Tyche: One Step Flow for Efficient Probabilistic Weather Forecasting

Fan Xu[1,2*], **Yuan Gao**[3*], **Kun Wang**[4], **Rui Su**[2†], **Fenghua Ling**[5], **Hao Wu**[3†], **Wanli Ouyang**[2,6†]
[1] University of Science and Technology of China [2] Shenzhen Loop Area Institute [3] Tsinghua University
[4] Nanyang Technological University [5] Shanghai AI Lab [6] The Chinese University of Hong Kong
[*] Equal Contribution, [†] Corresponding author
Main Contact: markxu@mail.ustc.edu.cn

Abstract

Probabilistic weather forecasting requires not only accurate trajectories, but calibrated distributions over plausible atmospheric futures. Recent data-driven systems have achieved remarkable deterministic skill, and diffusion-based ensemble forecasters have substantially improved sample realism and uncertainty quantification. However, their inference cost scales with forecast horizon, ensemble size, and the number of denoising steps required for each transition, making large operational ensembles expensive. To address this, we present `Tyche`, a one-step conditional flow model for efficient probabilistic weather forecasting. `Tyche` models the conditional forecast distribution with a destination-aware average-velocity flow that maps Gaussian noise directly to future weather states in a single function evaluation (1-NFE). To make this one-step transport learnable in high-dimensional geophysical fields, we derive a JVP-regularized rectification objective that enforces temporal self-consistency across source and destination flow timesteps without explicitly forming Jacobians. The transport field is parameterized by an isotropic Swin-style transformer that preserves fine-scale spatial structure while remaining scalable on global grids. To improve ensemble reliability under autoregressive forecasting, we further introduce a rollout-based finetuning stage with curriculum CRPS calibration supervision. Experiments on ERA5 at 1.5° and 6-hour resolution show that our `Tyche`, using merely a single NFE, matches or exceeds the forecast skill and calibration of state-of-the-art multi-step generative baselines and the operational ECMWF IFS ensemble. Codes are available.

1 Introduction

Many high-impact decisions in aviation, energy, agriculture, disaster response, and climate-risk management [33, 34, 41, 49] depend not on a single most likely atmospheric trajectory, but on a calibrated distribution over plausible futures. This is especially important at medium-range lead times, where small perturbations in the initial state can amplify into substantially different outcomes. Recent deterministic machine-learning weather prediction (MLWP) systems [1, 12, 19, 47, 15, 50], such as FourCastNet [38], Pangu-Weather [4], GraphCast [27], and NeuralGCM [25], have achieved comparable or better performance compared to traditional operational weather prediction systems [6, 36, 42]. Although they have much lower computing costs, these deterministic models fail to faithfully represent high-impact extreme events as forecast uncertainty grows over time.

Recent work on global probabilistic weather forecasting has begun to close this gap. One line of work revisits the forecast objective itself, replacing deterministic regression with explicitly probabilistic training, for example through hierarchical latent-variable graph models [37, 51] or proper-score optimization in AIFS-CRPS [28]. Another line embraces explicit generative modeling. Diffusion-based systems [24] such as GenCast produce sharp samples with realistic spectra and strong extreme-

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

event skill [39]. Follow-up work explores direct lead-time conditioning with temporally correlated noise [3], rolling diffusion with progressive uncertainty schedules [7], latent-space diffusion [53], and one-step consistency-style forecasting [45]. Together, these works establish that ensemble skill, calibration, and realism are all attainable in data-driven forecasting.

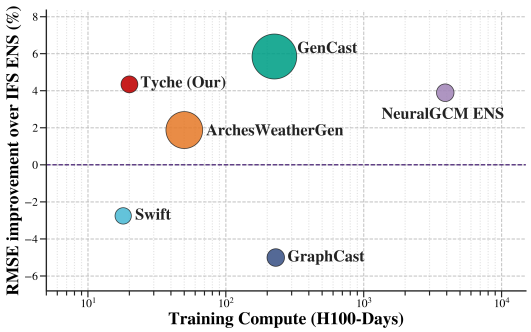


Figure 1: Visualization of forecast performance, training cost, and inference speed. The larger the circle, the slower the inference speed.

While, one important remaining bottleneck is efficiency [46]. Existing probabilistic approaches typically require more than 20 function evaluations, resulting in over an order of magnitude higher computational cost than deterministic approaches. In global probabilistic forecasting, the cost of inference compounds along three axes: the rollout length L, the ensemble size K, and the number of denoising or integration steps N_{NFE} required for each forecast transition. For diffusion and standard continuous-time flow models, this yields the unfavorable scaling total NFEs $\propto L \times K \times N_{\mathrm{NFE}}$. When $N_{\mathrm{NFE}} \gg 1$, the settings in which ensembles matter most, namely long forecast horizons and large ensemble sizes, are also the settings in which sampling becomes most expensive [3, 7, 39, 53]. Recent one-step models partially alleviate this tension [44, 32, 45], but its sampling methods are not flexible and often restricted to distillation frame. So the field still lacks a transport formulation explicitly designed for one-step conditional generation on high-dimensional geophysical states.

"For operational ensemble forecasting, the key design choice is not only the backbone, but the transport object." Standard flow matching parameterizes an instantaneous velocity field and therefore inherits numerical integration at test time [30, 48, 26]. By contrast, recent one-step generative work suggests that interval-averaged transport can make single-step sampling intrinsic rather than distilled [31, 17]. We adopt this viewpoint for probabilistic weather forecasting and propose `Tyche`, a one-step conditional flow model targeting *destination-aware average velocity*. Given a conditioning atmospheric state and a noisy latent state, `Tyche` predicts the average transport needed to move directly from the current point on the probability path to a chosen destination time, enabling direct 1-NFE sampling. The transport field is parameterized by an isotropic Swin-style transformer that preserves fixed spatial resolution, using local windows, shifted windows, and axial rotary position encoding to retain fine-scale meteorological structure on global grids. As efficient transport alone does not guarantee effective ensembles, we further separate transport learning from uncertainty calibration. Specifically, after one-step pretraining, we inference the model with parallel autoregressive ensemble rollouts, and finetune with a CRPS-based curriculum probabilistic loss. As shown in Figure 1, our method achieves highly competitive performance while maintaining low training and inference costs.

In summary, we make the following key contributions: ❶ ***Efficient Probabilistic Framework.*** We propose `Tyche`, a conditional 1-NFE flow-based generative framework specifically tailored for global probabilistic weather forecasting. `Tyche` effectively resolves the scalability bottleneck of multi-step diffusion models at the cost of deterministic baselines. ❷ ***Novel Two-Stage Strategy.*** To achieve distillation-free one-step generation on high dimensional atmospheric grids, we introduce a destination-aware average-velocity flow to bypass intractable computational overheads. Further, we propose a curriculum autoregressive finetuning strategy based on the CRPS objective to overcome the ensemble collapse and error accumulation issues. ❸ ***Superior Performance.*** Extensive experiments demonstrate that `Tyche` achieves highly calibrated ensembles and exceptional extreme weather prediction skills using merely a single function evaluation. To be specific, our proposed `Tyche` matches or exceeds the ensemble skill of state-of-the-art multi-step generative baselines, while running $12-43\times$ faster than these generative baselines.

2 Related Work

Probabilistic Weather Forecasting. Modern machine-learning weather prediction [8, 5, 2, 14] has been driven by highly accurate deterministic forecasters such as FourCastNet [38], Pangu-Weather [4], GraphCast [27], and NeuralGCM [25]. While these models achieve strong medium-range skill, they do not explicitly model the conditional distribution of future states, and regression-based training

tends to produce conditional-mean forecasts as uncertainty grows. To address this limitation, recent work has shifted toward probabilistic forecasting [29], especially explicit generative modeling like diffusion. GenCast [39] established diffusion as a strong paradigm for global probabilistic weather forecasting, showing that learned ensembles can achieve diverse samples and strong downstream utility. LaDCast [53] moves diffusion into a compressed latent space to improve scalability. ERDM [7] models a forecast window jointly with progressively increasing uncertainty, adapting EDM design choices to sequence forecasting. However, despite their competitive performance, most existing methods remain constrained by multi-step denoising, with inference costs often exceeding those of deterministic models by more than an order of magnitude. In contrast, our method aims to achieve exceptional and efficient probabilistic forecasting through intrinsic one-step conditional transport.

Efficient Generative Transport. Diffusion models and neural ODEs provide expressive continuous generative frameworks [9, 22, 11], while flow matching formulates generation through learned instantaneous dynamics [30, 31, 16]. However, these formulations typically require either multi-step denoising or numerical integration at inference time, which becomes increasingly expensive in high-dimensional autoregressive settings. Recent work therefore explores transport parameterizations that enable few-step or one-step generation, including consistency models [44, 32], Shortcut models [13], and MeanFlow-style formulations based on interval-averaged transport [17, 18, 52]. In weather forecasting, Swift [45] is the closest 1-NFE counterpart, utilizing continuous-time consistency models to improve long-horizon ensemble forecasting efficiency. Instead of parameterizing instantaneous dynamics or a distilled consistency map, our method directly learns a destination-aware average velocity through JVP-regularized rectification for one-step conditional geophysical transport.

3 Preliminaries

Problem setup. We consider probabilistic weather forecasting over a sequence of high-dimensional atmospheric fields $\{x^t\}_{t=1}^T$, where $x^t \in \mathbb{R}^{C\times H\times W}$ denotes the weather state at time t, with C physical variables (*e.g.*, *temperature, geopotential, humidity*), and H, W represent the spatial grid dimensions of latitude and longitude. Given the current observations x^t as conditioning state, our goal is to learn the conditional probability distribution $p_\psi(x^{t+1} \mid x^t)$ of the next future state x^{t+1}. Further, we model the forecast dynamics as a Markov chain, so that the conditional distribution at an arbitrary lead time h can be obtained through recursive transitions. Following the standard autoregressive forecasting formulation, this conditional distribution can be factorized as:

$$p_\psi\big(x^{t+1:t+h} \mid x^t\big) = \prod_{i=1}^{h} p_\psi\big(x^{t+i} \mid x^{t+i-1}\big)\,. \tag{1}$$

Diffusion and Flow Matching formulation. Diffusion models [22, 35] define a forward corruption process that gradually perturbs a clean target sample $x \sim p_{\text{data}}(x)$ into noise over a continuous timestep $t \in [0, 1]$. Specifically, the noisy state is constructed as $z_t = \varphi_t x + \sigma_t \epsilon$ with $\epsilon \sim \mathcal{N}(0, I)$, where φ_t and σ_t are pre-defined schedule coefficients that depend on t and satisfy $z_0 = x$ and $z_1 = \epsilon$. Flow matching [30] provides a deterministic alternative by choosing a straight probability path between the noise distribution and the data distribution, setting $\varphi_t = 1 - t$ and $\sigma_t = t$:

$$z_t = (1 - t)x + t\epsilon, \quad v_t \triangleq v(z_t, t|x) = \epsilon - x, \tag{2}$$

where v_t denotes the corresponding ground-truth velocity along the path. A neural vector field $v_\theta(z_t, t)$ is then trained to match this target velocity by minimizing:

$$\mathcal{L}_{\text{FM}}(\theta) = \mathbb{E}_{t,\,x,\,\epsilon}\left[\|v_\theta(z_t, t) - v_t\|_2^2\right]. \tag{3}$$

After training, generation is performed by solving the probability flow ODE (PF-ODE) $dz_t/dt = v_\theta(z_t, t)$ from $t = 1$ to $t = 0$, starting with an initial value $z_1 \sim \mathcal{N}(0, I)$.

4 Methodology

Framework Overview. Given a conditioning atmospheric state $c = x^\tau$, our goal is to model the conditional distribution of a future weather field $x = x^{\tau+1}$. As shown in Figure 2, `Tyche` combines a destination-aware average-velocity flow formulation with a Swin-based velocity network. The flow

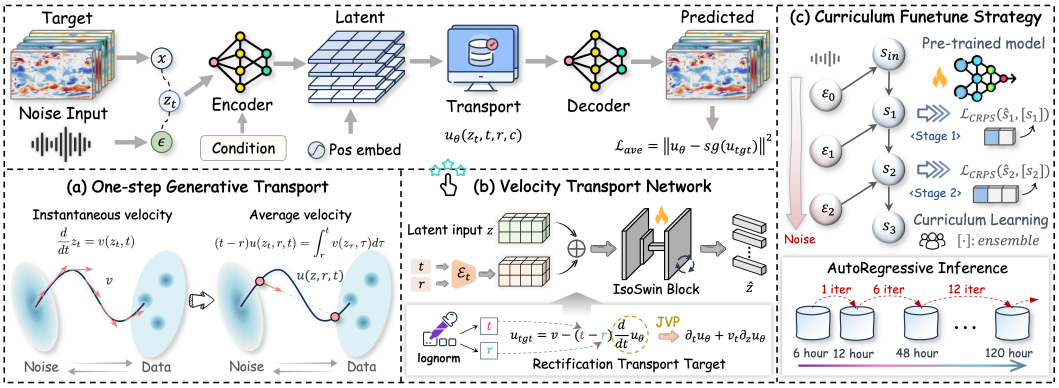


Figure 2: The overview of **Tyche**. *(a):* Learn a average-velocity field with JVP-based rectification for direct one-step transport. *(b):* Parameterize the transport field with a conditional isotropic Swin-style backbone. *(c):* Finetune the transport model with curriculum autoregressive ensemble rollouts.

formulation enables direct transport from Gaussian noise to a forecast state in one function evaluation, while the network provides a scalable parameterization for high-resolution global grids. Training is conducted in two stages: we first pretrain the transport field with a JVP-based rectification objective, and then finetune the model with rollout-based probabilistic supervision for ensemble calibration.

4.1 One-Step Generative Transport via Rectified Average Velocity

To distinguish the physical forecast horizon from the generative transport process, we reserve τ for the physical forecast index and use $t \in [0, 1]$ to denote the continuous flow time. Conditioned on the current atmospheric state x^τ, we consider the standard linear probability path between the target future field $x^{\tau+1}$ and Gaussian noise $\epsilon \sim \mathcal{N}(0, I)$:

$$z_t = (1-t)x^{\tau+1} + t\epsilon, \quad t \in [0, 1]. \tag{4}$$

The reference instantaneous velocity driving this path is constant, namely $v_t \triangleq \frac{\mathrm{d}z_t}{\mathrm{d}t} = \epsilon - x^{\tau+1}$.

Standard flow matching parameterizes an instantaneous vector field and typically requires multi-step numerical integration at inference time. Inspired by MeanFlow [17], we aim to directly learn the *destination-conditioned average velocity* required to transport a state from the current flow time t to a target destination time r ($0 \le r < t \le 1$).

$$u^\star(z_t, r, t, x^\tau) \triangleq \frac{1}{t-r}\int_r^t v(z_s, s, x^\tau)\,\mathrm{d}s = \frac{z_t - z_r}{t-r}, \tag{5}$$

where $v(z_s, s, x^\tau)$ denotes the underlying instantaneous velocity field of the conditional transport given x^τ. By definition, this average-velocity field induces the transport relation:

$$z_r = z_t - (t-r)u^\star(z_t, r, t, x^\tau), \tag{6}$$

Direct supervision of $u^\star(z_t, r, t, x^\tau)$ is intractable because it requires evaluating the time integral in Eq. (5). To obtain a trainable objective, we exploit the differential identity satisfied by the average-velocity field (with respect to t, while r is independent):

$$u^\star(z_t, r, t, x^\tau) = v_t - (t-r)\frac{d}{dt}u^\star(z_t, r, t, x^\tau), \quad \frac{d}{dt}u^\star(z_t, r, t, x^\tau) \triangleq v_t\,\partial_z u^\star + \partial_t u^\star, \tag{7}$$

where $u^\star$ is the abbreviation of $u^\star(z_t, r, t, x^\tau)$. We know that the total derivative can be interpreted as a Jacobian-vector product (JVP) between the Jacobian $[\partial_z u^\star, \partial_r u^\star, \partial_t u^\star]$ and the tangent vector $[v_t, 0, 1]$, and can therefore be computed efficiently using forward-mode automatic differentiation. We therefore parameterize a conditional transport field $u^\star_\theta(z_t, r, t, x^\tau)$ and construct the practical rectified target:

$$u_{\mathrm{tgt}}(z_t, r, t, x^\tau) = v_t - (t-r)\,(v_t\,\partial_z u^\star_\theta + \partial_t u^\star_\theta)). \tag{8}$$

We then optimize the first stage transport objective:

$$\mathcal{L}_{\mathrm{ave}}(\theta) = \mathbb{E}_{\epsilon,\,x^\tau,\,x^{\tau+1}}\,\mathbb{E}_{t\sim\mathcal{U}[0,1],\,r\sim\mathcal{U}[0,t]}\left[\|u_\theta(z_t, r, t, x^\tau) - \mathrm{sg}(u_{\mathrm{tgt}}(z_t, r, t, x^\tau))\|_F^2\right], \tag{9}$$

where $\mathrm{sg}(\cdot)$ denotes the stop-gradient operator. At inference time, one-step conditional generation is obtained by evaluating the learned average-velocity field only once at the noise endpoint:

$$\hat{x}^{\tau+1} = \epsilon - u_\theta(\epsilon, 0, 1, x^\tau), \tag{10}$$

which produces a forecast sample with a single neural function evaluation (1-NFE).

4.2 Velocity Transport Network Parameterization

Temporal Conditioning and Adaptive Modulation. We parameterize the average-velocity field $u_\theta(z_t, r, t, x^\tau)$ with a conditional isotropic Swin-style diffusion transformer. We first concatenate the noisy state z_t and the conditioning field x^τ, patchify the corresponding tensor, and add learned absolute positional embeddings P_{abs} to encode fixed geographic structure. Further, we sample the source and destination flow timesteps $(r, t) \in [0, 1]$ from a logit-normal distribution. By the way, for a fixed fraction of samples, we set $r = t$ to expose the model to the boundary case that recovers standard flow matching. To condition the transport field on both the current flow time t and the destination time r, we encode them separately with sinusoidal timestep embeddings followed by two independent MLPs, and sum the resulting embeddings. This temporal representation modulates each transformer block through Adaptive RMSNorm [23]:

$$\begin{aligned} t_{\mathrm{emb}} = f_t(\phi(t)) + f_r(\phi(r)), \quad [\alpha^{(l)}, \beta^{(l)}, \gamma^{(l)}] = W^{(l)}_{\mathrm{mod}} t_{\mathrm{emb}} + b^{(l)}_{\mathrm{mod}}, \\ \mathcal{M}^{(l)}(z) = \mathrm{RMSNorm}(z) \odot \left(1 + \alpha^{(l)}\right) + \beta^{(l)}. \end{aligned} \tag{11}$$

Here $\boldsymbol{\gamma}^{(l)}$ acts as a channel-wise residual gate. The modulation layers are zero-initialized, making each residual branch identity-like at initialization and stabilizing early-stage training.

Isotropic Swin Transformer. Following the Swin Transformer paradigm, self-attention is constrained within local windows of size $W_h \times W_w$, reducing the computational complexity from $\mathcal{O}(N^2)$ to $\mathcal{O}(N \cdot W_h W_w)$. To capture the relative movements of advective flows and vortices, we apply 2D Axial Rotary Positional Embeddings (RoPE) [20] to the queries Q and keys K along both X and Y spatial axes inside the local window prior to the dot-product attention calculation.

To reconstruct global atmospheric teleconnections without destroying the isotropic resolution, we alternate between regular and shifted window partitioning across consecutive layers. Let $\mathcal{S}$ denote the cyclic shift operator applied at odd layers to cross window boundaries, yielding the pre-aligned feature $\hat{z}^{(l)} = \mathcal{S}(z^{(l-1)})$. Integrating these mechanisms, the full forward pass for the l-th *IsoSwin Block* is formulated as a gated dynamical system:

$$\begin{aligned} \tilde{z}^{(l)} &= \hat{z}^{(l)} + \boldsymbol{\gamma}^{(l)}_{\mathrm{att}} \odot \mathrm{WAttn}_{\mathrm{RoPE}}\left(\mathcal{M}^{(l)}_{\mathrm{att}}(\hat{z}^{(l)})\right), \\ z^{(l)} &= \mathcal{S}^{-1}\Big(\tilde{z}^{(l)} + \boldsymbol{\gamma}^{(l)}_{\mathrm{ffn}} \odot \mathrm{SwiGLU}\left(\mathcal{M}^{(l)}_{\mathrm{ffn}}(\tilde{z}^{(l)})\right)\Big), \end{aligned} \tag{12}$$

where $\mathcal{S}^{-1}$ reverses the cyclic shift to align the spatial grid for the next layer, and SwiGLU [43] acts as the channel-mixing feed-forward network. After the final block, a lightweight output head linearly maps the latent tokens back to patch values and unpatchifies them to the physical grid, yielding the predicted transport field $u_\theta(z_t, r, t, x^\tau)$. Detailed layer-wise formulations and implementation specifics are deferred to the Appendix B.1.

4.3 Two-Stage Training Objective and Ensemble Inference

⋆ ***Stage I: One-step transport pretraining.*** We first pretrain the transport network to learn a one-step average-velocity field. For each training pair $(x^\tau, x^{\tau+1})$, we sample Gaussian noise $\epsilon \sim \mathcal{N}(0, I)$ and flow-time pair $r, t \in [0, 1]$. Then we utilize the $\mathcal{L}_{\mathrm{ave}}(\theta)$ and the corresponding Jacobian-vector product to supervise the direct average-velocity transport update $u_\theta(\epsilon, 0, 1, x^\tau)$.

⋆ ***Stage II: Curriculum probabilistic calibration.*** Although Stage I learns accurate local transport, recursive weather forecasting additionally requires calibrated uncertainty under autoregressive rollout. We therefore finetune the pretrained model using short ensemble rollouts and gradually increase the rollout horizon during training. At curriculum stage m, let R_m be the current rollout length. Starting from $c_0 = x^\tau$, each ensemble member is generated recursively by:

$$\hat{x}^{\tau+l}_k = \epsilon^l_k - u_\theta\left(\epsilon^l_k, 0, 1, c^{l-1}_k\right), \quad \epsilon^l_k \sim \mathcal{N}(0, I), \quad k = 1, \ldots, K, \tag{13}$$

where c_k^{l-1} is the conditioning state formed from the previous prediction, and K represents the number of ensemble members. Given the ensemble $\{\hat{x}_k^{\tau+l}\}_{k=1}^K$ and the ground-truth state $x^{\tau+l}$ at lead time l, we optimize the empirical Continuous Ranked Probability Score (CRPS):

$$\mathcal{L}_{\text{CRPS}}(l) = \frac{1}{K}\sum_{k=1}^{K}\left\|\hat{x}_k^{\tau+l} - x^{\tau+l}\right\|_1 - \frac{1}{2K(K-1)}\sum_{k=1}^{K}\sum_{k'=1}^{K}\left\|\hat{x}_k^{\tau+l} - \hat{x}_{k'}^{\tau+l}\right\|_1. \tag{14}$$

The first term encourages ensemble accuracy, while the second term discourages ensemble collapse by rewarding appropriate dispersion. Rather than supervising every intermediate rollout step, we evaluate the calibration loss only on the terminal forecast state of the rollout. In our experimental setting, we do not perform many rounds of iterative finetuning. We set $m = 2$, with the corresponding rollout horizons R_m set to 1 and 2, respectively. Further details are provided in the experiments.

⋆ ***1-NFE Ensemble Inference.*** To generate a K-member forecast ensemble, we sample K independent noise instances, set the current state to pure noise ($t = 1$), and the destination to the target forecast ($r = 0$). The future ensemble is generated via a single function evaluation (1-NFE):

$$\hat{x}_k^{\tau+l} = \epsilon_k^l - u_\theta\left(\epsilon_k^l, 0, 1, x_k^{\tau+l-1}\right), \quad l = 1, \ldots, H, \quad k = 1, \ldots, K. \tag{15}$$

Naturally, all K ensemble members for the next time step can be generated in parallel on a single GPU with a single batched inference pass. This circumvents the computationally prohibitive multi-step ODE solvers, reducing the generation time of high-fidelity weather ensembles from minutes to milliseconds, while maintaining physically consistent spatial spread.

5 Experiment

5.1 Experimental Settings

Datasets. We train and evaluate `Tyche` on the ERA5 reanalysis dataset [21] at 1.5° resolution (120 × 240 grid) and 6-hourly interval. `Tyche` is trained to forecast nine variables: four surface-level variable (`u10`, `v10`, `t2m`, `mslp`) and five atmospheric variables (`z`, `q`, `t`, `u`, `v`) at 13 pressure levels (`50`, `100`, `150`, `200`, `250`, `300`, `400`, `500`, `600`, `700`, `850`, `925`, `1000`). Totally, the data contains 69 channels. Before training, all variables are standardized by subtracting their respective means and dividing by their standard deviations. Specifically, we utilize data from 1979–2017 for training, 2018-2019 for validation, and 2020 for testing. More details are illustrated in Appendix A.1.

Baselines. To comprehensively evaluate `Tyche`, we conduct experiments against two main categories of state-of-the-art baselines: deterministic approaches and probabilistic approaches. ✣ For deterministic baselines, we choose Pangu [4] and GraphCast [27]. These two are based on advanced Transformer or GNN-style architectures. ✣ For probabilistic baselines, we select ARCI [3], Swift [45], ArchesWeatherGen (ArchesGen) [10], GenCast [39], and NeuralGCM-ENS [25]. All these approaches are combination of neural networks and Diffusion models. Fur better evaluate, we also compare with ECMWF IFS ensemble (IFS-ENS) [40]. See Appendix A.2 for detailed descriptions.

Implementation details. For `Tyche`, training pipeline is divided into two stages. In Stage I, the model is trained using the MSE loss over 300 epochs. We used the AdamW optimizer with an initial learning rate of 10^{-4}, scheduled by a cosine annealing policy with a minimum learning rate of 10^{-6}. In Stage II, we perform curriculum autoregressive training using the empirical CRPS loss with an ensemble size of 2. The rollout horizon progressively increases from 1 to 2. The learning rate is reduced to 10^{-5} for this stage. The architecture backbone consists of 12 transformer blocks, configured with a hidden dimension of 1024 and 12 attention heads. Spatial interactions are restricted to local windows of size 10×10, which are cyclically shifted by 5×5 in alternating blocks. In our setting, it has around 230 MB trainable parameters. For compared baselines except GenCast, NeuralGCM-ens, and IFS-ENS, we utilize the official implementation for all models and train from scratch to ensure relative fairness. Specifically, for other three, we download the saved inference data of corresponding year with certain dimensions from Weatherbench2. The ensemble number is as the same of other baselines. To ensure computational efficiency, we conducted all training on eight 80GB NVIDIA H100 GPUs, and inference on a single H100 GPU. More details see in Apendix A.2.

Evaluation. We aim to generate 20 members for each compared model. Specifically, for deterministic approaches, we add independent noise to initial state; and for probabilistic approaches, we directly

Table 1: We evaluate the performance of our **Tyche** again five baselines (two deterministic baselines and three probabilistic baselines) up to 10 days (40 iteration). We report the RMSE results of three variables: u10, t2m, and mslp, and a small RMSE (↓) identify better performance.

Model	6-Hour			1-Day			3-Day			7-Day			10-Day		
	u10	t2m	mslp	u10	t2m	mslp	u10	t2m	mslp	u10	t2m	mslp	u10	t2m	mslp
Pangu	0.58	0.92	0.64	1.14	1.50	1.47	2.53	1.90	3.62	4.05	3.23	7.02	4.85	4.05	10.29
GraphCast	**0.48**	**0.75**	0.53	**0.96**	1.24	1.09	2.21	2.28	2.86	3.97	3.62	6.64	4.94	4.50	10.28
ARCI	0.89	1.41	0.95	1.45	2.40	1.63	2.75	3.55	3.50	5.32	5.23	8.47	5.92	6.37	11.26
Swift	0.85	1.32	0.88	1.46	2.32	1.56	2.81	3.32	3.26	5.14	5.11	7.54	5.24	5.58	10.49
ArchesGen	0.62	0.82	0.57	1.11	1.25	1.27	2.15	1.88	2.71	4.25	3.35	6.84	4.88	4.17	9.91
Tyche (Our)	0.51	0.76	**0.49**	1.09	**1.19**	**0.93**	**2.13**	**1.68**	**2.49**	**3.89**	**2.77**	**6.03**	**4.71**	**3.74**	**8.58**

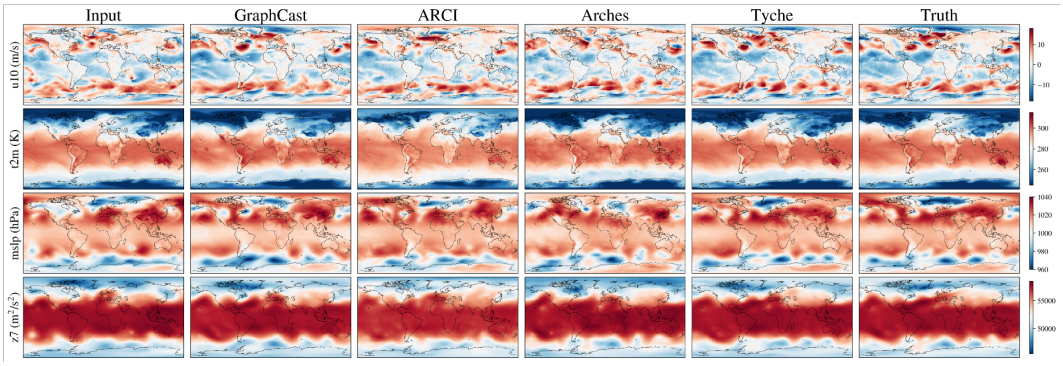


Figure 3: Comparison of 7-day forecasts for four variables: u10, t2m, mslp, and z7.

generate 20 members parallelly. For evaluation, we report metrics like latitude-weighted root mean square error (RMSE) and continuous ranked probability score (CRPS). Also, we report spread to skill ratio (SSR) to quantify the uncertainty capability. More details are demonstrated in Appendix A.3.

5.2 Main Results

Forecasting Performance. To evaluate the effectiveness of our method, we first compare **Tyche** with two deterministic baselines and three probabilistic baselines. Both of them are trained from scratch. Regarding the number of sampling steps during inference, we use corresponding multi-steps sampling for ARCI and ArchesGen. While for Swift and our method, which are designed for one-step generation, we use single-step sampling. Here, we do not use the ensemble mean of probabilistic models for comparison, we instead select only one inference member. Table 1 reports the latitude-weighted RMSE of **Tyche** and five baselines from 6 hours to 10 days on three representative surface variables, namely u10, t2m, and mslp. Overall, **Tyche** consistently achieves the best or second-best performance across almost all variables and forecast horizons. We observe that within the first day of forecasting, GraphCast achieves the best performance across multiple variables, while Pangu also delivers competitive results. This suggests that deterministic models retain a clear advantage at short lead times, where error accumulation remains limited. At 3 days, **Tyche** achieves the lowest RMSE on all three variables. Compared with GraphCast, **Tyche** reduces the 3-day RMSE from 2.21 to 2.13 on u10, from 2.28 to 1.68 on t2m, and from 2.86 to 2.49 on mslp. Typically, when the forecast horizon extending 3-day, **Tyche** still maintains the best performance across all three variables.

Figure 3 provides a qualitative comparison of 7-day forecasts for u10, t2m, mslp, and z7, where z7 represents geopotential at 500 hPa. The visual results are consistent with the quantitative findings in Table 1. Compared with other baselines, **Tyche** better preserves large-scale spatial structures and produces forecast fields that are visually closer to the ground truth. For u10, **Tyche** captures the major wind patterns in the mid-latitudes and Southern Ocean with less over-smoothing. For t2m, it maintains coherent meridional temperature gradients and more realistic regional variations. For mslp, **Tyche** better reconstructs large-scale pressure systems. For z7, **Tyche** also preserves the global geopotential structure more faithfully, indicating improved stability for upper-air variables. In summary, the quantitative and qualitative results jointly demonstrate the effectiveness of **Tyche**.

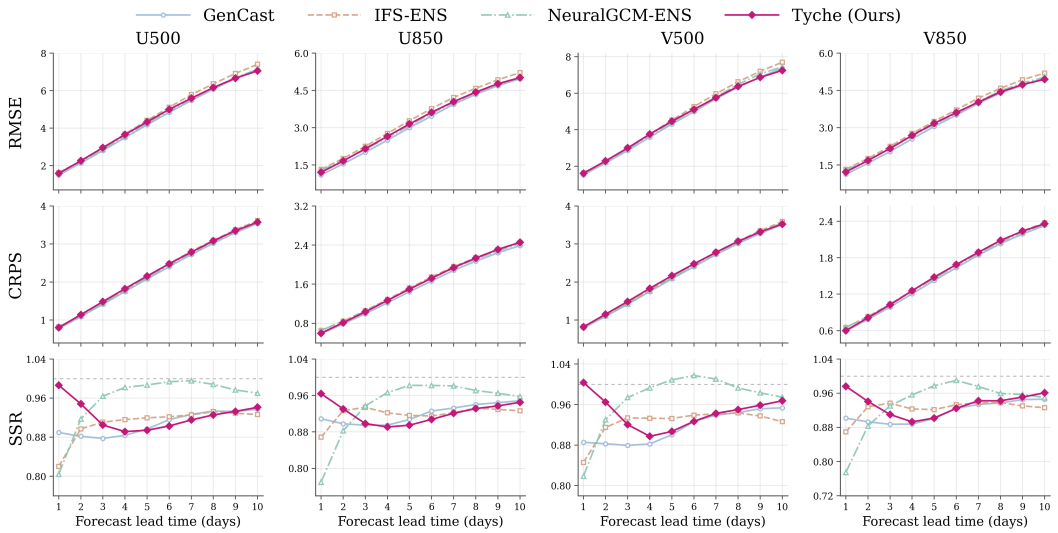


Figure 4: Ensemble Skill Curves Across Forecast Lead Time. We report the latitude-weighted RMSE (↓), latitude-weighted CRPS (↓), and SSR (→ 1) of several upper-air variables.

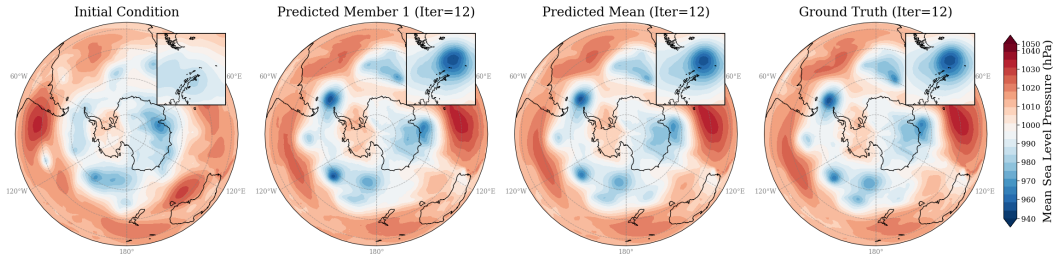


Figure 5: Visualization of 3-day MSLP ensemble forecast performance over the Southern Ocean.

Ensemble Performance. Further, we evaluate the probabilistic forecasting skills of `Tyche` by comparing it with state-of-the-art ensemble forecasting systems, including GenCast, NeuralGCM-ENS, and IFS-ENS. To ensure a fair comparison, we use 20 ensemble members for both `Tyche` and all compared ensemble baselines. Figure 4 reports the latitude-weighted RMSE, CRPS, and SSR over a 10-day forecast horizon on key upper-air variables, including `U500`, `U850`, `V500`, and `V850`. Unlike the single-member comparison in Table 1, this evaluation focuses on the performance of the ensemble skills. As illustrated in Figure 4, the curves of `Tyche` follow a similar growth trend to established ensemble baselines, indicating that the proposed one-step probabilistic generator does not suffer from rapid error divergence during autoregressive rollout. Specifically, `Tyche` demonstrates competitive performance against compared systems in terms of RMSE. Meanwhile, the CRPS curves show that `Tyche` produces meaningful ensemble distributions rather than merely accurate individual samples, demonstrating its ability to capture forecast uncertainty over medium-range lead times. Additionally, the SSR curves rigorously validate the model's probabilistic calibration. Following an initial short-term adjustment, the SSR of `Tyche` converges tightly toward the optimal value of 1. As known, GenCast follows a conventional diffusion-based generative paradigm and requires more than 20 denoising steps during inference, while NeuralGCM-ENS incorporates physics-based numerical solvers into the forecasting process. Meanwhile, `Tyche` is a purely data-driven probabilistic model and performs only single-step sampling at inference time. Beyond aggregated metrics, `Tyche` demonstrates exceptional physical fidelity in capturing high-frequency, synoptic-scale structures. As visualized in Figure 5, we examine the Mean Sea Level Pressure (*MSLP*) over the dynamically volatile Southern Ocean at a 3-day lead time (*Iteration=12*). An individual forecast (*Predicted Member 1*) exhibits physically realistic vortex structures that strictly adhere to the ground truth. And the predicted mean displays the characteristic smoothing expected from a well-calibrated ensemble, without the appearance of blur. This demonstrates the superior ensemble performance of our method.

Efficiency Analysis. The primary bottleneck of modern probabilistic weather forecasting lies in the inference computational cost, which strictly compounds along four axes: the forecast rollout length (L); the ensemble size (K); the number of function evaluations (N_{NFE}) per rollout; and the inference cost (M) per NFE. Consequently, the total inference cost scales as $\mathcal{O}(L \times K \times N_{\text{NFE}} \times M)$. Multi-step diffusion models (*e.g.*, GenCast, ArchesWeatherGen) typically require $N_{\text{NFE}} \geq 10$ per rollout, making the generation of multi-day high-fidelity weather ensembles take minutes and rendering large-scale operational deployment prohibitively expensive. `Tyche` fundamentally breaks this computational bottleneck through its average velocity transport formulation. As detailed in Table 2, `Tyche` can parallelly generate a 20-member, 15-day trajectory (60 autoregressive iteration) in merely $\sim$ 62 seconds. This achieves an order-of-magnitude acceleration, running approximately 12× to 43× faster than current probabilistic MLWP systems (with the same resolution of data). Thus, `Tyche` matches the inference speeds of deterministic methods, while preserving the robust diversity and calibration typically restricted to costly multi-step diffusion and operational NWP systems.

Table 2: Computational efficiency and inference comparison.

Model	Methodology	NFE per Rollout	Inference Latency
IFS-ENS	Operational NWP	N/A (Physics)	Hours
GenCast	EDM	20	$\sim$ 45 mins
ARCI	EDM	20	$\sim$ 12 mins
ArchesWeatherGen	Flow Matching	25	$\sim$ 36 mins
Tyche (Our)	**Average Transport**	**1**	**$\sim$ 62 secs**

Typhoon Tracking. We further evaluate `Tyche`'s capability in predicting high-impact synoptic-scale events through tropical cyclone tracking. Figure 6 compares the trajectories for Typhoon Ma-on (2022/08) and Ernesto (2024/08). Crucially, the individual ensemble members of `Tyche` (thin grey lines) form a well-calibrated cone of uncertainty. And the ensemble average trajectory fits the best trajectory very well. For the westward-moving Ma-on, the ensemble mean of `Tyche` (thick red line) tightly adheres to the ground truth, accurately reproducing the landfall direction. For the northeastward-curving Ernesto, `Tyche` successfully captures the complex recurvature dynamics, aligning perfectly with the late-stage evolution where IFS-HRES exhibits a noticeable westward cross-track error. These findings suggest that the proposed one-step transport effectively preserves coherent vortex motion, offering robust and scalable probabilistic forecasts for extreme weather phenomena.

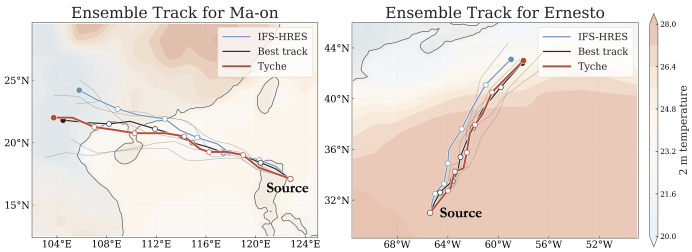


Figure 6: Typhoon ensemble track for Ma-on and Ernesto.

Ablation on Generative Transport Strategy. We further investigate the effectiveness of generative transport strategy. Specifically, we conduct an ablation study comparing `Tyche` against three alternative state-of-the-art generative transport strategies: ✤ ***EDM*** (*a unified diffusion model*) [24], ✤ ***Triglow*** (*a customized v-prediction flow matching*) [32], and ✤ ***sCM*** (*the continuous-time consistency model*) [32]. To isolate the effect of the generative transport strategy, we conduct all ablation experiments under a unified model architecture and vary only the diffusion or flow training strategy. For each compared strategy, we follow the hyperparameter settings recommended in the original paper. We evaluate each method under three sampling budgets, namely 1-NFE, 3-NFE, and 10-NFE, and report RMSE within two autoregressive horizons (the 1-step and 10-step rollouts mean), as shown in Table 3. The results reveal the efficiency tradeoff for conventional multi-step generative samplers. Both TrigFlow and EDM benefit from increasing the number of function evaluations. By contrast, sCM supports fast one-step generation, but its rollout error remains high. In contrast, `Tyche` achieves strong performance with only a single function evaluation. At 1-NFE, `Tyche` obtains 0.103 RMSE at Step-1 and 0.242 RMSE at Step-10, matching the 10-NFE EDM result while using only

Table 3: Ablation study on generative transport strategy.

Method	1-NFE		3-NFE		10-NFE	
	Step-1	Step-10	Step-1	Step-10	Step-1	Step-10
w/ EDM	–	–	0.145	0.428	0.108	0.245
w/ TrigFlow	–	–	0.134	0.390	<u>0.101</u>	0.250
w/ sCM	0.154	0.462	0.151	0.454	–	–
Ours	0.103	<u>0.242</u>	**0.099**	**0.237**	–	–

one tenth of the sampling budget. With 3-NFE inference, `Tyche` further improves the Step-1 and Step-10 RMSE to 0.099 and 0.237, achieving the best performance among all compared settings.

6 Conclusion & Limitation

In this paper, we introduced Tyche, a fundamentally new generative framework that breaks the computational bottleneck of multi-step diffusion and flow models in global probabilistic weather forecasting. Tyche achieves 1-NFE generation by learning a destination-aware average-velocity flow. Further, we effectively mitigated the theoretically bounded autoregressive error accumulation via a curriculum-based CRPS finetuning strategy. Experiments show that Tyche matches or surpasses the state-of-the-art multi-step generative methods. As a current limitation, the highly efficient inference of Tyche requires a relatively high GPU memory overhead, leaving room for further research.

Acknowledgement

This work was supported by ShenzhenLoop Area Institute (No. FP202602).

References

[1] Ferran Alet, Ilan Price, Andrew El-Kadi, Dominic Masters, Stratis Markou, Tom R Andersson, Jacklynn Stott, Remi Lam, Matthew Willson, Alvaro Sanchez-Gonzalez, et al. Skillful joint probabilistic weather forecasting from marginals. *arXiv preprint arXiv:2506.10772*, 2025.

[2] Anna Allen, Stratis Markou, Will Tebbutt, James Requeima, Wessel P Bruinsma, Tom R Andersson, Michael Herzog, Nicholas D Lane, Matthew Chantry, J Scott Hosking, et al. End-to-end data-driven weather prediction. *Nature*, 641(8065):1172–1179, 2025.

[3] Martin Andrae, Tomas Landelius, Joel Oskarsson, and Fredrik Lindsten. Continuous ensemble weather forecasting with diffusion models. In *The Thirteenth International Conference on Learning Representations*, 2025.

[4] Kaifeng Bi, Lingxi Xie, Hengheng Zhang, Xin Chen, Xiaotao Gu, and Qi Tian. Accurate medium-range global weather forecasting with 3d neural networks. *Nature*, 619(7970):533–538, 2023.

[5] Cristian Bodnar, Wessel P Bruinsma, Ana Lucic, Megan Stanley, Anna Allen, Johannes Brandstetter, Patrick Garvan, Maik Riechert, Jonathan A Weyn, Haiyu Dong, et al. A foundation model for the earth system. *Nature*, 641(8065):1180–1187, 2025.

[6] Jerald A Brotzge, Don Berchoff, DaNa L Carlis, Frederick H Carr, Rachel Hogan Carr, Jordan J Gerth, Brian D Gross, Thomas M Hamill, Sue Ellen Haupt, Neil Jacobs, et al. Challenges and opportunities in numerical weather prediction. *Bulletin of the American Meteorological Society*, 104(3):E698–E705, 2023.

[7] Salva Rühling Cachay, Miika Aittala, Karsten Kreis, Noah Brenowitz, Arash Vahdat, Morteza Mardani, and Rose Yu. Elucidated rolling diffusion models for probabilistic forecasting of complex dynamics. *arXiv preprint arXiv:2506.20024*, 2025.

[8] Lei Chen, Xiaohui Zhong, Feng Zhang, Yuan Cheng, Yinghui Xu, Yuan Qi, and Hao Li. Fuxi: A cascade machine learning forecasting system for 15-day global weather forecast. *npj climate and atmospheric science*, 6(1):190, 2023.

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

[10] Guillaume Couairon, Renu Singh, Anastase Charantonis, Christian Lessig, and Claire Monteleoni. Archesweather & archesweathergen: a deterministic and generative model for efficient ml weather forecasting. *arXiv preprint arXiv:2412.12971*, 2024.

[11] Florinel-Alin Croitoru, Vlad Hondru, Radu Tudor Ionescu, and Mubarak Shah. Diffusion models in vision: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 45(9):10850–10869, 2023.

[12] Yingzhe Cui, Ruohan Wu, Xiang Zhang, Ziqi Zhu, Bo Liu, Jun Shi, Junshi Chen, Hailong Liu, Shenghui Zhou, Liang Su, et al. Forecasting the eddying ocean with a deep neural network. *Nature Communications*, 16(1):2268, 2025.

[13] Kevin Frans, Danijar Hafner, Sergey Levine, and Pieter Abbeel. One step diffusion via shortcut models. *arXiv preprint arXiv:2410.12557*, 2024.

[14] Yuan Gao, Hao Wu, Ruiqi Shu, huanshuo dong, Fan Xu, Rui Ray Chen, Yibo Yan, Qingsong Wen, Xuming Hu, Kun Wang, Jiahao Wu, Li Qing, Hui Xiong, and Xiaomeng Huang. Oneforecast: A universal framework for global and regional weather forecasting. In *Forty-second International Conference on Machine Learning*, 2025.

[15] Yuan Gao, Hao Wu, Fan Xu, Yanfei Xiang, Ruijian Gou, Ruiqi Shu, Qingsong Wen, Xian Wu, Kun Wang, and Xiaomeng Huang. Neuralom: Neural ocean model for subseasonal-to-seasonal simulation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pages 14756–14764, 2026.

[16] Itai Gat, Tal Remez, Neta Shaul, Felix Kreuk, Ricky TQ Chen, Gabriel Synnaeve, Yossi Adi, and Yaron Lipman. Discrete flow matching. *Advances in Neural Information Processing Systems*, 37:133345–133385, 2024.

[17] Zhengyang Geng, Mingyang Deng, Xingjian Bai, J Zico Kolter, and Kaiming He. Mean flows for one-step generative modeling. *arXiv preprint arXiv:2505.13447*, 2025.

[18] Zhengyang Geng, Yiyang Lu, Zongze Wu, Eli Shechtman, J Zico Kolter, and Kaiming He. Improved mean flows: On the challenges of fastforward generative models. *arXiv preprint arXiv:2512.02012*, 2025.

[19] Aditya Grover, Ashish Kapoor, and Eric Horvitz. A deep hybrid model for weather forecasting. In *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*, pages 379–386, 2015.

[20] Byeongho Heo, Song Park, Dongyoon Han, and Sangdoo Yun. Rotary position embedding for vision transformer. In *European Conference on Computer Vision*, pages 289–305. Springer, 2024.

[21] Hans Hersbach, Bill Bell, Paul Berrisford, Shoji Hirahara, András Horányi, Joaquín Muñoz-Sabater, Julien Nicolas, Carole Peubey, Raluca Radu, Dinand Schepers, et al. The era5 global reanalysis. *Quarterly journal of the royal meteorological society*, 146(730):1999–2049, 2020.

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

[23] Zixuan Jiang, Jiaqi Gu, Hanqing Zhu, and David Pan. Pre-rmsnorm and pre-crmsnorm transformers: equivalent and efficient pre-ln transformers. *Advances in Neural Information Processing Systems*, 36:45777–45793, 2023.

[24] Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. *Advances in neural information processing systems*, 35:26565–26577, 2022.

[25] Dmitrii Kochkov, Janni Yuval, Ian Langmore, Peter Norgaard, Jamie Smith, Griffin Mooers, Milan Klöwer, James Lottes, Stephan Rasp, Peter Düben, et al. Neural general circulation models for weather and climate. *Nature*, 632(8027):1060–1066, 2024.

[26] Nikita Kornilov, Petr Mokrov, Alexander Gasnikov, and Alexander Korotin. Optimal flow matching: Learning straight trajectories in just one step. *Advances in Neural Information Processing Systems*, 37:104180–104204, 2024.

[27] Remi Lam, Alvaro Sanchez-Gonzalez, Matthew Willson, Peter Wirnsberger, Meire Fortunato, Ferran Alet, Suman Ravuri, Timo Ewalds, Zach Eaton-Rosen, Weihua Hu, et al. Learning skillful medium-range global weather forecasting. *Science*, 382(6677):1416–1421, 2023.

[28] Simon Lang, Mihai Alexe, Mariana CA Clare, Christopher Roberts, Rilwan Adewoyin, Zied Ben Bouallègue, Matthew Chantry, Jesper Dramsch, Peter D Dueben, Sara Hahner, et al. Aifs-crps: ensemble forecasting using a model trained with a loss function based on the continuous ranked probability score. *npj Artificial Intelligence*, 2(1):18, 2026.

[29] Qi Li, Zhenyu Zhang, Lei Yao, Zhaoxia Li, Tianyi Zhong, and Yong Zhang. Diffusion-based decoupled deterministic and uncertain framework for probabilistic multivariate time series forecasting. In *The Thirteenth International Conference on Learning Representations*, 2025.

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

[31] Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. *arXiv preprint arXiv:2209.03003*, 2022.

[32] Cheng Lu and Yang Song. Simplifying, stabilizing and scaling continuous-time consistency models. *arXiv preprint arXiv:2410.11081*, 2024.

[33] Lauren E McPhillips, Heejun Chang, Mikhail V Chester, Yaella Depietri, Erin Friedman, Nancy B Grimm, John S Kominoski, Timon McPhearson, Pablo Méndez-Lázaro, Emma J Rosi, et al. Defining extreme events: A cross-disciplinary review. *Earth's Future*, 6(3):441–455, 2018.

[34] Rajasekaran Meenal, D Binu, KC Ramya, Prawin Angel Michael, K Vinoth Kumar, E Rajasekaran, and B Sangeetha. Weather forecasting for renewable energy system: a review. *Archives of Computational Methods in Engineering*, 29(5):2875–2891, 2022.

[35] Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *International conference on machine learning*, pages 8162–8171. PMLR, 2021.

[36] Thomas N Nipen, Ivar A Seierstad, Cristian Lussana, Jørn Kristiansen, and Øystein Hov. Adopting citizen observations in operational weather prediction. *Bulletin of the American Meteorological Society*, 101(1):E43–E57, 2020.

[37] Joel Oskarsson, Tomas Landelius, Marc P Deisenroth, and Fredrik Lindsten. Probabilistic weather forecasting with hierarchical graph neural networks. *Advances in Neural Information Processing Systems*, 37:41577–41648, 2024.

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

[39] Ilan Price, Alvaro Sanchez-Gonzalez, Ferran Alet, Tom R Andersson, Andrew El-Kadi, Dominic Masters, Timo Ewalds, Jacklynn Stott, Shakir Mohamed, Peter Battaglia, et al. Probabilistic weather forecasting with machine learning. *Nature*, 637(8044):84–90, 2025.

[40] Stephan Rasp, Stephan Hoyer, Alexander Merose, Ian Langmore, Peter Battaglia, Tyler Russell, Alvaro Sanchez-Gonzalez, Vivian Yang, Rob Carver, Shreya Agrawal, et al. Weatherbench 2: A benchmark for the next generation of data-driven global weather models. *Journal of Advances in Modeling Earth Systems*, 16(6):e2023MS004019, 2024.

[41] Colin Raymond, Radley M Horton, Jakob Zscheischler, Olivia Martius, Amir AghaKouchak, Jennifer Balch, Steven G Bowen, Suzana J Camargo, Jeremy Hess, Kai Kornhuber, et al. Understanding and managing connected extreme events. *Nature climate change*, 10(7):611–621, 2020.

[42] Xiaoli Ren, Xiaoyong Li, Kaijun Ren, Junqiang Song, Zichen Xu, Kefeng Deng, and Xiang Wang. Deep learning-based weather prediction: a survey. *Big Data Research*, 23:100178, 2021.

[43] Noam Shazeer. Glu variants improve transformer. *arXiv preprint arXiv:2002.05202*, 2020.

[44] Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. *arXiv preprint arXiv:2303.01469*, 2023.

[45] Jason Stock, Troy Arcomano, and Rao Kotamarthi. Swift: An autoregressive consistency model for efficient weather forecasting. *arXiv preprint arXiv:2509.25631*, 2025.

[46] A Utku and U Can. An efficient hybrid weather prediction model based on deep learning. *International Journal of Environmental Science and Technology*, 20(10):11107–11120, 2023.

[47] Mengze Wang, Benedikt Barthel Sorensen, and Themistoklis Sapsis. Gen2: A generative prediction-correction framework for long-time emulations of spatially-resolved climate extremes. *arXiv preprint arXiv:2508.15196*, 2025.

[48] Jonas Wildberger, Maximilian Dax, Simon Buchholz, Stephen Green, Jakob H Macke, and Bernhard Schölkopf. Flow matching for scalable simulation-based inference. *Advances in Neural Information Processing Systems*, 36:16837–16864, 2023.

[49] Hao Wu, Yuan Gao, Ruijian Gou, Xian Wu, Chuhan Wu, Huahui Yi, Johannes Brandstetter, Fan Xu, Kun Wang, Penghao Zhao, et al. Advanced long-term earth system forecasting. *arXiv preprint arXiv:2505.19432*, 2025.

[50] Hao Wu, Fan Xu, Yuxu Lu, Penghao Zhao, Fan Zhang, Hao Jia, Yuxuan Liang, Ruijian Gou, Qingsong Wen, Xian Wu, Xiaomeng Huang, and Yuan Gao. Pnp-corrector: A universal correction framework for coupled spatiotemporal forecasting. In *Forty-third International Conference on Machine Learning*, 2026.

[51] Fan Xu, Wei Gong, Hao Wu, Lilan Peng, Nan Wang, Qingsong Wen, Xian Wu, Kun Wang, and Xibin Zhao. Advanced global wildfire activity modeling with hierarchical graph ode. In *Proceedings of the 32nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 1*, pages 1662–1672, 2026.

[52] Huijie Zhang, Aliaksandr Siarohin, Willi Menapace, Michael Vasilkovsky, Sergey Tulyakov, Qing Qu, and Ivan Skorokhodov. Alphaflow: Understanding and improving meanflow models. *arXiv preprint arXiv:2510.20771*, 2025.

[53] Yilin Zhuang and Karthik Duraisamy. Ladcast: A latent diffusion model for medium-range ensemble weather forecasting. *arXiv preprint arXiv:2506.09193*, 2025.

Appendix Contents

A **Detailed Experimental Settings** 15
A.1 Data Description 15
A.2 Details of Baselines and Implementations 15
A.2.1 Deterministic Baselines 15
A.2.2 Probabilistic and Generative Baselines 15
A.2.3 Operational NWP Baseline 16
A.2.4 Implementation Details 16
A.3 Evaluation Metrics 17
B **Details of `Tyche`** 18
B.1 Details of Network Architecture 18
B.2 Pseudocode of `Tyche` 19
C **Theoretical Analysis and Proofs** 19
C.1 The Discretization Error Perspective for One-Step Transport 19
C.2 Proof of Theorem 20
C.3 Bridging Theory and Practice: The Rationale for Stage II Design 22
D **Extended Experiments** 23
D.1 Ablation study 23
D.2 Comparison on more variables 23
D.3 Ensemble skills comparison 24
D.4 Detailed visualizations on more horizons and variables 24

A Detailed Experimental Settings

A.1 Data Description

We use a collection of upper-air and surface-level meteorological variables as both model inputs and prediction targets. The full list of variables is summarized in Table 4. For pressure-level fields, we include five atmospheric variables: Geopotential (z), Specific humidity (q), Temperature (t), U component of wind (u), and V component of wind (v). These variables are defined on 13 standard pressure levels, namely 50, 100, 150, 200, 250, 300, 400, 500, 600, 700, 850, 925, and 1000 hPa. In addition, we use four single-level variables, including 10 metre u wind component (u10), 10 metre v wind component (v10), 2 metre temperature (t2m), and Mean sea level pressure (mslp).

Before training, each variable is standardized using statistics computed from the training period (1979-2017). The same normalization parameters are then kept fixed for all experiments to ensure consistency across different settings. All data are processed on a 1.5° latitude–longitude grid. Following the common preprocessing strategy for this grid resolution, the southernmost latitude row is discarded from the original 1.5° grid (121×240) to make the latitude dimension even.

Table 4: Description of data variables used.

Type	Variable name	Short name	Param ID
Atmospheric	Geopotential	`z`	129
Atmospheric	Specific humidity	`q`	133
Atmospheric	Temperature	`t`	130
Atmospheric	U component of wind	`u`	131
Atmospheric	V component of wind	`v`	132
Single	10 metre u wind component	`u10`	165
Single	10 metre v wind component	`v10`	166
Single	2 metre temperature	`t2m`	167
Single	Mean sea level pressure	`mslp`	151

A.2 Details of Baselines and Implementations

To provide a comprehensive evaluation of **Tyche**, we compare it against a robust suite of state-of-the-art weather forecasting systems. These baselines are broadly categorized into purely deterministic ML models, modern generative/probabilistic ML models, and traditional operational numerical weather prediction (NWP).

A.2.1 Deterministic Baselines

These models aim to minimize the mean-squared error (MSE) of the forecast but are known to oversmooth predictions at longer lead times, lacking the capability to capture the full distribution of weather uncertainties.

- **Pangu-Weather [4]:** A highly influential deterministic MLWP model utilizing a 3D Earth-Specific Transformer. It processes data using height-specific attention mechanisms and trains independent networks for specific lead times (*e.g.*, 1h, 3h, 6h, 24h) to circumvent the error accumulation typical of high-frequency autoregressive rollouts.
- **GraphCast [27]:** A state-of-the-art deterministic model based on Graph Neural Networks (GNNs). It maps the latitude-longitude grid to a multi-scale icosahedral mesh using an encode-process-decode architecture, typically generating forecasts autoregressively at 6-hour intervals.

A.2.2 Probabilistic and Generative Baselines

These models represent the forefront of global AI weather ensemble forecasting (2024–2025), aiming to produce sharp, well-calibrated ensemble members that can capture extreme events and uncertainty growth.

- **GenCast [39]:** A full-space conditional autoregressive diffusion model that serves as the gold-standard ML probabilistic baseline. It learns the one-step conditional distribution and rolls out autoregressively (typically in 12-hour steps). While it achieves exceptional calibration and sharpness matching operational NWP, it relies on multiple denoising evaluations per step, making sampling computationally expensive.
- **ARCI [3]:** A continuous lead-time diffusion model designed for temporal interpolation. Instead of costly fine-grained autoregressive rollouts, ARCI conditions the diffusion directly on lead time and correlates the driving noise (via an Ornstein-Uhlenbeck process) across times. This allows independent marginals to form a temporally consistent trajectory, combining coarse autoregressive steps (*e.g.*, 24h) with cheap high-resolution (*e.g.*, 6h or 1h) interpolation.
- **ArchesWeatherGen [10]:** A computationally efficient, two-stage residual generative framework. It first utilizes a deterministic backbone (ArchesWeather, featuring Cross-Level Attention) to predict the mean state. Subsequently, a flow-matching generative model is trained to predict the normalized residuals between the deterministic output and the ERA5 ground truth. This explicitly decouples mean prediction from distributional correction.
- **Swift [45]:** A highly efficient 1-NFE consistency model. Swift trains a probability-flow/diffusion-style model into a single-step consistency model using trigflow. A key innovation is its autoregressive multi-step fine-tuning utilizing the Continuous Ranked Probability Score. It represents the state-of-the-art for ultra-fast, single-step ensemble generation.
- **NeuralGCM-ENS [25]:** A hybrid approach bridging traditional physics and machine learning. It relies on a differentiable general circulation model (dynamical core) combined with neural networks that parameterize unresolved sub-grid physics. The ensemble variant generates multiple trajectories via initial perturbations, providing a strong, physically-constrained probabilistic baseline.

A.2.3 Operational NWP Baseline

- **IFS-ENS [40]:** The ECMWF Integrated Forecasting System Ensemble. This is the gold-standard operational numerical weather prediction baseline. It relies on solving physical equations explicitly and uses Singular Vectors and Ensemble of Data Assimilations (EDA) to perturb initial conditions, typically generating a 50-member ensemble with plus one unperturbed control.

A.2.4 Implementation Details

In our experiments, `Tyche` is trained on the ERA5 dataset at a spatial resolution of $1.5°$, totally utilizing 69 atmospheric and surface variables. The model is trained on $8 \times$ 80GB NVIDIA H100 GPUs. During inference, an ensemble of 20 members is generated parallelly on a single GPU.

Architecture of `Tyche`. The backbone of `Tyche` is a non-hierarchical Swin Transformer configured with 12 transformer blocks. Each block features a hidden dimension of 1024 and 12 attention heads. Spatial interactions are restricted to local windows of size 10×10, which are cyclically shifted by 5×5 in alternating blocks. To effectively capture spatial dependencies and temporal dynamics, we employ learned absolute positional embeddings combined with 2D axial Rotary Positional Embeddings (RoPE) within the windows. Furthermore, the architecture utilizes pre-RMSNorm paired with adaptive modulation (generating scale, shift, and gating factors conditioned on the time steps) to stabilize the generative learning process. In this configuration, the model contains approximately 230 MB trainable parameters.

Training Pipeline of `Tyche`. The training pipeline is meticulously divided into two stages to decouple deterministic representation learning from probabilistic calibration:

➠ **Stage I (One-step flow pretraining):** The model is trained from scratch to learn the underlying vector field using an MSE-based objective with 300 epochs. We use the AdamW optimizer with a weight decay of $1e^{-4}$ and an initial learning rate of $1e^{-4}$, which is scheduled by a cosine annealing policy decaying to a minimum learning rate of $1e^{-6}$.

➠ **Stage II (Curriculum probabilistic calibration):** To explicitly calibrate the uncertainty and ensure long-rollout stability, we perform curriculum autoregressive training using an empirical CRPS loss. The training utilizes a small ensemble size of 2 to maintain computational tractability. The

autoregressive rollout horizon progressively increases from 1 to 2 steps, with each curriculum stage trained for 15 epochs. The learning rate is reduced and fixed at $1e^{-5}$ for this stage.

Baseline Configurations and Fairness. To ensure a rigorous and fair evaluation, the implementations of our selected baselines are handled as follows:

➠ **Trained from Scratch:** For Pangu, GraphCast, ARCI, Swift, and ArchesWeatherGen, we utilize their official codebases and train them from scratch on the exact same 1.5° ERA5 dataset split as `Tyche`. All training for these baselines is uniformly conducted on eight 80GB NVIDIA H100 GPUs.

➠ **Downloaded Operational/Large-Scale References:** For GenCast, NeuralGCM-ENS, and IFS-ENS, training from scratch is operationally restricted. We directly download their official saved inference data for the corresponding evaluation years and variables from WeatherBench2 [40].

Across all probabilistic methods and operational references, the evaluated ensemble number is strictly aligned to 20 members to ensure fair comparison of generative diversity and calibration metrics.

A.3 Evaluation Metrics

To comprehensively evaluate the performance of our probabilistic forecasts, we employ standard metrics from meteorology and uncertainty quantification, including the Latitude-weighted Root Mean Square Error (RMSE), Ensemble Spread, Spread-to-Skill Ratio (SSR), and Continuous Ranked Probability Score (CRPS).

To ensure fair evaluation across the globe, all spatial aggregations are weighted by latitude. Let M be the total number of grid cells. For a grid cell i at latitude θ_i, the normalized area-based latitude weight w_i is defined as:

$$w_i = \frac{\cos(\theta_i)}{\frac{1}{M}\sum_{j=1}^{M}\cos(\theta_j)}, \tag{16}$$

such that the weights average to 1 over the entire grid.

In the following formulations, let N denote the total number of forecast initialization times (indexed by n), and K denote the ensemble size (indexed by k). Let $y_{i,n}$ represent the ground truth for grid cell i at time n, $x_{i,n}^k$ represent the corresponding prediction of the k-th ensemble member, and $\bar{x}_{i,n} = \frac{1}{K}\sum_{k=1}^{K} x_{i,n}^k$ denote the ensemble mean.

Latitude-weighted RMSE (Skill). RMSE measures the deterministic accuracy of the forecast. For ensemble models, it is calculated based on the ensemble mean:

$$\text{RMSE} = \frac{1}{N}\sum_{n=1}^{N}\sqrt{\frac{1}{M}\sum_{i=1}^{M} w_i \left(y_{i,n} - \bar{x}_{i,n}\right)^2}. \tag{17}$$

Ensemble Spread. Spread quantifies the variability or uncertainty captured within the ensemble members. It is calculated as the root mean square of the ensemble variance:

$$\text{Spread} = \frac{1}{N}\sum_{n=1}^{N}\sqrt{\frac{1}{M}\sum_{i=1}^{M} w_i \left(\frac{1}{K-1}\sum_{k=1}^{K}(x_{i,n}^k - \bar{x}_{i,n})^2\right)}. \tag{18}$$

Spread-to-Skill Ratio (SSR). Ideally, a well-calibrated probabilistic forecast should achieve a balance where the ensemble spread is proportional to the RMSE error of the ensemble mean. The SSR evaluates this reliability, with values close to 1 indicating optimal effective uncertainty estimation:

$$\text{SSR} = \sqrt{\frac{K+1}{K}}\frac{\text{Spread}}{\text{RMSE}}, \tag{19}$$

where $\sqrt{(K+1)/K}$ is a statistical correction factor for finite ensemble sizes.

Continuous Ranked Probability Score (CRPS). CRPS is a widely used metric for probabilistic forecasting that compares the cumulative distribution function (CDF) of the predicted ensemble against the observation. It penalizes errors in both the location and the shape of the predicted distribution. Following the standard in WeatherBench2, we utilize the unbiased estimator of CRPS:

$$\text{CRPS} = \frac{1}{N}\sum_{n=1}^{N}\frac{1}{M}\sum_{i=1}^{M} w_i \left(\frac{1}{K}\sum_{k=1}^{K} |x_{i,n}^{k} - y_{i,n}| - \frac{1}{2K(K-1)}\sum_{k=1}^{K}\sum_{k'=1}^{K} |x_{i,n}^{k} - x_{i,n}^{k'}| \right). \quad (20)$$

B Details of Tyche

B.1 Details of Network Architecture

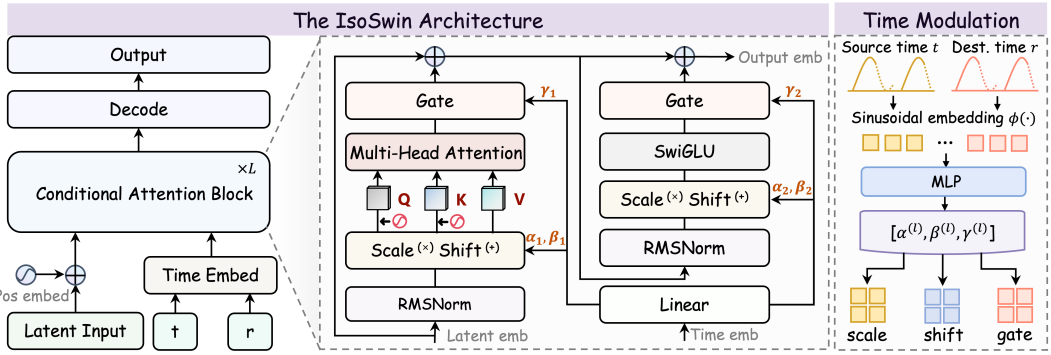


Figure 7: The details of network architecture.

The transport field in Eq. (9) is parameterized by an isotropic Swin-style transformer that operates on the concatenation of the conditioning field and the noisy target state. In formulation:

$$z_{\text{in}} = [c; z_t] \in \mathbb{R}^{2C\times H\times W}, \quad (21)$$

partition z_{in} into non-overlapping patches of size $P \times P$, and project them into tokens:

$$z^{(0)} = \text{Linear}\big(\text{Patchify}(z_{\text{in}})\big) + P_{\text{abs}}, \quad (22)$$

where P_{abs} is a learned absolute positional embedding that provides fixed geographic context.

We encode the source time t and destination time r by sinusoidal embeddings followed by an MLP:

$$t_{\text{emb}} = f_t(\phi(t)) + f_r(\phi(r)). \quad (23)$$

This temporal embedding modulates each transformer block through adaptive normalization. For the l-th block, we predict scale, shift, and residual gates from t_{emb}:

$$[\alpha^{(l)}, \beta^{(l)}, \gamma^{(l)}] = W_{\text{mod}}^{(l)} t_{\text{emb}} + b_{\text{mod}}^{(l)}. \quad (24)$$

The modulated features are:

$$\mathcal{M}^{(l)}(z) = \text{RMSNorm}(z) \odot (1 + \alpha^{(l)}) + \beta^{(l)}. \quad (25)$$

To preserve fine-scale meteorological structure, we avoid hierarchical downsampling and keep an isotropic resolution throughout the backbone. Self-attention is restricted to local windows, while consecutive blocks alternate between regular and shifted windows to exchange information across window boundaries. In addition, we apply 2D axial rotary positional embeddings (RoPE) to the query and key tokens inside each window to provide relative spatial awareness for advection and vortex-like motion. The l-th block is written as:

$$\tilde{z}^{(l)} = \hat{z}^{(l)} + \boldsymbol{\gamma}_{\text{att}}^{(l)} \odot \text{WAttn}_{\text{RoPE}}\left(\mathcal{M}_{\text{att}}^{(l)}(\hat{z}^{(l)})\right), \quad (26)$$

$$z^{(l)} = \mathcal{S}^{-1}\Big(\tilde{z}^{(l)} + \boldsymbol{\gamma}_{\text{ffn}}^{(l)} \odot \text{SwiGLU}\left(\mathcal{M}_{\text{ffn}}^{(l)}(\tilde{z}^{(l)})\right)\Big), \quad (27)$$

where $\mathcal{S}$ denotes the cyclic shift operator. The residual gates are zero-initialized to stabilize optimization in the early stage of training.

After the final block, a lightweight decoder maps the tokens back to the physical grid and outputs the velocity field:

$$u_\theta(z_s, s, c, r) = \mathrm{Unpatchify}\big(W_{\mathrm{out}} z^{(L)}\big). \tag{28}$$

We additionally attach a variance head that predicts a non-negative scale field σ_θ, which is used for uncertainty modulation during finetuning and inference.

B.2 Pseudocode of `Tyche`

The detailed pseudocode of `Tyche` is illustrated in Algorithm 1.

Algorithm 1 Training and Inference Workflow of `Tyche`

Require: Training dataset $\mathcal{D} = \{(x^\tau, x^{\tau+1}, \dots)\}$, model parameters θ, ensemble size K.

/* Stage I: One-step Transport Pretraining */

for each pretraining iteration **do**
  Sample $(x^\tau, x^{\tau+1}) \sim \mathcal{D}$, noise $\epsilon \sim \mathcal{N}(0, I)$, and times $r < t \in [0, 1]$.
  Construct intermediate state $z_t \leftarrow (1-t)x^{\tau+1} + t\epsilon$ and target $v_t \leftarrow \epsilon - x^{\tau+1}$.
  Compute total derivative $\frac{d}{dt}u_\theta^\star$ via Jacobian-Vector Product (JVP).
  Calculate rectified target $u_{\mathrm{tgt}} \leftarrow v_t - (t-r)\frac{d}{dt}u_\theta^\star$. ▷ Eq. 8
  Update θ by minimizing $\mathcal{L}_{\mathrm{ave}} = \mathrm{MSE}\big(u_\theta(z_t, r, t, x^\tau), \mathrm{sg}(u_{\mathrm{tgt}})\big)$. ▷ Eq. 9
end for

/* Stage II: Curriculum Probabilistic Calibration */

for each curriculum stage with rollout horizon R_m **do**
  for each finetuning iteration **do**
    Sample initial condition x^τ and target state $x^{\tau+R_m}$ from $\mathcal{D}$.
    Initialize K parallel paths: $c_k^0 \leftarrow x^\tau$ for $k \in \{1, \dots, K\}$.
    for lead time $l = 1$ to R_m **do** ▷ Autoregressive 1-NFE Rollout
      Sample $\epsilon_k^l \sim \mathcal{N}(0, I)$ and generate $\hat{x}_k^{\tau+l} \leftarrow \epsilon_k^l - u_\theta(\epsilon_k^l, 0, 1, c_k^{l-1})$.
      Update conditioning: $c_k^l \leftarrow \hat{x}_k^{\tau+l}$.
    end for
    Update θ by minimizing empirical $\mathcal{L}_{\mathrm{CRPS}}$ on terminal states $\{\hat{x}_k^{\tau+R_m}\}$. ▷ Eq. 14
  end for
end for

/* Inference: 1-NFE Ensemble Generation */

Sample K independent Gaussian noises $\{\epsilon_1, \dots, \epsilon_K\} \sim \mathcal{N}(0, I)$.
for member $k = 1$ to K **in parallel do**
  Generate forecast sample: $\hat{x}_k^{\tau+1} \leftarrow \epsilon_k - u_\theta(\epsilon_k, 0, 1, x^\tau)$. ▷ Eq. 15
end for
return Generated forecast ensemble $\hat{X} = \{\hat{x}_1^{\tau+1}, \dots, \hat{x}_K^{\tau+1}\}$.

C Theoretical Analysis and Proofs

We analyze how errors in the learned conditional transition distribution propagate during autoregressive forecasting. We first distinguish approximation error in average-velocity transport from numerical integration error. We then derive a rollout error bound under a Lipschitz assumption on the true transition kernel and relate the local kernel discrepancy to a pathwise rectification residual. Finally, we discuss the implications and limitations of this analysis for Stage II finetuning.

C.1 Approximation Error in One-Step Transport

Standard flow matching parameterizes an instantaneous velocity field, whose associated ordinary differential equation is typically solved by numerical integration. In contrast, `Tyche` parameterizes an interval-averaged velocity. For the exact average-velocity field $u^\star$, the endpoints of the corresponding

transport trajectory satisfy

$$z_r = z_s - (s-r)u^\star(z_s, r, s, c), \qquad 0 \le r < s \le 1. \tag{29}$$

Using a learned field u_θ gives $\widehat{z}_r = z_s - (s-r)u_\theta(z_s, r, s, c)$, and hence

$$\widehat{z}_r - z_r = -(s-r)\left(u_\theta(z_s, r, s, c) - u^\star(z_s, r, s, c)\right). \tag{30}$$

Thus, the endpoint error arises from approximation of the average-velocity field. The update should not be identified with a single Euler step applied to an instantaneous velocity field, nor does it incur an unavoidable Euler truncation error when the average velocity is exact.

During autoregressive forecasting, subsequent transitions are conditioned on generated states. Their distribution may differ from that of the observed conditioning states used in Stage I. The following analysis describes how local transition discrepancies and sensitivity to conditioning states contribute to the resulting rollout error.

C.2 Rollout Error Bound and Proof

We consider the Markov formulation used in the main text. Let $\mathcal{X} = \mathbb{R}^M$ denote the vectorized forecast state space. To reflect spatial weighting, equip this space with the weighted L_1 norm

$$\|x\|_{1,w} \coloneqq \sum_{i=1}^{M} w_i |x_i|, \qquad w_i > 0, \tag{31}$$

where the weights are fixed. Let $W_{1,w}$ be the Wasserstein-1 distance induced by this ground metric.

For a fixed conditioning state c, define the one-step sampler and its transition kernel by

$$\begin{aligned} g_\theta(\epsilon, c) &\coloneqq \epsilon - u_\theta(\epsilon, 0, 1, c), \qquad \epsilon \sim \mathcal{N}(0, I), \\ K_\theta(\cdot \mid c) &\coloneqq \mathrm{Law}\big(g_\theta(\epsilon, c)\big). \end{aligned} \tag{32}$$

Let $K^\star(\cdot \mid c)$ denote the true transition kernel. The local kernel discrepancy is

$$d_\theta(c) \coloneqq W_{1,w}\left(K_\theta(\cdot \mid c), K^\star(\cdot \mid c)\right). \tag{33}$$

We also define a residual-based upper bound. For each fixed c, independently sample $x \sim K^\star(\cdot \mid c)$ and $\epsilon \sim \mathcal{N}(0, I)$, and set

$$z_s^{x,\epsilon} = (1-s)x + s\epsilon, \qquad v^{x,\epsilon} = \epsilon - x. \tag{34}$$

The sample-path rectification residual at destination time $r = 0$ is

$$\rho_\theta^{x,\epsilon}(s, c) \coloneqq\ u_\theta(z_s^{x,\epsilon}, 0, s, c) - v^{x,\epsilon} + s\left[\partial_s u_\theta(z_s^{x,\epsilon}, 0, s, c) + J_z u_\theta(z_s^{x,\epsilon}, 0, s, c) v^{x,\epsilon}\right]. \tag{35}$$

Here, ∂_s differentiates the explicit source-time argument while holding the spatial argument fixed. Define

$$\delta_\theta(c) \coloneqq \int_0^1 \mathbb{E}_{x,\epsilon|c}\left[\|\rho_\theta^{x,\epsilon}(s, c)\|_{1,w}\right] \mathrm{d}s. \tag{36}$$

We assume that the kernels are measurable, that the transition and rollout distributions considered below have finite first moments, and that all residual expectations are finite. For each fixed c, we assume sufficient regularity of u_θ for the chain rule and the fundamental theorem of calculus along the paths below. In particular, the field is continuous at the diagonal $s = r$, continuously differentiable along the relevant paths, and its total derivative is integrable. Continuity at the diagonal ensures

$$\lim_{s \downarrow r} (s-r) u_\theta(z_s, r, s, c) = 0. \tag{37}$$

Theorem 1 (*State-sensitive rollout error bound*). *Assume that the true transition kernel satisfies*

$$W_{1,w}\left(K^\star(\cdot \mid c), K^\star(\cdot \mid c')\right) \le \Lambda \|c - c'\|_{1,w} \tag{38}$$

for all conditioning states in the common state space, including those visited by the model, where $\Lambda \ge 0$. Let μ_h^θ and $\mu_h^\star$ be the model and true rollout distributions, respectively, with

$$\mu_{h+1}^\theta = \mu_h^\theta K_\theta, \qquad \mu_{h+1}^\star = \mu_h^\star K^\star, \qquad \mu_0^\theta = \mu_0^\star.$$

Under the regularity and integrability assumptions above, for every $H \geq 1$,

$$W_{1,w}\left(\mu_H^\theta, \mu_H^\star\right) \leq \sum_{j=0}^{H-1} \Lambda^{H-1-j} \mathbb{E}_{c\sim\mu_j^\theta}\left[d_\theta(c)\right] \leq \sum_{j=0}^{H-1} \Lambda^{H-1-j} \mathbb{E}_{c\sim\mu_j^\theta}\left[\delta_\theta(c)\right]. \tag{39}$$

The coefficient for $j = H-1$ is understood to be 1, including when $\Lambda = 0$.

Proof. The proof first establishes a pathwise endpoint identity, then bounds the local kernel discrepancy, and finally propagates this discrepancy through the autoregressive chain.

Step 1: Residual-to-endpoint identity. Fix a conditioning state c, a destination time r, and a continuously differentiable path z_s with velocity $v_s = \mathrm{d}z_s/\mathrm{d}s$. For a candidate average-velocity field u, define its total derivative along this path by

$$D_s u(z_s, r, s, c) \coloneqq \partial_s u(z_s, r, s, c) + J_z u(z_s, r, s, c) v_s. \tag{40}$$

The associated pathwise residual is

$$\rho_u(s) \coloneqq u(z_s, r, s, c) - v_s + (s-r) D_s u(z_s, r, s, c). \tag{41}$$

Let $\widehat{z}_r(s) = z_s - (s-r)u(z_s, r, s, c)$. Differentiation gives

$$\frac{\mathrm{d}}{\mathrm{d}s}\widehat{z}_r(s) = -\rho_u(s). \tag{42}$$

The diagonal boundary condition implies $\widehat{z}_r(r) = z_r$. Integrating from r to s therefore yields

$$\widehat{z}_r(s) - z_r = -\int_r^s \rho_u(a)\,\mathrm{d}a. \tag{43}$$

By the triangle inequality,

$$\|\widehat{z}_r(s) - z_r\|_{1,w} \leq \int_r^s \|\rho_u(a)\|_{1,w}\,\mathrm{d}a. \tag{44}$$

Step 2: From endpoint error to kernel discrepancy. Apply Step 1 to the linear path in Eq. (34), with $u = u_\theta$, $r = 0$, and $s = 1$. Its endpoints are $z_0 = x$ and $z_1 = \epsilon$, while the predicted destination is $\widehat{x} = g_\theta(\epsilon, c)$. Consequently,

$$\|\widehat{x} - x\|_{1,w} \leq \int_0^1 \|\rho_\theta^{x,\epsilon}(s, c)\|_{1,w}\,\mathrm{d}s. \tag{45}$$

The pair $(\widehat{x}, x)$ is a valid coupling of $K_\theta(\cdot \mid c)$ and $K^\star(\cdot \mid c)$. Since the Wasserstein distance is the infimum of the expected ground cost over all couplings,

$$d_\theta(c) \leq \mathbb{E}_{x,\epsilon|c}\left[\|\widehat{x} - x\|_{1,w}\right] \leq \int_0^1 \mathbb{E}_{x,\epsilon|c}\left[\|\rho_\theta^{x,\epsilon}(s, c)\|_{1,w}\right]\,\mathrm{d}s = \delta_\theta(c). \tag{46}$$

Step 3: Autoregressive error propagation. Write $e_h = W_{1,w}(\mu_h^\theta, \mu_h^\star)$. The triangle inequality gives:

$$e_{h+1} \leq W_{1,w}\left(\mu_h^\theta K_\theta, \mu_h^\theta K^\star\right) + W_{1,w}\left(\mu_h^\theta K^\star, \mu_h^\star K^\star\right). \tag{47}$$

The mixture inequality for Wasserstein distance bounds the first term by

$$W_{1,w}\left(\mu_h^\theta K_\theta, \mu_h^\theta K^\star\right) \leq \mathbb{E}_{c\sim\mu_h^\theta}\left[d_\theta(c)\right]. \tag{48}$$

For the second term, let π_h be an optimal coupling of μ_h^θ and $\mu_h^\star$. Using Eq. (38),

$$\begin{aligned} W_{1,w}\left(\mu_h^\theta K^\star, \mu_h^\star K^\star\right) &\leq \int W_{1,w}\left(K^\star(\cdot \mid c), K^\star(\cdot \mid c')\right)\,\pi_h(\mathrm{d}c, \mathrm{d}c') \\ &\leq \Lambda \int \|c - c'\|_{1,w}\,\pi_h(\mathrm{d}c, \mathrm{d}c') = \Lambda e_h. \end{aligned} \tag{49}$$

Combining these inequalities yields

$$e_{h+1} \leq \Lambda e_h + \mathbb{E}_{c\sim\mu_h^\theta}\left[d_\theta(c)\right]. \tag{50}$$

Since $e_0 = 0$, unrolling the recursion proves the first inequality in Eq. (39). Applying $d_\theta(c) \leq \delta_\theta(c)$ proves the second. □

Interpretation of the residual upper bound. The quantity $d_\theta(c)$ is the actual discrepancy between the conditional transition distributions, whereas $\delta_\theta(c)$ is a sufficient, potentially loose upper bound. In Step 2, x and $\widehat{x}$ are independent conditional on c, because the verifying target and sampling noise are drawn independently. This coupling is valid but generally not optimal. For a non-degenerate true distribution, its expected cost can remain positive even when $K_\theta(\cdot \mid c) = K^\star(\cdot \mid c)$. Accordingly, $\delta_\theta(c)$ need not vanish for a distributionally correct sampler and should not be interpreted as an exact measure of distributional error. The residual certificate also uses the fixed destination $r = 0$. A small Stage I loss averaged over sampled (r, s) does not, by itself, establish a small residual on this boundary. Such a conclusion would require additional control of the time dependence or explicit supervision of the relevant boundary slice.

Interpretation of the sensitivity factor. The constant Λ bounds the sensitivity of the true transition kernel in the chosen Wasserstein metric. It is not identified with a Lyapunov exponent. When $\Lambda > 1$, the bound permits amplification of earlier transition discrepancies; it does not imply that the actual forecast error necessarily grows at that rate. For example, if $\mathbb{E}_{c\sim\mu_j^\theta}[d_\theta(c)] \le \bar{d}$ for all steps under consideration, then:

$$e_H \le \bar{d} \sum_{k=0}^{H-1} \Lambda^k. \tag{51}$$

This gives a horizon-independent bound $e_H \le \bar{d}/(1-\Lambda)$ when $\Lambda < 1$, a bound $e_H \le H\bar{d}$ when $\Lambda = 1$, and a potentially exponentially growing upper bound when $\Lambda > 1$. The theorem therefore does not establish long-horizon stability without additional assumptions.

C.3 Bridging Theory and Practice: The Rationale for Stage II Design

Theorem 1 evaluates local kernel discrepancies under the model-generated state distributions μ_j^θ. This highlights the importance of prediction quality under generated conditioning states, beyond the observed states used during Stage I. Motivated by this observation, Stage II uses autoregressive ensemble rollouts and probabilistic supervision at their terminal forecast states.

Marginal Calibration with Fair CRPS. For a fixed initial condition, forecast lead time, and scalar forecast component, let F_θ and G denote the predictive and true cumulative distribution functions. Given $K \ge 2$ conditionally independent ensemble samples $X^{1:K} \sim F_\theta$ and an observation $Y \sim G$, independent of the sampling noise, the fair CRPS estimator is

$$\widehat{\mathrm{CRPS}}_{\mathrm{fair}}(X^{1:K}, Y) = \frac{1}{K}\sum_{k=1}^{K} |X^k - Y| - \frac{1}{2K(K-1)} \sum_{\substack{k,k'=1 \\ k\neq k'}}^{K} |X^k - X^{k'}|. \tag{52}$$

This is an unbiased estimator of the population CRPS $\mathrm{CRPS}(F_\theta, Y)$, conditional on Y and the initial condition. For distributions with finite first moments, the expected excess score satisfies

$$\mathbb{E}_{X^{1:K},Y}\left[\widehat{\mathrm{CRPS}}_{\mathrm{fair}}(X^{1:K}, Y)\right] - \mathbb{E}_{Y\sim G}\left[\mathrm{CRPS}(G, Y)\right] = \int_{\mathbb{R}} \left(F_\theta(z) - G(z)\right)^2 \, \mathrm{d}z. \tag{53}$$

This follows from the CDF representation

$$\mathrm{CRPS}(F, y) = \int_{\mathbb{R}} \left(F(z) - \mathbf{1}\{y \le z\}\right)^2 \, \mathrm{d}z.$$

Thus, the expected score is minimized when the predictive marginal matches the true marginal. Applying this score to individual forecast components and aggregating with latitude weights provides a tractable objective for marginal distributional accuracy.

The resulting objective is not equivalent to the joint Wasserstein distance in Theorem 1. Matching all scored marginals does not, by itself, identify the spatial or cross-variable dependence structure. Furthermore, terminal CRPS evaluates the forecast distribution produced by a rollout; it does not directly estimate $d_\theta(c)$ or the residual upper bound $\delta_\theta(c)$ at intermediate generated states. Its role is to provide probabilistic supervision for rollout outcomes, consistent with the motivation to train beyond observed conditioning states.

Short Rollouts for Training on Generated States. We use a curriculum with rollout horizons $R = 1$ and $R = 2$ to introduce autoregressive feedback while keeping training computation and

memory requirements manageable. The one-step stage supervises forecasts conditioned on observed states. In the two-step stage, the second transition is conditioned on the model's own first-step prediction, and the terminal CRPS evaluates the resulting two-step forecast distribution. Thus, the second transition encounters conditioning states distributed according to μ_1^θ, while μ_2^θ describes the terminal predictions.

The amplification factors in Theorem 1 motivate attention to errors along generated trajectories, but they do not determine the optimal training horizon or characterize gradient variance. The short-rollout curriculum is therefore a practical training choice. Its effectiveness must be assessed empirically, and the theorem alone does not guarantee that short-rollout finetuning controls errors at arbitrary forecast horizons.

D Extended Experiments

D.1 Ecternal ablation study

To validate the necessity of our proposed two-stage training strategy, we conduct an ablation study isolating the impact of the Stage II curriculum probabilistic calibration. Figure 8 compares the predictive performance of the pretrained base model (**Tyche** *(base)*) against the fully finetuned model (**Tyche** *(fine-tune)*) over a 10-day autoregressive rollout. We report both CRPS and RMSE across representative upper-air kinematic and thermodynamic variables (Q700, U500, U850, V500, V850).

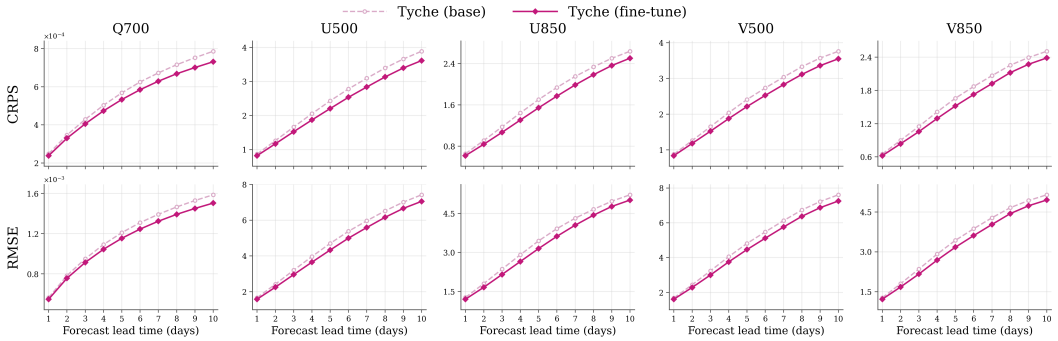


Figure 8: Ablation study on the ensemble finetuning. We compare the latitude-weighted CRPS (top) and RMSE (bottom) of the pretrained **Tyche** *(base)* versus the fully trained **Tyche** *(fine-tune)* across five upper-air variables over a 10-day horizon.

As illustrated in Figure 8, the base model is significantly outperformed by its fine-tuned counterpart, suffering from increasingly severe compounding errors during inference. Conversely, the fine-tuned model effectively constrains this error accumulation. Such a performance gap provides strong empirical corroboration for our theoretical analysis. Specifically, the CRPS-based curriculum fine-tuning functions as an *amortized correction mechanism*; it exposes the model to its own autoregressive drift (on-policy), thereby enforcing the temporal self-consistency required for stable long-term probabilistic forecasting.

D.2 Comparison on more variables

To comprehensively evaluate predictive skill across diverse meteorological fields, we report the latitude-weighted RMSE for 4 surface and 10 upper-air variables. We evaluate under two distinct settings: single-trajectory deterministic skill (Table 5) and ensemble forecasting capability (Table 6).

Single-Trajectory Predictive Skill. Table 5 compares the deterministic model GraphCast against a *single ensemble member* from the probabilistic models (ArchesGen and **Tyche**). As expected, GraphCast holds a slight advantage in short-term (1-day) forecasts, particularly for wind components. However, a single member of **Tyche** remains highly competitive early on and excels in thermodynamic variables (e.g., t2m, q7). Notably, at medium-range horizons (3 to 10 days), **Tyche**'s single trajectory systematically outperforms both GraphCast and ArchesWeatherGen across the vast majority of variables. This demonstrates that **Tyche** produces highly realistic and stable individual atmospheric trajectories without suffering from severe autoregressive error accumulation.

Ensemble Forecasting Capability. Table 6 specifically assesses uncertainty modeling by comparing the RMSE of the *20-member ensemble mean* for probabilistic models. Under this rigorous ensemble evaluation, `Tyche` exhibits absolute superiority, achieving the lowest RMSE across *all* 14 variables at *every* evaluated lead time (1 to 10 days). This consistent across-the-board improvement confirms that `Tyche`'s one-step generative transport not only maintains trajectory diversity but also guarantees a highly accurate and calibrated central predictive tendency for complex global weather fields.

Table 5: Extended evaluation of latitude-weighted RMSE for surface and upper-air variables across different forecast lead times. Best results are highlighted in **bold**.

Lead Time	Model	Surface Variables				Upper-Air Variables (Pressure Levels)									
		u10	v10	t2m	mslp	z7	z9	q7	q9	t8	t10	U7	U9	V7	V9
1-Day	GraphCast	**0.96**	**1.01**	1.24	1.09	**0.80**	0.71	3.23	6.39	0.69	0.98	**1.81**	**1.57**	**1.92**	**1.56**
	ArchesGen	1.11	1.16	1.25	1.27	1.22	0.95	3.20	6.36	0.77	1.00	2.27	1.88	2.35	1.95
	Tyche	1.09	1.12	**1.19**	**0.93**	0.83	**0.69**	**2.71**	**5.17**	**0.60**	**0.95**	2.13	1.85	2.21	1.86
3-Day	GraphCast	2.21	2.36	2.28	2.86	2.73	2.29	5.46	10.23	1.44	1.90	**4.12**	**3.20**	4.63	**3.38**
	ArchesGen	2.15	**2.22**	1.88	2.71	2.53	1.94	5.30	10.30	1.57	1.78	4.28	3.37	4.69	3.49
	Tyche	**2.13**	2.31	**1.68**	**2.49**	**2.29**	**1.85**	**5.11**	**9.84**	**1.39**	**1.63**	4.27	3.32	**4.56**	3.53
7-Day	GraphCast	3.97	3.76	3.62	6.64	6.73	5.34	7.87	15.28	3.15	3.49	7.85	**5.84**	8.02	6.04
	ArchesGen	4.25	4.05	3.35	6.84	7.65	6.08	7.81	14.72	3.85	3.59	8.84	7.06	8.97	6.71
	Tyche	**3.89**	**3.52**	**2.77**	**6.03**	**6.02**	**4.57**	**7.12**	**14.24**	**3.08**	**3.02**	**7.48**	5.85	**7.80**	**5.73**
10-Day	GraphCast	4.94	**4.59**	4.50	10.28	10.51	8.37	9.22	17.88	3.89	4.24	10.20	7.54	10.09	7.14
	ArchesGen	4.88	4.87	4.17	9.91	9.69	7.56	8.62	17.12	4.33	4.33	10.34	7.75	10.23	7.57
	Tyche	**4.71**	4.64	**3.74**	**8.58**	**8.97**	**6.94**	**8.44**	**16.53**	**3.87**	**3.94**	**9.62**	**7.31**	**9.77**	**7.00**

Table 6: Comparison of latitude-weighted RMSE between ArchesGen and `Tyche` for surface and upper-air variables. Best results for each lead time are highlighted in **bold**.

Lead Time	Model	Surface Variables				Upper-Air Variables (Pressure Levels)									
		u10	v10	t2m	mslp	z7	z9	q7	q9	t8	t10	U7	U9	V7	V9
1-Day	ArchesGen	0.86	0.92	1.06	1.02	0.74	0.65	2.54	4.99	0.62	0.83	1.81	1.48	1.81	1.46
	Tyche	**0.79**	**0.83**	**0.88**	**0.71**	**0.58**	**0.50**	**2.35**	**4.69**	**0.50**	**0.73**	**1.57**	**1.36**	**1.63**	**1.35**
3-Day	ArchesGen	1.88	1.94	1.70	2.60	2.36	1.81	4.40	8.30	1.42	1.62	3.76	2.91	4.10	3.01
	Tyche	**1.63**	**1.71**	**1.32**	**2.00**	**1.83**	**1.45**	**3.90**	**7.68**	**1.14**	**1.29**	**3.29**	**2.55**	**3.59**	**2.67**
7-Day	ArchesGen	3.62	3.15	2.68	6.02	6.59	5.09	6.38	12.02	3.24	2.89	6.86	5.22	6.67	5.28
	Tyche	**3.12**	**2.77**	**2.03**	**5.03**	**5.00**	**3.92**	**5.53**	**11.12**	**2.39**	**2.32**	**6.13**	**4.76**	**6.19**	**4.61**
10-Day	ArchesGen	3.79	3.85	3.37	8.15	8.24	6.32	7.14	14.11	3.62	3.58	7.95	5.93	8.23	5.98
	Tyche	**3.74**	**3.68**	**3.31**	**7.29**	**7.58**	**5.80**	**6.44**	**12.53**	**3.38**	**3.38**	**7.67**	**5.83**	**7.79**	**5.73**

D.3 Comparison of Spread skills

To comprehensively evaluate the spatial reliability of uncertainty estimation, we conducted a comparative visual analysis between the ensemble spread and the absolute error of the ensemble mean ($|\text{mean} - \text{truth}|$). In a well-calibrated forecasting system, the spatial distribution of the spread should tightly align with the forecast error. Figure 9 visualizes this alignment at a 5-day lead time for key surface variables (t2m, mslp, u10, and v10).

As illustrated in Figure 9, `Tyche` demonstrates a precise spatial correlation between its predicted uncertainty and actual errors. It accurately captures the complex variance in dynamically active regions (*e.g.*, the Southern Ocean) without suffering from pathological under-dispersion. Compared to multi-step generative baselines like ArchesWeatherGen and ARCI, `Tyche` maintains highly competitive and physically coherent uncertainty structures. This empirically validates that our CRPS-based fine-tuning effectively calibrates spatial dispersion and achieves robust reliability.

D.4 Detailed visualizations on more horizons and variables

To further demonstrate the physical fidelity and temporal stability of our method, the following Figure 10, 11, 12 present comprehensive visualizations of `Tyche`'s single-member forecasts and absolute errors across nine meteorological variables at 1-day, 3-day, and 7-day lead times.

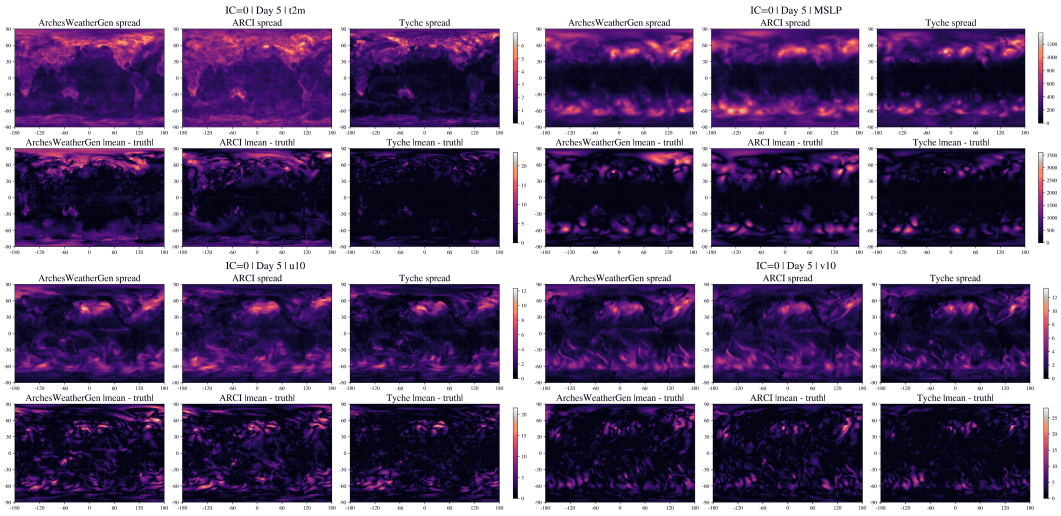


Figure 9: Spatial comparison of ensemble spread (top rows) and absolute mean error (bottom rows) at a 5-day lead time.

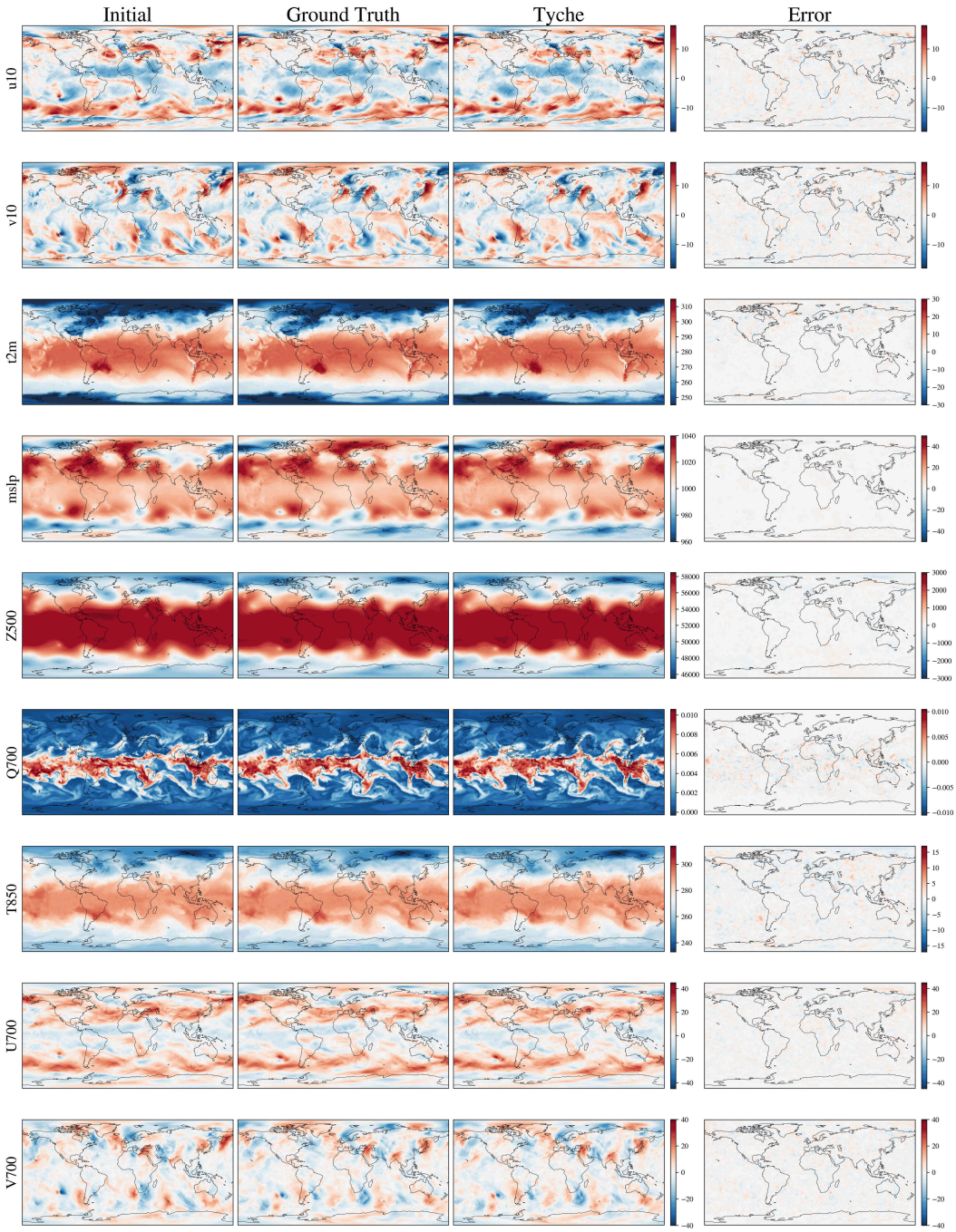


Figure 10: Detailed visualization of 1-day single-member forecasts for nine selected variables.

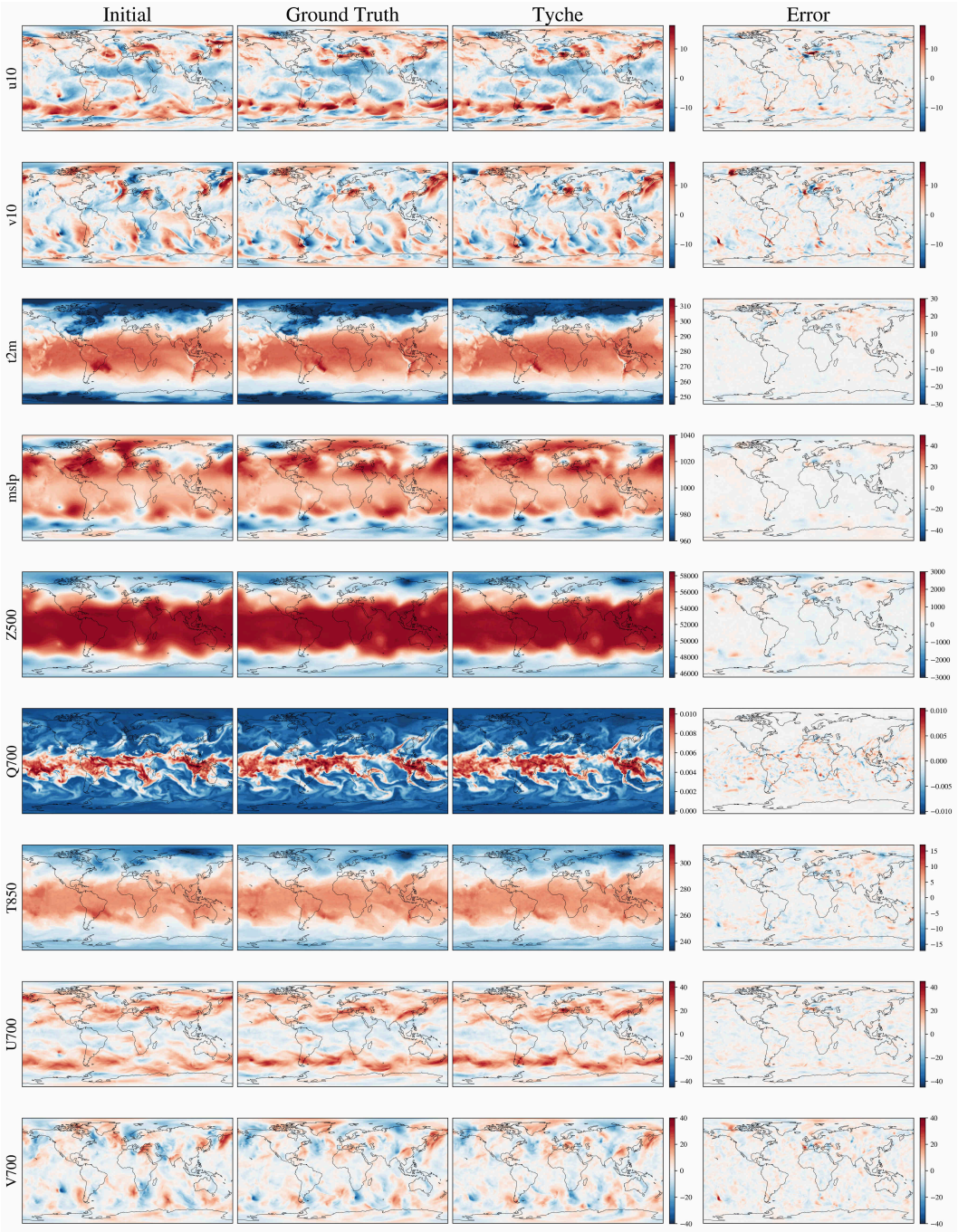


Figure 11: Detailed visualization of 3-day single-member forecasts for nine selected variables.

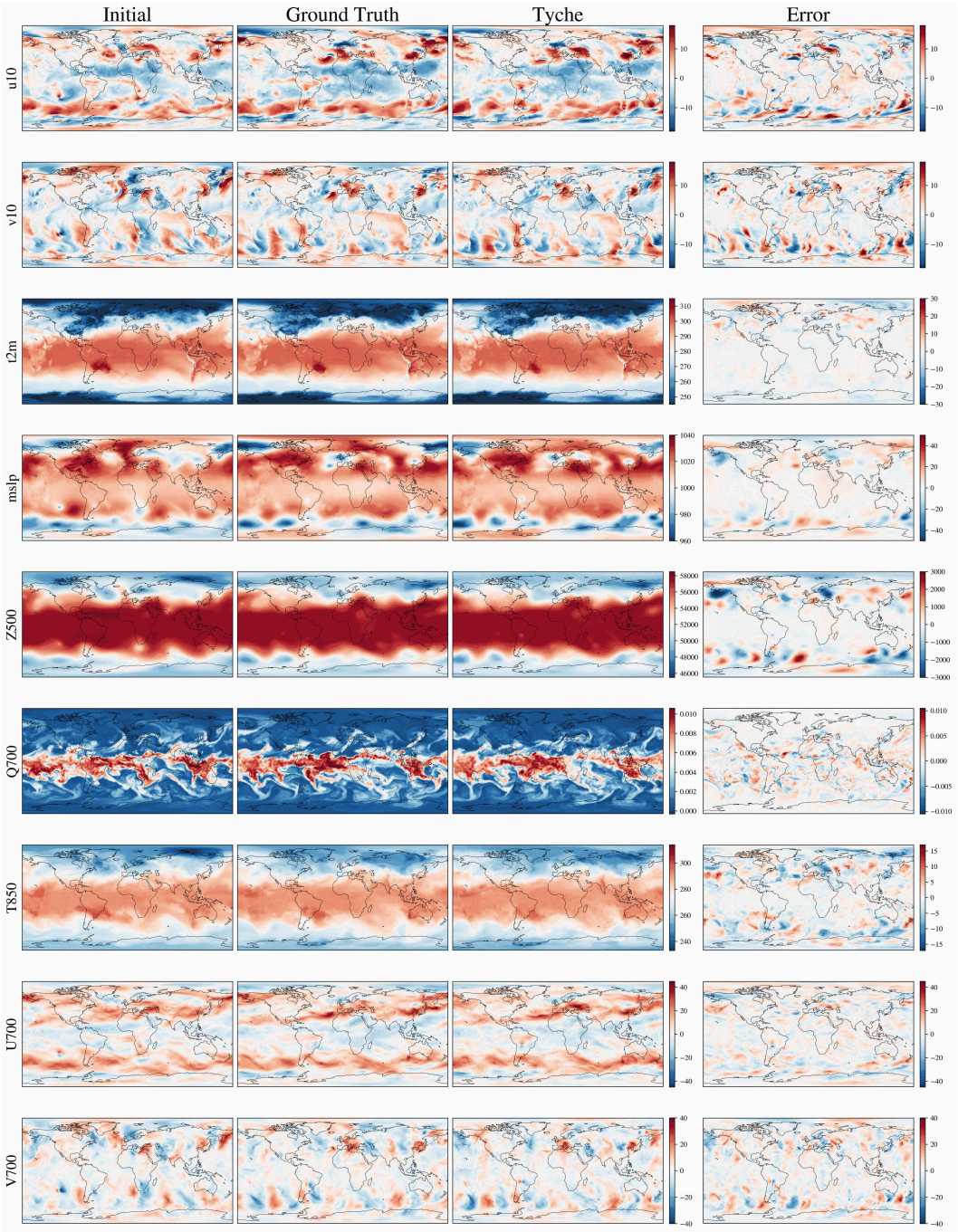


Figure 12: Detailed visualization of 7-day single-member forecasts for nine selected variables.

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 main claims regarding the 1-NFE efficiency, the generative transport method, and competitive performance are explicitly stated in the Abstract and Introduction (Section 1), and are fully supported by theoretical analysis (Appendix C) and empirical results (Section 5).

   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: We have included it in the conclusion.

   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: Theorem and rigorous mathematical proofs are formally provided in Appendix C.

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: We provide comprehensive details regarding the dataset preprocessing (Appendix A.1), baseline configurations (Appendix A.2, network architecture (Appendix B) and training pipeline (Appendix A.2.4), and evaluation metrics (Appendix A.3) to fully facilitate reproducibility.

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 project repository is linked in the abstract, and the ERA5 dataset and WeatherBench2 are publicly available. We have provided detailed implementation in Appendix A.2.

   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: Detailed experimental settings, including data splits (years 1979-2017 for training, 2020 for testing), optimization hyperparameters, curriculum stages, and architecture specifics, are extensively documented in Section 5.1 and Appendix A.

   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: Following standard practices in global ML weather forecasting (e.g., WeatherBench2), we evaluate performance by aggregating metrics (RMSE, CRPS) over a massive number of spatial grid points and a full year of temporal test data (2020), which naturally stabilizes the results.

   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: We explicitly state the hardware resources required for training (eight 80GB NVIDIA H100 GPUs) and inference (a single GPU) in Section 5.1 and Appendix A.2.4.

   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: This research focuses on improving atmospheric forecasting using publicly available scientific data, raising no ethical concerns, and strictly conforms to 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: The potential positive societal impacts of the work, such as improving high-impact decision-making in aviation, energy, agriculture, and disaster response, are discussed in the Introduction.

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 focuses on weather forecasting models based on physical atmospheric data, which do not pose high risks for malicious misuse (e.g., generation of deceptive content).

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 properly cite the original papers for the ERA5 dataset and the WeatherBench2 evaluation framework, respecting their respective open-access terms.

Guidelines:

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

13. **New assets**

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

    Answer: [N/A]

    Justification: The paper introduces a novel methodology and architecture but does not release new benchmark datasets.

    Guidelines:

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

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

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

    Answer: [N/A]

    Justification: The research does not involve crowdsourcing or 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 research does not involve human subjects.

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: The core method development in this research does not involve LLMs as any component.

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.