	0.005±.001	**0.023±.003**	0.059±.007	0.019±.002	0.049±.007	0.116±.006
	Diffusion-TS [36]	0.014±.001	0.208±.052	0.132±.010	0.015±.001	0.092±.020	**0.108±.006**
	DiffTime [7]	0.008±.002	0.236±.074	0.299±.044	0.188±.028	0.289±.045	0.324±.045
	Diffwave [35]	0.014±.002	0.232±.032	0.873±.061	0.393±.041	1.031±.131	0.244±.018
	TimeGAN [12]	0.101±.014	0.105±.025	0.300±.013	0.565±.028	0.742±.133	1.292±.218
	TimeVAE [31]	0.307±.060	0.215±.035	0.805±.186	0.251±.015	1.631±.142	14.449±.969
	Cot-GAN [29]	1.337±.068	0.408±.086	0.980±.071	1.094±.079	1.039±.028	7.813±.550
Correlational Score ↓	Ours	**0.017±.004**	0.007±.003	**0.022±.010**	0.203±.030	**0.495±.033**	0.898±.025
	SMF [44]	0.027±.012	0.010±.007	0.034±.006	0.210±.032	0.900±.277	**0.774±.017**
	Diffusion-TS [36]	0.060±.008	0.013±.009	0.059±.008	**0.188±.035**	0.859±.183	1.416±.028
	DiffTime [7]	**0.017±.004**	**0.006±.002**	0.067±.005	0.238±.031	1.158±.095	1.701±.048
	Diffwave [35]	0.022±.005	0.030±.020	0.175±.006	0.479±.018	5.001±.154	3.927±.049
	TimeGAN [12]	0.045±.010	0.063±.005	0.210±.006	0.886±.035	4.010±.104	23.502±.039
	TimeVAE [31]	0.131±.010	0.095±.008	0.111±.020	0.388±.041	1.728±.226	17.296±.526
	Cot-GAN [29]	0.049±.010	0.087±.004	0.249±.009	1.042±.007	3.164±.061	26.824±.449
Discriminative Score ↓	Ours	**0.004±.003**	0.030±.004	**0.005±.004**	**0.004±.008**	**0.039±.008**	**0.129±.053**
	SMF [44]	0.005±.006	**0.027±.014**	0.017±.009	0.005±.004	0.150±.018	0.136±.139
	Diffusion-TS [36]	0.006±.005	0.086±.024	0.077±.010	0.012±.006	0.139±.012	0.200±.083
	DiffTime [7]	0.013±.006	0.107±.016	0.110±.007	0.154±.045	0.485±.002	0.245±.051
	Diffwave [35]	0.017±.008	0.232±.061	0.190±.008	0.203±.096	0.473±.004	0.402±.029
	TimeGAN [12]	0.011±.008	0.102±.021	0.114±.055	0.238±.068	0.236±.012	0.484±.042
	TimeVAE [31]	0.041±.044	0.145±.120	0.209±.058	0.230±.102	0.499±.000	0.476±.044
	Cot-GAN [29]	0.254±.137	0.230±.016	0.325±.099	0.426±.022	0.498±.002	0.492±.018
	RNN-AR [12]	0.495±.001	0.226±.035	-	-	0.483±.004	-
Predictive Score ↓	Ours	**0.093±.000**	**0.037±.000**	0.122±.003	**0.006±.003**	0.251±.003	**0.100±.000**
	SMF [44]	**0.093±.000**	**0.037±.000**	0.123±.005	0.008±.001	**0.193±.045**	**0.100±.000**
	Diffusion-TS [36]	0.095±.000	**0.037±.000**	**0.118±.008**	0.007±.001	0.251±.000	**0.100±.000**
	DiffTime [7]	**0.093±.000**	0.038±.001	0.121±.004	0.010±.001	0.252±.000	**0.100±.000**
	Diffwave [35]	**0.093±.000**	0.047±.000	0.130±.001	0.013±.000	0.251±.000	0.101±.000
	TimeGAN [12]	**0.093±.019**	0.038±.001	0.124±.001	0.025±.003	0.273±.004	0.126±.002
	TimeVAE [31]	**0.093±.000**	0.039±.000	0.126±.004	0.012±.002	0.292±.000	0.113±.003
	Cot-GAN [29]	0.100±.000	0.047±.001	0.129±.000	0.068±.009	0.259±.000	0.185±.003
	RNN-AR [12]	0.150±.022	0.038±.001	-	-	0.315±.005	-
	Original	0.094±.001	0.036±.001	0.121±.005	0.007±.001	0.250±.003	0.090±.001

we introduce a WTA loss that updates only the selected expert for each (t, x_t). Given the winner k_t^* from Eq. (10), we optimize

$$\mathcal{L}_{\mathrm{WTA}} = \mathbb{E}_{t,x_0,x_1}\left[\lambda_t \, \mathcal{S}_t^{k_t^*}(t, x_t, x_1)\right], \tag{11}$$

where λ_t is a time-dependent weighting schedule along the flow trajectory. Gradients from $\mathcal{L}_{\mathrm{WTA}}$ are explicitly masked for all non-selected experts, so specialization is driven only by the expert that best explains the current sample.

To avoid routing collapse and encourage diverse expert usage, we add a load-balancing regularizer:

$$\bar{\pi} = \frac{1}{B}\sum_{i=1}^{B} \pi_{t^{(i)}}^{(i)}, \qquad \mathcal{L}_{\mathrm{BAL}} = \mathrm{KL}(u \,\|\, \bar{\pi}), \tag{12}$$

where $u = (1/K, \ldots, 1/K)$ is the uniform prior and B is the batch size. Since hard routing can overuse a few experts early in training, $\mathcal{L}_{\mathrm{BAL}}$ penalizes batch-level imbalance, mitigating dead experts and promoting sufficient specialization signals for all experts.

Thus, the overall objective is

$$\mathcal{L}_{\mathrm{total}} = \mathcal{L}_{\mathrm{CFM}}(\theta) + \alpha_W \, \mathcal{L}_{\mathrm{WTA}}(\eta, \psi, \xi, \omega, A) + \alpha_B \, \mathcal{L}_{\mathrm{BAL}}(\omega). \tag{13}$$

Here, $\mathcal{L}_{\mathrm{CFM}}$ learns the shared global transport estimator, $\mathcal{L}_{\mathrm{WTA}}$ promotes mode-specific residual corrections through hard expert assignment, and $\mathcal{L}_{\mathrm{BAL}}$ prevents degenerate routing.

4.4 Sampling with Expert-Informed Residual Dynamics

During inference, PrismFlow follows the globally learned FM transport field while the router selects a dominant expert $k_t^* = \arg\max_k \pi_t^k$ for each state. The selected expert provides a residual correction that steers the trajectory toward mode-consistent local evolution:

$$x_{t+\Delta t} = x_t + \Big(v_t^{\theta}(x_t, t) + \gamma \lambda_t v_t^{\psi, k_t^*}(x_t, t)\Big)\Delta t, \tag{14}$$

This update preserves spectral patterns that tend to be lost under global averaging, yielding samples that are statistically consistent and dynamically expressive.

Table 2: Performance under different training data sizes on the Stocks dataset.

Metric	Methods	20%	40%	60%	80%
Context-FID Score ↓	Ours	**1.920±.574**	**1.890±.251**	**1.789±.437**	**0.824±.434**
	SMF	6.406±.222	3.900±.455	2.613±.567	0.887±.054
	Diffusion-TS	5.197±.583	3.469±.389	2.854±.583	0.860±.204
Discriminative Score ↓	Ours	**0.303±.004**	**0.242±.012**	**0.194±.005**	**0.084±.005**
	SMF	0.347±.008	0.259±.014	0.202±.006	0.097±.006
	Diffusion-TS	0.351±.006	0.267±.010	0.221±.008	0.158±.018

5 Empirical Evaluation

We evaluate PrismFlow on real-world and synthetic datasets for unconditional and conditional generation, measuring sample quality, spectral diversity, and multi-scale temporal preservation. We also run ablations to verify each component and examine how experts partition the data manifold.

Datasets & Metrics. Following [12, 36], we evaluate our model on four real-world benchmarks (Stocks, ETTh, Energy, and fMRI) and two synthetic datasets (Sines and MuJoCo), covering varying degrees of dynamical complexity. For evaluation, we adopt Context-FID [30] and the Correlational Score [47] to measure distributional alignment and assess temporal dependencies. We additionally use the Discriminative Score [12] and Predictive Score [12] to evaluate statistical fidelity.

5.1 Evaluation of Mode Diversity

We first examine mode collapse by comparing PrismFlow with vanilla Flow Matching. Figure 2 illustrates the DMD eigenvalue distributions on the Stocks and fMRI datasets. The imaginary axis reflects oscillation frequency, and a wider spread indicates richer temporal dynamics across slow and fast variations. The real data (blue circles) exhibit broad spectral support and clear geometric structure, suggesting complex multi-scale dynamics rather than a single dominant time scale.

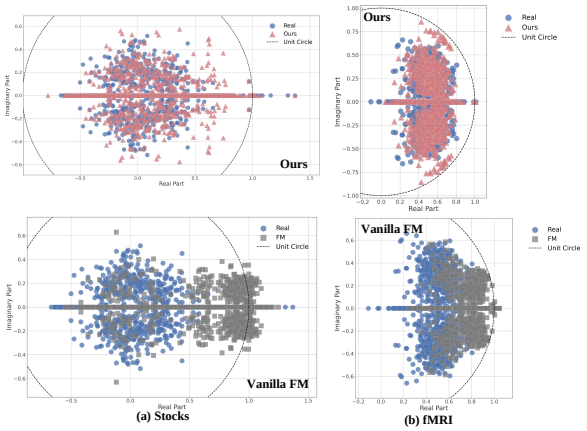


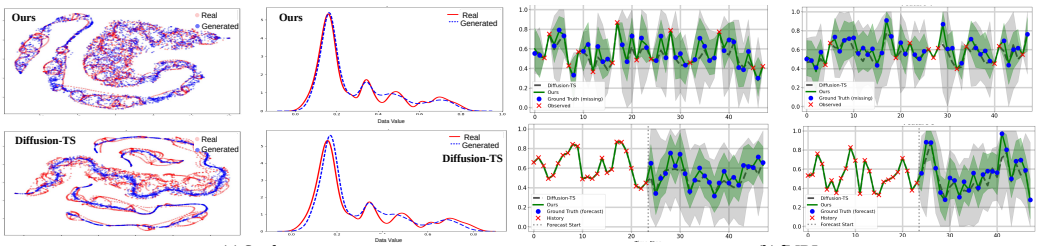
Figure 2: Comparison of DMD eigenvalues between real and generated time series for our PrismFlow and vanilla FM. The imaginary axis represents the oscillation frequency (larger absolute values indicate faster oscillations), while the distance from the origin indicates system stability.

Vanilla FM shows evident spectral contraction: its eigenvalues (gray squares) cluster near the real axis and span a narrower range along the imaginary direction. This pattern indicates that the learned single-field estimator favors a few slow modes, while attenuating high-frequency and transient components after ODE integration. Such contraction is consistent with the estimator-level regression-to-the-mean behavior discussed in Sec. 3, and results in overly smooth generated trajectories. But PrismFlow (pink triangles) achieves much denser spectral overlap with the real data. By using expert-informed residual dynamics to complement the global transport estimator, PrismFlow reduces mean-seeking effects and preserves a wider range of oscillatory modes. Consequently, the generated signals retain broader spectral diversity and multi-scale temporal structure, better matching the dynamical patterns of the empirical distribution.

5.2 Unconditional Time Series Generation

We evaluate PrismFlow on unconditional time-series generation under comparable parameter budgets for all baselines. The compared methods include generative adversarial networks (GANs) [12, 29], variational autoencoders (VAEs) [31], diffusion-based models [35, 36, 7], and FM-based models [44].

Table 1 reports generation performance across diverse datasets. PrismFlow achieves the best results in 16 out of 24 cases, outperforming recent baselines such as SMF and Diffusion-TS. These results are consistent with the spectral analysis above: by using routed residual experts to complement



(a) Stocks (b) fMRI

Figure 3: Comparison of unconditional time-series generation on the Stocks dataset (left) and imputation (top row) and forecasting (bottom row) on the fMRI dataset (right) between our PrismFlow and Diffusion-TS.

the global estimator, PrismFlow alleviates estimator-level mean seeking and better preserves multi-scale temporal modes. The benefit is especially clear on complex datasets such as Energy, where PrismFlow reduces Context-FID by over 55% and Discriminative Score by about 74% compared with the second-best method. On the high-dimensional fMRI dataset, PrismFlow also achieves the lowest Discriminative Score, indicating that it better preserves rhythmic and structured dynamics that are often attenuated by over-smoothed learned flows. Overall, the results show that PrismFlow improves both statistical fidelity and temporal realism by maintaining a broader spectral range and capturing multi-scale structures in time-series data.

We further visualize generation quality using t-SNE projections and kernel density estimation (KDE). As shown in Figure 3(a), samples generated by PrismFlow overlap closely with real data in both views. In the t-SNE plot, generated trajectories follow the global structure of real samples and cover both central regions and more complex outer areas. In contrast, Diffusion-TS leaves visible gaps in the sample space, suggesting partial mode dropping. The KDE plots show a similar trend: PrismFlow closely matches the real density curves and captures the multiple peaks and sharp variations in the Stocks dataset. Appendix D.3 further shows that PrismFlow achieves this performance with high sampling efficiency.

Small-Scale Settings. We also evaluate PrismFlow under severe data scarcity by training on reduced subsets of the training data. As shown in Tab. 2, PrismFlow remains stable across all low-data settings and achieves the best Context-FID and Discriminative Scores among competing baselines. With only 20% of the training data, PrismFlow still yields clear gains, improving Context-FID by 63.1% and Discriminative Score by 12.7%. We attribute this robustness to expert-specific residual modeling: by assigning distinct temporal regimes to specialized experts, PrismFlow preserves subtle temporal variations that are easily averaged out in low-data regimes. As a result, it maintains fine-grained dynamics and high generation quality even with substantially fewer training examples.

5.3 Conditional Time Series Generation

Beyond unconditional generation, we extend PrismFlow to conditional tasks, including forecasting and imputation, using flow-matching guidance [48]. Specifically, we add a guidance gradient that enforces consistency with observed values and use it as an additional correction to the sampling velocity. As shown in Figure 3(b), PrismFlow better follows sharp transitions and sudden spikes where Diffusion-TS often falls short. This advantage comes from the Koopman-inspired experts, which provide structured temporal inductive biases, and the WTA routing mechanism, which promotes specialization over fine-grained temporal modes. As a result, the model can recover more faithful trajectories from limited local observations.

5.4 Ablation Study

Table 3 further shows the contribution of each component. PrismFlow consistently outperforms vanilla FM, confirming the benefit of augmenting the global transport estimator with mode-specific residual dynamics. Among all components, removing the WTA objective $\mathcal{L}_{\mathrm{WTA}}$ causes the largest performance drop, especially on the high-dimensional fMRI dataset. Without competitive specialization, experts are less clearly separated across temporal regimes, and the learned dynamics become more mean-seeking, producing smoother trajectories that miss fine-grained structures.

The load-balancing term $\mathcal{L}_{\mathrm{BAL}}$ and confidence weight β further improve routing behavior by preventing a small subset of experts from dominating. This is particularly useful for datasets such as Stocks, where temporal patterns are noisy and shift over time. By encouraging broader expert utilization, these components help retain both frequent and less common temporal modes. Finally, performance degrades when $\gamma = 0$, indicating that expert-informed residual dynamics during sampling are essential for generation quality. Without this residual correction, the sampler relies mainly on the global estimator and becomes less adaptive to diverse temporal regimes, losing important local variations. This supports our design principle: combining stable global transport with competitively routed residual experts mitigates estimator-level averaging and better captures complex temporal structure.

Table 3: Ablation study on the model design.

Metric	Variant	Stocks	fMRI
Context-FID Score ↓	Ours	**0.027±.008**	**0.155±.002**
	Vanilla FM	0.116±.012	0.324±.030
	Training w/o $\mathcal{L}_{\mathrm{WTA}}$	0.107±.011	0.294±.018
	Training w/o $\mathcal{L}_{\mathrm{BAL}}$	0.088±.015	0.178±.007
	Training $\beta = 0$	0.095±.010	0.248±.019
	Sampling $\gamma = 0$	0.091±.010	0.185±.014
Discriminative Score ↓	Ours	**0.030±.004**	**0.129±.053**
	Vanilla FM	0.186±.005	0.245±.050
	Training w/o $\mathcal{L}_{\mathrm{WTA}}$	0.112±.014	0.402±.029
	Training w/o $\mathcal{L}_{\mathrm{BAL}}$	0.106±.045	0.218±.020
	Training $\beta = 0$	0.102±.074	0.229±.058
	Sampling $\gamma = 0$	0.075±.012	0.275±.029

5.5 Analysis of Mode-Specific Experts

To better understand how PrismFlow reproduces the temporal modes in Figure 2, we examine the DMD eigenvalue spectra of the trained fMRI experts in Figure 4(a). The expert spectra show a clear division of labor that mirrors the main structures of the real spectrum. Expert 1 mainly produces eigenvalues close to the real axis, corresponding to non-oscillatory components such as slow drifts and smooth trend-like dynamics. These modes match the band near the real axis in Figure 2. In contrast, Expert 2 focuses on oscillatory components with larger imaginary parts, forming conjugate clusters above and below the real axis. This structure matches the symmetric clouds in the real DMD spectrum and reflects rhythmic fluctuations in fMRI signals. These results explain why PrismFlow improves spectral fidelity: rather than forcing a single learned estimator to average incompatible dynamics, the WTA routing mechanism assigns different temporal regimes to specialized experts. As a result, the generated sequences preserve diverse dynamics and yield spectra that better match the empirical distribution.

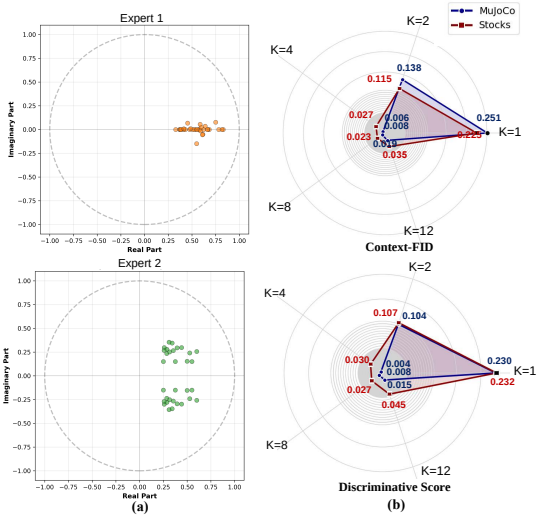


Figure 4: (a) DMD eigenvalues for Expert 1 and Expert 2 on fMRI. (b) Effect of the number of experts on the unconditional task for the MuJoCo and Stocks datasets. Lower is better.

We further analyze the effect of the number of experts K in Figure 4(b). Across datasets, $K = 1$ performs the worst, confirming that a single expert lacks sufficient capacity to capture heterogeneous temporal structures. Using $K = 4$ provides consistently strong performance and serves as our default setting, while the optimal capacity still depends on data complexity. For Stocks, increasing K to 8 further improves performance, suggesting that additional experts help model its diverse and shifting temporal patterns. For MuJoCo, performance improves with a moderate number of experts but slightly drops when too many are used, likely because the expert pool exceeds the intrinsic complexity of the data and weakens clear specialization.

6 Conclusion

In this paper, we introduce PrismFlow, a new FM method with residual dynamical experts for time-series generation. The method is motivated by a key limitation of learned single-field FM models: when time-series data contain multiple competing temporal regimes, a single learned transport field may suffer from estimator-level averaging, which smooths distinct dynamics and can lead to mode collapse. PrismFlow addresses this problem by decomposing the transport process into a stable global backbone and hard-routed Koopman-inspired residual experts. The global field captures shared temporal evolution, while the experts model regime-specific residual dynamics, such as oscillatory and high-frequency patterns. Empirical results on various generation tasks show that PrismFlow achieves strong performance and effectively mitigates mode collapse by preserving diverse spectral patterns, unlocking a new paradigm for robust generative modeling.

Acknowledgements

This project is supported by AIMfg, A*STAR, Singapore, under its MTI WS Fund (W24MCMF012), and by the Ministry of Education, Singapore, under its Academic Research Fund Tier 1 (RG101/24).

References

[1] Shruti Kaushik, Abhinav Choudhury, Pankaj Kumar Sheron, Nataraj Dasgupta, Sayee Natarajan, Larry A Pickett, and Varun Dutt. Ai in healthcare: time-series forecasting using statistical, neural, and ensemble architectures. *Frontiers in big data*, 3:4, 2020.

[2] Junru Zhang, Lang Feng, Xu Guo, Han Yu, Yabo Dong, and Duanqing Xu. Diffusion-guided diversity for single domain generalization in time series classification. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*, pages 3764–3773, 2025.

[3] Hongbin Huang, Minghua Chen, and Xiao Qiao. Generative learning for financial time series with irregular and scale-invariant patterns. In *The Twelfth International Conference on Learning Representations*, 2024.

[4] Fatemeh Chitsaz and Saman Haratizadeh. Dual adaptation of time-series foundation models for financial forecasting. In *1st ICML Workshop on Foundation Models for Structured Data*.

[5] Zhen Liu, Wenbin Pei, Disen Lan, and Qianli Ma. Diffusion language-shapelets for semi-supervised time-series classification. In *Proceedings of the AAAI conference on artificial intelligence*, volume 38, pages 14079–14087, 2024.

[6] Yang He, Yuhan Wu, Junru Zhang, and Yabo Dong. Ltcr: Long temporal characteristic reconstruction for segmentation in contrastive learning. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 355–371. Springer, 2024.

[7] Andrea Coletta, Sriram Gopalakrishnan, Daniel Borrajo, and Svitlana Vyetrenko. On the constrained time-series generation problem. *Advances in Neural Information Processing Systems*, 36:61048–61059, 2023.

[8] Yuhan Wu, Xiyu Meng, Junru Zhang, Yang He, Joseph A Romo, Yabo Dong, and Dongming Lu. Effective lstms with seasonal-trend decomposition and adaptive learning and niching-based backtracking search algorithm for time series forecasting. *Expert Systems with Applications*, 236:121202, 2024.

[9] Oluwatoyin Ajoke Farayola, Oluwabukunmi Latifat Olorunfemi, and Philip Olaseni Shoetan. Data privacy and security in it: a review of techniques and challenges. *Computer Science & IT Research Journal*, 5(3):606–615, 2024.

[10] Tal Gonen, Itai Pemper, Ilan Naiman, Nimrod Berman, and Omri Azencot. Time series generation under data scarcity: A unified generative modeling approach. *arXiv preprint arXiv:2505.20446*, 2025.

[11] Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.

[12] Jinsung Yoon, Daniel Jarrett, and Mihaela Van der Schaar. Time-series generative adversarial networks. *Advances in neural information processing systems*, 32, 2019.

[13] Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.

[14] Michael S Albergo, Nicholas M Boffi, and Eric Vanden-Eijnden. Stochastic interpolants: A unifying framework for flows and diffusions. *arXiv preprint arXiv:2303.08797*, 2023.

[15] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

[16] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.

[17] Johannes Schusterbauer, Ming Gui, Frank Fundel, and Björn Ommer. Diff2flow: Training flow matching models via diffusion model alignment. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 28347–28357, 2025.

[18] Sanjeev Arora, Rong Ge, Yingyu Liang, Tengyu Ma, and Yi Zhang. Generalization and equilibrium in generative adversarial nets (gans). In *International conference on machine learning*, pages 224–232. PMLR, 2017.

[19] Duhyeon Bang and Hyunjung Shim. Improved training of generative adversarial networks using representative features. In *International conference on machine learning*, pages 433–442. PMLR, 2018.

[20] Yassine Abbahaddou and Amine M Aboussalah. A geometry-aware metric for mode collapse in time series generative models. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL `https://openreview.net/forum?id=YAcOO13qMc`.

[21] Aksel Wilhelm Wold Eide, Eilif Solberg, and Ingebjørg Kåsen. Sample weighting as an explanation for mode collapse in generative adversarial networks. *arXiv preprint arXiv:2010.02035*, 2020.

[22] Ziqi Pan, Li Niu, and Liqing Zhang. Unigan: Reducing mode collapse in gans using a uniform generator. *Advances in neural information processing systems*, 35:37690–37703, 2022.

[23] Bernard O Koopman. Hamiltonian systems and transformation in hilbert space. *Proceedings of the National Academy of Sciences*, 17(5):315–318, 1931.

[24] Yueheng Lan and Igor Mezić. Linearization in the large of nonlinear systems and koopman operator spectrum. *Physica D: Nonlinear Phenomena*, 242(1):42–53, 2013.

[25] Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *arXiv preprint arXiv:1701.06538*, 2017.

[26] Robert A Jacobs, Michael I Jordan, Steven J Nowlan, and Geoffrey E Hinton. Adaptive mixtures of local experts. *Neural computation*, 3(1):79–87, 1991.

[27] Olof Mogren. C-rnn-gan: Continuous recurrent neural networks with adversarial training. *arXiv preprint arXiv:1611.09904*, 2016.

[28] Cristóbal Esteban, Stephanie L Hyland, and Gunnar Rätsch. Real-valued (medical) time series generation with recurrent conditional gans. *arXiv preprint arXiv:1706.02633*, 2017.

[29] Tianlin Xu, Li Kevin Wenliang, Michael Munn, and Beatrice Acciaio. Cot-gan: Generating sequential data via causal optimal transport. *Advances in neural information processing systems*, 33:8798–8809, 2020.

[30] Paul Jeha, Michael Bohlke-Schneider, Pedro Mercado, Shubham Kapoor, Rajbir Singh Nirwan, Valentin Flunkert, Jan Gasthaus, and Tim Januschowski. Psa-gan: Progressive self attention gans for synthetic time series. In *The tenth international conference on learning representations*, 2022.

[31] Abhyuday Desai, Cynthia Freeman, Zuhui Wang, and Ian Beaver. Timevae: A variational auto-encoder for multivariate time series generation. *arXiv preprint arXiv:2111.08095*, 2021.

[32] Hangwei Qian, Sinno Jialin Pan, and Chunyan Miao. Latent independent excitation for generalizable sensor-based cross-person activity recognition. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pages 11921–11929, 2021.

[33] Junru Zhang, Lang Feng, Zhidan Liu, Yuhan Wu, Yang He, Yabo Dong, and Duanqing Xu. Diverse intra-and inter-domain activity style fusion for cross-person generalization in activity recognition. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 4213–4222, 2024.

[34] Yiyuan Yang, Ming Jin, Haomin Wen, Chaoli Zhang, Yuxuan Liang, Lintao Ma, Yi Wang, Chenghao Liu, Bin Yang, Zenglin Xu, et al. A survey on diffusion models for time series and spatio-temporal data. *ACM Computing Surveys*, 2024.

[35] Zhifeng Kong, Wei Ping, Jiaji Huang, Kexin Zhao, and Bryan Catanzaro. Diffwave: A versatile diffusion model for audio synthesis. *arXiv preprint arXiv:2009.09761*, 2020.

[36] Xinyu Yuan and Yan Qiao. Diffusion-ts: Interpretable diffusion for general time series generation. *arXiv preprint arXiv:2403.01742*, 2024.

[37] Zhicheng Chen, FENG SHIBO, Zhong Zhang, Xi Xiao, Xingyu Gao, and Peilin Zhao. Sdformer: Similarity-driven discrete transformer for time series generation. *Advances in Neural Information Processing Systems*, 37:132179–132207, 2024.

[38] Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. In *Forty-first international conference on machine learning*, 2024.

[39] Shoufa Chen, Chongjian Ge, Yuqi Zhang, Yida Zhang, Fengda Zhu, Hao Yang, Hongxiang Hao, Hui Wu, Zhichao Lai, Yifei Hu, et al. Goku: Flow based video generative foundation models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 23516–23527, 2025.

[40] Ella Tamir, Najwa Laabid, Markus Heinonen, Vikas Garg, and Arno Solin. Conditional flow matching for time series modelling. In *ICML 2024 Workshop on Structured Probabilistic Inference {\&} Generative Modeling*, 2024.

[41] Yunfeng Ge, Jiawei Li, Yiji Zhao, Haomin Wen, Zhao Li, Meikang Qiu, Hongyan Li, Ming Jin, and Shirui Pan. T2s: High-resolution time series generation with text-to-series diffusion models. *arXiv preprint arXiv:2505.02417*, 2025.

[42] Yong Liu, Guo Qin, Zhiyuan Shi, Zhi Chen, Caiyin Yang, Xiangdong Huang, Jianmin Wang, and Mingsheng Long. Sundial: A family of highly capable time series foundation models. *arXiv preprint arXiv:2502.00816*, 2025.

[43] Marcel Kollovieh, Marten Lienen, David Lüdke, Leo Schwinn, and Stephan Günnemann. Flow matching with gaussian process priors for probabilistic time series forecasting. *arXiv preprint arXiv:2410.03024*, 2024.

[44] Jinwoo Kim, Max Beier, Petar Bevanda, Nayun Kim, and Seunghoon Hong. Sequence modeling with spectral mean flows. *arXiv preprint arXiv:2510.15366*, 2025.

[45] Adrien Cortés, Rémi Rehm, and Victor Letzelter. Winner-takes-all for multivariate probabilistic time series forecasting. *arXiv preprint arXiv:2506.05515*, 2025.

[46] Peter J Schmid. Dynamic mode decomposition of numerical and experimental data. *Journal of fluid mechanics*, 656:5–28, 2010.

[47] Shujian Liao, Hao Ni, Lukasz Szpruch, Magnus Wiese, Marc Sabate-Vidales, and Baoren Xiao. Conditional sig-wasserstein gans for time series generation. *arXiv preprint arXiv:2006.05421*, 2020.

[48] Ruiqi Feng, Chenglei Yu, Wenhao Deng, Peiyan Hu, and Tailin Wu. On the guidance of flow matching. *arXiv preprint arXiv:2502.02150*, 2025.

[49] Yifan Chen, Mark Goldstein, Mengjian Hua, Michael S Albergo, Nicholas M Boffi, and Eric Vanden-Eijnden. Probabilistic forecasting with stochastic interpolants and f\" ollmer processes. *arXiv preprint arXiv:2403.13724*, 2024.

[50] Yusuke Tashiro, Jiaming Song, Yang Song, and Stefano Ermon. Csdi: Conditional score-based diffusion models for probabilistic time series imputation. *Advances in neural information processing systems*, 34:24804–24816, 2021.

[51] Zhihan Yue, Yujing Wang, Juanyong Duan, Tianmeng Yang, Congrui Huang, Yunhai Tong, and Bixiong Xu. Ts2vec: Towards universal representation of time series. In *Proceedings of the AAAI conference on artificial intelligence*, volume 36, pages 8980–8987, 2022.

A Limitation

Despite these strengths, PrismFlow has several limitations that motivate future work. First, the current model uses a fixed expert capacity. Although this design is simple, stable, and effective in our experiments, different datasets may benefit from different degrees of expert specialization. Future work will explore adaptive expert allocation, allowing the model to dynamically adjust expert capacity according to data complexity. Moreover, our current experiments are conducted in a dataset-specific training setting, which allows us to carefully evaluate the modeling ability of PrismFlow on individual benchmarks. A promising future direction is to extend PrismFlow to cross-dataset learning, where a shared global transport backbone captures common temporal dynamics across datasets, while dynamically activated domain-specific experts model dataset-specific patterns. Such an extension could further improve scalability and generality, and the learned routing behavior may also provide interpretable insights into how different datasets rely on different temporal dynamics or expert components.

B Broader Impacts

The societal impact of our work lies primarily in its potential to broaden the applicability of machine learning models in real-world time-series scenarios. Time-series data are widely used in critical sectors such as transportation, energy, finance, healthcare, and industrial systems. By improving time-series generation, imputation, and forecasting, our method can support data augmentation, simulation, model evaluation, and intelligent decision-making, especially when real data are scarce, costly, or difficult to collect. At the same time, we acknowledge that synthetic time series should be used with care. Generated data may inherit biases or distributional limitations from the training data, and inappropriate use in high-stakes domains could lead to unreliable decisions. Practical applications should involve domain-specific validation, privacy checks, and human expert oversight.

C Pseudocode

We present the complete training and sampling procedures of PrismFlow in Algorithms 1 and 2. Algorithm 1 jointly learns a global Flow Matching backbone and a set of Koopman-inspired residual experts. The global backbone is supervised by the Conditional Flow Matching objective, while the residual experts are trained through a confidence-aware Winner-Take-All (WTA) objective that assigns each intermediate flow state to a single expert and masks gradients to the remaining experts. A load-balancing regularizer is further used to prevent degenerate routing and to encourage diverse expert utilization. Algorithm 2 describes the generation process, where the router selects the dominant expert at each integration step and the selected expert provides a residual dynamical correction to the global transport field.

D Additional Experimental Results

D.1 Analytic Gaussian-Mixture Diagnostic

To separate estimator difficulty from the validity of the underlying transport, we add a controlled Gaussian-mixture experiment where the probability path and conditional velocity are analytically available. This diagnostic is not intended to claim that the true FM vector field collapses. Instead, it tests whether a finite learned single-field estimator trained with an ℓ_2 objective loses branch-specific velocity information in regions where different mixture components overlap. The target distribution is a four-component Gaussian mixture whose components induce distinct oscillatory time-series prototypes. We compare vanilla FM and PrismFlow using the same backbone width and training budget.

Table 4 shows that the exact transport problem is not ill-posed, but the learned vanilla FM estimator still loses modal structure near branch-overlap regions. PrismFlow reduces this gap because WTA experts regress on routed subsets of conditional targets rather than forcing all targets through one conditional mean. This directly addresses the distinction between the true velocity field v_t and the learned estimator $\hat{v}_t$.

Algorithm 1 Training Procedure of PrismFlow

Require: Dataset $\mathcal{D}$, number of experts K, max iterations N, batch size B
Require: Hyperparameters $\alpha_W, \alpha_B, \beta, \lambda_t$, stability margin δ
Initialize global FM parameters θ, projector ξ, expert decoder ψ, router ω, and expert parameters $\{S^k, R^k\}_{k=1}^K$
Let $\Theta \leftarrow \{\theta, \xi, \psi, \omega, \{S^k, R^k\}_{k=1}^K\}$
for $n = 1$ to N **do**
  Sample batch $x_1 \sim \mathcal{D}$, noise $x_0 \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, and time $t \sim \mathcal{U}[0, 1]$
  // Flow-matching interpolation
  $x_t \leftarrow (1-t)x_0 + tx_1, \quad u_t \leftarrow x_1 - x_0$
  // Shared representation and router
  $h_t \leftarrow \phi_\eta(x_t), \quad z_t \leftarrow \xi(h_t), \quad \pi_t \leftarrow \mathcal{R}_\omega(t, h_t)$
  // Global transport field
  $v_t^{\text{global}} \leftarrow v_t^\theta(x_t, t)$
  // Koopman residual experts
  for $k = 1$ to K **do**
    $A^k \leftarrow (S^k - S^{k\top}) - R^{k\top} R^k - \delta I$
    $r_t^k \leftarrow v_t^{\psi,k}(x_t, t) \leftarrow g_\psi(z_t, A^k z_t)$
    $\tilde{u}_t^k \leftarrow \text{sg}[v_t^{\text{global}}] + r_t^k$
    $\hat{x}_1^k \leftarrow x_t + (1-t)\tilde{u}_t^k$
    $\mathcal{S}_t^k \leftarrow \|\hat{x}_1^k - x_1\|_2^2 - \beta \log(\pi_t^k + \varepsilon)$
  end for
  // Confidence-aware WTA selection
  $k_t^* \leftarrow \arg\min_{k \in \{1,\dots,K\}} \text{sg}[\mathcal{S}_t^k]$
  $m_t^k \leftarrow \mathbb{1}[k = k_t^*]$
  // Loss computation
  $\mathcal{L}_{\text{CFM}} \leftarrow \frac{1}{B} \sum_{i=1}^B \|v_t^{\text{global},(i)} - u_t^{(i)}\|_2^2$
  $\mathcal{L}_{\text{WTA}} \leftarrow \frac{1}{B} \sum_{i=1}^B \lambda_t^{(i)} \sum_{k=1}^K m_t^{k,(i)} \mathcal{S}_t^{k,(i)}$
  $\bar{\pi} \leftarrow \frac{1}{B} \sum_{i=1}^B \pi_t^{(i)}$
  $\mathcal{L}_{\text{BAL}} \leftarrow \text{KL}(\mathcal{U}_K \,\|\, \bar{\pi})$
  $\mathcal{L}_{\text{total}} \leftarrow \mathcal{L}_{\text{CFM}} + \alpha_W \mathcal{L}_{\text{WTA}} + \alpha_B \mathcal{L}_{\text{BAL}}$
  Update $\Theta \leftarrow \Theta - \eta_{\text{lr}} \nabla_\Theta \mathcal{L}_{\text{total}}$
end for
Return: Learned parameters Θ

Table 4: Analytic Gaussian-mixture diagnostic. The transport field is well-defined; the gap measures learned-estimator behavior under overlapping conditional velocities.

Method	DMD-GEN [20] ↓	Context-FID ↓	Predictive Score ↓
Vanilla FM	0.58	0.76	0.132
PrismFlow	**0.14**	**0.22**	**0.041**

D.2 Comparison with a Diversity-Promoting Interpolant Baseline

We further compare PrismFlow with a stochastic-interpolant forecasting method [49], which is designed to encourage sample diversity in probabilistic forecasting. Specifically, we adapt this baseline to the Stocks dataset (20% setting) and evaluate it under identical experimental conditions.

Table 5: Comparison with a diversity-promoting stochastic-interpolant baseline on the 20% Stocks low-data setting.

Method	Context-FID ↓	Discriminative Score ↓
Diffusion-TS	5.197	0.351
Adapted stochastic interpolant	2.451	0.335
PrismFlow	**1.920**	**0.303**

As shown in Table 5, the adapted stochastic-interpolant baseline [49] improves over Diffusion-TS in this low-data setting, while PrismFlow achieves the best results on both metrics. These results indicate that the gains of PrismFlow are not solely due to increased output-level diversity. Instead, the

Algorithm 2 Sampling Procedure of PrismFlow

Require: Learned parameters Θ, integration steps N, residual scale γ, schedule λ_t
Sample initial noise $x_0 \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
$\Delta t \leftarrow 1/N, \quad x_{\text{curr}} \leftarrow x_0$
for $i = 0$ to $N - 1$ **do**
  $t \leftarrow i/N$
  // Router-guided expert selection
  $h_t \leftarrow \phi_\eta(x_{\text{curr}}), \quad z_t \leftarrow \xi(h_t), \quad \pi_t \leftarrow \mathcal{R}_\omega(t, h_t)$
  $k_t^* \leftarrow \arg\max_{k \in \{1,\dots,K\}} \pi_t^k$
  // Selected Koopman residual expert
  $A^{k_t^*} \leftarrow (S^{k_t^*} - S^{k_t^*\top}) - R^{k_t^*\top} R^{k_t^*} - \delta I$
  $r_t^{k_t^*} \leftarrow v_t^{\psi,k_t^*}(x_{\text{curr}}, t) \leftarrow g_\psi(z_t, A^{k_t^*} z_t)$
  // Global transport plus residual correction
  $v_{\text{curr}} \leftarrow v_t^\theta(x_{\text{curr}}, t) + \gamma \lambda_t r_t^{k_t^*}$
  // Numerical integration
  $x_{\text{curr}} \leftarrow x_{\text{curr}} + v_{\text{curr}} \Delta t$
end for
Return: $\hat{x} \leftarrow x_{\text{curr}}$

routed residual dynamics help reduce estimator-level averaging within the learned flow field, leading to improved fidelity and discriminability.

D.3 Comparison of Training and Inference Time

Table 6: Training and sampling time comparison (seconds)

Training Time	Sines	Stocks	ETTh	MuJoCo	Energy	fMRI
SMF	431.85	216.94	690.31	2242.07	1001.37	1247.64
Diffusion-TS	199.66	177.88	399.81	283.41	649.51	489.44
Ours	1732.47	835.96	3561.08	2842.96	4923.66	3080.99
Sampling Time	**Sines**	**Stocks**	**ETTh**	**MuJoCo**	**Energy**	**fMRI**
SMF	203.51	76.59	346.29	787.09	1365.49	764.80
Diffusion-TS	437.96	167.42	821.38	1650.49	3262.70	1879.91
Ours	**2.49**	**6.12**	**18.36**	**85.75**	**49.87**	**25.62**

All experiments were conducted on a single NVIDIA GeForce RTX 4090 (24GB). We report wall-clock training and sampling times using default settings and the same data size for all models. The results are shown in Table 6.

PrismFlow takes longer to train than the baseline methods. This is expected, since our model uses a more expressive architecture with specialized dynamical experts. The model spends more computation during training to better learn the structure of the vector field. This design leads to a clear advantage during inference. Unlike diffusion models that require hundreds of steps, PrismFlow maintains the efficiency of flow matching while generating high-quality samples with a modest number of sampling steps. As a result, our method is much faster than both SMF and Diffusion-TS during sampling on all datasets. The advantage is especially clear on high-dimensional datasets such as fMRI, where our sampling time is often more than ten times faster. By doing so, it becomes a practical, off-the-shelf solution for real-world applications that require fast generation.

D.4 Comparison of Model Size

Tab. 7 presents a comparison of parameter counts across different models and datasets. PrismFlow maintains a comparable model size to existing baselines, with its parameter count varying according to dataset characteristics and model configurations. These results suggest that PrismFlow offers a flexible and scalable architecture, providing sufficient modeling capacity across diverse time-series generation tasks.

Table 7: Parameter count comparison across models and datasets

Dataset	PrismFlow	SMF	Diffusion-TS
Sines	330,709	356,736	235,249
Stocks	363,350	356,800	294,390
ETTh	392,775	356,864	364,043
MuJoCo	931,022	1,233,696	360,286
Energy	1,416,364	1,086,432	1,155,520
fMRI	1,480,274	1,237,152	1,425,586

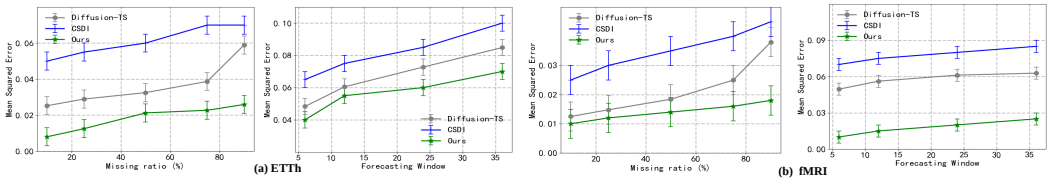


Figure 5: Comparison between our PrismFlow and diffusion-based methods for time-series imputation and forecasting.

D.5 Small-Scale Settings Evaluation

Table 8 shows that PrismFlow remains strong even with very limited training data. Under most low-data settings, it consistently outperforms the baselines while maintaining high generation quality, indicating that the model does not rely heavily on large-scale supervision to learn meaningful dynamics. This robustness stems from its expert-based design. When data is scarce, monolithic models often produce over-smoothed results because they struggle to represent diverse temporal patterns with limited examples. In contrast, PrismFlow assigns different temporal dynamics to specialized experts, which reduces interference during training and helps preserve fine-grained temporal structures. Each expert can focus on a subset of behaviors, allowing the overall model to retain variability that would otherwise be averaged out. As a result, PrismFlow captures the underlying dynamics more effectively and degrades more gracefully as the data size decreases. This makes it particularly suitable for real-world scenarios where collecting large, well-annotated time-series datasets is expensive or impractical.

D.6 Conditional Generation Tasks

We evaluate PrismFlow on two conditional time-series tasks: imputation and forecasting. Although SMF is also based on flow matching, it is designed specifically for unconditional settings and does not include the guidance mechanisms required for conditional generation. Therefore, we compare PrismFlow with competitive conditional approaches, including Diffusion-TS and CSDI [50]. As shown in Figure 5, PrismFlow consistently achieves lower Mean Squared Error (MSE) than diffusion-based baselines such as Diffusion-TS and CSDI. This demonstrates the effectiveness of our model in capturing complex temporal dynamics under conditional settings.

E Ablation Study

More ablation results in Table 9 show that both the expert design and the routing strategy matter. Using generic MLP or CNN experts instead of Koopman-based experts reduces performance, since these models lack an explicit bias toward structured temporal behavior. Moreover, replacing WTA routing with soft gating worsens results because soft gating blends multiple expert velocities and reintroduces the averaging effect that PrismFlow is designed to avoid. Random routing performs even worse, confirming that proper expert selection is necessary for learning mode-specific corrections. These results also clarify why hard routing remains stable in practice. The confidence-aware WTA score in Eq. (9) prevents early assignments from depending only on noisy endpoint errors, while $\mathcal{L}_{\mathrm{BAL}}$ discourages batch-level routing collapse. At the same time, the global CFM objective remains active for every sample, so the backbone continues to provide a stable coarse transport path even before

Table 8: Performance comparison under different data sizes.

Dataset	Metrics	Method	0.2	0.4	0.6	0.8
Stocks	Context-FID Score ↓	Ours	**1.920±.574**	**1.890±.251**	**1.789±.437**	**0.824±.434**
		Diffusion-TS	5.197±.583	3.469±.389	2.854±.583	0.860±.204
		SMF	6.406±.222	3.900±.455	2.613±.567	0.887±.054
	Correlational Score ↓	Ours	0.495±.008	0.207±.006	0.045±.016	0.035±.009
		Diffusion-TS	**0.127±.009**	**0.057±.010**	**0.013±.009**	0.006±.005
		SMF	0.139±.003	0.713±.008	0.008±.007	**0.005±.004**
	Discriminative Score ↓	Ours	**0.303±.004**	**0.242±.012**	**0.194±.005**	**0.084±.005**
		Diffusion-TS	0.351±.006	0.267±.010	0.221±.008	0.158±.018
		SMF	0.347±.008	0.259±.014	0.202±.006	0.097±.006
	Predictive Score ↓	Ours	**0.049±.032**	0.127±.004	**0.037±.001**	**0.037±.002**
		Diffusion-TS	0.243±.006	**0.039±.001**	0.039±.002	0.038±.000
		SMF	0.243±.005	0.042±.004	0.073±.000	0.073±.000
ETTh	Context-FID Score ↓	Ours	6.684±.912	**1.457±.112**	1.077±.112	0.332±.018
		Diffusion-TS	**0.459±.080**	1.549±.270	1.044±.030	0.316±.044
		SMF	6.844±.470	1.556±.141	**0.947±.101**	**0.237±.034**
	Correlational Score ↓	Ours	**0.607±.009**	**0.311±.011**	**0.172±.007**	**0.072±.011**
		Diffusion-TS	2.846±.165	0.344±.010	0.187±.008	0.088±.005
		SMF	0.629±.007	0.391±.016	0.180±.014	0.068±.010
	Discriminative Score ↓	Ours	0.388±.008	**0.220±.012**	**0.162±.010**	0.091±.005
		Diffusion-TS	**0.155±.006**	0.234±.005	0.173±.004	0.111±.005
		SMF	0.377±.009	0.227±.005	**0.162±.004**	**0.080±.010**
	Predictive Score ↓	Ours	0.295±.027	**0.157±.013**	**0.152±.003**	**0.133±.005**
		Diffusion-TS	**0.252±.001**	0.166±.009	0.156±.004	0.127±.005
		SMF	0.279±.170	0.164±.004	0.149±.006	**0.133±.004**
Energy	Context-FID Score ↓	Ours	**5.701**±.916	**3.464**±.269	1.886±.181	**0.439**±.117
		Diffusion-TS	6.667±.715	3.554±.523	1.393±.138	0.459±.080
		SMF	6.633±.477	3.893±.418	**1.255±.288**	0.446±.090
	Correlational Score ↓	Ours	**6.841±.162**	**5.969±.132**	**4.201±.156**	**2.722±.165**
		Diffusion-TS	7.002±.101	6.414±.183	4.543±.117	3.144±.159
		SMF	7.097±.106	6.004±.048	4.288±.241	2.846±.173
	Discriminative Score ↓	Ours	**0.398±.001**	**0.291±.003**	**0.201±.009**	**0.121±.006**
		Diffusion-TS	0.499±.001	0.498±.001	0.497±.003	0.498±.003
		SMF	0.406±.005	0.299±.009	0.208±.007	0.155±.013
	Predictive Score ↓	Ours	0.279±.011	0.262±.009	**0.251±.001**	**0.251±.001**
		Diffusion-TS	0.263±.003	0.264±.001	0.264±.000	0.265±.001
		SMF	**0.253±.001**	**0.251±.000**	**0.251±.000**	**0.251±.000**

expert assignments become fully reliable. The degradation observed when removing β, removing $\mathcal{L}_{\mathrm{BAL}}$, or replacing WTA with soft gating supports the importance of these routing-stabilization components. The most severe drop occurs when the global flow is removed and only the experts are used. In our framework, the global backbone provides the coarse transport from noise to the data manifold, while the experts model mode-specific local corrections. Without the backbone, the experts must handle both global distribution alignment and local temporal refinement, which leads to a sharp degradation in both FID and discriminative score. These results support our design: the backbone defines the main transport path, and the routed experts refine the dynamics through local corrections.

F Model and Implementation Details

We use a unified multi-expert architecture with $K = 4$ experts for all datasets. The shared backbone is a Transformer encoder–decoder with n_{enc} encoder layers and n_{dec} decoder layers, which extracts sequence features and produces hidden representations h_t. A small MLP projector maps the pooled hidden features into a latent Koopman space $z_t \in \mathbb{R}^{d_z}$ ($d_z = 32$ for most datasets). Each expert

Table 9: Additional ablation studies on expert architecture, routing strategies, and the necessity of the global backbone.

Category	Variant	Stocks		fMRI	
		FID ↓	Dis ↓	FID ↓	Dis ↓
Full	Ours	**0.027**	**0.030**	**0.155**	**0.129**
Expert Type	MLP-based Experts	0.072	0.085	0.224	0.268
	CNN-based Experts	0.065	0.079	0.210	0.242
Routing	Soft Gating (Standard MoE)	0.098	0.115	0.285	0.342
	Random Routing	0.142	0.185	0.360	0.415
Backbone	w/o Global Flow (Experts only)	0.215	0.242	0.510	0.584

applies linear dynamics in this latent space and uses an MLP decoder to output a data-space velocity field. Expert selection is handled by a time-aware router implemented as an MLP that takes hidden features and a learned time embedding as input. Model size is adjusted to dataset scale: smaller datasets such as Sines use lighter encoder–decoder stacks and $d_{\text{model}} = 64$, while larger datasets such as fMRI use deeper stacks with $d_{\text{model}} = 96$. Architecture depth and width are adjusted according to dataset scale: smaller datasets such as Sines use lighter encoder-decoder stacks ($n_{\text{enc}} = 2$, $n_{\text{dec}} = 2$) with $d_{\text{model}} = 64$, while larger datasets such as fMRI use deeper networks ($n_{\text{enc}} = 4$, $n_{\text{dec}} = 4$) with $d_{\text{model}} = 96$. All models are trained with the Adam optimizer and a warmup ratio of 0.2. Batch sizes range from 64 to 256 depending on the dataset, and training runs for 10k to 28k epochs. This follows the same training protocol [36] across baselines for fairness.

G Additional Experimental Details

G.1 Datasets

Table 10 summarizes the main statistics of the datasets used in our experiments. All datasets are publicly available, and the corresponding sources are provided in the table.

Table 10: Dataset Details.

Dataset	# of Samples	dim	Source
Sines	10000	5	TimeGAN
Stocks	3773	6	Yahoo
ETTh(1)	17420	7	ETDataset
MuJoCo	10000	14	Google DeepMind
Energy	19711	28	UCI
fMRI	10000	50	Netsim

G.2 Baselines

We evaluate PrismFlow against a set of widely-used time-series generative baselines, with all models matched to comparable parameter budgets. The baselines cover four families: GANs, VAEs, diffusion models, and flow matching (FM). For GANs, we use TimeGAN [12] and Cot-GAN [29]. For VAEs, we use TimeVAE [31]. For diffusion, we include Diffwave [35], DiffTime [7], and Diffusion-TS [36]. In particular, Diffusion-TS is a recent diffusion model that has shown strong performance in both conditional and unconditional generation, and is often used as a competitive reference point [20]. We also include SMF [44] as the FM baseline; since SMF is formulated for unconditional generation, we only report its results in the unconditional setting.

G.3 Evaluation Metrics

Following [36, 44], we use four commonly adopted metrics to evaluate generation quality. We report Context-FID, where smaller values indicate that the generated sequences are closer to the real data

distribution. We compute an FID-like distance in a learned representation space by replacing the Inception network with TS2Vec [51]. Concretely, we sample real and synthetic sequences, encode them using a pretrained TS2Vec encoder, and compute the FID between the two sets of embeddings.

We also report a correlational score to quantify how well the synthetic data matches the dependency structure of the real data [36]. We first estimate the covariance between the i^{th} and j^{th} variables:

$$\text{cov}_{i,j} = \frac{1}{T}\sum_{t=1}^{T} X_i^t X_j^t - \left(\frac{1}{T}\sum_{t=1}^{T} X_i^t\right)\left(\frac{1}{T}\sum_{t=1}^{T} X_j^t\right). \tag{15}$$

We then compute the average absolute gap between the correlation matrices of real (r) and synthetic (f) data:

$$\frac{1}{10}\sum_{i,j}^{d}\left|\frac{\text{cov}_{i,j}^r}{\sqrt{\text{cov}_{i,i}^r}\text{cov}_{j,j}^r} - \frac{\text{cov}_{i,j}^f}{\sqrt{\text{cov}_{i,i}^f}\text{cov}_{j,j}^f}\right|. \tag{16}$$

For discriminative score, we follow TimeGAN [12] and train a classifier to distinguish real from synthetic sequences. The discriminative score is defined as $|\text{accuracy} - 0.5|$, where smaller values indicate that the classifier cannot reliably tell the two distributions apart.

For predictive score, we again follow TimeGAN and train a sequence prediction model for one-step-ahead forecasting. The predictive score is computed as the mean absolute error (MAE) between predicted values and the ground-truth values on the test set, where smaller is better. For both discriminative and predictive scores, we use the same model choices as [36]: a single-layer GRU classifier and a single-layer GRU sequence predictor, respectively.

We use the metrics above for the unconditional setting and MSE for the conditional setting. For the unconditional generation task, we perform five evaluations to obtain the experimental results. Similarly, for the conditional generation task, we run the procedure five times when computing performance metrics, and report the median imputation values as well as the mean and confidence intervals following [36].

G.4 Extension to Conditional Tasks

While we focus on unconditional time-series generation, PrismFlow naturally extends to conditional tasks such as imputation and forecasting by using partial observations as guidance, without retraining. Following flow-matching guidance [48], we treat the condition as an energy penalty on the terminal prediction and add the resulting covariance-guidance drift to the expert-augmented sampling velocity. This yields the guided sampler in Eq. (31), which improves consistency with observed values while largely preserving the unconditional transport dynamics learned by PrismFlow.

H More Visualization Results

We provide additional visualizations of unconditional generation using t-SNE and KDE in Figure 6. Imputation and forecasting results are shown in Figure 7. Figure 8 further compares the DMD eigenvalue distributions of generated data between SMF, a recent time-series foundation model based on signal vector fields, and our method. We also visualize the behavior of mode-specific experts in Figure 9, which illustrates the distinct temporal patterns captured by different experts.

I Theoretical Analysis

We provide a formal account of the mechanism underlying PrismFlow. The analysis separates three claims. First, a learned single-field FM estimator trained with a squared loss is a conditional-mean estimator and can lose conditional velocity energy in multi-modal regions. Second, hard WTA routing changes the regression problem from one global conditional mean to cell-wise conditional means, yielding an exact variance decomposition that quantifies the recovered mode energy. Third, the Koopman parameterization in Eq. (7) guarantees stable expert dynamics in latent space.

I.1 Mean-Seeking Bias and Energy Attenuation in Flow Matching

Recall the CFM objective in Eq. (3) with target velocity $u_t(x_0, x_1) = x_1 - x_0$ and interpolated state $x_t = (1-t)x_0 + tx_1$. For a fixed flow state $x_t = x$, let $U_t = u_t(x_0, x_1)$. All conditional expectations below are assumed finite.

Theorem I.1 (Conditional Projection and Velocity-Energy Gap)**.** *For any fixed (t, x), the unique minimizer of*

$$\min_{v \in \mathcal{X}} \mathbb{E}\big[\|v - U_t\|_2^2 \mid x_t = x\big]$$

is

$$v_t^*(x, t) = \mathbb{E}[U_t \mid x_t = x]. \tag{17}$$

Moreover,

$$\mathbb{E}\big[\|U_t\|_2^2 \mid x_t = x\big] - \|v_t^*(x, t)\|_2^2 = \mathrm{Tr}(\mathrm{Cov}(U_t \mid x_t = x))\,. \tag{18}$$

Thus, whenever $\mathrm{Cov}(U_t \mid x_t = x)$ is nonzero, the single-output ℓ_2 estimator has strictly smaller squared magnitude than the conditional second moment of the target velocities.

Proof. Let $m = \mathbb{E}[U_t \mid x_t = x]$. For any deterministic $v \in \mathcal{X}$,

$$\mathbb{E}[\|v - U_t\|_2^2 \mid x_t = x] = \|v - m\|_2^2 + \mathbb{E}[\|U_t - m\|_2^2 \mid x_t = x],$$

because the cross term equals $2(v-m)^\top \mathbb{E}[m - U_t \mid x_t = x] = 0$. The risk is therefore uniquely minimized at $v = m$, proving Eq. (17). Taking $v = 0$ in the same decomposition gives

$$\mathbb{E}[\|U_t\|_2^2 \mid x_t = x] = \|m\|_2^2 + \mathbb{E}[\|U_t - m\|_2^2 \mid x_t = x],$$

where the final term is the trace of the conditional covariance. □

Remark I.2 (Estimator-Level, Not Transport-Level). Theorem I.1 characterizes the population optimum of a single-output squared-loss estimator at a fixed flow state. It does not imply that the exact FM transport field is ill-defined or suffers from mode collapse. Rather, it explains why a learned monolithic estimator can average incompatible conditional velocities when $q(U_t \mid x_t = x)$ is multi-modal.

Lemma I.3 (Two-Branch Cancellation)**.** *Suppose U_t conditioned on $x_t = x$ takes values a and b with probabilities ρ and $1 - \rho$. Then the single-field optimum satisfies*

$$\|v_t^*(x, t)\|_2^2 = \rho\|a\|_2^2 + (1-\rho)\|b\|_2^2 - \rho(1-\rho)\|a - b\|_2^2. \tag{19}$$

For balanced opposite branches $b = -a$ and $\rho = 1/2$, the optimum is $v_t^*(x, t) = 0$ although every target velocity has norm $\|a\|_2$.

Proof. The conditional mean is $m = \rho a + (1-\rho)b$. Expanding $\|m\|_2^2$ and subtracting it from $\rho\|a\|_2^2 + (1-\rho)\|b\|_2^2$ gives $\rho(1-\rho)\|a - b\|_2^2$. The special case $b = -a$ and $\rho = 1/2$ follows immediately. □

The lemma makes the failure mode explicit: branch disagreement is converted into a loss of velocity energy. For time series, repeated ODE integration of such attenuated velocities can suppress oscillatory and transient components, which corresponds to the spectral contraction observed by DMD in Sec. 5.

I.2 Mode Specialization Induced by WTA Routing

PrismFlow introduces a bank of experts and selects a single expert at each (t, x_t) using the score in Eq. (9) and the winner rule in Eq. (10). For a fixed $(t, x_t = x)$, assume ties occur with probability zero. The WTA rule induces a measurable partition of the conditional target space:

$$\mathcal{V}_t^k(x) = \left\{ x_1 \;\middle|\; k = \arg\min_{j \in \{1,\dots,K\}} \mathcal{S}_t^j(t, x, x_1) \right\}, \tag{20}$$

where $\mathcal{S}_t^k$ is defined in Eq. (9). Let C_k denote the event $\{x_1 \in \mathcal{V}_t^k(x)\}$ and $\pi_k = \mathbb{P}(C_k \mid x_t = x)$.

Theorem I.4 (Cell-Wise Regression Induced by Hard WTA)**.** *Consider a fixed flow state (t, x) and a fixed WTA partition $\{C_k\}_{k=1}^K$. For expert predictions $e_k \in \mathcal{X}$, define*

$$\mathcal{R}_{\mathrm{WTA}}(e_1, \ldots, e_K) = \sum_{k=1}^{K} \pi_k \, \mathbb{E}\big[\|e_k - U_t\|_2^2 \mid x_t = x, C_k\big] \, .$$

Then, for every cell with $\pi_k > 0$, the unique minimizer is

$$e_k^*(x, t) = \mathbb{E}[U_t \mid x_t = x, C_k]. \tag{21}$$

With the implementation in Eq. (11)*, gradient masking performs stochastic optimization of this cell-wise risk: expert k receives gradients only from samples assigned to C_k.*

Proof. The risk decomposes into independent conditional least-squares problems, one for each cell. For any k with $\pi_k > 0$, the projection argument in Theorem I.1 gives the unique minimizer $e_k^* = \mathbb{E}[U_t \mid x_t = x, C_k]$. In the algorithm, the sample loss contains only the winning expert index k_t^* and masks gradients to all non-winning experts. Thus, the stochastic gradient for expert k is sampled only from the event C_k, giving an unbiased stochastic gradient of the k-th summand of $\mathcal{R}_{\mathrm{WTA}}$. □

Theorem I.5 (Exact Variance Decomposition and Energy Recovery)**.** *Let $m = \mathbb{E}[U_t \mid x_t = x]$ and $m_k = \mathbb{E}[U_t \mid x_t = x, C_k]$. Then*

$$\sum_{k=1}^{K} \pi_k \|m_k\|_2^2 - \|m\|_2^2 = \sum_{k=1}^{K} \pi_k \|m_k - m\|_2^2 \geq 0. \tag{22}$$

Equivalently,

$$\mathrm{Tr}(\mathrm{Cov}(U_t \mid x_t = x)) = \sum_{k=1}^{K} \pi_k \, \mathrm{Tr}(\mathrm{Cov}(U_t \mid x_t = x, C_k)) + \sum_{k=1}^{K} \pi_k \|m_k - m\|_2^2. \tag{23}$$

Consequently, WTA routing recovers exactly the between-cell velocity energy removed by the single-field conditional mean. The improvement is strict whenever two positive-probability cells have different conditional means.

Proof. Since $m = \sum_k \pi_k m_k$,

$$\sum_k \pi_k \|m_k\|_2^2 - \|m\|_2^2 = \sum_k \pi_k \|m_k - m\|_2^2,$$

which proves Eq. (22). For Eq. (23), write $U_t - m = (U_t - m_k) + (m_k - m)$ on C_k. Taking conditional squared norms and averaging over cells eliminates the cross term because $\mathbb{E}[U_t - m_k \mid x_t = x, C_k] = 0$. This gives the trace form of the law of total covariance. □

Remark I.6 (Residual Experts). The same decomposition applies to PrismFlow's residual experts by replacing U_t with $U_t - b_t(x)$, where $b_t(x)$ is the global FM prediction at the fixed state. Subtracting a deterministic baseline does not change the between-cell variance term. Thus, WTA residual experts recover mode-specific corrections that a single residual regressor would average away.

I.3 Koopman-Structured Stability of the Experts

Each expert evolves in latent space via Eq. (5) with generator A^k parameterized by Eq. (7). We show that this construction guarantees stable expert dynamics.

Theorem I.7 (Contractive Koopman Expert Dynamics)**.** *Consider the latent ODE $\dot{z}_t = A^k z_t$ with*

$$A^k = \big(S^k - (S^k)^\top\big) - (R^k)^\top R^k - \delta I \quad \textit{(Eq. (7))},$$

where $\delta \geq 0$. Then the flow generated by expert k is non-expansive for $\delta = 0$ and exponentially contractive for $\delta > 0$:

$$\|z_\tau\|_2 \leq e^{-\delta(\tau - s)} \|z_s\|_2, \qquad 0 \leq s \leq \tau. \tag{24}$$

In particular, for $\delta > 0$, the origin is globally exponentially stable.

Proof. Let $V(z) = \frac{1}{2}\|z\|_2^2$. Along trajectories,

$$\dot{V} = z^\top A^k z = z^\top \big(S^k - (S^k)^\top\big) z - \|R^k z\|_2^2 - \delta\|z\|_2^2.$$

The first term is zero because $S^k - (S^k)^\top$ is skew-symmetric. Hence $\dot{V} \le -\delta\|z\|_2^2 = -2\delta V$. Applying Grönwall's inequality on $[s, \tau]$ yields $V(z_\tau) \le e^{-2\delta(\tau - s)} V(z_s)$, which proves Eq. (24). □

Corollary I.8 (Bounded Data-Space Expert Velocity)**.** *Assume the reachable latent set is bounded at the start of an expert step and that g_ψ is L_g-Lipschitz on the reachable set of pairs $(z, A^k z)$, with $\|g_\psi(0, 0)\| \le c_g$. Then along the expert dynamics,*

$$\|v_t^{\psi,k}(x_t, t)\|_2 \le c_g + L_g(1 + \|A^k\|_2) e^{-\delta(t-s)} \|z_s\|_2.$$

Proof. By Eq. (6), $v_t^{\psi,k} = g_\psi(z_t, A^k z_t)$. Lipschitz continuity gives

$$\|g_\psi(z_t, A^k z_t)\| \le c_g + L_g(\|z_t\| + \|A^k z_t\|).$$

Using $\|A^k z_t\| \le \|A^k\|_2 \|z_t\|$ and Theorem I.7 proves the claim. □

Theorem I.1 and Lemma I.3 formalize estimator-level mean seeking. Theorems I.4–I.5 show that WTA routing converts global averaging into cell-wise regression and recovers the between-cell velocity energy lost by a monolithic conditional mean. Theorem I.7 and Corollary I.8 show that Koopman experts can represent oscillatory modes through skew-symmetric components while remaining stable through dissipative components.

J Theory of Conditional Guidance

We next justify the inference-time observation-consistency guidance used for imputation and forecasting. The goal is to show that conditional sampling can be obtained by tilting the terminal distribution with an observation energy and adding the corresponding guidance drift to the unconditional PrismFlow vector field.

J.1 Energy-Based Formulation of Conditional Generation

Let y denote a partial observation or external condition. We introduce a time-independent energy potential $\mathcal{J} : \mathcal{X} \to \mathbb{R}$ such that the target conditional distribution satisfies

$$p(x \mid y) \propto p(x) e^{-\mathcal{J}(x)}. \tag{25}$$

For Gaussian imputation with observation mask M, for example,

$$\mathcal{J}(x) = \frac{1}{2\sigma_y^2} \|M \odot (x - y)\|_2^2.$$

Let $p(x_1 \mid x_t)$ be the endpoint posterior induced by the unconditional flow at time t. Tilting the terminal distribution by $e^{-\mathcal{J}}$ gives

$$p_{\mathcal{J}}(x_1 \mid x_t) = \frac{e^{-\mathcal{J}(x_1)} p(x_1 \mid x_t)}{\mathbb{E}_{p(x_1 \mid x_t)}[e^{-\mathcal{J}(x_1)}]}. \tag{26}$$

If $v_{t|1}(x_t, x_1)$ denotes the conditional velocity associated with endpoint x_1, the exact guided drift is

$$u_t(x_t) = \mathbb{E}_{p_{\mathcal{J}}(x_1 \mid x_t)}[v_{t|1}(x_t, x_1)]. \tag{27}$$

Thus, the additive guidance field is

$$g_t(x_t) = u_t(x_t) - \mathbb{E}_{p(x_1 \mid x_t)}[v_{t|1}(x_t, x_1)].$$

The exact expectation in Eq. (27) is generally intractable, motivating the local approximation below.

J.2 Localized Covariance Guidance

Assumption J.1 (Local Endpoint Posterior)**.** Conditioned on x_t, the endpoint posterior is locally approximated by $X_1 \sim \mathcal{N}(\hat{x}_1, \Sigma_{1|t})$. The energy $\mathcal{J}$ is twice continuously differentiable near $\hat{x}_1$, with Hessian operator norm bounded by L_J. The tilted distribution in Eq. (26) satisfies

$$\mathbb{E}_{p_{\mathcal{J}}}[\|X_1 - \hat{x}_1\|_2 \mid x_t] \le c_0 \|\Sigma_{1|t}\|_2^{1/2}$$

for a finite constant c_0.

Lemma J.2 (First-Order Gaussian Tilt)**.** *Under Assumption J.1,*

$$\mathbb{E}_{p_{\mathcal{J}}}[X_1 \mid x_t] - \hat{x}_1 = -\Sigma_{1|t} \nabla \mathcal{J}(\hat{x}_1) + r_t, \qquad \|r_t\|_2 \le c\, L_J \|\Sigma_{1|t}\|_2^{3/2}, \tag{28}$$

with $c = c_0$.

Proof. For a Gaussian base posterior, Stein's identity gives

$$\mathbb{E}_{p_{\mathcal{J}}}[X_1 - \hat{x}_1 \mid x_t] = \Sigma_{1|t} \mathbb{E}_{p_{\mathcal{J}}}[\nabla_{x_1} \log e^{-\mathcal{J}(X_1)} \mid x_t] = -\Sigma_{1|t} \mathbb{E}_{p_{\mathcal{J}}}[\nabla \mathcal{J}(X_1) \mid x_t].$$

Adding and subtracting $\nabla \mathcal{J}(\hat{x}_1)$ yields the stated first-order term plus a remainder. Since $\nabla \mathcal{J}$ is locally L_J-Lipschitz, the remainder is bounded by

$$\|\Sigma_{1|t}\|_2 L_J \mathbb{E}_{p_{\mathcal{J}}}[\|X_1 - \hat{x}_1\|_2 \mid x_t],$$

and the moment bound in Assumption J.1 proves Eq. (28). □

Theorem J.3 (Affine-Path Covariance Guidance)**.** *Assume the affine path $x_t = (1-t)x_0 + t x_1$ with conditional velocity*

$$v_{t|1}(x_t, x_1) = \frac{x_1 - x_t}{1 - t},$$

and suppose Assumption J.1 holds. Then the additive guidance field satisfies

$$g_t(x_t) = -\frac{1}{1-t} \Sigma_{1|t} \nabla_{\hat{x}_1} \mathcal{J}(\hat{x}_1) + \varepsilon_t, \qquad \|\varepsilon_t\|_2 \le \frac{c L_J}{1-t} \|\Sigma_{1|t}\|_2^{3/2}. \tag{29}$$

Thus, covariance guidance is the first-order drift correction induced by terminal energy tilting.

Proof. For the affine path, $v_{t|1}(x_t, x_1) = (x_1 - x_t)/(1-t)$. Therefore,

$$g_t(x_t) = \frac{1}{1-t} \left(\mathbb{E}_{p_{\mathcal{J}}}[X_1 \mid x_t] - \mathbb{E}_p[X_1 \mid x_t] \right).$$

Since $\mathbb{E}_p[X_1 \mid x_t] = \hat{x}_1$, substituting Lemma J.2 proves Eq. (29). □

Remark J.4 (Practical Scalar Scheduling). In implementation, $\Sigma_{1|t}$ can be replaced by a diagonal, scalar, or learned proxy, and the singular factor $1/(1-t)$ is absorbed into a bounded guidance schedule. The theorem justifies the direction of the guidance correction, while the scalar schedule controls numerical stiffness near the terminal endpoint.

J.3 Compatibility with PrismFlow Sampling Dynamics

PrismFlow samples from the unconditional model distribution using the hard-routed expert-augmented vector field

$$\tilde{\mathbf{v}}_t(x_t, t) := v_t^{\theta}(x_t, t) + \gamma\, \lambda_t\, v_t^{\psi, k_t^*}(x_t, t), \tag{30}$$

which serves as the base probability flow for the learned unconditional distribution. The guidance derivation depends only on the endpoint posterior induced by the chosen base flow, not on whether the base field is represented as a single network or as a global-plus-expert decomposition. Therefore, the conditional PrismFlow sampler uses

$$u_t(x_t, t) = \tilde{\mathbf{v}}_t(x_t, t) + g_t(x_t, t), \tag{31}$$

where g_t is approximated by Eq. (29) with a practical scalar schedule. The expert term improves the unconditional prior dynamics by recovering mode-specific temporal structure, while the guidance term tilts the terminal endpoint distribution toward observation consistency. Since both terms are vector-field corrections evaluated at the same state, they can be composed additively during inference without retraining the global backbone or the expert bank.

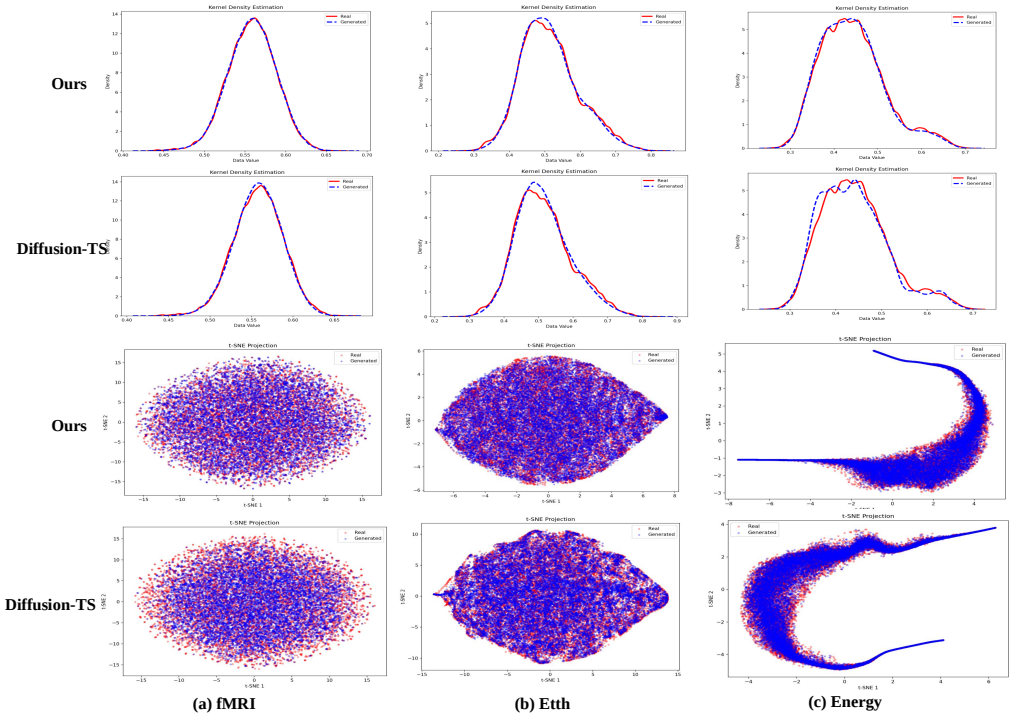
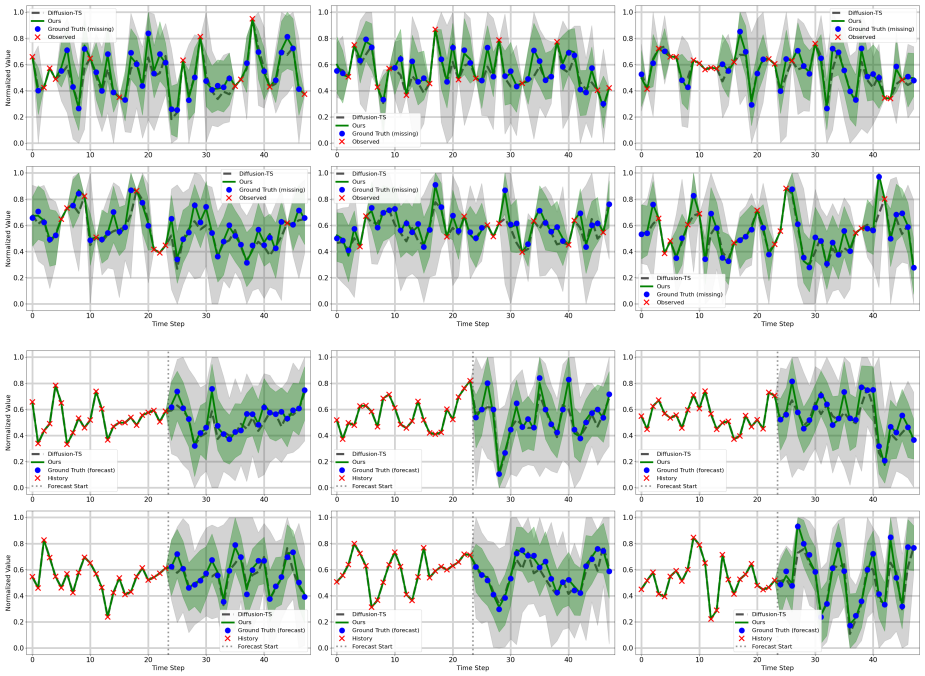
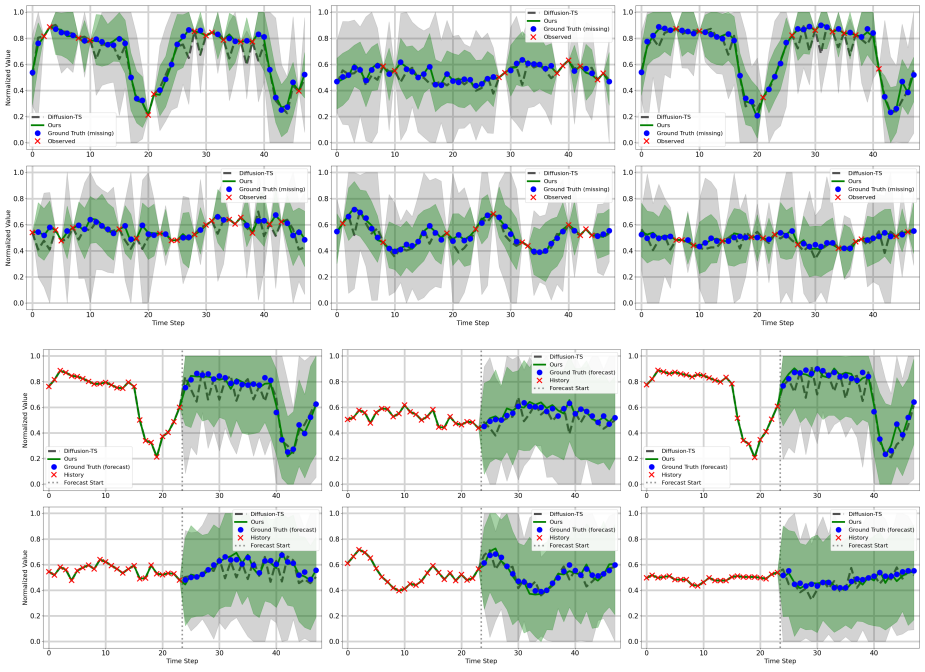


Figure 6: t-SNE and KDE visualizations of time series synthesized by our method and Diffusion-TS.



(a) fMRI



(b) Etth

Figure 7: Comparison of conditional imputation and forecasting tasks. For each method, the median imputed values are shown as a line, and the 5% and 95% quantiles are shown as a shaded region.

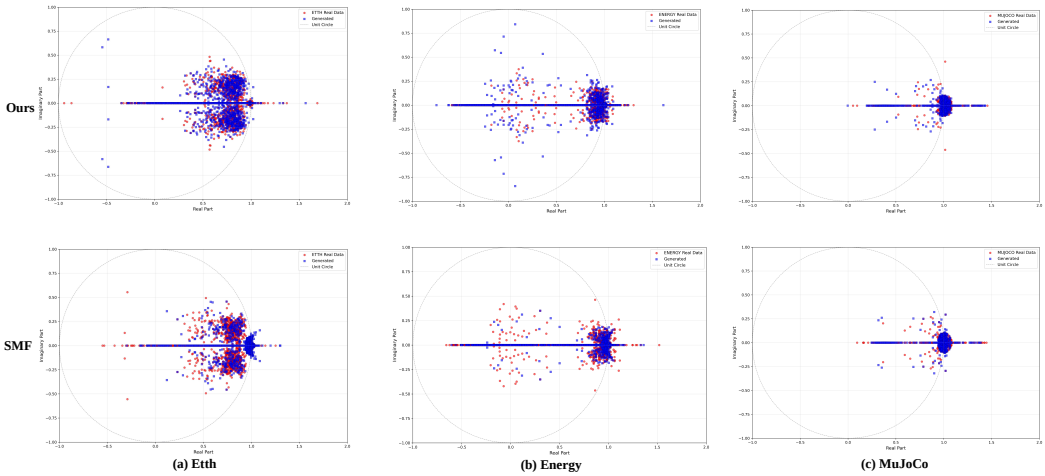


Figure 8: Comparison of DMD Eigenvalues.

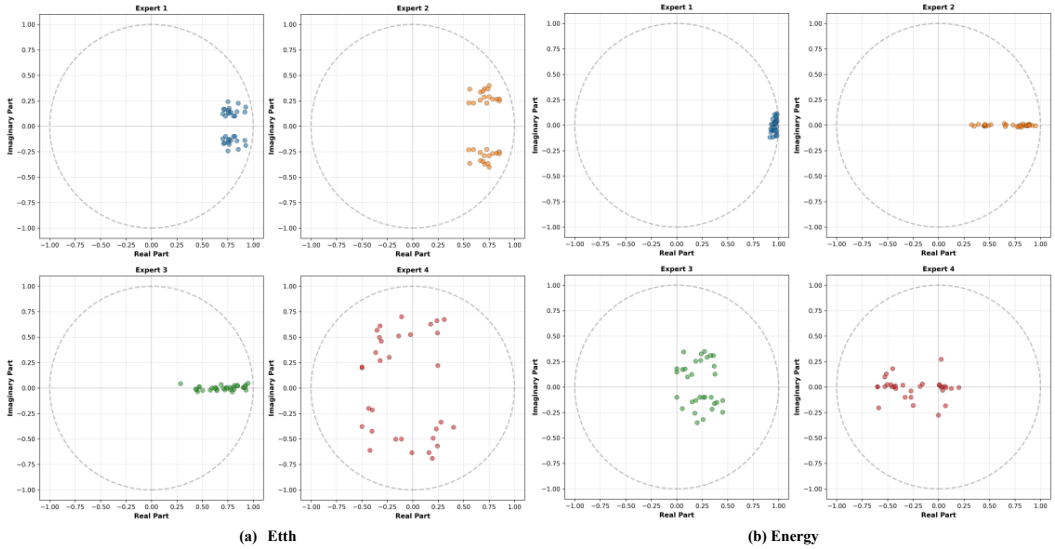


Figure 9: DMD Eigenvalues of four Experts for the ETTh and Energy datasets.

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 motivation, method, scope, and empirical contributions of PrismFlow, including its residual expert design and the reported gains across generation, forecasting, and imputation tasks.

   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 in Appendix A.

   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: Theoretical assumptions, formal statements, and proofs are provided in Appendix I and Appendix J.

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 paper provides the method, pseudo-code, datasets, metrics, model details in Sec. 5, Appendix C, and Appendix G, and includes the code package in the supplementary material.

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: The paper states that the code is provided in the supplementary material.

   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: The experimental settings, including datasets, evaluation protocols, and hyperparameter settings, are specified in Appendix G.

   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: Main quantitative tables report results with mean and variation terms over repeated evaluations, and the appendix states that unconditional and conditional metrics are computed over five runs with confidence intervals for conditional tasks.

   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: Appendix D.3 reports wall-clock training and sampling time on a single NVIDIA GeForce RTX 4090 (24GB), and the implementation details provide training epochs, batch sizes, timesteps, and sampling steps.

   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 research uses public time-series benchmarks and standard experimental protocols, and we have reviewed and followed 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: Appendix B discusses impacts for real-world time-series applications.

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 high-risk pretrained language models, image generators, scraped datasets, or other assets with a high risk of misuse.

    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: We use publicly available datasets and baseline methods for research and evaluation. The original sources are credited in the main text and Appendix G. For each dataset, Appendix G reports the source. We do not redistribute the datasets and use them only in accordance with their stated research-use conditions.

    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: [Yes]

    Justification: The released code and documentation are provided in the supplemental material, including implementation details and reproduction instructions.

    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.

    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 experiments or research with human subjects, so 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: We do not use LLMs for developing our core idea and methodology.

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.