User-defined Event Sampling and Uncertainty Quantification in Diffusion Models for Physical Dynamical Systems

Marc Finzi [1 2 3] **Anudhyan Boral** [1] **Andrew Gordon Wilson** [2] **Fei Sha** [1] **Leonardo Zepeda-Núñez** [1]

Abstract

Diffusion models are a class of probabilistic generative models that have been widely used as a prior for image processing tasks like text conditional generation and inpainting. We demonstrate that these models can be adapted to make predictions and provide uncertainty quantification for chaotic dynamical systems. In these applications, diffusion models can implicitly represent knowledge about outliers and extreme events; however, querying that knowledge through conditional sampling or measuring probabilities is surprisingly difficult. Existing methods for conditional sampling at inference time seek mainly to enforce the constraints, which is insufficient to match the statistics of the distribution or compute the probability of the chosen events. To achieve these ends, optimally one would use the conditional score function, but its computation is typically intractable. In this work, we develop a probabilistic approximation scheme for the conditional score function which provably converges to the true distribution as the noise level decreases. With this scheme we are able to sample conditionally on nonlinear user-defined events at inference time, and matches data statistics even when sampling from the tails of the distribution.

1. Introduction

Accurately predicting trajectories for chaotic dynamical systems is a great scientific challenge of societal importance. For instance, despite the impressive progress of numerical weather prediction (Richardson, 1922; Bauer et al., 2015), current methodologies still struggle to forecast extreme events. Heat waves (Perkins & Alexander, 2013), flooding (Mosavi et al., 2018), and oceanic rogue waves (Dysthe et al., 2008), are examples of catastrophic events of enormous socio-economic impact. Part of the difficulty of forecasting such extreme events can be attributed to the chaotic behavior of the dynamical systems associated with weather prediction (Lorenz, 1963; Hochman et al., 2019). Furthermore, extreme events are often located at the tail of the distribution and are non-trivial to characterize, rendering them hard to sample efficiently, which in return has spun several methods seeking to attenuate this issue (Kahn & Harris, 1951; Rosenbluth & Rosenbluth, 1955; Farazmand & Sapsis, 2019; Qi & Majda, 2020). Most of the methods above are based on rejection-sampling (Hastings, 1970; Rossky et al., 1978), whose cost, measured as the number of on-demand large-scale simulations of the dynamical system, increases as events become rarer, rapidly becoming prohibitive.

Recent advances in deep generative models, particularly diffusion-based models, have shown remarkable results in capturing statistics of high-dimensional variables (such as images) and generating new samples from the learned probabilistic models (Sohl-Dickstein et al., 2015; Ho et al., 2020; Song & Ermon, 2019; Song et al., 2020). In constrast with GANs (Goodfellow et al., 2020) which often struggle with dropping modes that are difficult to model, diffusion models have proven better at capturing the full diversity of the data.

In this paper, we investigate the application of diffusion models to modeling trajectories. In particular, we are interested in whether such models can be used as surrogate models for the physical systems. We consider three classical dynamical systems: Lorenz strange attractors ("butterfly") (Figure 1 left), the double pendulum, and FitzHugh-Nagumo neuron model (Figure 1 right). Even though these models are fairly simple they retain the core difficulty of more complex ones, e.g., the first two exhibit chaotic trajectories while the last one exhibits extreme events.

Concretely, we study two questions. First, *can we learn diffusion-based generative models directly from trajecto-*

[1]Google Research, 1600 Amphitheatre Pkwy Mountain View CA 94043, USA [2]Department of Computer Science, New York University, NYC, USA [3]Work done during an internship at Google Research. Correspondence to: Marc Finzi <maf820@nyu.edu>.

Proceedings of the 40th International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023. Copyright 2023 by the author(s).

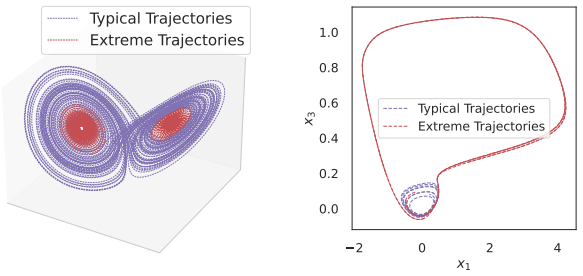


Figure 1: Chaotic nonlinear dynamical systems often have outlier events, and these events can be difficult to predict due to the chaotic nature of the system. **Left:** Trajectories of the Lorenz attractor, split into the trajectories which do not cross over to the opposite arm of the attractor in a given time horizon vs those that do. **Right:** Trajectories of the FitzHugh-Nagumo model, which feature the rare and unpredictible neuron spikes shown in red, which are nestled in with typical trajectories shown in purple.

ries without explicit knowledge of the underlying differential equations and sample high-fidelity trajectories from the models? Second, *can we condition the sampling process to generate user-specified events — trajectories of certain properties — without the need to retrain the model?* Positive answers will enable researchers and practitioners to query the learned models with amortized computational costs, and the flexibility of studying events in tails of the distribution.

We answer the first question by building diffusion models capable of learning the trajectories of the three classical systems mentioned above. The models can produce trajectories with low error and calibrated uncertainties even when the underlying system is chaotic.

We answer the second question by deriving an approximation scheme to compute the conditional score function that enables conditioning on user-defined nonlinear statistics at inference time. The key idea is to use moment-matching to derive the distribution of the denoised sample conditioned on a noised sample, which is typically intractable. We show that the approximation becomes exact as the noise scale vanishes. Using this method, we can directly sample events in the tail of the distribution and quantify their likelihood.

2. Related Work

Denoising Diffusion Probabilistic Models Denoising Diffusion Probablisitic Models (DDPMs) (Sohl-Dickstein et al., 2015; Ho et al., 2020; Song & Ermon, 2019) construct a forward process where each training example from the data distribution is sequentially corrupted by increasingly larger noise. At the final step of this process, the sample is distributed according to a standard Gaussian distribution, completely erasing the data. The reverse process defines a generative model where, starting from a standard Gaussian sample, we follow the reverse denoising process using a neural network. These diffusion models are trained using score matching (Hyvärinen, 2005; Vincent, 2011) or denoising (Ho et al., 2020) objectives. Song et al. (2020) introduce a continuous formulation of the diffusion process using stochastic differential equations (SDEs). Furthermore, by leveraging the connection with Neural Ordinary Differential Equations (NeuralODEs) (Chen et al., 2018), Song et al. (2020) show how to perform exact likelihood computation.

A posteriori conditioning

Inference time (*a posteriori*) conditioning is a promising and powerful paradigm for training large prior models and using them to perform different downstream tasks like inpainting, colorization, reconstruction, and solving general inverse problems. In contrast with (*a priori*) train time conditioning, where the form of the conditioning must be known and used at training time, inference time conditioning enables using a unconditional diffusion model as a *prior* and then conditioning it on different observations at inference time.

Song et al. (2020) provide a crude approximation of the conditional score function to perform inpainting. Meng et al. (2021) use a similar method, but instead they perform unconditional generation from a noised version of a guide image. Chung & Ye (2022) and Chung et al. (2022c) generalize the method to linear transformations (such as in MRI reconstruction) and apply an explicit projection onto the constraints at each iteration. As shown by Lugmayr et al. (2022) and others (Chung et al., 2022b), these projection strategies (e.g. replacing noised versions of known pixels with the Gaussian samples $p(x_t|x_0)$ for inpainting) produces samples that lack global coherence: inpainted regions do not properly integrate information from the known regions. Lugmayr et al. (2022) address this by iterating forwards and backwards multiple times in order to better harmonize the information. Chung et al. (2022b) take a different approach with manifold constrained gradients (MCG): they combine constraint projection with an additional term that encourages the conditional sample iterates x_t to lie on the data manifold given by $p(x_t)$. Both methods address the global coherency issue of prior methods; however, they are only valid for linear equality constraints and it is unclear how the samples relate to the true conditional distribution of the generative model.

Graikos et al. (2022) take a more general approach for inverse problems using optimization, but only produce point estimates. Recently, Chung et al. (2022a) proposed an improved version of MCG that removes the projections and enables it to work both when there is measurement noise and nonlinear constraints. This method makes a probabilis-

tic approximation of the score function, which becomes degenerate as the measurement noise goes to 0, thus limiting its applicability for tail sampling of the deterministic events we consider here.

Likelihoods Even though diffusion models are not explicitly designed as likelihood models, they can be used to compute the likelihood $p(x_0)$ of a particular data point according to the model. In order to obtain more than a lower bound on the likelihood, one needs to exploit the connection between the probability flow ODE and continuous normalizing flows (Grathwohl et al., 2018) as done in Song et al. (2020), unfortunately, such connection is only available in continuous time.

Extreme event prediction Qi & Majda (2020) apply deep convolutional neural networks to the prediction of extreme events in dynamical systems. Wan et al. (2018) use reduced order modelling in conjunction with LSTM-RNN networks to model extreme events. Asch et al. (2022) tackle the problem of lack of data when training deep networks for extreme event prediction. Doan et al. (2021) use reservoir-computing based model to forecast extreme events. Guth & Sapsis (2019) use machine learning to detect extreme events in advance from a given trajectory; rather than modeling trajectories of dynamical systems directly. In contrast to the above works, our approach uses likelihood-based generative models which allows us to provide probability estimates for the extreme event occurring. Deep generative models, particularly NeuralODEs (Chen et al., 2018; Yildiz et al., 2019) have shown promise in modeling dynamical systems (Lai et al., 2021), but their effectiveness in capturing tail events is yet to be ascertained.

3. Diffusion Model for Dynamical Systems

Background Diffusion models are composed of a forward noising process and its corresponding backward denoising process, which we describe below in its continuous-time formulation (Song et al., 2020).

The forward process evolves a given clean signal $x_0 \in \mathbb{R}^d$ from time $t = 0$ to $t = 1$ via the Itô Stochastic Differential Equation (SDE)

$$dx_t = f(x_t, t)dt + g(t)dW,$$

where W is the Wiener process, adding noise at each step. Ultimately, x_t, the signal at time t, is a transformed (due to the drift term $f(x, t)$) and noised (due to the diffusion term $g(t)dW$) version of x_0. These values are also chosen so that the marginal distribution $p(x_1)$ is simply a spherical Gaussian. Here we denote the distribution of the data given the noise level at time t as $p(x_t)$, though this distribution depends on t and is sometimes written as $p_{X_t}(x_t)$ or $p_t(x_t)$.

If $f(x, t)$ is affine, the noise kernel $p(x_t|x_0)$ can be computed in closed-form. In this work, we define

$$f(x, t) = \tfrac{\dot{s}_t}{s_t} x \text{ and } g(t)^2 = \sigma_t \dot{\sigma}_t - \sigma_t^2 \tfrac{\dot{s}_t}{s_t}. \quad (1)$$

If these relations are specified, then the noise kernel has a simple form:

$$p(x_t|x_0) = \mathcal{N}(x_t; s_t x_0, \sigma_t^2 I). \quad (2)$$

Namely, the diffusion model describes how x_0 is scaled and blurred. The derivation of Equation 1 can be found in Särkkä & Solin (2019) and specific choices of s_t and σ_t are described in Song et al. (2020); Karras et al. (2022).

The reverse process removes the noise from data. Specifically, given a noisy sample x_1 from the marginal distribution $p(x_1) = \mathcal{N}(0, \sigma_1^2 I)$, the backward process evolves x_1 to x_0 by the following SDE

$$dx_t = \big(f(x_t, t) - g(t)^2 \nabla_{x_t} \log p(x_t)\big)dt + g(t)d\bar{W}, \quad (3)$$

where $\bar{W}$ is the Wiener process running backwards. $\nabla_{x_t} \log p(x_t)$ is the score function for the noised data, defined as the gradient of $\log p(x_t)$, the marginal probability of x_t. The diffusion model can be seen as a sequence of denoising steps according to the model score $s_\theta(x_t, t)$ aimed to match the noised score function of the true data $\nabla_{x_t} \log p_{\text{data}}(x_t)$. This can be achieved with standard optimization algorithms on the score matching loss. From now on we will refer to $\nabla_{x_t} \log p(x_t)$ not as the noised scores of the data distribution, but as the noised scores of the model distribution.

We use the continuous time score matching formulation of diffusion models (Song et al., 2020) to enable likelihood computations, and we train using the score matching loss

$$L(\theta) = \mathbb{E}_{(t,x_0,x_t)} \|s_\theta(x_t, t) - \nabla_{x_t} \log p(x_t|x_0)\|^2 / \sigma_t^2,$$

where $x_0 \sim \mathcal{D}$ (the data distribution), $x_t \sim p(x_t|x_0)$, and $t \sim \mathrm{U}[0, 1]$. For hyperparameters and additional training setup, see Appendix G. After training, we sample from the model using the Euler-Maruyama integrator with 1000 uniformly spaced timesteps applied to the SDE in Equation 3.

Application to Dynamical Systems The trajectory of a dynamical system for a given initial condition $x(0)$, is the function $x(\tau) : [0, T] \to \mathbb{R}^d$, which is the solution to

$$\dot{x}(\tau) = g(x(\tau), \tau),$$

for a given dynamics function g. We use ODE time τ to distinguish it from the diffusion time t. Note that the initial condition $x(0)$ should not be confused with x_0, where the latter refers to a noise free data point.

We suppose that the initial condition $x(0)$ follows a certain distribution, which in return generates a distribution of

trajectories. We assume each trajectory is discretized into m timesteps, yielding a $m \times d$-dimensional array. We use diffusion models to model the collection of the trajectories. The architecture of the diffusion model's score function is described in Appendix H.

4. A Posteriori Conditioning

Once $p(x)$ is learnt, we would like to obtain samples from it with properties of interest. Abstractly, we seek

$$p(x_0|E)$$

where the property E is a set given by $E = \{x_0 : C(x_0) = y\}$, for equality constraints or $E = \{x_0 : C(x_0) \leq y\}$, for inequality constraints, for some smooth function $C : \mathbb{R}^{md} \to \mathbb{R}^n$. This construction is fairly general and it can be used for different downstream applications, e.g., in the task of image inpainting, the property $C(x_0) = y$ encodes whether the observed portion of a sampled image x_0 corresponds to known pixel values y.

In order to perform *a posteriori* conditional sampling using the learned model, we use the score function of the conditional distribution

$$\begin{aligned}\nabla_{x_t} \log p(x_t|E) &= \nabla_{x_t} \log p(E|x_t) + \nabla_{x_t} \log p(x_t) \\ &= \nabla_{x_t} \log p(E|x_t) + s_\theta(x_t, t) \end{aligned} \tag{4}$$

in the reverse diffusion process. The challenge is to compute the first term while knowing the definition of E only *after* $p(x_t)$ is learned (without knowledge of E).

Directly computing this quantity is hard: $p(E|x_t) = \int \mathbb{I}_{[x_0 \in E]} p(x_0|x_t) dx_0$, which is intractable. A naive approach would be to use $p(E|x_0)$ in place of $p(E|x_t)$. However, this approach leads to conditional samples lacking global coherence (Lugmayr et al., 2022; Chung et al., 2022b). Instead, we derive an approximation to $p(x_0|x_t)$ based on moment matching, which we use to perform conditioning with linear and nonlinear equality constraints as well as inequality constraints.

4.1. Moment-matching Based Approximation

We can view the forward diffusion process as a Bayesian inference task:

$$x_0 \sim p(x_0), \quad p(x_t|x_0) = \mathcal{N}(x_t; s_t x_0, \sigma_t^2 I). \tag{5}$$

To compute the mean of $p(x_0|x_t)$, we apply Tweedie's formula (Robbins, 1992; Efron, 2011)

$$\mathbb{E}[x_0|x_t] = \frac{x_t + \sigma_t^2 \nabla_{x_t} \log p(x_t)}{s_t} := \hat{x}_0(x_t), \tag{6}$$

where $\hat{x}_0$ represents the best guess for x_0 given a value of x_t. Furthermore, Tweedie's formula can also be applied to the higher moments. As we derive in Appendix A, the conditional covariance matrix can be expressed exactly as

$$\mathrm{Cov}[x_0|x_t] = \left[\frac{\sigma_t^2}{s_t^2}\left(I + \sigma_t^2 \nabla^2 \log p(x_t)\right)\right] := \hat{\Sigma}(x_t), \tag{7}$$

where $\nabla^2 \log p(x_t)$ is the Hessian of the log probability, or, equivalently, the Jacobian of the score function.

Using these two expressions for the conditional mean and covariance of x_0, we can approximate $p(x_0|x_t)$ with a Gaussian

$$p(x_0|x_t) \approx \mathcal{N}(\hat{x}_0, \hat{\Sigma}), \tag{8}$$

which can be conveniently applied to constraining x_0 to satisfy the desired property E.

4.2. Linear Equality Constraints

As an example, we consider the goal of imposing the set of linear constraints $E = \{x_0 : Cx_0 = y\}$ onto samples from the diffusion model for a given constraint matrix $C \in \mathbb{R}^{r \times d}$ and $y \in \mathbb{R}^r$. The linear transformation Cx_0 leads to yet another Gaussian

$$p(Cx_0|x_t) \approx \mathcal{N}(C\hat{x}_0, C\hat{\Sigma}C^\top). \tag{9}$$

Both $\hat{x}_0$ and $\hat{\Sigma}$ depend on x_t, and the matrix $C\hat{\Sigma}C^\top$ can be computed using automatic differentiation involving the Jacobian of the score function. Specifically, the matrix $C\hat{\Sigma}$ can be computed as the Jacobian of the map $x_t \mapsto (\sigma_t^2/s_t)C\hat{x}_0$. We now have,

$$\nabla_{x_t} \log p(E|x_t) \approx \nabla_{x_t} \log \mathcal{N}(y; C\hat{x}_0, C\hat{\Sigma}C^\top) \tag{10}$$

This enables using the modified score function 4 to sample from the constrained generative process.

Remark In the case of linear constraints studied here, our work generalizes the recent work on constraining samples conditioned on linear constraints to be consistent with the data manifold (Chung et al., 2022b). Specifically, if we approximate the Hessian as $\sigma_t^2 \nabla^2 \log p(x_t) \approx \lambda I$, we arrive at

$$\nabla_{x_t} \log p(x_t|y) = s_\theta(x_t, t) - \frac{s_t^2}{2(1+\lambda)\sigma_t^2} \nabla_{x_t} \|C\hat{x}_0 - y\|^2,$$

which reproduces eq.(14) in (Chung et al., 2022b), ignoring constraint projections, and up to a scaling matrix W lacking the s_t^2/σ_t^2 factor. In Appendix D, we show how the additional constraint projection for linear constraints arises under different circumstances with our moment matching approximation.

However, using (Chung et al., 2022b) or (Chung et al., 2022a) directly with adaptive step size integrators in the continuous time formulation leads to numerical issues due

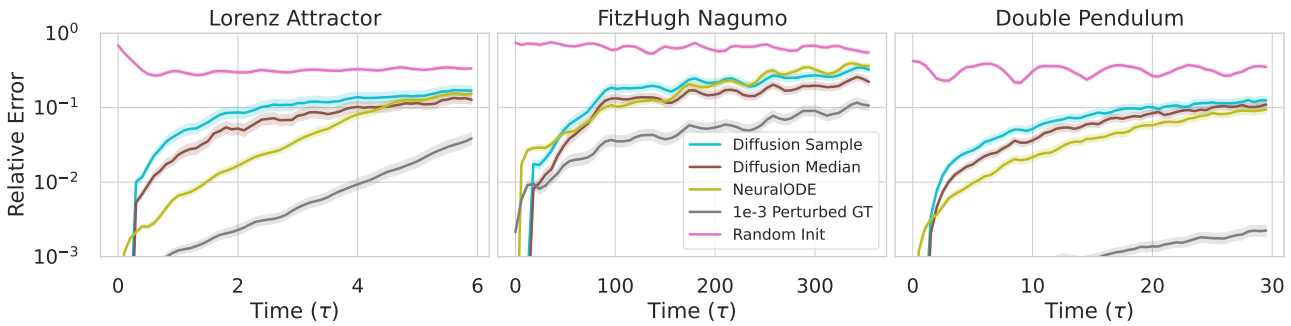


Figure 2: Relative error of sample diffusion model samples and pointwise median of 20 samples when conditioned on initial condition compared to ground truth trajectories, NeuralODE rollouts, as well as errors for small perturbations of the initial condition evolved with the simulator, and independently sampled initial conditions for comparison. Shaded lines show two standard errors computed in log space. Diffusion model is evaluated on **Left:** Lorenz attractor, **Middle:** Fitzhugh-Nagumo model, and **Right:** double pendulum. The diffusion model and NeuralODE perform similarly, despite the diffusion model lacking ODE specific inductive biases of NeuralODE.

to the stiffness of the problem; the ODE integrator step sizes shrink to zero and the integration does not complete. We explore these issues further in Appendix E. Observing that the ratio $s_t^2/\sigma_t^2 = \text{SNR}$ (signal to noise ratio) typically varies over 12 orders of magnitude in the region $t \in (0, 1]$, we can now understand how the misscaling of this term leads to numerical problems. Adding the additional scaling factor s_t^2/σ_t^2 enables us to use the method in continuous time. Additionally, we can now leverage our probabilistic interpretation and the full covariance matrix to determine how to condition on nonlinear and inequality constraints.

4.3. Nonlinear Equality Constraints

For a set of nonlinear constraints $C(x_0) = y$, we approximate $C(x_0)$ with its first-order Taylor expansion:

$$C(x_0) \approx C(\hat{x}_0) + \nabla C \cdot (x_0 - \hat{x}_0). \tag{11}$$

The Jacobian ∇C is evaluated at $\hat{x}_0$. In this "linearized" constraint, we approximate the desired probability using the result from the previous section on linear constraints:

$$p(C(x_0) = y|x_t) \approx \mathcal{N}(C(\hat{x}_0), \nabla C\, \hat{\Sigma}\, \nabla C^\top). \tag{12}$$

The quality of the approximation depends on how much the function C varies over the scale of the variance of x_0, which we evaluate in subsection 5.3.

4.4. Inequality Constraints

With the Gaussian approximations of Equation 10 and Equation 12 for linear and nonlinear equality constraints, we can also handle inequality constraints such as $E = \{x_0 : C(x_0) > y\}$. For a one-dimensional inequality constraint,

$$p(E|x_t) = p(C(x_0) > y|x_t) \approx \Phi\left(\frac{C(\hat{x}_0) - y}{\sqrt{\nabla C^\top \hat{\Sigma} \nabla C}}\right),$$

where Φ is the Gaussian CDF.

Plugging the above into the score function for sampling conditional distribution Equation 4, we can directly sample from the tails of the distribution according to any user defined nonlinear statistic $C(\cdot)$, focusing the model on extreme and rare events, using the conditional scores

$$\nabla_{x_t} \log p(x_t|E) \approx s_\theta(x_t, t) + \nabla_{x_t} \log \Phi\left(\frac{C(\hat{x}_0) - y}{\sqrt{\nabla C^\top \hat{\Sigma} \nabla C}}\right).$$

4.5. Likelihoods of Events

Since we can obtain samples from $p(x_0|E)$, for a given user-defined event E, we can also compute the marginal likelihood of such an event. Concretely, given a sample x_0, we can break down the likelihood using Bayes rule:

$$\log p(E) = \log p(x_0) - \log p(x_0|E) + \log p(E|x_0). \tag{13}$$

When sampling $x_0 \sim p(x_0|E)$, the last term is zero when x_0 is a member of E. The two probabilities $p(x_0)$ and $p(x_0|E)$ can be computed by integrating the probability flow ODEs (Song et al., 2020)

$$\dot{x}_t = f(x_t, t) - \tfrac{1}{2} g(t)^2 \nabla_{x_t} \log p(x_t), \tag{14}$$

$$\dot{x}_t = f(x_t, t) - \tfrac{1}{2} g(t)^2 \nabla_{x_t} \log p(x_t|E), \tag{15}$$

forwards in time with the continuous change of variables formula derived in FFJORD (Grathwohl et al., 2018). To reduce variance and the integration time, we use a second-order Heun integrator with a fixed time step, and compute the Jacobian log determinant directly using autograd. For more details, see Appendix I. While the above procedure is theoretically valid for a single sample x_0, we can average the estimate over multiple samples $x_0|E$ for improved accuracy.

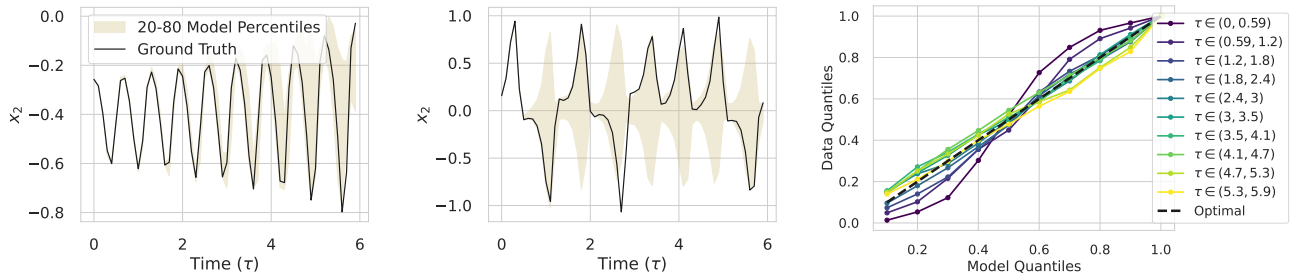


Figure 3: Uncertainty quantification captured by the learned diffusion model. (**Left**) 20-80th precentiles state values for the x_2 component of the Lorenz attractor diffusion model, compared to the actual trajectory. As this trajectory lies in a region of the state space with a lower local Lyapunov exponent, the dynamics are less chaotic here, and the model uncertainties are relatively low. (**Middle**) For samples in the very chaotic region, state transitions to the opposite arm of the attractor are common and captured by the uncertainties. (**Right**) Calibration of the predicted uncertainty quantiles for pointwise predictions as compared to the ground truth empirical quantiles, where optimal lies along $x = y$. While not perfect, the models are reasonably well calibrated, with bias to uncertainties being slightly to broad at early times and too narrow at late times.

5. Results

In order to evaluate the capability of diffusion models to express chaotic nonlinear dynamics, we train the models to fit the distribution of trajectories over different initial conditions for the test problems. For each system, we choose a Gaussian initial condition distribution, and integrate for a sufficient time to allow the distribution to reach equilibrium before recording the data. We choose s_t and σ_t according to the variance exploding process, i.e., $s_t = 1$ and $\sigma_t = \sigma_{\min}\sqrt{(\sigma_{\max}/\sigma_{\min})^{2t} - 1}$. Equation parameters and collection details are specified in Appendix F. We train on the following three dynamical systems:

Lorenz Attractor The Lorenz attractor (Lorenz, 1963) is a well-studied example of chaotic behavior, governed by a coupled three dimensional nonlinear ODE. The system contains two prominent arms of a strange attractor, and trajectories chaotically switch between the two arms.

FitzHugh-Nagumo The FitzHugh-Nagumo model (FitzHugh, 1961) is a nonlinear ODE given by the coupled equations

$$\frac{dx_i}{d\tau} = x_i(a_i - x_i)(x_i - 1) + y_i + k\sum_{j=1}^{n} A_{ij}(x_j - x_i),$$

$$\frac{dy_i}{d\tau} = b_i x_i - c_i y_i,$$

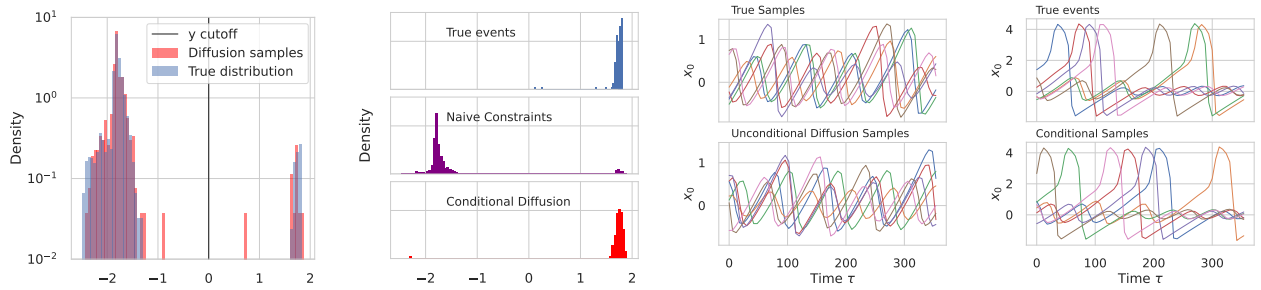
for $i = 1, 2$, leading to a set of four equations modeling the dynamics of two coupled neurons. The values for the parameters are specified in Appendix F and match the choice of Farazmand & Sapsis (2019) which cause the system to exhibit rare but high magnitude neuron spikes, such as the examples shown in Figure 1 (right) and Figure 4 (d).

Double Pendulum The double pendulum is another classic example of a chaotic system produced by the dynamics of a rigid pendulum with two point masses under the influence of gravity, but with trajectories having distinct energies unlike the previous two systems.

5.1. Unconditional Samples

Point Predictions We first evaluate the extent to which the model captures the system dynamics by measuring its ability to make accurate predictions of a trajectory given an initial condition. For each dataset, we train a conditional model that takes in the first three timesteps of a trajectory so as to make it conditional on the initial conditions (prior conditioning). Both the errors of a single sample and the pointwise median of 20 samples are shown, which are computed against the ground truth from a held out set of initial conditions. We compare the prediction relative errors against NeuralODE models, which were trained on the same data (see Appendix F for details), and perturbed ground truth, in which we use a ground truth trajectory but with an initial condition perturbed by Gaussian noise with standard deviation 10^{-3}. The last comparison specifies how the error of a small perturbation is chaotically amplified, and therefore provides a lower bound on performance. We measure relative error as $\text{RelativeError}(a, b) = \|a - b\|/(\|a\| + \|b\|)$. Figure 2 shows that diffusion models capture the dynamics of the system performing similarly to NeuralODEs on pure point predictions, despite the fact that NeuralODEs make use of the ODE structure bias whereas the diffusion model does not.

Uncertainty Quantification While the model makes accurate point predictions, we are interested in the *distribution* over the outputs captured by the model. In particular, we provide numerical evidence showing that the pointwise uncertainties over the state values are reasonably well calibrated, i.e., the quantiles of the model predictive distribution at a given point includes the truth the appropriate fraction of the time. In principle, there are two sources of



(a) $C(x)$ (b) $C(x)|E$ (c) $x \sim p(x)$ (d) $x \sim p(x|E)$

Figure 4: (**a**) Histograms of statistic values $C(x)$ in the Fitzhugh-Nagumo system, for trajectories sampled unconditionally from from the dataset and the model. Here the event describes the unusual large scale neuron spikes that exist as a small cluster of outliers at $C(x) > 1.5$. Notably the unconditional model distribution captures the outliers. (**b**) Statistic values on samples produced by our method of tail sampling vs actual extreme events vs naive approach to enforce the constraints. (**c**) Unconditional trajectory samples from both the data and model distributions. (**d**) Example event conditional samples compared to actual events. When conditioning on the event defined by the statistic value $E = [C(x) > 0]$, the model is able to conditionally sample from the distribution, unlike for the naive approach to incorporate the constraints.

uncertainty with the model when trained on these dynamical systems: uncertainty associated with the chaotic growth of error that introduces *defacto* randomness, and the epistemic uncertainty of the model fit, both of which can be captured by the model. In Figure 3 (left, middle) we show the 20-80th percentile of the state value of the 2nd component (x_2) of the Lorenz system per point as produced by our model in the shaded regions vs a ground truth trajectory, showing that the trajectory lies within the prediction interval and that it correctly captures that the state may venture to the other arm of the attractor (having larger x_2 value). In Figure 3 (right) we show that the model's uncertainties are calibrated quantitatively, by binning the empirical quantiles of the output samples per point and measuring the rate at which the actual values fall into those quantiles.

5.2. Conditional Samples

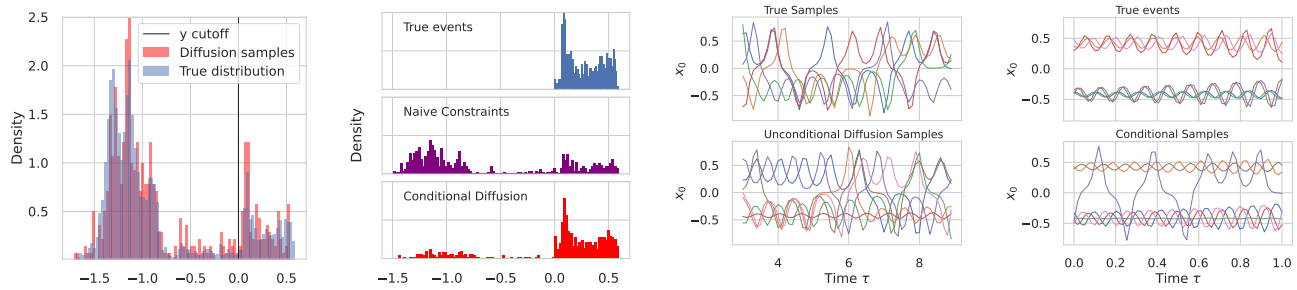
We showcase the ability to sample unlikely and extreme events with our method, by sampling extreme events in the Fitzhugh-Nagumo system described previously. Over the time-horizon considered, the neuron spiking events occurs in only in roughly $1/30$ of the trajectories, which are qualitatively very different from the average trajectories. These unusual trajectories are shown in Figure 1 (right) and Figure 4 (d). We define an event through the quantity $C(x) = \max_\tau(x_1(\tau) + x_2(\tau))/2 - 2.5$, $E = [C(x) > 0]$. The statistics of C for the ground truth unconditional and conditional distributions are shown in Figure 4 (a) and (c) respectively. Note the unusual neuron fires out past value 4, which are also produced occasionally when sampling the trained model. When conditioning on this nonlinear inequality constraint using our method, we can sample directly from this cluster of outliers producing event samples (Figure 4 b) that mirror true events, and match the statistic values (Figure 4 c). We compare our method (Conditional Diffusion in the figure) to a more naive approach (Naive Constraints) of using $p(E|x_t) \approx \Phi(\frac{s_t}{\sigma_t}(C(\hat{x}_0) - y))$ which doesn't make use of the 2nd order Tweedies formula information of the conditional covariance. Unlike our method, this approach is not as effective at sampling from the tail.

Next, we perform the same tail sampling for the Lorenz system. We separate trajectories in a region of the state space where switching between the two arms of the attractor is common vs where it is not. For this purpose we define the nonlinear statistic $C(x) = 0.6 - \|F[x - \bar{x}]\|_1$ where F is the Fourier transform applied to the trajectory time τ and $\|\|_1$ is the 1-norm taken over both the Fourier components and the 3 dimensions of the state, and $\bar{x}$ is just the average value of x over τ. As shown in Figure 5, this statistic separates the distribution into two populations. We condition on this inequality constraint $C(x) > 0$ and we generate samples satisfying the inequality constraint and that do not change arms, as shown in Figure 5. With both systems, the conditional samples preserve the diversity in the distribution, rather than collapsing to a conditional mode as one would have with optimization based methods.

5.3. Convergence of the Gaussian Approximation

Here, we study the accuracy of our approximation $p(C(x_0)|x_t) \approx \mathcal{N}(C(\hat{x}_0), \nabla C^\top \hat{\Sigma} \nabla C)$ which combines Gaussian moment matching on $x_0|x_t$ with a linearization of the constraints around the mode of the Gaussian. We show that our approximation becomes increasingly exact as the time variable and hence the noise goes to 0, and the asymptotics of the error with σ_t are quantified in Appendix B.

Theorem 1. *Suppose C is an analytic function and $\{p(x_t)\}_{t\in[0,1]}$ is collection of smooth density functions with a smooth dependence on t (associated to x_t in equation 3),*



(a) $C(x)$ (b) $C(x)|E$ (c) $x \sim p(x)$ (d) $x \sim p(x|E)$

Figure 5: (**a**) Histograms of statistic values $C(x)$ in the Lorenz system, for trajectories sampled unconditionally from from the dataset and the model. The event describes trajectories which do not cross between the two arms of the strange attractor (in the limited time horizon). Notably the unconditional model distribution captures the outliers. (**b**) Statistic values on samples produced by our method of tail sampling vs actual extreme events vs naive approach to enforce the constraints. (**c**) Unconditional trajectories from the data and model distribution. (**d**) Example event conditional samples compared to actual events. When conditioning on the event defined by the statistic value $E = [C(x) > 0]$, the model is able to conditionally sample from the distribution. Note that while the majority of conditional samples satisfy the event, there are a few stragglers that do not satisfy the event shown by the population the left of $C(x) = 0$ in (b) and the purple trajectory in (d).

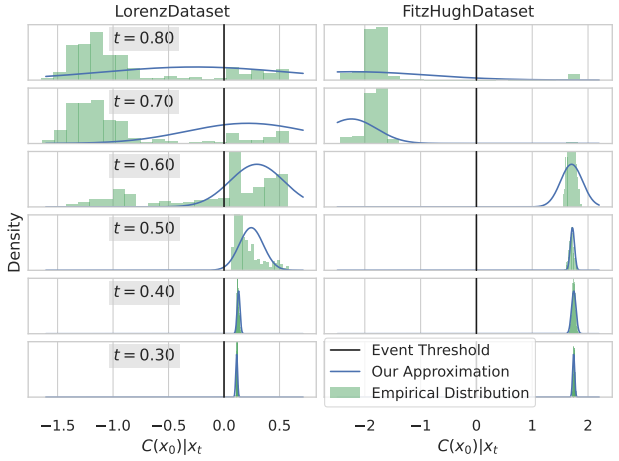


Figure 6: The empirical distribution of the event statistic $C(x_0)$ computed over distinct sample paths given the value x_t at a particular noise level for the events chosen in the two datasets. The linearized moment matching approximation loosely guides the generation at high noise levels, and becomes increasingly accurate as the noise level decreases.

then the random variable $(\nabla C^\top \hat{\Sigma} \nabla C)^{-1/2}\big(C(x_0) - C(\hat{x}_0)\big)$ *(conditioned on* x_t*) converges in distribution to a multivariate normal as* $t \to 0$*,*

$$(\nabla C^\top \hat{\Sigma} \nabla C)^{-1/2}(C(x_0) - C(\hat{x}_0)) \to_d \mathcal{N}(0, I). \quad (16)$$

Proof: See Appendix B.

To better understand the transition from high- to low-noise regimes in terms of the convergence of the approximation, we empirically evaluate the distribution $p(C(x_0)|x_t)$ at different points and at various times between $t = 1$ and $t = 0$ for a trajectory conditioned on the event $C(x_0) > 0$. We compare this empirical distribution from sampling with our linearized moment matching approximation in Figure 6. For high noise levels ($t > 0.5$), the approximation only loosely guides x_t towards satisfying the constraint, however for low noise levels ($t < 0.5$) the approximation becomes increasingly accurate despite the highly nonlinear function C. When the noise level is high, fine details of the event gradients $\nabla_{x_t} \log p(E|x_t)$ are less important since they are drowned out by the noise; however, when these fine details matter later on in the generation the constraint approximation becomes increasingly accurate.

5.4. Computing Marginal Likelihoods of Events

Our method enables computation of the marginal probability $p(E)$ of the event according to the model, taking into account the different ways the event can happen. In Appendix C we investigate estimating these likelihoods when applied to predict whether or not a given initial condition will produce an extreme event $C(x) > 0$ in the time window for the neuron firing in the Fitzhugh-Nagumo system. We compare computing the event likelihood directly sampling from the model, using the conditional likelihoods described in subsection 4.5, as well as with importance sampling. For the methods directly using the conditional likelihoods, the difference in likelihoods $\log p(x_0) - \log p(x_0|E)$ is small enough that the two quantities need to be computed extremely precisely, and small errors can produce large errors in the estimated marginal probability $p(E)$. On the other hand, the model has captured the true event likelihood well as evidenced by direct sampling.

6. Conclusion

In this work, we successfully build diffusion models for the probabilistic modeling of trajectories of chaotic dynamical systems which are able to capture the dynamics of these systems, and more importantly, provide calibrated uncertainty estimates. We have developed a probabilistic approximation with theoretical guarantees that enables conditioning the model on nonlinear equality and inequality constraints without retraining the model. With the approach, we are able to sample directly from the tails of the distribution. We discuss limitations of our method in Appendix J.

While in this work we considered ODEs, the applicability of diffusion models extends far beyond that, and we envision a future where a handful of extremely large diffusion models are employed for spatiotemporal weather prediction. We hope that inference time conditioning capabilities will enable querying the model in different ways, such as predicting and anticipating extreme temperatures or adverse events. As the capabilities of these models grow, and the distributions they model become more multifaceted and diverse, exhaustive direct sampling becomes infeasible and retraining to specialize for a given type of conditioning is prohibitive both in terms of compute and data. Additionally, we hope that marginal likelihood computations can be refined in future work, which can be extremely valuable for estimation of extreme events.

Acknowledgements

We would like to thank Zhong Yi Wan for insightful discussions during the internship.

Marc Finzi and Andrew Gordon Wilson were partially supported by NSF CAREER IIS-2145492, NSF CDSE-MSS 2134216, NSF HDR-2118310, NSF I-DISRE 193471, NIH R01DA048764-01A1, NSF IIS-1910266, NSF 1922658 NRT-HDR, Meta Core Data Science, Google AI Research, BigHat Biosciences, Capital One, and an Amazon Research Award.

References

Asch, A., J. Brady, E., Gallardo, H., Hood, J., Chu, B., and Farazmand, M. Model-assisted deep learning of rare extreme events from partial observations. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 32(4): 043112, 2022.

Bauer, P., Thorpe, A., and Brunet, G. The quiet revolution of numerical weather prediction. *Nature*, 525(7567):47–55, 2015.

Chen, R. T., Rubanova, Y., Bettencourt, J., and Duvenaud, D. K. Neural ordinary differential equations. *Advances in neural information processing systems*, 31, 2018.

Chen, R. T., Behrmann, J., Duvenaud, D. K., and Jacobsen, J.-H. Residual flows for invertible generative modeling. *Advances in Neural Information Processing Systems*, 32, 2019.

Chung, H. and Ye, J. C. Score-based diffusion models for accelerated mri. *Medical Image Analysis*, pp. 102479, 2022.

Chung, H., Kim, J., Mccann, M. T., Klasky, M. L., and Ye, J. C. Diffusion posterior sampling for general noisy inverse problems. *arXiv preprint arXiv:2209.14687*, 2022a.

Chung, H., Sim, B., Ryu, D., and Ye, J. C. Improving diffusion models for inverse problems using manifold constraints. *arXiv preprint arXiv:2206.00941*, 2022b.

Chung, H., Sim, B., and Ye, J. C. Come-closer-diffuse-faster: Accelerating conditional diffusion models for inverse problems through stochastic contraction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12413–12422, 2022c.

Doan, N., Polifke, W., and Magri, L. Short-and long-term predictions of chaotic flows and extreme events: a physics-constrained reservoir computing approach. *Proceedings of the Royal Society A*, 477(2253):20210135, 2021.

Dysthe, K., Krogstad, H. E., and Müller, P. Oceanic rogue waves. *Annu. Rev. Fluid Mech.*, 40:287–310, 2008.

Efron, B. Tweedie's formula and selection bias. *Journal of the American Statistical Association*, 106(496):1602–1614, 2011.

Farazmand, M. and Sapsis, T. P. Extreme events: Mechanisms and prediction. *Applied Mechanics Reviews*, 71 (5), 2019.

Finzi, M., Wang, K. A., and Wilson, A. G. Simplifying hamiltonian and lagrangian neural networks via explicit constraints. *Advances in neural information processing systems*, 33:13880–13889, 2020.

FitzHugh, R. Impulses and physiological states in theoretical models of nerve membrane. *Biophysical journal*, 1 (6):445–466, 1961.

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.

Graikos, A., Malkin, N., Jojic, N., and Samaras, D. Diffusion models as plug-and-play priors. *arXiv preprint arXiv:2206.09012*, 2022.

Grathwohl, W., Chen, R. T., Bettencourt, J., Sutskever, I., and Duvenaud, D. Ffjord: Free-form continuous dynamics for scalable reversible generative models. *arXiv preprint arXiv:1810.01367*, 2018.

Guth, S. and Sapsis, T. P. Machine learning predictors of extreme events occurring in complex dynamical systems. *Entropy*, 21(10):925, 2019.

Hastings, W. K. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57 (1):97–109, 04 1970. ISSN 0006-3444. doi: 10.1093/biomet/57.1.97. URL https://doi.org/10.1093/biomet/57.1.97.

Ho, J., Jain, A., and Abbeel, P. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020.

Hochman, A., Alpert, P., Harpaz, T., Saaroni, H., and Messori, G. A new dynamical systems perspective on atmospheric predictability: Eastern mediterranean weather regimes as a case study. *Science Advances*, 5(6):eaau0936, 2019. doi: 10.1126/sciadv.aau0936. URL https://www.science.org/doi/abs/10.1126/sciadv.aau0936.

Hyvärinen, A. Estimation of non-normalized statistical models by score matching. *Journal of Machine Learning Research*, 6:695–709, 2005.

Janson, S. Normal convergence by higher semiinvariants with applications to sums of dependent random variables and random graphs. *The Annals of Probability*, pp. 305–312, 1988.

Kahn, H. and Harris, T. E. Estimation of particle transmission by random sampling. *National Bureau of Standards applied mathematics series*, 12:27–30, 1951.

Karras, T., Aittala, M., Aila, T., and Laine, S. Elucidating the design space of diffusion-based generative models. *arXiv preprint arXiv:2206.00364*, 2022.

Kingma, D., Salimans, T., Poole, B., and Ho, J. Variational diffusion models. *Advances in neural information processing systems*, 34:21696–21707, 2021.

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

Lai, Z., Mylonas, C., Nagarajaiah, S., and Chatzi, E. Structural identification with physics-informed neural ordinary differential equations. *Journal of Sound and Vibration*, 508:116196, 2021.

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

Lugmayr, A., Danelljan, M., Romero, A., Yu, F., Timofte, R., and Van Gool, L. Repaint: Inpainting using denoising diffusion probabilistic models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11461–11471, 2022.

Meng, C., Song, Y., Song, J., Wu, J., Zhu, J.-Y., and Ermon, S. Sdedit: Image synthesis and editing with stochastic differential equations. *arXiv preprint arXiv:2108.01073*, 2021.

Mosavi, A., Ozturk, P., and Chau, K.-w. Flood prediction using machine learning models: Literature review. *Water*, 10(11):1536, 2018.

Perkins, S. E. and Alexander, L. V. On the measurement of heat waves. *Journal of climate*, 26(13):4500–4517, 2013.

Qi, D. and Majda, A. J. Using machine learning to predict extreme events in complex systems. *Proceedings of the National Academy of Sciences*, 117(1):52–59, 2020. doi: 10.1073/pnas.1917285117. URL https://www.pnas.org/doi/abs/10.1073/pnas.1917285117.

Richardson, L. F. *Weather Prediction by Numerical Process*. Cambridge Mathematical Library. Cambridge University Press, 1 edition, 1922. doi: 10.1017/CBO9780511618291.

Robbins, H. E. An empirical bayes approach to statistics. In *Breakthroughs in statistics*, pp. 388–394. Springer, 1992.

Rosenbluth, M. N. and Rosenbluth, A. W. Monte carlo calculation of the average extension of molecular chains. *The Journal of Chemical Physics*, 23(2):356–359, 1955. doi: 10.1063/1.1741967. URL https://doi.org/10.1063/1.1741967.

Rossky, P. J., Doll, J. D., and Friedman, H. L. Brownian dynamics as smart monte carlo simulation. *The Journal of Chemical Physics*, 69(10):4628–4633, 1978. doi: 10.1063/1.436415. URL https://doi.org/10.1063/1.436415.

Saharia, C., Chan, W., Saxena, S., Li, L., Whang, J., Denton, E., Ghasemipour, S. K. S., Ayan, B. K., Mahdavi, S. S., Lopes, R. G., et al. Photorealistic text-to-image diffusion models with deep language understanding. *arXiv preprint arXiv:2205.11487*, 2022.

Särkkä, S. and Solin, A. *Applied stochastic differential equations*, volume 10. Cambridge University Press, 2019.

Sohl-Dickstein, J., Weiss, E., Maheswaranathan, N., and Ganguli, S. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning*, pp. 2256–2265. PMLR, 2015.

Song, Y. and Ermon, S. Generative modeling by estimating gradients of the data distribution. *Advances in Neural Information Processing Systems*, 32, 2019.

Song, Y., Sohl-Dickstein, J., Kingma, D. P., Kumar, A., Ermon, S., and Poole, B. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020.

Vincent, P. A connection between score matching and denoising autoencoders. *Neural computation*, 23(7):1661–1674, 2011.

Wan, Z. Y., Vlachas, P., Koumoutsakos, P., and Sapsis, T. Data-assisted reduced-order modeling of extreme events in complex dynamical systems. *PloS one*, 13(5): e0197704, 2018.

Yildiz, C., Heinonen, M., and Lahdesmaki, H. Ode2vae: Deep generative second order odes with bayesian neural networks. *Advances in Neural Information Processing Systems*, 32, 2019.

A. Tweedie's Covariance

Consider the noise relation $x = z + \sigma\epsilon$ where $\epsilon \sim \mathcal{N}(0, I)$. We write the Gaussian $x|z$ in exponential family form:

$$p(x|z) = \exp\left[z^\top T(x) - A(z)\right]h(x) \tag{17}$$

where $h(x) = e^{-x^\top x/2\sigma^2}/(2\pi\sigma^2)^{d/2}$, $A(z) = z^\top z/2\sigma^2$, and $T(x) = x/\sigma^2$ is the sufficient statistic.

Using Bayes rule, $p(z|x) = p(x|z)p(z)/p(x)$, we can rewrite $p(z|x)$ and also express it in exponential family form:

$$p(z|x) = \exp\left(z^\top T(x) - A(z)\right)h(x)p(z)/p(x) \tag{18}$$

$$\begin{aligned} p(z|x) &= \exp\left(x^\top z/\sigma^2 - \log\tfrac{p(x)}{h(x)}\right)[p(z)e^{-A(z)}] \\ &= \exp\left(x^\top \bar{T}(z) - \bar{A}(x)\right)\bar{h}(z), \end{aligned} \tag{19}$$

where $\bar{h}(z) = p(z)e^{-A(z)}$, and $\bar{A}(x) = \log\frac{p(x)}{h(x)}$, and $\bar{T}(z) = z/\sigma^2$. Despite the fact that the distribution $p(z)$ is not known, $p(z|x)$ is guaranteed to be in exponential family.

A convenient fact is that for exponential families, $\bar{A}(x)$ is the cumulant generating function for $\bar{T}(z)$, and derivatives produce the cumulants:

$$\mathbb{E}[\bar{T}(z)|x] = \nabla\bar{A}(x) = \nabla\log p(x) - \nabla\log h(x) \tag{20}$$

$$\mathrm{Cov}[\bar{T}(z)|x] = \nabla^2\bar{A}(x) = \nabla^2\log p(x) - \nabla^2\log h(x) \tag{21}$$

and so forth for higher order cumulants (here ∇^2 denotes the hessian).

Plugging in $h(x) = e^{-x^\top x/2\sigma^2}/(2\pi\sigma^2)^{d/2}$ and $\bar{T}(z) = z/\sigma^2$, and moving the σ^2 to the other side, we get

$$\mathbb{E}[z|x] = \sigma^2(\nabla\log p(x) + x/\sigma^2) \tag{22}$$

$$\mathrm{Cov}[z|x] = \sigma^4(\nabla^2\log p(x) + I/\sigma^2) \tag{23}$$

Finally, with the noise relation in the diffusion models $x_t = s_t x_0 + \sigma_t\epsilon$ we can substitute in $z \to s_t x_0$ and $x \to x_t$ to get

$$\mathbb{E}[x_0|x_t] = (x_t + \sigma_t^2\nabla\log p(x_t))/s_t \tag{24}$$

$$\mathrm{Cov}[x_0|x_t] = \tfrac{\sigma_t^2}{s_t^2}(I + \sigma_t^2\nabla^2\log p(x_t)), \tag{25}$$

which proves the relation that we use in the main text.

B. Convergence of the Moment Matching and Linearization Approximations

In this section we show that the linearized moment-matching approximation becomes exact in the limit as the noise scale approaches 0. First consider the moment matching approximation by itself.

Theorem 2. *Suppose that $\{p(x_t)\}_{t\in[0,1]}$ is a family of smooth probability density functions that depends smoothly on t and where x_t is given by equation 3. Then the variable $\hat{\Sigma}^{-1/2}(x_0 - \hat{x}_0)$ (conditioned on x_t) converges in distribution to a standard multivariate Gaussian, in the limit as $t \to 0$ (or equivalently as $\sigma_t \to 0$):*

$$\hat{\Sigma}^{-1/2}(x_0 - \hat{x}_0) \to_d \mathcal{N}(0, I), \tag{26}$$

where $\hat{x}_0 = \mathbb{E}[x_0|x_t]$ and $\hat{\Sigma} = \mathrm{Cov}[x_0|x_t]$.

Proof. To determine the convergence of $u := \hat{\Sigma}^{-1/2}(x_0 - \hat{x}_0)$, it is sufficient to show that the expectation converges to 0, the covariance converges to I, and the higher order moments converge to 0 (Janson, 1988).

To start, we can derive the higher order *cumulants* of the conditional distribution $x_0|x_t$ by taking additional derivatives of the cumulant generating function $\bar{A}$.

Applying the same substitution as in Appendix A, we obtain the n-th order cumulant for $n \geq 3$ given by

$$k_n(x_0|x_t) := \frac{\sigma_t^{2n}}{s_t^n}\nabla^{\otimes n}\log p(x_t), \tag{27}$$

where $\nabla^{\otimes n} = \underbrace{\nabla \otimes \cdots \otimes \nabla}_{n}$ and $\otimes$ is the tensor product.

Now, consider the cumulants of the random variable $u := \hat{\Sigma}^{-1/2}(x_0 - \hat{x}_0)$, which has 0 mean and a constant scale.

$$k_1(u) = \mathbb{E}[u] = 0 \tag{28}$$

$$k_2(u) = \mathrm{Cov}[u] = \hat{\Sigma}^{-1/2}\hat{\Sigma}\hat{\Sigma}^{-1/2} = I \tag{29}$$

$$k_n(u) = \sigma_t^n(\tfrac{\sigma_t}{s_t}\hat{\Sigma}^{-1/2}\nabla)^{\otimes n}\log p(x_t) \quad \text{for } n \geq 3 \tag{30}$$

Investigating the limiting behavior of $k_n(u)$, we can make use of two important facts: firstly

$$\lim_{t\to 0}\tfrac{\sigma_t}{s_t}\hat{\Sigma}^{-1/2} = \left(\lim_{\sigma_t\to 0} I + \sigma_t^2\nabla^2\log p(x_t)\right)^{-1/2} = I$$

and secondly $\lim_{t\to 0}\nabla^{\otimes n}\log p(x_t) = \nabla^{\otimes n}\log p(x_0)$ since $\log p(x_0)$ is smooth.

Therefore we see that $\lim_{t\to 0} k_n(u) = \nabla^{\otimes n}\log p(x_0)(\lim_{t\to 0}\sigma_t^n) = 0$, the higher order cumulants converge to 0 at a rate of σ_t^n, whereas the mean and variance are fixed. Therefore according to (Janson, 1988), u converges in distribution to a multivariate normal $u \to_d \mathcal{N}(0, I)$. □

Next we consider the full approximation, including the linearization of the constraint as described in subsection 4.3. The constraint is linearized from the Taylor expansion

$$\begin{aligned} C(x_0) &= C(\hat{x}_0) \\ &\quad + \nabla C^\top(x_0 - \hat{x}_0) + (x_0 - \hat{x}_0)^\top\nabla^2 C(x_0 - \hat{x}_0)/2, \end{aligned}$$

(with additional higher order terms omitted for brevity). For simplicity we have used notation for scalar function C, but the result below holds for vector C analogously. Now we state the convergence result for our approximation

$$p(C(x_0)|x_t) \approx \mathcal{N}\big(C(\hat{x}_0), \nabla C^\top \hat{\Sigma} \nabla C\big). \tag{31}$$

Theorem 3. *Suppose that C is analytic and $p(x_t)$ is a smooth function then as $\sigma_t \to 0$, the random variable $(\nabla C^\top \hat{\Sigma} \nabla C)^{-1/2}\big(C(x_0) - C(\hat{x}_0)\big)$ converges in distribution to a multivariate normal:*

$$(\nabla C^\top \hat{\Sigma} \nabla C)^{-1/2}(C(x_0) - C(\hat{x}_0)) \to_d \mathcal{N}(0, I). \tag{32}$$

Proof. To start, we note that $\lim_{\sigma_t \to 0} \frac{s_t^2}{\sigma_t^2} \nabla C^\top \hat{\Sigma} \nabla C = \nabla C^\top \nabla C$, so if we can prove that for the random variable $v = \frac{s_t}{\sigma_t}\big(C(x_0) - C(\hat{x}_0)\big)$ converges to $\mathcal{N}(0, \nabla C^\top \nabla C)$ then we have proven the claim.

For convenience, define $A = (s_t/\sigma_t)\hat{\Sigma}^{1/2}$, keeping in mind that $\lim_{\sigma_t \to 0} A = I$

Recalling the random variable $u = \hat{\Sigma}^{-1/2}(x_0 - \hat{x}_0)$, we can rewrite v using the Taylor series as

$$v = \nabla C^\top A u + \tfrac{1}{2}\big(\tfrac{\sigma_t}{s_t}\big) u^\top A^\top \nabla^2 C A u + O\big(\big(\tfrac{\sigma_t}{s_t}\big)^2\big). \tag{33}$$

Notably, A, u, $u^\top \nabla^2 C u$, and higher order terms converge to a fixed scale as $\sigma \to 0$ (since C is assumed to be twice continuously differentiable and u converges to a normal).

Writing out the cumulants of this random variable we see a similar pattern as before:

$$\begin{aligned}
k_1(v) &= \mathbb{E}[v] = O\big(\big(\tfrac{\sigma_t}{s_t}\big)\big) \\
k_2(v) &= \mathrm{Cov}[v] = \nabla C^\top A \mathrm{Cov}[u] A^\top \nabla C + O\big(\big(\tfrac{\sigma_t}{s_t}\big)\big) \\
k_n(v) &= \sigma_t^n (\nabla C^\top A \nabla)^{\otimes n} \log p(x_t) + O\big(\big(\tfrac{\sigma_t}{s_t}\big)\big) \quad \text{for } n \geq 3.
\end{aligned}$$

Note that $\lim_{\sigma_t \to 0} (\nabla C^\top A \nabla)^{\otimes n} \log p(x_t) = (\nabla C^\top \nabla)^{\otimes n} \log p(x_0)$ since p is smooth. In the limit as $t \to 0$, the cumulants become

$$\begin{aligned}
\lim_{t \to 0} k_1(v) &= 0 \\
\lim_{t \to 0} k_2(v) &= \nabla C^\top \nabla C \\
\lim_{t \to 0} k_n(v) &= 0 \quad \text{for } n \geq 3.
\end{aligned}$$

Therefore, according to (Janson, 1988), v converges in distribution to $\mathcal{N}(0, \nabla C^\top \nabla C)$. □

In contrast with the mere moment matching Gaussian approximation the convergence rate is considerably slower however, with higher order cumulants only decaying as $O\big(\big(\frac{\sigma_t}{s_t}\big)\big)$ and depending on the smoothness of C. Nevertheless, as the noise scale gets smaller, the combined linearization and moment matching approximation (informally stated)

$$p(C(x_0)|x_t) \underset{t \to 0}{\to} \mathcal{N}\big(C(\hat{x}_0), \nabla C^\top \hat{\Sigma} \nabla C\big) \tag{34}$$

becomes exact.

C. Marginal Likelihood Estimation

In estimating the marginal event likelihood $p(E)$, there are multiple ways of extracting this quantity from the model. The simplest, and least scalable to extremely low likelihood events is extensively sample from the model and compute the fraction which satisfy the event $p(E) = \mathbb{E}_{x \sim p(x)}[\mathbb{1}[x \in E]]$. Second is to use the method we introduce in Equation 13, and compute the average increase in the log likelihood

$$p(E) = \exp\left(\mathbb{E}_{x \sim p(x|E)}[\log p(x) - \log p(x|E)]\right). \tag{35}$$

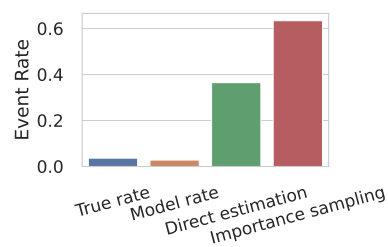


Figure 7: Calibration of different estimators of the marginal event likelihood for Fitzhugh-Nagumo extreme event. True rate is computed over sampling from the original dataset and model rate is computed over samples from the diffusion model. Direct estimation uses Equation 35 and importance sampling uses the importance sampled version of the estimator, both are computed using 3 samples from $x \sim p(x|E)$. Likelihood based estimators are not well calibrated to the true rate of 1/30, but the model rate is.

The potential downside of this approach is that its validity depends on the quality of the approximation used to compute $p(x|E)$. Alternatively, we can instead drop this requirement and view $p(x|E) = q(x)$ merely as a strong proposal distribution for importance sampling. Sampling from q, $p(E) = \int \mathbb{1}[x \in E] p(x) dx = \mathbb{E}_{x \sim p(x|E)}[p(x)/p(x|E)]$ giving a very closely related estimator to Equation 35, but that has no requirements on $p(x|E)$ other than it covers the event space. In Figure 7, we evaluate the predictions of these three estimators on the FitzHugh-Nagumo neuron spiking event, and compare to the ground truth event rate.

While the sampling based method $p(E) = \mathbb{E}_{x \sim p(x)}[\mathbb{1}[x \in E]]$ (Model rate in the figure) well approximates the true model event rate, unfortunately neither of the likelihood based methods (Direct estimation and

Importance sampling) for estimate $p(E)$ appear to be calibrated when estimated over a small number of samples $p(x_0|E)$, even though theoretically they should produce the value consistent with the model distribution. We suspect this has to do with the accumulation of numerical errors in the estimation of the Jacobian log determinants for $p(x|E)$ which needs to be estimated very precisely because $\log p(x|E)$ is typically on the order of 1000 and should differ from $\log p(x)$ only by a few tenths of a percent.

D. Probabilistic Origin of Constraint Projection

In this section we investigate how the additional constraint projection steps used in Chung et al. (2022c;b) can arise in our probabilistic framework when applied to linear constraints, but performing one additional approximation. Consider the goal of imposing the set of linear constraints $Cx = y$ onto samples from the diffusion model for a given constraint matrix $C \in \mathbb{R}^{r\times d}$ and $y \in \mathbb{R}^r$. We seek to sample $x \sim p(x|Cx = y)$ using a diffusion models.

Without loss of generality, we can orthogonalize the linear constraints. Decomposing C with the SVD into the right nullspace of C and its complement:

$$C = \begin{bmatrix} U & V \end{bmatrix} \begin{bmatrix} \Sigma & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} Q^\top \\ P^\top \end{bmatrix}. \tag{36}$$

Here the matrix $Q \in \mathbb{R}^{d\times r}$ and $P \in \mathbb{R}^{d\times d-r}$ correspond to the subspace of x that is determined by the constraint, and the orthogonal complement which is linearly independent of the constraint. We can now rewrite the constraint $Cx = y$ as $U\Sigma Q^\top x = y$ or equivalently: $Q^\top x = u$ where $u = \Sigma^{-1}U^\top y$. We can decompose x in these two components: its projection onto the row space of C and onto its orthogonal complement, $x = Qu + Pv$, where $v := P^\top x$. In contrast with the derivation in section 4, we will split up the diffusion process into these two subspaces.

In order to sample conditionally, we need the conditional scores

$$\nabla_{x_t} \log p(x_t|Cx_0 = y) = \nabla_{x_t} \log p(x_t|Q^\top x_0 = u_0). \tag{37}$$

We can express the gradients with respect to x_t as the sum of the projected gradients with respect to u_t and with respect to v_t using the chain rule: $\nabla_{x_t} = Q\nabla_{u_t} + P\nabla_{v_t}$. To see this, let $O = [Q, P] \in \mathbb{R}^{d\times d}$ be the concatenation of Q and P that forms a full rank orthogonal matrix ($O^\top O = I$). Let

$$z_t := \begin{bmatrix} u_t \\ v_t \end{bmatrix} = O^\top x_t, \tag{38}$$

which can be inverted to get $Oz_t = x_t$. Applying the chain rule, it follows that

$$\nabla_{x_t} = O\nabla_{z_t} = Q\nabla_{u_t} + P\nabla_{v_t}. \tag{39}$$

Therefore, we can split up the conditional scores into gradients with respect to the two variables u_t and v_t:

$$\nabla_{x_t} \log p(x_t|u_0) = Q\nabla_{u_t} \log p(x_t|u_0) \tag{40}$$
$$+ P\nabla_{v_t} \log p(x_t|u_0). \tag{41}$$

So far, this equation merely expresses section 4 in a different form. The additional approximation needed to produce the explicit constraint projection is to replace or approximate $\nabla_{u_t} \log p(x_t|u_0)$ with $\nabla_{u_t} \log p(u_t|u_0)$. Despite being closely related they are different: $\nabla_{u_t} \log p(x_t|u_0) = \nabla_{u_t} \log p(u_t|u_0) + \nabla_{u_t} \log p(v_t|u_t, u_0)$. If we use this approximation, then this first term can be easily computed from the forward noising process $u_t \sim \mathcal{N}(s_t u_0, \sigma_t^2 I)$ and therefore

$$\nabla_{u_t} \log p(u_t|u_0) = \frac{s_t u_0 - u_t}{\sigma_t^2}. \tag{42}$$

For the second term, one may verify from Bayes rule that

$$p(x_t|u_0) = p(v_t|u_t)p(u_0|x_t)p(u_t|u_0)/p(u_0|u_t) \tag{43}$$

$$\nabla_{v_t} \log p(x_t|u_0) = \nabla_{v_t} \log p(v_t|u_t) + \nabla_{v_t} \log p(u_0|x_t), \tag{44}$$

as all but these two terms do not depend on v_t. The first term can be identified as simply the scores of the unconditional network projected onto the unknown subspace:

$$\nabla_{v_t} \log p(v_t|u_t) = \nabla_{v_t} \log p(v_t, u_t) = P^\top s_\theta(x_t, t). \tag{45}$$

Performing moment matching on $p(u_0|x_t)$ like before, we can approximate the distribution with a multivariate Gaussian which shares its true mean and covariance:

$$p(u_0|x_t) \approx \mathcal{N}(Q^\top \hat{x}_0, Q^\top \hat{\Sigma} Q). \tag{46}$$

With the combination of the two approximations, the conditional scores become:

$$\nabla_{x_t} \log p(x_t|u_0) = Q\big[s_t u_0 - u_t\big]/\sigma_t^2 + P\nabla_{v_t} \log p(x_t|u_0) \tag{47}$$

where

$$\nabla_{v_t} \log p(x_t|u_0) \approx P^\top s_\theta(x_t, t) + \nabla_{v_t} \log \mathcal{N}(u_0; Q^\top \hat{x}_0, Q^\top \hat{\Sigma} Q) \tag{48}$$

Substituting $\hat{\Sigma} = 2\alpha I$ to match Chung et al. (2022b) (which neglects the scaling with σ_t/s_t), and applying the

chain rule to convert derivatives with respect to v_t to derivatives with respect to x_t,

$$\nabla_{v_t} \log \mathcal{N}(u_0; Q^\top \hat{x}_0, 2\alpha I) = -\alpha P^\top \nabla_{x_t} \|Q^\top \hat{x}_0 - u_0\|^2. \tag{49}$$

Finally, assembling the terms, we have

$$\begin{aligned}\bar{s}(x_t, t) = Q\big[s_t u_0 - u_t\big]/\sigma_t^2 \\ + PP^\top[s(x_t, t) - \alpha \nabla_{x_t} \|Q^\top \hat{x}_0 - u_0\|^2],\end{aligned} \tag{50}$$

where $\bar{s}(x_t, t)$ denotes the approximation for $\nabla_{x_t} \log p(x_t|u_0)$ when using these additional assumptions. The three terms can be understood as: (1) the analytically known diffusion in the known subspace, (2) the projection of the unconditional score function to the unknown subspace, and (3) the Gaussian correction term projected into the unknown subspace.

We can identify each of the terms in this equation directly with equations 7 and 8 in Chung et al. (2022b) where the terms are renamed as follows. Their variable names correspond to the following quantities: $y \leftarrow u_0 = R^{-T} y$, $P \leftarrow Q^\top$, and $A = I - P^\top P \leftarrow I - QQ^\top = PP^\top$. While here we have the results expressed in terms of the score function rather than a denoising step, the two are consistent. If framed as a denoising step, $Q\big[s_t u_0 - u_t\big]/\sigma_t^2$ would become the b term: $b \leftarrow Q\epsilon$ where $\epsilon \sim \mathcal{N}(s_t u_0, \sigma_t^2 I)$. The only difference is the W matrix which is not fully specified in their method (they only provide a couple examples where it is suggested what it should be).

Therefore we see that while the constraint projections of Chung et al. (2022b) is not *equivalent* to our derivation in section 4 for the linear case with identity covariance, it naturally arises when making the additional replacement $\nabla_{u_t} \log p(x_t|u_0) \mapsto \nabla_{u_t} \log p(u_t|u_0)$. Intuitively speaking, this additional assumption assumes that we can evolve the noised version of the known values u_t without considering the unknown values.

E. Conditional Score Convergence in Continuous Time

When considering equality constraints, both Manifold Constrained Gradients (MCG) (Chung et al., 2022b) (linear constraints only) and Diffusion Posterior Sampling (DPS) (Chung et al., 2022a) (nonlinear equality constraints with or without noise) can be used for inference time conditional sampling. While these two approaches are effective in this setting, they are not well suited to the continuous time ODE formulation, a requirement for computing likelihoods through the change of variables formula.

The difference can be summarized on a one dimensional constraint $c^\top x = y$, for which the approach of Chung et al. (2022a) would give the conditional scores as:

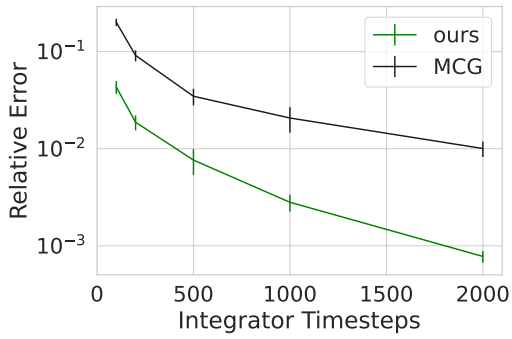


Figure 8: Convergence of conditional samples with varying numbers of ODE integrator steps. Due to the missing scaling factors of MCG (Chung et al., 2022b), the method requires many more integration steps to converge.

$$\nabla_{x_t} \log p(x_t|y) = s_\theta(x_t, t) - \alpha \nabla_{x_t} \|c^\top \hat{x}_0 - y\|^2, \tag{51}$$

whereas our method yields the scores

$$\nabla_{x_t} \log p(x_t|y) = s_\theta(x_t, t) - \nabla_{x_t} \frac{\|c^\top \hat{x}_0 - y\|^2}{\sqrt{c^\top \hat{\Sigma} c}}. \tag{52}$$

The $\hat{\Sigma}$ matrix scales as σ_t^2/s_t^2 which varies by many orders of magnitude. While MCG and DPS include the tuneable parameter α, it has one fixed value, and cannot match the σ_t^2/s_t^2 scale inside $\hat{\Sigma}$. As a result, when incorporating Equation 51 into adaptive step size integrators, the scales at different times of the integration will either be too small or too large in comparison to $s_\theta(x_t, t)$. We see this in practice that with MCG or DPS and an adaptive step size integrator, the integration fails to converge.

In Figure 8 we demonstrate that this is a problem even for fixed step size integrators too, by evaluating the convergence of the an ODE integrated trajectory sampling from a linear equality constraint with $c = \mathbf{1}/d$ and $y = 0.08$ on the FitzHugh model. We measure the relative error between the conditional sample produced with 4000 uniformly spaced ODE integrator steps and the conditional sample produced with smaller numbers of integrator steps. As shown in the figure, our conditional score functions lead to a much more rapidly converging solution.

F. Dataset Construction

For all datasets, we generate 4000 trajectories discretized to 60 timesteps to be used for training, and hold out an additional 500 trajectories for testing.

Lorenz Attractor We generate trajectories from the clas-

sical dynamics

$$\dot{x} = 10(y - x) \tag{53}$$
$$\dot{y} = x(28 - z) - y \tag{54}$$
$$\dot{z} = xy - (8/3)z \tag{55}$$

Writing these components as the vector $\mathbf{x} = [x, y, z]$, these dynamics can be written as $\dot{\mathbf{x}} = F(\mathbf{x})$ for the above function F. Since these dynamics range over the scale $(-60, 60)$ we rescale the dynamics by defining a rescaled version of the Lorenz dynamics: $\tilde{F}(\mathbf{x}) = F(20\mathbf{x})/20$, which preserves the dynamics but scales down the values into the range $(-3, 3)$ which makes the learning of the diffusion model and Neural ODE more stable. We sample the initial conditions from the standard normal $\mathbf{x}(0) \sim \mathcal{N}(0, I)$, we integrate for a total of 10 seconds and then discard the first 3 seconds as burn-in to approximately reach the stationary distribution. The trajectories are then discretized to 60 evenly spaced timesteps when training the diffusion model.

FitzHugh Nagumo We follow Farazmand & Sapsis (2019) in their choice of parameters to produce the rare events. We sample initial conditions from $\mathbf{x}(0) \sim .2\mathcal{N}(0, I)$ on the 4-dimensional system. We integrate for 4000 units of time and discard the first 1500 units of time for burn-in, which we find to be sufficient for the distribution to approximately reach the stationary distribution.

Double Pendulum We use Hamiltonian from Finzi et al. (2020) for the double pendulum in angular coordinates, and then integrate the Hamiltonian dynamics, with a final post-processing step of converting the canonical momenta back into angular velocities. We set the mass and length parameters to 1 for simplicity and integrate for $\tau = 30$ seconds. We sample the initial from Gaussians. For the angles from $\theta(0)$ we use standardard deviation $\sigma = 1$ and the initial momenta we use $\sigma = .2$ for p_{θ_1} and $\sigma = .3$ for p_{θ_2} .

When fitting the system with the diffusion model, we only proved the angle values, and we first embed the two angles into the plane using $\cos\theta_i$ and $\sin\theta_i$ for the two angles $i = 1, 2$. We performed this additional featurization since some trajectories wrap the angles around many times, and the large angle values can cause problems. For the Neural ODE, we trained it in the usual way feeding in both θ_i and $\dot{\theta}_i$ as done in Finzi et al. (2020).

When training the NeuralODE (Chen et al., 2018), for each of the systems, we chunk the 4000 length 60 trajectories into a total of 24000 chunks of length 10, which is a standard practice to improve the stability and efficiency of training (see e.g. (Finzi et al., 2020). We train using the L_1 loss on the prediction error on the 10 evaluation points for each trajectory. The NeuralODE uses a simple 3-layer MLP with swish nonlinearities and 128 hidden units.

G. Training and Hyperparameters

When training, we sample diffusion times t on a shifted grid following Kingma et al. (2021) for reduced training times. We optimize the score matching loss with ADAM (Kingma & Ba, 2014) for 10000 epochs with lr 10^{-4} and bs 500. We use the variance exploding schedule for σ_t and s_t, as described in Song et al. (2020). We evaluate all models using the exponential moving average of the parameters at the final epoch, where the EMA period is 2000 epochs.

H. Model

We use a convolutional UNet architecture similar to Saharia et al. (2022) but scaled down, with the self attention layers removed (we did not find them helpful at this scale), with a modified embedding to handle continuous input times (the method used in Song et al. (2020), and with 2D convolutions replaced with 1D convolutions. At a high level, the architecture can be summarized with the following table with value $c = 32$, and d represents the input and output dimension.

Convolutional UNet Architecture:	
ResBlock(c)	x4
Downsample(2)	
ResBlock($2c$)	x8
Downsample(2)	
ResBlock($4c$)	x8
SkipResBlock($4c$)	x8
Upsample(2)	
SkipResBlock($2c$)	x8
Upsample(2)	
SkipResBlock(c)	x4
Conv(128)	
Conv(d)	

The resblock follows the standard structure, but with GroupNorm instead of BatchNorm, using swish nonlinearities, and 1D convolutions. SkipResBlocks utilize skip

ResBlock(c):
GroupNorm(groups=c//4)
Swish
Conv(channels=3, ksize=3)
GroupNorm(groups=c//4)
Swish
Conv(channels=3, ksize=3)
SkipConnection

connections with the corresponding residual block in the downsampling portion of the network, as done in typical diffusion UNets such as in Karras et al. (2022).

For gradients of the Gaussian CDF function, we instead

use the logit approximation $\Phi(z) \approx \sigma(1.6z)$ which is more numerically stable.

I. Computing Likelihoods

While the two probabilities $p(x_0)$ and $p(x_0|E)$ can in principle be computed by integrating the probability flow ODEs (Equation 14) forwards in time with the continuous change of variables formula used in FFJORD (Grathwohl et al., 2018; Song et al., 2020), the variance of the likelihood estimator is large (requiring a large number of probe variables) and the integration times can be very long. The large variance is typically not a problem when averaging to form the average log likelihoods of the entire dataset (which reduces the variance) and also because only a small number of significant digits are required for the metrics. On the other hand, for our purposes where we need to compute the log likelihood on a single data point very precisely, the estimator variance is too large.

Instead of using the continuous change of variables approach, we instead use a fixed timestep 2nd order Heun integrator to control the compute time, and then consider this integrator just as some deterministic and invertible function $x_{t+\Delta t} = H(x_t)$. While one cannot apply the continuous change of variables formula with fixed timesteps, we can instead simply compute the Jacobian of the ODE integrator. With this Jacobian, we can compute the likelihoods exactly without sampling and even when the timesteps are large using the relation:

$$\log p_{t=0}(x_0|E) = \log p_{t=1}(x_1) + \sum_i \log \det DH(x_{t_i}), \tag{56}$$

where $p(x_1) = \mathcal{N}(0, \sigma_1^2)$. We compute the Jacobians exactly which removes the need for sampling, however it's possible to replace this computation with Jacobian vector products even for the discrete trajectory by using the matrix logarithm expansion and Russian roulette estimator used for Residual Flows (Chen et al., 2019).

J. Limitations

We introduced a general and principled method for sampling conditionally on user defined events according to nonlinear equality and inequality constraints. However, our method has several limitations.

Scope: Our method was designed for deterministic events, and while it is easy to extend to noisy measurements so long as they have Gaussian observation noise, for non Gaussian observation noise our approximation will not be valid. Furthermore, more complex set based event constraints (such as the values being contained in a given region) may not be easily expressed as equality or inequality constraints that are supported by our method.

Computational cost: Our second order approximation requires computing Jacobian vector products with the diffusion score function which can be expensive. If this cost is prohibitive, we recommend using the cruder but still principled approximation

$$p(C(x_0)|x_t) \approx \mathcal{N}(C(\hat{x}_0), (\sigma_t/s_t)^2 \nabla C^\top \nabla C), \tag{57}$$

when using our method. However, the costs of this approximation are still quadratic in the number of constraints r. In situations where the constraints are very high dimensional and even computing $\nabla C^\top \nabla C$ is not possible, we recommend falling back to the diagonal approximation $p(C(x_0)|x_t) \approx \mathcal{N}(C(\hat{x}_0), \lambda(\sigma_t/s_t)^2 I)$, a version of Chung et al. (2022a) requiring the tunable λ parameter.

Likelihood evaluation is even more expensive, requiring computation of the Jacobian log determinant of the ODE integration step. This cost is $O((md)^3)$ and is prohibitive for large signals such as images. In future work this can be addressed such as by using the Russian roulette estimator from Chen et al. (2019) which will reduce the cost to $O(md)$. Finally, the marginal likelihood computation $p(E)$ via the likelihoods is difficult to estimate accurately due to the differencing of two similar values, we hope this can be addressed in future work.