DySLIM: Dynamics Stable Learning by Invariant Measure for Chaotic Systems

Yair Schiff [1] **Zhong Yi Wan** [2] **Jeffrey B. Parker** [2] **Stephan Hoyer** [2] **Volodymyr Kuleshov** [1] **Fei Sha** [2]
Leonardo Zepeda-Núñez [2 3]

Abstract

Learning dynamics from dissipative chaotic systems is notoriously difficult due to their inherent instability, as formalized by their positive Lyapunov exponents, which exponentially amplify errors in the learned dynamics. However, many of these systems exhibit ergodicity and an attractor: a compact and highly complex manifold, to which trajectories converge in finite-time, that supports an invariant measure, i.e., a probability distribution that is invariant under the action of the dynamics, which dictates the long-term statistical behavior of the system. In this work, we leverage this structure to propose a new framework that targets learning the invariant measure as well as the dynamics, in contrast with typical methods that only target the misfit between trajectories, which often leads to divergence as the trajectories' length increases. We use our framework to propose a tractable and sample efficient objective that can be used with any existing learning objectives. Our **Dy**namics **S**table **L**earning by **I**nvariant **M**easure (DySLIM) objective enables model training that achieves better point-wise tracking and long-term statistical accuracy relative to other learning objectives. By targeting the distribution with a scalable regularization term, we hope that this approach can be extended to more complex systems exhibiting slowly-variant distributions, such as weather and climate models. Code to reproduce our experiments is available here.

[1]Department of Computer Sciences, Cornell Tech, New York, NY, USA [2]Google Research, Mountain View, CA, USA [3]Department of Mathematics, University of Wisconsin-Madison, WI, USA. Correspondence to: Yair Schiff <yairschiff@cs.cornell.edu>, Leonardo Zepeda-Núñez <lzepedanunez@google.com>.

Proceedings of the 41^{st} International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024.

1. Introduction

Building data-driven surrogate models to emulate the dynamics of complex time-dependent systems is a cornerstone task in scientific machine learning (Farmer & Sidorowich, 1987), with applications ranging from fluid dynamics (Sanchez-Gonzalez et al., 2020), weather forecasting (Lam et al., 2022; Pathak et al., 2022; Bi et al., 2023), climate modeling (Kochkov et al., 2023), molecular dynamics (Jia et al., 2020; Merchant et al., 2023), quantum chemistry (Chen et al., 2020; Zepeda-Núñez et al., 2021), and plasma physics (Mathews et al., 2021; Anirudh et al., 2022).

Historically, various methods based on PCA (Pearson, 1901) and Koopman theory (Koopman, 1931) have been proposed to learn emulators by leveraging large datasets to build a surrogate model during a, typically expensive, offline phase (Schmid, 2010; Alexander & Giannakis, 2020; Kaiser et al., 2021; Schmid, 2022). The learned emulator provides fast and inexpensive inference, which is then used to accelerate downstream tasks such as design, control, optimization, data assimilation, and uncertainty quantification. Alas, many of these techniques are inherently linear, which renders them inadequate for problems with highly non-linear dynamics.

Indeed, many of the underlying physical processes driving target applications are described by non-linear and chaotic systems, which are characterized by strong instabilities, particularly with respect to initial conditions: trajectories with close initial conditions diverge quickly due to the positive Lyapunov exponents (Medio & Lines, 2001; Strogatz, 2018). Fortunately, many of these systems are dissipative, which implies the existence of a compact set, often called an **attractor**, towards which all bounded sets of initial conditions converge in finite-time (Temam, 2012). In addition, many of these systems also empirically exhibit **ergodicity**, whose main consequence translates to the existence of an attractor-induced **invariant measure**, a measure that is unchanged by the dynamics of the system, which captures the *long-term behavior* of the system (Stuart & Humphries, 1998).

Recent advances in machine learning (ML) have driven the development of several techniques for learning data-driven surrogates for non-linear dynamics (Rajendra & Brahmajirao, 2020; Roy & Rana, 2021; Brunton & Kutz, 2022; Ghadami & Epureanu, 2022; Nghiem et al., 2023). In the

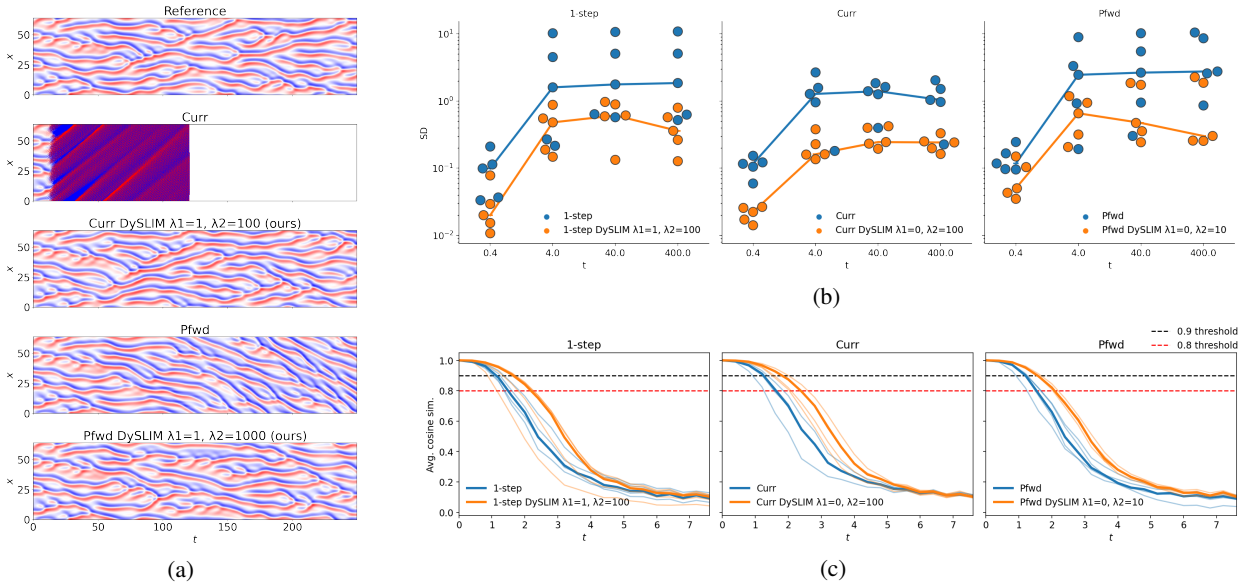


Figure 1. Improved stability with regularized DySLIM objectives in the Kuramoto–Sivashinsky (KS) and Lorenz 63 systems. (a) Sample ground truth and predicted trajectory across models trained on the KS system using Curriculum training (Curr) and the Pushforward trick (Pfwd) with/without regularization. The base versions showcase the blow-up issue (Curr) and wrong long-time dynamics (Pfwd). (b) Sinkhorn Divergence (SD; ↓) between trajectories at various rollout times of the Lorenz 63 system. Each point represents a random training seed, with the solid line indicating median values. (c) Cosine similarity (↑) over time for the Lorenz 63 system. Each line corresponds to one of five random training seeds with bolded lines indicating median values.

context of data-driven learning, autoregressive models (the focus of this paper) are a prevalent approach due to their ability to infer trajectories of arbitrary length. These autoregressive models predict a system's state at time $t + \Delta t$ based on its state at time t. Iterative application (unrolling) allows for trajectory forecasting far beyond training time horizons.

However, learning chaotic dynamics using autoregressive models in a stable manner has proven elusive. Due to memory and computational constraints, traditional ML-based approaches focus on learning short-term dynamics by minimizing the mean square error (MSE) between reference trajectories and those generated by unrolling a learned model; commonly using recurrent neural networks (Vlachas et al., 2018; Fan et al., 2020) or learning a projection from a stochastic trajectory using reservoir computing (Pathak et al., 2017; Bollt, 2021; Hara & Kokubu, 2022). Unfortunately, these models usually overfit to the short-term dynamics to the detriment of accurately predicting long-term behavior (Bonev et al., 2023). This manifests as trajectory blow-up, the values of the state variables diverging to infinity, or inaccurate long-term statistics during inference with large time-horizon, as depicted in Figure 1 (a). Recent works have focused on minimizing the misfit between increasingly longer trajectories (Keisler, 2022). Although these methods have been shown to attenuate the instability, the underlying difficulty remains: due to chaotic divergence, the losses become increasingly uninformative, which causes their gradients[1] to diverge, as shown by Mikhaeil et al. (2022).

A prime example of this challenge is weather and climate. While state-of-the-art ML models can learn short-term weather patterns (Lam et al., 2022), learning long-term climate behavior remains a very challenging open problem (Kochkov et al., 2023; Watt-Meyer et al., 2023). Thus the question we seek to answer becomes: *How do we incorporate knowledge of a system's long-term behavior into the learning stage, so that models remain point-wise accurate for short-term predictions and statistically accurate for long-term ones.*

This work addresses this challenge by leveraging the presence of attractors and their invariant measures. We propose a framework that directly targets the learning of long-term statistics by a measure-matching regularization loss.

Contributions The contributions of this paper are threefold: first, we propose a probabilistic and scalable framework for learning chaotic dynamics using data-driven, ML-based methods. Our framework introduces a system-agnostic[2] measure-matching regularization term into the

[1]Gradients are computed by backpropagation through the unrolled steps and are prone to exacerbate instabilities in the system. This is related to the well-known issue of exploding/vanishing gradients (Pascanu et al., 2013).

[2]We assume no knowledge of the systems, such as the expression of the equations driving the dynamics.

loss that induces stable and accurate trajectories satisfying the long-term statistics while enhancing short-term predictive power. Second, we use our framework to propose a tractable, sample and computationally-efficient[3] objective that we dub **Dy**namics **S**table **L**earning by **I**nvariant **M**easure (DySLIM) that can be used in conjunction with any existing dynamical system learning objective. Third, we demonstrate that DySLIM is capable of tackling larger and more complex systems than competing probabilistic methods, up to a state-dimension of 4,096 with complex 2D dynamics. Namely, we show competitive results in three increasingly complex and higher dimensional problems: the Lorenz 63 system (Lorenz, 1963; Tucker, 1999), a prototypical chaotic systems with the well known "butterfly" attractor, the Kuramoto-Sivashinsky (KS) equation, a 1D chaotic PDE, and the Kolmogorov-Flow (Obukhov, 1983), a 2D chaotic system that is routinely used as a benchmark for turbulent fluid dynamics (Kochkov et al., 2021). For the largest systems, we show that DySLIM performance remains stable even for large batch sizes and learning rates, regimes in which the performance of other methods deteriorates rapidly. These capabilities are potentially useful for accelerating the training stage by leveraging data parallelism and large learning rates.

2. Background

We consider autonomous systems of the form

$$\partial_t \boldsymbol{u} = \mathcal{F}[\boldsymbol{u}(t)], \tag{1}$$

where $\boldsymbol{u}(t)$ is the state of the system at time t.

For a given fixed time-step Δt, we discretize Equation 1 in space and time, and we define $\boldsymbol{u}_k = \boldsymbol{u}(k\Delta t) \in \mathbb{R}^D$ together with the discrete dynamical system

$$\boldsymbol{u}_{k+1} = \mathcal{S}_{\Delta t}(\boldsymbol{u}_k), \tag{2}$$

where $\mathcal{S}_{\Delta t}$ is the map obtained by integrating Equation 1 in time by a period Δt. In what follows, for brevity, we drop the subscript Δt. We can use the operator $\mathcal{S}$ to unroll, or advance in time, the solution of the dynamical system,

$$\boldsymbol{u}_k = \mathcal{S}(\boldsymbol{u}_{k-1}) = \mathcal{S} \circ \mathcal{S}(\boldsymbol{u}_{k-2}) = ... = \mathcal{S}^k(\boldsymbol{u}_0). \tag{3}$$

Chaotic Systems A chaotic system can be loosely defined as one whose trajectories are highly *sensitive to initial conditions.* Let $(\mathcal{U}, \mathrm{d})$ be an Euclidean metric space. We say that the system given by $\mathcal{S}$ is chaotic if there exists $\varepsilon > 0$ such that for all $\boldsymbol{u}_0 \in \mathcal{U}$ and $\delta > 0$, there exists $\boldsymbol{v}_0 \in B_\delta(\boldsymbol{u}_0)$ and $k \in \mathbb{N}$, such that:

$$\mathrm{d}(\mathcal{S}^k(\boldsymbol{u}_0), \mathcal{S}^k(\boldsymbol{v}_0)) \geq \varepsilon, \tag{4}$$

where $B_\delta(\boldsymbol{u}_0) = \{\boldsymbol{y} \mid \mathrm{d}(\boldsymbol{u}_0, \boldsymbol{y}) < \delta\}$ is a ball of radius δ centered at $\boldsymbol{u}_0$. Chaotic systems are also characterized by having a positive Lyapunov exponent: small discrepancies in the initial conditions are exaggerated exponentially over time (Strogatz, 2018).

Invariant Measures and Attractors We assume that the state space is measurable $(\mathcal{U}, \mathcal{A})$, where $\mathcal{A}$ is the Borel σ-algebra on $(\mathcal{U}, \mathrm{d})$, and we have a probability measure $\mu : \mathcal{A} \to [0, 1]$. If the discrete-time dynamical system map $\mathcal{S}$ is measurable, then it also defines a probability distribution $\mathcal{S}_\# \mu : \mathcal{A} \to [0, 1]$ which is called the pushforward of μ by $\mathcal{S}$, with $\mathcal{S}_\# \mu(A) = \mu(\mathcal{S}^{-1}(A))$, for all $A \in \mathcal{A}$. We say $\mathcal{S}$ preserves a measure μ, also denoted as μ is invariant[4] under/to $\mathcal{S}$, if:

$$\mu(\mathcal{S}^{-1}(A)) = \mu(A), \forall A \in \mathcal{A}, \text{ or equivalently } \mathcal{S}_\# \mu = \mu.$$

Intuitively, an attractor is a subset of the state space that characterizes the 'long-run' or 'typical' condition of the system. Formally, $A^* \subseteq \mathcal{U}$ is an attractor if it is a minimal set that satisfies the following properties: *i*) for all $\boldsymbol{a} \in A^*$ and $k \geq 0$, $\mathcal{S}^k(\boldsymbol{a}) \in A^*$ (i.e., A^* is invariant under $\mathcal{S}$) and *ii*) there exists $B \subseteq \mathcal{U}$, known as the basin of attraction, such that for all $\boldsymbol{b} \in B$ and $\varepsilon > 0$ there exists some $k^\star > 0$ such that $\mathcal{S}^k(\boldsymbol{b})$ is in an ε-neighborhood of A^*, for all $k \geq k^\star$. If the basin of attraction consists of the entire state space, then A^* is said to be a global attractor (Stuart, 1994). As an example, Figure 4 depicts the Lorenz 63 attractor.

3. Learning Dynamical Systems

Our goal is to find a Markovian parametric model $\mathcal{S}_\theta$ that governs our system in a manner consistent with the true dynamics defined by $\mathcal{S}$. To do so, we leverage previously collected data, which consists of n trajectories: $\mathcal{D} = \{(\boldsymbol{u}_j^{(i)})_{j=0}^{\ell^{(i)}}\}_{i=1}^n$, where $\ell^{(i)}$ is the length of the i-th trajectory, whose initial conditions are sampled from an invariant measure supported on the attractor, i.e., $\{\boldsymbol{u}_0^{(i)}\}_{i=1}^n \overset{\text{iid}}{\sim} \mu_0 = \mu^*$. Letting μ_j be the distribution over states $\boldsymbol{u}_j$, i.e., states after j time-steps, for ergodic dissipative systems, we have that $\mu_j := \mathcal{S}_\sharp^j \mu_0 = \mathcal{S}_\sharp^j \mu^* = \mu^*$.

$\mathcal{S}_\theta$ is trained by minimizing an empirical estimate of the mismatch between predicted and observed trajectories. Most of these estimates are based on MSE, e.g., the *one-step objective*:

$$\mathcal{L}^{\text{1-step}}(\theta) = \mathbb{E}_j \mathbb{E}_{\boldsymbol{u}_j \sim \mu_j} \left[\|\mathcal{S}_\theta(\boldsymbol{u}_j) - \mathcal{S}(\boldsymbol{u}_j)\|^2 \right], \tag{5}$$

for a norm $||\cdot||$ induced by d in Equation 4, and where the outer expectation $\mathbb{E}_j$ represents averages along trajectories.

[3] Our regularization loss incurs an extra cost depending only quadratically on the batch size.

[4] We note that invariant measures may not be unique. For example, transformations with high degree of symmetries, such as a rigid-body transformation (e.g., translation and rotation), can have an infinite number of invariant measures.

At inference, learned models generate trajectories by autoregressively unrolling predictions, as in Equation 3: starting from a given $\boldsymbol{u}_0$, we generate $\tilde{\boldsymbol{u}}_k = \mathcal{S}_\theta^k(\boldsymbol{u}_0)$. As we unroll for large k, the learning dynamics can become unstable, by either diverging or converging to a different attractor.

Multi-step Objectives To attenuate this issue, two popular objectives have been introduced recently, which have been used to train state-of-the-art models (Brandstetter et al., 2022; Lam et al., 2022; Kochkov et al., 2023). Specifically, we examine a generalization of $\mathcal{L}^{\text{1-step}}$, the ℓ-step objective:

$$\mathcal{L}^{\ell\text{-step}}(\theta) = \mathbb{E}_j \mathbb{E}_{\boldsymbol{u}_j \sim \mu_j} \sum_{k=1}^{\ell} \omega(k) \left\| \mathcal{S}_\theta^k(\boldsymbol{u}_j) - \mathcal{S}^k(\boldsymbol{u}_j) \right\|^2, \tag{6}$$

where $\omega(k)$ is a *discount factor* used to stabilize training[5]. Training paradigms where ℓ starts at one and is gradually increased are known as *curriculum training* (Curr; Krishnapriyan et al. (2021); Keisler (2022)), and we denote them as $\mathcal{L}^{\text{Curr}}$.

Alas, $\mathcal{L}^{\ell\text{-step}}$ introduces several difficulties. By the chain rule, computing the gradient of Equation 6 requires the storage of k intermediate evaluations for each term in the inner sum in order to calculate the Jacobian $\nabla_\theta \mathcal{S}_\theta^k(\boldsymbol{u}_0)$, which can be prohibitive unless gradient checkpointing is used (Chen et al., 2016). Crucially, for chaotic systems, Mikhaeil et al. (2022) proved that these gradients necessarily 'explode' as the length of the trajectory grows.

To reduce computational cost and further induce stability, one can use the *pushforward trick* (Pfwd), introduced in Brandstetter et al. (2022). The pushforward trick replaces inputs $\boldsymbol{u}_j$ to the parametric model with noised states $\tilde{\boldsymbol{u}}_j$ drawn from an adversarial distribution induced by the model, e.g., $\tilde{\boldsymbol{u}}_j = \mathtt{sg}(\mathcal{S}_\theta(\boldsymbol{u}_{j-1}))$, where $\mathtt{sg}(\cdot)$ represents the stop-gradient operation. The noise can be also generated by the repetitive application of the to-be-learned model[6] $\mathcal{S}_\theta$, e.g. $\tilde{\boldsymbol{u}}_{j+k} = \mathtt{sg}(\mathcal{S}_\theta^k(\boldsymbol{u}_j))$. In such cases, the pushforward objective can be written in general form as:

$$\mathcal{L}^{\text{Pfwd},\ell}(\theta) = \tag{7}$$
$$\mathbb{E}_j \mathbb{E}_{\boldsymbol{u}_j \sim \mu_j} [\omega(\ell) || \mathcal{S}_\theta(\mathtt{sg}(\mathcal{S}_\theta^{\ell-1}(\boldsymbol{u}_j))) - \mathcal{S}^\ell(\boldsymbol{u}_j) ||^2].$$

This objective can either be used to replace or in addition to those defined in Equations 5 and 6.

Sources of Instability We recast the instability of learned dynamical models as short-term *overfitting* and long-term *distribution shift*: parameters θ that minimize $\mathcal{L}^{\text{1-step}}$ on training data often overfit to this data and lead to $\mathcal{S}_{\theta\#}\mu_j \neq \mu_{j+1}$. When deployed, the learned dynamical model will accumulate errors along a predicted trajectory as the distribution of predicted states veers further away from that of the actual system. Recent techniques (including the Curr and Pfwd training) attempt to mitigate this issue by encouraging the model to recover from deviations caused by pushing forward by $\mathcal{S}_\theta$. However, these objectives are still prone to instabilities. For example, Figure 1 (a) depicts issues for the chaotic KS equation. The model trained with Curr training fails to generalize beyond the first ℓ steps on which it was trained: the trajectory quickly enters an unstable attractor, from which it blows up. Similarly, the model trained using the Pfwd training is able to learn the short-term dynamics, however, as time increases, the trajectories enter a different attractor, one in which the dynamics are biased towards the right. In both cases, by introducing our proposed regularization, we are able to correct the long-term behavior.

4. Main Idea and Methods

To tackle the issue of distribution shift, we propose to focus on systems' invariant measure preservation. Specifically, many systems of interest have some measure μ^* supported on an attractor that is invariant to the transformation $\mathcal{S}$ (Tucker, 2002; Weinan & Liu, 2002; Luzzatto et al., 2005; Ferrario, 2008; Hawkins, 2021). We cast our learning problem as finding parameters θ such that a surrogate $\mathcal{S}_\theta$ preserves μ^* while approximating $\mathcal{S}$ locally, which defines the following constrained optimization:

$$\min_\theta \mathcal{L}(\theta) \quad \text{s.t. } \mu_\theta^* = \mu^*, \tag{8}$$

where $\mathcal{L}(\theta)$ is the short-term loss, and μ_θ^* is the invariant measure of $\mathcal{S}_\theta$, i.e., $(\mathcal{S}_\theta)_\sharp \mu_\theta^* = \mu_\theta^*$, which we assume exists[7]. Solving this constrained optimization inherently alleviates the distribution shift problem. Since trivial solutions exist for measure preservation, e.g., if $\mathcal{S}_\theta$ is the identity, the trajectory matching component $\mathcal{L}(\theta)$ of this constrained objective is necessary for producing useful surrogates.

The intractability of the constrained problem in Equation 8 leads us to consider a relaxed version by turning the problem into a regularized objective of the form:

$$\mathcal{L}_\lambda^{\text{D}}(\theta) = \mathcal{L}(\theta) + \lambda \text{D}(\mu^*, \mu_\theta^*), \tag{9}$$

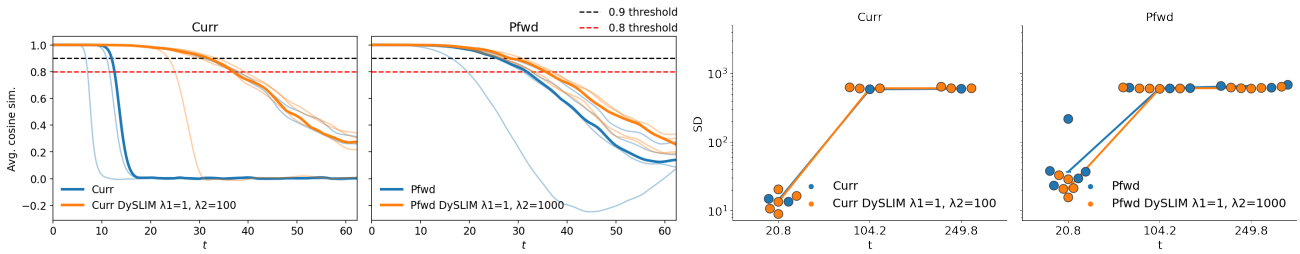
where the hyperparameter λ controls the strength of regularization, and D is a measure distance / divergence.

This formulation raises three additional questions: *i*) which metric to use for measuring distance between distributions,

[5] Since matching further rolled-out steps increases in difficulty with k, especially for chaotic systems, we consider a monotonically decreasing discount factor of the form $\omega(k) = r^{k-1}$, $0 < r < 1$, inspired by Kochkov et al. (2023).

[6] The pushforward trick can be re-framed in our measure-matching framework although using a discrete Wasserstein 2 metric. See Appendix A for more details.

[7] This is a key hypothesis in our methodology. Otherwise, this assumption can be enforced in $\mathcal{S}_\theta$ by adding a potential term, as done in Li et al. (2022).



(a) (b)

Figure 2. Regularized DySLIM objectives outperform baselines for the KS system. (a) Cosine similarity (↑) over time. Each line corresponds to the mean over trajectories of each of five random training seeds, with bold lines indicating median values. (b) Sinkhorn Divergence (SD; ↓) between trajectories at various rollout times. Each point represents a random training seed that remains stable, with the solid line indicating median values.

ii) how to sample from μ_θ^*, which is unknown, and *iii*) how to estimate the regularization with a finite (and potentially small) number of samples, which is crucial for solving Equation 9 using stochastic optimization pipelines.

Measure Distance Our choice of measure distance needs to satisfy several desiderata, namely it should: *i*) respect the underlying geometry of $\mathcal{U}$ and support comparison between measures with non-overlapping supports, *ii*) admit an unbiased, sampled-based estimator, *iii*) have low computational complexity with respect to the system dimension and number of samples, *iv)* entail convergence properties on the space of measures defined on $\mathcal{U}$ (informally, $\mathrm{D}(\mu_\theta^*, \mu^*) \to 0 \implies \mu_\theta^* \to \mu^*$), and *v)* enjoy parametric rates of estimation (i.e., sampling error $|\mathrm{D} - \widehat{\mathrm{D}}|$ is independent of system dimension).

Some popular notions of distance / divergence from statistical learning theory include the Kullback-Leibler and Hellinger. However, these do not take into account the distance metric of the space on which the distributions are defined (Genevay et al., 2018; Feydy et al., 2019) and in some cases are undefined for non-overlapping supports.

In contrast, Integral Probability Metrics (IPMs; Müller (1997)) represent a general purpose tool for comparing two distributions. Among the class of IPMs, the Maximum Mean Discrepancy (MMD; Gretton et al. (2012)) stands out as it has a closed form expression and satisfies all our requirements described above. Deferring several details about the MMD to Appendix B, we define it here as,

$$\mathrm{MMD}^2(\mu^*, \mu_\theta^*) = \mathbb{E}_{\boldsymbol{u},\boldsymbol{u}'\sim\mu^*}[\kappa(\boldsymbol{u},\boldsymbol{u}')] + \mathbb{E}_{\boldsymbol{v},\boldsymbol{v}'\sim\mu_\theta^*}[\kappa(\boldsymbol{v},\boldsymbol{v}')] - 2\mathbb{E}_{\boldsymbol{u}\sim\mu^*,\boldsymbol{v}\sim\mu_\theta^*}[\kappa(\boldsymbol{u},\boldsymbol{v})], \tag{10}$$

where $\kappa : \mathcal{U} \times \mathcal{U} \to \mathbb{R}$ is a kernel.[8] For two sets of n samples $\{\boldsymbol{u}^{(i)}\}_{i=1}^n \sim \mu^*$ and $\{\boldsymbol{v}^{(i)}\}_{i=1}^n \sim \mu_\theta^*$, Equation 10 admits an unbiased estimator (Gretton et al., 2012):

$$\widehat{\mathrm{MMD}}^2(\mu^*, \mu_\theta^*) = \frac{1}{n(n-1)}\left[\sum_{i=1}^n\sum_{j\neq i}^n[\kappa(\boldsymbol{u}^{(i)},\boldsymbol{u}^{(j)})] + \sum_{i=1}^n\sum_{j\neq i}^n[\kappa(\boldsymbol{v}^{(i)},\boldsymbol{v}^{(j)})]\right] - \frac{2}{n^2}\sum_{i=1}^n\sum_{j=1}^n[\kappa(\boldsymbol{u}^{(i)},\boldsymbol{v}^{(j)})],$$

This estimator can be easily computed in $\mathcal{O}(n^2)$ operations. Additionally, the MMD entails convergence properties on the space of measures defined on $\mathcal{U}$. That is, it metrizes weak convergence (Simon-Gabriel et al., 2023), and, for characteristic kernels, we have $\mathrm{MMD}(\mu^*, \mu_\theta^*) = 0 \iff \mu^* = \mu_\theta^*$ (Sriperumbudur et al., 2010). The MMD also enjoys parametric rates of estimation, with $\mathcal{O}(1/\sqrt{n})$ sampling error (Gretton et al., 2006; Tolstikhin et al., 2016). We point out that, in practice, n is the batch size, since we employ stochastic optimization methods.

Approximate Sampling from the Invariant Measure by Time-stepping Even though the metric above satisfies several desirable properties, we do not have access to samples of μ_θ^*. Fortunately, if we assume that $\mathcal{S}_\theta$ has an attractor, then $\mathcal{S}_\theta^k(\boldsymbol{u}_0)$ will become a sample of μ_θ^* for sufficiently large k and $\boldsymbol{u}_0$ in the basin of attraction. Analogous to Equation 2, we have that applying $\mathcal{S}_\theta$ to $\boldsymbol{u}_0$ is equivalent to stepping forward in time according to the learned dynamics, i.e., sampling from μ_θ^* is equivalent to unrolling the trajectory in time. Then we approximate the invariant measure, μ_θ^*, associated with $\mathcal{S}_\theta$ by time-unrolling samples of μ^*, i.e., $\left(\mathcal{S}_\theta^k\right)_\sharp \mu^* \approx \mu_\theta^*$ for a large k. This observation allows us to approximate the regularization term in Equation 9 by

$$\mathrm{D}(\mu^*, \mu_\theta^*) \approx \mathrm{D}(\mu^*, \left(\mathcal{S}_\theta^k\right)_\sharp \mu^*). \tag{11}$$

Conditional and Unconditional Regularization Using the approximation in Equation 11 in the context of a stochastic optimization pipeline requires that we estimate this term

[8] The choice of kernel has important practical implications (Liu et al., 2020; Schrab et al., 2023), and many kernels κ_σ are controlled by a bandwidth hyperparameter σ that should be tuned.

with a potentially small batch size. In fact, besides some simple systems, one typically can only afford small batch sizes, which means we may not be fully capturing both μ^* and μ^*_θ. Although our choice of regularization loss comes with an unbiased estimator with parametric rates of error, in small regimes, estimation error can still diminish its effectiveness at providing a meaningful signal.

We therefore manipulate the expression in Equation 9 to obtain a different yet equivalent loss. Using the fact that $\mu^* = \mathcal{S}_\sharp \mu^* = (\mathcal{S}^k)_\sharp \mu^*$, we obtain the equivalent expressions $\mathrm{D}(\mu^*, \left(\mathcal{S}^k_\theta\right)_\sharp \mu^*)$ and $\mathrm{D}((\mathcal{S}^k)_\sharp \mu^*, (\mathcal{S}^k_\theta)_\sharp \mu^*)$, which we respectively dub as the **unconditional** and **conditional regularization**. We can easily manipulate the latter expression using Equation 10 to yield

$$\mathrm{MMD}^2((\mathcal{S}^k)_\sharp \mu^*, (\mathcal{S}^k_\theta)_\sharp \mu^*) = \mathbb{E}_{\boldsymbol{u},\boldsymbol{v}\sim\mu^*}[\kappa(\mathcal{S}^k(\boldsymbol{u}), \mathcal{S}^k(\boldsymbol{v})) + \kappa(\mathcal{S}^k_\theta(\boldsymbol{u}), \mathcal{S}^k_\theta(\boldsymbol{v}')) - 2\kappa(\mathcal{S}^k(\boldsymbol{u}), \mathcal{S}^k_\theta(\boldsymbol{v})], \tag{12}$$

which can be estimated by extracting a subset of initial conditions, time-evolving them k steps, and then computing the estimator on the time-evolved samples. We point out that this is equivalent to conditioning the loss on the initial conditions, hence the name conditional regularization. When using samples to estimate MMD, we use a collection of initial conditions $\{\boldsymbol{u}_0^{(i)}\}_{i=1}^n$ for unconditional regularization and a collection of samples time-evolved by the true system, i.e., ones that come from later time steps in training trajectories, $\{\boldsymbol{u}_k^{(i)} = \mathcal{S}^k(\boldsymbol{u}_0^{(i)}))\}_{i=1}^n$ for conditional regularization.

Although the expressions for both regularizations are equivalent, they lead to different finite-sample estimators. The former compares initial samples with ones evolved using $\mathcal{S}_\theta$, while the latter compares samples evolved using both $\mathcal{S}$ and $\mathcal{S}_\theta$. In our experiments, we incorporate both terms and explore different weighting schemes. Empirically, we find that the unconditional regularization is useful when the dynamical system's state dimension is small, and one can afford a large batch; but it becomes uninformative as the dimension of the dynamical system state increases, due to larger distances between samples and sparse coverage of the attractor, which also becomes higher dimensional.

DySLIM Combining the elements above leads us to our proposed objective, DySLIM:

$$\widehat{\mathcal{L}}^{\mathrm{D}}_\lambda(\theta) = \widehat{\mathcal{L}}^{\mathrm{obj}}(\theta) + \lambda_1 \widehat{\mathrm{D}}(\mu^*, (\mathcal{S}^\ell_\theta)_\sharp \mu^*) + \lambda_2 \widehat{\mathrm{D}}((\mathcal{S}^\ell)_\sharp \mu^*, (\mathcal{S}^\ell_\theta)_\sharp \mu^*), \tag{13}$$

where ℓ depends on the type of baseline objective $\widehat{\mathcal{L}}^{\mathrm{obj}}(\theta)$ used. The second and third terms in Equation 13 correspond to the unconditional and conditional regularization, respectively. We perform a hyperparameter search over λ_1 and λ_2, taking $\lambda_1 \in \{0, 1\}$ and $\lambda_2 \in \{1, 10, 100, 1000\}$. Importantly, this objective can be used in conjunction with any of the base losses introduced above.

For the measure distance $\widehat{\mathrm{D}}$ in Equation 13, we use $\widehat{\mathrm{MMD}}^2$ and define $\kappa_{\boldsymbol{\sigma}}$ as a mixture of rational quadratic kernels (Rasmussen et al., 2006; Li et al., 2015):

$$\kappa_{\boldsymbol{\sigma}}(\boldsymbol{u}, \boldsymbol{v}) = \sum_{\sigma_q \in \boldsymbol{\sigma}} \kappa_{\sigma_q}(\boldsymbol{u}, \boldsymbol{v}) = \sum_{\sigma_q \in \boldsymbol{\sigma}} \frac{\sigma_q^2}{\sigma_q^2 + ||\boldsymbol{u} - \boldsymbol{v}||_2^2},$$

where we select the set $\boldsymbol{\sigma}$ depending on the dynamical system, see Appendix E for details.

5. Experiments

Baselines Our baseline models are trained with $\widehat{\mathcal{L}}^{\mathrm{obj}}$, where obj $\in \{$1-step, Curr, Pfwd$\}$ and where $||\cdot||$ is the L^2 norm. For each system and objective, the same model architecture, learning rate, and optimizer hyperparameters are used. All experiments are repeated with five different random seeds.

Evaluation We evaluate models both for their 'short-term' predictive ability and 'long-run' stability. The former is measured by a cosine similarity statistic between true and predicted trajectories. The latter is measured by system-specific metrics (see Wan et al. (2023a)) capturing the distributional similarity between true and generated trajectories, along with their visual inspection. In particular, we use the Sinkhorn Divergence (**SD**) (Genevay et al., 2018) between the empirical distributions of ground truth and predicted trajectories at various time steps to quantify distributional overlap. We also use the mean energy log ratio (**MELR**), which measures the average deviation of the energy at each Fourier mode of the generated snapshots when compared to the ground truth. We also consider its weighted variant (**MELRw**), which up-weights modes with higher energy, in particular the low-frequency modes. We also use the mean of the Frobenius norm of covariance matrix (**covRMSE**) that measures the spatial statistical properties of the generated samples. We additionally compute point-wise Wasserstein metrics. Finally, we consider a time correlation metric (**TCM**) which provides a measure of temporal behavior, in contrast to most of the metrics above, which are snapshot-based. For more detail on and precise definitions of the evaluation criteria, see Appendix C.

5.1. Lorenz 63

The first system we examine is the Lorenz 63 model (Lorenz, 1963), which is a simplified model of atmospheric convection and is defined by a non-linear ordinary differential equation. Our models use a simple MLP network, due to the low-dimensional nature of the problem. For more details about this differential equation and the experimental setup for this system, see Appendix E.1.

Figure 1 (b) and (c) demonstrate the improved stability from adding invariant measure regularization to the different training objectives considered in this paper. In particular, Figure 1 (b) demonstrates improved long-term statistics, as the distribution of points for models trained with DySLIM are closer to that of ground truth trajectories compared to those from models trained with unregularized objectives. In addition, Figure 1 (c) shows that with DySLIM we obtain the added benefits of improving the short-term model prediction, with longer de-correlation times. For further results with different metrics, see Appendix F.1.

5.2. Kuramoto-Sivashinsky

We next experiment in the more difficult setting of the high-order PDE known as the Kuramoto-Sivashinsky (KS) equation, which is discussed, along with the experimental setup, in Appendix E.2. For this experiment, the 1-step objective proved to be too unstable, even when regularization was applied, so we focus only on Curr and Pfwd objectives.

In Figure 2, we observe better short-term predictions and improved long-term stability, as measured by lower SD between ground truth and predicted trajectory distributions. In Figure 1 (a), we see example trajectories that highlight the difference between models trained with and without regularized objectives. For the Curr objective, models often diverge and produce numerical instability, while for the Pfwd objective, models deviate from the attractor. In contrast, the regularized versions of these objectives yield more stable models that remain on the correct attractor manifold.

5.3. Kolmogorov Flow

Finally, we study chaotic 2D fluid flow defined by the Navier-Stokes equations with Kolmogorov forcing. Information about the PDE and experiment setups is available in Appendix E.3. For this system, the SD becomes non-discriminative due to the high-dimension of the state space, so we rely on the other metrics outlined above.

Figure 3 (left) shows typical behavior of the unrolled trained models for a given initial condition: the baselines become highly dissipative and quickly veer towards the mean, whereas DySLIM greatly improves long-term behavior. Figure 3 (right) shows a similar results to those in the other experiments: the short-term behavior of the solution is enhanced by the regularization (see Figure 8 for further comparisons). We find that curriculum training is often worse due to more stringent memory requirements that prevent us from unrolling for longer time-horizons during training.

As an ablation, we sweep over different batch sizes and learning rates (see Appendix E.3 for the specific details.) The results are summarized in Table 1, which shows that models trained with DySLIM either have an edge or remain competitive across the spectrum of different learning rates and batch sizes considered. However, as batch size and learning rate increase, the behavior of the model trained with DySLM remains consistent, whereas the models trained only using the original objective deteriorate quickly.

6. Related Work

ROM Methods Classical reduced order model (ROM) methods build surrogates by identifying low-dimensional linear approximation spaces tailored to representing target system states. Such spaces are usually derived from data samples (Aubry et al., 1988; Barrault et al., 2004; Chinesta et al., 2011; Amsallem et al., 2012), and ROMs are obtained by projecting the system equations onto the approximation space (Galerkin, 1915). Although these methods inherently leverage the linear behavior of underlying dynamics, they have been recently extended to handle mildly non-linear dynamics (Willcox, 2006; Astrid et al., 2008; Chaturantabut & Sorensen, 2010; Ayed et al., 2019; Geelen et al., 2022). However, their performance deteriorates rapidly for highly non-linear advection-dominated systems, such as KS and Kolmogorov flow (Peherstorfer, 2022)

Hybrid Physics-ML More recent methods hybridize classical numerical methods with contemporary data-driven deep learning techniques (Mishra, 2018; Bar-Sinai et al., 2019; Bruno et al., 2021; Kochkov et al., 2021; List et al., 2022; Frezat et al., 2022; Dresdner et al., 2022; Boral et al., 2023). These approaches *learn* corrections to numerical schemes from high-resolution simulation data, resulting in fast, low-resolution methods with high-accuracy. However, they require knowledge of the underlying PDE.

Pure ML-surrogates and Stabilization Techniques Recent works have focused on short-term training trajectories using recurrent networks (Vlachas et al., 2018) and reservoir computing techniques (Vlachas et al., 2020; Platt et al., 2021). Other approaches seek to regularize the training stage by leveraging properties of the systems. Such stabilization can be achieved by incorporating noise (Sanchez-Gonzalez et al., 2020), which can be induced by the learned model (Brandstetter et al., 2022); by back-propagating the gradient along many time steps (Um et al., 2020), and learning the dynamics on a latent space (Stachenfeld et al., 2022; Serrano et al., 2023), while promoting smoothness in the latent space (Wan et al., 2023b). Or by using a generative teacher network (Lamb et al., 2016), or leveraging an approximate inertial form (Lu et al., 2017). Related to the Curriculum training baseline, Hess et al. (2023) use states that interpolate between model predicted and ground truth states to mitigate gradient explosion. Finally, a somewhat related stabilization method, introduced in Wang et al. (2014); Blonigan et al. (2018), develops a shadowing technique for sensitivity analysis of long-term averaged gradients.

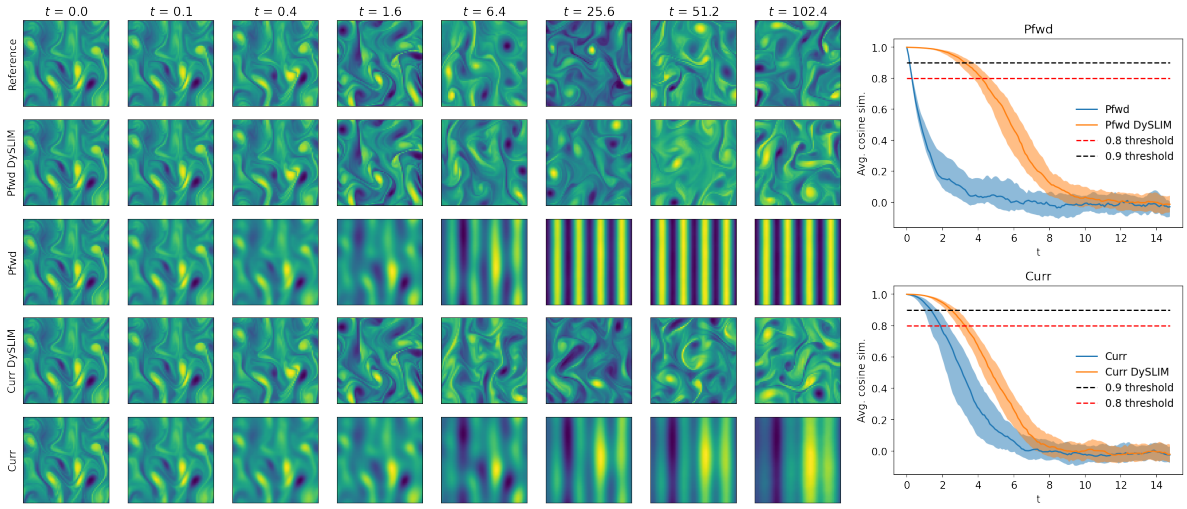


Figure 3. (*Left*) Sample reference and predicted trajectory across models trained on the Kolmogorov Flow data using the Curr and Pfwd objectives, together with the regularized versions. (*Right*) Evolution of the cosine similarity over time for Curr and Pfwd objectives with and without regularization. The solid line is the median among 160 runs (32 trajectories for each of the 5 random seeds), and the shaded regions correspond to the second and third quartile. ($\lambda_1 = 0$, $\lambda_2 = 100$, batch size = 128 and learning rate = 5e^{-4}).

Table 1. Kolmogorov flow: Metrics for 1-step, curriculum, and pushforward objectives without and with regularization ($\lambda_1 = 0$, $\lambda_2 = 100$). Boldface numbers indicate that the metric is improved by our regularization. All values displayed are in units of $\times 10^{-2}$.

	Batch size	LR	MELR (↓)		MELRw (↓)		covRMSE (↓)		Wass1 (↓)		TCM (↓)	
			Base	DySLIM	Base	DySLIM	Base	DySLIM	Base	DySLIM	Base	DySLIM
1-step	64	5e-4	2.77	**1.84**	0.44	0.85	7.93	**7.30**	16.2	**5.55**	5.39	**2.45**
Curr	64	5e-4	5.35	**1.64**	0.95	**0.45**	8.13	**6.95**	9.66	**4.76**	3.50	**2.83**
Pfwd	128	1e-4	3.19	**2.46**	0.53	0.53	6.81	**6.69**	4.64	**4.51**	3.68	**0.72**

Table 2. Complexities for each objective, with d denoting state dimension, $|\theta|$ number of parameters, which is implementation dependent, NN complexity of the neural network for one application, n_b batch size, and n_t number of maximum rollout steps.

Objective	Cost $\mathcal{O}(\cdot)$	Memory footprint $\mathcal{O}(\cdot)$
1-step	$n_b d + n_b NN$	$n_b d + n_b \lvert\theta\rvert$
+ DySLIM	$n_b^2 d + n_b NN$	$n_b^2 d + n_b \lvert\theta\rvert$
Curr	$n_t n_b d + n_t n_b NN$	$n_t n_b d + n_t n_b \lvert\theta\rvert$
+ DySLIM	$n_t n_b^2 d + n_b NN$	$(n_b^2 + n_b) n_t d + n_t n_b \lvert\theta\rvert$
Pfwd	$n_b d + n_b NN$	$n_b d + n_b \lvert\theta\rvert$
+ DySLIM	$n_b^2 d + n_b NN$	$n_b^2 d + n_b \lvert\theta\rvert$

Operator Learning Neural operators seek to learn the integro-differential operators directly, without explicit PDE-informed components.These methods often leverage classical fast-methods (Fan et al., 2019; Li et al., 2020; 2021; Tran et al., 2021), or approximation-theoretic structures (Lu et al., 2021) to achieve computational efficiency. Some of these techniques have been extended to handle dissipative systems (Li et al., 2022) by hard-coding a dissipative term at both training and inference time.

Learning Invariant Measures Botvinick-Greenhouse et al. (2023) use an Eulerian approach to learn dynamics on invariant measures for low-dimension ODEs using the Feynman-Kac formula coupled with PDE-constrained optimization. In recent work, Jiang et al. (2023) use neural operators and optimal transport to match the distribution of system-specific summary statistics, which are built using knowledge of the underlying equation driving the system dynamics. However, the approach was only applied to small systems with low-dimensional attractors, and it is not clear how well it scales to high-dimensional problems, such as the Kolmogorov flow. Similarly, Platt et al. (2023) regularize using other invariants, such as the Lyapunov spectrum.

Our proposed method sits between stabilization techniques and learning invariant measures. In particular, we stabilize the training by implicitly learning the invariant measure of the system along the short-term dynamics. While our work elects to use the MMD, we note that using other Optimal Transport-based metrics for measure matching, such as the Sinkhorn Divergence, as in (Jiang et al., 2023), is a reasonable choice when the system state is small, and batch

Table 3. Median execution times (rounded to the hour) for 5 training runs (720k steps) on Kolmogorov flow using the PushFwd (max rollout of 10) and Curr (max rollout of 5).

Batch size	Pfwd		Curr	
	Base	DYSLIM	Base	DYSLIM
32	40	40	48	48
64	74	76	161	163
128	145	146	OOM	OOM

size is large, as shown in Appendix G. However, as system size increases and batch sizes decrease (due to memory constraints), we find that the models trained using SD in the regularization perform worse compared to those trained using MMD.

MMD in Generative Modeling MMD-based regularization has been used in the context of generative modeling, e.g., the MMD has been used to distinguish between samples of the generated and true distributions (Li et al., 2015; Dziugaite et al., 2015; Li et al., 2017; Bińkowski et al., 2018) within the framework of generative adversarial network (Goodfellow et al., 2020). Additionally, drawing on the close connection between the MMD and a related proper scoring rule (Gneiting & Raftery, 2007; Ramdas et al., 2017), Si et al. (2021; 2023) use the energy distance (Baringhaus & Franz, 2004; Székely et al., 2004), a special case of the MMD (Sejdinovic et al., 2013), to train normalizing flow generative models (Rezende & Mohamed, 2015; Papamakarios et al., 2021).

Complexity Our methodology incurs relatively small overhead compared to the baselines. Table 2 shows that our methodology adds an extra cost depending only quadratically on the batch size and linearly on state dimension. We see this empirically in Table 3: wall-clock times are roughly equal with and without our regularization.

7. Conclusion

In this work, we have presented a tractable, scalable, and system-agnostic regularized training objective, DySLIM, that leverages a key property of many dynamical systems of interest in order to produce more stable learned models. Specifically, by pushing learned system models to preserve the invariant measure of an underlying dynamical system, we demonstrated that both short-term predictive capabilities and long-term stability can be improved across a range of well-studied systems, e.g., Lorenz 63, KS, and Kolmogorov Flows. We hope that the principles of invariant measure preservation introduced in our work, coupled with a tractable and scalable formulation, can serve to stabilize real-world dynamical system models with slowly varying measures, such as those used in global weather prediction.

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none of which we feel must be specifically highlighted here.

References

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., and Zheng, X. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. URL `https://www.tensorflow.org/`. Software available from tensorflow.org.

Alexander, R. and Giannakis, D. Operator-theoretic framework for forecasting nonlinear time series with kernel analog techniques. *Physica D: Nonlinear Phenomena*, 409:132520, 2020.

Amsallem, D., Zahr, M. J., and Farhat, C. Nonlinear model order reduction based on local reduced-order bases. *International Journal for Numerical Methods in Engineering*, 92(10):891–916, 2012.

Anirudh, R., Archibald, R., Asif, M. S., Becker, M. M., Benkadda, S., Bremer, P.-T., Budé, R. H., Chang, C.-S., Chen, L., Churchill, R., et al. 2022 review of data-driven plasma science. *arXiv preprint arXiv:2205.15832*, 2022.

Astrid, P., Weiland, S., Willcox, K., and Backx, T. Missing point estimation in models described by proper orthogonal decomposition. *IEEE Transactions on Automatic Control*, 53(10):2237–2251, 2008. doi: 10.1109/TAC.2008.2006102.

Aubry, N., Holmes, P., Lumley, J. L., and Stone, E. The dynamics of coherent structures in the wall region of a turbulent boundary layer. *Journal of Fluid Mechanics*, 192:115–173, 1988. doi: 10.1017/S0022112088001818.

Ayed, I., de Bezenac, E., Pajot, A., Brajard, J., and Gallinari, P. Learning dynamical systems from partial observations, 2019.

Bar-Sinai, Y., Hoyer, S., Hickey, J., and Brenner, M. P. Learning data-driven discretizations for partial differential equations. *Proceedings of the National Academy of Sciences*, 116(31):15344–15349, July 2019. ISSN 0027-8424, 1091-6490. doi: 10.1073/pnas.1814058116.

URL https://pnas.org/doi/full/10.1073/pnas.1814058116.

Baringhaus, L. and Franz, C. On a new multivariate two-sample test. *Journal of multivariate analysis*, 88(1):190–206, 2004.

Barrault, M., Maday, Y., Nguyen, N. C., and Patera, A. T. An 'empirical interpolation' method: application to efficient reduced-basis discretization of partial differential equations. *Comptes Rendus Mathematique*, 339(9):667–672, 2004. ISSN 1631-073X. doi: https://doi.org/10.1016/j.crma.2004.08.006. URL https://www.sciencedirect.com/science/article/pii/S1631073X04004248.

Bi, K., Xie, L., Zhang, H., Chen, X., Gu, X., and Tian, Q. Accurate medium-range global weather forecasting with 3d neural networks. *Nature*, 619(7970):533–538, 2023.

Bińkowski, M., Sutherland, D. J., Arbel, M., and Gretton, A. Demystifying MMD GANs. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=r1lUOzWCW.

Blonigan, P. J., Wang, Q., Nielsen, E. J., and Diskin, B. Least-squares shadowing sensitivity analysis of chaotic flow around a two-dimensional airfoil. *AIAA Journal*, 56 (2):658–672, 2018.

Bollt, E. On explaining the surprising success of reservoir computing forecaster of chaos? the universal machine learning dynamical system with contrast to var and dmd. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 31(1), 2021.

Bonev, B., Kurth, T., Hundt, C., Pathak, J., Baust, M., Kashinath, K., and Anandkumar, A. Spherical Fourier neural operators: Learning stable dynamics on the sphere. *arXiv preprint arXiv:2306.03838*, 2023.

Boral, A., Wan, Z. Y., Zepeda-Núñez, L., Lottes, J., Wang, Q., Chen, Y.-F., Anderson, J. R., and Sha, F. Neural ideal large eddy simulation: Modeling turbulence with neural stochastic differential equations. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=x6cOcxRnxG.

Botvinick-Greenhouse, J., Martin, R., and Yang, Y. Learning dynamics on invariant measures using PDE-constrained optimization. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 33(6), 2023.

Bradbury, J., Frostig, R., Hawkins, P., Johnson, M. J., Leary, C., Maclaurin, D., Necula, G., Paszke, A., VanderPlas, J., Wanderman-Milne, S., and Zhang, Q. JAX: composable transformations of Python+NumPy programs, 2018. URL http://github.com/google/jax.

Brandstetter, J., Worrall, D., and Welling, M. Message passing neural PDE solvers. *arXiv preprint arXiv:2202.03376*, 2022.

Brezis, H. Remarks on the Monge–Kantorovich problem in the discrete setting. *Comptes Rendus. Mathématique*, 356(2):207–213, 2018.

Bruno, O. P., Hesthaven, J. S., and Leibovici, D. V. FC-based shock-dynamics solver with neural-network localized artificial-viscosity assignment. *arXiv:2111.01315 [cs, math]*, November 2021. arXiv: 2111.01315.

Brunton, S. L. and Kutz, J. N. *Data-driven science and engineering: Machine learning, dynamical systems, and control*. Cambridge University Press, 2022.

Canuto, C. G., Hussaini, M. Y., Quarteroni, A. M., and Zang, T. A. *Spectral Methods: Evolution to Complex Geometries and Applications to Fluid Dynamics (Scientific Computation)*. Springer-Verlag, Berlin, Heidelberg, 2007. ISBN 3540307273.

Chaturantabut, S. and Sorensen, D. C. Nonlinear model reduction via discrete empirical interpolation. *SIAM Journal on Scientific Computing*, 32(5):2737–2764, 2010. doi: 10.1137/090766498. URL https://doi.org/10.1137/090766498.

Chen, T., Xu, B., Zhang, C., and Guestrin, C. Training deep nets with sublinear memory cost. *arXiv preprint arXiv:1604.06174*, 2016.

Chen, Y., Zhang, L., Wang, H., and E, W. DeePKS: A comprehensive data-driven approach toward chemically accurate density functional theory. *Journal of Chemical Theory and Computation*, 17(1):170–181, 2020.

Chinesta, F., Ladeveze, P., and Cueto, E. A short review on model order reduction based on proper generalized decomposition. *Archives of Computational Methods in Engineering*, 18(4):395–404, 2011.

Cooley, J. W. and Tukey, J. W. An algorithm for the machine calculation of complex Fourier series. *Math. Comput.*, 19 (90):297–301, 1965. ISSN 00255718, 10886842. URL http://www.jstor.org/stable/2003354.

Cuturi, M. Sinkhorn distances: Lightspeed computation of optimal transport. *Advances in neural information processing systems*, 26, 2013.

Cuturi, M., Meng-Papaxanthos, L., Tian, Y., Bunne, C., Davis, G., and Teboul, O. Optimal transport tools (ott): A JAX toolbox for all things Wasserstein. *arXiv preprint arXiv:2201.12324*, 2022.

Dresdner, G., Kochkov, D., Norgaard, P., Zepeda-Núñez, L., Smith, J. A., Brenner, M. P., and Hoyer, S. Learning to correct spectral methods for simulating turbulent flows. *arXiv preprint arXiv:2207.00556*, 2022.

Dziugaite, G. K., Roy, D. M., and Ghahramani, Z. Training generative neural networks via maximum mean discrepancy optimization. *arXiv preprint arXiv:1505.03906*, 2015.

Fan, H., Jiang, J., Zhang, C., Wang, X., and Lai, Y.-C. Long-term prediction of chaotic systems with machine learning. *Physical Review Research*, 2(1):012080, 2020.

Fan, Y., Feliu-Fabà, J., Lin, L., Ying, L., and Zepeda-Núñez, L. A multiscale neural network based on hierarchical nested bases. *Research in the Mathematical Sciences*, 6, Mar. 2019. ISSN 2197-9847. doi: 10.1007/s40687-019-0183-3.

Farmer, J. D. and Sidorowich, J. J. Predicting chaotic time series. *Phys. Rev. Lett.*, 59:845–848, Aug 1987. doi: 10.1103/PhysRevLett.59.845. URL https://link.aps.org/doi/10.1103/PhysRevLett.59.845.

Ferrario, B. Invariant measures for a stochastic kuramoto–sivashinsky equation. *Stochastic analysis and applications*, 26(2):379–407, 2008.

Feydy, J., Séjourné, T., Vialard, F.-X., Amari, S.-i., Trouvé, A., and Peyré, G. Interpolating between optimal transport and MMD using sinkhorn divergences. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pp. 2681–2690. PMLR, 2019.

Frezat, H., Le Sommer, J., Fablet, R., Balarac, G., and Lguensat, R. A posteriori learning for quasi-geostrophic turbulence parametrization. *arXiv*, April 2022.

Galerkin, B. G. Series occurring in various questions concerning the elastic equilibrium of rods and plates. *Vestnik Inzhenernov i Tekhnikov*, 19:897–908, 1915.

Geelen, R., Wright, S., and Willcox, K. Operator inference for non-intrusive model reduction with nonlinear manifolds. *arXiv:2205.02304 [math.NA]*, May 2022. URL https://arxiv.org/abs/2205.02304. arXiv: 2205.02304.

Genevay, A., Peyré, G., and Cuturi, M. Learning generative models with sinkhorn divergences. In *International Conference on Artificial Intelligence and Statistics*, pp. 1608–1617. PMLR, 2018.

Ghadami, A. and Epureanu, B. I. Data-driven prediction in dynamical systems: recent developments. *Philosophical Transactions of the Royal Society A*, 380(2229):20210213, 2022.

Gneiting, T. and Raftery, A. E. Strictly proper scoring rules, prediction, and estimation. *Journal of the American statistical Association*, pp. 359–378, 2007.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020.

Gretton, A., Borgwardt, K., Rasch, M., Schölkopf, B., and Smola, A. A kernel method for the two-sample-problem. *Advances in neural information processing systems*, 19, 2006.

Gretton, A., Borgwardt, K. M., Rasch, M. J., Schölkopf, B., and Smola, A. A kernel two-sample test. *The Journal of Machine Learning Research*, 13(1):723–773, 2012.

Hara, M. and Kokubu, H. Learning dynamics by reservoir computing (in memory of prof. pavol brunovskỳ). *Journal of Dynamics and Differential Equations*, pp. 1–26, 2022.

Harris, C. R., Millman, K. J., van der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., Kern, R., Picus, M., Hoyer, S., van Kerkwijk, M. H., Brett, M., Haldane, A., del Río, J. F., Wiebe, M., Peterson, P., Gérard-Marchant, P., Sheppard, K., Reddy, T., Weckesser, W., Abbasi, H., Gohlke, C., and Oliphant, T. E. Array programming with NumPy. *Nature*, 585(7825):357–362, September 2020. doi: 10.1038/s41586-020-2649-2. URL https://doi.org/10.1038/s41586-020-2649-2.

Hawkins, J. Attractors in dynamical systems. *Ergodic dynamics: From basic theory to applications*, pp. 27–39, 2021.

Heek, J., Levskaya, A., Oliver, A., Ritter, M., Rondepierre, B., Steiner, A., and van Zee, M. Flax: A neural network library and ecosystem for JAX, 2023. URL http://github.com/google/flax.

Hess, F., Monfared, Z., Brenner, M., and Durstewitz, D. Generalized teacher forcing for learning chaotic dynamics. *arXiv preprint arXiv:2306.04406*, 2023.

Hoyer, S. and Hamman, J. xarray: N-D labeled arrays and datasets in Python. *Journal of Open Research Software*, 5 (1), 2017. doi: 10.5334/jors.148. URL https://doi.org/10.5334/jors.148.

Hunter, J. D. Matplotlib: A 2d graphics environment. *Computing in Science & Engineering*, 9(3):90–95, 2007. doi: 10.1109/MCSE.2007.55.

Jia, W., Wang, H., Chen, M., Lu, D., Lin, L., Car, R., Weinan, E., and Zhang, L. Pushing the limit of molecular dynamics with ab initio accuracy to 100 million atoms

with machine learning. In *SC20: International conference for high performance computing, networking, storage and analysis*, pp. 1–14. IEEE, 2020.

Jiang, R., Lu, P. Y., Orlova, E., and Willett, R. Training neural operators to preserve invariant measures of chaotic attractors. *arXiv preprint arXiv:2306.01187*, 2023.

Kaiser, E., Kutz, J. N., and Brunton, S. L. Data-driven discovery of Koopman eigenfunctions for control. *Machine Learning: Science and Technology*, 2(3):035023, 2021.

Kantorovich, L. V. On the translocation of masses. In *Dokl. Akad. Nauk. USSR (NS)*, volume 37, pp. 199–201, 1942.

Keisler, R. Forecasting global weather with graph neural networks. *arXiv preprint arXiv:2202.07575*, 2022.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Kochkov, D., Smith, J. A., Alieva, A., Wang, Q., Brenner, M. P., and Hoyer, S. Machine learning–accelerated computational fluid dynamics. *Proc. Natl. Acad. Sci. U. S. A.*, 118(21), May 2021. URL https://www.pnas.org/content/118/21/e2101784118.

Kochkov, D., Yuval, J., Langmore, I., Norgaard, P., Smith, J., Mooers, G., Lottes, J., Rasp, S., Düben, P., Klöwer, M., et al. Neural general circulation models. *arXiv preprint arXiv:2311.07222*, 2023.

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

Krishnapriyan, A., Gholami, A., Zhe, S., Kirby, R., and Mahoney, M. W. Characterizing possible failure modes in physics-informed neural networks. *Advances in Neural Information Processing Systems*, 34:26548–26560, 2021.

Kuramoto, Y. Diffusion-induced chaos in reaction systems. *Progress of Theoretical Physics Supplement*, 64:346–367, 1978.

Kutta, W. *Beitrag zur näherungsweisen Integration totaler Differentialgleichungen*. Teubner, 1901.

Lam, R., Sanchez-Gonzalez, A., Willson, M., Wirnsberger, P., Fortunato, M., Alet, F., Ravuri, S., Ewalds, T., Eaton-Rosen, Z., Hu, W., et al. Graphcast: Learning skillful medium-range global weather forecasting. *arXiv preprint arXiv:2212.12794*, 2022.

Lamb, A. M., ALIAS PARTH GOYAL, A. G., Zhang, Y., Zhang, S., Courville, A. C., and Bengio, Y. Professor forcing: A new algorithm for training recurrent networks. *Advances in neural information processing systems*, 29, 2016.

Li, C.-L., Chang, W.-C., Cheng, Y., Yang, Y., and Póczos, B. MMD GAN: Towards deeper understanding of moment matching network. *Advances in neural information processing systems*, 30, 2017.

Li, Y., Swersky, K., and Zemel, R. Generative moment matching networks. In *International conference on machine learning*, pp. 1718–1727. PMLR, 2015.

Li, Z., Kovachki, N., Azizzadenesheli, K., Liu, B., Bhattacharya, K., Stuart, A., and Anandkumar, A. Multipole graph neural operator for parametric partial differential equations. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS'20, Red Hook, NY, USA, 2020. Curran Associates Inc. ISBN 9781713829546.

Li, Z., Kovachki, N., Azizzadenesheli, K., Liu, B., Bhattacharya, K., Stuart, A., and Anandkumar, A. Fourier Neural Operator for Parametric Partial Differential Equations. *arXiv:2010.08895 [cs, math]*, May 2021. URL http://arxiv.org/abs/2010.08895. arXiv: 2010.08895.

Li, Z., Liu-Schiaffini, M., Kovachki, N., Azizzadenesheli, K., Liu, B., Bhattacharya, K., Stuart, A., and Anandkumar, A. Learning chaotic dynamics in dissipative systems. *Advances in Neural Information Processing Systems*, 35: 16768–16781, 2022.

List, B., Chen, L.-W., and Thuerey, N. Learned Turbulence Modelling with Differentiable Fluid Solvers. *arXiv:2202.06988 [physics]*, February 2022. URL http://arxiv.org/abs/2202.06988. arXiv: 2202.06988.

Liu, F., Xu, W., Lu, J., Zhang, G., Gretton, A., and Sutherland, D. J. Learning deep kernels for non-parametric two-sample tests. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119, pp. 6316–6326. PMLR, 2020.

Lorenz, E. N. Deterministic nonperiodic flow. *Journal of atmospheric sciences*, 20(2):130–141, 1963.

Lu, F., Lin, K. K., and Chorin, A. J. Data-based stochastic model reduction for the Kuramoto–Sivashinsky equation. *Physica D: Nonlinear Phenomena*, 340:46–57, 2017.

Lu, L., Jin, P., Pang, G., Zhang, Z., and Karniadakis, G. E. Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators. *Nature Machine Intelligence*, 3(3):218–229, 2021.

Luzzatto, S., Melbourne, I., and Paccaut, F. The lorenz attractor is mixing. *Communications in Mathematical Physics*, 260:393–401, 2005.

Mathews, A., Francisquez, M., Hughes, J. W., Hatch, D. R., Zhu, B., and Rogers, B. N. Uncovering turbulent plasma dynamics via deep learning from partial observations. *Physical Review E*, 104(2):025205, 2021.

Medio, A. and Lines, M. *Nonlinear dynamics: A primer*. Cambridge University Press, 2001.

Merchant, A., Batzner, S., Schoenholz, S. S., Aykol, M., Cheon, G., and Cubuk, E. D. Scaling deep learning for materials discovery. *Nature*, pp. 1–6, 2023.

Mikhaeil, J., Monfared, Z., and Durstewitz, D. On the difficulty of learning chaotic dynamics with RNNs. *Advances in Neural Information Processing Systems*, 35: 11297–11312, 2022.

Mishra, S. A machine learning framework for data driven acceleration of computations of differential equations. *Mathematics in Engineering*, 1(1):118–146, 2018. ISSN 2640-3501. doi: 10.3934/Mine.2018.1.118. URL https://www.aimspress.com/article/doi/10.3934/Mine.2018.1.118.

Müller, A. Integral probability metrics and their generating classes of functions. *Advances in applied probability*, 29 (2):429–443, 1997.

Nghiem, T. X., Drgoňa, J., Jones, C., Nagy, Z., Schwan, R., Dey, B., Chakrabarty, A., Di Cairano, S., Paulson, J. A., Carron, A., et al. Physics-informed machine learning for modeling and control of dynamical systems. *arXiv preprint arXiv:2306.13867*, 2023.

Obukhov, A. Kolmogorov flow and laboratory simulation of it. *Russ. Math. Surv*, 38(4):113–126, 1983.

pandas development team, T. pandas-dev/pandas: Pandas, February 2020. URL https://doi.org/10.5281/zenodo.3509134.

Papageorgiou, D. T. and Smyrlis, Y. S. The route to chaos for the kuramoto-sivashinsky equation. *Theoretical and Computational Fluid Dynamics*, 3(1):15–42, 1991.

Papamakarios, G., Nalisnick, E., Rezende, D. J., Mohamed, S., and Lakshminarayanan, B. Normalizing flows for probabilistic modeling and inference. *The Journal of Machine Learning Research*, 22(1):2617–2680, 2021.

Pascanu, R., Mikolov, T., and Bengio, Y. On the difficulty of training recurrent neural networks. In *International conference on machine learning*, pp. 1310–1318. Pmlr, 2013.

Pathak, J., Lu, Z., Hunt, B. R., Girvan, M., and Ott, E. Using machine learning to replicate chaotic attractors and calculate Lyapunov exponents from data. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 27(12), 2017.

Pathak, J., Subramanian, S., Harrington, P., Raja, S., Chattopadhyay, A., Mardani, M., Kurth, T., Hall, D., Li, Z., Azizzadenesheli, K., et al. Fourcastnet: A global data-driven high-resolution weather model using adaptive fourier neural operators. *arXiv preprint arXiv:2202.11214*, 2022.

Pearson, K. Liii. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11):559–572, 1901. doi: 10.1080/14786440109462720. URL https://doi.org/10.1080/14786440109462720.

Peherstorfer, B. Breaking the kolmogorov barrier with nonlinear model reduction. *Notices of the American Mathematical Society*, 69(5):725–733, 2022.

Peyré, G., Cuturi, M., et al. Computational optimal transport: With applications to data science. *Foundations and Trends® in Machine Learning*, 11(5-6):355–607, 2019.

Platt, J. A., Wong, A., Clark, R., Penny, S. G., and Abarbanel, H. D. Robust forecasting using predictive generalized synchronization in reservoir computing. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 31(12), 2021.

Platt, J. A., Penny, S. G., Smith, T. A., Chen, T.-C., and Abarbanel, H. D. Constraining chaos: Enforcing dynamical invariants in the training of recurrent neural networks. *arXiv preprint arXiv:2304.12865*, 2023.

Rajendra, P. and Brahmajirao, V. Modeling of dynamical systems through deep learning. *Biophysical Reviews*, 12 (6):1311–1320, 2020.

Ramachandran, P., Zoph, B., and Le, Q. V. Searching for activation functions. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Workshop Track Proceedings*. OpenReview.net, 2018. URL https://openreview.net/forum?id=Hkuq2EkPf.

Ramdas, A., García Trillos, N., and Cuturi, M. On wasserstein two-sample testing and related families of nonparametric tests. *Entropy*, 19(2):47, 2017.

Rasmussen, C. E., Williams, C. K., et al. *Gaussian processes for machine learning*, volume 1. Springer, 2006.

Rezende, D. and Mohamed, S. Variational inference with normalizing flows. In *International conference on machine learning*, pp. 1530–1538. PMLR, 2015.

Roy, S. and Rana, D. Machine learning in nonlinear dynamical systems. *Resonance*, 26(7):953–970, 2021.

Runge, C. Über die numerische auflösung von differentialgleichungen. *Mathematische Annalen*, 46(2):167–178, 1895.

Sanchez-Gonzalez, A., Godwin, J., Pfaff, T., Ying, R., Leskovec, J., and Battaglia, P. Learning to simulate complex physics with graph networks. In *International conference on machine learning*, pp. 8459–8468. PMLR, 2020.

Santambrogio, F. Optimal transport for applied mathematicians. *Birkäuser, NY*, 55(58-63):94, 2015.

Schmid, P. J. Dynamic mode decomposition of numerical and experimental data. *Journal of Fluid Mechanics*, 656: 5–28, 2010. doi: 10.1017/S0022112010001217.

Schmid, P. J. Dynamic mode decomposition and its variants. *Annual Review of Fluid Mechanics*, 54:225–254, 2022.

Schrab, A., Kim, I., Albert, M., Laurent, B., Guedj, B., and Gretton, A. MMD aggregated two-sample test. *Journal of Machine Learning Research*, 24(194):1–81, 2023.

Sejdinovic, D., Sriperumbudur, B., Gretton, A., and Fukumizu, K. Equivalence of distance-based and RKHS-based statistics in hypothesis testing. *The Annals of Statistics*, 41 (5):2263 – 2291, 2013. doi: 10.1214/13-AOS1140. URL https://doi.org/10.1214/13-AOS1140.

Serrano, L., Boudec, L. L., Koupaï, A. K., Wang, T. X., Yin, Y., Vittaut, J.-N., and Gallinari, P. Operator learning with neural fields: Tackling PDEs on general geometries. *arXiv preprint arXiv:2306.07266*, 2023.

Si, P., Bishop, A., and Kuleshov, V. Autoregressive quantile flows for predictive uncertainty estimation. *arXiv preprint arXiv:2112.04643*, 2021.

Si, P., Chen, Z., Sahoo, S. S., Schiff, Y., and Kuleshov, V. Semi-autoregressive energy flows: exploring likelihood-free training of normalizing flows. In *International Conference on Machine Learning*, pp. 31732–31753. PMLR, 2023.

Simon-Gabriel, C.-J., Barp, A., Schölkopf, B., and Mackey, L. Metrizing weak convergence with maximum mean discrepancies. *Journal of Machine Learning Research*, 24(184):1–20, 2023.

Sivashinsky, G. I. Nonlinear analysis of hydrodynamic instability in laminar flames—i. derivation of basic equations. In *Dynamics of Curved Fronts*, pp. 459–488. Elsevier, 1988.

Sriperumbudur, B. K., Fukumizu, K., Gretton, A., Schölkopf, B., and Lanckriet, G. R. On integral probability metrics,\phi-divergences and binary classification. *arXiv preprint arXiv:0901.2698*, 2009.

Sriperumbudur, B. K., Gretton, A., Fukumizu, K., Schölkopf, B., and Lanckriet, G. R. Hilbert space embeddings and metrics on probability measures. *The Journal of Machine Learning Research*, 11:1517–1561, 2010.

Stachenfeld, K., Fielding, D. B., Kochkov, D., Cranmer, M., Pfaff, T., Godwin, J., Cui, C., Ho, S., Battaglia, P., and Sanchez-Gonzalez, A. Learned Coarse Models for Efficient Turbulence Simulation. *arXiv:2112.15275 [physics]*, January 2022. URL http://arxiv.org/abs/2112.15275. arXiv: 2112.15275.

Strogatz, S. H. *Nonlinear dynamics and chaos with student solutions manual: With applications to physics, biology, chemistry, and engineering*. CRC press, 2018.

Stuart, A. and Humphries, A. R. *Dynamical systems and numerical analysis*, volume 2. Cambridge University Press, 1998.

Stuart, A. M. Numerical analysis of dynamical systems. *Acta numerica*, 3:467–572, 1994.

Székely, G. J., Rizzo, M. L., et al. Testing for equal distributions in high dimension. *InterStat*, 5(16.10):1249–1272, 2004.

Temam, R. *Infinite-dimensional dynamical systems in mechanics and physics*, volume 68. Springer Science & Business Media, 2012.

Tolstikhin, I. O., Sriperumbudur, B. K., and Schölkopf, B. Minimax estimation of maximum mean discrepancy with radial kernels. In Lee, D., Sugiyama, M., Luxburg, U., Guyon, I., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 29, 2016. URL https://proceedings.neurips.cc/paper_files/paper/2016/file/5055cbf43fac3f7e2336b27310f0b9ef-Paper.pdf.

Tran, A., Mathews, A., Xie, L., and Ong, C. S. Factorized Fourier Neural Operators. *arXiv:2111.13802 [cs]*, November 2021. arXiv: 2111.13802.

Trefethen, L. N. *Spectral Methods in MATLAB*. Society for industrial and applied mathematics (SIAM), 2000.

Tucker, W. The Lorenz attractor exists. *Comptes Rendus de l'Académie des Sciences-Series I-Mathematics*, 328(12): 1197–1202, 1999.

Tucker, W. A rigorous ODE solver and Smale's 14th problem. *Foundations of Computational Mathematics*, 2:53–117, 2002.

Um, K., Brand, R., Fei, Y. R., Holl, P., and Thuerey, N. Solver-in-the-loop: Learning from differentiable physics to interact with iterative pde-solvers. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H. (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 6111–6122. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/43e4e6a6f341e00671e123714de019a8-Paper.pdf.

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S. J., Brett, M., Wilson, J., Millman, K. J., Mayorov, N., Nelson, A. R. J., Jones, E., Kern, R., Larson, E., Carey, C. J., Polat, İ., Feng, Y., Moore, E. W., VanderPlas, J., Laxalde, D., Perktold, J., Cimrman, R., Henriksen, I., Quintero, E. A., Harris, C. R., Archibald, A. M., Ribeiro, A. H., Pedregosa, F., van Mulbregt, P., and SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020. doi: 10.1038/s41592-019-0686-2.

Vlachas, P. R., Byeon, W., Wan, Z. Y., Sapsis, T. P., and Koumoutsakos, P. Data-driven forecasting of high-dimensional chaotic systems with long short-term memory networks. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 474(2213): 20170844, 2018.

Vlachas, P. R., Pathak, J., Hunt, B. R., Sapsis, T. P., Girvan, M., Ott, E., and Koumoutsakos, P. Backpropagation algorithms and reservoir computing in recurrent neural networks for the forecasting of complex spatiotemporal dynamics. *Neural Networks*, 126:191–217, 2020.

Wan, Z. Y., Baptista, R., Boral, A., Chen, Y.-F., Anderson, J., Sha, F., and Zepeda-Núñez, L. Debias coarsely, sample conditionally: Statistical downscaling through optimal transport and probabilistic diffusion models. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023a. URL https://openreview.net/forum?id=5NxJuc0T1P.

Wan, Z. Y., Zepeda-Núñez, L., Boral, A., and Sha, F. Evolve smoothly, fit consistently: Learning smooth latent dynamics for advection-dominated systems. In *The Eleventh International Conference on Learning Representations*, 2023b.

Wang, Q., Hu, R., and Blonigan, P. Least squares shadowing sensitivity analysis of chaotic limit cycle oscillations. *Journal of Computational Physics*, 267:210–224, 2014.

Waskom, M. L. seaborn: statistical data visualization. *Journal of Open Source Software*, 6(60):3021, 2021. doi: 10.21105/joss.03021. URL https://doi.org/10.21105/joss.03021.

Watt-Meyer, O., Dresdner, G., McGibbon, J., Clark, S. K., Henn, B., Duncan, J., Brenowitz, N. D., Kashinath, K., Pritchard, M. S., Bonev, B., et al. ACE: A fast, skillful learned global atmospheric model for climate prediction. *arXiv preprint arXiv:2310.02074*, 2023.

Weinan, E. and Liu, D. Gibbsian dynamics and invariant measures for stochastic dissipative PDEs. *Journal of Statistical Physics*, 108:1125–1156, 2002.

Willcox, K. Unsteady flow sensing and estimation via the gappy proper orthogonal decomposition. *Computers & Fluids*, 35(2):208–226, 2006. ISSN 0045-7930. doi: https://doi.org/10.1016/j.compfluid.2004.11.006. URL https://www.sciencedirect.com/science/article/pii/S0045793005000113.

Zepeda-Núñez, L., Chen, Y., Zhang, J., Jia, W., Zhang, L., and Lin, L. Deep density: circumventing the Kohn-Sham equations via symmetry preserving neural networks. *Journal of Computational Physics*, 443:110523, 2021.

A. Relation to the Pushforward trick

We note that the pushforward trick (Brandstetter et al., 2022) can be reformulated using our framework as a weak measure fitting loss. A finite-sample approximation of $\mathbb{E}_{\boldsymbol{u}_0\sim\mu^*}[||\mathcal{S}_\theta(\mathtt{sg}(\mathcal{S}_\theta^{k-1}(\boldsymbol{u}_0))) - \boldsymbol{u}_k||^2]$ is an upper bound of the discrete Wasserstein-2 distance between $\mu^\star$ and the approximation of $\mu_\theta^\star$. Formally, we have that

$$\mathcal{L}^{\text{Pfwd}}(\theta) = \mathbb{E}_{\boldsymbol{u}_0\sim\mu^*}[||\mathcal{S}_\theta(\mathtt{sg}(\mathcal{S}_\theta^{k-1}(\boldsymbol{u}_0))) - \boldsymbol{u}_k||^2] \gtrsim \mathcal{W}_2(\mu, \mu_\theta^*) := \inf_{\gamma\in\Gamma(\mu^*,\mu_\theta^*)} \int \|\boldsymbol{u}-\boldsymbol{v}\|^2 \, d\gamma(\boldsymbol{u},\boldsymbol{v}), \tag{14}$$

where $\Gamma(\mu^*, \mu_\theta^*)$ is the set of all couplings between μ^* and μ_θ^*.

By relying on an estimate of the loss, we have that for a given set of initial conditions $\{\boldsymbol{u}^{(i)}\}_{i=1}^n \sim \mu^*$,

$$\widehat{\mathcal{L}}^{\text{Pfwd}}(\theta) := n^{-1}\Sigma_{i=1}^n \|\mathcal{S}_\theta(\mathtt{sg}(\mathcal{S}_\theta^{k-1}(\boldsymbol{u}^{(i)}))) - \mathcal{S}^k(\boldsymbol{u}^{(i)})\|^2 = n^{-1}\Sigma_{i=1}^n \|\mathcal{S}_\theta^k(\boldsymbol{u}^{(i)}) - \mathcal{S}^k(\boldsymbol{u}^{(i)})\|^2, \tag{15}$$

which can be lower bounded by the following

$$n^{-1}\Sigma_{i=1}^n \|\mathcal{S}_\theta^k(\boldsymbol{u}^{(i)}) - \mathcal{S}^k(\boldsymbol{u}^{(i)})\|^2 \geq n^{-1} \min_\pi \Sigma_{i=1}^n \|\mathcal{S}^k(\boldsymbol{u}^{(i)}) - \mathcal{S}_\theta^k(\boldsymbol{u}^{(\pi(i))})\|^2,$$

where π is a permutation operator. Given that we are in the discrete setting where the Monge and Kantorovich problems are equivalent (Brezis, 2018), we have that

$$n^{-1} \min_\pi \Sigma_{i=1}^n \|\mathcal{S}^k(\boldsymbol{u}^{(i)}) - \mathcal{S}_\theta^k(\boldsymbol{u}^{(\pi(i))})\|^2 = \inf_{T\in\Pi} \Sigma_{i,j} T_{i,j} C_{i,j} := \widehat{W}_2((\mathcal{S}_\theta^k)_\sharp\mu^*, (\mathcal{S}^k)_\sharp\mu^*), \tag{16}$$

where Π is the set of all valid discrete transport maps (i.e., matrices that satisfy $T_{i,j} \geq 0$, $\sum_j T_{i,j} = \sum_i T_{i,j} = 1$), C is the quadratic cost function ($C_{i,j} = \|\mathcal{S}^k(\boldsymbol{u}^{(i)}) - \mathcal{S}_\theta^k(\boldsymbol{u}^{(j)})\|^2$), and $\widehat{W}_2$ is a discrete estimate of the Wasserstein-2 metric.

We can further refine this expression using the same approximation as in Equation 11, i.e., $\mathcal{S}_\theta^k(\boldsymbol{u}^{(i)}) \sim \mu_\theta^*$ for large k and $\mathcal{S}^k(\boldsymbol{u}^{(i)}) \sim \mu^*$, we have that

$$\widehat{W}_2((\mathcal{S}_\theta^k)_\sharp\mu^*, (\mathcal{S}^k)_\sharp\mu^*) \approx \widehat{W}_2(\mu_\theta^*, \mu^*). \tag{17}$$

Therefore, in summary we have that

$$\hat{\mathcal{L}}^{\text{Pfwd}}(\theta) \gtrsim \widehat{W}_2(\mu_\theta^*, \mu^*) \approx \inf_{\gamma\in\Gamma(\mu^*,\mu_\theta^*)} \int \|\boldsymbol{u}-\boldsymbol{v}\|^2 \, d\gamma(\boldsymbol{u},\boldsymbol{v}) = \mathcal{W}_2(\mu, \mu_\theta^*). \tag{18}$$

Thus one can argue that minimizing the Pfwd objective also induces a minimization of the discrete Wasserstein-2 metric between the two invariant measures.

B. Maximum Mean Discrepancy

In this section, we provide additional information and context about the Maximum Mean Discrepancy (MMD). The MMD is an instance of an integral probability metric (IPM; Müller (1997), which is a useful construction that allows us to measure distance between distributions. For any two distributions μ and ν, IPMs are defined with a function class $\mathcal{G}$ as:

$$\text{IPM}(\mu, \nu) = \sup_{g\in\mathcal{G}} \left|\mathbb{E}_{\boldsymbol{u}\sim\mu}[g(\boldsymbol{u})] - \mathbb{E}_{\boldsymbol{u}\sim\nu}[g(\boldsymbol{u})]\right|. \tag{19}$$

Given that we seek our model $\mathcal{S}_\theta$ to preserve μ^*, we can use an IPM as the distance D in Equation 9, since, for a rich enough function class $\mathcal{G}$, $\text{IPM}(\mu^*, \mathcal{S}_{\theta\#}\mu^*) \to 0$ implies $\mathcal{S}_{\theta\#}\mu^* \to \mu^*$.

One instance of an IPM is when $\mathcal{G}$ is the space of functions with bounded norm in a reproducing kernel Hilbert space $\mathcal{H}_\kappa$, i.e., $\mathcal{G} = \{g : ||g||_{\mathcal{H}_\kappa} \leq 1\}$, in which case, Equation 19 coincides with the Maximum Mean Discrepancy (MMD) (Gretton et al., 2012; Sriperumbudur et al., 2009), where κ is the reproduced kernel. Using the reproducing property of $\mathcal{H}_\kappa$ and the Riesz representation theorem, we have that the MMD can be expressed as follows:

$$\text{MMD}^2 = ||\mathbb{E}_{\mu^*}[\kappa(\boldsymbol{u}, \cdot)] - \mathbb{E}_\nu[\kappa(\boldsymbol{v}, \cdot)]||^2_{\mathcal{H}_\kappa}, \tag{20}$$

where $\mathbb{E}_\mu[\kappa(\boldsymbol{u}, \cdot)]$ is the mean embedding of μ (Gretton et al., 2012). Applying the reproducing property of $\mathcal{H}_\kappa$ again allows us to equivalently write Equation 20 as in Equation 10 (Gretton et al., 2012).

As described in Section 5, we use a rational quadratic kernel. While other works that use MMD for distribution matching, such as Li et al. (2015) and Dziugaite et al. (2015), also explored the squared exponential kernel, $\kappa_\sigma(\boldsymbol{u}, \boldsymbol{v}) = \exp(\frac{-1}{2\sigma}||\boldsymbol{u} - \boldsymbol{v}||_2^2)$, they found that careful tuning of the bandwidth parameter was required. In contrast, other than the highest dimension Kolmogorov flow experiments, we found that the mixture of bandwidths used in our rational kernel was comparatively robust and did not require a comprehensive hyperparameter search. We therefore rely on this kernel and do not explore the more sensitive squared exponential kernel.

C. Evaluation Criteria

In this section, we provide further detail about the evaluation criteria used in Section 5.

C.1. Cosine Similarity

Letting $\{\boldsymbol{u}_{t_k}^{(i)}\}_{i=1}^n$ and $\{\tilde{\boldsymbol{u}}_{t_k}^{(i)}\}_{i=1}^n$ be the ground truth and predicted states (respectively) at time t_k, for $k = 1, ..., N$, across test set trajectories, the cosine similarity at each time step is defined as:

$$\text{avg. cosine sim}(t_k) = \frac{1}{n}\sum_{i=1}^{n} \frac{(\boldsymbol{u}_{t_k}^{(i)} - \bar{\boldsymbol{u}}_{t_k})^\top (\tilde{\boldsymbol{u}}_{t_k}^{(i)} - \bar{\boldsymbol{u}}_{t_k})}{||(\boldsymbol{u}_{t_k}^{(i)} - \bar{\boldsymbol{u}}_{t_k})|| \cdot ||(\tilde{\boldsymbol{u}}_{t_k}^{(i)} - \bar{\boldsymbol{u}}_{t_k})||},$$

where $\bar{\boldsymbol{u}}_{t_k} = \frac{1}{n}\sum_{i=1}^n \boldsymbol{u}_{t_k}$ is the mean of the ground truth trajectories at each time step. Here $t_k = k \cdot \Delta t$ refers to number of discrete time steps multiplied by the time resolution of the trajectories. Intuitively this metric provides the angle between the different trajectories, i.e., it measures if the snapshots are "pointing" in the same direction.

C.2. Sinkhorn Divergence

Popular metrics used to measure distance between distributions include Optimal Transport (OT) based metrics, such as the Sinkhorn divergence, which we describe below. The field of OT is concerned with transforming (or transporting) one distribution into another, i.e., finding a map between them, in an optimal manner with respect to a pre-defined cost. The cost of the minimal (or optimal) transformation, often called the cost of the OT map, can then be used to define distances between distributions that 'lifts' the underlying metric d defined on $\mathcal{U}$ to one over the space of probability measures $\mathcal{P}(\mathcal{U})$ (Santambrogio, 2015).

In this context, we define the Kantorovich formulation of the OT cost (Kantorovich, 1942) as

$$\mathcal{W}(\mu, \nu) = \min_{\gamma \in \Gamma(\mu,\nu)} \int_{\mathcal{U}\times\mathcal{U}} c(\boldsymbol{u}, \boldsymbol{v}) d\gamma(\boldsymbol{u}, \boldsymbol{v}),$$

where $c : \mathcal{U} \times \mathcal{U} \to \mathbb{R}^+$ is an arbitrary cost function for transporting a unit of mass from $\boldsymbol{u}$ to $\boldsymbol{v}$, and Γ is the set of joint distributions defined on $\mathcal{U} \times \mathcal{U}$ with correct marginals, i.e.,

$$\Gamma(\mu, \nu) = \{\gamma \in \mathcal{P}(\mathcal{U} \times \mathcal{U}) \mid P_{1\#}\gamma = \mu, P_{2\#}\gamma = \nu\},$$

with $P_1(\boldsymbol{u}, \boldsymbol{v}) = \boldsymbol{u}$ and $P_2(\boldsymbol{u}, \boldsymbol{v}) = \boldsymbol{v}$ being simple projection operators. When $c(\boldsymbol{u}, \boldsymbol{v}) = \mathrm{d}(\boldsymbol{u}, \boldsymbol{v})^p$ with $p \geq 1$, then $\mathcal{W}^{1/p}$ is known as a Wasserstein-p distance.

Practically, finding OT maps is a computationally expensive procedure. We therefore use entropic regularized versions of OT costs, which are amenable to efficient implementation on computational accelerators, by means of the Sinkhorn algorithm (Cuturi, 2013; Peyré et al., 2019):

$$\mathcal{W}_\varepsilon(\mu, \nu) = \min_{\gamma \in \Gamma(\mu,\nu)} \mathcal{W} + \mathrm{KL}(\gamma||\mu \otimes \nu), \tag{21}$$

where KL is the Kullback-Leibler divergence, and $\mu \otimes \nu$ is the product of the marginal distributions. This gives rise to the Sinkhorn Divergence (SD):

$$\mathrm{SD}(\mu, \nu) = 2\mathcal{W}_\varepsilon(\mu, \nu) - \mathcal{W}_\varepsilon(\mu, \mu) - \mathcal{W}_\varepsilon(\nu, \nu),$$

which alleviates the entropic bias present in Equation 21, i.e. $\mathcal{W}_\varepsilon(\mu,\mu) \neq 0$. Of note, the SD can be shown to interpolate between a pure OT cost $\mathcal{W}$ (as $\varepsilon \to 0$) and a MMD (as $\varepsilon \to \infty$) (Ramachandran et al., 2018; Genevay et al., 2018; Feydy et al., 2019).

In Section 5, the SD was used to compare empirical version of the ground truth and predicted distributions of trajectories. We use the Optimal Transport Tools library (Cuturi et al., 2022) with its default hyperparameters to perform this computation. We also explored using the Sinkhorn Divergence as the measure distance in Equation 13. However, especially in higher dimension experiments, we found this divergence to be less informative in guiding training, likely owing to its less favorable estimation properties compared to the MMD, particularly in the high-dimensional regime, see Appendix G for more details.

C.3. Radially Averaged Energy Spectrum

The energy spectrum is one of the main metrics to quantitatively assess generated samples (Wan et al., 2023a). In a nutshell, the energy spectrum measures the energy in each Fourier mode, thereby providing insights into the similarity between the generated and reference samples.

The energy spectrum is defined[9] as

$$E(K) = \sum_{|\underline{K}|=K} |\hat{\boldsymbol{u}}(\underline{K})|^2 = \sum_{|\underline{K}|=K} \left| \sum_{i,j} \boldsymbol{u}(x_{i,j}) \exp(-j2\pi \underline{K} \cdot x_{i,j}/L) \right|^2 \tag{22}$$

where $\boldsymbol{u}$ is a snapshot system state, K is the magnitude of the wave-number (wave-vector in 2D) $\underline{K}$, and $x_{i,j}$ is the underlying (possibly 2D) spatial grid. To assess the overall consistency of the spectrum between the generated and reference samples using a single scalar measure, we consider the mean energy log ratio (MELR):

$$\text{MELR} = \sum_K w_K \left| \log \left(E_{\text{pred}}(K)/E_{\text{ref}}(K) \right) \right|, \tag{23}$$

where w_K represents the weight assigned to each K. We further define $w_K^{\text{unweighted}} = 1/\text{card}(K)$ and $w_K^{\text{weighted}} = E_{\text{ref}}(K)/\sum_K E_{\text{ref}}(K)$. The latter skews more towards high-energy/low-frequency modes.

C.4. Covariance RMSE (covRMSE)

The covariance root mean squared error quantifies the difference in the long-term spatial correlation structure between the prediction and the reference. It involves first computing the (empirical) covariance on a long rollout:

$$\text{Cov}(\boldsymbol{u}) = \frac{1}{N \cdot n} \sum_{i=1}^{n} \sum_{k=1}^{N} (\boldsymbol{u}_{t_k}^{(i)} - \bar{\boldsymbol{u}})(\boldsymbol{u}_{t_k}^{(i)} - \bar{\boldsymbol{u}})^T, \quad \bar{\boldsymbol{u}} = \frac{1}{N \cdot n} \sum_{i=1}^{n} \sum_{k=1}^{N} \boldsymbol{u}_{t_k}^{(i)}, \tag{24}$$

where $\boldsymbol{u}_{t_k}^{(i)}$ are realizations of the multi-dimensional random variable U (in this case, they are just the snapshots of the trajectory i at time steps t_k.) For 2D Kolmogorov flow, we leverage the translation invariance in the system to compute the covariance on slices with fixed x-coordinate. The error is then given by:

$$\text{covRMSE} = \frac{\|\text{Cov}_{\text{pred}} - \text{Cov}_{\text{ref}}\|}{\|\text{Cov}_{\text{ref}}\|}, \tag{25}$$

where $\|\cdot\|$ is taken to be the Frobenious norm.

C.5. Time Correlation Metric (TCM)

The quantities introduced above, such as the energy spectrum, are single-time quantities. Compared to single-time quantities, examining multiple-time statistics can provide a better view of more complex temporal behavior.

We leverage the spatial homogeneity and compute pointwise statistics for a scalar time series u, then average over space. Assuming stationarity, one definition of the autocorrelation function is $\rho(t) = C(t)/C(0)$, where

$$C(t_i) = \frac{1}{N} \sum_{k=1}^{N} (u_{t_k} - \bar{u})(u_{t_{k-i}} - \bar{u}), \quad \bar{u} = \frac{1}{N} \sum_{k=1}^{N} u_{t_k} \tag{26}$$

[9] This definition is applied to each sample and averaged to obtain the metric (same for MELR).

The *autocorrelation time* τ, which is defined as

$$\tau = \Delta t \left(1 + 2\sum_{i=1}^{\infty} \rho(t_i)\right), \tag{27}$$

can be interpreted as the time for the signal to forget its past. We compute the average pixel-wise τ for ground truth as well as prediction rollouts and take their absolute difference to form a metric.

D. Regularization

For better reproducibility of our work, we provide explicit formulas for the regularized objective functions. We reproduce Equation 13 from Section 5

$$\widehat{\mathcal{L}}^{\mathrm{D}}_{\lambda}(\theta) = \widehat{\mathcal{L}}^{\mathrm{obj}}(\theta) + \lambda_1 \widehat{\mathrm{D}}(\mu^*, (\mathcal{S}^{\ell}_{\theta})_{\sharp}\mu^*) + \lambda_2 \widehat{\mathrm{D}}((\mathcal{S}^{\ell})_{\sharp}\mu^*, (\mathcal{S}^{\ell}_{\theta})_{\sharp}\mu^*).$$

For each type of objective the training schedule is slightly different, namely:

- When $\widehat{\mathcal{L}}^{\mathrm{obj}}(\theta)$ corresponds to the 1-step objective, $\widehat{\mathcal{L}}^{\text{1-step}}(\theta)$, then $\ell = 1$.
- When $\widehat{\mathcal{L}}^{\mathrm{obj}}(\theta)$ corresponds to the Curr objective, $\widehat{\mathcal{L}}^{\mathrm{Curr}}(\theta)$, then we gradually increase ℓ from 1 to some maximum rollout value according to a schedule determined by the number of training steps, as described in Appendix E.1, Appendix E.2, and Appendix E.3, below.
- When $\widehat{\mathcal{L}}^{\mathrm{obj}}(\theta)$ corresponds to the Pfwd objective, we use the same schedule as in Curriculum training, but randomly sample the rollout length up to ℓ for each batch, following the implementation provided by Brandstetter et al. (2022)[10].

For the Curr objectives, we have the following formulas for the regularization terms. Suppose that $\{\boldsymbol{u}^{(i)}\}_{i=1}^{n} \sim \mu^*$ is a mini-batch of size n sampled from the invariant measure, then using the sample-based MMD estimator, the estimate of the term $\widehat{\mathrm{D}}(\mu^*, (\mathcal{S}^{\ell}_{\theta})_{\sharp}\mu^*)$ in Equation 13, i.e., the unconditional regularization term, can be written as

$$\widehat{\mathrm{MMD}}^2(\mu^*, (\mathcal{S}^k_{\theta})_{\sharp}\mu^*) = \frac{1}{n^2}\sum_{i,j}\kappa(\boldsymbol{u}^{(i)}, \boldsymbol{u}^{(j)}) + \frac{1}{n^2}\sum_{i,j}\kappa(\mathcal{S}^k_{\theta}(\boldsymbol{u}^{(i)}), \mathcal{S}^k_{\theta}(\boldsymbol{u}^{(j)})) - \frac{2}{n^2}\sum_{i,j}\kappa(\boldsymbol{u}^{(i)}, \mathcal{S}^k_{\theta}(\boldsymbol{u}^{(j)})). \tag{28}$$

The last term in Equation 13, i.e., the conditional regularization term given by $\widehat{\mathrm{D}}((\mathcal{S}^{\ell})_{\sharp}\mu^*, (\mathcal{S}^{\ell}_{\theta})_{\sharp}\mu^*)$, can be written as

$$\begin{aligned}\widehat{\mathrm{MMD}}^2((\mathcal{S}^k)_{\sharp}\mu^*, (\mathcal{S}^k_{\theta})_{\sharp}\mu^*) = &\frac{1}{n^2}\sum_{i,j}\kappa(\mathcal{S}^k(\boldsymbol{u}^{(i)}), \mathcal{S}^k(\boldsymbol{u}^{(j)})) + \frac{1}{n^2}\sum_{i,j}\kappa(\mathcal{S}^k_{\theta}(\boldsymbol{u}^{(i)}), \mathcal{S}^k_{\theta}(\boldsymbol{u}^{(j)})) \\ &- \frac{2}{n^2}\sum_{i,j}\kappa(\mathcal{S}^k(\boldsymbol{u}^{(i)}), \mathcal{S}^k_{\theta}(\boldsymbol{u}^{(j)})).\end{aligned} \tag{29}$$

Similar formulas are also presented for the Pfwd objectives, although they introduce a stop gradient in the second to last unrolling step, namely

$$\begin{aligned}\widehat{\mathrm{MMD}}^2(\mu^*, (\mathcal{S}^k_{\theta})_{\sharp}\mu^*) = &\frac{1}{n^2}\sum_{i,j}\kappa(\boldsymbol{u}^{(i)}, \boldsymbol{u}^{(j)}) + \frac{1}{n^2}\sum_{i,j}\kappa(\mathcal{S}_{\theta}(\mathtt{sg}(\mathcal{S}^{k-1}_{\theta}(\boldsymbol{u}^{(i)})), \mathcal{S}_{\theta}(\mathtt{sg}(\mathcal{S}^{k-1}_{\theta}(\boldsymbol{u}^{(j)})))) \\ &- \frac{2}{n^2}\sum_{i,j}\kappa(\boldsymbol{u}^{(i)}, \mathcal{S}_{\theta}(\mathtt{sg}(\mathcal{S}^{k-1}_{\theta}(\boldsymbol{u}^{(j)})))),\end{aligned} \tag{30}$$

and

$$\begin{aligned}\widehat{\mathrm{MMD}}^2((\mathcal{S}^k)_{\sharp}\mu^*, (\mathcal{S}^k_{\theta})_{\sharp}\mu^*) = &\frac{1}{n^2}\sum_{i,j}\kappa(\mathcal{S}^k(\boldsymbol{u}^{(i)}), \mathcal{S}^k(\boldsymbol{u}^{(j)})) + \frac{1}{n^2}\sum_{i,j}\kappa(\mathcal{S}_{\theta}(\mathtt{sg}(\mathcal{S}^{k-1}_{\theta}(\boldsymbol{u}^{(i)}))), \mathcal{S}_{\theta}(\mathtt{sg}(\mathcal{S}^{k-1}_{\theta}(\boldsymbol{u}^{(j)})))) \\ &- \frac{2}{n^2}\sum_{i,j}\kappa(\mathcal{S}^k(\boldsymbol{u}^{(i)}), \mathcal{S}_{\theta}(\mathtt{sg}(\mathcal{S}^{k-1}_{\theta}(\boldsymbol{u}^{(j)})))).\end{aligned} \tag{31}$$

[10] See https://github.com/brandstetter-johannes/MP-Neural-PDE-Solvers for more details.

E. Experimental Setup

Below, we provide information about each dynamical system from Section 5 and their corresponding experimental setup. In Table 4, we give an overview of the model, learning rate, and number of training steps used in each experiment.

Table 4. Model, learning rate, and number of training steps for each experiment in Section 5.

System	$\mathcal{S}_\theta$	LR	Training steps
Lorenz 63	MLP w/residual connection to input	$1e^{-4}$	500k
Kuramoto–Sivashinsky	Dilated convolutional network (Stachenfeld et al., 2022)	$5e^{-4}$	300k
Kolmogorov Flow	Dilated convolutional network (Stachenfeld et al., 2022)	$5e^{-4}$	720k

E.1. Lorenz 63

The Lorenz 63 model (Lorenz, 1963) is defined on a 3-dimensional state space by the following non-linear ordinary differential equation $\dot{\boldsymbol{u}} = f(\boldsymbol{u})$:

$$\begin{aligned} \dot{x} &= \sigma(y - x) \\ \dot{y} &= \rho x - y - xz \\ \dot{z} &= xy - \beta z \end{aligned} \tag{32}$$

The Lorenz 63 system is typically associated to parameter values of $\sigma = 10, \rho = 28$, and $\beta = 8/3$ and is known to be chaotic with an attractor that supports an ergodic measure (Tucker, 2002; Luzzatto et al., 2005).

Training and evaluation data were generated using a 4$^{\text{th}}$ order Runge-Kutta numerical integrator (Runge, 1895; Kutta, 1901) with time scale $\Delta t = 0.001$. We first selected random initial conditions. Trajectories were then rolled out for 100,000 warm-up steps to ensure that points were sampled from the the invariant measure supported on the Lorenz attractor. These warm-up steps were subsequently discarded. Starting from initial conditions sampled from μ^*, we generate 5,000 training trajectories each of length 100,000 steps and 20,000 test trajectories of length 1,000,000 steps. At training and evaluation time these trajectories are down-sampled along the temporal dimension by a factor of 400, so that the effective time scale was $\Delta t = 0.4$. Data were normalized to have roughly zero mean and unit variance based on statistics of the training set During training we randomly sample batches of size 2,048 that consist of 10 step windows in the training trajectories.

We define $\mathcal{S}_\theta$ as a one-step finite difference model: $\tilde{\boldsymbol{u}}_{k+1} = \mathcal{S}_\theta(\boldsymbol{u}_k) = \boldsymbol{u}_k + \Delta t f_\theta(\boldsymbol{u}_k)$, where f_θ is a parametric model of the continuous time dynamics. We parameterize f_θ by a multi-layer perceptron (MLP) with two hidden layers each of dimension 32 and use the ReLU activation function. We trained with an ADAM optimizer (Kingma & Ba, 2014) with learning rate $1e^{-4}$.

Models were trained for 500,000 steps. For the curriculum training (and its regularized counterpart), we increase ℓ by one every 50,000 training steps, and hence by the end of training $\mathcal{S}_\theta$ is predicting trajectories of length 10. For curriculum training, we weight rollout loss using a geometric weighting $\omega(k) = \max(0.1^{k-1}, 1e^{-7})$. For pushforward training we use the same rollout schedule as in curriculum training, but the rollout loss weight is $\omega(k) = \max(0.1^{k-1}, 1e^{-4})$. These weighting schemes were chosen empirically to ensure that training loss was of the same order of magnitude throughout training, even as rollout length increased. The MMD bandwidth values used were $\boldsymbol{\sigma} = \{0.2, 0.5, 0.9, 1.3\}$.

E.2. Kuramoto–Sivashinsky

The non-linear PDE known as the Kuramoto–Sivashinsky equation (KS) (Kuramoto, 1978; Sivashinsky, 1988), has the following form:

$$\partial_t \boldsymbol{u} + u\partial_x \boldsymbol{u} + \nu\partial_{xx}\boldsymbol{u} - \nu\partial_{xxxx}\boldsymbol{u} = 0 \qquad \text{in } [0, L] \times \mathbb{R}^+, \tag{33}$$

with periodic boundary conditions, and $L = 64$. Here the domain is re-scaled in order to balance the diffusion and anti-diffusion components so the solutions are chaotic (Dresdner et al., 2022).

The KS system is known to be chaotic (Papageorgiou & Smyrlis, 1991) and, when stochastically forced, ergodic with an invariant measure (Weinan & Liu, 2002; Ferrario, 2008). We generate data for this system using a spectral solver (Dresdner

et al., 2022) on a spatial grid $[0, 64]$ with 512 equally-spaced points and a 4th-order implicit-explicit Crack-Nicolson Runge-Kutta scheme (Canuto et al., 2007), with a time resolution of $\Delta t = 0.001$. For each trajectory, we start with a randomly generated initial condition given by

$$\boldsymbol{u}_0(x) = \sum_{j=1}^{n_c} a_j \sin(\omega_j * x + \phi_j), \tag{34}$$

where ω_j is chosen randomly from $\{2\pi/L, 4\pi/L, 6\pi/L\}$, a_j is sampled from a uniform distribution on $[-0.5, 0.5]$, and phase ϕ_j follows a uniform distribution on $[0, 2\pi]$. We use $n_c = 30$. We let the system "warm up" for 20 units of time, before recording the trajectories. The training dataset consists of 800 trajectories of 1,200 steps with a time sampling rate $\Delta t = 0.2$ time units, from which we randomly sample batches of size 128 and trajectory length of 10 steps. Our evaluation set consists of 100 trajectories of length 1,000 steps.

We parameterize $\mathcal{S}_\theta$ as a dilated convolution neural network with residual connections, as described in (Stachenfeld et al., 2022). In contrast to the Lorenz 63 model, we do not involve the time step Δt directly in the computation of the update, instead we use $\tilde{\boldsymbol{u}}_{k+1} = \mathcal{S}_\theta(\boldsymbol{u}_k)$. The architecture consists of an encoder convolutional module, four dilated convolutional blocks, and a decoder convolutional module. There exists a residual connection from the encoder to the output of the first dilated convolution block and from the input to the decoder output. The intermediate representations have 48 channels. The encoder, decoder, and intermediate dilated convolutions use kernels of width 5. Within each dilated convolution block, there are four dilated convolutional layers followed by ReLU non-linear activations, with dilation factor increasing by a multiple of 2 for each layer. Each block has a residual connection to the previous one. The model has a total of 324,433 parameters. The model was trained using the ADAM optimizer and an initial learning rate of $5\mathrm{e}^{-4}$. A staircase exponential decay learning rate scheduler was used with a decay factor of 0.5 and decay transitions every 60,000 steps.

Models were trained for 300,000 steps with the rollout increased every 60,000 steps, and hence by the end of training $\mathcal{S}_\theta$ is predicting trajectories of length 5. For both the Curr and Pfwd objectives we use the same rollout schedule and rollout loss weight: $\omega(k) = \max(0.9^{k-1}, 1\mathrm{e}^{-3})$. The MMD bandwidth values used were $\boldsymbol{\sigma} = \{0.2, 0.5, 0.9, 1.3\}$.

E.3. Kolmogorov Flow

We also consider the Navier-Stokes equation with Kolmogorov forcing given by

$$\frac{\partial \boldsymbol{u}}{\partial t} = -\nabla \cdot (\boldsymbol{u} \otimes \boldsymbol{u}) + \nu \nabla^2 - \frac{1}{\rho} \nabla p + \mathbf{f} \qquad \text{in } \Omega, \tag{35}$$

$$\nabla \cdot \boldsymbol{u} = 0 \qquad \text{in } \Omega, \tag{36}$$

where $\Omega = [0, 2\pi]^2$, $\boldsymbol{u}(x, y) = (\boldsymbol{u}_x, \boldsymbol{u}_y)$ is the field, ρ is the density, p is the pressure, and $\mathbf{f}$ is the forcing term given by

$$\mathbf{f} = \begin{pmatrix} 0 \\ \sin(k_0 y) \end{pmatrix} + 0.1\boldsymbol{u}, \tag{37}$$

where $k_0 = 4$. The forcing only acts in the y coordinate. Following Kochkov et al. (2021), we add a small drag term to dissipate energy. An equivalent problem is given by its vorticity formulation

$$\partial_t \omega = -\boldsymbol{u} \cdot \nabla \omega + \nu \nabla^2 \omega - \alpha\, \omega + f, \tag{38}$$

where $\omega := \partial_x \boldsymbol{u}_y - \partial_y \boldsymbol{u}_x$, which we use for spectral method which avoids the need to separately enforce the incompressibility condition $\nabla \cdot \boldsymbol{v} = 0$. The initial conditions are the same as the ones proposed in Kochkov et al. (2021).

Pseudo-Spectral Discretization Equations 33 and 38 were discretized using a pseudo-spectral discretization, to avoid issues stemming from dispersion errors. Pseudo-spectral methods are known to be dispersion free, due to the *exact* evaluation of the derivatives in Fourier space, while possessing excellent approximation guarantees (Trefethen, 2000). We use the `jax-cfd` spectral elements tool box (Dresdner et al., 2022). Learning to correct spectral methods for simulating turbulent flows, which leverages the Fast Fourier Transform (Cooley & Tukey, 1965) to compute the Fourier transform in space of the field $\boldsymbol{u}(x, t)$, denoted by $\hat{\boldsymbol{u}}(t)$, allows for a very efficient computation of spatial derivatives by diagonal rescaling following $\partial_x \hat{\boldsymbol{u}}_K = iK\hat{\boldsymbol{u}}_K$, where K is the wave number. This renders the application and inversion of linear differential operators trivial, since they are simply element-wise operations (Trefethen, 2000).

This procedure transforms Equation 33 and Equation 35 to a system in Fourier domain of the form

$$\partial_t \hat{\boldsymbol{u}}(t) = \mathbf{D}\hat{\boldsymbol{u}}(t) + \mathbf{N}(\hat{\boldsymbol{u}}(t)), \tag{39}$$

where $\mathbf{D}$ denotes the linear differential operators in the Fourier domain and is often a diagonal matrix whose entries only depend on the wave number K and $\mathbf{N}$ denotes the nonlinear part. These non-linear terms are computed in real space.

Equation 38 was discretized with spatial discretization $n_x = n_y = 256$ and a 4th order implicit-explicit Crack-Nicolson Runge-Kutta scheme (Canuto et al., 2007), where we treat the linear part implicitly and the nonlinear one explicitly with $\Delta t = 0.001$ using `jax-cfd`.

For each trajectory, we let the solver "warm up" for 50 units of time, in order for the trajectory to reach the attractor. We further evolve the equation for 120 units of time, and we sample the trajectories at a rate of $\Delta t = 0.1$. Finally, we downsample the trajectories by a factor 4 in each spatial direction. We repeated the process 128 times to obtain the training data, and 32 times for both the validation and test data.

E.3.1. KOLMOGOROV FLOW EXPERIMENTS FOR EACH OBJECTIVE.

We set the learning rate to be $5\mathrm{e}^{-4}$, and we vary the batch size from 32 to 512 (depending of the experiment) in increments of power to two. We use an exponential learning rate scheduler, which halves the learning rate every 72,000 iterations. We trained the models for up to 720,000 iterations. Unless otherwise stated the MMD bandwidth used was $\boldsymbol{\sigma} = \{2, 5, 9, 13, 20, 50, 90, 120\}$. This value was found after a quick hyperparameter tuning on a small dataset.

We parametrize $\mathcal{S}_\theta$ using a two-dimensional dilated convolutional neural network with residual connections and periodic boundary conditions following (Stachenfeld et al., 2022) the total number of parameters is 6,458,785. We follow the same unrolling scheme as in the KS system, i.e., $\tilde{\boldsymbol{u}}_{k+1} = \mathcal{S}(\boldsymbol{u}_k)$.

One-step For this objective, we use the same set up as experiments above. We halved the learning rate every 72,000 iterations and the models were trained for 720,000 iterations while keeping ℓ constant and equal to one.

Pushforward For Pfwd training, we consider a rollout schedule that follows the learning rate schedule: every 72,000 iterations we increase the number of unrolling steps ℓ by one, where ℓ increases from 1 to 10. The effective number of unrolling step at each training step is sampled uniformly from 1 to ℓ.

Curriculum Given the higher memory requirement, we decrease the number of maximum unrolling steps from 10 to just 5. Also, depending on the batch size, we further decrease the maximum number of unrolling steps. In particular, for large batch sizes, we cannot afford unrolling more than 2 time steps. All the other parameters were kept constant relative to Pushfoward experiments.

F. Further Results

F.1. Lorenz 63

Figure 4 depicts the attractor for the Lorenz 63 system for an ensemble of trajectories (with randomly chosen initial conditions close to the attractor) at time $t = 400$. We observe that DySLIM regularization provides some extra symmetry to the attractor, as evidenced by a better defined right-wing.

In addition, we provide several other metrics to showcase the advantage of our methodology. Figure 5 shows the behavior of the MMD metric for much longer horizons to the ones used for training. We can observe that DySLIM outperforms all the baselines. We also, studied the Wasserstein-1 distance of different features involved in the ground truth dynamics shown in Equation 32. In this case, Figure 6 shows that the distribution of each of the components is better captured in the models trained with DySLIM.

F.2. Kuramoto-Sivashinsky

In addition to the results shown in the main text, Figure 7 shows the improved stability from DySLIM by comparing the distribution of first order ($\boldsymbol{u}_x$) and second order ($\boldsymbol{u}_{xx}$) spatial derivatives for ground truth and predicted trajectories, which are relevant quantities that appear in the PDE that defines this system in Equation 33. We use finite difference methods to

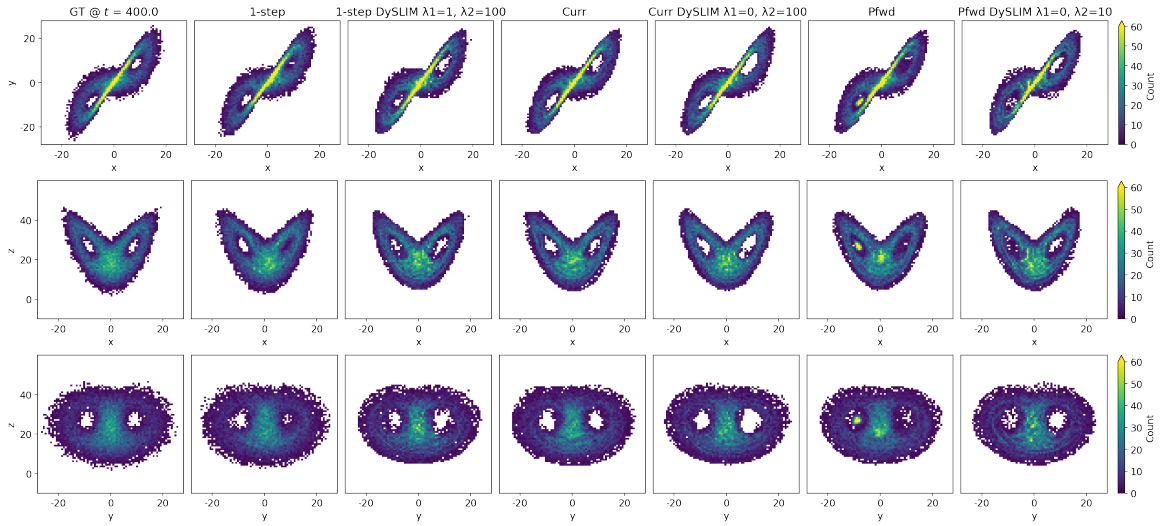


Figure 4. Histograms of trajectories at rollout time $t = 400$ for one of the random training seeds. We showcase the well known "butterfly" attractor.

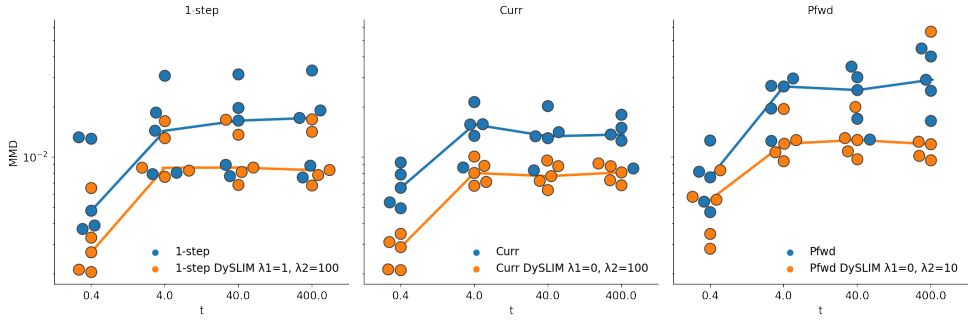


Figure 5. Values of the MMD metric for the baselines and the DySLIM regularization. Each point represents a random training seed that remains stable, with the solid line indicating median values.

compute $\boldsymbol{u}_x$ and $\boldsymbol{u}_{xx}$ at each point in time for each trajectory in the test set. At each point in time, we thus have a distribution for these derivatives across test set trajectories and spatial grid. We use the Wasserstein-1 distance to compare ground truth and predicted distributions and find that models trained with regularized objectives better match the distribution of ground truth spatial derivatives. In summary, Figure 7 shows that by regularizing the loss, we obtain a closer distribution on the derivatives than when using the unregularized loss. In fact, for some cases of the curriculum training, the Wasserstein-1 matrix explodes as the trajectories are highly unstable.

F.3. Kolmogorov Flow

In this section, we provide ablation results and additional trajectory samples.

Table 5 compiles additional results for the model trained with and without regularization following the description in Appendix E.3.1. This table shows that using DySLIM either improves or roughly maintains values for all metrics. We point out that for curriculum training, the error tends to increase with batch size due to the lower number of rollout steps during training, which is a direct consequence of the higher memory footprint required for curriculum training.

Figure 9 provides additional samples of the trajectories presented in Figure 3 for the training with the pushforward objective. From Figure 9, we observe that models trained with the unregularized objectives remain highly dissipative despite using different random seeds. For this configuration of parameters, we were able obtain only one stable model trained without the regularization among the random seeds, which we present in Figure 10. In this case, even though the trajectories are visually

Table 5. Metrics for pushforward, curriculum and 1-step baselines with ($\lambda_1 = 0$, $\lambda_2 = 100$) and without regularization under various batch sizes and learning rates for the Kolmogorov flow. Boldface numbers indicate that the metric is improved by our regularizations. The best-performing run shown in Table 1 is highlighted in green.

Batch size	Learning rate	MELR (↓) ($\times 10^{-2}$)		MELRw (↓) ($\times 10^{-2}$)		covRMSE (↓) ($\times 10^{-2}$)		Wass1 (↓) ($\times 10^{-2}$)		TCM (↓) ($\times 10^{-2}$)	
		Base	DySLIM	Base	DySLIM	Base	DySLIM	Base	DySLIM	Base	DySLIM
Pushforward											
32	5e-4	17.2	**2.65**	9.87	**0.66**	23.6	**7.41**	69.3	**5.20**	83.5	**3.90**
	1e-4	3.16	**3.13**	0.49	0.77	6.27	7.28	5.96	**5.04**	1.17	3.15
	5e-5	4.37	**4.05**	0.61	0.82	6.74	**6.57**	5.89	**4.99**	2.75	**2.62**
	1e-5	11.2	**9.11**	1.29	**0.99**	9.00	**8.32**	8.33	**7.86**	4.80	**3.98**
64	5e-4	18.3	**2.40**	9.81	**0.66**	22.6	**7.33**	68.6	**4.49**	85.6	**2.07**
	1e-4	3.19	**2.95**	0.54	0.70	6.90	6.91	5.09	5.15	1.62	3.30
	5e-5	4.52	**3.71**	0.71	0.82	7.02	**6.66**	5.81	**5.21**	4.12	4.59
	1e-5	10.6	**8.27**	1.22	**0.91**	7.94	**7.52**	9.44	**6.61**	4.58	**3.75**
128	5e-4	73.2	**2.35**	61.3	**0.60**	80.2	**7.27**	26.3	**5.30**	33.4	**2.28**
	1e-4	3.19	**2.46**	0.53	0.53	6.81	**6.69**	4.64	**4.51**	3.68	**0.72**
	5e-5	4.18	**3.53**	0.55	0.71	6.73	7.55	5.20	5.33	1.69	3.55
	1e-5	9.66	**7.54**	1.09	**0.75**	8.71	**8.10**	8.59	**6.06**	4.59	**3.00**
Curriculum											
32	5e-4	5.14	**2.21**	0.74	0.74	8.12	**7.41**	13.6	**4.42**	4.60	**2.45**
64	5e-4	5.35	**1.64**	0.95	**0.45**	8.13	**6.95**	9.66	**4.76**	3.50	**2.83**
128	5e-4	6.80	**3.06**	1.19	1.47	8.60	**8.13**	70.5	**8.77**	27.4	**3.50**
256	5e-4	42.8	**3.35**	31.3	**1.53**	diverge	**8.91**	89.1	**19.7**	817	**65.0**
512	5e-4	25.8	**4.42**	23.1	**1.78**	diverge	**9.31**	166	**21.4**	819	**7.53**
1-step											
32	5e-4	3.11	**2.05**	0.45	0.98	7.30	8.20	8.17	**4.87**	2.58	3.76
64	5e-4	2.77	**1.84**	0.44	0.85	7.93	**7.30**	16.2	**5.55**	5.39	**2.45**
128	5e-4	1.47	1.80	0.28	0.89	7.03	**7.02**	6.65	**5.92**	2.66	3.82
256	5e-4	30.5	**1.90**	24.7	**0.88**	116	**7.04**	30.4	**18.4**	347	**4.10**

more realistic, if we consider the metrics used for evaluation, Table 7 shows that the models trained with the DySLIM regularization still provide better statistics.

We point out that these highly dissipative models are also present as we increase the batch size. For example, Figures 11 and 12 show the same phenomenon for trajectories learned without regularization for batch sizes of 256 and 512.

In addition, Table 6 shows the statistics of models trained with much longer time-horizons (40 time steps instead of 10) for the Kolmogorov flow. The baseline in this case becomes completely uninformative, whereas the regularized version still provides models with reasonable statistics.

G. Sinkhorn Divergence for Measure Matching

In this section we provide an ablation for using SD in place of MMD as the measure distance regularizer D.

Lorenz 63 Ablation From Figures 13, 14, and 15, we can observe that using SD to regularize the objectives provides some benefits. For the short term behavior in Figure 13, we see that the gains are very similar between using SD or MMD in place of D. For the measure matching metrics, we see that both measure-matching metrics stabilize the trajectories. This can be further be seen in Figures 16 and 17, which show that SD does stabilize some of the summary metrics. We point out

Table 6. Kolmogorov flow: metrics for pushforward training for 40 time steps. ($\lambda_1 = 0$, $\lambda_2 = 100$). Boldface numbers indicate that the metric is improved by our regularization.

	Batch size	Learning rate	MELR (↓) ($\times 10^{-2}$)		MELRw (↓) ($\times 10^{-2}$)		covRMSE (↓) ($\times 10^{-2}$)		Wass1 (↓) ($\times 10^{-2}$)		TCM (↓) ($\times 10^{-2}$)	
			Base	DySLIM	Base	DySLIM	Base	DySLIM	Base	DySLIM	Base	DySLIM
Pushforward	128	5e-4	62.22	**8.51**	24.01	**1.65**	42.6	**8.07**	175	**11.34**	161	**4.3**

Table 7. Metrics of the best model using the unregularized pushforward loss versus the average of the models trained using the regularized pushforward loss, for batch size = 128 and learning rate = 5e-5.

	MELR (↓) ($\times 10^{-2}$)	MELRw (↓) ($\times 10^{-2}$)	covRMSE (↓) ($\times 10^{-2}$)
Pfwd	4.10	0.670	8.10
+DySLIM	**2.34**	**0.588**	**7.89**

that even though the performance is competitive, using MMD still seems to have an edge, albeit fairly small.

Kuramoto–Sivashinsky Ablation In Figure 18, we find evidence that using SD as the regularizer for the KS system does improve stability, but performance with respect to using MMD starts to deteriorate. In Figure 18 (a), we see that there is indeed a better performance when using MMD as opposed to SD. In Figure 18 (b) we find that even though the SD regularization helps, many of the trajectories still diverge particularly for curriculum training.

The conclusion from this ablation is that while SD is compatible with our DySLIM framework and does provide increased stability, we find that using MMD in the regularizer leads to superior performance, especially for systems with larger dimension, e.g., KS.

H. Assets

H.1. Software and Libraries

In Table 8, we list relevant open-source software, and corresponding licenses, used in this work:

Table 8. Open source libraries used in this work, with corresponding licenses.

Library	License
Flax (Heek et al., 2023)	Apache 2.0
Jax (Bradbury et al., 2018)	Apache 2.0
Jax-CFD (Dresdner et al., 2022)	Apache 2.0
NumPy (Harris et al., 2020)	NumPy license
Matplotlib (Hunter, 2007)	Matplotib license
ML Collections	Apache 2.0
Optax	Apache 2.0
Orbax	Apache 2.0
OTT-Jax (Cuturi et al., 2022)	Apache 2.0
Pandas (pandas development team, 2020)	BSD 3-Clause “New” or “Revised”
SciPy (Virtanen et al., 2020)	SciPy license
Seaborn (Waskom, 2021)	BSD 3-Clause “New” or “Revised”
Swirl Dynamics (Wan et al., 2023b)	Apache 2.0
TensorFlow (Abadi et al., 2015)	Apache 2.0
Xarray (Hoyer & Hamman, 2017)	Apache 2.0

Table 9. Computational resources by experiment.

Experiment	Hardware
Lorenz 63	1 V100 GPU, 16 GB
Kuramoto–Sivashinsky	1 V100/A100 GPU, 16/40GB
Kolmogorov Flow	1 A100 GPU, 40GB

H.2. Computational Resources

Experiments were submitted as resource-restricted jobs to a shared compute cluster. Computational resources used in each dynamical system experiment are listed in Table 9.

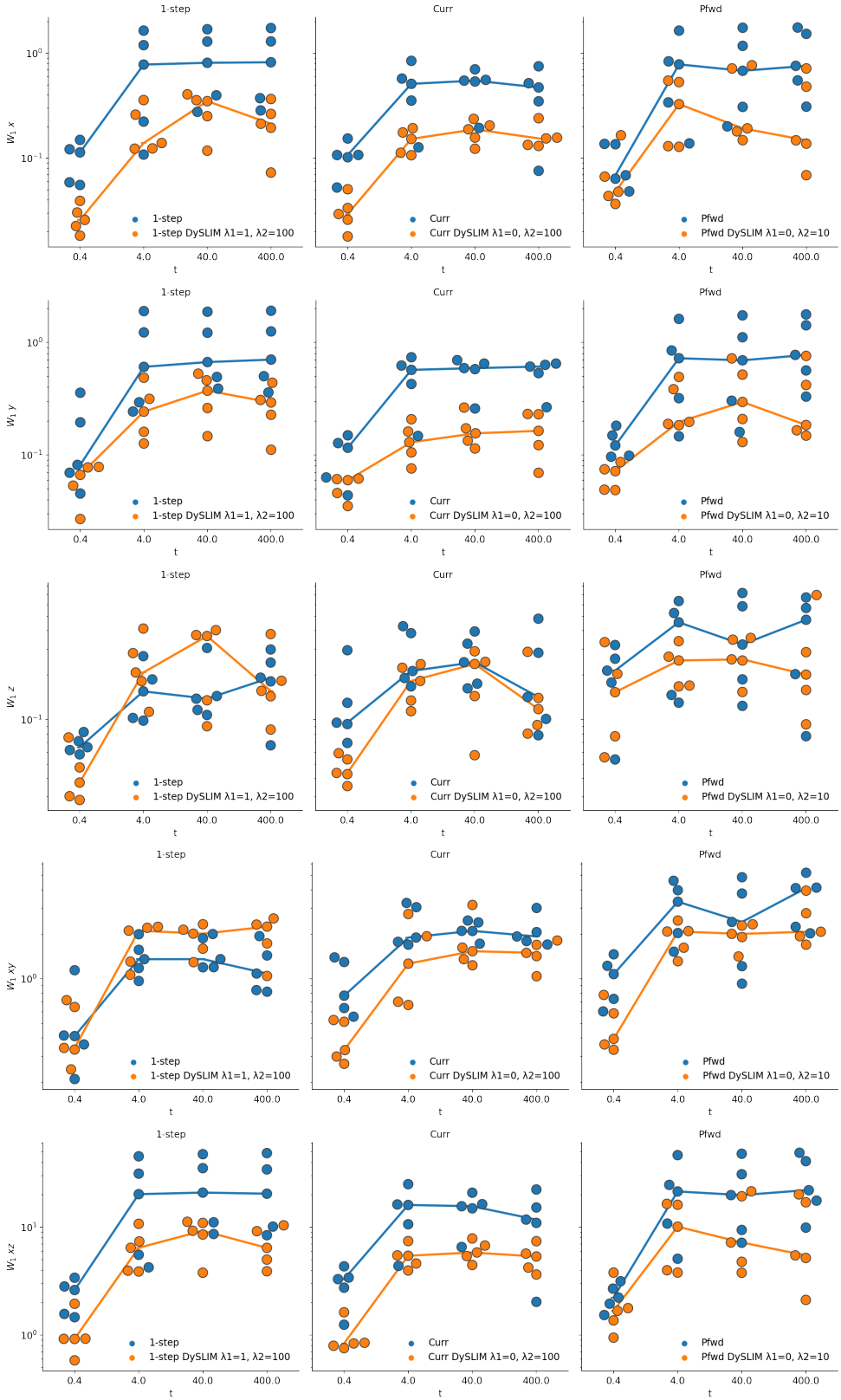


Figure 6. Wasserstein-1 distance (↓) between the different components of the forcing in Equation 32 at different unrolling times. From top to bottom, x, y, z, and the crossed products xy and xz. Each point represents a random training seed that remains stable, with the solid line indicating median values.

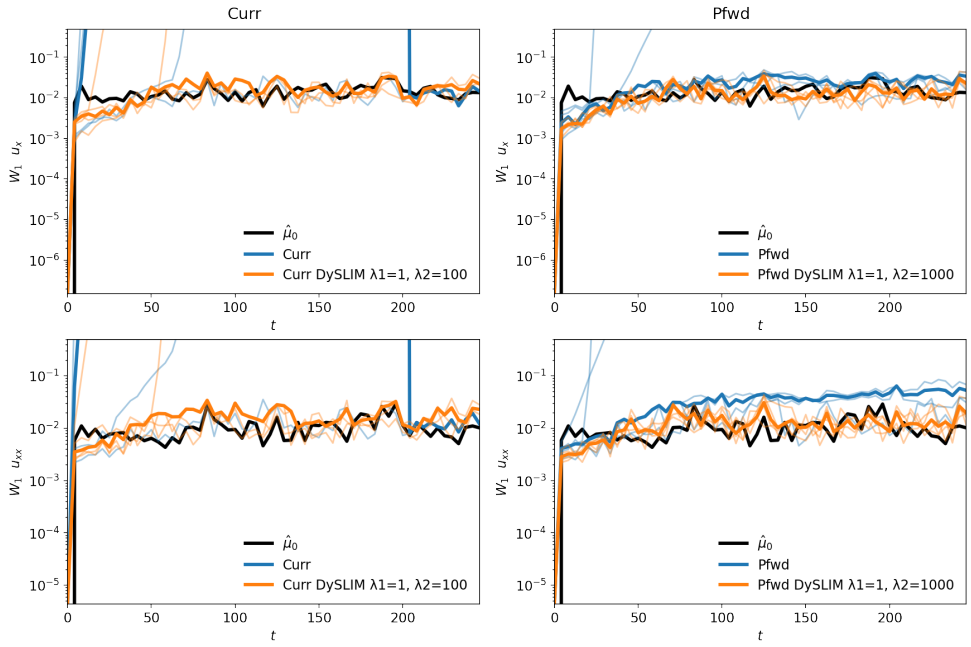


Figure 7. Wasserstein-1 distance (↓) on distribution of first order (*Top*) and second order (*Bottom*) spatial derivatives across time. Values greater than 3 are not shown on the plot. Each line corresponds to one of five random training seeds with bolded lines indicating median values (excluding trajectories that produce `NaN` values). The solid black line corresponds to statistics calculated from the distribution of initial conditions.

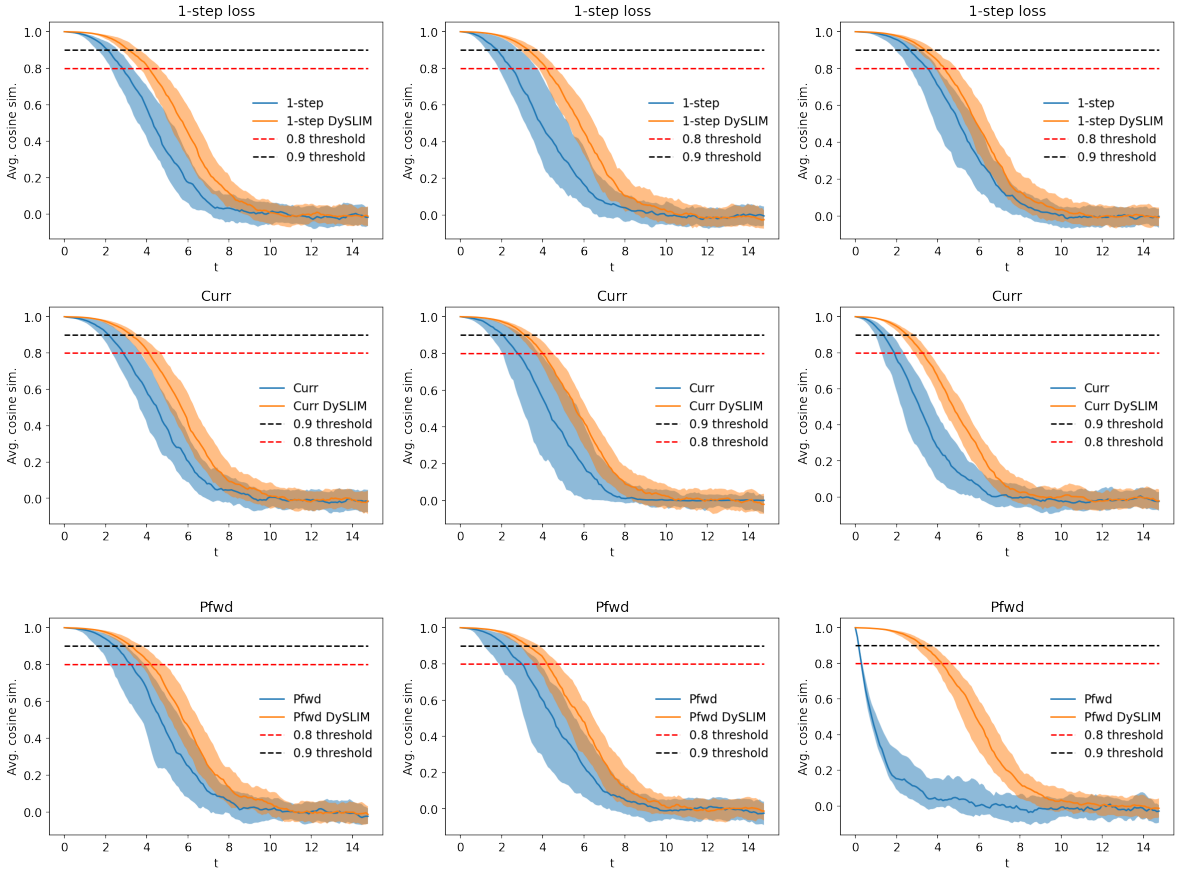


Figure 8. Evolution of the cosine similarity (↑) overt time for trajectories trained with both regularized and unregularized objectives for different batch sizes of (a) 32, (b) 64, and (c) 128. The solid line is the median among 160 runs, and the shaded regions corresponds second and third quartile. ($\lambda_1 = 0$, $\lambda_2 = 100$, batch size = 128 and learning rate= $5\mathrm{e}^{-4}$).

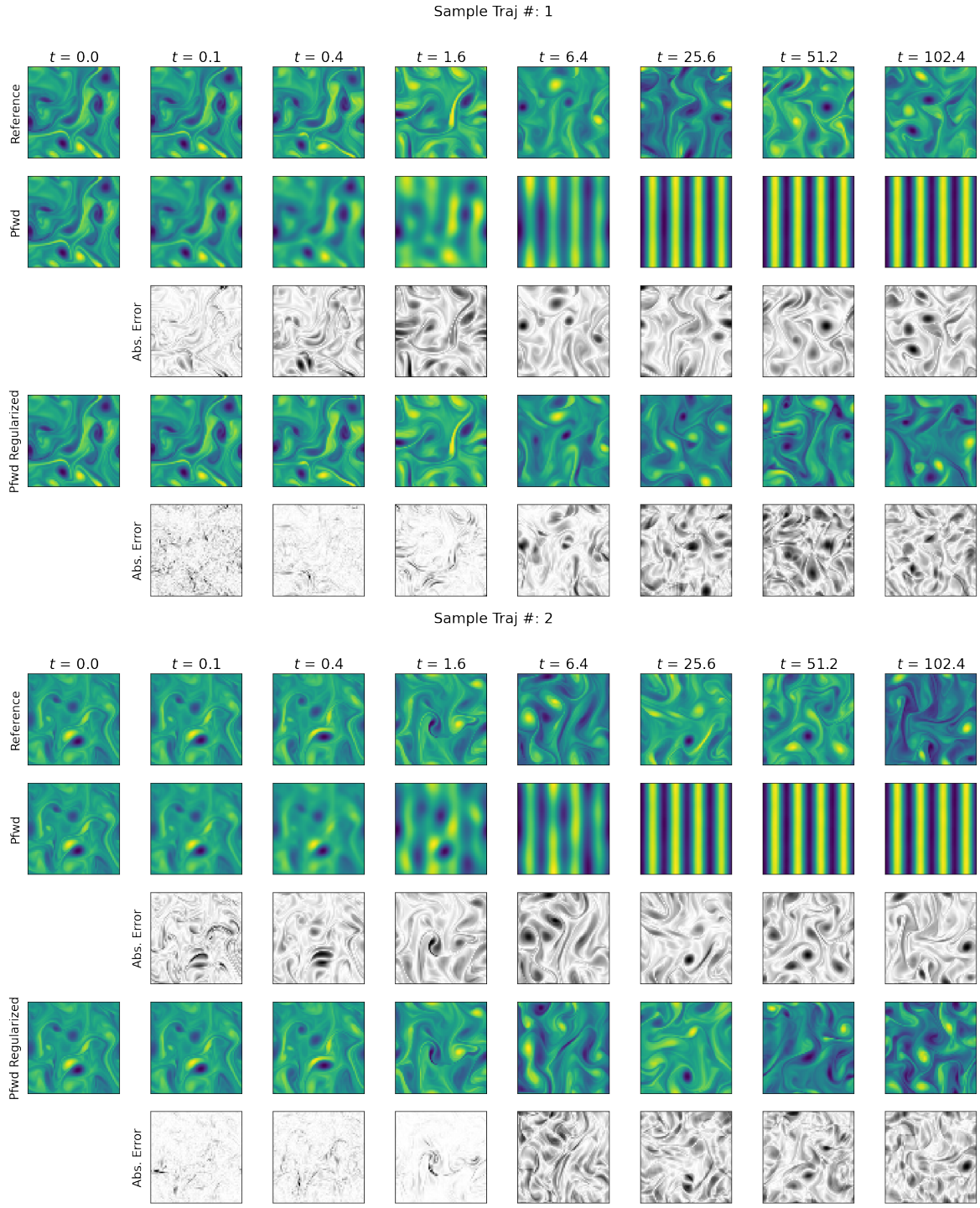


Figure 9. Additional samples of reference trajectories of Navier Stokes system with Kolmogorov forcing and predicted trajectory generated with models trained using the pushforward objective, with and without regularization. ($\lambda_1 = 0$, $\lambda_2 = 100$, batch size = 128 and learning rate= $5\mathrm{e}^{-4}$).

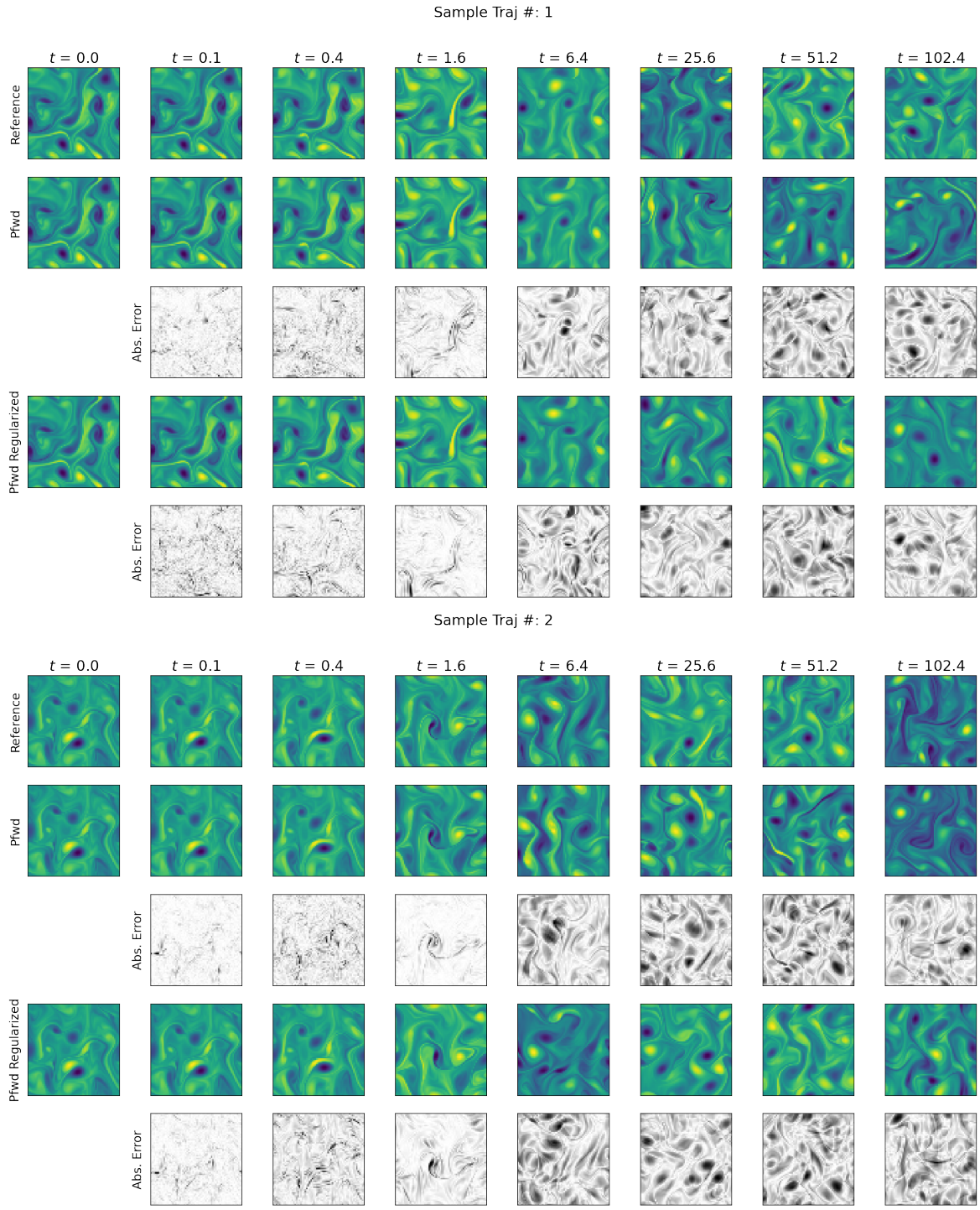


Figure 10. Samples of reference trajectories of Navier Stokes system with Kolmogorov forcing and predicted trajectory generated with models trained the pushforward loss, with and without regularization. In this case, we show samples of the random seed with the best results for the unregularized training. ($\lambda_1 = 0$, $\lambda_2 = 100$, `batch size` = 128 and `learning rate`= 5e-4).

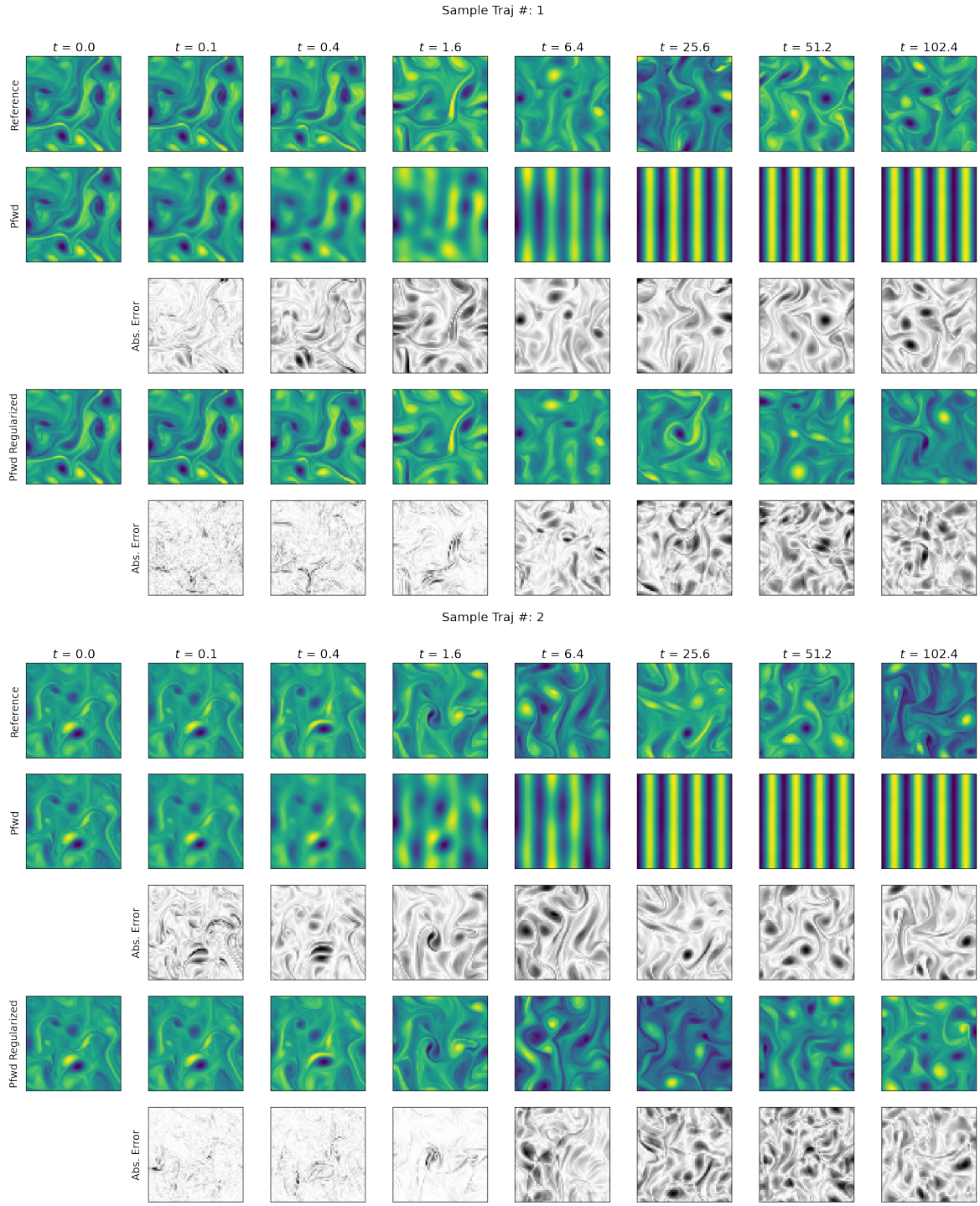


Figure 11. Samples of reference trajectories of Navier Stokes system with Kolmogorov forcing and predicted trajectory generated with models trained the pushforward loss, with and without regularization with a batch size of 256 for the same random seed. ($\lambda_1 = 0$, $\lambda_2 = 100$, and learning rate = $5\mathrm{e}^{-4}$).

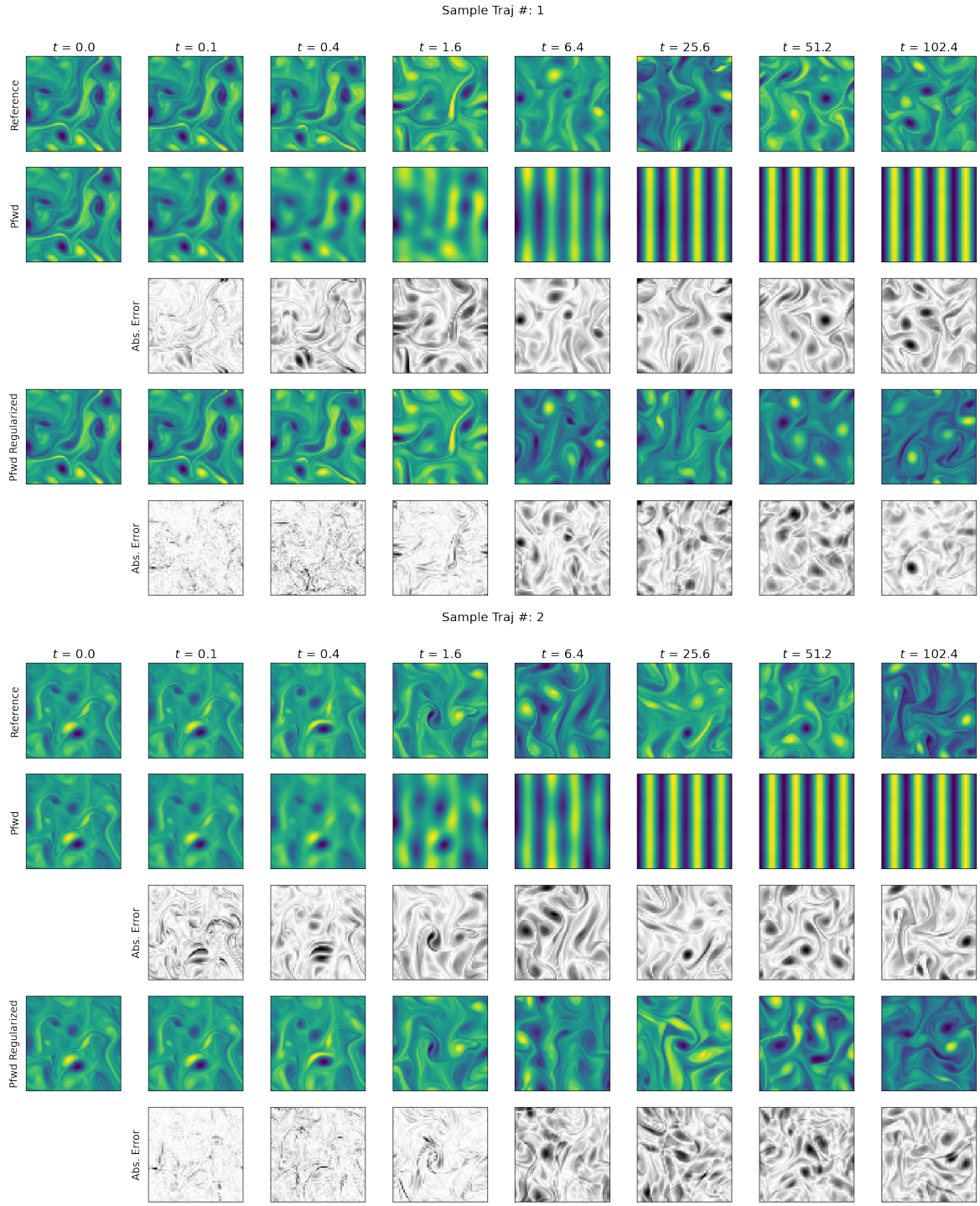


Figure 12. Samples of reference trajectories of Navier Stokes system with Kolmogorov forcing and predicted trajectory generated with models trained the pushforward loss, with and without regularization with a batch size of 512 for the same random seed. ($\lambda_1 = 0$, $\lambda_2 = 100$, and learning rate = $5\mathrm{e}^{-4}$).

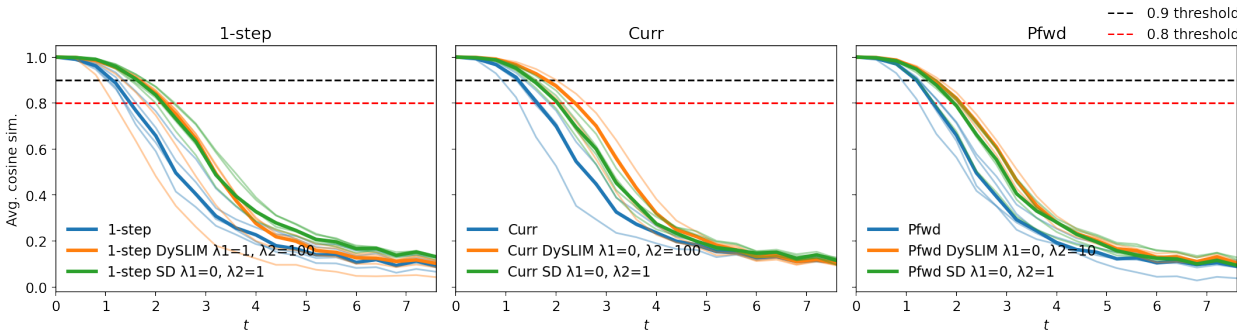


Figure 13. Cosine similarity metric (↑) for test trajectories at various rollout times of the Lorenz 63 system using MMD and SD for measure matching during training. Each line corresponds to the mean over trajectories of each of five random training seeds, with bold lines indicating median values.

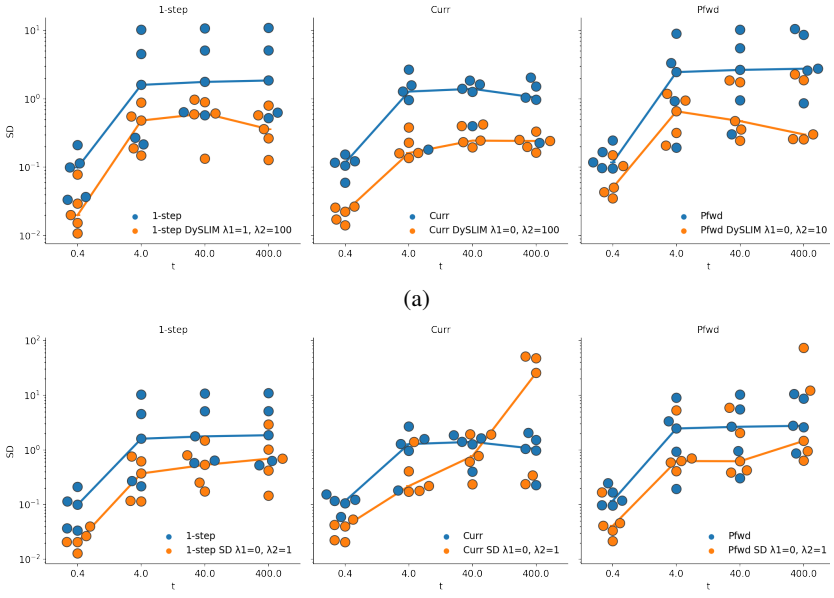
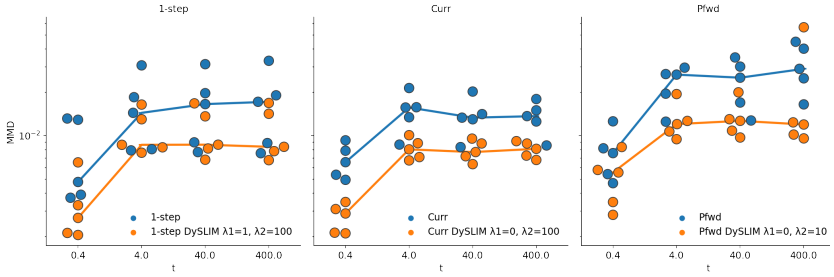
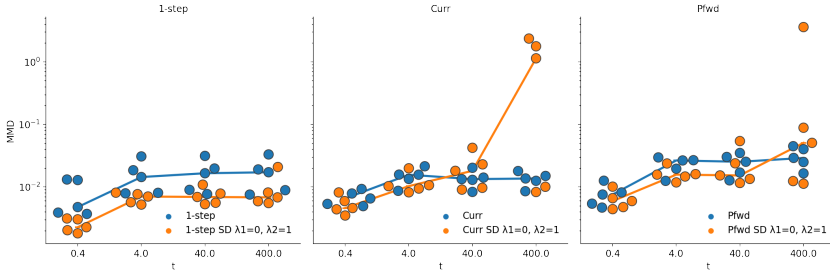


Figure 14. Sinkhorn Divergence (SD; ↓) between trajectories at various rollout times of the Lorenz 63 system. (a) Values when using MMD for measure matching at training. (b) Values when using SD for measure matching at training. Each point represents a random training seed that remains stable, with the solid line indicating median values.

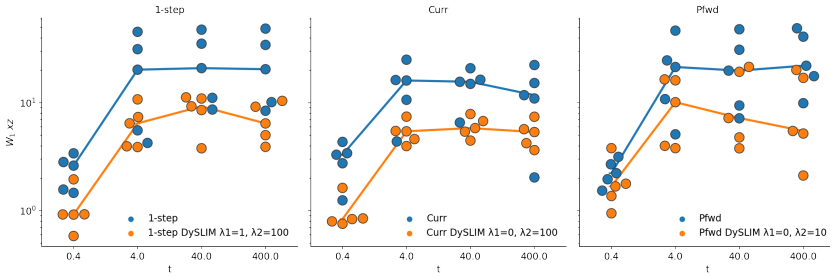


(a)

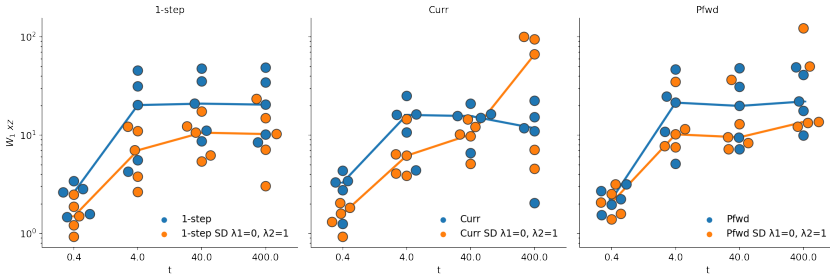


(b)

Figure 15. Maximum Mean Discrepancy (MMD; ↓) between trajectories at various rollout times of the Lorenz 63 system. (a) Values when using MMD for measure matching at training. (b) Values when using SD for measure matching at training. Each point represents a random training seed that remains stable, with the solid line indicating median values.



(a)



(b)

Figure 16. Wasserstein-1 metric (↓) for the xz values between trajectories at various rollout times of the Lorenz 63 system. (a) Values when using MMD for measure matching at training. (b) Values when using SD for measure matching at training. Each point represents a random training seed that remains stable, with the solid line indicating median values.

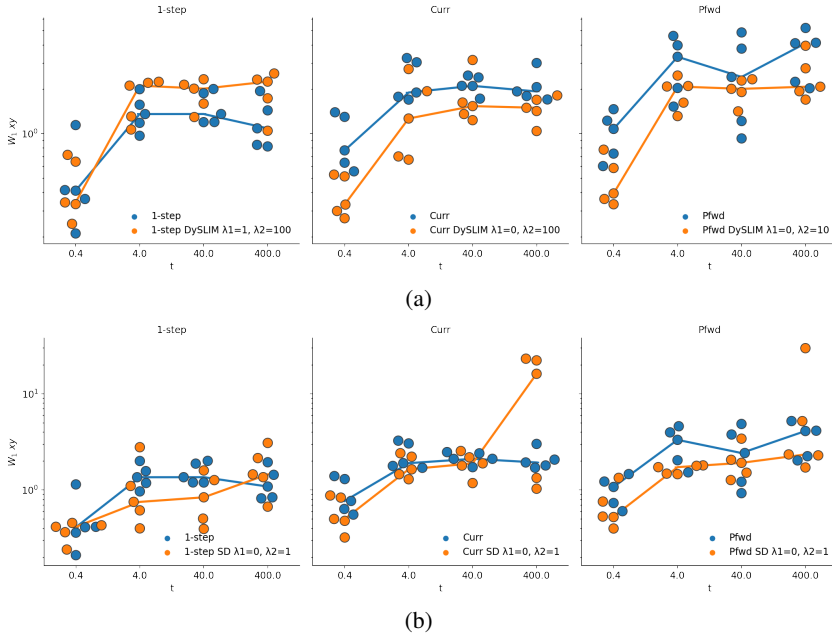


Figure 17. Wasserstein-1 metric (↓) for the xy coordinates between trajectories at various rollout times of the Lorenz 63 system. (a) Values when using MMD for measure matching at training. (b) Values when using SD for measure matching at training. Each point represents a random training seed that remains stable, with the solid line indicating median values.

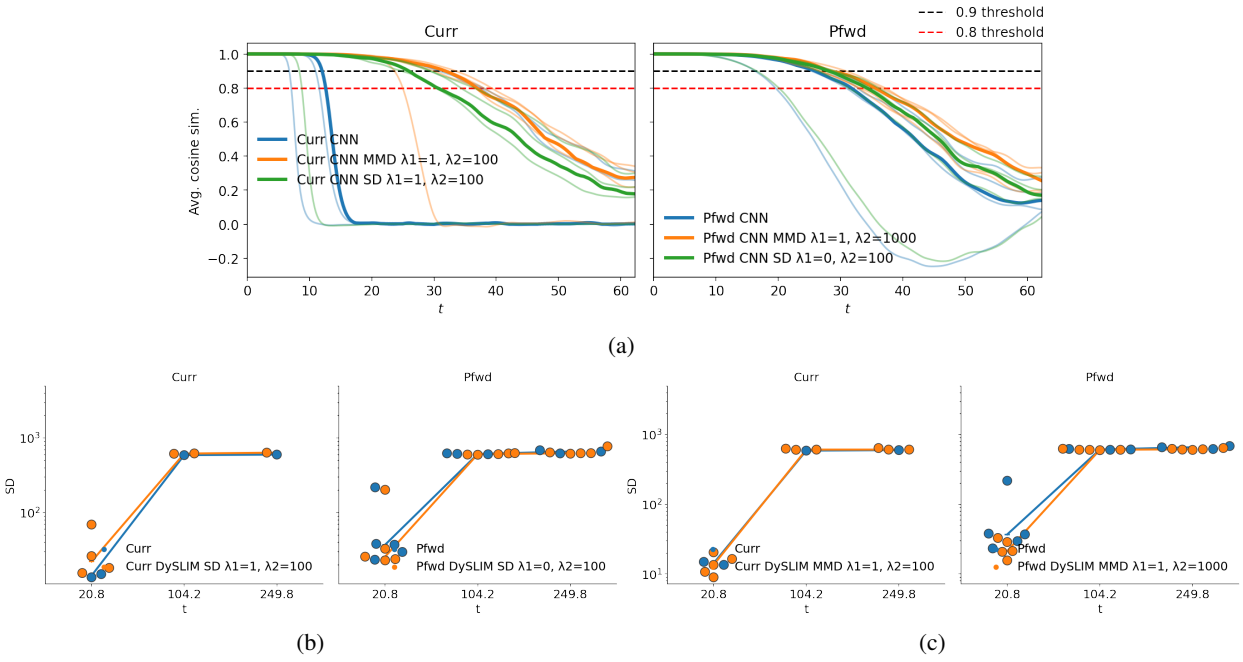


Figure 18. Metrics for KS system. (a) Cosine Similarity (↑) between the different trajectories. Each line corresponds to the mean over trajectories of each of five random training seeds, with bold lines indicating median values. (b) Sinkhorn Divergence (SD; ↓) between trajectories at various rollout times when using MMD for measure matching at training. (c) SD between trajectories at various rollout times when using SD for measure matching at training. Each point represents a random training seed that remains stable, with the solid line indicating median values.