Efficient probabilistic surrogate modeling techniques for partially-observed large-scale dynamical systems

Hans Harder HANS.HARDER@UNI-PADERBORN.DE
University of Paderborn & Lamarr Institute for Machine Learning and Artificial Intelligence

Abhijeet Vishwasrao ABVISH@UMICH.EDU
University of Michigan

Luca Guastoni LUCA.GUASTONI@TUM.DE
Technical University of Munich

Ricardo Vinuesa RVINUESA@UMICH.EDU
University of Michigan

Sebastian Peitz SEBASTIAN.PEITZ@TU-DORTMUND.DE
Technical University of Dortmund & Lamarr Institute for Machine Learning and Artificial Intelligence

Editors: G. Sukhatme, L. Lindemann, S. Tu, A. Wierman, N. Atanasov

Abstract

This paper is concerned with probabilistic techniques for forecasting dynamical systems described by partial differential equations (such as, for example, the Navier–Stokes equations). In particular, it is investigating and comparing various extensions to the flow matching paradigm that reduce the number of sampling steps. In this regard, it compares direct distillation, progressive distillation, adversarial diffusion distillation, Wasserstein GANs and rectified flows. Moreover, experiments are conducted on a set of challenging systems. In particular, we also address the challenge of directly predicting 2D slices of large-scale 3D simulations, paving the way for efficient inflow generation for solvers.

Keywords: Flow matching, progressive distillation, adversarial diffusion distillation, Wasserstein GAN, dynamical systems, partial observations, surrogate modeling, partial differential equations, Navier-Stokes equations, Rayleigh-Taylor instability.

1. Introduction

Partial differential equations (PDEs) are used to describe physical phenomena such as heat transfer, fluid flows, or wave propagation in different media. Solving these equations requires high-resolution spatial discretization and accurate time stepping schemes, leading to high computational cost. This hinders the availability of fast forecasting models based on classical numerical approaches. To address these shortcomings, approximate solutions with data-driven surrogate models have been developed, for example in the case of meteorology (Lam et al., 2023; Price et al., 2025), climate (Watt-Meyer et al., 2023) or ocean dynamics (Chattopadhyay et al., 2024). These models are typically trained in a supervised fashion to match the output of the numerical solver or real data. They are defined on the same or a downscaled version of the solver's spatial mesh (even though there are also mesh-agnostic methods, *e.g.*, Li et al., 2020, 2021) and usually predict larger time steps.

Learning approximate solutions becomes a necessity when the target dynamical system is only partially observable or when the mathematical description of the problem does not perfectly capture the underlying physical system (*e.g.*, when a model for weather forecasting ignores local topology).

While the latter problem can be addressed by conditioning on real data, the former appears more challenging. One way to approach this issue is to include information about the dynamics from past events using, for instance, recurrent network architectures (*e.g.,* Srinivasan et al., 2019; Vlachas et al., 2018; Fromme et al., 2025), motivated in part by Taken's embedding theorem (Takens, 1980). But these models are typically hard to train, require many training samples and access to complete trajectories.

Alternatively, it is possible to lean into the lack of information by learning a probabilistic model instead: With the recent advent of denoising diffusion (Sohl-Dickstein et al., 2015; Ho et al., 2020) and flow matching (Lipman et al., 2023), high-quality generative models have become widely accessible and easy to train. However, using these models comes with a caveat: To generate a new sample, one needs to solve an ordinary differential equation (ODE), requiring multiple evaluations of the trained model. This is a relatively general issue, so there have been efforts to improve the sampling speed. These approaches, further detailed in Section 2.3, range from training a new model to learn the ODE's flow via "distillation" to methods that straighten the underlying velocity field and, even more recently, adversarial approaches that introduce discriminator networks into training.

Our main goal is to demonstrate that flow matching together with these extensions poses a serious alternative to deterministic surrogate modeling techniques. Insofar, we want to show that

- these methods generate trajectories that correctly approximate the real solutions, are temporally coherent, and closely match the reference as the setting becomes more deterministic,
- further distillation/rectification speeds up the sampling process, allowing single-step sampling in some cases, while being superior in physical accuracy to GANs,
- these methods open up entirely new possibilities for prediction or re-initialization of expensive solvers. We demonstrate this with experiments on the compressible Navier-Stokes (NS) equations and the Rayleigh-Taylor instability (RTI) in two and three dimensions.

The code for this paper is publicly available under github.com/graps1/flow-matching-for-time-series.

1.1. Related work and discussion

Denoising diffusion, score matching and flow matching progressively denoise samples from an initial Gaussian noise distribution. These methods were found to be largely equivalent (Holderrieth et al., 2025). Diffusion probabilistic models were developed by Sohl-Dickstein et al. (2015), which were popularized and found to be equivalent to score matching by Song and Ermon (2019); Ho et al. (2020); Dhariwal and Nichol (2021); Nichol and Dhariwal (2021). Denoising diffusion "implicit" models (Song et al., 2021) made the first step towards a deterministic sampling process (except for the initial noise sample) by formulating the solution as an ordinary instead of a stochastic differential equation. In the same spirit, flow matching (Lipman et al., 2023, 2024) popularized the "Gaussian optimal transport path", an advantageous coupling of noise and target distribution.

In these years, a lot of progress has been made in speeding up the sampling process: First the transition from SDEs to ODEs, then the popularization of better transport couplings. However, there is still room for improvement when comparing the sampling speed to that of GANs. Insofar, the recent literature provides us with a few approaches: Distillation-based methods such as consistency distillation (Song et al., 2023; Song and Dhariwal, 2024) or progressive distillation (Salimans and Ho, 2022) learn to predict the ODE's solution from the initial sample. One can also employ adversarial methods by including a discriminator loss in the distillation (Lin et al., 2024; Sauer et al.,

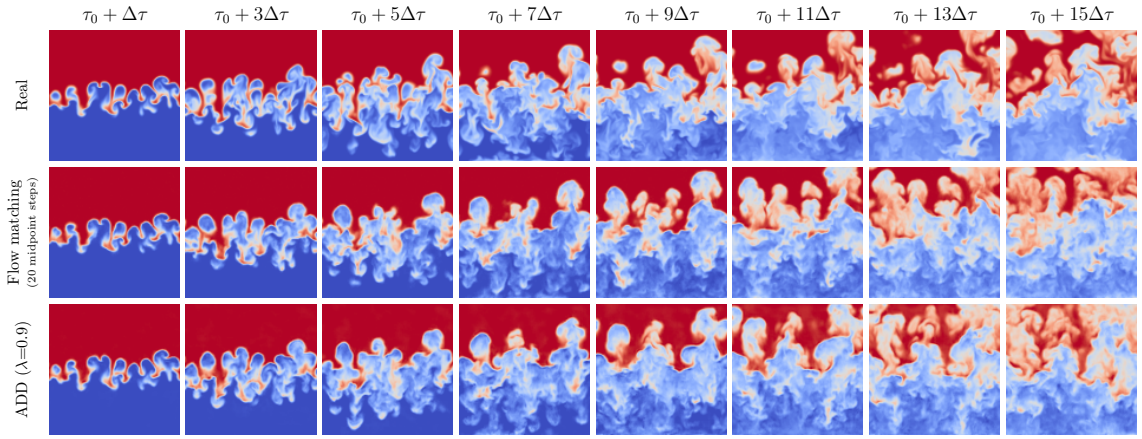


Figure 1: Autoregressively generated trajectories on the "sliced" Rayleigh-Taylor instability dataset, starting with the same initial condition at (simulation) time τ_0.

2023) or diffusion (Xiao et al., 2022) procedure. Finally, rectified flows (Liu et al., 2023, 2024) "straighten" the underlying flow, allowing larger step sizes when solving the ODE.

Diffusion models have shown great potential for the surrogate modeling of dynamical systems, including turbulent flows (Guastoni and Vinuesa, 2025; Vishwasrao et al., 2025). Most similar to our work are the papers by Li et al. (2025); Luo et al. (2024); Price et al. (2025); Oommen et al. (2025); Kohl et al. (2024) and Shysheya et al. (2024). They all describe methods for forecasting systems autoregressively using conditional diffusion models, either in latent-spaces (Li et al., 2025), in the fully-observed setting Luo et al. (2024); Oommen et al. (2025); Kohl et al. (2024) or for partially observed/real-world data Shysheya et al. (2024); Price et al. (2025). Further work includes the paper by Yang and Sommer (2023), which predicts the system's solution given only the initial condition, and Cachay et al. (2023), which couples the denoising and forecasting processes. The paper of El-Gazzar and van Gerven (2025) is dealing with forecasting ordinary differential equations. It is also worth noting that one can also use a deterministic latent-space model and instead reconstruct the full state using a diffusion model (Gao et al., 2024).

In contrast, our focus is on accelerating the sampling of flow matching-based models.

2. Flow matching

Flow matching (FM, Lipman et al. (2023, 2024)) defines a continuous transformation from samples of a Gaussian distribution $\boldsymbol{x}_0 \sim p_0 = \mathcal{N}(0, I)$ at time $t = 0$ to samples of some target distribution $\boldsymbol{x}_1 \sim p_1$ at time $t = 1$. Both $\boldsymbol{x}_0$ and $\boldsymbol{x}_1$ take values in $\mathbb{R}^n$. The starting point is the definition of an interpolating marginal distribution $p_t(x_t)$, which is classically defined as the the distribution of

$$\boldsymbol{x}_t = t\boldsymbol{x}_1 + (1-t)\boldsymbol{x}_0 =: \psi_t(\boldsymbol{x}_0|\boldsymbol{x}_1), \tag{1}$$

where $\boldsymbol{x}_0$ and $\boldsymbol{x}_1$ are samples from p_0 and p_1. Equation (1) defines the *Gaussian optimal transport path*, which is simply a linear interpolation between source and target samples. The goal is to learn the velocity field $\dot{x}_t = u_t(x_t)$ that pushes $p_t(x_t)$ forward in time: If a particle tracks the velocity field $u_t(\boldsymbol{x}_t)$ such that the initial condition is sampled from $\boldsymbol{x}_0 \sim p_0$, then $\boldsymbol{x}_t$ tracks p_t in distribution

until it finally reaches the target distribution at $t = 1$. Remarkably, if $v_t^\theta(x_t)$ ($v^\theta : [0,1] \times \mathbb{R}^n \to \mathbb{R}^n$) is a neural network parametrized by θ, one can fit it to $u_t(x_t)$ by solving the optimization problem

$$\min_\theta \mathbb{E}_{p_1(\boldsymbol{x}_1),p_0(\boldsymbol{x}_0),U(\boldsymbol{t};0,1)} \|\underbrace{v_{\boldsymbol{t}}^\theta(\boldsymbol{x_t}) - \dot{\boldsymbol{x}}_{\boldsymbol{t}}}_{=v_{\boldsymbol{t}}^\theta(\boldsymbol{t}\boldsymbol{x}_1+(1-\boldsymbol{t})\boldsymbol{x}_0)-(\boldsymbol{x}_1-\boldsymbol{x}_0)}\|^2, \quad \text{where} \quad \boldsymbol{x_t} = \psi_{\boldsymbol{t}}(\boldsymbol{x}_0|\boldsymbol{x}_1). \tag{2}$$

Approximate solutions can be obtained by discretizing in time, for instance by using the explicit Euler method. After training, the inference process consists in sampling from the initial distribution $\boldsymbol{x}_0 \sim p_0$ and then solving for $\boldsymbol{x}_1 = \phi_1^\theta(\boldsymbol{x}_0)$, where $\phi_t^\theta(x_0)$ is the solution of the learned ODE at time t, i.e., $\phi_t^\theta(x_0) = x_t$ if $\dot{x}_t = v_t^\theta(x_t)$.

When dealing with a *conditional target distribution* $\boldsymbol{x}_1 \sim p_1(\cdot|y)$, the training procedure changes minimally. One adds y as an additional input to the velocity network, which becomes $v_t^\theta(x_t|y)$, and one assumes a joint sampling distribution $(\boldsymbol{x}_1, \boldsymbol{y}) \sim p_1(\cdot,\cdot)$. The problem one solves is then

$$\min_\theta \mathbb{E}_{p_0(\boldsymbol{x}_0),p_1(\boldsymbol{x}_1,\boldsymbol{y}),U(\boldsymbol{t};0,1)} \|v_{\boldsymbol{t}}^\theta(\boldsymbol{x_t}|\boldsymbol{y}) - \dot{\boldsymbol{x}}_{\boldsymbol{t}}\|^2, \quad \text{where} \quad \boldsymbol{x_t} = \psi_{\boldsymbol{t}}(\boldsymbol{x}_0|\boldsymbol{x}_1). \tag{3}$$

As for the unconditional case, we denote solutions by $\phi_t^\theta(x_0|y) = x_t$ if x_t tracks the conditional flow matching ODE, i.e., $\dot{x}_t = v_t^\theta(x_t|y)$.

2.1. Flow matching for dynamical systems

We model transitions between partial observations as a Markov chain

$$\boldsymbol{y}_{k+1} \sim p_1(\cdot|\boldsymbol{y}_k), \quad k = 0, 1, 2, \ldots \tag{4}$$

with $\boldsymbol{y}_0$ stemming from some initial distribution, dependent on the problem at hand. The density $p_1(\cdot|y)$ for a given y is the conditional target distribution of a flow matching model. In other words, we are learning a conditional flow matching velocity model $v_t^\theta(x_t|y)$ such that transitioning via

$$\boldsymbol{y}_{k+1} = \phi_1^\theta(\boldsymbol{x}_0^k|\boldsymbol{y}_k), \quad \boldsymbol{x}_0^k \sim p_0 = \mathcal{N}(0, I) \tag{5}$$

yields approximately the same distribution as (4).

2.2. Deterministic models & the initial flow matching direction

Before discussing any extensions to flow matching, we want to argue why it is a good fit for our applications. In particular, we want to show that it becomes more efficient as the underlying distribution becomes more concentrated, i.e., as transitions become more deterministic, in the sense that a single Euler step resembles the output of a deterministic surrogate model.

Suppose therefore we were to train a deterministic predictor $w^\theta : \mathbb{R}^n \to \mathbb{R}^n$ by minimizing a least-squares problem

$$\mathbb{E}_{p_1(\boldsymbol{y}_{k+1},\boldsymbol{y}_k)} \|w^\theta(\boldsymbol{y}_k) - \boldsymbol{y}_{k+1}\|^2. \tag{6}$$

It turns out that (6) is up to an additive constant identical to the objective

$$\mathbb{E}_{p_1(\boldsymbol{y}_k)} \|w^\theta(\boldsymbol{y}_k) - \mathbb{E}_{p_1(\boldsymbol{y}_{k+1}|\boldsymbol{y}_k)}[\boldsymbol{y}_{k+1}]\|^2, \tag{7}$$

see Remark 1 in the appendix for further discussion. In other words: A deterministic model that is trained by minimizing (6) is implicitly fitted to the expected value of $\boldsymbol{y}_{k+1}$ (given $\boldsymbol{y}_k$). Since the

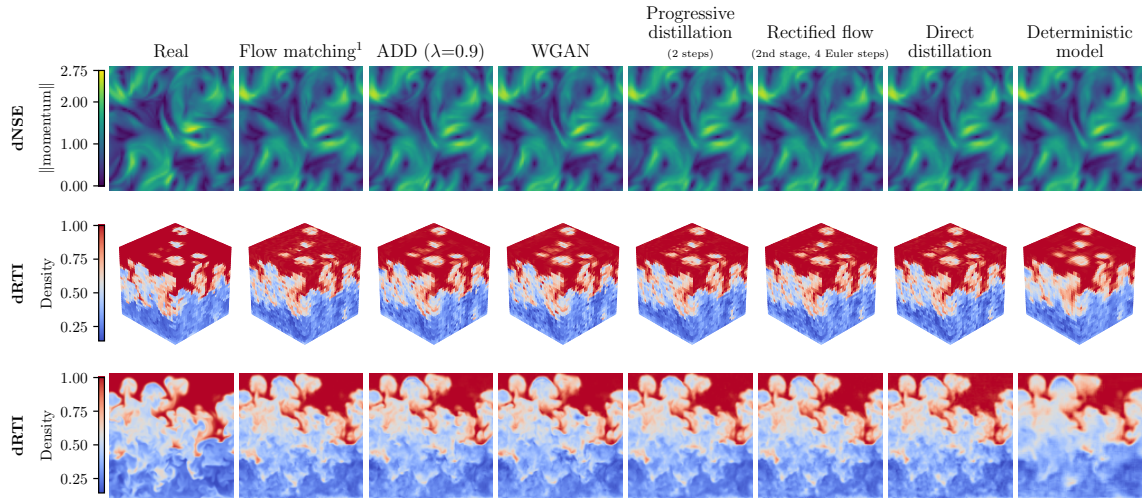


Figure 2: Results after a single prediction step. [1]FM solved with 5 midpoint steps on the **dNSE** and **dRTI** datasets, and with 20 midpoint steps on the **sRTI** dataset.

expected value is a convex combination of all possible $\boldsymbol{y}_{k+1}$, it tends to be relatively smooth, a property that we also observe in our experiments (e.g., Figure 2). The target (7) can now be connected to the initial direction provided by a flow matching model. Starting with the flow matching objective (3), let us fix $\boldsymbol{x}_1 := \boldsymbol{y}_{k+1}$, $\boldsymbol{y} := \boldsymbol{y}_k$ and consider the objective at time $\boldsymbol{t} = 0$,

$$\mathbb{E}_{p_1(\boldsymbol{y}_{k+1},\boldsymbol{y}_k),p_0(\boldsymbol{x}_0)}\|v_0^\theta(\boldsymbol{x}_0|\boldsymbol{y}_k) - (\boldsymbol{y}_{k+1} - \boldsymbol{x}_0)\|^2, \tag{8}$$

which has the same structure as (6). Similarly, it is up to a constant identical to the objective

$$\mathbb{E}_{p_1(\boldsymbol{y}_k),p_0(\boldsymbol{x}_0)}\|v_0^\theta(\boldsymbol{x}_0|\boldsymbol{y}_k) - (\mathbb{E}_{p_1(\boldsymbol{y}_{k+1}|\boldsymbol{y}_k)}[\boldsymbol{y}_{k+1}] - \boldsymbol{x}_0)\|^2. \tag{9}$$

This loss is zero when

$$x_0 + v_0^\theta(x_0|y_k) = \mathbb{E}_{p_1(\boldsymbol{y}_{k+1}|y_k)}[\boldsymbol{y}_{k+1}] \quad \text{for almost all } (x_0, y_k). \tag{10}$$

Crucially, the expression on the left-hand side of (10) is also computed when solving the flow matching ODE with a single explicit Euler step. This shows that a well-trained flow matching model has the capacity to "mimic" a deterministic model, making it highly efficient for fully observed systems. Moreover, it shows that flow matching provides a good baseline for further extensions, which are discussed in the next section.

2.3. Improving the sampling efficiency

For brevity, we will only discuss methods for the unconditional case, since their extension to conditional target distributions is straightforward.

Direct distillation learns a new model $G^\xi : \mathbb{R}^n \to \mathbb{R}^n$, parametrized by ξ, that predicts the solution of the flow matching ODE from the inital random sample. By direct distillation we understand methods that directly fit G^ξ to ϕ_1^θ in, for example, a least-squares sense:

$$\min_\xi \mathbb{E}_{p_0(\boldsymbol{x}_0)}\|G^\xi(\boldsymbol{x}_0) - \phi_1^\theta(\boldsymbol{x}_0)\|^2. \tag{11}$$

A reasonable way to define G^ξ for a flow matching model is to take $G^\xi(\boldsymbol{x}_0) = \boldsymbol{x}_0 + v_0^\xi(\boldsymbol{x}_0)$, where ξ is initialized as a copy of θ, resembling an explicit Euler step. Optimizing (11) requires solving the flow matching ODE for every training sample, which can be challenging when many discretization steps are necessary.

Progressive distillation (Salimans and Ho, 2022) addresses this problem by learning the ODE's solution at increasing time steps. It keeps a "student" and an "expert" model, where the former turns into the latter after each training run. The student model learns to predict two consecutive steps of the expert model, say the student learns to make a step of size δ whereas the expert makes two steps of size $\delta/2$. The step sizes are exponentially increasing ($\delta = \ldots, \frac{1}{8}, \frac{1}{4}, \frac{1}{2}, 1$) until they reach a final value of 1. At that point, the student approximately solves the ODE in a single step. Though originally formulated in the DDIM (Song et al. (2021)) context, one can adapt this idea to flow matching by defining the following iteration: Start with some $N > 0$ that is a power of 2, e.g. $N = 32$, corresponding to an initial step size of $1/N$. As N becomes larger, the Euler iteration

$$G_t^{\xi_N}(x_t) := x_t + \tfrac{1}{N} v_t^{\xi_N}(x_t)$$

with $\xi_N = \theta$ tracks the flow more and more accurately. Starting with $m = N$, one can then solve the optimization problem

$$\xi_{m/2} := \operatorname*{argmin}_\xi \mathbb{E}_{p_{\boldsymbol{t}}(\boldsymbol{x_t}), U(\boldsymbol{t}; \{1/k, 2/k, \ldots, 1-2/k\})} \| G_{\boldsymbol{t}}^\xi(\boldsymbol{x_t}) - G_{\boldsymbol{t}}^{\xi_k}(G_{\boldsymbol{t}}^{\xi_k}(\boldsymbol{x_t})) \|^2, \tag{12}$$

halving m with each training run. Upon reaching $m = 1$, the ODE is approximately solved by $G_0^{\xi_1}$. In contrast to direct distillation, this procedure is more efficient, but also biased. With each training run, the model is fitted to targets produced by another model, which can lead to growing inaccuracy.

Adversarial diffusion distillation (ADD, Sauer et al., 2023; Lin et al., 2024) is a distillation method that includes a discriminator loss, trained GAN-style by identifying the difference between fake (produced by the distilled model) and real samples. The discriminator $D^\zeta : \mathbb{R}^n \to \mathbb{R}$ approximates the Wasserstein-1 distance between the fake and the true distribution, substituting the hard Lipschitz constraint by a soft gradient penalty. We use a similar setup as implemented by geometric GANs (Lim and Ye, 2017; Sauer et al., 2023), in which the discriminator aims to minimize

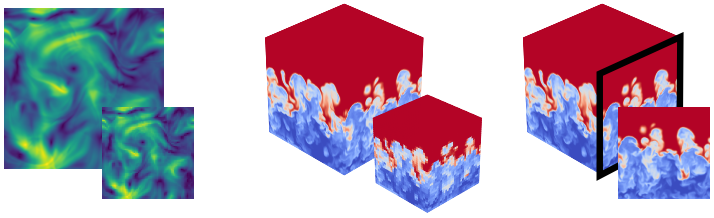
$$\mathbb{E}_{p_0(\boldsymbol{x}_0)}[\max\{0, 1 - D^\zeta(G^\xi(\boldsymbol{x}_0))\}] + \mathbb{E}_{p_1(\boldsymbol{x}_1)}[\max\{0, 1 + D^\zeta(\boldsymbol{x}_1)\} + \gamma \|\nabla D^\zeta(\boldsymbol{x}_1)\|^2], \tag{13}$$

where $\gamma \geq 0$ is the weight of the gradient penalty. The generator minimizes

$$\lambda \mathbb{E}_{p_0(\boldsymbol{x}_0)} \|G^\xi(\boldsymbol{x}_0) - \phi_1^\theta(\boldsymbol{x}_0)\|^2 + (1-\lambda)\mathbb{E}_{p_0(\boldsymbol{x}_0)}[D^\zeta(G^\xi(\boldsymbol{x}_0))], \tag{14}$$

where $\lambda \in [0, 1]$ weights the distillation loss. For the case $\lambda = 0$, this trains a geometric Wasserstein GAN (WGAN), whereas $\lambda = 1$ corresponds to direct distillation. In contrast to Sauer et al. (2023), the generator is only trained on samples from the initial noise distribution ($t = 0$), not on intermediate ones ($t > 0$). Both generator and discriminator are based on a pre-trained flow matching model. In particular, we define them as

$$G^\xi(x_0) = x_0 + v_0^\xi(x_0) \quad \text{and} \quad D^\zeta(x_0) = \textstyle\sum_{i=1}^n \left(v_0^\zeta(x_0)\right)_i, \tag{15}$$



Downsampled NS (**dNSE**)
(showing magnitude of momentum)

Downsampled/sliced RTI (**dRTI/sRTI**)
(showing density)

Figure 3: Samples and partial observations from the three datasets we are working with.

with the parameters ξ and ζ initialized as copies of θ. Notably, the discriminator learns to distinguish between samples from the *true* data distribution and fake samples, which means that it is not limited by the output quality of the flow matching model. When training an ADD model, it is important that the discriminator learns faster than the generator, as it is part of the optimization problem's "inner loop". One typically achieves this by choosing a larger learning rate compared to the generator, or alternatively by iterating 5-10 discriminator steps for each generator update.

Rectified flows (Liu et al., 2023, 2024) change the "coupling" of the variables $\boldsymbol{x}_0$ and $\boldsymbol{x}_1$ from an independent joint distribution into one that has lower transport cost by retraining (rectifying) the flow matching model. After training a set of parameters θ_0 with the classic flow matching formulation (2), one iterates the following optimization problem:

$$\theta_{m+1} := \underset{\theta}{\operatorname{argmin}}\ \mathbb{E}_{p_0(\boldsymbol{x}_0),U(\boldsymbol{t};0,1)}\|v^{\theta}_{\boldsymbol{t}}(\boldsymbol{x}_{\boldsymbol{t}}) - \dot{\boldsymbol{x}}_{\boldsymbol{t}}\|^2, \quad \text{where} \quad \boldsymbol{x}_{\boldsymbol{t}} = \boldsymbol{t}\phi_1^{\theta_m}(\boldsymbol{x}_0) + (1-\boldsymbol{t})\boldsymbol{x}_0. \quad (16)$$

In theory, each time the problem is solved, the paths generated by v^{θ_m} become more straight and thus need fewer discretization steps. However, since the model is trained on the outputs of another model, approximation errors can be introduced. As for direct distillation, one has to solve the flow matching ODE for each training sample.

3. Experiments

Depending on the system, the transition distribution (4) has lower or higher variance. In the low variance regime, we conduct experiments on downsampled NS (referred to as **dNSE**) and RTI (referred to as **dRTI**) simulations (see left and center plots in Figure 3). Our main goal is to show that probabilistic models can produce accurate forecasts while being as efficient as deterministic surrogate models. Additionally, we argue and demonstrate numerically that they perform better in the high-frequency energy spectrum and tend to produce sharper features. On the other hand, if the measurements only partially capture the system dynamics, deterministic surrogate models become less and less reliable. In this regime, we train models to forecast two-dimensional "slices" of the three-dimensional RTI (referred to as **sRTI**, right plot in Figure 3). This scenario can occur, for instance, when generating inflow conditions for fluid flow simulations. For both **dRTI** and **sRTI** datasets, we augment the observations by an additional channel containing a positional encoding for the vertical axis, and another channel containing the current time step of the simulation. The former is done because of the qualitatively different features at the bottom/top of a state, while the latter ensures that the model generates an appropriate amount of mixing for the given simulation time.

	Training		Inference
Method	$\frac{\#\text{Iters}}{\text{second}}$	FM ODE solved with	#Model evaluations
Flow matching	71.03	—	10 (**dNSE**, **dRTI**) / 40 (**sRTI**)
Deterministic model	69.11	—	1
Progressive distillation	45.16	—	2
WGAN ($\frac{\#D \text{ iters}}{\#G \text{ iters}} = 5$)	12.95	—	1
Rectified flow	12.61	10 midpoint steps	4
Direct distillation	12.56	10 midpoint steps	1
ADD ($\lambda \in (0, 1)$, $\frac{\#D \text{ iters}}{\#G \text{ iters}} = 5$)	10.01	10 Euler steps	1

Table 1: Training iterations per seconds on the **dNSE** dataset with a batch size of 8 & number of function evaluations used to generate all plots in this paper (same model architecture overall).

The NS dataset is taken from the "PDEBench" dataset (Takamoto et al. (2022), where we consider the low-viscosity case with shear and bulk viscosity $\eta = 10^{-8}$, $\zeta = 10^{-8}$, respectively). The computational domain $[0, 1]^2$ with periodic boundary conditions is discretized by an equidistant 512^2 grid, and for our purposes downsampled to 64^2. The RTI dataset is taken from "The Well" (Ohana et al., 2024; Cabot and Cook, 2006), with an Atwood number of $\mathrm{At} = 3/4$, defined on the domain $[0, 1]^3$ and discretized by a 128^3 grid. After downsampling (not done for the slices experiment), this becomes 32^3. The domain is equipped with periodic boundary conditions in the horizontal directions, and free slip boundary conditions at the bottom and top. The RTI comes with 9 training and 2 test trajectories, each of length 119, and our models predict 5 steps into the future.

3.1. Training

We have trained flow matching models based on a standard UNet architecture; details are in the appendix. All experiments were conducted on a single A100 GPU.

Training speed. The number of training iterations per second is recorded in Table 1. While flow matching training is efficient, extensions that require solutions of the flow matching ODE spend more time per iteration. This includes direct distillation, rectified flows and ADD. However, for ADD, solving the ODE is only necessary for the generator. Typically, the generator is updated only every 5 or 10 iterations, which means that not much overhead is added by solving the ODE. Instead, the relatively low iteration speed stems from the gradient penalty that is computed for every discriminator update. For progressive distillation, we note that while a single training iteration can be computed more efficiently than direct distillation, the "exponential efficiency" of this approach (i.e., training a model to make N Euler steps requires only $\log_2 N$ training runs) is achieved only after multiple training runs: Most flow matching ODEs are solved to sufficient accuracy in 8-16 Euler steps, which would require at least 3-4 full training runs.

Setup complexity. Direct distillation is the easiest to setup. It works well for problems that have clear successor states (e.g., **dNSE**), but for more nondeterministic transitions, it produces blurry features. The setup for progressive distillation is more involved. It should be noted, however, that a separate model is trained for each stage, allowing us to flexibly choose the desired accuracy. Rectified flows are comparable with progressive distillation setup-wise due to the multi-stage setting. Finally, ADD requires careful tuning of generator/discriminator learning rates, the gradient penalty

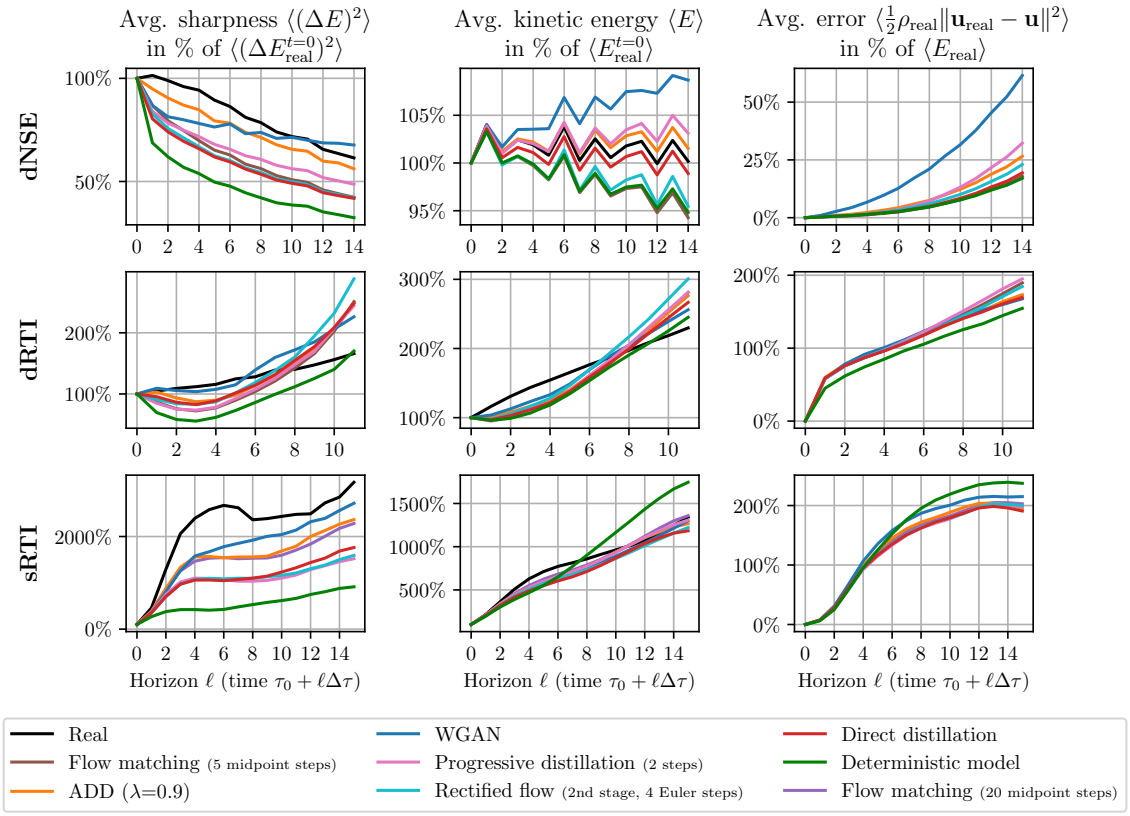


Figure 4: Statistics (sharpness, energy and error to real trajectory) of model predictions over prediction horizon. Results averaged over all initial conditions.

and the ratio of generator to discriminator updates. However, tackling these issues can be rewarding, as it makes both stable predictions and generates physically-accurate features.

3.2. Evaluation

In order to provide a qualitative assessment of the trained models, we have shown their outputs after a single prediction step in Figure 2, see also Figure 1 for some sample trajectories on the **sRTI** dataset (we show more trajectories in the appendix, Figures 8 to 10).

We first compare the deviation from the baseline trajectory over multiple steps in terms of averaged physical quantities composed of the density ρ and velocity $\mathbf{u}$. One way to define such a distance is to aggregate the pointwise error between the velocities, weighted by the real density:

$$\langle\tfrac{1}{2}\rho_{\text{real}}\|\mathbf{u}_{\text{real}}-\mathbf{u}\|^2\rangle, \tag{17}$$

where $\langle\cdot\rangle$ denotes the average over the computational domain. In particular, we can relate this error to the kinetic energy, which is defined as $E=\langle\frac{1}{2}\rho\|\mathbf{u}\|^2\rangle$. The error (17) is easy to compute, but it has its caveats. It is less sensitive to small-scale errors such as blurring, but sensitive to more qualitative changes, e.g., when shifting one of the states in space. For this, it is only really expressive in the case where the transition distribution has low variance. However, it allows us to

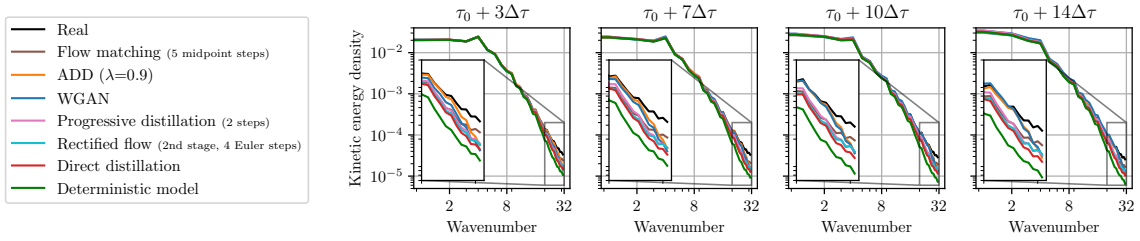


Figure 5: Spectra of the kinetic energy density on the **dNSE** dataset after multiple autoregressive prediction steps with equal initial conditions. The spectra are then averaged over all initial conditions. (See also Figure 7 in the appendix for further comparison.)

quantify strong deviations from the baseline trajectory. We have plotted this deviation in the right column of Figure 4. Here, we can observe that in particular the WGAN (ADD with $\lambda = 0$) diverges strongly for the **dNSE** dataset.

It is apparent that the deterministic model produces blurry outputs, especially for sliced RTI, something that is expected given the arguments in Section 2.2. We first study the energy spectrum focusing on the higher frequencies, representing the kinetic energy contained in smaller eddies. In Figure 5 one can observe the drop in kinetic energy of the deterministic model for large wavenumbers on the **dNSE** dataset compared with the other models. As can be seen in the figure, the spread between the deterministic and the remaining forecasts even tends to increase over time.

Another measure of "sharpness" used in older autofocus applications (Groen et al., 1985) is given by the squared Laplacian of the kinetic energy, quantifying how strongly values deviate from the average given by their neighbors. We have plotted this quantity in the left column of Figure 4. The takeaway is similar to that of Figure 5: The deterministic model ranks lowest, while the WGAN produces the sharpest results. Interestingly, most methods tend to have lower sharpness than the real trajectory on each but the **dRTI** dataset (cf. Figure 9 in the appendix).

For a final point of reference, we compare the average kinetic energy in the center column of Figure 4. For both **dRTI** and **sRTI** one can observe a strong increase over time, which is due to the fact that the potential energy of the initial state is transformed into kinetic energy as the fluids start mixing. This is captured by all models. For the **dNSE** dataset, a direct comparison with the kinetic energy of the baseline trajectory is more interesting: Some models (WGAN, progressive distillation, ADD with $\lambda = 0.9$) tend to introduce, while some others (FM, direct distillation, rectified FM, deterministic model) tend to remove energy from the system.

4. Conclusion

We have discussed extensions to the flow matching paradigm for the probabilistic surrogate modeling of partially-observed dynamical systems. In particular, we have demonstrated and compared these methods on a set of challenging datasets, including simulations of the compressible Navier–Stokes equations and the Rayleigh–Taylor instability. We have found that while harder to tune, adversarial diffusion distillation provides the best results, especially with regards to fast sampling and the generation of plausible features. In contrast, if 2–4 more function evaluations are acceptable, progressive distillation is an attractive alternative that both has a simple setup and can be trained efficiently.

Acknowledgments

HH acknowledges support by "SAIL: SustAInable Lifecycle of Intelligent Socio-Technical Systems" (Grant ID NW21-059D), funded by the Ministry of Culture and Science of the State of Northrhine Westphalia (NRW), Germany. SP acknowledges support by the European Union via the ERC Starting Grant "KoOpeRaDE" (Grant ID 101161457).

References

William H. Cabot and Andrew W. Cook. Reynolds number effects on Rayleigh–Taylor instability with possible implications for type Ia supernovae. *Nature Physics*, 2(8):562–568, August 2006. ISSN 1745-2473, 1745-2481.

Salva Rühling Cachay, Bo Zhao, Hailey Joren, and Rose Yu. DYffusion: A Dynamics-informed Diffusion Model for Spatiotemporal Forecasting. In *Proceedings of the 37th Annual Conference on Neural Information Processing Systems*, 2023.

Ashesh Chattopadhyay, Michael Gray, Tianning Wu, Anna B. Lowe, and Ruoying He. OceanNet: A principled neural operator-based digital twin for regional oceans. *Scientific Reports*, 14(1): 21181, September 2024. ISSN 2045-2322.

Prafulla Dhariwal and Alex Nichol. Diffusion Models Beat GANs on Image Synthesis. In *Proceedings of the 35th Annual Conference on Neural Information Processing Systems*, volume 34, pages 8780–8794, June 2021.

Ahmed El-Gazzar and Marcel van Gerven. Probabilistic Forecasting via Autoregressive Flow Matching, March 2025. Preprint, arXiv:2503.10375.

Fynn Fromme, Hans Harder, Christine Allen-Blanchette, and Sebastian Peitz. Surrogate Modeling of 3D Rayleigh-Benard Convection with Equivariant Autoencoders, September 2025. Preprint, arXiv:2505.13569.

Han Gao, Sebastian Kaltenbach, and Petros Koumoutsakos. Generative Learning for Forecasting the Dynamics of Complex Systems, February 2024. Preprint, arXiv:2402.17157.

Frans C. A. Groen, Ian T. Young, and Guido Ligthart. A comparison of different focus functions for use in autofocus algorithms. *Cytometry*, 6(2):81–91, March 1985. ISSN 0196-4763, 1097-0320.

Luca Guastoni and Ricardo Vinuesa. A new perspective on the simulation of stochastic problems in fluid mechanics with diffusion models. *Nature Machine Intelligence*, 7(6):816–817, Jun 2025. ISSN 2522-5839.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising Diffusion Probabilistic Models. In *Proceedings of the 34th Annual Conference on Neural Information Processing Systems*, volume 33, December 2020.

Peter Holderrieth, Marton Havasi, Jason Yim, Neta Shaul, Itai Gat, Tommi Jaakkola, Brian Karrer, Ricky T. Q. Chen, and Yaron Lipman. Generator matching: Generative modeling with arbitrary markov processes. In *The Thirteenth International Conference on Learning Representations*, 2025.

Georg Kohl, Li-Wei Chen, and Nils Thuerey. Benchmarking Autoregressive Conditional Diffusion Models for Turbulent Flow Simulation, December 2024. Preprint, arXiv:2309.01745.

Remi Lam, Alvaro Sanchez-Gonzalez, Matthew Willson, Peter Wirnsberger, Meire Fortunato, Ferran Alet, Suman Ravuri, Timo Ewalds, Zach Eaton-Rosen, Weihua Hu, Alexander Merose, Stephan Hoyer, George Holland, Oriol Vinyals, Jacklynn Stott, Alexander Pritzel, Shakir Mohamed, and Peter Battaglia. Learning skillful medium-range global weather forecasting. *Science*, 382(6677):1416–1421, December 2023. ISSN 0036-8075, 1095-9203.

Zijie Li, Anthony Zhou, and Amir Barati Farimani. Generative Latent Neural PDE Solver using Flow Matching, March 2025. Preprint, arXiv:2503.22600.

Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Neural Operator: Graph Kernel Network for Partial Differential Equations. In *Proceedings of the ICLR 2020 Workshop on Integration of Deep Neural Models and Differential Equations*, March 2020.

Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Fourier Neural Operator for Parametric Partial Differential Equations. In *Proceedings of the 9th International Conference on Learning Representations*, May 2021.

Jae Hyun Lim and Jong Chul Ye. Geometric GAN, May 2017. Preprint, arXiv:1705.02894.

Shanchuan Lin, Anran Wang, and Xiao Yang. SDXL-Lightning: Progressive Adversarial Diffusion Distillation, March 2024. Preprint, arXiv:2402.13929.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow Matching for Generative Modeling. In *Proceedings of the 11th International Conference on Learning Representations*, 2023.

Yaron Lipman, Marton Havasi, Peter Holderrieth, Neta Shaul, Matt Le, Brian Karrer, Ricky T. Q. Chen, David Lopez-Paz, Heli Ben-Hamu, and Itai Gat. Flow Matching Guide and Code, December 2024. Preprint, arXiv:2412.06264.

Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow Straight and Fast: Learning to Generate and Transfer Data with Rectified Flow. In *Proceedings of the 11th International Conference on Learning Representations*, 2023.

Xingchao Liu, Xiwen Zhang, Jianzhu Ma, Jian Peng, and Qiang Liu. InstaFlow: One Step is Enough for High-Quality Diffusion-Based Text-to-Image Generation. In *Proceedings of the 12th International Conference on Learning Representations*, March 2024.

Dongyu Luo, Jianyu Wu, Jing Wang, Hairun Xie, Xiangyu Yue, and Shixiang Tang. DiffFluid: Plain Diffusion Models are Effective Predictors of Flow Dynamics, September 2024. Preprint, arXiv:2409.13665.

Alexander Quinn Nichol and Prafulla Dhariwal. Improved Denoising Diffusion Probabilistic Models. In *Proceedings of the 38th International Conference on Machine Learning*, pages 8162–8171. PMLR, July 2021.

Ruben Ohana, Michael McCabe, Lucas Meyer, Rudy Morel, Fruzsina J Agocs, Miguel Beneitez, Marsha Berger, Blakesley Burkhart, Stuart B Dalziel, Drummond B Fielding, Daniel Fortunato, Jared A Goldberg, Keiya Hirashima, Yan-Fei Jiang, Rich R Kerswell, Suryanarayana Maddu, Jonah Miller, Payel Mukhopadhyay, Stefan S Nixon, Jeff Shen, Romain Watteaux, Bruno Régaldo-Saint Blancard, François Rozet, Liam H Parker, Miles Cranmer, and Shirley Ho. The Well: A Large-Scale Collection of Diverse Physics Simulations for Machine Learning. In *Proceedings of the 38th Annual Conference on Neural Information Processing Systems*, volume 37, pages 44989–45037, 2024.

Vivek Oommen, Siavash Khodakarami, Aniruddha Bora, Zhicheng Wang, and George Em Karniadakis. Learning Turbulent Flows with Generative Models: Super-resolution, Forecasting, and Sparse Flow Reconstruction, September 2025. Preprint, arXiv:2509.08752.

Ilan Price, Alvaro Sanchez-Gonzalez, Ferran Alet, Tom R. Andersson, Andrew El-Kadi, Dominic Masters, Timo Ewalds, Jacklynn Stott, Shakir Mohamed, Peter Battaglia, Remi Lam, and Matthew Willson. Probabilistic weather forecasting with machine learning. *Nature*, 637(8044): 84–90, January 2025. ISSN 0028-0836, 1476-4687.

Tim Salimans and Jonathan Ho. Progressive Distillation for Fast Sampling of Diffusion Models. In *Proceedings of the 10th International Conference on Learning Representations*, 2022.

Axel Sauer, Dominik Lorenz, Andreas Blattmann, and Robin Rombach. Adversarial Diffusion Distillation. In *Proceedings of the 18th European Conference on Computer Vision*, November 2023.

Aliaksandra Shysheya, Cristiana Diaconu, and Federico Bergamin. On conditional diffusion models for PDE simulations. In *Proceedings of the 38th Annual Conference on Neural Information Processing Systems*, volume 38, 2024.

Jascha Sohl-Dickstein, Eric A. Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep Unsupervised Learning using Nonequilibrium Thermodynamics. In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37, pages 2256–2265. JMLR.org, November 2015.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising Diffusion Implicit Models. In *Proceedings of the 9th International Conference on Learning Representations*, 2021.

Yang Song and Prafulla Dhariwal. Improved Techniques for Training Consistency Models. In *Proceedings of the 12th International Conference on Learning Representations (ICLR)*, volume 12, 2024.

Yang Song and Stefano Ermon. Generative Modeling by Estimating Gradients of the Data Distribution. In *Proceedings of the 33rd Annual Conference on Neural Information Processing Systems*, volume 32, pages 11895–11907. Curran Associates, Inc., 2019.

Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency Models. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, May 2023.

P. A. Srinivasan, L. Guastoni, H. Azizpour, P. Schlatter, and R. Vinuesa. Predictions of turbulent shear flows using deep neural networks. *Phys. Rev. Fluids*, 4:054603, May 2019.

Makoto Takamoto, Timothy Praditia, Raphael Leiteritz, Dan MacKinlay, Francesco Alesiani, Dirk Pflüger, and Mathias Niepert. PDEBench: An Extensive Benchmark for Scientific Machine Learning. In *Proceedings of the 36th Annual Conference on Neural Information Processing Systems*, volume 35, pages 1596–1611, 2022.

Floris Takens. Detecting strange attractors in turbulence. In *Dynamical Systems and Turbulence, Warwick 1980: proceedings of a symposium held at the University of Warwick 1979/80*, pages 366–381. Springer, 1980.

Abhijeet Vishwasrao, Sai Bharath Chandra Gutha, Andres Cremades, Klas Wijk, Aakash Patil, Catherine Gorle, Beverley J. McKeon, Hossein Azizpour, and Ricardo Vinuesa. Diff-SPORT: Diffusion-based Sensor Placement Optimization and Reconstruction of Turbulent flows in urban environments, May 2025. Preprint, arXiv:2506.00214.

Pantelis R. Vlachas, Wonmin Byeon, Zhong Y. Wan, Themistoklis P. Sapsis, and Petros Koumoutsakos. 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, May 2018. ISSN 1364-5021, 1471-2946.

Oliver Watt-Meyer, Gideon Dresdner, Jeremy McGibbon, Spencer K. Clark, Brian Henn, James Duncan, Noah D. Brenowitz, Karthik Kashinath, Michael S. Pritchard, Boris Bonev, Matthew E. Peters, and Christopher S. Bretherton. ACE: A fast, skillful learned global atmospheric model for climate prediction. In *Proceedings of the NeurIPS 2023 Workshop on Tackling Climate Change with Machine Learning*, December 2023.

Zhisheng Xiao, Karsten Kreis, and Arash Vahdat. Tackling the Generative Learning Trilemma with Denoising Diffusion GANs. In *Proceedings of the 10th International Conference on Learning Representations*, 2022.

Gefan Yang and Stefan Sommer. A Denoising Diffusion Model for Fluid Field Prediction, 2023. Preprint, arXiv:2301.11661.

Appendix A. Additional content

Here we collect additional material that is helpful for understanding the main paper. In Remark 1, we argue why determinsitic models learn the expected value. The architecture of our UNet is shown in Figure 6. In Figure 7, we relate the kinetic energy density on the **dNSE** dataset to the reference. Table 2 contains details for our training setup. Finally, full trajectories are plotted in Figures 8 to 10.

Remark 1 *The equivalence of* (6) *and* (7) *can be shown as follows. Decompose* (6) *as*

$$\mathbb{E}_{p_1(\boldsymbol{y}_{k+1},\boldsymbol{y}_k)}\|w^\theta(\boldsymbol{y}_k)\|^2 - 2\mathbb{E}_{p_1(\boldsymbol{y}_{k+1},\boldsymbol{y}_k)}[w^\theta(\boldsymbol{y}_k)\cdot\boldsymbol{y}_{k+1}] + \mathbb{E}_{p_1(\boldsymbol{y}_{k+1},\boldsymbol{y}_k)}\|\boldsymbol{y}_{k+1}\|^2, \tag{18}$$

where the last term is independent of θ*. Moreover, we have*

$$\mathbb{E}_{p_1(\boldsymbol{y}_{k+1},\boldsymbol{y}_k)}\|w^\theta(\boldsymbol{y}_k)\|^2 = \mathbb{E}_{p_1(\boldsymbol{y}_k)}\|w^\theta(\boldsymbol{y}_k)\|^2$$
$$\textit{and}\quad \mathbb{E}_{p_1(\boldsymbol{y}_{k+1},\boldsymbol{y}_k)}[w^\theta(\boldsymbol{y}_k)\cdot\boldsymbol{y}_{k+1}] = \mathbb{E}_{p_1(\boldsymbol{y}_k)}[w^\theta(\boldsymbol{y}_k)\cdot\mathbb{E}_{p_1(\boldsymbol{y}_{k+1}|\boldsymbol{y}_k)}[\boldsymbol{y}_{k+1}]]$$

by independence of $w^\theta(\boldsymbol{y}_k)$ *from* $\boldsymbol{y}_{k+1}$*. Substituting the right-hand sides into* (18) *and changing the constant term to* $\mathbb{E}_{p_1(\boldsymbol{y}_k)}\|\mathbb{E}_{p_1(\boldsymbol{y}_{k+1}|\boldsymbol{y}_k)}[\boldsymbol{y}_{k+1}]\|^2$*, we can rewrite the expression to yield* (7).

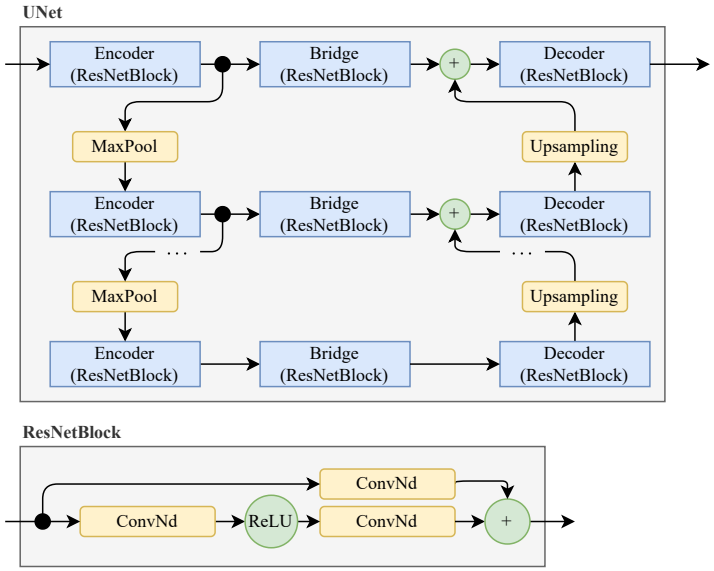


Figure 6: UNet architecture.

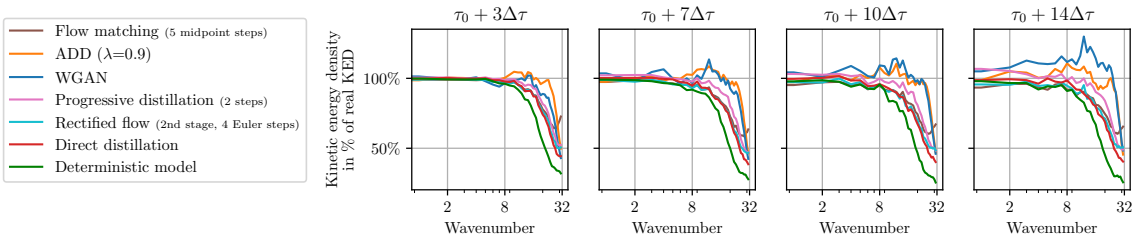


Figure 7: Kinetic energy densities from Figure 5 in relationship to the real kinetic energy density.

	Dataset	#Iters	$\frac{\#\text{Iters}}{\text{second}}$	Batch	Learning rate	Channels[1]	#Params
Flow Matching	dNSE	820k	27.85	32	1e-5	$128, 196, 196$	4,4m
	dRTI	70k	3.46	8	1e-5	$128, 196$	6,4m
	sRTI	350k	7.11	32	1e-5	$128, 256, 256, 128$	8,6m

	Dataset	#Iters	$\frac{\#\text{Iters}}{\text{second}}$	Batch	Learning rate
Deterministic Model	dNSE	200k	69.11	8	1e-5
	dRTI	60k	10.33	8	1e-5
	sRTI	500k	25.54	8	1e-5

	Dataset	#Iters	$\frac{\#\text{Iters}}{\text{second}}$	Batch	Learning rate
Direct Distillation[2]	dNSE	100k	12.56	8	1e-5
	dRTI	40k	3.28	4	1e-5
	sRTI	350k	6.07	4	1e-5

	Dataset	$\frac{\#\text{Iters}}{\text{stage}}$	$\frac{\#\text{Iters}}{\text{second}}$	Batch	Learning rate	Stages
Progressive Distillation	dNSE	100k	45.16	8	1e-5	$m = 16, 8, 4, 2, 1$
	dRTI	30k	6.58	8	1e-5	$m = 16, 8, 4, 2, 1$
	sRTI	100k	14.71	8	1e-5	$m = 16, 8, 4, 2, 1$

	Dataset	$\frac{\#\text{Iters}}{\text{stage}}$	$\frac{\#\text{Iters}}{\text{second}}$	Batch	Learning rate	#Stages
Rectifiying Flows[2]	dNSE	100k	12.61	8	1e-5	2
	dRTI	20k	1.77	8	1e-5	2
	sRTI	100k	3.73	8	1e-5	2

					Learning rate			
	Dataset	#Iters	$\frac{\#\text{Iters}}{\text{second}}$	Batch	G	D	$\frac{\#D \text{ iters}}{\#G \text{ iters}}$	γ
ADD[3]	dNSE	150k	10.01[4]/12.95[5]	8	5e-6	5e-5	5	5.0
		+20k			1e-6	1e-5		
	dRTI	35k	1.45[4]/3.33[5]	4	1e-6	1e-5	10	25.0
	sRTI	150k	3.16[4]/3.45[5]	8	1e-5	1e-4	5	25.0
		+10k			1e-6	5e-5		

Table 2: Training settings for each method and dataset. All models are based on the same UNet architecture given by the corresponding flow matching model. [1]Channels along the downsampling path of the UNet. [2]FM ODE solved with 10 midpoint steps. [3]FM ODE solved with 10 Euler steps. [4]With distillation loss. [5]Without distillation loss, i.e., WGAN.

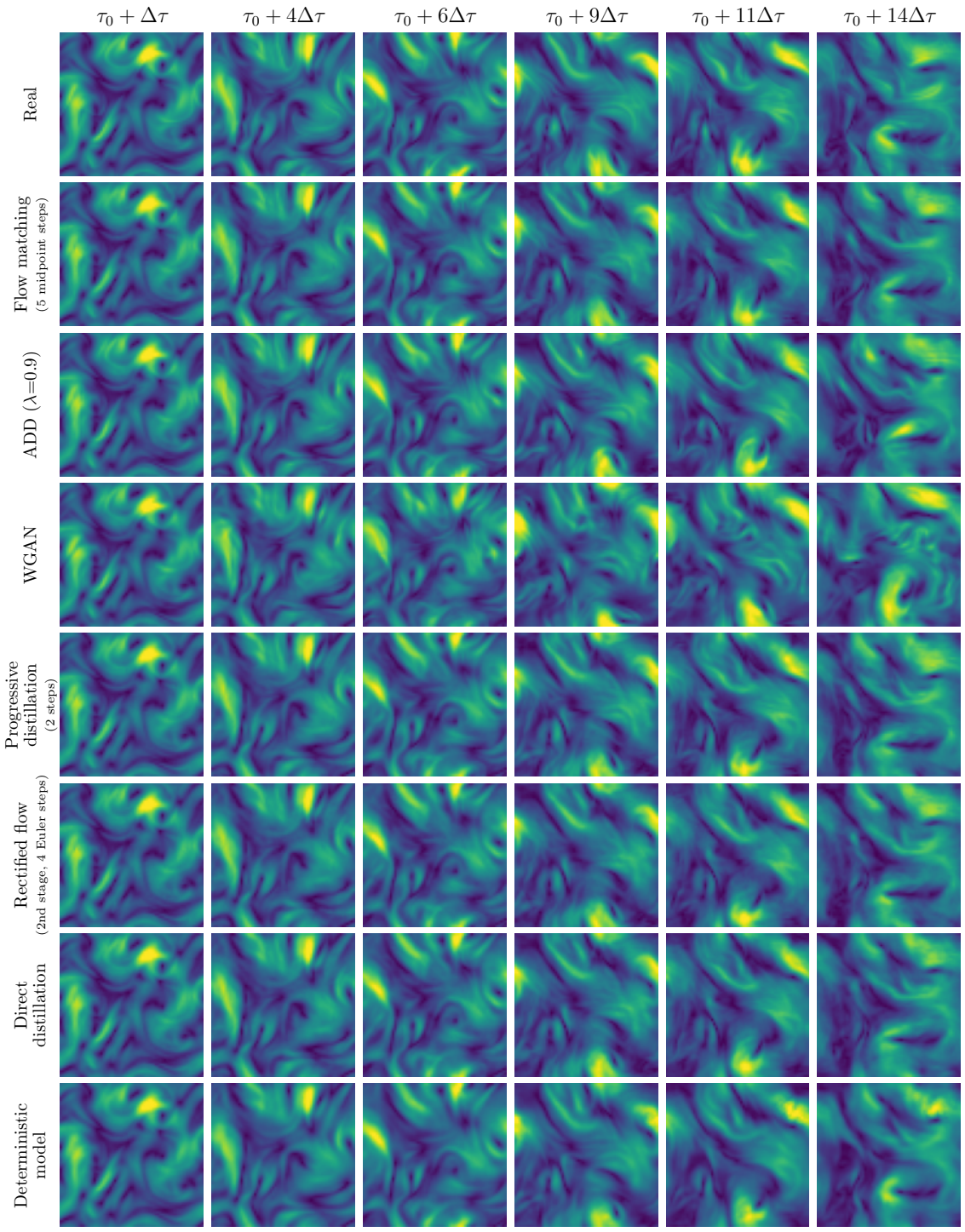


Figure 8: Multiple trajectories on the **dNSE** dataset.

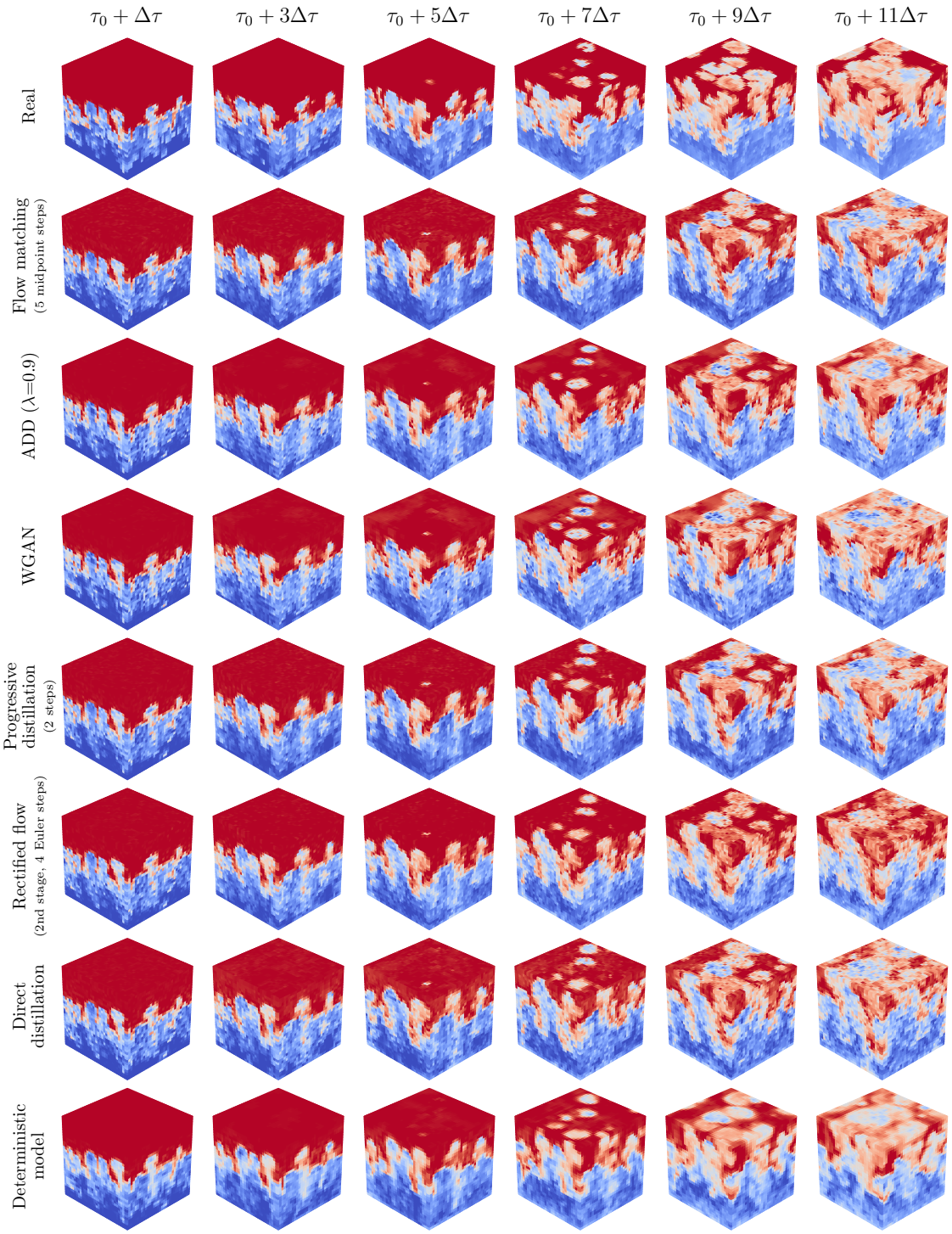


Figure 9: Multiple trajectories on the **dRTI** dataset.

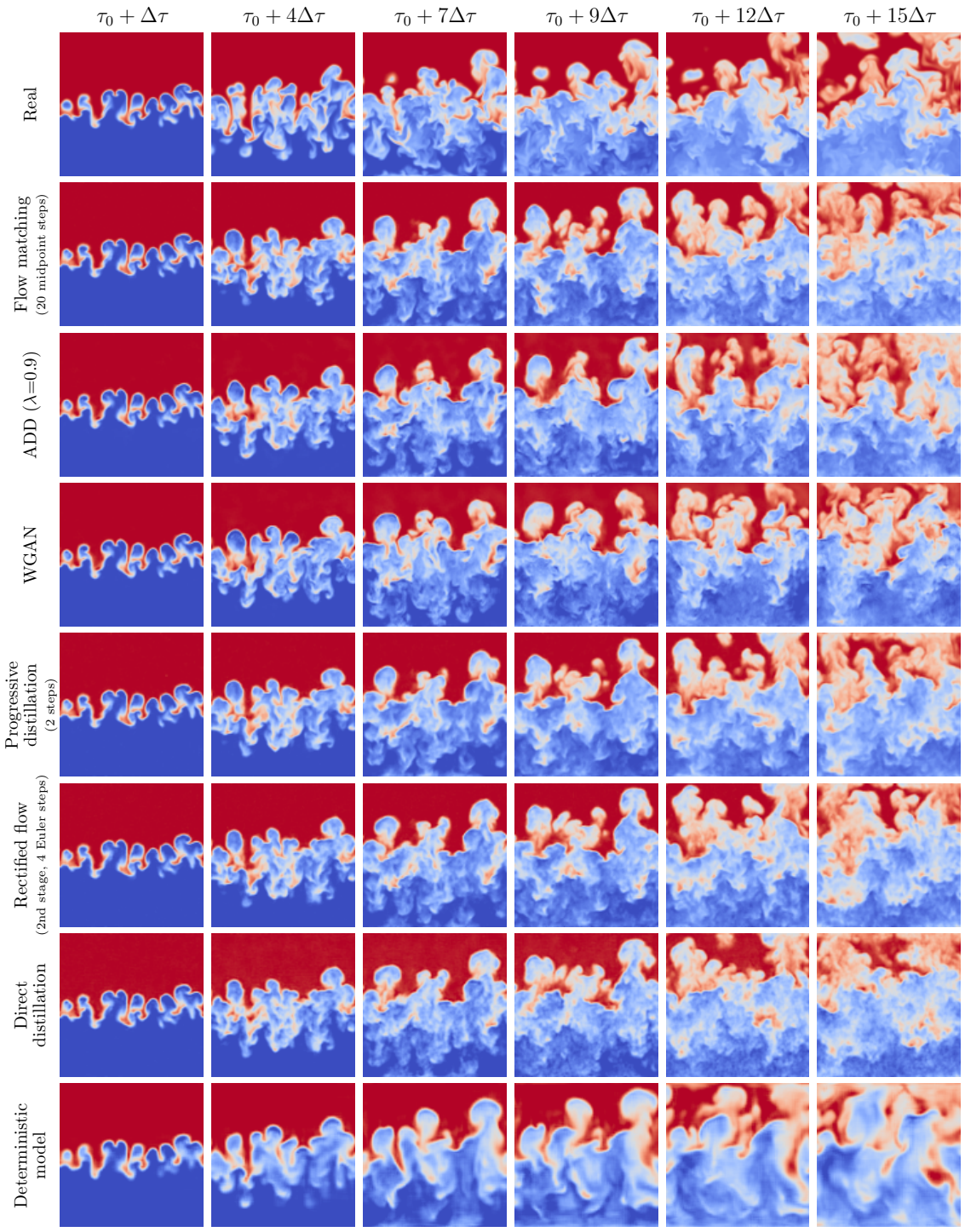


Figure 10: Multiple trajectories on the **sRTI** dataset.