Static and Dynamic Diffusion Emulators: From Sampling Gray Swan Extreme Events to Suffering from Model Collapse

Karan Jakhar
Department of Mechanical Engineering
Rice University
Houston TX 77005
Department of Geophysical Sciences
University of Chicago
Chicago IL 60637
karanj@uchicago.edu

Pedram Hassanzadeh
Department of Geophysical Sciences and
Committee on Computational and Applied Mathematics
Chicago IL 60637
pedramh@uchicago.edu

Björn Lütjens
IBM Research
NY 10598
lutjens@ibm.com

Jonathan Weare
Courant Institute of Mathematical Sciences
New York NY 10012
weare@nyu.edu

Abstract

Characterizing rare extreme events is a major challenge in climate and other sciences, and AI emulators may offer a solution: The idea is that an emulator trained on a small dataset can generate large synthetic datasets with extreme events beyond those that were absent from the training set. In principle. AI cannot generalize beyond the training set (i.e., extrapolate), however, because the governing equations are fixed, there are speculations that generative models, especially those that learn from fast dynamics, can reproduce stronger events from the same dynamical system. Here, we investigate the most promising approaches using 2D geophysical turbulence test cases. First, we train an unconditional static emulator that samples plausible flow states from noise; second, an autoregressive dynamic emulator; and third, iteratively augment the training dataset using the emulator's output. We show that both static and dynamic emulators can produce extreme events stronger than those in the training set, effectively creating "gray swans". However, neither could accurately reproduce the frequency (return period) of these rare events. We show that both emulators are constrained by the small training set, and trying to increase the data size by iteratively training emulators on their own output leads to model collapse - a degenerative feedback loop that progressively worsens the quality of generated samples and extreme estimates. Overall, these findings indicate that purely data-driven diffusion models are not capable of learning the underlying data-generation process from a small training set.

1 Introduction

Accurately modeling and predicting the rarest, yet most impactful, extreme events is a major challenge in many scientific disciplines. For example, in climate science, quantifying the statistics of weather extremes through directly sampling high-fidelity physics-based simulations is often computationally prohibitive [Lucarini and Chekroun, 2023, de Burgh-Day and Leeuwenburg, 2023, Materia et al., 2024, Lai et al., 2024, Bracco et al., 2025]. Various statistical and algorithmic tools have been developed to supplement physics-based simulations, including extreme-value theory (EVT) [Coles et al., 2001], large deviation theory [Ragone et al., 2018, Ragone and Bouchet, 2020], and **different forms of rare event sampling** [Webber et al., 2019, Abbot et al., 2021, Finkel et al., 2023] such as ensemble boosting [Vanden-Eijnden and Weare, 2012, Gessner et al., 2021] and importance sampling [Ragone and Bouchet, 2021]. However, these methods often have significant limitations; for example, EVT provides only statistical distributions without describing the circulation patterns and dynamics of an event, while others still require substantial, targeted simulations that limit their broad application. These constraints have recently shifted attention towards artificial intelligence (AI) emulators as a more flexible and potentially powerful approach for estimating extremes and their associated uncertainty [Materia et al., 2024, Camps-Valls et al., 2025].

Building on the success of deep learning in short- and medium-range weather forecasting [Pathak et al., 2022, Lam et al., 2023, Bi et al., 2023, Chen et al., 2023, Rühling Cachay et al., 2023, Lang et al., 2024, Nguyen et al., 2024, Price et al., 2025], the core idea is to train an AI emulator on a small amount of available high-fidelity data (e.g., from high-resolution physics-based simulations). The trained emulator can then be used to generate large, long ensembles (e.g., orders of magnitude more than the training data) at a fraction of the computational cost [Lai et al., 2024, Watson-Parris, 2021, Bracco et al., 2025, Li et al., 2024a, Lam et al., 2023, Pathak et al., 2022], providing enough data for direct sampling of rare extremes. In particular, this approach may enable the characterization of gray swans: strong, extreme events that are physically plausible, but so rare that they were absent from the limited training dataset [Lin and Emanuel, 2016, Sun et al., 2025a].

The success of this approach hinges on several critical requirements. For an AI emulator to be a reliable tool for extreme event analysis, it must:

1. Generate outputs that are stable and physically consistent over long integrations, accurately reproducing the system's mean statistics.
2. Extrapolate beyond its training data to generate gray swan events more intense than anything seen during training.
3. Reproduce the correct statistics (e.g., frequency or return period) of these extrapolated events.

Satisfying the first requirement for long-term stability is now a well-established capability of many AI emulators [Kochkov et al., 2024, Dheeshjith et al., 2025, Chapman et al., 2025, Watt-Meyer et al., 2025, Rühling Cachay et al., 2023]. The key unresolved challenge, however, is meeting the other two requirements by extrapolating to generate extreme, out-of-distribution (OOD) events with the correct statistics. While OOD generalization (i.e., extrapolation) is a general weakness of neural networks, there is growing hope that this might be possible given the evidence that AI models can learn physical dynamics [Rackow et al., 2024, Sun et al., 2025b, Meng et al., 2025]. That said, there are also studies emerging showing a lack of extrapolation to gray swans in some of the AI weather models [Sun et al., 2025a, Zhang et al., 2025]

To investigate (2)-(3), we rigorously evaluate two distinct strategies designed to enable generative emulators to predict extreme events beyond their training data, a process outlined in Figure 1. A key challenge in assessing out-of-distribution performance is the lack of very long ground-truth records for comparison. We overcome this by using a 2D geophysical turbulence test case, for which we can generate extensive ground-truth data. Our first strategy focuses on building two diffusion emulators [Ho et al., 2020, Batzolis et al., 2021], as shown in Figure 1(b). We investigate both a self-supervised static diffusion model (hereafter 'static') that directly samples plausible system states [Lienen et al., 2023, Li et al., 2024b,a, Whittaker et al., 2024, Brenowitz et al., 2025], and a supervised autoregressive conditional diffusion model (hereafter 'dynamic') that learns the system's temporal evolution, like the current state-of-the-art weather models [Kohl et al., 2023, Rühling Cachay et al., 2023, Price et al., 2025, Stamatelopoulos and Sapsis, 2025, Bassetti et al., 2024]. A potential advantage of an autoregressive emulator is that it can learn the underlying (fast) dynamics and can have a better chance of producing gray swans. We train these emulators on a very short dataset from the 2D turbulent flow and then use them to generate much longer emulations.

Our second strategy, outlined in Figure 1(e)-(f), explores an often mentioned potential solution to the small-data problem: using the emulator to generate synthetic data for further training of itself [Antoniou et al., 2017, Lai et al., 2024]. If an emulator can successfully generate physically consistent data containing gray swan events, it is natural to investigate whether this new dataset can be used to retrain and improve the emulator's performance on even more extreme events, reproducing the correct statistics. However, similar experiments in other domains (e.g., LLMs) have shown that such an approach leads to *model autophagy disorder* or *model collapse* [Alemohammad et al., 2024, Shumailov et al., 2024, Feng et al., 2024, Dohmatob et al., 2024, Kazdan et al.], an autophagous (self-consuming) feedback loop where models iteratively trained on their own output progressively lose fidelity. Here, we aim to test whether this approach for dynamical systems, especially using the dynamic emulator, can address the small-data problem or lead to model collapse.

Overall, this two-part investigation allows us to evaluate the current capabilities of generative emulators, concluding that they can partially satisfy requirements (1) and (2), but fail to meet requirement (3).

2 Methods and Data

2D Turbulence Test Case: We use 2D turbulence flow as the test case. This canonical flow has been extensively used for testing novel AI methods [Jakhar et al., 2024a, Guan et al., 2022, Pedersen et al., 2025, Lippe et al., 2023, Maulik et al., 2019]. We consider 2D turbulence described by the dimensionless Navier-Stokes equations for an incompressible flow in (x, y) spatial dimensions [Jakhar et al., 2024a, Guan et al., 2022]:

$$\nabla \cdot \boldsymbol{u} = 0, \tag{1}$$

$$\frac{\partial \boldsymbol{u}}{\partial t} + (\boldsymbol{u} \cdot \nabla)\, \boldsymbol{u} = -\nabla p + \frac{1}{Re}\nabla^2 \boldsymbol{u} + \boldsymbol{\mathcal{F}} - \chi \boldsymbol{u} + \boldsymbol{\mathcal{B}}, \tag{2}$$

where $\boldsymbol{u} = [u(x, y, t), v(x, y, t)]$ is the velocity, $p(x, y, t)$ is the pressure, $\chi = 0.1$ is the Rayleigh drag coefficient, and $Re = 5 \times 10^2$ is the Reynolds number. $\boldsymbol{\mathcal{F}}(x, y)$ represents an time-constant external forcing with $\nabla \times \boldsymbol{\mathcal{F}} = k_f \cos(k_f x) + k_f \cos(k_f y)$ at a wavenumber, $k_f = 4$. $\boldsymbol{\mathcal{B}}(x, y)$, with $\nabla \times \boldsymbol{\mathcal{B}} = -\beta v$, and $\beta = 20$ represents Earth's rotation leading to zonal jets mimicking jet streams and other strong coherent current in the atmosphere and ocean [Vallis, 2017]. The domain is doubly periodic with length $L = 2\pi$.

Training Data: (u, v) fields on a 64^2 grid. Datasets A, B: 2,500 snapshots each; A10x: 25,000snapshots (see Section A.1 for details).

Evaluation: Ground truth (referred to as "truth" hereafter) with 5 independent sequences (250,000 snapshots each), compared with 5 emulator-generated sequences of equal length to test for gray-swan extremes (see Section A.1 for details).

Diffusion Emulators: U-Net DDPM with circular padding. *Static Emulator* samples instantaneous states, conditioned on grid coordinates (x, y). *Dynamic Emulator* autoregressively predicts $(u_{t+\Delta t}, v_{t+\Delta t})$ conditioned on (u_t, v_t) where Δt is the emulator timestep (see Figure 1(b) and Section A.2 for details).

Iterative Training: We use two strategies on the training dataset B. *all gen*: cumulative augmentation, where an emulator is trained on the original data plus all synthetic data from generations. *last gen* trained only on the synthetic data from immediately preceding generation (see Figure 1(e-f) and Section A.3 for details).

3 Results and Discussion

Our analysis reveals that both the static and dynamic emulators only partially satisfy requirement (1) (see Section C.1). This calls for more work on building stable, physically consistent emulators with outputs that accurately reproduce the system's mean statistics. While the dynamic emulator proves to be stable (a metric not applicable to the static emulator), neither is physically consistent, as for

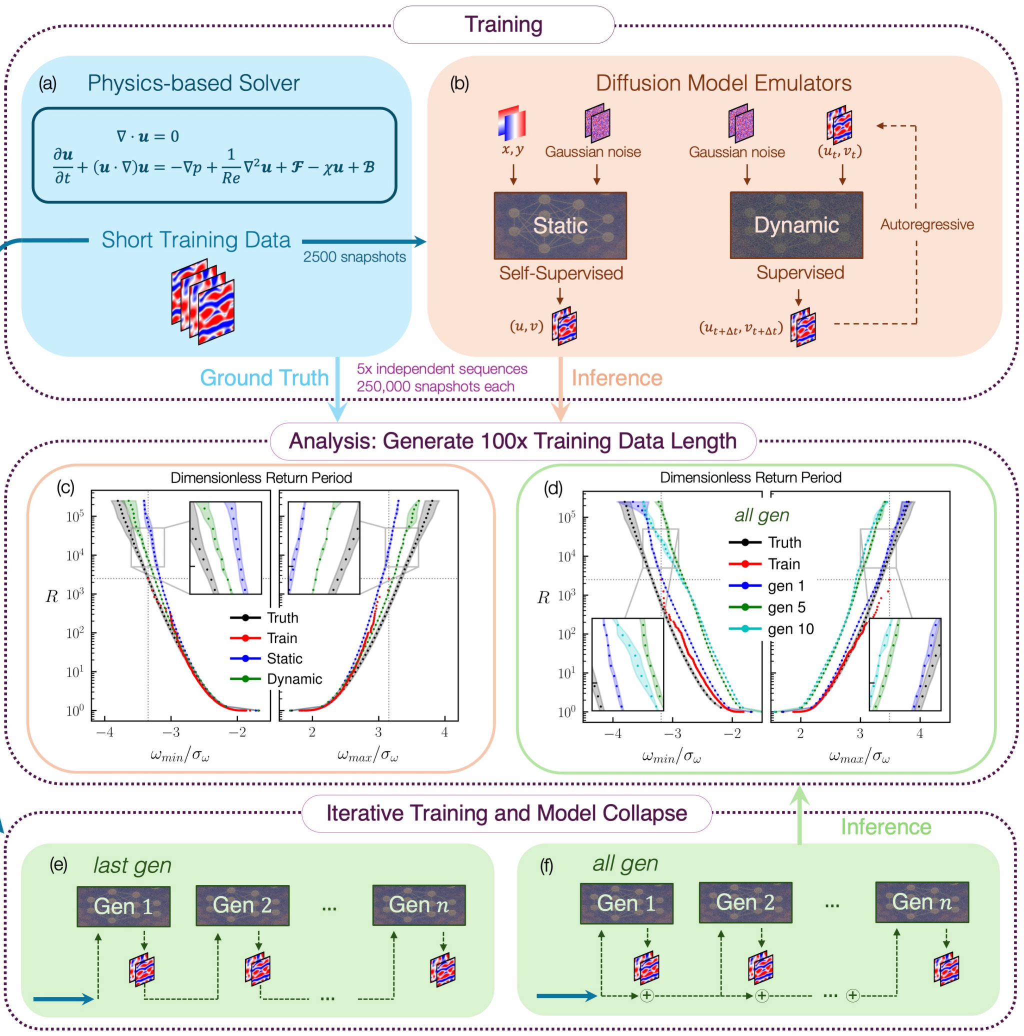

Training
(a) Physics-based Solver
$\nabla \cdot \boldsymbol{u} = 0$
$\frac{\partial \boldsymbol{u}}{\partial t} + (\boldsymbol{u}\cdot\nabla)\boldsymbol{u} = -\nabla p + \frac{1}{Re}\nabla^2 \boldsymbol{u} + \mathcal{F} - \chi \boldsymbol{u} + \mathcal{B}$
Short Training Data
2500 snapshots
(b) Diffusion Model Emulators
x, y
Gaussian noise
Gaussian noise
(u_t, v_t)
Static
Dynamic
Autoregressive
Self-Supervised
Supervised
(u, v)
$(u_{t+\Delta t}, v_{t+\Delta t})$
Ground Truth
5x independent sequences
250,000 snapshots each
Inference
Analysis: Generate 100x Training Data Length
(c) Dimensionless Return Period
Truth
Train
Static
Dynamic
R
$\omega_{min}/\sigma_\omega$
$\omega_{max}/\sigma_\omega$
(d) Dimensionless Return Period
all gen
Truth
Train
gen 1
gen 5
gen 10
Iterative Training and Model Collapse
Inference
(e) last gen
Gen 1
Gen 2
Gen n
(f) all gen
Gen 1
Gen 2
Gen n

Figure 1: **Experimental Design:** We train conditional diffusion emulators on very short datasets and evaluate their ability to predict previously unseen extremes. (a) A physics-based numerical solver is used to generate training data and a 100× longer ground truth. (b) Two types of diffusion model architectures are considered: (1) a self-supervised static diffusion model (static) that generates samples of flow fields conditioned on spatial grid coordinates, and (2) an autoregressive conditional diffusion model (dynamic) that rolls forward in time conditioned on the previous temporal step. (c) Both trained models are used to generate emulation datasets 100x longer than the training dataset. Return period curves, R, are computed to assess their ability to predict extremes. (d-f) **Model Collapse:** Iteratively training static emulator models on their own output leads to model collapse. Two configurations are explored: (e) *last gen*, where only data from the immediately preceding generation is used for training, and (f) *all gen*, where data from all previous generations are cumulatively combined for training. As shown in (d), the quality of R progressively degrades for *all gen*.

example, they fail to maintain the divergence-free condition, $\nabla \times \boldsymbol{u} \neq 0$. Furthermore, their ability to reproduce the truth's means statistics are fundamentally limited; they replicate the statistics of their training set, which can differ from the truth (Section C.1 and Figure 4.

3.1 Extreme Event Prediction and Return Period Analysis

The central question of this study is how well the emulators can predict extreme events that were absent from the training dataset (the gray swans). We address this by comparing the distribution of extremes in the emulator-generated data against that of the truth. Specifically, to characterize the frequency of rare events, we use return period curves, a common measure in theoretical and applied investigation of climate extremes.

Dimensionless Return Period: We define return period as $R = 1/\mathbb{P}(E)$. Here, E is a scalar value representing the single maximum (or minimum) value found across an entire 2D snapshot of the flow state. To empirically estimate R, we first create a time series by extracting the maximum (or minimum) value from each snapshot in our simulations. These scalar maxima (or minima) are then ranked to compute their exceedance probability, $\mathbb{P}(E)$. This entire procedure is performed separately for both the maxima and minima of the flow fields. Return periods are also computed for *debiased* flow states (anomalies). These are calculated by subtracting the 2D temporal mean from the flow state, $u - \langle u \rangle$, and computing extremes of these anomalies. The means are calculated separately for the truth, training set, and emulated dataset to remove the effect of the mean biases on the return period.

Note that because the static emulator does not have an explicit notion of time, the true time-dependent return period cannot be calculated. The metric $1/\mathbb{P}(E)$ nonetheless provides a meaningful way to quantify the extremeness of an event, allowing the performance of the static and dynamic emulators to be directly compared.

The return periods are calculated for both the prognostic u flow states output by the emulator (Figure 2(b)), the diagnostic flow state ω (Figure 2(a,c,e)), and debiased ω (Figure 2(d,f)). Note that ω can be thought of as "weather" in this canonical flow: positive and negative vorticity correspond to low-pressure (cyclonic) and high-pressure (anticyclonic) systems. The return period curves reveal the following key points about the emulated extreme events:

- Both static and dynamic emulators demonstrate an ability to generate events more intense than those in the training set. While this might suggest that geneative emulators can produce "gray swans", we caution that some of these large values might be due to unphysical artifacts. Figure 2(a) shows an example of this problem, which is an issue for ω, a diagnostic variable involving differentiation. However, examining more cases, particularly for the prognostic variable u, demonstrates that these generative emulators are indeed not limited by the training dataset's maximum or minimum values, showing the ability to "extrapolate".
- The emulators fail to reproduce the correct statistics (frequency or return period) of these extrapolated events, deviating from the truth return periods for the most extreme values.
- The predicted return periods are strongly constrained by the distribution of the training dataset. This is evident with dataset B, where the return periods for u in the training dataset were significantly larger than those of the truth 2(b). Consequently, both the static and dynamic emulators produced events that closely tracked the training dataset's biased return period curve, failing to match the truth. The dynamic emulator also struggled to capture extremes of ω for this dataset (see 2(a)). This is because of poor training (due to a small dataset), which resulted in unphysical artifacts in ω.
- The dynamic emulator predicts extremes more accurately than the static emulator for dataset A and A10$\times$. Its return period curve for high-amplitude extremes lies closer to the truth; this may be due to the inductive bias of autoregressive models for learning dynamics [Cheng and Weare, 2024]. It was not true for dataset B because of the reasons mentioned above (Figure 2(c,e)), and this conclusion holds even after removing the mean bias from the flow states (Figure 2(d,f)). The dynamic model's superior performance may be attributed to the inductive bias of autoregressive models for learning dynamics [Cheng and Weare, 2024]. In contrast, this advantage was not observed for dataset B due to the poor training of the dynamic emulator (Figure 2(a)).

- The extrapolation of return periods for OOD extreme events is not robust when using smaller training datasets, showing high sensitivity to stochastic training elements like weight initialization. This indicates that a simple autoregressive model struggles without sufficient data. However, this challenge could be overcome by increasing the training dataset tenfold. A direct comparison shows that the return period curves for the large dataset, A10x (Figure 2(e,f)), are closer to the truth than those for the smaller dataset, A (Figure 2(c,d)). The larger dataset provides the emulator with enough examples of extremes to learn a tail distribution that more accurately reflects the true system dynamics.

3.2 Mitigating Data Scarcity: Iterative Training and Model Collapse

Given the challenges above, a practical question arises: can we extend the training data using the emulator itself, e.g, to expose it to more extreme events? This idea is conceptually similar to bootstrapping and speculated in some studies as a way of addressing the small-data regime in climate applications [Watson-Parris, 2021, Dueben et al., 2022, Lai et al., 2024]. We investigate this approach through iterative training as described in Methods Section A.3. Our results show that this approach fails to solve the data scarcity issue and ultimately leads to model collapse.

In *all gen* strategy, where the training dataset includes the original real data plus all synthetic data from every previous generation, the emulator's performance gradually degrades over generations. The energy spectrum showed a progressive loss of power at higher wavenumbers, which was noticeable by generation 5 and significant by generation 10 (Figure 3(a)). While the 2D long-time mean of u in generation 5 remained similar to the original one, it started to degrade by generation 10 (Figure 3(a)). Crucially, this data augmentation strategy did not improve the prediction of extreme events. By generation 5, the model began underpredicting the rarest events and showed no subsequent improvements. The fact that iterative training with more synthetic data did not improve, but rather degraded, the return period curves confirms that this data augmentation approach cannot solve the small-data problem. The apparent good performance of some metrics (like the gen 5 temporal mean of u) can be misleading, highlighting the importance of a holistic evaluation of emulators' performance.

The *last gen* strategy, a stress test where each new model was trained only on the previous generation's synthetic data, demonstrated a much faster failure. The model began to diverge after just one iteration. The output from the gen 2 emulator (trained solely on gen 1's synthetic data) was visibly degraded (Figure 1b). By gen 3, the emulator had effectively collapsed, unable to reproduce the 2D temporal mean of u (Figure 3(b)). The energy spectrum reveals that gen 2 spectrum shows an increase in energy at high wavenumbers. By generation 10, the spectrum was corrupted across all scales.

Our findings clearly show that iterative training of the emulator is not a viable strategy for overcoming data scarcity. Instead of improving performance, this self-consuming loop leads to model collapse. The degradation is gradual when the original data is retained (*all gen*) but catastrophic and rapid when it is discarded (*last gen*). This demonstrates that synthetic data generated by the model itself progressively loses fidelity and introduces artifacts that corrupt subsequent training cycles.

4 Conclusion

We explored the capability of diffusion emulators to quantify the statistics of rare extreme events by training emulators on a small amount of high-fidelity data in 2D geophysical turbulence, a canonical test case for atmospheric and oceanic turbulence. We examined three key requirements for extreme event emulators: (1) stable, physically consistent outputs reproducing mean statistics, (2) generation of OOD ("gray swan") extreme events, and (3) reproduction of the correct statistics (frequency or return period) of those extremes. We evaluated two strategies: the first builds two emulators, a static (distribution sampling) and a dynamic (autoregressive) emulator, and the second uses iterative training on synthetic data generated by the emulators. By comparing statistics from emulator-generated data ($100\times$ the training dataset) to a high-fidelity truth simulation of equal length, we aim to examine these emulators' ability to satisfy requirements (1)-(3). Our study yields several takeaways:

- Diffusion emulators can produce events more extreme than those present in their training data, demonstrating an ability to extrapolate beyond the training data distribution, satisfying requirement (2). Notably, the dynamic emulator showed a slight advantage in generating high-magnitude events over the static emulator, likely because it can learn the underlying fast

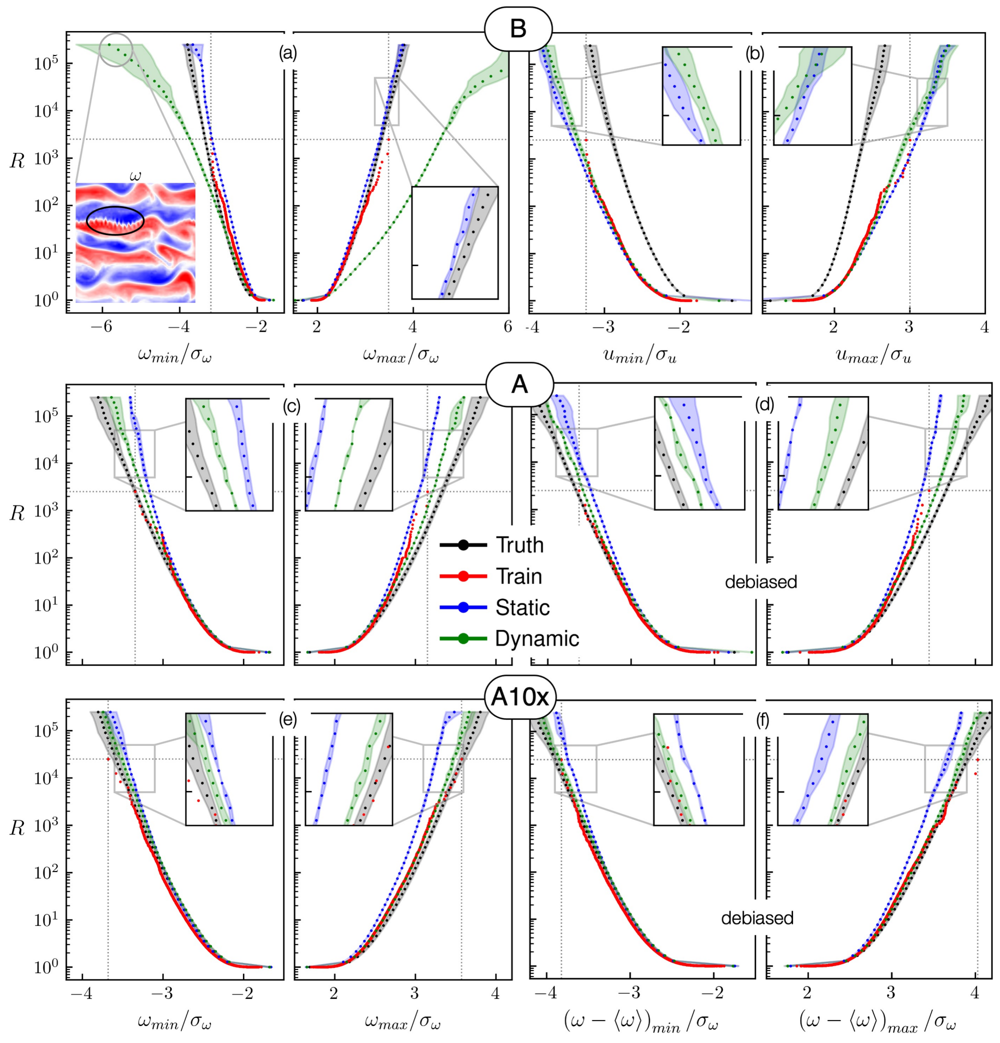


Figure 2: **Dimensionless return periods for extremes** (R)**.** Emulators generate extremes beyond those in the training dataset but fail to reliably capture the correct frequency. R shown for training datasets B (a–b), A (c-d), and A10x (e-f). (a) The dynamic emulator overestimates the R of ω; this is explained by the unphysical artifacts appearing in ω shown in the inset for such extremes, indicating inaccurate learning of dynamics in the small-data regime. (b) The R of u for both emulators is biased towards the training set, which is distinct from the truth. (c-f) R was computed both from raw outputs and from debiased flow states to account for systematic biases. The dynamic emulator is closer to the truth than the static emulator for the R of ω compared to the dynamic emulator for R of ω (c,e) and debiased ω (d,f), indicating that the dynamic emulator is learning the underlying dynamics. The return period curves for the large dataset, A10x (Figure 2(e,f)), are closer to the truth than those for the smaller dataset A (Figure 2(c,d)). In each panel, the dotted grey line corresponds to the most extreme values present in the training dataset. Shaded bands denote sampling variability (25th-75th percentiles), and the insets zoom in on the high return period range. All values are normalized by the corresponding standard deviation, σ.

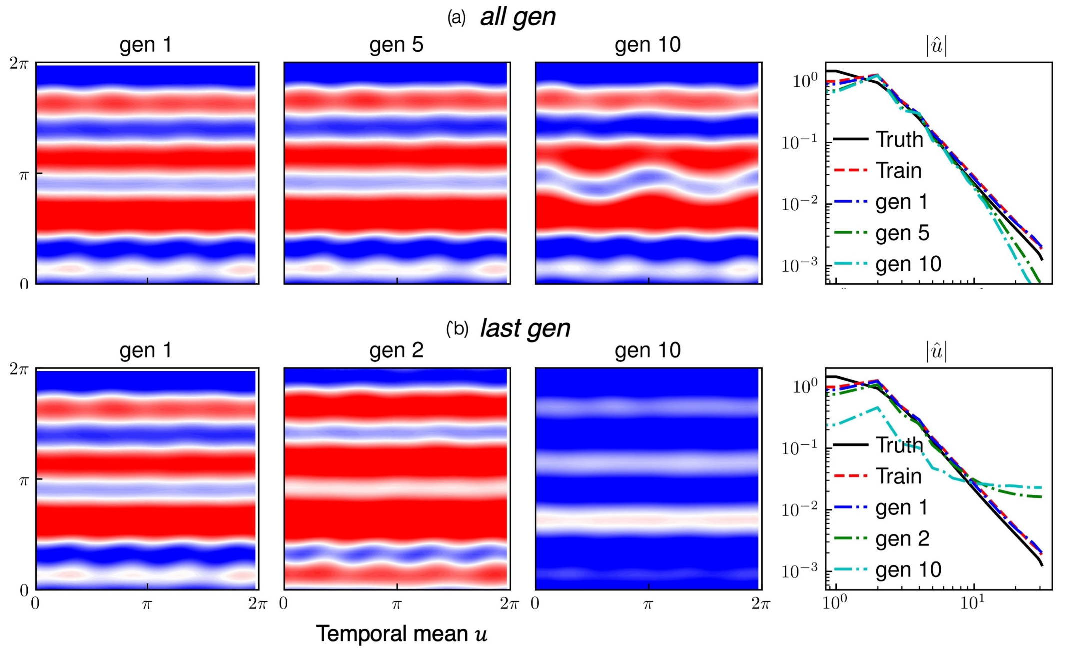


Figure 3: **Model collapse in the static emulator**. Contours of the temporal mean of u and corresponding spectra for successive emulator generations for training dataset B. (a) ***all gen:*** Each generation is trained on the cumulative outputs from all previous emulators. Drift is slower with gens 1–5 remaining close to training data, while gen 10 shows emerging discrepancies in spectra at both low and high wavenumbers. (b) ***last gen:*** Each generation is trained only on data from the immediately preceding emulator. Collapse appears by gen 2, with a strong mean bias and an increase in high-wavenumber energy; by gen 10, the degradation is severe. Figure 1 shows the return period of *all gen*, confirming that the performance degrades by gen 10.

dynamics. However, some of these OOD extremes may be unphysical artifacts, highlighting the critical need for holistic evaluation across multiple physical and statistical metrics. In contrast, the static emulator treats each snapshot independently, missing temporal causation, though it retains an advantage in predicting mean flow statistics. This performance gap underscores that the choice to formulate an AI emulator as a time-stepper versus a one-shot sampler has a significant impact on its ability to capture rare events. We therefore recommend further testing of such autoregressive models on more complex systems (e.g., global weather or climate models) to verify whether this advantage holds broadly.

- Both emulators' representation of extremes is fundamentally constrained by their training dataset. If the training dataset is too short or unrepresentative of the truth's mean and variability, the emulator will inherit and reproduce those biases, failing to match the truth's mean statistics and extreme event frequencies. While increasing the training data size tenfold improved performance, it underscores a central challenge: accurately capturing rare events may require observing them first!

- Iterative training on synthetic data leads to model collapse. Using the emulator to generate more data for subsequent training is not a viable solution for data scarcity. This self-consuming feedback loop degrades performance over generations, corrupts the energy spectrum, and fails to improve extreme event prediction.

- One intriguing direction is to explicitly tune the diffusion model to produce more (or fewer) extremes. Some studies have proposed modifying the diffusion sampling process (e.g., adjusting noise levels or using guided sampling) to bias generation towards rare events. Developing methods to steer generative models toward the tails (e.g., classifier-free guidance [Ho and Salimans, 2022]) could significantly aid the study of extremes.

- While our emulators maintain some physical consistency, they fail to enforce all system constraints (e.g., incompressibility). Incorporating explicit physical constraints or conditioning may be necessary to satisfy requirement (3).

Taken together, our results provide only a partial fulfillment of the three viability requirements of AI-based emulators. The emulators are generally stable and may produce mean statistics (requirement 1), and they can generate OOD extremes (requirement 2), with the dynamic model outperforming the static model in peak magnitudes. However, neither design reproduces the correct frequency of rare events (requirement 3). Performance remains limited by the representativeness of the training data, and iterative retraining on synthetically generated data fails to improve return periods and can induce model collapse. We therefore recommend cautious use of generative diffusion emulators for quantitative extreme-risk estimation, and focusing future work on longer, more representative training sets together with physics-aware objectives, evaluated with the same diagnostics used here (including return period curves and physical consistency checks).

References

Valerio Lucarini and Mickaël D Chekroun. Theoretical tools for understanding the climate crisis from hasselmann's programme and beyond. *Nature Reviews Physics*, 5(12):744–765, 2023.

Catherine O de Burgh-Day and Tennessee Leeuwenburg. Machine learning for numerical weather and climate modelling: a review. *Geoscientific Model Development*, 16(22):6433–6477, 2023.

Stefano Materia, Lluís Palma García, Chiem van Straaten, Sungmin O, Antonios Mamalakis, Leone Cavicchia, Dim Coumou, Paolo de Luca, Marlene Kretschmer, and Markus Donat. Artificial intelligence for climate prediction of extremes: State of the art, challenges, and future perspectives. *Wiley Interdisciplinary Reviews: Climate Change*, 15(6):e914, 2024.

Ching-Yao Lai, Pedram Hassanzadeh, Aditi Sheshadri, Maike Sonnewald, Raffaele Ferrari, and Venkatramani Balaji. Machine learning for climate physics and simulations. *Annual Review of Condensed Matter Physics*, 16, 2024.

Annalisa Bracco, Julien Brajard, Henk A Dijkstra, Pedram Hassanzadeh, Christian Lessig, and Claire Monteleoni. Machine learning for the physics of climate. *Nature Reviews Physics*, 7(1):6–20, 2025.

Stuart Coles, Joanna Bawa, Lesley Trenner, and Pat Dorazio. *An introduction to statistical modeling of extreme values*, volume 208. Springer, 2001.

Francesco Ragone, Jeroen Wouters, and Freddy Bouchet. Computation of extreme heat waves in climate models using a large deviation algorithm. *Proceedings of the National Academy of Sciences*, 115(1):24–29, 2018.

Francesco Ragone and Freddy Bouchet. Computation of extreme values of time averaged observables in climate models with large deviation techniques. *Journal of Statistical Physics*, 179(5):1637–1665, 2020.

Robert J Webber, David A Plotkin, Morgan E O'Neill, Dorian S Abbot, and Jonathan Weare. Practical rare event sampling for extreme mesoscale weather. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 29(5), 2019.

Dorian S Abbot, Robert J Webber, Sam Hadden, Darryl Seligman, and Jonathan Weare. Rare event sampling improves mercury instability statistics. *The Astrophysical Journal*, 923(2):236, 2021.

Justin Finkel, Edwin P Gerber, Dorian S Abbot, and Jonathan Weare. Revealing the statistics of extreme events hidden in short weather forecast data. *AGU Advances*, 4(2):e2023AV000881, 2023.

Eric Vanden-Eijnden and Jonathan Weare. Rare event simulation of small noise diffusions. *Communications on Pure and Applied Mathematics*, 65(12):1770–1803, 2012.

Claudia Gessner, Erich M Fischer, Urs Beyerle, and Reto Knutti. Very rare heat extremes: quantifying and understanding using ensemble reinitialization. *Journal of Climate*, 34(16):6619–6634, 2021.

Francesco Ragone and Freddy Bouchet. Rare event algorithm study of extreme warm summers and heatwaves over europe. *Geophysical Research Letters*, 48(12):e2020GL091197, 2021.

Gustau Camps-Valls, Miguel-Ángel Fernández-Torres, Kai-Hendrik Cohrs, Adrian Höhl, Andrea Castelletti, Aytac Pacal, Claire Robin, Francesco Martinuzzi, Ioannis Papoutsis, Ioannis Prapas, et al. Artificial intelligence for modeling and understanding extreme weather and climate events. *Nature Communications*, 16(1):1919, 2025.

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.

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.

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.

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.

Salva Rühling Cachay, Bo Zhao, Hailey Joren, and Rose Yu. Dyffusion: A dynamics-informed diffusion model for spatiotemporal forecasting. *Advances in neural information processing systems*, 36:45259–45287, 2023.

Simon Lang, Mihai Alexe, Matthew Chantry, Jesper Dramsch, Florian Pinault, Baudouin Raoult, Mariana CA Clare, Christian Lessig, Michael Maier-Gerber, Linus Magnusson, et al. Aifs–ecmwf's data-driven forecasting system. *arXiv preprint arXiv:2406.01465*, 2024.

Tung Nguyen, Rohan Shah, Hritik Bansal, Troy Arcomano, Sandeep Madireddy, Romit Maulik, Veerabhadra Kotamarthi, Ian Foster, and Aditya Grover. Scaling transformers for skillful and reliable medium-range weather forecasting. In *ICLR 2024 Workshop on AI4DifferentialEquations In Science*, 2024.

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.

Duncan Watson-Parris. Machine learning for weather and climate are worlds apart. *Philosophical Transactions of the Royal Society A*, 379(2194):20200098, 2021.

Tianyi Li, Samuele Tommasi, Michele Buzzicotti, Fabio Bonaccorso, and Luca Biferale. Generative diffusion models for synthetic trajectories of heavy and light particles in turbulence. *International Journal of Multiphase Flow*, 181:104980, 2024a.

Ning Lin and Kerry Emanuel. Grey swan tropical cyclones. *Nature Climate Change*, 6(1):106–111, 2016.

Y Qiang Sun, Pedram Hassanzadeh, Mohsen Zand, Ashesh Chattopadhyay, Jonathan Weare, and Dorian S Abbot. Can ai weather models predict out-of-distribution gray swan tropical cyclones? *Proceedings of the National Academy of Sciences*, 122(21):e2420914122, 2025a.

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.

Surya Dheeshjith, Adam Subel, Alistair Adcroft, Julius Busecke, Carlos Fernandez-Granda, Shubham Gupta, and Laure Zanna. Samudra: An ai global ocean emulator for climate. *Geophysical Research Letters*, 52(10): e2024GL114318, 2025.

William E Chapman, John S Schreck, Yingkai Sha, David John Gagne II, Dhamma Kimpara, Laure Zanna, Kirsten J Mayer, and Judith Berner. Camulator: Fast emulation of the community atmosphere model. *arXiv preprint arXiv:2504.06007*, 2025.

Oliver Watt-Meyer, Brian Henn, Jeremy McGibbon, Spencer K Clark, Anna Kwa, W Andre Perkins, Elynn Wu, Lucas Harris, and Christopher S Bretherton. Ace2: accurately learning subseasonal to decadal atmospheric variability and forced responses. *npj Climate and Atmospheric Science*, 8(1):205, 2025.

Thomas Rackow, Nikolay Koldunov, Christian Lessig, Irina Sandu, Mihai Alexe, Matthew Chantry, Mariana Clare, Jesper Dramsch, Florian Pappenberger, Xabier Pedruzo-Bagazgoitia, et al. Robustness of ai-based weather forecasts in a changing climate. *arXiv preprint arXiv:2409.18529*, 2024.

Y Qiang Sun, Pedram Hassanzadeh, Tiffany Shaw, and Hamid A Pahlavan. Predicting beyond training data via extrapolation versus translocation: Ai weather models and dubai's unprecedented 2024 rainfall. *arXiv preprint arXiv:2505.10241*, 2025b.

Zilu Meng, Gregory J Hakim, Wenchang Yang, and Gabriel A Vecchi. Deep learning atmospheric models reliably simulate out-of-sample land heat and cold wave frequencies. *arXiv preprint arXiv:2507.03176*, 2025.

Zhongwei Zhang, Erich Fischer, Jakob Zscheischler, and Sebastian Engelke. Numerical models outperform ai weather forecasts of record-breaking extremes. *arXiv preprint arXiv:2508.15724*, 2025.

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

Georgios Batzolis, Jan Stanczuk, Carola-Bibiane Schönlieb, and Christian Etmann. Conditional image generation with score-based diffusion models. *arXiv preprint arXiv:2111.13606*, 2021.

Marten Lienen, David Lüdke, Jan Hansen-Palmus, and Stephan Günnemann. From zero to turbulence: Generative modeling for 3d flow simulation. *arXiv preprint arXiv:2306.01776*, 2023.

Tianyi Li, Luca Biferale, Fabio Bonaccorso, Martino Andrea Scarpolini, and Michele Buzzicotti. Synthetic lagrangian turbulence by generative diffusion models. *Nature Machine Intelligence*, 6(4):393–403, 2024b.

Tim Whittaker, Romuald A Janik, and Yaron Oz. Turbulence scaling from deep learning diffusion generative models. *Journal of Computational Physics*, 514:113239, 2024.

Noah D Brenowitz, Tao Ge, Akshay Subramaniam, Aayush Gupta, David M Hall, Morteza Mardani, Arash Vahdat, Karthik Kashinath, and Michael S Pritchard. Climate in a bottle: Towards a generative foundation model for the kilometer-scale global atmosphere. *arXiv preprint arXiv:2505.06474*, 2025.

Georg Kohl, Li-Wei Chen, and Nils Thuerey. Benchmarking autoregressive conditional diffusion models for turbulent flow simulation. *arXiv preprint arXiv:2309.01745*, 2023.

Stamatis Stamatelopoulos and Themistoklis P Sapsis. Can diffusion models capture extreme event statistics? *Computer Methods in Applied Mechanics and Engineering*, 435:117589, 2025.

Seth Bassetti, Brian Hutchinson, Claudia Tebaldi, and Ben Kravitz. Diffesm: Conditional emulation of temperature and precipitation in earth system models with 3d diffusion models. *Journal of Advances in Modeling Earth Systems*, 16(10):e2023MS004194, 2024.

Antreas Antoniou, Amos Storkey, and Harrison Edwards. Data augmentation generative adversarial networks. *arXiv preprint arXiv:1711.04340*, 2017.

Sina Alemohammad, Josue Casco-Rodriguez, Lorenzo Luzi, Ahmed Imtiaz Humayun, Hossein Babaei, Daniel LeJeune, Ali Siahkoohi, and Richard G Baraniuk. Self-consuming generative models go mad. International Conference on Learning Representations (ICLR), 2024.

Ilia Shumailov, Zakhar Shumaylov, Yiren Zhao, Nicolas Papernot, Ross Anderson, and Yarin Gal. Ai models collapse when trained on recursively generated data. *Nature*, 631(8022):755–759, 2024.

Yunzhen Feng, Elvis Dohmatob, Pu Yang, Francois Charton, and Julia Kempe. Beyond model collapse: Scaling up with synthesized data requires reinforcement. In *ICML 2024 workshop on theoretical foundations of foundation models*, 2024.

Elvis Dohmatob, Yunzhen Feng, Pu Yang, Francois Charton, and Julia Kempe. A tale of tails: Model collapse as a change of scaling laws. *arXiv preprint arXiv:2402.07043*, 2024.

Joshua Kazdan, Stanford Statistics, Rylan Schaeffer, Apratim Dey, Matthias Gerstgrasser, Rafael Rafailov, David Donoho, and Sanmi Koyejo. Accumulating data avoids model collapse.

Karan Jakhar, Yifei Guan, Rambod Mojgani, Ashesh Chattopadhyay, and Pedram Hassanzadeh. Learning closed-form equations for subgrid-scale closures from high-fidelity data: Promises and challenges. *Journal of Advances in Modeling Earth Systems*, 16(7):e2023MS003874, 2024a.

Yifei Guan, Ashesh Chattopadhyay, Adam Subel, and Pedram Hassanzadeh. Stable a posteriori les of 2d turbulence using convolutional neural networks: Backscattering analysis and generalization to higher re via transfer learning. *Journal of Computational Physics*, 458:111090, 2022.

Chris Pedersen, Laure Zanna, and Joan Bruna. Thermalizer: Stable autoregressive neural emulation of spatiotemporal chaos. *arXiv preprint arXiv:2503.18731*, 2025.

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

Romit Maulik, Omer San, Adil Rasheed, and Prakash Vedula. Subgrid modelling for two-dimensional turbulence using neural networks. *Journal of Fluid Mechanics*, 858:122–144, 2019.

Geoffrey K Vallis. *Atmospheric and oceanic fluid dynamics*. Cambridge University Press, 2017.

Xiaoou Cheng and Jonathan Weare. The surprising efficiency of temporal difference learning for rare event prediction. *Advances in Neural Information Processing Systems*, 37:81257–81286, 2024.

Peter D Dueben, Martin G Schultz, Matthew Chantry, David John Gagne, David Matthew Hall, and Amy McGovern. Challenges and benchmark datasets for machine learning in the atmospheric sciences: Definition, status, and outlook. *Artificial Intelligence for the Earth Systems*, 1(3):e210002, 2022.

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

Karan Jakhar, Rambod Mojgani, Moein Darman, Yifei Guan, and Pedram Hassanzadeh. py2d: High-performance 2D Navier-Stokes solver in python (version 0.1), 2024b. URL `https://github.com/envfluids/py2d`.

Tom Beucler, Stephan Rasp, Michael Pritchard, and Pierre Gentine. Achieving conservation of energy in neural network emulators for climate modeling. *arXiv preprint arXiv:1906.06622*, 2019.

A Experimental Details

A.1 Ground Truth and Training Dataset

We train the emulators on the (u, v) flow fields. Training data is generated using "py2d" [Jakhar et al., 2024b] on a 64^2 grid, with two short training datasets with uncorrelated initial conditions and 2,500 snapshots each (called datasets A and B), and a 10x longer dataset with 25,000 snapshots (dataset A10x). Datasets A and B were chosen so that B's mean and extreme statistics deviate much more from the ground truth than A's. *This setup tests whether the emulator simply mimics the training data's statistics (e.g., distribution), or learns the dynamics or a meaningful representation of the ground truth* (even when trained with a short dataset). The larger A10x dataset is more representative of the ground truth statistics by virtue of its length. We operate in the small-data regime deliberately to mimic the real-world application. 2,500 snapshots were about the minimum required to train a working static emulator, making this a stringent test of learning from scarce data. Indeed, training in this regime was challenging; both emulators' performance was highly sensitive to hyperparameters and initialization, sometimes failing altogether. In contrast, the 10-times longer dataset A10x yielded more stable training and more consistent results across runs.

To evaluate long-term performance, we generate ground-truth (referred to as "truth" hereafter) much longer than the training data length: Starting from various random initial conditions, we run the numerical solver to produce an ensemble of 5 independent sequences, each 100 times the training length, i.e, 250,000 snapshots each. After training, each emulator (static and dynamic) is used to generate 5 independent sequences, each 250,000 snapshots in length. This approach allows robust uncertainty quantification for the statistics of rare events. We will show in Section 3.1 that the extremes of interest (gray swan events) in these long runs were never seen during training – they emerge only in the 100x longer truth and emulation datasets.

A.2 Diffusion Emulators

We adopt the Denoising Diffusion Probabilistic Model (DDPM) framework for generative modeling of the turbulent flow fields. Our network architecture is a U-Net convolutional neural network similar to that of Ho et al. (2020), which progressively downscales and upscales the input [Ho et al., 2020, Batzolis et al., 2021]. We use circular padding to avoid boundary artifacts, which had introduced artificial extremes during initial tests. We train two variants of the diffusion-based emulator corresponding to different ways of conditioning the model:

Static emulator: The static diffusion emulator learns the distribution of instantaneous states. It does not produce a time-evolving trajectory; i.e., it is a climate distribution sampler rather than a simulator [Brenowitz et al., 2025, Li et al., 2024b, Whittaker et al., 2024, Li et al., 2024a]. This emulator treats each snapshot in time independently. It is trained to generate a state of the flow (the velocity fields (u, v) on the grid) without temporal context. The model is conditioned on a constant field of 2D grid coordinates, (x, y), concatenated with the velocity field, (u, v). This conditioning prevents spatial drift, a problem that arises because the circular padding used to enforce periodicity causes the U-Net to lose its sense of absolute location.

Dynamic emulator: This dynamic emulator is an autoregressive conditional diffusion model emulator designed to mimic a time-stepping solver by producing the next state given the current state [Kohl et al., 2023, Lienen et al., 2023, Price et al., 2025, Rühling Cachay et al., 2023]. We frame the diffusion model to output $(u_{t+\Delta t}, v_{t+\Delta t})$ conditioned on (u_t, v_t). Here, Δt (the ML emulator's time step) is chosen much larger than the numerical solver's step – specifically, $\Delta t = 6 \times 10^{-2}$, which is 120 times the solver's base time step, $\Delta t_{\text{solver}} = 5 \times 10^{-4}$.

Table 1: Hyperparameters for training emulators.

Component	Setting
Architecture	Periodic UNet; down block [32,64,128,256]; mid block [256,256,128];
Conditioning	static: Spatial grid (x, y); dynamic: Previous state (u_{t-1}, v_{t-1})
Diffusion steps	1000 (linear noise schedule)
Optimizer	AdamW
LR schedule	CosineAnnealingLR (peak 3×10^{-4}, min 1×10^{-6})
Epochs	800
Seeds	Best out of 4
Loss	noise, $\|\epsilon - \epsilon_\theta\|^2$

Grid coordinates as conditioning are not added to the dynamic emulator since they are found to provide no improvement in performance.

For each emulator type and dataset, we train with 4 random seeds. Training details are mentioned in Table 1.

A.3 Iterative Training and Model Collapse

We designed two experiments using the static emulator on the more challenging training dataset B. The first experiment (all gen) tests the viability of using synthetic data for augmentation, while the second (last gen) is a stress test of the iterative training process and the quality of the generative emulators' output.

all gen: This experiment follows a cumulative data augmentation strategy. Here, we begin with generation 1 (gen 1) trained on the *original* dataset B. Then, during inference, generate a synthetic dataset of the same length as the training dataset (2500 snapshots). Next, we train a new static emulator on a combined dataset of the original data and gen 1's synthetic data output (5000 total snapshots). This gen 2 emulator generates another 2500 samples. Train a new emulator using a combined dataset of original, gen 1, and gen 2 data (7500 total snapshots), generate another 2500 samples (gen 3), and so forth. In each cycle, the newest emulator is trained on the original dataset plus all the synthetic data from preceding generations.

last gen:: This experiment is designed to reveal the limits of iterative retraining and measure the system's susceptibility to model collapse. Here, we start with generation 1 (gen 1) trained on the *original* dataset B. Then, during inference, generate a synthetic dataset of the same length as the training dataset (2500 snapshots). Then, train a new static emulator using only gen 1's output (synthetic data) and generate another 2500 samples (gen 2). Train a new emulator using the gen 2 data, generate another 2500 samples (gen 3), and so forth. In each cycle, the newest emulator sees only data produced by its immediate predecessor. This is an extreme test where errors may compound with each generation.

B Training Details

We found that training is sensitive to initialization, especially in the small-data regime (datasets A and B). Some seeds yielded significantly better models (analyzed in the context of mean statistics), reflecting the challenges of non-convex optimization with limited data. While training on the large dataset (A10x) was more robust, it still remained seed-dependent. All results reported here use the best performing seed on validation metrics for the respective configuration. This variability underlines that multiple trials may be needed to obtain a reliable emulator, especially when data are scarce.

Table 1 lists the hyperparameters used for training.

C Experiments

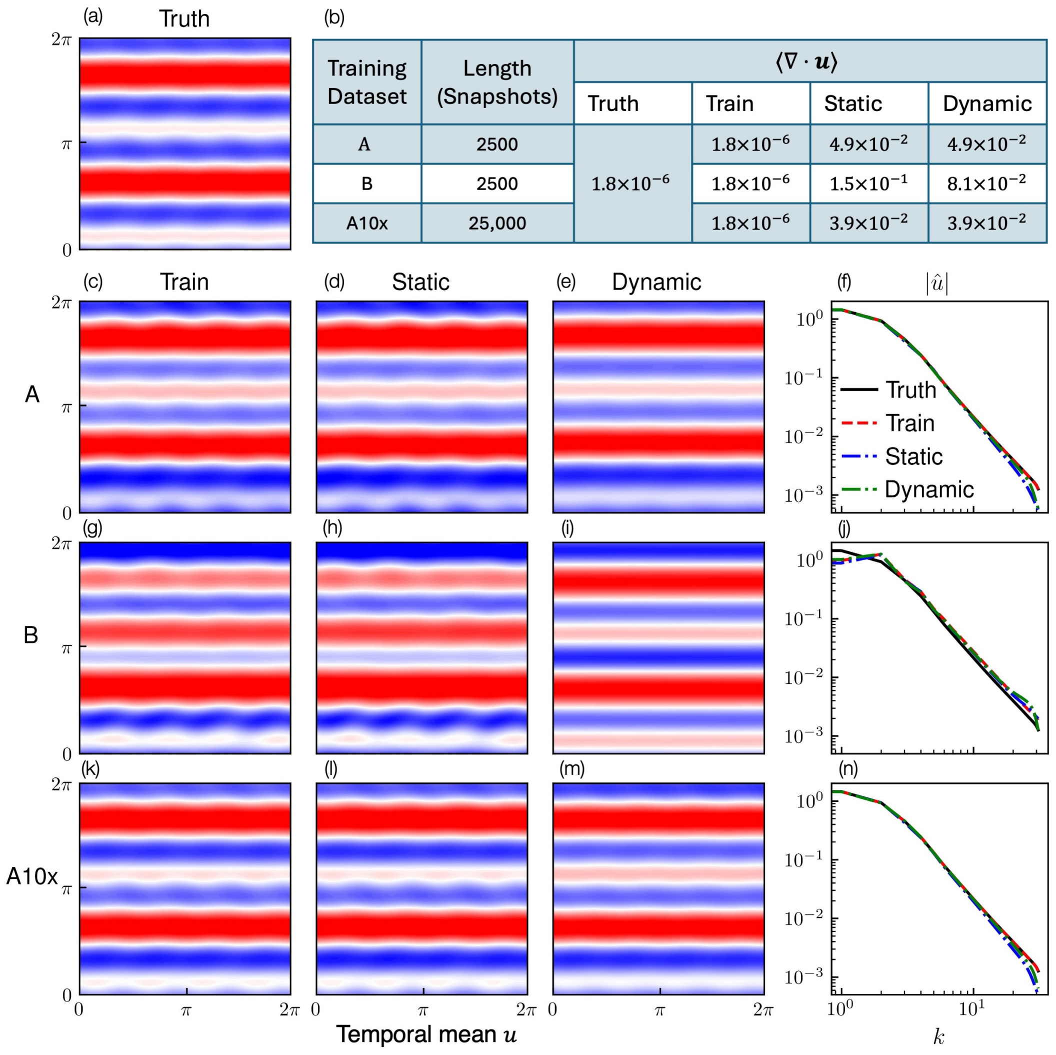
C.1 Mean Flow Statistics and Physical Consistency

Figure 4 compares the temporal mean u from the training data, emulator output, and the truth simulation for each case. For datasets A and B (2,500-snapshot training), the emulated means coincide with the training means, which differ from the true climate mean; in case A10x (25,000-snapshot training), the training mean was already close to the true mean, and accordingly, the emulator's mean aligns with both. The emulator spectra for all cases align with that of the respective training data, especially for dataset B, where the training data spectrum does not overlap with the truth spectrum at low wavenumbers. This suggests that the emulators faithfully reproduce the statistical features present in their training data rather than the truth.

Dataset B proved harder to train on, and its statistics deviate further from the long-term truth. The spectra of Dataset B are different from the truth (Figure 4(j)), while this is not the case for Dataset A (Figure 4(f)). Additionally, the temporal mean of u for Dataset A is much closer to the truth than that of Dataset B (Figures 4(a, c, g)).

Additionally, the emulators did not enforce incompressibility; $\nabla \cdot \boldsymbol{u} = 0$ is a good test for physical consistency. The numerical solver's output is divergence-free by construction (to machine precision $\approx 10^{-6}$), but the divergence-free condition was not explicitly enforced for training emulators. Consequently, the generated velocity fields have a non-zero divergence (of order 10^{-2}) as mentioned in Figure 3. Although this did not affect the large-scale mean or energy spectrum, it could have implications for long-term statistics. Enforcing physical constraints in ML emulators is an active area of research [Beucler et al., 2019, Chapman et al., 2025, Watt-Meyer et al., 2025], and techniques such as divergence-penalizing loss terms could be explored.

Overall, for bulk statistics like mean and energy spectrum, emulators can perform well when the training data is representative of the truth. If the training period is too short or atypical, the emulator will faithfully reproduce those biases. This underscores the importance of training on data that captures the variability of the system.



(b)

Training Dataset	Length (Snapshots)	$\langle \nabla \cdot \boldsymbol{u} \rangle$			
		Truth	Train	Static	Dynamic
A	2500	1.8×10^{-6}	1.8×10^{-6}	4.9×10^{-2}	4.9×10^{-2}
B	2500		1.8×10^{-6}	1.5×10^{-1}	8.1×10^{-2}
A10x	25,000		1.8×10^{-6}	3.9×10^{-2}	3.9×10^{-2}

Figure 4: **Training datasets and emulator performance on mean statistics.** (a) Truth temporal mean of the velocity, u. (b) Table summarizing the three training datasets - A, B (2500 snapshots each), and A10x (25000 snapshots), together with divergence of velocity, $\nabla \cdot \mathbf{u}$. The training dataset and truth generated from the numerical solver are divergence-free up to single-precision round-off, 10^{-6}, whereas static and dynamic emulator outputs show divergence of order 10^{-2}, indicating that the emulators do not enforce incompressibility. (c-d, g-i, k-m) Contours of temporal mean of u corresponding to training dataset (c-d) A, (g-i) B, and (k-m) A10x. The static emulator best reproduces the training-set mean, with the dynamic emulator being relatively less accurate. (f,j,n) Spectra of u corresponding to training dataset A, B, A10x. Both emulators match the training dataset spectra across scales with limited spectral bias. (j) Spectra of training dataset B are distinct from truth at lower wavenumbers, both static and dynamic emulators follow training dataset's spectra.