A Continuous Time Framework for Discrete Denoising Models

Andrew Campbell[1] **Joe Benton**[1] **Valentin De Bortoli**[2]

Tom Rainforth[1] **George Deligiannidis**[1] **Arnaud Doucet**[1]

[1]Department of Statistics, University of Oxford, UK [2]CNRS ENS Ulm, Paris, France
{ campbell, benton, rainforth, deligian, doucet}@stats.ox.ac.uk
valentin.debortoli@gmail.com

Abstract

We provide the first complete continuous time framework for denoising diffusion models of discrete data. This is achieved by formulating the forward noising process and corresponding reverse time generative process as Continuous Time Markov Chains (CTMCs). The model can be efficiently trained using a continuous time version of the ELBO. We simulate the high dimensional CTMC using techniques developed in chemical physics and exploit our continuous time framework to derive high performance samplers that we show can outperform discrete time methods for discrete data. The continuous time treatment also enables us to derive a novel theoretical result bounding the error between the generated sample distribution and the true data distribution.

1 Introduction

Diffusion/score-based/denoising models [1, 2, 3, 4] are a popular class of generative models that achieve state-of-the-art sample quality with good coverage of the data distribution [5] all whilst using a stable, non-adversarial, simple to implement training objective. The general framework is to define a forward noising process that takes in data and gradually corrupts it until the data distribution is transformed into a simple distribution that is easy to sample. The model then learns to reverse this process by learning the logarithmic gradient of the noised marginal distributions known as the score.

Most previous work on denoising models operates on a continuous state space. However, there are many problems for which the data we would like to model is discrete. This occurs, for example, in text, segmentation maps, categorical features, discrete latent spaces, and the direct 8-bit representation of images. Previous work has tried to realize the benefits of the denoising framework on discrete data problems, with promising initial results [6, 7, 8, 9, 10, 11, 12, 13].

All of these previous approaches train and sample the model in discrete *time*. Unfortunately, working in discrete time has notable drawbacks. It generally forces the user to pick a partition of the process at training time and the model only learns to denoise at these fixed time points. Due to the fixed partition, we are then limited to a simple ancestral sampling strategy. In continuous time, the model instead learns to denoise for any arbitrary time point in the process. This complete specification of the reverse process enables much greater flexibility in defining the reverse sampling scheme. For example, in continuous state spaces, continuous time samplers that greatly reduce the sampling time have been devised [14, 15, 16, 17] as well as ones that improve sample quality [4, 18]. The continuous time interpretation has also enabled the derivation of interesting theoretical properties such as error bounds [19] in continuous state spaces.

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

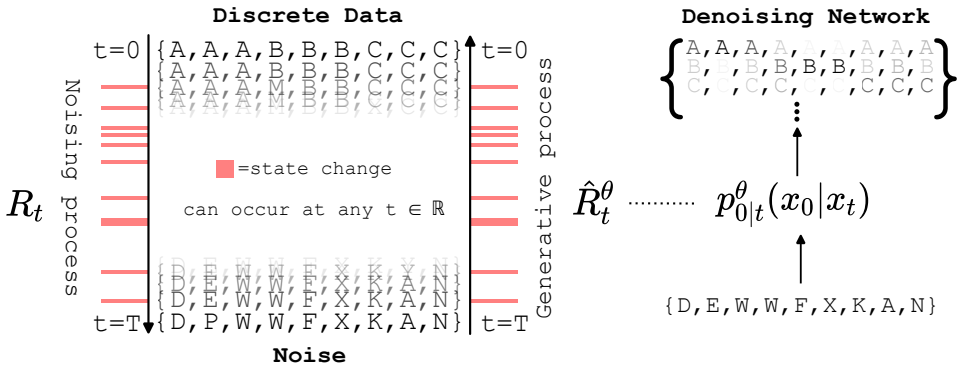

Figure 1: The forward noising process corrupts data according to R_t, the rate of corruption events at time t. The noising process' time reversal gives the generative process which is defined through $\hat{R}_t^\theta$, the rate of generative events at time t. $\hat{R}_t^\theta$ is parameterized through the denoising network, $p_{0|t}^\theta(x_0|x_t)$, which outputs categorical probabilities over clean x_0 values conditioned on a noisy x_t.

To allow these benefits to be exploited for discrete state spaces as well, we formulate a continuous time framework for discrete denoising models. Specifically, our contributions are as follows. We formulate the forward noising process as a Continuous Time Markov Chain (CTMC) and identify the generative CTMC that is the time-reversal of this process. We then bound the log likelihood of the generated data distribution, giving a continuous time equivalent of the ELBO that can be used for efficient training of a parametric approximation to the true generative reverse process. To efficiently simulate the parametric reverse process, we leverage tau-leaping [20] and propose a novel predictor-corrector type scheme that can be used to improve simulation accuracy. The continuous time framework allows us to derive a bound on the error between the true data distribution and the samples generated from the approximate reverse process simulated with tau-leaping. Finally, we demonstrate our proposed method on the generative modeling of images from the CIFAR-10 dataset and monophonic music sequences. Notably, we find our tau-leaping with predictor-corrector sampler can provide higher quality CIFAR10 samples than previous discrete time discrete state approaches, further closing the performance gap between when images are modeled as discrete data or as continuous data.

Proofs for all propositions and theorems are given in the Appendix.

2 Background on Discrete Denoising Models

In the discrete time, discrete state space case, we aim to model discrete data $x_0 \in \mathcal{X}$ with finite cardinality $S = |\mathcal{X}|$. We assume $x_0 \sim p_{\text{data}}(x_0)$ for some discrete data distribution $p_{\text{data}}(x_0)$. We define a forward noising process that transforms $p_{\text{data}}(x_0)$ to some distribution $q_K(x_K)$ that closely approximates an easy to sample distribution $p_{\text{ref}}(x_K)$. This is done by defining forward kernels $q_{k+1|k}(x_{k+1}|x_k)$ that all admit p_{ref} as a stationary distribution and mix reasonably quickly. For example, one can use a simple uniform kernel [6, 8], $q_{k+1|k}(x_{k+1}|x_k) = \delta_{x_{k+1},x_k}(1-\beta) + (1 - \delta_{x_{k+1},x_k})\beta/(S-1)$ where δ is a Kronecker delta. The corresponding p_{ref} is the uniform distribution over all states. Other choices include: an absorbing state kernel—where for each state there is a small probability that it transitions to some absorbing state—or a discretized Gaussian kernel—where only transitions to nearby states have significant probability (valid for spaces with ordinal structure) [8].

After defining $q_{k+1|k}$, we have a forward joint decomposition as follows

$$q_{0:K}(x_{0:K}) = p_{\text{data}}(x_0) \prod_{k=0}^{K-1} q_{k+1|k}(x_{k+1}|x_k).$$

The joint distribution $q_{0:K}(x_{0:K})$ also admits a reverse decomposition:

$$q_{0:K}(x_{0:K}) = q_K(x_K) \prod_{k=0}^{K-1} q_{k|k+1}(x_k|x_{k+1}) \quad \text{where} \quad q_{k|k+1}(x_k|x_{k+1}) = \frac{q_{k+1|k}(x_{k+1}|x_k) q_k(x_k)}{q_{k+1}(x_{k+1})}.$$

Here $q_k(x_k)$ denotes the marginal of $q_{0:K}(x_{0:K})$ at time k. If one had access to $q_{k|k+1}$ and could sample $q_K(x_K)$ exactly, then samples from $p_{\text{data}}(x_0)$ could be produced by first sampling $q_K(x_K)$ and then ancestrally sampling the $q_{k|k+1}$ reverse kernels.

However, in practice, $q_{k|k+1}$ is intractable and needs to be approximated with a parametric reverse kernel, $p^\theta_{k|k+1}$. This kernel is commonly defined through the analytic $q_{k|k+1,0}$ distribution and a parametric 'denoising' model $p^\theta_{0|k+1}$ [6, 8],

$$\begin{aligned} p^\theta_{k|k+1}(x_k|x_{k+1}) &\triangleq \textstyle\sum_{x_0} q_{k|k+1,0}(x_k|x_{k+1}, x_0) p^\theta_{0|k+1}(x_0|x_{k+1}) \\ &= q_{k+1|k}(x_{k+1}|x_k) \textstyle\sum_{x_0} \frac{q_{k|0}(x_k|x_0)}{q_{k+1|0}(x_{k+1}|x_0)} p^\theta_{0|k+1}(x_0|x_{k+1}). \end{aligned} \tag{1}$$

Though $q_K(x_K)$ is also intractable, for large K we can reliably approximate it with $p_{\text{ref}}(x_K)$. Note that the faster the transitions mix, the more accurate this approximation becomes. Approximate samples from $p_{\text{data}}(x_0)$ can then be obtained by sampling the generative joint distribution

$$p^\theta_{0:K}(x_{0:K}) = p_{\text{ref}}(x_K) \textstyle\prod_{k=0}^{K-1} p^\theta_{k|k+1}(x_k|x_{k+1}),$$

where θ is trained through minimizing the negative discrete time (DT) ELBO which is an upper bound on the negative model log-likelihood

$$\mathbb{E}_{p_{\text{data}}(x_0)} \left[-\log p^\theta_0(x_0) \right] \leq \mathbb{E}_{q_{0:K}(x_{0:K})} \left[-\log \frac{p^\theta_{0:K}(x_{0:K})}{q_{1:K|0}(x_{1:K}|x_0)} \right] = \mathcal{L}_{\text{DT}}(\theta).$$

It was shown in [1] that $\mathcal{L}_{\text{DT}}$ can be re-written as

$$\begin{aligned} \mathcal{L}_{\text{DT}}(\theta) = \mathbb{E}_{p_{\text{data}}(x_0)} \Big[&\text{KL}(q_{K|0}(x_K|x_0)||p_{\text{ref}}(x_K)) - \mathbb{E}_{q_{1|0}(x_1|x_0)} \left[\log p^\theta_{0|1}(x_0|x_1) \right] \\ &+ \textstyle\sum_{k=1}^{K-1} \mathbb{E}_{q_{k+1|0}(x_{k+1}|x_0)} \left[\text{KL}(q_{k|k+1,0}(x_k|x_{k+1}, x_0)||p^\theta_{k|k+1}(x_k|x_{k+1})) \right] \Big] \end{aligned}$$

where KL is the Kullback-Leibler divergence. The forward kernels $q_{k+1|k}$ are chosen such that $q_{k|0}(x_k|x_0)$ can be computed efficiently in a time independent of k. With this, θ can be efficiently trained by taking a random selection of terms from $\mathcal{L}_{\text{DT}}$ in each minibatch and performing a stochastic gradient step.

3 Continuous Time Framework

3.1 Forward process and its time reversal

Our method is built upon a continuous time process from $t = 0$ to $t = T$. State transitions can occur at any time during this process as opposed to the discrete time case where transitions only occur when one of the finite number of transition kernels is applied (see Figure 1). This process is known as a Continuous Time Markov Chain (CTMC), we provide a short overview of CTMCs in Appendix A for completeness. Giving an intuitive introduction here, we can define a CTMC through an initial distribution q_0 and a transition rate matrix $R_t \in \mathbb{R}^{S \times S}$. If the current state is $\tilde{x}$, then the transition rate matrix entry $R_t(\tilde{x}, x)$ is the instantaneous rate (occurrences per unit time) at which state $\tilde{x}$ transitions to state x. Loosely speaking, the next state in the process will likely be one for which $R_t(\tilde{x}, x)$ is high, and furthermore, the higher the rate is, the less time it will take for this transition to occur.

It turns out that the transition rate, R_t, also defines the infinitesimal transition probability for the process between the two time points $t - \Delta t$ and t

$$q_{t|t-\Delta t}(x|\tilde{x}) = \delta_{x,\tilde{x}} + R_t(\tilde{x}, x)\Delta t + o(\Delta t),$$

where $o(\Delta t)$ represents terms that tend to zero at a faster rate than Δt. Comparing to the discrete time case, we see that R_t assumes an analogous role to the discrete time forward kernel $q_{k+1|k}$ in how we define the forward process. Therefore, just as in discrete time, we design R_t such that: i) the forward process mixes quickly towards an easy to sample (stationary) distribution, p_{ref}, (e.g. uniform), ii) we can analytically obtain $q_{t|0}(x_t|x_0)$ distributions to enable efficient training (see Section 4.1 for how this is done). We initialize the forward CTMC at $q_0(x_0) = p_{\text{data}}(x_0)$ at time $t = 0$. We denote the marginal at time $t = T$ as $q_T(x_T)$, which should be close to $p_{\text{ref}}(x_T)$.

We now consider the time reversal of the forward process, which will take us from the marginal $q_T(x_T)$ back to the data distribution $p_{\text{data}}(x_0)$ through a reverse transition rate matrix, $\hat{R}_t \in \mathbb{R}^{S \times S}$:

$$q_{t|t+\Delta t}(\tilde{x}|x) = \delta_{\tilde{x},x} + \hat{R}_t(x, \tilde{x})\Delta t + o(\Delta t).$$

In discrete time, one uses Bayes rule to go from $q_{k+1|k}$ to $q_{k|k+1}$. We can use similar ideas to calculate $\hat{R}_t$ from R_t as per the following result.

Proposition 1. *For a forward in time CTMC, $\{x_t\}_{t\in[0,T]}$, with rate matrix R_t, initial distribution $p_{\text{data}}(x_0)$ and terminal distribution $q_T(x_T)$, there exists a CTMC with initial distribution $q_T(x_T)$ at $t = T$, terminal distribution $p_{\text{data}}(x_0)$ at $t = 0$ and transition rate matrix $\hat{R}_t$ that runs backwards in time and is almost everywhere equivalent to the time reversal of the forward CTMC, $\{x_t\}_{t\in[T,0]}$. Furthermore, $\hat{R}_t$ is related to R_t by the following expression*

$$\hat{R}_t(x, \tilde{x}) = R_t(\tilde{x}, x) \textstyle\sum_{x_0} \frac{q_{t|0}(\tilde{x}|x_0)}{q_{t|0}(x|x_0)} q_{0|t}(x_0|x) \quad \textit{for} \quad x \neq \tilde{x},$$

where $q_{t|0}(x|x_0)$ are the conditional marginals of the forward process and $q_{0|t}(x_0|x) = q_{t|0}(x|x_0)p_{\text{data}}(x_0)/q_t(x)$ with $q_t(x)$ being the marginal of the forward process at time t. When $x = \tilde{x}$, $\hat{R}_t(x, x) = -\sum_{x'\neq x} \hat{R}_t(x, x')$ because the rows must sum to zero (see Appendix A).

Unfortunately, $\hat{R}_t$ is intractable due to the intractability of $q_t(x)$ and thus of $q_{0|t}(x_0|x)$. Therefore, we consider an approximation $\hat{R}^\theta_t$ of $\hat{R}_t$ by approximating $q_{0|t}(x_0|x)$ with a parametric denoising model, $p^\theta_{0|t}(x_0|x)$:

$$\hat{R}^\theta_t(x, \tilde{x}) = R_t(\tilde{x}, x) \textstyle\sum_{x_0} \frac{q_{t|0}(\tilde{x}|x_0)}{q_{t|0}(x|x_0)} p^\theta_{0|t}(x_0|x) \quad \text{for} \quad x \neq \tilde{x}.$$

$\hat{R}^\theta_t(x, x) = -\sum_{x'\neq x} \hat{R}^\theta_t(x, x')$ as before. As a further analogy to the discrete time case, notice that when $x \neq \tilde{x}$, $\hat{R}^\theta_t$ has the same form as the discrete time parametric reverse kernel, $p^\theta_{k|k+1}$ defined in eq (1) but with the forward kernel, $q_{k+1|k}$, replaced by the forward rate, R_t.

3.2 Continuous Time ELBO

In discrete time, θ is trained by minimizing the discrete time negative ELBO, $\mathcal{L}_{\text{DT}}$, formed from the forward and reverse processes. We mirror this approach in continuous time by minimizing the corresponding continuous time (CT) negative ELBO, $\mathcal{L}_{\text{CT}}$, as derived below.

Proposition 2. *For the reverse in time CTMC with initial distribution $p_{ref}(x_T)$, terminal distribution $p^\theta_0(x_0)$, and reverse rate $\hat{R}^\theta_t$, an upper bound on the negative model log-likelihood, $\mathbb{E}_{p_{\text{data}}(x_0)}[-\log p^\theta_0(x_0)]$, is given by*

$$\mathcal{L}_{\text{CT}}(\theta) = T\, \mathbb{E}_{t\sim\mathcal{U}(0,T)q_t(x)r_t(\tilde{x}|x)} \Big[\Big\{ \textstyle\sum_{x'\neq x} \hat{R}^\theta_t(x, x') \Big\} - \mathcal{Z}^t(x) \log \Big(\hat{R}^\theta_t(\tilde{x}, x) \Big) \Big] + C,$$

where C is a constant independent of θ and

$$\mathcal{Z}^t(x) = \textstyle\sum_{x'\neq x} R_t(x, x') \qquad\qquad r_t(\tilde{x}|x) = (1 - \delta_{\tilde{x},x}) R_t(x, \tilde{x})/\mathcal{Z}^t(x).$$

$r_t(\tilde{x}|x)$ gives the probability of transitioning from x to $\tilde{x}$, given that we know a transition occurs at time t. We can optimize this objective efficiently with stochastic gradient descent. For a gradient update, we sample a batch of datapoints from $p_{\text{data}}(x_0)$, noise each datapoint using a random time, $t \sim \mathcal{U}(0, T)$, $x \sim q_{t|0}(x|x_0)$ and finally sample an auxiliary $\tilde{x}$ from $r_t(\tilde{x}|x)$ for each x. Intuitively, $(x, \tilde{x})$ are a pair of states following the forward in time noising process. Minimizing the second term in $\mathcal{L}_{\text{CT}}$ maximizes the reverse rate for this pair, but going in the backwards direction, $\tilde{x}$ to x. This is how $\hat{R}^\theta_t$ learns to reverse the noising process. Intuition on the first term and a direct comparison to $\mathcal{L}_{\text{DT}}$ is given in Appendix C.1.

The first argument of $\hat{R}^\theta_t$ is input into $p^\theta_{0|t}$ so we naively require two network forward passes on x and $\tilde{x}$ to evaluate the objective. We can avoid this by approximating the $q_t(x)$ sample in the first term with $\tilde{x}$ meaning we need only evaluate the network once on $\tilde{x}$. The approximation is valid because, as we show in Appendix C.4, $\tilde{x}$ is approximately distributed according to $q_{t+\delta t}$ for δt very small.

4 Efficient Forward and Backward Sampling

4.1 Choice of Forward Process

The transition rate matrix R_t needs to be chosen such that the forward process: i) mixes quickly towards p_{ref}, and ii) the $q_{t|0}(x|x_0)$ distributions can be analytically obtained. The Kolmogorov

differential equation for the CTMC needs to be integrated to obtain $q_{t|0}(x|x_0)$. This can be done analytically when R_t and $R_{t'}$ commute for all t, t', see Appendix E. An easy way to meet this condition is to let $R_t = \beta(t)R_b$ where $R_b \in \mathbb{R}^{S\times S}$ is a user-specified time independent base rate matrix and $\beta(t) \in \mathbb{R}$ is a time dependent scalar. We then obtain the analytic expression

$$q_{t|0}(x = j|x_0 = i) = \left(Q\exp\left[\Lambda \int_0^t \beta(s)ds\right] Q^{-1}\right)_{ij}$$

where $R_b = Q\Lambda Q^{-1}$ is the eigendecomposition of matrix R_b and $\exp[\cdot]$ the element-wise exponential.

Our choice of β schedule is guided by [3, 4], $\beta(t) = ab^t \log(b)$. The hyperparameters a and b are selected such that $q_T(x_T) \approx p_{\text{ref}}(x_T)$ at the terminal time $t = T$ while having a steady speed of 'information corruption' which ensures that $\hat{R}_t$ does not vary quickly in a short span of time.

We experiment with a variety of R_b matrices, for example, a uniform rate, $R_b = \mathbb{1}\mathbb{1}^T - S\text{Id}$, where $\mathbb{1}\mathbb{1}^T$ is a matrix of ones and Id is the identity. For problems with a heavy spatial bias, e.g. images, we can instead use a forward rate that only encourages transitions to nearby states; details and the links to the corresponding discrete time processes can be found in Appendix E.

4.2 Factorizing Over Dimensions

Our aim is to model data that is D dimensional, with each dimension taking one value from S possibilities. We now slightly redefine notation and say $\boldsymbol{x}^{1:D} \in \mathcal{X}^D$, $|\mathcal{X}| = S$. In this setting, calculating transition probabilities naively would require calculating S^D rate values corresponding to each of the possible next states. This is intractable for any reasonably sized S and D. We avoid this problem simply by factorizing the forward process such that each dimension propagates independently. Since this is a continuous time process and each dimension's forward process is independent of the others, the probability two or more dimensions transition at exactly the same time is zero. Therefore, overall in the full dimensional forward CTMC, each transition only ever involves a change in exactly one dimension. For the time reversal CTMC, it will also be true that exactly one dimension changes in each transition. This makes computation tractable because of the S^D rate values, only $D \times (S-1) + 1$ are non-zero - those corresponding to transitions where exactly one dimension changes plus the no change transition. Finally, we note that even though dimensions propagate independently in the forward direction, they are not independent in the reverse direction because the starting points for each dimension's forward process are not independent for non factorized p_{data}. The following proposition shows the exact forms for the forward and reverse rates in this case.

Proposition 3. *If the forward process factorizes as $q_{t|s}(\boldsymbol{x}_t^{1:D}|\boldsymbol{x}_s^{1:D}) = \prod_{d=1}^D q_{t|s}(x_t^d|x_s^d)$, $t > s$, then the forward and reverse rates are of the form*

$$R_t^{1:D}(\tilde{\boldsymbol{x}}^{1:D}, \boldsymbol{x}^{1:D}) = \sum_{d=1}^D R_t^d(\tilde{x}^d, x^d)\delta_{\boldsymbol{x}^{1:D\setminus d}, \tilde{\boldsymbol{x}}^{1:D\setminus d}},$$

$$\hat{R}_t^{1:D}(\boldsymbol{x}^{1:D}, \tilde{\boldsymbol{x}}^{1:D}) = \sum_{d=1}^D R_t^d(\tilde{x}^d, x^d)\delta_{\boldsymbol{x}^{1:D\setminus d}, \tilde{\boldsymbol{x}}^{1:D\setminus d}} \sum_{x_0^d} q_{0|t}(x_0^d|\boldsymbol{x}^{1:D}) \frac{q_{t|0}(\tilde{x}^d|x_0^d)}{q_{t|0}(x^d|x_0^d)},$$

where $R_t^d \in \mathbb{R}^{S\times S}$ and $\delta_{\boldsymbol{x}^{1:D\setminus d}, \tilde{\boldsymbol{x}}^{1:D\setminus d}}$ is 1 when all dimensions except for d are equal.

To find $\hat{R}_t^{\theta\,1:D}$ we simply replace $q_{0|t}(x_0^d|\boldsymbol{x}^{1:D})$ with $p_{0|t}^\theta(x_0^d|\boldsymbol{x}^{1:D})$ which is easily modeled with a neural network that outputs conditionally independent state probabilities in each dimension. In Appendix C.3 we derive the form of $\mathcal{L}_{\text{CT}}$ when we use this factorized form for $R_t^{1:D}$ and $\hat{R}_t^{\theta\,1:D}$.

4.3 Simulating the Generative Reverse Process with Tau-Leaping

The parametric generative reverse process is a CTMC with rate matrix $\hat{R}_t^{\theta\,1:D}$. Simulating this process from distribution $p_{\text{ref}}(\boldsymbol{x}_T^{1:D})$ at time $t = T$ back to $t = 0$ will produce approximate samples from $p_{\text{data}}(\boldsymbol{x}_0^{1:D})$. The process could be simulated exactly using Gillespie's Algorithm [21, 22, 23] which alternates between i) sampling a holding time to remain in the current state and ii) sampling a new state according to the current rate matrix, $\hat{R}_t^{\theta\,1:D}$ (see Appendix F). This is inefficient for large D because we would need to step through each transition individually and so only one dimension would change for each simulation step.

Instead, we use tau-leaping [20, 23], a very popular approximate simulation method developed in chemical physics. Rather than step back through time one transition to the next, tau-leaping leaps

from t to $t-\tau$ and applies all transitions that occurred in $[t, t-\tau]$ simultaneously. To make a leap, we assume $\hat{R}_t^{\theta\,1:D}$ and $\boldsymbol{x}_t^{1:D}$ remain constant in $[t, t-\tau]$. As we propagate from t to $t-\tau$, we count all of the transitions that occur, but hold off on actually applying them until we reach $t-\tau$, such that $\boldsymbol{x}_t^{1:D}$ remains constant in $[t, t-\tau]$. Assuming $\hat{R}_t^{\theta\,1:D}$ and $\boldsymbol{x}_t^{1:D}$ remain constant, the number of times a transition from $\boldsymbol{x}_t^{1:D}$ to $\tilde{\boldsymbol{x}}^{1:D}$ occurs in $[t, t-\tau]$ is Poisson distributed with mean $\tau \hat{R}_t^{\theta\,1:D}(\boldsymbol{x}_t^{1:D}, \tilde{\boldsymbol{x}}^{1:D})$. Once we reach $t-\tau$, we apply all transitions that occurred simultaneously i.e. $\boldsymbol{x}_{t-\tau}^{1:D} = \boldsymbol{x}_t^{1:D} + \sum_i P_i(\tilde{\boldsymbol{x}}_i^{1:D} - \boldsymbol{x}_t^{1:D})$ where P_i is a Poisson random variable with mean $\tau \hat{R}_t^{\theta\,1:D}(\boldsymbol{x}_t^{1:D}, \tilde{\boldsymbol{x}}_i^{1:D})$. Note the sum assumes a mapping from $\mathcal{X}$ to $\mathbb{Z}$.

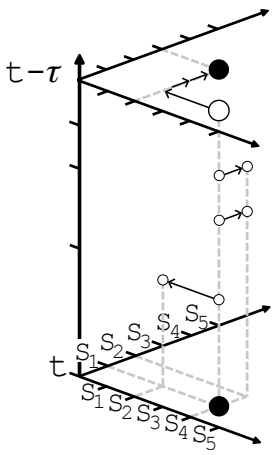

Figure 2: 3D visualization of one tau-leaping step from $x_t^{1:2} = \{S_4, S_1\}$ to $x_{t-\tau}^{1:2} = \{S_2, S_3\}$. Here, $D = 2$, $|\mathcal{X}| = 5$, $P_{12} = 1$, $P_{22} = 2$, all other $P_{ds} = 0$.

Using our knowledge of $\hat{R}_t^{\theta\,1:D}$, we can further unpack this update. Namely, $\hat{R}_t^{\theta\,1:D}(\boldsymbol{x}_t^{1:D}, \tilde{\boldsymbol{x}}^{1:D})$ can only be non-zero when $\tilde{\boldsymbol{x}}^{1:D}$ has a different value to $\boldsymbol{x}_t^{1:D}$ in exactly one dimension (rates for multi-dimensional changes are zero). Explicitly summing over these options we get $\boldsymbol{x}_{t-\tau}^{1:D} = \boldsymbol{x}_t^{1:D} + \sum_{d=1}^{D} \sum_{s=1 \setminus x_t^d}^{S} P_{ds}(s - x_t^d)\boldsymbol{e}^d$ where $\boldsymbol{e}^d$ is a one-hot vector with a 1 at dimension d and P_{ds} is a Poisson random variable with mean $\tau \hat{R}_t^{\theta\,1:D}(\boldsymbol{x}_t^{1:D}, \boldsymbol{x}_t^{1:D} + (s - x_t^d)\boldsymbol{e}^d)$. Since multiple P_{ds} can be non-zero, we see that tau-leaping allows $\boldsymbol{x}_t^{1:D}$ to change in multiple dimensions in a single step. Figure 2 visualizes this idea. During the $[t, t-\tau]$ interval, one jump occurs in dimension 1 and two jumps occur in dimension 2. These are all applied simultaneously once we reach $t-\tau$. When our discrete data has ordinal structure (e.g. Section 6.2) our mapping to $\mathbb{Z}$ is not arbitrary and making multiple jumps within the same dimension ($\sum_{s=1 \setminus x_t^d}^{S} P_{ds} > 1$) is meaningful. In the non-ordinal/categorical case (e.g. Section 6.3) the mapping to $\mathbb{Z}$ is arbitrary and so, although taking simultaneous jumps in different dimensions is meaningful, taking multiple jumps within the same dimension is not. For this type of data, we reject changes to x_t^d for any d for which $\sum_{s=1 \setminus x_t^d}^{S} P_{ds} > 1$. In practice, the rejection rate is very small when $\hat{R}_t^{1:D}$ is suitable for categorical data (e.g. uniform), see Appendix H.3. In Section 4.5, our error bound accounts for this low probability of rejection and also the low probability of an out of bounds jump that we observe in practice in the ordinal case.

The tau-leaping approximation improves with smaller τ, recovering exact simulation in the limit as $\tau \to 0$. Exact simulation is similar to an autoregressive model in that only one dimension changes per step. Increasing τ and thus the average number of dimensions changing per step gives us a natural way to modulate the 'autoregressiveness' of the model and trade sample quality with compute (Figure 4 right). We refer to our method of using tau-leaping to simulate the reverse CTMC as τLDR (tau-leaping denoising reversal) which we formalize in Algorithm 1 in Appendix F.

We note that theoretically, one could approximate $\hat{R}_t^{\theta\,1:D}$ as constant in the interval $[t, t-\tau]$, and construct a transition probability matrix by solving the forward Kolmogorov equation with the matrix exponential $P_{t-\tau|t} \approx \exp(\tau \hat{R}_t^{\theta\,1:D})$. However, for the learned $\hat{R}_t^{\theta\,1:D} \in \mathbb{R}^{S^D \times S^D}$ matrix, it is intractable to compute this matrix exponential so we use tau-leaping for sampling instead.

4.4 Predictor-Corrector

During approximate reverse sampling, we aim for the marginal distribution of samples at time t to be close to $q_t(x_t)$ (the marginal at time t of the true CTMC). The continuous time framework allows us to exploit additional information to more accurately follow the reverse progression of marginals, $\{q_t(x_t)\}_{t \in [T,0]}$ and improve sample quality. Namely, after a tau-leaping 'predictor' step using rate $\hat{R}_t^{\theta}$, we can apply 'corrector' steps with rate R_t^c which has $q_t(x_t)$ as its stationary distribution. The corrector steps bring the distribution of samples at time t closer to the desired $q_t(x_t)$ marginal. R_t^c is easy to calculate as stated below

Proposition 4. *For a forward CTMC with marginals $\{q_t(x_t)\}_{t \in [0,T]}$, forward rate, R_t, and corresponding reverse CTMC with rate $\hat{R}_t$, the rate $R_t^c = R_t + \hat{R}_t$ has $q_t(x_t)$ as its stationary distribution.*

In practice, we approximate R_t^c by replacing $\hat{R}_t$ with $\hat{R}_t^\theta$. This is directly analogous to Predictor-Corrector sampling in continuous state spaces [4] that predict by integrating the reverse SDE and correct with score-based Markov chain Monte Carlo steps, see Appendix F.2 for further discussion.

4.5 Error Bound

Our continuous time framework also allows us to provide a novel theoretical bound on the error between the true data distribution and the sample distribution generated via tau-leaping (without predictor-corrector steps), in terms of the error in our approximation of the reverse rate and the mixing of the forward noising process.

We assume we have a time-homogeneous rate matrix R_t on $\mathcal{X}$, from which we construct the factorized rate matrix $R_t^{1:D}$ on $\mathcal{X}^D$ by setting $R_t^d = R_t$ for each d. Note that by rescaling time by a factor of $\beta(t)$ we can transform our choice of rate from Section 4.1 to be time-homogeneous. We will denote $|R| = \sup_{t\in[0,T],x\in\mathcal{X}} |R_t(x,x)|$, and let t_{mix} be the (1/4)-mixing time of the CTMC with rate R_t (see [24, Chapter 4.5]).

Theorem 1. *For any $D \geq 1$ and distribution p_{data} on $\mathcal{X}^D$, let $\{x_t\}_{t\in[0,T]}$ be a CTMC starting in p_{data} with rate matrix $R_t^{1:D}$ as above. Suppose that $\hat{R}_t^{\theta\,1:D}$ is an approximation to the reverse rate matrix and let $(y_k)_{k=0,1,\ldots,N}$ be a tau-leaping approximation to the reverse dynamics with maximum step size τ. Suppose further that there is some constant $M > 0$ independent of D such that*

$$\sum_{y\neq x} \left| \hat{R}_t^{1:D}(x,y) - \hat{R}_t^{\theta\,1:D}(x,y) \right| \leq M \tag{2}$$

for all $t \in [0,T]$. Then under the assumptions in Appendix B.5, there are constants $C_1, C_2 > 0$ depending on $\mathcal{X}$ and R_t but not D such that, if $\mathcal{L}(y_0)$ denotes the law of y_0, we have the total variation bound

$$||\mathcal{L}(y_0) - p_{\text{data}}||_{\text{TV}} \leq 3MT + \left\{ \left(|R|SDC_1\right)^2 + \tfrac{1}{2}C_2(M + C_1SD|R|) \right\} \tau T + 2\exp\left\{ -\frac{T\log^2 2}{t_{\text{mix}}\log 4D} \right\}$$

The first term of the above bound captures the error introduced by our approximation of the reverse rate $\hat{R}_t^{1:D}$ with $\hat{R}_t^{\theta\,1:D}$. The second term reflects the error introduced by the tau-leaping approximation, and is linear in both T and τ, showing that as we take our tau-leaping steps to be arbitrarily small, the error introduced by tau-leaping goes to zero. The final term describes the mixing of the forward chain, and captures the error introduced since p_{ref} and q_T are not exactly equal.

We choose to make the dependence of the bound on the dimension D explicit, since we are specifically interested in applying tau-leaping to high dimensional problems where we make transitions in different dimensions simultaneously in a single time step. The bound grows at worst quadratically in the dimension, versus e.g. exponentially. The bound is therefore useful in showing us that we do not need to make τ impractically small in high dimensions. Other than gaining these intuitions, we do not expect the bound to be particularly tight in practice and further it would not be practical to compute because of the difficulty in finding M, C_1 and C_2.

The assumptions listed in Appendix B.5 hold approximately for tau-leaping in practice when we use spatially biased rates for ordinal data such that jump sizes are small or uniform rates for non-ordinal data such that the dimensional rejection rate is small. These assumptions could be weakened, however, Theorem 1 would become much more involved, obscuring the intuition and structure of the problem.

5 Related Work

The application of denoising models to discrete data was first described in [1] using a binomial diffusion process for a binary dataset. Each reverse kernel $p^\theta_{k|k+1}$ was directly parameterized without using a denoising model $p^\theta_{0|k}$. In [25] an approach for discrete categorical data was suggested using a uniform forward noising kernel, $q_{k+1|k}$, and a reverse kernel parameterized through a denoising model, though no experiments were performed with the approach. Experiments on text and segmentation maps were then performed with a similar model in [6]. Other forward kernels were introduced in [8] that are more appropriate for certain data types such as the spatially biased Gaussian kernel. [9, 13] apply the approach to discrete latent space modeling using uniform and absorbing state forward

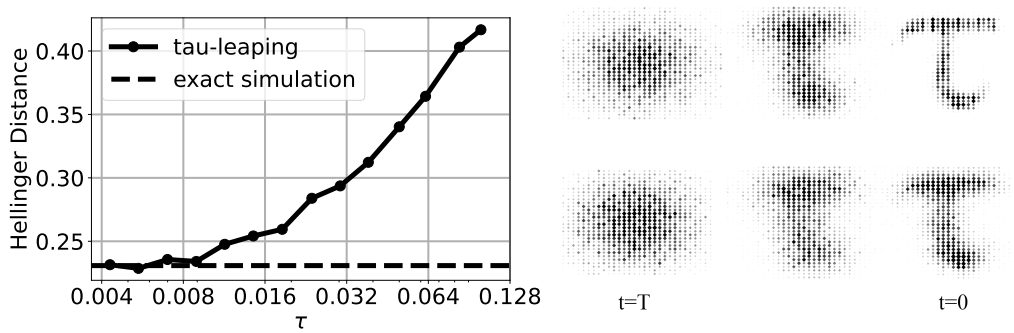

Figure 3: **Left:** Hellinger distance between the true training distribution and generated sample distributions with exact simulation or tau-leaping. With τ small, we simulate the reverse CTMC with the same fidelity as the exact simulation. **Top Right:** Histograms of the marginals during the reverse generative process simulated using tau-leaping with $\tau = 0.004$. Darker and larger diamonds represent increased density. **Bottom Right:** The same for $\tau = 0.1$, note the reduced sample quality.

kernels. Whilst a link to continuous time for the forward process is mentioned in [8], all of these approaches train and sample in discrete time. We show in Appendix G that this involves making an implicit approximation for multi-dimensional data. We extend this line of work by training and sampling in continuous time.

Other works also operate in discrete space but less rigidly follow the diffusion framework. A corruption process tailored to text is proposed in [12], whereby token deletion and insertion is also incorporated. [26] also focus on text, creating a generative reverse chain that repeatedly applies the same denoising kernel. The corruption distribution is also defined through the same denoising kernel to reduce distribution shift between training and sampling. In [7], a more standard masking based forward process is used but the reversal is interpreted from an order agnostic autoregressive perspective. They also describe how their model can be interpreted as the reversal of a continuous time absorbing state diffusion but do not utilize this perspective in training or sampling. [27] propose a denoising type framework that can be used on binary data where the forward and reverse process share the same transition kernel. Finally, in [11], the discrete latent space of a VQVAE is modeled by quantizing an underlying continuous state space diffusion with probabilistic quantization functions.

6 Experiments

6.1 Demonstrative Example

We first verify the method can accurately produce samples from the entire support of the data distribution and that tau-leaping can accurately simulate the reverse CTMC. To do this, we create a dataset formed of 2d samples of a state space of 32 arranged such that the histogram of the training dataset forms a 'τ' shape. We train a denoising model using the $\mathcal{L}_{\text{CT}}$ objective with $p^{\theta}_{0|t}$ parameterized through a residual MLP (full details in Appendix H.1). We then sample the parameterized reverse process using an exact method (up to needing to numerically integrate the reverse rate) and tau-leaping. Figure 3 top-right shows the marginals during reverse simulation with $\tau = 0.004$ and we indeed produce samples from the entire support of p_{data}. Furthermore, we find that with sufficiently small τ, we can match the fidelity of exact simulation of the reverse CTMC (Figure 3 left). The value of τ dictates the number of network evaluations in the reverse process according to NFE $= T/\tau$. In all experiments we use $T = 1$. Exact simulation results in a non zero Hellinger distance between the generated and training distributions because of imperfections in the learned $\hat{R}^{\theta}_{t}$ model.

6.2 Image Modeling

We now demonstrate that our continuous time framework gives us improved generative modeling performance versus operating in discrete time. We show this on the CIFAR-10 image dataset. Images are typically stored as discrete data, each pixel channel taking one value from 256 possibilities. Continuous state space methods have to somehow get around this fact by, for example, adding a discretization function at the end of the generative process [3] or adding uniform noise to the data.

Table 1: Sample quality metrics and model likelihoods for diffusion methods modeling CIFAR10 in discrete state space. Diffusion methods modeling CIFAR10 in continuous space are included for reference. The Inception Score (IS) and Fréchet Inception Distance (FID) are calculated using 50000 generated samples with respect to the training dataset as is standard practice. The ELBO values are reported on the test set in bits per dimension.

	Method	IS (↑)	FID (↓)	ELBO (↑)
Discrete state	D3PM Absorbing [8]	6.78	30.97	-4.40
	D3PM Gauss [8]	8.56	7.34	$\mathbf{-3.44}$
	τLDR-0 (ours)	8.74	8.10	-3.59
	τLDR-10 (ours)	**9.49**	**3.74**	-3.59
Continuous state	DDPM [3]	9.46	3.17	-3.75
	NCSN [4]	9.89	2.20	-

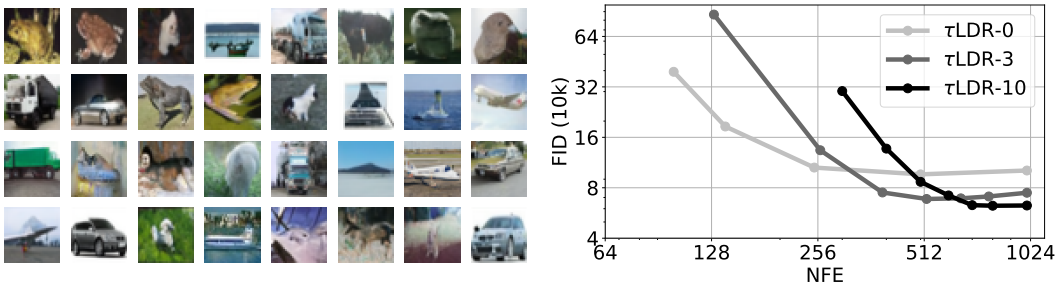

Figure 4: **Left:** Unconditional CIFAR10 samples from our τLDR-10 model **Right:** FID scores for the generated CIFAR10 samples versus number of $p^{\theta}_{0|t}$ evaluations during sampling (variation induced by varying τ). Calculated with 10k samples, hence the discrepancy with Table 1 [28].

Here, we model the images directly in discrete space. We parameterize $p^{\theta}_{0|t}$ using the standard U-net architecture [3] with the modifications for discrete state space suggested by [8]. We use a spatially biased rate matrix and train with an augmented $\mathcal{L}_{\text{CT}}$ loss including direct $p^{\theta}_{0|t}$ supervision, full experimental details are in Appendix H.2.

Figure 4 left shows randomly generated unconditional CIFAR10 samples from the model and we report sample quality metrics in Table 1. We see that our method (τLDR-0) with 0 corrector steps has better Inception Score but worse FID than the D3PM discrete time method. However, our τLDR-10 method with 10 corrector steps per predictor step at the end of the reverse sampling process ($t < 0.1T$) greatly improves sample quality, beating the discrete time method in both metrics and further closes the performance gap with methods modeling images as continuous data. The derivation of the corrector rate which gave us this improved performance required our continuous time framework. D3PM achieves the highest ELBO but we note that this does not correlate well with sample quality. In Table 1, τ was adjusted such that both τLDR-0 and τLDR-10 used 1000 $p^{\theta}_{0|t}$ evaluations in the reverse sampling procedure. We show how FID score varies with number of $p^{\theta}_{0|t}$ evaluations for τLDR-$\{0, 3, 10\}$ in Figure 4 right. The optimum number of corrector steps depends on the sampling budget, with lower numbers of corrector steps being optimal for tighter budgets. This is due to the increased τ required to maintain a fixed budget when we use a larger number of corrector steps.

6.3 Monophonic Music

In this experiment, we demonstrate our continuous time model improves generation quality on non-ordinal/categorical discrete data. We model songs from the Lakh pianoroll dataset [29, 30]. We select all monophonic sequences from the dataset such that at each of the 256 time steps either one from 128 notes is played or it is a rest. Therefore, our data has state space size $S = 129$ and dimension $D = 256$. We scramble the ordering of the state space when mapping to $\mathbb{Z}$ to destroy any ordinal structure. We parameterize $p^{\theta}_{0|t}$ with a transformer architecture [31] and train using a conditional form of $\mathcal{L}_{\text{CT}}$ targeting the conditional distribution of the final 14 bars (224 time steps) given the first 2 bars of the song. We use a uniform forward rate matrix, R_t, full experimental details

Table 2: Metrics comparing generated conditional samples and ground truth completions. We compute these over the test set showing mean±std with respect to 5 samples for each test song.

Model	Hellinger Distance	Proportion of Outliers
τLDR-0 Birth/Death	0.3928 ± 0.0010	0.1316 ± 0.0012
τLDR-0 Uniform	0.3765 ± 0.0013	0.1106 ± 0.0010
τLDR-2 Uniform	$\mathbf{0.3762 \pm 0.0015}$	$\mathbf{0.1091 \pm 0.0014}$
D3PM Uniform [8]	0.3839 ± 0.0002	0.1137 ± 0.0010

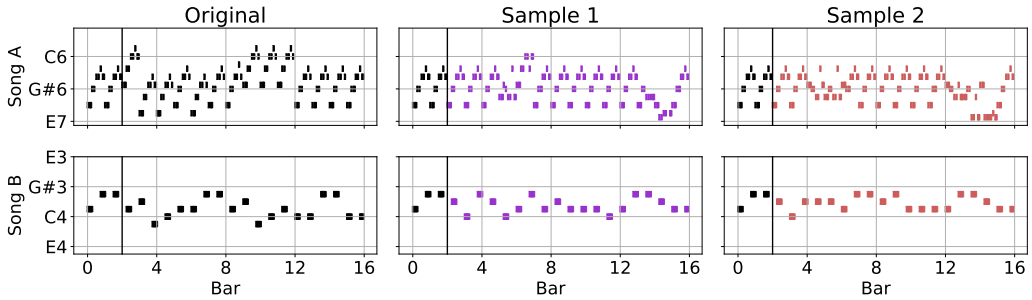

Figure 5: Conditional completions of an unseen music sequence. The conditioning 2 bars are shown to the left of the black line. More examples and audio recordings are linked in Appendix H.3.

are given in Appendix H.3. Conditional completions of unseen test songs are shown in Figure 5. The model is able to faithfully complete the piece in the same style as the conditioning bars.

We quantify sample quality in Table 2. We use two metrics: the Hellinger distance between the histograms of generated and ground truth notes and the proportion of outlier notes in the generations but not in the ground truth. Using our method, we compare between a birth/death and uniform forward rate matrix R_t. The birth/death rate is only non-zero for adjacent states whereas the uniform rate allows transitions between arbitrary states which is more appropriate for the categorical case thus giving improved sample quality. Adding 2 corrector steps per predictor step further improves sample quality. We also compare to the discrete time method D3PM [8] with its most suitable corruption process for categorical data. We find it performs worse than our continuous time method.

7 Discussion

We have presented a continuous time framework for discrete denoising models. We showed how to efficiently sample the generative process with tau-leaping and provided a bound on the error of the generated samples. On discrete data problems, we found our predictor-corrector sampler improved sample quality versus discrete time methods. Regarding limitations, our model requires many model evaluations to produce a sample. Our work has opened the door to applying the work improving sampling speed on continuous data [14, 15, 16, 17, 32] to discrete data problems too. Modeling performance on images is also slightly behind continuous state space models, we hope this gap is further closed with bespoke discrete state architectures and corruption process tuning. Finally, we note that the ELBO values for the discrete time model on CIFAR10 are better than for our method. In this work, we focused on sample quality rather than using our model to give data likelihoods e.g. for compression downstream tasks.

Acknowledgements

Andrew Campbell and Joe Benton acknowledge support from the EPSRC CDT in Modern Statistics and Statistical Machine Learning (EP/S023151/1). Arnaud Doucet is partly supported by the EPSRC grant EP/R034710/1. He also acknowledges support of the UK Defence Science and Technology Laboratory (DSTL) and EPSRC under grant EP/R013616/1. This is part of the collaboration between US DOD, UK MOD and UK EPSRC under the Multidisciplinary University Research Initiative. This project made use of time on Tier 2 HPC facility JADE2, funded by EPSRC (EP/T022205/1).

References

[1] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. *International Conference on Machine Learning*, 2015.

[2] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. *Advances in Neural Information Processing Systems*, 2019.

[3] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 2020.

[4] Yang Song, Jascha Sohl-Dickstein, Diederik P. Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *International Conference on Learning Representations*, 2021.

[5] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat GANs on image synthesis. *Advances in Neural Information Processing Systems*, 2021.

[6] Emiel Hoogeboom, Didrik Nielsen, Priyank Jaini, Patrick Forré, and Max Welling. Argmax flows and multinomial diffusion: Learning categorical distributions. *Advances in Neural Information Processing Systems*, 2021.

[7] Emiel Hoogeboom, Alexey A Gritsenko, Jasmijn Bastings, Ben Poole, Rianne van den Berg, and Tim Salimans. Autoregressive diffusion models. *International Conference on Learning Representations*, 2022.

[8] Jacob Austin, Daniel Johnson, Jonathan Ho, Daniel Tarlow, and Rianne van den Berg. Structured denoising diffusion models in discrete state-spaces. *Advances in Neural Information Processing Systems*, 2021.

[9] Patrick Esser, Robin Rombach, Andreas Blattmann, and Bjorn Ommer. Imagebart: Bidirectional context with multinomial diffusion for autoregressive image synthesis. *Advances in Neural Information Processing Systems*, 2021.

[10] Emiel Hoogeboom, Victor Garcia Satorras, Clément Vignac, and Max Welling. Equivariant diffusion for molecule generation in 3d. *arXiv preprint arXiv:2203.17003*, 2022.

[11] Max Cohen, Guillaume Quispe, Sylvain Le Corff, Charles Ollion, and Eric Moulines. Diffusion bridges vector quantized variational autoencoders. *arXiv preprint arXiv:2202.04895*, 2022.

[12] Daniel D Johnson, Jacob Austin, Rianne van den Berg, and Daniel Tarlow. Beyond in-place corruption: Insertion and deletion in denoising probabilistic models. *ICML Workshop on Invertible Neural Networks, Normalizing Flows, and Explicit Likelihood Models (INNF+)*, 2021.

[13] Shuyang Gu, Dong Chen, Jianmin Bao, Fang Wen, Bo Zhang, Dongdong Chen, Lu Yuan, and Baining Guo. Vector quantized diffusion model for text-to-image synthesis. *arXiv preprint arXiv:2111.14822*, 2021.

[14] Alexia Jolicoeur-Martineau, Ke Li, Rémi Piché-Taillefer, Tal Kachman, and Ioannis Mitliagkas. Gotta go fast when generating data with score-based models. *arXiv preprint arXiv:2105.14080*, 2021.

[15] Qinsheng Zhang and Yongxin Chen. Fast sampling of diffusion models with exponential integrator. *arXiv preprint arXiv:2204.13902*, 2022.

[16] Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. *International Conference on Learning Representations*, 2022.

[17] Hyungjin Chung, Byeongsu Sim, and Jong Chul Ye. Come-closer-diffuse-faster: Accelerating conditional diffusion models for inverse problems through stochastic contraction. *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022.

[18] Tim Dockhorn, Arash Vahdat, and Karsten Kreis. Score-based generative modeling with critically-damped Langevin diffusion. *International Conference on Learning Representations*, 2022.

[19] Valentin De Bortoli, James Thornton, Jeremy Heng, and Arnaud Doucet. Diffusion schrödinger bridge with applications to score-based generative modeling. *Advances in Neural Information Processing Systems*, 2021.

[20] Daniel T Gillespie. Approximate accelerated stochastic simulation of chemically reacting systems. *The Journal of Chemical Physics*, 115(4):1716–1733, 2001.

[21] Daniel T Gillespie. A general method for numerically simulating the stochastic time evolution of coupled chemical reactions. *Journal of Computational Physics*, 22(4):403–434, 1976.

[22] Daniel T Gillespie. Exact stochastic simulation of coupled chemical reactions. *The Journal of Physical Chemistry*, 81(25):2340–2361, 1977.

[23] Darren J Wilkinson. *Stochastic Modelling for Systems Biology*. Chapman and Hall/CRC, 2018.

[24] David Levin, Yuval Peres, and Elizabeth Wilmer. *Markov Chains and Mixing Times*. American Mathematical Society, 2009.

[25] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *International Conference on Learning Representations*, 2021.

[26] Nikolay Savinov, Junyoung Chung, Mikolaj Binkowski, Erich Elsen, and Aaron van den Oord. Step-unrolled denoising autoencoders for text generation. *International Conference on Learning Representations*, 2022.

[27] Anirudh Goyal, Nan Rosemary Ke, Surya Ganguli, and Yoshua Bengio. Variational walkback: Learning a transition operator as a stochastic recurrent net. *Advances in Neural Information Processing Systems*, 2017.

[28] Min Jin Chong and David Forsyth. Effectively unbiased fid and inception score and where to find them. *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020.

[29] Colin Raffel. *Learning-based methods for comparing sequences, with applications to audio-to-midi alignment and matching*. PhD thesis, Columbia University, 2016.

[30] Hao-Wen Dong, Wen-Yi Hsiao, Li-Chia Yang, and Yi-Hsuan Yang. Musegan: Multi-track sequential generative adversarial networks for symbolic music generation and accompaniment. *AAAI Conference on Artificial Intelligence*, 2018.

[31] Gautam Mittal, Jesse Engel, Curtis Hawthorne, and Ian Simon. Symbolic music generation with diffusion models. *International Society for Music Information Retrieval*, 2021.

[32] Fan Bao, Chongxuan Li, Jun Zhu, and Bo Zhang. Analytic-dpm: an analytic estimate of the optimal reverse variance in diffusion probabilistic models. *International Conference on Learning Representations*, 2022.

[33] David F Anderson. A modified next reaction method for simulating chemical systems with time dependent propensities and delays. *The Journal of Chemical Physics*, 127(21):214107, 2007.

[34] Chie Furusawa, Shinya Kitaoka, Michael Li, and Yuri Odagiri. Generative probabilistic image colorization. *arXiv preprint arXiv:2109.14518*, 2021.

[35] Ethan Perez, Florian Strub, Harm De Vries, Vincent Dumoulin, and Aaron Courville. Film: Visual reasoning with a general conditioning layer. *AAAI Conference on Artificial Intelligence*, 2018.

[36] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in Neural Information Processing Systems*, 2017.

[37] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. *International Conference on Medical Image Computing and Computer-assisted Intervention*, 2015.

[38] Tim Salimans, Andrej Karpathy, Xi Chen, and Diederik P Kingma. Pixelcnn++: Improving the pixelcnn with discretized logistic mixture likelihood and other modifications. *International Conference on Learning Representations*, 2017.

[39] Xi Chen, Nikhil Mishra, Mostafa Rohaninejad, and Pieter Abbeel. Pixelsnail: An improved autoregressive generative model. *International Conference on Machine Learning*, 2018.

[40] Stefan Elfwing, Eiji Uchibe, and Kenji Doya. Sigmoid-weighted linear units for neural network function approximation in reinforcement learning. *Neural Networks*, 107:3–11, 2018.

[41] Maximilian Seitzer. pytorch-fid: FID Score for PyTorch. `https://github.com/mseitzer/pytorch-fid`, August 2020. Version 0.2.1.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes] We detail the continuous time framework in Section 3, show our theoretical result in Section 4.5 and demonstrate improved performance in Section 6
   (b) Did you describe the limitations of your work? [Yes] See Section 7
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] See Appendix I
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes] See Appendix B.5
   (b) Did you include complete proofs of all theoretical results? [Yes] See Appendix B
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] The code is available at `https://github.com/andrew-cr/tauLDR`
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See Appendix H
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] See Table 2
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See Appendix H
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] We cite the authors of the models and datasets we use in Section 6
   (b) Did you mention the license of the assets? [Yes] See Appendix H
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes] We include our curated data in the supplement
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...

(a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]

(b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]

(c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

Appendix

Contents

A Primer on Continuous Time Markov Chains **16**

B Proofs **17**

B.1 Proof of Proposition 1 17

B.2 Proof of Proposition 2 18

B.3 Proof of Proposition 3 22

B.4 Proof of Proposition 4 23

B.5 Proof of Theorem 1 23

C Continuous Time ELBO Details **28**

C.1 Comparison with the Discrete Time ELBO 28

C.2 Conditional Form 28

C.3 Continuous Time ELBO with Factorization Assumptions 28

C.4 One Forward Pass 30

D Direct Denoising Model Supervision **31**

E Choice of Forward Process **33**

F CTMC Simulation **35**

F.1 Exact CTMC and Tau-Leaping 35

F.2 Predictor-Corrector Discussion 35

G Implicit Dimensional Assumptions Made in Discrete Time **37**

H Experimental Details **38**

H.1 Demonstrative Example 38

H.2 Image Modeling 39

H.3 Monophonic Music 40

I Ethical Considerations **43**

The appendix is organized as follows. In Section A, we provide a short introduction to Continuous Time Markov Chains, including the relevant results we use in this work. Proofs for all the Propositions and Theorems from the main text are in Section B. We then describe in Section C some additional intuitions and forms of our proposed objective, $\mathcal{L}_{\text{CT}}$. In Section D, we describe how an additional direct denoising model supervision term can be added to the objective to improve empirical performance. Details for how we define the forward process in our model can be found in Section E. Section F describes in more detail how CTMCs can be simulated and includes the algorithmic description of tau-leaping. We argue in Section G that operating in discrete time forces an implicit assumption when using a factorized forward process on multi-dimensional data. Full experimental details for all investigations can be found in Section H as well as additional plots and results from our models. Finally, in Section I, we consider the social impacts of our research.

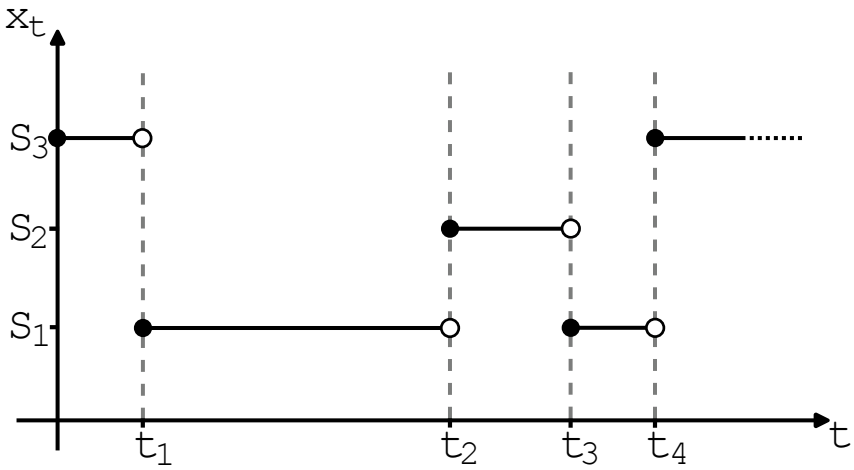

Figure 6: Schematic representation of a 1-dimensional CTMC with 3 states.

A Primer on Continuous Time Markov Chains

A Continuous Time Markov Chain (CTMC) is a right continuous stochastic process $\{x_t\}_{t\in[0,T]}$ satisfying the Markov property, with x_t taking values in a discrete state space $\mathcal{X}$. Since the CTMC is Markov, future behaviour of the process depends only on the current state and not the history. A schematic representation of a CTMC path is shown in Figure 6. The process repeatedly transitions from one state to another after having waited in the previous state for a randomly determined amount of time.

A CTMC can be completely characterised by its jumps and holding times. Specifically, the time between each jump or *holding time* is exponentially distributed with mean $\nu(x)$ where x is the state in which the process is holding. The next state that is jumped to is drawn from a jump probability distribution $r(\tilde{x}|x)$. The holding and jumping procedure is then repeated.

There is an equivalent definition involving the transition rate matrix, $R \in \mathbb{R}^{S\times S}$, that we use in the main paper. The transition rate matrix is defined as

$$R(\tilde{x}, x) = \lim_{\Delta t\to 0} \frac{q_{t|t-\Delta t}(x|\tilde{x}) - \delta_{x,\tilde{x}}}{\Delta t} \tag{3}$$

where $R(\tilde{x}, x)$ is the $(\tilde{x}, x)$ element of the transition rate matrix and $q_{t|t-\Delta t}(x|\tilde{x})$ is the infinitesimal transition probability of being in state x at time t given that the process was in state $\tilde{x}$ at time $t - \Delta t$. Conversely, the CTMC can itself be defined through this infinitesimal transition probability

$$q_{t|t-\Delta t}(x|\tilde{x}) = \delta_{x,\tilde{x}} + R(\tilde{x}, x)\Delta t + o(\Delta t) \tag{4}$$

where $o(\Delta t)$ represents terms that tend to zero at a faster rate than Δt. From this definition of the transition rate matrix, we can infer the following properties:

$$R(\tilde{x}, x) \geq 0 \quad \text{for} \quad \tilde{x} \neq x, \qquad R(x, x) \leq 0, \qquad R(x, x) = -\textstyle\sum_{x'\neq x} R(x, x') \tag{5}$$

$R(\tilde{x}, x)$ is the rate at which probability mass moves from state $\tilde{x}$ to x. $R(x, x)$ is the total rate at which probability mass moves out of state x and is thus negative.

In the time-homogeneous case, R has simple relations to the jump and holding time definitions.

$$\nu(x) = -\frac{1}{R(x, x)} \qquad\qquad r(\tilde{x}|x) = (1 - \delta_{\tilde{x},x})\frac{R(x, \tilde{x})}{-R(x, x)}$$

In the time-inhomogeneous case, our transition rate matrix will now depend on time, R_t, and these simple relations to the jump and holding time definition do not hold. However, R_t will still follow equations (3), (4) and (5).

The CTMC transition probabilities satisfy the Kolmogorov forward and backward equations. For $t > s$,

$$\text{Kolmogorov forward equation} \quad \partial_t q_{t|s}(x|\tilde{x}) = \sum_y q_{t|s}(y|\tilde{x}) R_t(y, x)$$

$$\text{Kolmogorov backward equation} \quad \partial_s q_{t|s}(x|\tilde{x}) = -\sum_y R_s(\tilde{x}, y) q_{t|s}(x|y)$$

The Kolmogorov forward equation also gives us a differential equation for the marginals of the CTMC.

$$\partial_t q_t(x) = \sum_y q_t(y) R_t(y, x)$$

Exponential and Poisson Random Variables In the time homogeneous case, holding times are exponentially distributed with mean $\nu(x) = -1/R(x, x)$. The tau-leaping algorithm relies on the fact that the number of events in interval $[0, t]$ is Poisson distributed with mean $\frac{1}{\nu}t$ when the inter-event times are exponentially distributed with mean ν.

B Proofs

B.1 Proof of Proposition 1

Proof. We recall that a process $\{x_t\}_{t\in[0,T]}$ taking values in $\mathcal{X}$ is called a CTMC if it is right-continuous and satisfies the Markov property. Denote $\{y_t\}_{t\in[0,T]} = \{x_{T-t}\}_{t\in[0,T]}$ except at the jump times of the forward process τ_n with $n \in \mathbb{N}$, where $y_{T-\tau_n} = x^-_{\tau_n} = \lim_{t\leq\tau_n, t\to\tau_n} x_t$. Hence, $\{y_t\}_{t\in[0,T]}$ is almost surely equal to $\{x_{T-t}\}_{t\in[0,T]}$ and is right-continuous. Since the Markov property is symmetric, we get that $\{y_t\}_{t\in[0,T]}$ is a CTMC. We now compute its transition matrix. Let $x, \tilde{x} \in \mathcal{X}$ with $x \neq \tilde{x}$, using the Kolmogorov forward equation, we have

$$\partial_t p_{t|s}(\tilde{x}|x) = \textstyle\sum_{y\in\mathcal{X}} p_{t|s}(y|x)\hat{R}_{T-t}(y, \tilde{x}) \ ,$$

where $\{p_{t|s},\ s, t \in [0, T],\ t > s\}$ is the transition probability system associated with $\{y_t\}_{t\in[0,T]}$ and $\{\hat{R}_{T-t}\}_{t\in[0,T]}$ is the transition rate matrix associated with $\{y_t\}_{t\in[0,T]}$. Note that

$$\begin{aligned}
p_{t|s}(x = j|\tilde{x} = i) &= \mathbb{P}(y_t = j \mid y_s = i) \\
&= \mathbb{P}(x_{T-t} = j \mid x_{T-s} = i) \\
&= \mathbb{P}(x_{T-s} = i|x_{T-t} = j)\frac{\mathbb{P}(x_{T-t} = j)}{\mathbb{P}(x_{T-s} = i)} \\
&= q_{T-s|T-t}(\tilde{x} = i|x = j)\frac{q_{T-t}(x = j)}{q_{T-s}(\tilde{x} = i)} \ .
\end{aligned}$$

where $\{q_{t|s}, s, t \in [0, T], t > s\}$ is the transition probability system associated with $\{x_t\}_{t\in[0,T]}$ and $\{q_t, t \in [0, T]\}$ are the marginals of $\{x_t\}_{t\in[0,T]}$. Now, writing the backward Kolmogorov equation for $\{x_t\}_{t\in[0,T]}$

$$\partial_s q_{t|s}(\tilde{x}|x) = -\textstyle\sum_{y\in\mathcal{X}} R_s(x, y) q_{t|s}(\tilde{x}|y)$$

Re-labeling the time indices we obtain,

$$\begin{aligned}
\partial_{T-t} q_{T-s|T-t}(\tilde{x}|x) &= -\textstyle\sum_{y\in\mathcal{X}} R_{T-t}(x, y) q_{T-s|T-t}(\tilde{x}|y) \\
\partial_t q_{T-s|T-t}(\tilde{x}|x) &= \textstyle\sum_{y\in\mathcal{X}} R_{T-t}(x, y) q_{T-s|T-t}(\tilde{x}|y)
\end{aligned}$$

Letting $s \to t$ and using that $\lim_{s\to t} q_{T-s|T-t}(x|\tilde{x}) = 0$, we get that

$$\begin{aligned}
\hat{R}_{T-t}(x,\tilde{x}) &= \lim_{s\to t} \partial_t p_{t|s}(\tilde{x}|x) \\
&= \lim_{s\to t} \partial_t \left(q_{T-s|T-t}(x|\tilde{x}) \frac{q_{T-t}(\tilde{x})}{q_{T-s}(x)} \right) \\
&= \lim_{s\to t} \left[\partial_t \left(q_{T-s|T-t}(x|\tilde{x}) \right) \frac{q_{T-t}(\tilde{x})}{q_{T-s}(x)} + q_{T-s|T-t}(x|\tilde{x}) \frac{\partial_t q_{T-t}(\tilde{x})}{q_{T-s}(x)} \right] \\
&= \lim_{s\to t} \partial_t \left(q_{T-s|T-t}(x|\tilde{x}) \right) \frac{q_{T-t}(\tilde{x})}{q_{T-s}(x)} \\
&= R_{T-t}(\tilde{x},x) \frac{q_{T-t}(\tilde{x})}{q_{T-t}(x)}
\end{aligned}$$

Re-labeling the time-indices on the rate matrices, we obtain

$$\hat{R}_t(x,\tilde{x}) = R_t(\tilde{x},x) \frac{q_t(\tilde{x})}{q_t(x)}$$

Now we write the marginal ratio $\frac{q_t(\tilde{x})}{q_t(x)}$ in a different form

$$\begin{aligned}
\frac{q_t(\tilde{x})}{q_t(x)} &= \sum_{x_0} \frac{p_{\text{data}}(x_0)}{q_t(x)} q_{t|0}(\tilde{x}|x_0) \\
&= \sum_{x_0} \frac{q_{0|t}(x_0|x)}{q_{t|0}(x|x_0)} q_{t|0}(\tilde{x}|x_0).
\end{aligned}$$

Substituting in this form for the marginal ratio concludes the proof.

□

B.2 Proof of Proposition 2

In this section, we detail two proofs for Proposition 2. The first is a formal proof using results from stochastic processes. We then provide a second informal proof for the same result to gain intuition into the $\mathcal{L}_{\text{CT}}$ objective that only relies on elementary results from CTMCs.

Proof 1 - Stochastic Processes

Proof. Let us write $\mathbb{Q}$ for the path measure of the forward CTMC with rate matrix R_t, $\hat{\mathbb{Q}}$ for the path measure of its exact time reversal and $\mathbb{P}^\theta$ for the path measure of the approximate reverse process with rate matrix $\hat{R}_t^\theta$. Also, we use superscripts to notate conditioning on the starting point, for example $\mathbb{Q}^{x_0}$ denotes the path measure of the forward process conditioned to start in x_0.

With this notation, we have

$$\begin{aligned}
-\log p_0^\theta(x_0) &= -\log \int p_{\text{ref}}(\mathrm{d}x_T) \int_{\{\hat{W}_T = x_0\}} \mathbb{P}^{\theta,x_T}(\mathrm{d}w) \\
&= -\log \int q_{T|0}(\mathrm{d}x_T) \int_{\{\hat{W}_T = x_0\}} \hat{\mathbb{Q}}^{x_T}(\mathrm{d}w) \frac{\mathrm{d}p_{\text{ref}}}{\mathrm{d}q_{T|0}}(x_T) \frac{\mathrm{d}\mathbb{P}^{\theta,x_T}}{\mathrm{d}\hat{\mathbb{Q}}^{x_T}}(w) \\
&= -\log \int q_{T|0}(\mathrm{d}x_T) \int \hat{\mathbb{Q}}(\mathrm{d}w|\hat{W}_0 = x_T, \hat{W}_T = x_0) \frac{\mathrm{d}p_{\text{ref}}}{\mathrm{d}q_{T|0}}(x_T) \frac{\mathrm{d}\mathbb{P}^{\theta,x_T}}{\mathrm{d}\hat{\mathbb{Q}}^{x_T}}(w) \mathbb{Q}^{x_T}\{\hat{W}_0 = x_0\} \\
&\leq \int q_{T|0}(\mathrm{d}x_T) \int \hat{\mathbb{Q}}(\mathrm{d}w|\hat{W}_0 = x_T, \hat{W}_T = x_0) \left\{ -\log \frac{\mathrm{d}\mathbb{P}^{\theta,x_T}}{\mathrm{d}\hat{\mathbb{Q}}^{x_T}}(w) \right\} + C,
\end{aligned}$$

where $\mathbb{P}^\theta, \hat{\mathbb{Q}}$ run in the reverse time direction. Writing $\hat{W}_s$ for a reverse path and integrating wrt $p_{\text{data}}(\mathrm{d}x_0)$ we have

$$\int p_{\text{data}}(x_0)[-\log p_0^\theta(x_0)] \leq \int p_{\text{data}}(x_0) \int q_{T|0}(\mathrm{d}x_T) \int \hat{\mathbb{Q}}(\mathrm{d}\hat{W}|\hat{W}_0 = x_T, \hat{W}_T = x_0)$$
$$\times \left\{ \int_{s=0}^{T} \hat{R}_{T-s}^\theta(\hat{W}_s)\mathrm{d}s - \sum_{s:\hat{W}_{s-}\neq\hat{W}_s} \log \mathbb{P}_{T-s}^\theta(\hat{W}_s|\hat{W}_{s-}) R_{T-s}^\theta(\hat{W}_{s-}) \right\} + C.$$

where $\hat{R}_t^\theta(x)$ is shorthand for $-\hat{R}_t^\theta(x,x)$.

When $x_0 \sim p_{\text{data}}, x_T \sim q_{T|0}(\cdot|x_0), \hat{W} \sim \hat{\mathbb{Q}}(\mathrm{d}W|\hat{W}_0 = x_T, \hat{W}_T = x_0)$, the reverse path is distributed according to $p_{\text{data}}(\mathrm{d}x_0)\mathbb{Q}_{x_0}(\mathrm{d}W)$ and therefore $(\hat{W}_{s-}, \hat{W}_s)$ is distributed like $(W_{T-s}, W_{(T-s)-})$ and thus we have

$$\int p_{\text{data}}(x_0)[-\log p_0^\theta(x_0)]$$
$$\leq \int p_{\text{data}}(x_0)\mathbb{Q}_{x_0}(\mathrm{d}W) \left\{ \int_{s=0}^{T} \hat{R}_{T-s}^\theta(W_{(T-s)-})\mathrm{d}s - \sum_{s:W_{(T-s)-}\neq W_{T-s}} \log \mathbb{P}_{T-s}^\theta(W_{(T-s)-}|W_{T-s}) \hat{R}_{T-s}^\theta(W_{T-s}) \right\} + C$$

Using Dynkin's lemma and the fact that $\mathbb{P}_t^\theta(x|y)\hat{R}_t^\theta(y) = \hat{R}_t^\theta(y,x)$ we can re-expresss this final line as

$$= \iint_{s=0}^{T} q_{T-s}(\mathrm{d}x) \left\{ \sum_{z\neq x} \hat{R}_{T-s}^\theta(x,z) - \sum_{z\neq x} R_{T-s}(x,z) \frac{\sum_{y\neq x} R_{T-s}(x,y)}{\sum_{z\neq x} R_{T-s}(x,z)} \log \hat{R}_{T-s}^\theta(y,x) \right\}$$
$$= \iint_{s=0}^{T} q_{T-s}(\mathrm{d}x) r_{T-s}(\mathrm{d}y|x) \left\{ \sum_{z\neq x} \hat{R}_{T-s}^\theta(x,z) - \sum_{z\neq x} R_{T-s}(x,z) \log \hat{R}_{T-s}^\theta(y,x) \right\}$$
$$= \iint_{s=0}^{T} q_s(\mathrm{d}x) r_s(\mathrm{d}y|x) \left\{ \sum_{z\neq x} \hat{R}_s^\theta(x,z) - \sum_{z\neq x} R_s(x,z) \log \hat{R}_s^\theta(y,x) \right\}.$$

which rearranges to give the continuous time ELBO in the form of Proposition 2.

□

Proof 2 - Limit of Discrete Time ELBO

Proof. Consider a partitioning of $[0,T]$, $0 = t_0 < t_1 < \cdots < t_{k-1} < t_k < t_{k+1} < \cdots < t_{K-1} < t_K = T$. Let $t_k - t_{k-1} = \Delta t$ for all k. In subscripts we use k as a shorthand for t_k when this does not cause confusion. Considering a CTMC with this time partitioning converts the problem into a discrete time Markov Chain with forward transition kernel, $q_{k+1|k}(x_{k+1}|x_k)$ and parameterized reverse kernel, $p_{k|k+1}^\theta(x_k|x_{k+1})$. Therefore, we can write the negative ELBO in its discrete time form, $\mathcal{L}_{\text{DT}}$

$$\mathcal{L}_{\text{DT}}(\theta) = \mathbb{E}_{p_{\text{data}}(x_0)} \Big[\text{KL}(q_{K|0}(x_K|x_0)||p_{\text{ref}}(x_K)) - \mathbb{E}_{q_{1|0}(x_1|x_0)} \left[\log p_{0|1}^\theta(x_0|x_1) \right]$$
$$+ \sum_{k=1}^{K-1} \mathbb{E}_{q_{k+1|0}(x_{k+1}|x_0)} \left[\text{KL}(q_{k|k+1,0}(x_k|x_{k+1},x_0)||p_{k|k+1}^\theta(x_k|x_{k+1})) \right] \Big]$$

In the following, we will write the transition kernels in terms of the CTMC rate matrices and take the limit as $\Delta t \to 0$ to obtain a continuous time negative ELBO.

First, consider one item from the inner sum of $\mathcal{L}_{\text{DT}}$

$$L_k = \mathbb{E}_{p_{\text{data}}(x_0)q_{k+1|0}(x_{k+1}|x_0)} \left[\text{KL}(q_{k|k+1,0}(x_k|x_{k+1},x_0)||p_{k|k+1}^\theta(x_k|x_{k+1})) \right]$$
$$= -\mathbb{E}_{p_{\text{data}}(x_0)q_{k+1|0}(x_{k+1}|x_0)q_{k|k+1,0}(x_k|x_{k+1},x_0)} \left[\log p_{k|k+1}^\theta(x_k|x_{k+1}) \right] + C$$
$$= -\mathbb{E}_{q_k(x_k)q_{k+1|k}(x_{k+1}|x_k)} \left[\log p_{k|k+1}^\theta(x_k|x_{k+1}) \right] + C$$

where we have absorbed terms that do not depend on θ into C. We now write $p^\theta_{k|k+1}(x_k|x_{k+1})$ in terms of $\hat{R}^\theta_k$.

$$p^\theta_{k|k+1}(x_k|x_{k+1}) = \delta_{x_k,x_{k+1}} + \hat{R}^\theta_k(x_{k+1},x_k)\Delta t + O(\Delta t^2)$$

$$\begin{aligned}
\log p^\theta_{k|k+1}(x_k|x_{k+1}) =& \log\left(\delta_{x_k,x_{k+1}} + \hat{R}^\theta_k(x_{k+1},x_k)\Delta t + O(\Delta t^2)\right) \\
=& \delta_{x_k,x_{k+1}} \log\left(1 + \hat{R}^\theta_k(x_k,x_k)\Delta t + O(\Delta t^2)\right) \\
&+ (1-\delta_{x_k,x_{k+1}}) \log\left(\hat{R}^\theta_k(x_{k+1},x_k)\Delta t + O(\Delta t^2)\right) \\
=& \delta_{x_k,x_{k+1}} \left(\hat{R}^\theta_k(x_k,x_k)\Delta t + O(\Delta t^2)\right) \\
&+ (1-\delta_{x_k,x_{k+1}}) \log\left(\hat{R}^\theta_k(x_{k+1},x_k)\Delta t + O(\Delta t^2)\right) \qquad (6)
\end{aligned}$$

where on the last line we have used the series expansion for $\log(1+z) = z - \frac{z^2}{2} + O(z^3)$ valid for $|z| \leq 1, z \neq -1$. For any finite $R^\theta_k(x_k,x_k)$, Δt can be taken small enough such that the series expansion holds. We now substitute this form for $\log p^\theta_{k|k+1}$ into L_k and further write the expectation over $q_{k+1|k}(x_{k+1}|x_k) = \delta_{x_k,x_{k+1}} + R_k(x_k,x_{k+1})\Delta t + O(\Delta t^2)$ as an explicit sum.

$$\begin{aligned}
L_k = -\mathbb{E}_{q_k(x_k)}\Bigg[\sum_{x_{k+1}} \Bigg\{ &\left[\delta_{x_k,x_{k+1}} + R_k(x_k,x_{k+1})\Delta t + O(\Delta t^2)\right] \times \\
&\Big[\delta_{x_k,x_{k+1}} \left(\hat{R}^\theta_k(x_k,x_k)\Delta t + O(\Delta t^2)\right) \\
&+ \left(1-\delta_{x_k,x_{k+1}}\right) \log\left(\hat{R}^\theta_k(x_{k+1},x_k)\Delta t + O(\Delta t^2)\right)\Big]\Bigg\}\Bigg] + C
\end{aligned}$$

$$\begin{aligned}
L_k = -\mathbb{E}_{q_k(x_k)}\Bigg[\sum_{x_{k+1}} \Bigg\{ &\delta_{x_k,x_{k+1}} \hat{R}^\theta_k(x_k,x_k)\Delta t \\
&+ (1-\delta_{x_k,x_{k+1}}) R_k(x_k,x_{k+1})\Delta t \times \\
&\log\left(\hat{R}^\theta_k(x_{k+1},x_k)\Delta t + O(\Delta t^2)\right) + O(\Delta t^2)\Bigg\}\Bigg] + C
\end{aligned}$$

We can isolate $\hat{R}^\theta_k$ within the $\log$ through the following re-arrangement

$$\begin{aligned}
&\Delta t \log\left(\hat{R}^\theta_k(x_{k+1},x_k)\Delta t + O(\Delta t^2)\right) \\
&= \Delta t \log \Delta t + \Delta t \log\left(\hat{R}^\theta_k(x_{k+1},x_k) + O(\Delta t)\right) \\
&= \Delta t \log \Delta t + \Delta t \log\left(1 + O(\Delta t)\right) + \Delta t \log\left(\hat{R}^\theta_k(x_{k+1},x_k)\right)
\end{aligned}$$

where the first two terms are independent of θ and tend to 0 as $\Delta t \to 0$. Note that we assume $\hat{R}^\theta_k(x_{k+1},x_k) > 0$ for $x_{k+1} \neq x_k$ pairs which have $R_k(x_k,x_{k+1}) > 0$. This assumption is valid because, for $x_{k+1} \neq x_k$, we have

$$\hat{R}^\theta_k(x_{k+1},x_k) = R_k(x_k,x_{k+1}) \sum_{x_0} \frac{q_{k|0}(x_k|x_0)}{q_{k|0}(x_{k+1}|x_0)} p^\theta_{0|k}(x_0|x_{k+1})$$

and we assume $p^\theta_{0|k}(x_0|x_{k+1}) > 0$ which is valid when we parameterize $p^\theta_{0|k}$ with a softmax output. We assume an irreducible Markov chain, hence $q_{k|0} > 0$ for $t_k > 0$.

With this re-arrangement, and absorbing constant terms into C, we obtain

$$L_k = -\mathbb{E}_{q_k(x_k)}\Bigg[\sum_{x_{k+1}}\Bigg\{\delta_{x_k,x_{k+1}}\hat{R}_k^\theta(x_k,x_k)\Delta t + (1-\delta_{x_k,x_{k+1}})R_k(x_k,x_{k+1})\Delta t\log\left(\hat{R}_k^\theta(x_{k+1},x_k)\right) + O(\Delta t^2)\Bigg\}\Bigg] + C$$

$$L_k = -\mathbb{E}_{q_k(x_k)}\left[\hat{R}_k^\theta(x_k,x_k)\Delta t + \sum_{x_{k+1}\neq x_k} R_k(x_k,x_{k+1})\Delta t\log\hat{R}_k^\theta(x_{k+1},x_k) + O(\Delta t^2)\right]$$

The second term can be re-written so that it is more efficient to approximate with Monte Carlo. Currently the denoising model $p_{0|k}^\theta$ has to be evaluated for each term in the sum $\sum_{x_{k+1}\neq x_k}$ which would require multiple forward passes of the neural network. We can instead create a new probability distribution to sample from as follows. Define

$$r_k(x_{k+1}|x_k) = (1-\delta_{x_k,x_{k+1}})\frac{R_k(x_k,x_{k+1})}{\mathcal{Z}^k(x_k)}$$

where

$$\mathcal{Z}^k(x_k) = \sum_{x'_{k+1}\neq x_k} R_k(x_k,x'_{k+1})$$

So we now have

$$L_k = -\mathbb{E}_{q_k(x_k)r_k(x_{k+1}|x_k)}\left[\hat{R}_k^\theta(x_k,x_k)\Delta t + \mathcal{Z}^k(x_k)\Delta t\log\hat{R}_k^\theta(x_{k+1},x_k) + O(\Delta t^2)\right]$$

Examining the other terms in $\mathcal{L}_{\text{DT}}$ we have $\mathbb{E}_{p_{\text{data}}(x_0)}\left[\text{KL}(q_{K|0}(x_K|x_0)||p_{\text{ref}}(x_K))\right]$ which does not depend on θ and $\mathbb{E}_{q_{1|0}(x_1|x_0)}\left[\log p_{0|1}^\theta(x_0|x_1)\right]$ which we expand here

$$\begin{aligned}
&\mathbb{E}_{q_{1|0}(x_1|x_0)}\left[\log p_{0|1}^\theta(x_0|x_1)\right] \\
&= \sum_{x_1}\left\{\delta_{x_1,x_0} + \Delta t R_1(x_0,x_1) + O(\Delta t^2)\right\}\log p_{0|1}^\theta(x_0|x_1) \\
&= \log p_{0|1}^\theta(x_0|x_0) + \Delta t\sum_{x_1} R_1(x_0,x_1)\log p_{0|1}^\theta(x_0|x_1) + O(\Delta t^2) \\
&= \Delta t\hat{R}_1^\theta(x_0,x_0) + \Delta t\sum_{x_1} R_1(x_0,x_1)\log p_{0|1}^\theta(x_0|x_1) + O(\Delta t^2)
\end{aligned}$$

where on the final line we have used eq 6. In summary,

$$\begin{aligned}
\mathcal{L}_{\text{DT}} =& \Delta t\mathbb{E}_{p_{\text{data}}(x_0)q_{1|0}(x_1|x_0)}\left[-\hat{R}_1^\theta(x_0,x_0) + \sum_{x_1} R_1(x_0,x_1)\log p_{0|1}^\theta(x_0|x_1)\right] \\
&- \Delta t\sum_{k=1}^{K-1}\mathbb{E}_{q_k(x_k)r_k(x_{k+1}|x_k)}\left[\hat{R}_k^\theta(x_k,x_k) + \mathcal{Z}^k(x_k)\log\hat{R}_k^\theta(x_{k+1},x_k)\right] \\
&+ O(\Delta t^2) + C
\end{aligned}$$

We now take the limit of $\mathcal{L}_{\text{DT}}$ as $\Delta t \to 0$ and $K \to \infty$.

$$\lim_{\Delta t\to 0}\mathcal{L}_{\text{DT}} = \mathcal{L}_{\text{CT}} = -\int_0^T \mathbb{E}_{q_t(x)r_t(\tilde{x}|x)}\left[\hat{R}_t^\theta(x,x) + \mathcal{Z}^t(x)\log\left(\hat{R}_t^\theta(\tilde{x},x)\right)\right]dt + C$$

We can estimate the integral with Monte Carlo if we consider it to be an expectation with respect to a uniform distribution over times $(0,T)$. We also write $\hat{R}_t^\theta(x,x)$ explicitly as the negative off diagonal row sum.

$$\mathcal{L}_{\text{CT}}(\theta) = T\,\mathbb{E}_{t\sim\mathcal{U}(0,T)q_t(x)r_t(\tilde{x}|x)}\left[\left\{\sum_{x'\neq x}\hat{R}_t^\theta(x,x')\right\} - \mathcal{Z}^t(x)\log\left(\hat{R}_t^\theta(\tilde{x},x)\right)\right] + C,$$

□

B.3 Proof of Proposition 3

Proof. We assume $q_{t|s}(\boldsymbol{x}_t^{1:D}|\boldsymbol{x}_s^{1:D})$ factorizes as $\prod_{d=1}^{D} q_{t|s}(x_t^d|x_s^d)$ where $q_{t|s}(x_t^d|x_s^d)$, $d = 1, \ldots, D$ are the transition probabilities for independent singular dimensional CTMCs each with forward rate $R_t^d(\tilde{x}^d, x^d)$. In the following, we will drop time subscripts on x arguments. To find the correspondence between $R_t^{1:D}$ and R_t^d, we use the Kolmogorov forward equation

$$\partial_t q_{t|s}(\boldsymbol{x}^{1:D}|\tilde{\boldsymbol{x}}^{1:D}) = \sum_{\boldsymbol{y}^{1:D}} q_{t|s}(\boldsymbol{y}^{1:D}|\tilde{\boldsymbol{x}}^{1:D}) R_t^{1:D}(\boldsymbol{y}^{1:D}, \boldsymbol{x}^{1:D})$$

Substitute in our factorized form for $q_{t|s}$ into the LHS

$$\begin{aligned}
\partial_t q_{t|s}(\boldsymbol{x}^{1:D}|\tilde{\boldsymbol{x}}^{1:D}) &= \partial_t \left\{ \prod_{d=1}^{D} q_{t|s}(x^d|\tilde{x}^d) \right\} \\
&= \sum_{d=1}^{D} q_{t|s}(\boldsymbol{x}^{1:D\setminus d}|\tilde{\boldsymbol{x}}^{1:D\setminus d}) \partial_t q_{t|s}(x^d|\tilde{x}^d) \\
&= \sum_{d=1}^{D} q_{t|s}(\boldsymbol{x}^{1:D\setminus d}|\tilde{\boldsymbol{x}}^{1:D\setminus d}) \sum_{y^d} q_{t|s}(y^d|\tilde{x}^d) R_t^d(y^d, x^d) \\
&= \sum_{d=1}^{D} \sum_{\boldsymbol{y}^{1:D}} q_{t|s}(\boldsymbol{x}^{1:D\setminus d}|\tilde{\boldsymbol{x}}^{1:D\setminus d}) q_{t|s}(y^d|\tilde{x}^d) R_t^d(y^d, x^d) \delta_{\boldsymbol{x}^{1:D\setminus d}, \boldsymbol{y}^{1:D\setminus d}} \\
&= \sum_{d=1}^{D} \sum_{\boldsymbol{y}^{1:D}} q_{t|s}(\boldsymbol{y}^{1:D\setminus d}|\tilde{\boldsymbol{x}}^{1:D\setminus d}) q_{t|s}(y^d|\tilde{x}^d) R_t^d(y^d, x^d) \delta_{\boldsymbol{x}^{1:D\setminus d}, \boldsymbol{y}^{1:D\setminus d}} \\
&= \sum_{\boldsymbol{y}^{1:D}} q_{t|s}(\boldsymbol{y}^{1:D}|\tilde{\boldsymbol{x}}^{1:D}) \sum_{d=1}^{D} R_t^d(y^d, x^d) \delta_{\boldsymbol{x}^{1:D\setminus d}, \boldsymbol{y}^{1:D\setminus d}}
\end{aligned}$$

We therefore obtain

$$\sum_{\boldsymbol{y}^{1:D}} q_{t|s}(\boldsymbol{y}^{1:D}|\tilde{\boldsymbol{x}}^{1:D}) R_t^{1:D}(\boldsymbol{y}^{1:D}, \boldsymbol{x}^{1:D}) = \sum_{\boldsymbol{y}^{1:D}} q_{t|s}(\boldsymbol{y}^{1:D}|\tilde{\boldsymbol{x}}^{1:D}) \sum_{d=1}^{D} R_t^d(y^d, x^d) \delta_{\boldsymbol{x}^{1:D\setminus d}, \boldsymbol{y}^{1:D\setminus d}}$$

This must be true for all possible factorizable forward process transitions, $q_{t|s}$, including $q_{t|s}(\boldsymbol{y}^{1:D}|\tilde{\boldsymbol{x}}^{1:D}) = \delta_{\boldsymbol{y}^{1:D}, \tilde{\boldsymbol{x}}^{1:D}}$. This choice gives us our forward rate relation

$$R_t^{1:D}(\tilde{\boldsymbol{x}}^{1:D}, \boldsymbol{x}^{1:D}) = \sum_{d=1}^{D} R_t^d(\tilde{x}^d, x^d) \delta_{\boldsymbol{x}^{1:D\setminus d}, \tilde{\boldsymbol{x}}^{1:D\setminus d}}$$

Substituting this into our expression for the reverse rate from Proposition 1 we obtain

$$\begin{aligned}
\hat{R}_t^{1:D}(\boldsymbol{x}^{1:D}, \tilde{\boldsymbol{x}}^{1:D}) &= \sum_{\boldsymbol{x}_0^{1:D}} \sum_{d=1}^{D} R_t^d(\tilde{x}^d, x^d) \frac{q_t(\tilde{\boldsymbol{x}}^{1:D}|\boldsymbol{x}_0^{1:D})}{q_t(\boldsymbol{x}^{1:D}|\boldsymbol{x}_0^{1:D})} q_{0|t}(\boldsymbol{x}_0^{1:D}|\boldsymbol{x}^{1:D}) \delta_{\boldsymbol{x}^{1:D\setminus d}, \tilde{\boldsymbol{x}}^{1:D\setminus d}} \\
&= \sum_{\boldsymbol{x}_0^{1:D}} \sum_{d=1}^{D} R_t^d(\tilde{x}^d, x^d) \frac{q_{t|0}(\tilde{x}^d|x_0^d)}{q_{t|0}(x^d|x_0^d)} q_{0|t}(\boldsymbol{x}_0^{1:D}|\boldsymbol{x}^{1:D}) \delta_{\boldsymbol{x}^{1:D\setminus d}, \tilde{\boldsymbol{x}}^{1:D\setminus d}} \\
&= \sum_{d=1}^{D} R_t^d(\tilde{x}^d, x^d) \delta_{\boldsymbol{x}^{1:D\setminus d}, \tilde{\boldsymbol{x}}^{1:D\setminus d}} \sum_{x_0^d} q_{0|t}(x_0^d|\boldsymbol{x}^{1:D}) \frac{q_{t|0}(\tilde{x}^d|x_0^d)}{q_{t|0}(x^d|x_0^d)} \sum_{\boldsymbol{x}_0^{1:D\setminus d}} q_{0|t}(\boldsymbol{x}_0^{1:D\setminus d}|x_0^d, \boldsymbol{x}^{1:D}) \\
&= \sum_{d=1}^{D} R_t^d(\tilde{x}^d, x^d) \delta_{\boldsymbol{x}^{1:D\setminus d}, \tilde{\boldsymbol{x}}^{1:D\setminus d}} \sum_{x_0^d} q_{0|t}(x_0^d|\boldsymbol{x}^{1:D}) \frac{q_{t|0}(\tilde{x}^d|x_0^d)}{q_{t|0}(x^d|x_0^d)}
\end{aligned}$$

□

B.4 Proof of Proposition 4

Proof. By the Kolmogorov forward equation applied to the forwards process, we have

$$\partial_t q_t(x_t) = \sum_y R_t(y, x_t) q_t(y)$$

In addition, applying the Kolmogorov forward equation to the reverse process, which has the same marginals as the forward but time-reversed, we get

$$-\partial_t q_t(x_t) = \sum_y \hat{R}_t(y, x_t) q_t(y)$$

Summing these two equations gives

$$\sum_y \left\{ R_t(y, x_t) + \hat{R}_t(y, x_t) \right\} q_t(y) = 0$$

Therefore, by comparison with the Kolmogorov equation, $R_t + \hat{R}_t$ is the rate matrix of a CTMC with invariant distribution q_t. □

B.5 Proof of Theorem 1

In this section, we derive a bound on the error of our tau-leaping diffusion model. Because the tau-leaping approximation is only interesting in the case where multiple jumps are made along different dimensions in a single step, we choose to make the dependence of our bound on the dimension of our model explicit, rather than simply considering the case of fixed D and $\tau \to 0$.

Recall from the main text that we have a time-homogeneous rate matrix R_t on $\mathcal{X}$, from which we construct the factorised rate matrix $R_t^{1:D}$ on $\mathcal{X}^D$ by setting $R_t^d = R_t$ for each d, and will denote $|R| = \sup_{t \in [0,T], x \in \mathcal{X}} |R_t(x, x)|$, and let t_{mix} be the (1/4)-mixing time of the CTMC with rate R_t. We also define addition on the state space $\mathcal{X}^D$ using a mapping from $\mathcal{X}$ to $\mathbb{Z}$ as in Section 4.3 and component-wise addition.

Theorem 1. *For any $D \geq 1$ and distribution p_{data} on $\mathcal{X}^D$, let $\{x_t\}_{t \in [0,T]}$ be a CTMC starting in p_{data} with rate matrix $R_t^{1:D}$ as above. Suppose that $\hat{R}_t^{\theta\, 1:D}$ is an approximation to the reverse rate matrix and let $(y_k)_{k=0,1,\ldots,N}$ be a tau-leaping approximation to the reverse dynamics with maximum step size τ. Suppose further that there is some constant $M > 0$ independent of D such that*

$$\sum_{y \neq x} \left| \hat{R}_t^{1:D}(x, y) - \hat{R}_t^{\theta\, 1:D}(x, y) \right| \leq M$$

for all $t \in [0, T]$. Then under the assumptions listed below, there are constants $C_1, C_2 > 0$ depending on $\mathcal{X}$ and R_t but not D such that, if $\mathcal{L}(y_0)$ denotes the law of y_0, we have the total variation bound

$$||\mathcal{L}(y_0) - p_{\text{data}}||_{\text{TV}} \leq 3MT + \left\{ \left(|R| S D C_1 \right)^2 + \tfrac{1}{2} C_2 (M + C_1 S D |R|) \right\} \tau T + 2 \exp\left\{ -\frac{T \log^2 2}{t_{\text{mix}} \log 4D} \right\}$$

The above theorem holds under the following assumptions, where we write $x \sim y$ for $x, y \in S^D$ if they differ in at most one coordinate.

Assumption 1. *The data distribution p_{data} is strictly positive.*

Assumption 2. *There exists a constant $C_1 > 0$, depending on S and R_t but not D, such that for all $t \in [0, T]$ and $x, y \in S^D$ such that $x \sim y$, we have*

$$\frac{q_t(x)}{q_t(y)} \leq C_1$$

Assumption 3. *There exists a constant $C_2 > 0$, depending on S and R_t but not D, such that for all $t \in [0, T]$ and all $x, y \in S^D$ such that $x \sim y$, we have*

$$\sum_z \left| \hat{R}_t(x, x + z) - \hat{R}_t(y, y + z) \right| \leq C_2$$

If instead we were to allow C_1 and C_2 to depend on the dimension D, then Assumptions 2 and 3 follow trivially from Assumption 1 and the finiteness of the state space. However, we choose the stronger formulation above in order to make explicit the dependence of the error bound on the dimension, as previously explained.

As remarked in the main text, in most cases of practical interest (including the two examples explored in Section 6), Assumption 3 holds only approximately. However, we still expect the bound in Assumption 3 to hold whenever x, y are in addition chosen such that the tau-leaping approximation of the reverse process makes a jump between them with reasonably high probability. For example, in the case where our data is ordinal, we expect that for any $x \sim y$ jumps from x to y are only common when x is close to y, and thus $\hat{R}_t(x, x+z)$ and $\hat{R}_t(y, y+z)$ should be reasonably close whenever a jump from x to y occurs. Under a weaker assumption of this form, the proof of Theorem 1 can be adapted to work along similar lines, at the cost of a significant increase in technicality. We therefore choose to focus on the simpler case where Assumption 3 holds as it illustrates the key ideas.

In order to prove Theorem 1 we will require the following lemmas.

Proposition 5. *Let $(x_t)_{t\in[0,T]}$ and $(y_t)_{t\in[0,T]}$ be continuous time Markov chains on a finite state space S with generators G_t and H_t respectively which are both bounded and continuous in t. Let the Markov kernels associated to X and Y be K and L respectively. Then for any probability distribution ν on S we have*

$$||\nu K - \nu L||_{\text{TV}} \leq \int_0^T \sup_{x\in S} \Big\{ \sum_{y\neq x} |G_t(x,y) - H_t(x,y)| \Big\} \, \mathrm{d}t$$

Proof. We define a coupling of $(x_t)_{t\in[0,T]}$ and $(y_t)_{t\in[0,T]}$ as follows, based on the construction in Chapter 20.1 of [24]. First take $Z \sim \nu$ and set $x_0 = y_0 = Z$. Also define the variables $\tilde{x}_0 = \tilde{y}_0 = Z$.

Next, fix λ such that $|G_t(x,x)|, |H_t(x,x)| \leq \lambda$ for all $x \in S$, $t \in [0,T]$, let $(N_s)_{1\leq s\leq T}$ be a Poisson process on $[0,T]$ of rate λ, and set $N_0 = 0$. We write $N = N_T$, and $S_1, S_2, \ldots, S_N$ for the arrival times and set $S_{n+1} = T$. We construct x_t and y_t for $t > 0$ inductively as follows. For $t \in [0, S_1)$ let $x_t = y_t = x_0$. Let $1 \leq j \leq N$. Given $(x_r : r < S_j)$, $(y_r : r < S_j)$, and $\tilde{x}_j, \tilde{y}_j$, define the probability measures

$$\rho_j(\tilde{x}_j, w) := \begin{cases} G_{S_j}(\tilde{x}_j, w)/\lambda, & w \neq \tilde{x}_j \\ 1 - G_{S_j}(\tilde{x}_j, w)/\lambda, & w = \tilde{x}_j \end{cases} \qquad \rho'_j(\tilde{y}_j, w) := \begin{cases} H_{S_j}(\tilde{y}_j, w)/\lambda, & w \neq \tilde{y}_j \\ 1 - H_{S_j}(\tilde{y}_j, w)/\lambda, & w = \tilde{y}_j \end{cases} .$$

Sample $(\tilde{x}_{j+1}, \tilde{y}_{j+1})$ from a maximal coupling of (ρ_j, ρ'_j) and for $t \in [S_j, S_{j+1})$ set $x_t = \tilde{x}_{j+1}$, $y_t = \tilde{y}_{j+1}$. Finally set $x_T = x_{S_N}$ and $y_T = y_{S_N}$.

Now, observe that $(x_t, y_t)_{t\in[0,T]}$ defined in this way is a coupling of the given Markov chains. Moreover,

$$\begin{aligned} ||\nu K - \nu L||_{\text{TV}} &\leq \mathbb{P}(x_T \neq y_T) \\ &= \mathbb{E}\left[\sum_{j=1}^{N} \mathbb{I}\{x_s = y_s,\, s < S_j\} \, \mathbb{I}\{x_{S_j} \neq y_{S_j}\} \right] \\ &= \sum_{n=0}^{\infty} \frac{\lambda^n e^{-\lambda}}{n!} \sum_{j=0}^{n} \mathbb{E}\left[\mathbb{I}\{x_s = y_s,\, s < S_j\} \, \mathbb{I}\{x_{S_j} \neq y_{S_j}\} \right] \end{aligned}$$

and using the fact that jumps are coupled maximally

$$
\begin{aligned}
&= \sum_{n=0}^{\infty} \frac{\lambda^n e^{-\lambda}}{n!} \sum_{j=0}^{n} \mathbb{E}\Big[\mathbb{I}\{x_s = y_s,\ s < S_j\} \times \|\rho_j(X_{S_{j-1}}, \cdot) - \tilde{\rho}_j(X_{S_{j-1}}, \cdot)\|_{\mathrm{TV}}\Big] \\
&= \sum_{n=0}^{\infty} \frac{\lambda^n e^{-\lambda}}{n!} \sum_{j=0}^{n} \mathbb{E}\Big[\mathbb{I}\{x_s = y_s,\ s < S_j\} \frac{1}{\lambda} \sum_z |G_{S_j}(x_{S_{j-1}}, z) - H_{S_j}(x_{S_{j-1}}, z)|\Big] \\
&= \frac{1}{\lambda} \mathbb{E}\Big[\sum_{s: x_s \neq x_{s-}} \sum_z |G_s(x_{s-}, z) - H_s(x_{s-}, z)|\Big] \\
&= \frac{1}{\lambda} \int_{s=0}^{T} \mathbb{E}\Big[\lambda \sum_z |G_s(x_{s-}, z) - H_s(x_{s-}, z)|\Big] \\
&= \int_{s=0}^{T} \mathbb{E}\Big[\sum_z |G_s(x_{s-}, z) - H_s(x_{s-}, z)|\Big] \mathrm{d}s
\end{aligned}
$$

as required. □

Proposition 6. *For all $t \in [0, T]$ and $x, y \in \mathcal{X}^D$ such that $x \sim y$, we have*

$$|\partial_t \hat{R}_t(x, y)| \leq 2|R|^2 SDC_1^2$$

Moreover, it follows that $\hat{R}_t$ is bounded and continuous in t.

Proof. Omitting the superscripts for brevity where the notation is clear, we have

$$
\begin{aligned}
\left|\partial_t \hat{R}_t^{1:D}(x^{1:D}, y^{1:D})\right| &= \left|R_t(y, x) \partial_t \left\{\frac{q_t(y)}{q_t(x)}\right\}\right| \\
&= \left|R_t(y, x) \left\{\frac{q_t(y)}{q_t(x)} \frac{\sum_z R_t(z, y) q_t(z)}{q_t(y)} - \frac{q_t(y)}{q_t(x)} \frac{\sum_z R_t(z, x) q_t(z)}{q_t(x)}\right\}\right| \\
&\leq 2|R|^2 SDC_1^2
\end{aligned}
$$

where the second line follows from Kolmogorov's forward equation and the final inequality follows from Assumption 2 plus the fact that $R_t(z, x)$ (resp. $R_t(z, y)$) is only non-zero when $x \sim z$ (resp. $y \sim z$), and there are at most $|S||D|$ values of x (resp. y) for which this holds. □

We now give the proof of Theorem 1.

Proof of Theorem 1. Let us label the time steps used in tau-leaping by $0 = t_0 < t_1 < \cdots < t_N = T$, denote $\tau_k = t_k - t_{k-1}$, and denote the target stationary distribution by $\pi^D(x^{1:D}) = \prod_{d=1}^{D} \pi(x^d)$, where π is the invariant distribution of the single-dimensional transition matrix R_t^1.

Also, let $\mathcal{R}_k^{\theta,(\tau)}$ be the Markov kernel corresponding to applying the tau-leaping approximation with rate matrix $\hat{R}_{t_k}^{\theta}$ to move from t_k to t_{k-1}, and denote $\mathcal{R}^{\theta,(\tau)} = \mathcal{R}_N^{\theta,(\tau)} \mathcal{R}_{N-1}^{\theta,(\tau)} \ldots \mathcal{R}_1^{\theta,(\tau)}$ so that $\mathcal{R}^{\theta,(\tau)}$ expresses the full dynamics of the tau-leaping process and we have $\mathcal{L}(\hat{y}_0) = \pi^D \mathcal{R}^{\theta,(\tau)}$.

Then, as in [19] we can decompose

$$||\pi^D \mathcal{R}^{\theta,(\tau)} - p_d||_{\mathrm{TV}} \leq ||\pi^D \mathcal{R}^{\theta,(\tau)} - \pi^D (\mathbb{P}^R)_{T|0}||_{\mathrm{TV}} + ||\pi^D - q_T||_{\mathrm{TV}}$$

where $\mathbb{P}^R$ is the path measure of the exact reverse process.

We deal with the second term first. Let t_{mix} be the (1/4)-mixing time of the single-dimension CTMC with rate matrix R_t^1, i.e.

$$t_{\text{mix}} = \inf\left\{t \geq 0 : \sup_{x_0^1 \in S} ||q_{t|0}(\,\cdot\,|x_0^1) - \pi||_{\mathrm{TV}} \leq \frac{1}{4}\right\}$$

It then follows from

$$||q_{t|0}(\,\cdot\,|x_0^{1:D}) - \pi^D||_{\text{TV}} \leq \sum_{d=1}^{D} ||q_{t|0}(\,\cdot\,|x_0^d) - \pi||_{\text{TV}}$$

that t^D_{mix}, the (1/4)-mixing time of the full CTMC with rate matrix $R_t^{1:D}$, satisfies the inequality $t^D_{mix} \leq \{1 + \lceil \log_2 D \rceil\} t_{\text{mix}}$. If we view $(x_{mt^D_{mix}})_{m\in\mathbb{N}}$ as a discrete-time Markov chain, then standard results on Markov chain mixing (see, for example, Chapter 4.5 of [24]) show that

$$||q_{mt^D_{mix}|0}(\,\cdot\,|x_0^{1:D}) - \pi^D||_{\text{TV}} \leq 2^{-m}$$

It then follows that for any $T \geq 0$ we have

$$||\pi^D - q_T||_{\text{TV}} \leq 2\exp\left\{-\frac{T\log 2}{t^D_{mix}}\right\} \leq 2\exp\left\{-\frac{T\log^2 2}{t_{\text{mix}}\log 4D}\right\}$$

completing the bound on the second term.

To bound the first term, we define $\mathcal{P}_k = (\mathbb{P}^R)_{T-t_{k-1}|T-t_k}$ and decompose it as

$$\begin{aligned}
||\pi\mathcal{R}^{\theta,(\tau)} - \pi(\mathbb{P}^R)_{T|0}||_{\text{TV}} &\leq \sup_{\nu} ||\nu\mathcal{R}_N^{\theta,(\tau)}\dots\mathcal{R}_1^{\theta,(\tau)} - \nu\mathcal{P}_N\dots\mathcal{P}_1||_{\text{TV}} \\
&\leq \sup_{\nu} ||\nu\mathcal{R}_N^{\theta,(\tau)}\mathcal{R}_{N-1}^{\theta,(\tau)}\dots\mathcal{R}_1^{\theta,(\tau)} - \nu\mathcal{R}_N^{\theta,(\tau)}\mathcal{P}_{N-1}\dots\mathcal{P}_1||_{\text{TV}} \\
&\quad + \sup_{\nu} ||\nu\mathcal{R}_N^{\theta,(\tau)}\mathcal{P}_{N-1}\dots\mathcal{P}_1 - \nu\mathcal{P}_N\mathcal{P}_{N-1}\dots\mathcal{P}_1||_{\text{TV}} \\
&\leq \sup_{\nu} ||\nu\mathcal{R}_{N-1}^{\theta,(\tau)}\dots\mathcal{R}_1^{\theta,(\tau)} - \nu\mathcal{P}_{N-1}\dots\mathcal{P}_1||_{\text{TV}} + \sup_{\nu} ||\nu\mathcal{R}_N^{\theta,(\tau)} - \nu\mathcal{P}_N||_{\text{TV}} \\
&\leq \sum_{k=1}^{N} \sup_{\nu} ||\nu\mathcal{R}_k^{\theta,(\tau)} - \nu\mathcal{P}_k||_{\text{TV}}
\end{aligned}$$

by proceeding inductively. So it suffices to find bounds on the total variation distance accumulated on each interval $[t_{k-1}, t_k]$.

Let $\mathcal{R}_k^\theta$ be the Markov kernel corresponding to running the chain from t_k to t_{k-1} with constant rate matrix $\hat{R}^\theta_{t_k}$. Since by Proposition 6 the reverse rate matrix $\hat{R}_t$ is bounded and continuous in t, using Proposition 5 we made deduce that for any distribution ν on S we have

$$\begin{aligned}
||\nu\mathcal{P}_k - \nu\mathcal{R}_k^\theta||_{\text{TV}} &\leq \int_{t_{k-1}}^{t_k} \sup_{x\in S}\Big\{\sum_{y\neq x}\big|\hat{R}_t(x,y) - \hat{R}^\theta_{t_k}(x,y)\big|\Big\}\,\mathrm{d}t \\
&\leq \int_{t_{k-1}}^{t_k} \sup_{x\in S}\Big\{\sum_{y\neq x}\big|\hat{R}_t(x,y) - \hat{R}_{t_k}(x,y)\big|\Big\}\,\mathrm{d}t \\
&\quad + \int_{t_{k-1}}^{t_k} \sup_{x\in S}\Big\{\sum_{y\neq x}\big|\hat{R}_{t_k}(x,y) - \hat{R}^\theta_{t_k}(x,y)\big|\Big\}\,\mathrm{d}t
\end{aligned}$$

The first half of this expression can be bounded using the Mean Value Theorem, according to

$$\begin{aligned}
\int_{t_{k-1}}^{t_k} \sup_{x\in S}\Big\{\sum_{y\neq x}\big|\hat{R}_t(x,y) - \hat{R}_{t_k}(x,y)\big|\Big\}\,\mathrm{d}t &\leq \int_{t_{k-1}}^{t_k} |t - t_k|\cdot 2|R|^2S^2D^2C_1^2\,\mathrm{d}t \\
&\leq \big(|R|SDC_1\tau_k\big)^2
\end{aligned}$$

where in the first line we have used that the summand is only non-zero when $y \sim x$, and there are at most $|S||D|$ values of y for which this holds. The second term can be bounded using condition (2), to get

$$\int_{t_{k-1}}^{t_k} \sup_{x\in S}\Big\{\sum_{y\neq x}\big|\hat{R}_{t_k}(x,y) - \hat{R}^\theta_{t_k}(x,y)\big|\Big\}\,\mathrm{d}t \leq M\tau_k$$

Combining these two expressions, we get a bound on $||\nu\mathcal{P}_k - \nu\mathcal{R}_k^\theta||_{\text{TV}}$.

$$||\nu\mathcal{P}_k - \nu\mathcal{R}_k^\theta||_{\text{TV}} \leq \big(|R|SDC_1\tau_k\big)^2 + M\tau_k$$

It remains to bound $||\nu\mathcal{R}_k^\theta - \nu\mathcal{R}_k^{\theta,(\tau)}||_{\text{TV}}$. Note that performing tau-leaping with rate matrix $\hat{R}_{t_k}^\theta$ starting in x_{t_k} is equivalent to running a continuous time Markov chain from time t_k to t_{k-1} with constant rate matrix $\hat{R}_{t_k}^{\theta,(\tau)}$ given by

$$\hat{R}_{t_k}^{\theta,(\tau)}(x,y) = \hat{R}_{t_k}^\theta(x_{t_k}, y - x + x_{t_k})$$

(followed potentially by a clamping operation to keep us within $\mathcal{X}^D$). By an analogous argument to the proof of Proposition 5,

$$||\delta_{x_{t_k}}\mathcal{R}_k^\theta - \delta_{x_{t_k}}\mathcal{R}_k^{\theta,(\tau)}||_{\text{TV}} \leq \int_{t_{k-1}}^{t_k} \mathbb{E}\Big[\sum_{y\neq x_t} |\hat{R}_{t_k}^\theta(x_t,y) - \hat{R}_{t_k}^\theta(x_{t_k}, y - x_t + x_{t_k})|\Big]\,\mathrm{d}t$$

where the expectation is taken over $(x_t)_{t\in[t_{k-1},t_k]}$ distributed according to the exact CTMC with rate matrix $\hat{R}_{t_k}^\theta$. (Note we have disregarded the clamping operation, since this can only decrease the resulting total variation distance.)

We may rewrite this bound in terms of the exact reverse process using condition (2) to get

$$||\delta_{x_{t_k}}\mathcal{R}_k^\theta - \delta_{x_{t_k}}\mathcal{R}_k^{\theta,(\tau)}||_{\text{TV}} \leq \int_{t_{k-1}}^{t_k} \mathbb{E}\Big[2M + \sum_{y\neq x_t} |\hat{R}_{t_k}(x_t,y) - \hat{R}_{t_k}(x_{t_k}, y - x_t + x_{t_k})|\Big]\,\mathrm{d}t$$

Let J_t be the number of jumps that (x_t) makes between t_k and t, and label the times of these jumps as $s_1,\dots,s_j$ where $t \leq s_1 \leq \dots \leq s_j \leq t_k$ and $j = J_t$ for convenience. Then by Assumption 3, we have

$$\begin{aligned}\sum_{y\neq x_t} |\hat{R}_{t_k}(x_t,y) - \hat{R}_{t_k}(x_{t_k}, y - x_t + x_{t_k})| &\leq \sum_z |\hat{R}_{t_k}(x_t, x_t + z) - \hat{R}_{t_k}(x_{s_1}, x_{s_1} + z)| + \dots \\ &\quad + \sum_z |\hat{R}_{t_k}(x_{s_j}, x_{s_j} + z) - \hat{R}_{t_k}(x_{t_k}, x_{t_k} + z)| \\ &\leq C_2 J_t\end{aligned}$$

where we have made the substitution $z = y - x_{t_k}$. We conclude that

$$\begin{aligned}||\delta_{x_{t_k}}\mathcal{R}_k^\theta - \delta_{x_{t_k}}\mathcal{R}_k^{\theta,(\tau)}||_{\text{TV}} &\leq \int_{t_{k-1}}^{t_k} \mathbb{E}\left[2M + C_2 J_t\right]\,\mathrm{d}t \\ &\leq 2M|t_k - t_{k-1}| + C_2\int_{t_{k-1}}^{t_k} |t_k - t| \cdot \sup_x |\hat{R}_{t_k}^\theta(x,x)|\,\mathrm{d}t \\ &\leq 2M\gamma_k + \frac{1}{2}C_2|\hat{R}_{t_k}^\theta|\tau_k^2 \\ &\leq 2M\tau_k + \frac{1}{2}C_2(M + C_1SD|R|)\tau_k^2\end{aligned}$$

where to bound $\mathbb{E}[J_t]$ we have observed that jumps of (x_t) occur at a rate bounded above by $\sup_x |\hat{R}_{t_k}^\theta(x,x)|$, and in the last line we have used the condition (2) and Assumption 2. Since the above holds for any choice of x_{t_k}, it follows that

$$\sup_\nu ||\nu\mathcal{R}_k^\theta - \nu\mathcal{R}_k^{\theta,(\tau)}||_{\text{TV}} \leq 2M\tau_k + \frac{1}{2}C_2(M + C_1SD|R|)\tau_k^2$$

Summing over k and putting all our bounds together, we get

$$||\mathcal{L}(y_0) - p_{\text{data}}||_{\text{TV}} \leq 3MT + \left\{\big(|R|SDC_1\big)^2 + \tfrac{1}{2}C_2(M + C_1SD|R|)\right\}\tau T + 2\exp\left\{-\tfrac{T\log^2 2}{t_{\text{mix}}\log 4D}\right\}$$

as required. □

C Continuous Time ELBO Details

C.1 Comparison with the Discrete Time ELBO

It is easiest to gain intuition on the $\mathcal{L}_{\text{CT}}$ objective by comparing it to its discrete time counterpart, $\mathcal{L}_{\text{DT}}$, and examining the way in which $\mathcal{L}_{\text{DT}}$ in the limit becomes $\mathcal{L}_{\text{CT}}$ when we take the time step size to be very small. We repeat the definition of $\mathcal{L}_{\text{CT}}$ here for convenience

$$\mathcal{L}_{\text{CT}}(\theta) = T\,\mathbb{E}_{t\sim\mathcal{U}(0,T)q_t(x)r_t(\tilde{x}|x)}\left[\left\{\sum_{x'\neq x}\hat{R}_t^\theta(x,x')\right\} - \mathcal{Z}^t(x)\log\left(\hat{R}_t^\theta(\tilde{x},x)\right)\right] + C,$$

Recall that a single term from the KL sum in $\mathcal{L}_{\text{DT}}$ up to an additive constant independent of θ is

$$-\mathbb{E}_{q_k(x_k)q_{k+1|k}(x_{k+1}|x_k)}\left[\log p^\theta_{k|k+1}(x_k|x_{k+1})\right].$$

Minimizing this term is to sample (x_k, x_{k+1}) from the forward dynamics and then maximize the assigned model probability for the pairing in the reverse direction. A similar idea can be used to understand $\mathcal{L}_{\text{CT}}$. First, we write $\log p^\theta_{k|k+1}(x_k|x_{k+1})$ in terms of $\hat{R}^\theta_k$ as

$$\begin{aligned}\log p^\theta_{k|k+1}(x_k|x_{k+1}) =& \delta_{x_k,x_{k+1}}\left(\hat{R}^\theta_k(x_k,x_k)\Delta t + O(\Delta t^2)\right)\\ &+ (1-\delta_{x_k,x_{k+1}})\log\left(\hat{R}^\theta_k(x_{k+1},x_k)\Delta t + O(\Delta t^2)\right)\end{aligned}$$

where we have separated the cases when $x_k = x_{k+1}$ and when $x_k \neq x_{k+1}$ (see the proof of $\mathcal{L}_{\text{CT}}$ for the full details). The first term will become the $\sum_{x'\neq x}\hat{R}^\theta_t(x,x')$ term in $\mathcal{L}_{\text{CT}}$ whilst the second term will become the $\mathcal{Z}^t(x)\log\left(\hat{R}^\theta_t(\tilde{x},x)\right)$ term. Now, when we minimize $\mathcal{L}_{\text{CT}}$, we are sampling $(x,\tilde{x})$ from the forward process and then maximizing the assigned model probability for the pairing in the reverse direction, just as in $\mathcal{L}_{\text{DT}}$. The slight extra complexity comes from the fact we are considering the case when $x_k = x_{k+1}$ and the case when $x_k \neq x_{k+1}$ separately. When $x_k = x_{k+1}$, this corresponds to the first term in $\mathcal{L}_{\text{CT}}$ which we can see is minimizing the reverse rate out of x which is exactly maximizing the model probability for no transition to occur. When $x_k \neq x_{k+1}$, this corresponds to the second term in $\mathcal{L}_{\text{CT}}$, which is maximizing the reverse rate from $\tilde{x}$ to x which in turn maximizes the model probability for the $\tilde{x}$ to x transition to occur.

C.2 Conditional Form

For the conditional form of $\mathcal{L}_{\text{CT}}$, denoted as $\bar{\mathcal{L}}_{\text{CT}}$, we instead upper bound the negative conditional model log-likelihood, $\mathbb{E}_{p_{\text{data}}(x_0,y)}[-\log p^\theta_0(x_0|y)]$ where y is our conditioner. $\bar{\mathcal{L}}_{\text{CT}}$ has the following form

$$\bar{\mathcal{L}}_{\text{CT}}(\theta) = T\,\mathbb{E}_{t\sim\mathcal{U}(0,T)p_{\text{data}}(x_0,y)q_{t|0}(x|x_0)r_t(\tilde{x}|x)}\left[\left\{\sum_{x'\neq x}\hat{R}^\theta_t(x,x'|y)\right\} - \mathcal{Z}^t(x)\log\left(\hat{R}^\theta_t(\tilde{x},x|y)\right)\right] + C,$$

where

$$\begin{aligned}\hat{R}^\theta_t(x,\tilde{x}|y) =& R_t(\tilde{x},x)\sum_{x_0}\frac{q_{t|0}(\tilde{x}|x_0)}{q_{t|0}(x|x_0)}p^\theta_{0|t}(x_0|x,y) \quad \text{for} \quad x\neq\tilde{x}.\\ =& -\sum_{x'\neq x}\hat{R}^\theta_t(x,x'|y) \quad \text{for} \quad x=\tilde{x}\end{aligned}$$

This follows easily from considering the conditional form of the discret time ELBO, $\bar{\mathcal{L}}_{\text{DT}}$ and using the same arguments as before to go from discrete time to continuous time.

$$\mathbb{E}_{p_{\text{data}}(x_0,y)}[-\log p^\theta_0(x_0|y)] \leq \mathbb{E}_{p_{\text{data}}(x_0,y)q_{1:K|0}(x_{1:K}|x_0)}\left[-\log\frac{p^\theta_{0:K}(x_{0:K}|y)}{q_{1:K|0}(x_{1:K}|x_0)}\right] = \bar{\mathcal{L}}_{\text{DT}}$$

C.3 Continuous Time ELBO with Factorization Assumptions

In the following Proposition, we show the form of $\mathcal{L}_{\text{CT}}$ when we use a factorized forward process. We note that in the proof we rearrange the sampling distribution from $p_{\text{data}}(\boldsymbol{x}_0^{1:D})q_{t|0}(\boldsymbol{x}^{1:D}|\boldsymbol{x}_0^{1:D})r_t(\tilde{\boldsymbol{x}}^{1:D}|\boldsymbol{x}^{1:D})$ to $p_{\text{data}}(\boldsymbol{x}_0^{1:D})\psi_t(\tilde{\boldsymbol{x}}^{1:D}|\boldsymbol{x}_0^{1:D})\phi_t(\boldsymbol{x}^{1:D}|\tilde{\boldsymbol{x}}^{1:D},\boldsymbol{x}_0^{1:D})$. This is not strictly necessary but it allows us to analytically sum over the intermediate $\boldsymbol{x}^{1:D}$ variable which greatly reduces the variance of the resulting objective.

Proposition 7. *The $\mathcal{L}_{\text{CT}}$ objective when we substitute in the factorized forms for the forward and reverse process given in Proposition 3 is*

$$\mathcal{L}_{\text{CT}} = T\,\mathbb{E}_{t\sim\mathcal{U}(0,T)p_{\text{data}}(\boldsymbol{x}_0^{1:D})q_{t|0}(\boldsymbol{x}^{1:D}|\boldsymbol{x}_0^{1:D})}\left[\sum_{d=1}^{D}\sum_{x'^d\neq x^d}\hat{R}_t^{\theta\,d}(\boldsymbol{x}^{1:D},x'^d)\right]$$
$$- T\,\mathbb{E}_{t\sim\mathcal{U}(0,T)p_{\text{data}}(\boldsymbol{x}_0^{1:D})\psi_t(\tilde{\boldsymbol{x}}^{1:D}|\boldsymbol{x}_0^{1:D})}\left[\sum_{d=1}^{D}\sum_{x^d\neq\tilde{x}^d}\phi_t(x^d|\tilde{\boldsymbol{x}}^{1:D},\boldsymbol{x}_0^{1:D})\mathcal{Z}^t(\tilde{\boldsymbol{x}}^{1:D/d}\circ x^d)\log\left(\hat{R}_t^{\theta\,d}(\tilde{\boldsymbol{x}}^{1:D},x^d)\right)\right]$$
$$+ C$$

with

$$\hat{R}_t^{\theta\,d}(\boldsymbol{x}^{1:D},\tilde{x}^d) = R_t^d(\tilde{x}^d,x^d)\sum_{x_0^d}p_{0|t}^{\theta}(x_0^d|\boldsymbol{x}^{1:D})\frac{q_{t|0}(\tilde{x}^d|x_0^d)}{q_{t|0}(x^d|x_0^d)}$$

$$\mathcal{Z}^t(\boldsymbol{x}^{1:D}) = \sum_{d=1}^{D}\sum_{\tilde{x}^d\neq x^d}R_t^d(x^d,\tilde{x}^d)$$

$$\phi_t(x^d|\tilde{\boldsymbol{x}}^{1:D},\boldsymbol{x}_0^{1:D}) = \frac{R_t^d(x^d,\tilde{x}^d)q_{t|0}(\tilde{\boldsymbol{x}}^{1:D\setminus d}\circ x^d|\boldsymbol{x}_0^{1:D})}{\mathcal{Z}^t(\tilde{\boldsymbol{x}}^{1:D\setminus d}\circ x^d)\sum_{d'=1}^{D}\sum_{x'^{d'}\neq\tilde{x}^{d'}}\frac{R_t^{d'}(x'^{d'},\tilde{x}^{d'})}{\mathcal{Z}^t(\tilde{\boldsymbol{x}}^{1:D\setminus d'}\circ x'^{d'})}q_{t|0}(\tilde{\boldsymbol{x}}^{1:D\setminus d'}\circ x'^{d'}|\boldsymbol{x}_0^{1:D})}$$

where $\circ$ represents the concatenation of a $D-1$ dimensional vector, $\boldsymbol{x}^{1:D\setminus d}$ with a scalar x^d, such that the resultant D dimensional vector has x^d at its dth dimension. $\psi_t(\tilde{\boldsymbol{x}}^{1:D}|\boldsymbol{x}_0^{1:D})$ is defined as the marginal of the forward noising process joint, $\int q_{t|0}(\boldsymbol{x}^{1:D}|\boldsymbol{x}_0^{1:D})r_t(\tilde{\boldsymbol{x}}^{1:D}|\boldsymbol{x}^{1:D})d\boldsymbol{x}^{1:D}$.

Proof. We first re-write the general form of $\mathcal{L}_{\text{CT}}$ here

$$\mathcal{L}_{\text{CT}}(\theta) = T\,\mathbb{E}_{t\sim\mathcal{U}(0,T)q_t(x)r_t(\tilde{x}|x)}\left[\left\{\sum_{x'\neq x}\hat{R}_t^{\theta}(x,x')\right\} - \mathcal{Z}^t(x)\log\left(\hat{R}_t^{\theta}(\tilde{x},x)\right)\right] + C$$

where

$$\mathcal{Z}^t(x) = \sum_{x'\neq x}R_t(x,x') \qquad\qquad r_t(\tilde{x}|x) = (1-\delta_{\tilde{x},x})R_t(x,\tilde{x})/\mathcal{Z}^t(x).$$

With a factorized forward process, $\hat{R}_t^{\theta}$ becomes

$$\hat{R}_t^{\theta\,1:D}(\boldsymbol{x}^{1:D},\tilde{\boldsymbol{x}}^{1:D}) = \sum_{d=1}^{D}\hat{R}_t^{\theta\,d}(\boldsymbol{x}^{1:D},\tilde{x}^d)\delta_{\boldsymbol{x}^{1:D\setminus d},\tilde{\boldsymbol{x}}^{1:D\setminus d}}$$

where

$$\hat{R}_t^{\theta\,d}(\boldsymbol{x}^{1:D},\tilde{x}^d) = R_t^d(\tilde{x}^d,x^d)\sum_{x_0^d}p_{0|t}^{\theta}(x_0^d|\boldsymbol{x}^{1:D})\frac{q_{t|0}(\tilde{x}^d|x_0^d)}{q_{t|0}(x^d|x_0^d)}$$

Substituting this form for $\hat{R}_t^{\theta\,1:D}$ into the first term in $\mathcal{L}_{\text{CT}}$ we get

$$\sum_{\boldsymbol{x}'^{1:D}\neq\boldsymbol{x}^{1:D}}\sum_{d=1}^{D}\hat{R}_t^{\theta\,d}(\boldsymbol{x}^{1:D},x'^d)\delta_{\boldsymbol{x}^{1:D\setminus d},\boldsymbol{x}'^{1:D\setminus d}}$$
$$= \sum_{d=1}^{D}\sum_{x'^d}\hat{R}_t^{\theta\,d}(\boldsymbol{x}^{1:D},x'^d)\sum_{\boldsymbol{x}'^{1:D\setminus d}}\delta_{\boldsymbol{x}^{1:D\setminus d},\boldsymbol{x}'^{1:D\setminus d}}(1-\delta_{\boldsymbol{x}'^{1:D},\boldsymbol{x}^{1:D}})$$
$$= \sum_{d=1}^{D}\sum_{x'^d\neq x^d}\hat{R}_t^{\theta\,d}(\boldsymbol{x}^{1:D},x'^d)$$

Now we tackle the second term in $\mathcal{L}_{\text{CT}}$. We first re-arrange the distribution over which we take the expectation:

$$p_{\text{data}}(\boldsymbol{x}_0^{1:D})q_{t|0}(\boldsymbol{x}^{1:D}|\boldsymbol{x}_0^{1:D})r_t(\tilde{\boldsymbol{x}}^{1:D}|\boldsymbol{x}^{1:D}) = p_{\text{data}}(\boldsymbol{x}_0^{1:D})\psi_t(\tilde{\boldsymbol{x}}^{1:D}|\boldsymbol{x}_0^{1:D})\phi_t(\boldsymbol{x}^{1:D}|\tilde{\boldsymbol{x}}^{1:D},\boldsymbol{x}_0^{1:D})$$

We have,

$$\begin{aligned}\phi_t(\boldsymbol{x}^{1:D}|\tilde{\boldsymbol{x}}^{1:D}, \boldsymbol{x}_0^{1:D}) &\propto q_{t|0}(\boldsymbol{x}^{1:D}|\boldsymbol{x}_0^{1:D}) r_t(\tilde{\boldsymbol{x}}^{1:D}|\boldsymbol{x}^{1:D}) \\ &= q_{t|0}(\boldsymbol{x}^{1:D}|\boldsymbol{x}_0^{1:D})(1-\delta_{\tilde{\boldsymbol{x}}^{1:D},\boldsymbol{x}^{1:D}}) \frac{\sum_{d=1}^D R_t^d(x^d, \tilde{x}^d)\delta_{\boldsymbol{x}^{1:D\setminus d},\tilde{\boldsymbol{x}}^{1:D\setminus d}}}{\mathcal{Z}^t(\boldsymbol{x}^{1:D})} \\ &= \sum_{d=1}^D \frac{R_t^d(x^d, \tilde{x}^d)}{\mathcal{Z}^t(\tilde{\boldsymbol{x}}^{1:D\setminus d} \circ x^d)} q_{t|0}(\tilde{\boldsymbol{x}}^{1:D\setminus d} \circ x^d|\boldsymbol{x}_0^{1:D}) \delta_{\boldsymbol{x}^{1:D\setminus d},\tilde{\boldsymbol{x}}^{1:D\setminus d}}(1-\delta_{\tilde{\boldsymbol{x}}^{1:D},\boldsymbol{x}^{1:D}})\end{aligned}$$

To find the normalization constant, we can sum the proportional term over $\boldsymbol{x}^{1:D}$

$$\begin{aligned}&\sum_{\boldsymbol{x}^{1:D}} \sum_{d=1}^D \frac{R_t^d(x^d, \tilde{x}^d)}{\mathcal{Z}^t(\tilde{\boldsymbol{x}}^{1:D\setminus d} \circ x^d)} q_{t|0}(\tilde{\boldsymbol{x}}^{1:D\setminus d} \circ x^d|\boldsymbol{x}_0^{1:D}) \delta_{\boldsymbol{x}^{1:D\setminus d},\tilde{\boldsymbol{x}}^{1:D\setminus d}}(1-\delta_{\tilde{\boldsymbol{x}}^{1:D},\boldsymbol{x}^{1:D}}) \\ &= \sum_{d=1}^D \sum_{x^d \neq \tilde{x}^d} \frac{R_t^d(x^d, \tilde{x}^d)}{\mathcal{Z}^t(\tilde{\boldsymbol{x}}^{1:D\setminus d} \circ x^d)} q_{t|0}(\tilde{\boldsymbol{x}}^{1:D\setminus d} \circ x^d|\boldsymbol{x}_0^{1:D})\end{aligned}$$

Therefore,

$$\phi_t(\boldsymbol{x}^{1:D}|\tilde{\boldsymbol{x}}^{1:D}, \boldsymbol{x}_0^{1:D}) = (1-\delta_{\tilde{\boldsymbol{x}}^{1:D},\boldsymbol{x}^{1:D}}) \sum_{d=1}^D \phi_t(x^d|\tilde{\boldsymbol{x}}^{1:D}, \boldsymbol{x}_0^{1:D}) \delta_{\boldsymbol{x}^{1:D\setminus d},\tilde{\boldsymbol{x}}^{1:D\setminus d}}$$

where

$$\phi_t(x^d|\tilde{\boldsymbol{x}}^{1:D}, \boldsymbol{x}_0^{1:D}) = \frac{R_t^d(x^d, \tilde{x}^d) q_{t|0}(\tilde{\boldsymbol{x}}^{1:D\setminus d} \circ x^d|\boldsymbol{x}_0^{1:D})}{\mathcal{Z}^t(\tilde{\boldsymbol{x}}^{1:D\setminus d} \circ x^d) \sum_{d'=1}^D \sum_{x'^{d'} \neq \tilde{x}^{d'}} \frac{R_t^{d'}(x'^{d'}, \tilde{x}^{d'})}{\mathcal{Z}^t(\tilde{\boldsymbol{x}}^{1:D\setminus d'} \circ x'^{d'})} q_{t|0}(\tilde{\boldsymbol{x}}^{1:D\setminus d'} \circ x'^{d'}|\boldsymbol{x}_0^{1:D})}$$

Now we write the second term as

$$\begin{aligned}&T\,\mathbb{E}_{t\sim\mathcal{U}(0,T) p_{\text{data}}(\boldsymbol{x}_0^{1:D}) \psi_t(\tilde{\boldsymbol{x}}^{1:D}|\boldsymbol{x}_0^{1:D})} \left[-\sum_{\boldsymbol{x}^{1:D}} \phi_t(\boldsymbol{x}^{1:D}|\tilde{\boldsymbol{x}}^{1:D}, \boldsymbol{x}_0^{1:D}) \mathcal{Z}^t(\boldsymbol{x}^{1:D}) \log \hat{R}_t^{\theta\,1:D}(\tilde{\boldsymbol{x}}^{1:D}, \boldsymbol{x}^{1:D}) \right] \\ &= -T\,\mathbb{E}_{t\sim\mathcal{U}(0,T) p_{\text{data}}(\boldsymbol{x}_0^{1:D}) \psi_t(\tilde{\boldsymbol{x}}^{1:D}|\boldsymbol{x}_0^{1:D})} \left[\sum_{d=1}^D \sum_{x^d \neq \tilde{x}^d} \phi_t(x^d|\tilde{\boldsymbol{x}}^{1:D}, \boldsymbol{x}_0^{1:D}) \mathcal{Z}^t(\tilde{\boldsymbol{x}}^{1:D/d} \circ x^d) \log \left(\hat{R}_t^{\theta\,d}(\tilde{\boldsymbol{x}}^{1:D}, x^d) \right) \right]\end{aligned}$$

□

C.4 One Forward Pass

To evaluate the $\mathcal{L}_{\text{CT}}$ objective, we naively need to perform two forward passes of the denoising network: $p_{0|t}^\theta(x_0|x)$ to calculate $\hat{R}_t^\theta(x, x')$ and $p_{0|t}^\theta(x_0|\tilde{x})$ to calculate $\hat{R}_t^\theta(\tilde{x}, x)$. This is wasteful because $\tilde{x}$ is created from x by applying a single forward transition which on multi-dimensional problems means $\tilde{x}$ differs from x in only a single dimension. To exploit the fact that $\tilde{x}$ and x are very similar, we approximate the sample $x \sim q_t(x)$ with the sample $\tilde{x} \sim \sum_x q_t(x) r_t(\tilde{x}|x)$. This gives the more efficient objective,

$$\mathcal{L}_{\text{eCT}}(\theta) = T\,\mathbb{E}_{t\sim\mathcal{U}(0,T) q_t(x) r_t(\tilde{x}|x)} \left[\left\{ \sum_{x' \neq \tilde{x}} \hat{R}_t^\theta(\tilde{x}, x') \right\} - \mathcal{Z}^t(x) \log \left(\hat{R}_t^\theta(\tilde{x}, x) \right) \right] + C$$

Table 3: Metrics on the monophonic music dataset comparing training with the efficient $\mathcal{L}_{\text{eCT}}$ objective vs the original $\mathcal{L}_{\text{CT}}$ objective. We compute these over the test set showing mean$\pm$std with respect to 5 samples for each test song.

Model	Hellinger Distance	Proportion of Outliers
τLDR-0 Uniform $\mathcal{L}_{\text{eCT}}$	0.3765 ± 0.0013	0.1106 ± 0.0010
τLDR-0 Uniform $\mathcal{L}_{\text{CT}}$	0.3797 ± 0.0009	0.1128 ± 0.0007

The approximation is valid because $q_t(x)$ and $\sum_x q_t(x) r_t(\tilde{x}|x)$ are very similar distributions, as we now show.

$$\begin{aligned}
\sum_x q_t(x) r_t(\tilde{x}|x) &= \sum_x q_t(x)(1-\delta_{x,\tilde{x}}) \frac{R_t(x,\tilde{x})}{\sum_{x' \neq x} R_t(x,x')} \\
&\propto -q_t(\tilde{x}) R_t(\tilde{x},\tilde{x}) + \sum_x q_t(x) R_t(x,\tilde{x}) \\
&= q_t(\tilde{x}) \sum_{x' \neq \tilde{x}} R_t(\tilde{x},x') + \partial_t q_t(\tilde{x}) \\
&\propto q_t(\tilde{x}) + \frac{1}{\sum_{x' \neq x} R_t(\tilde{x},x')} \partial_t q_t(\tilde{x}) \\
&= q_t(\tilde{x}) + \delta t \, \partial_t q_t(\tilde{x})
\end{aligned}$$

where on the third line we have used the Kolmogorov forward equation and defined $\delta_t = 1/\sum_{x' \neq x} R_t(\tilde{x},x')$. The distribution $\sum_x q_t(x) r_t(\tilde{x}|x)$ is therefore $q_{t+\delta t}(\tilde{x})$ approximated using a first-order Taylor expansion around $q_t(\tilde{x})$. We notice that δt is the average time to the next transition at time t. δt can be calculated for the practical settings we consider, its varies between $2\times10^{-6}T$ and $2\times10^{-8}T$ in the image modelling task and is $1\times10^{-3}T$ in the monophonic music task.

We perform an ablation experiment comparing between training with $\mathcal{L}_{\text{eCT}}$ and $\mathcal{L}_{\text{CT}}$ on the monophonic music dataset, the results are shown in Table 3. We find that we gain a small boost in performance when using the more efficient $\mathcal{L}_{\text{eCT}}$ objective alongside the improved efficiency. We hypothesize that this is because of a slight reduction in variance for the $\mathcal{L}_{\text{eCT}}$ objective due to increased negative correlation between the two terms in the objective when $\tilde{x}$ is shared between them.

D Direct Denoising Model Supervision

Following [8], we can introduce direct $p^\theta_{0|t}$ supervision into the optimization objective which has been found empirically to improve performance. We first contextualize the change by expressing $\mathcal{L}_{\text{CT}}$ with the dependence on $p^\theta_{0|t}$ made explicit.

$$\begin{aligned}
\mathcal{L}_{\text{CT}} = T\,\mathbb{E}_{t\sim\mathcal{U}(0,T) q_t(x) r_t(\tilde{x}|x)} \Big[\Big\{ \sum_{x' \neq x} R_t(x',x) \sum_{x_0} \frac{q_{t|0}(x'|x_0)}{q_{t|0}(x|x_0)} p^\theta_{0|t}(x_0|x) \Big\} \\
- \mathcal{Z}^t(x) \log \Big(R_t(x,\tilde{x}) \sum_{x_0} \frac{q_{t|0}(x|x_0)}{q_{t|0}(\tilde{x}|x_0)} p^\theta_{0|t}(x_0|\tilde{x}) \Big) \Big] + C
\end{aligned}$$

The signal for $p^\theta_{0|t}(x_0|x)$ comes through a sum over x_0 weighted by the ratio $\frac{q_{t|0}(x|x_0)}{q_{t|0}(\tilde{x}|x_0)}$. We can also provide a direct denoising signal by predicting the clean datapoint x_0 from the corrupted version x and using the negative log-likelihood loss.

$$L_{ll}(\theta) = T\,\mathbb{E}_{t\sim\mathcal{U}(0,T) p_{\text{data}}(x_0) q_{t|0}(x|x_0)} \Big[-\log p^\theta_{0|t}(x_0|x) \Big]$$

Proposition 8. *The true denoising distribution, $q_{0|t}$, minimizes L_{ll}*

Proof.

$$T\,\mathbb{E}_{t\sim\mathcal{U}(0,T)q_t(x)}\left[\mathrm{KL}\left(q_{0|t}(x_0|x)\,||\,p^\theta_{0|t}(x_0|x)\right)\right]$$
$$= T\,\mathbb{E}_{t\sim\mathcal{U}(0,T)p_{\text{data}}(x_0)q_{t|0}(x|x_0)}\left[-\log p^\theta_{0|t}(x_0|x)\right] + C$$

where C is a constant independent of θ. Therefore, minimizing L_{ll} is equivalent to minimizing the KL divergence between $q_{0|t}$ and $p^\theta_{0|t}$, which is minimized when $p^\theta_{0|t} = q_{0|t}$. □

If we obtain the true denoising distribution, $p^\theta_{0|t} = q_{0|t}$, then we will have the true reverse rate, $\hat{R}_t$. [8] find that optimizing with an objective combining L_{ll} and $\mathcal{L}_{\text{DT}}$ performs best, which we can also do in continuous time

$$\min_\theta \quad \mathcal{L}_{\text{CT}}(\theta) + \lambda L_{ll}(\theta)$$

where λ is a hyperparameter. In [8], it was found that training with L_{ll} alone resulted in poorer performance than when the ELBO was included in the loss. We provide a theoretical hypothesis as to why this may be the case here. We show that minimizing L_{ll} is equivalent to minimizing an upper bound on the negative ELBO in discrete time and thus by training with L_{ll} we are simply minimizing a looser bound on the negative model log-likelihood than if we were to use the negative ELBO directly.

Proposition 9. *Minimizing the sum of negative log-likelihoods*

$$\sum_{k=0}^{K-1}\mathbb{E}_{p_{\text{data}}(x_0)q_{k+1|0}(x_{k+1}|x_0)}\left[-\log p^\theta_{0|k+1}(x_0|x_{k+1})\right]$$

is equivalent to minimizing an upper bound on the negative ELBO.

Proof.

$$\mathcal{L}_{\text{DT}}(\theta) = \mathbb{E}_{p_{\text{data}}(x_0)}\Big[\mathrm{KL}(q_{K|0}(x_K|x_0)||p_{\text{ref}}(x_K)) - \mathbb{E}_{q_{1|0}(x_1|x_0)}\left[\log p^\theta_{0|1}(x_0|x_1)\right]$$
$$+ \textstyle\sum_{k=1}^{K-1}\mathbb{E}_{q_{k+1|0}(x_{k+1}|x_0)}\left[\mathrm{KL}(q_{k|k+1,0}(x_k|x_{k+1},x_0)||p^\theta_{k|k+1}(x_k|x_{k+1}))\right]\Big]$$

Consider one term from the sum

$$L_k = \mathbb{E}_{p_{\text{data}}(x_0)q_{k+1|0}(x_{k+1}|x_0)}\left[\mathrm{KL}(q_{k|k+1,0}(x_k|x_{k+1},x_0)||p^\theta_{k|k+1}(x_k|x_{k+1})\right]$$
$$= \mathbb{E}_{q_{k+1}(x_{k+1})q_{k|k+1}(x_k|x_{k+1})}\left[-\log p^\theta_{k|k+1}(x_k|x_{k+1})\right]$$
$$+ \mathbb{E}_{p_{\text{data}}(x_0)q_{k+1|0}(x_{k+1}|x_0)q_{k|k+1,0}(x_k|x_{k+1},x_0)}\left[\log q_{k|k+1,0}(x_k|x_{k+1},x_0)\right]$$

Now,

$$\mathbb{E}_{q_{k+1}(x_{k+1})q_{k|k+1}(x_k|x_{k+1})}\left[-\log p^\theta_{k|k+1}(x_k|x_{k+1})\right]$$
$$= \mathbb{E}_{q_{k+1}(x_{k+1})q_{k|k+1}(x_k|x_{k+1})}\left[-\log\sum_{\tilde{x}_0} q(x_k|\tilde{x}_0,x_{k+1})p^\theta_{0|k+1}(\tilde{x}_0|x_{k+1})\right]$$
$$= \mathbb{E}_{q_{k+1}(x_{k+1})q_{k|k+1}(x_k|x_{k+1})}\left[-\log\sum_{\tilde{x}_0}\frac{q_{0|k}(\tilde{x}_0|x_k)q_{k|k+1}(x_k|x_{k+1})}{q_{0|k+1}(\tilde{x}_0|x_{k+1})}p^\theta_{0|k+1}(\tilde{x}_0|x_{k+1})\right]$$
$$\leq \mathbb{E}_{q_{k+1}(x_{k+1})q_{k|k+1}(x_k|x_{k+1})q_{0|k}(\tilde{x}_0|x_k)}\left[-\log\frac{q_{k|k+1}(x_k|x_{k+1})}{q_{0|k+1}(\tilde{x}_0|x_{k+1})}p^\theta_{0|k+1}(\tilde{x}_0|x_{k+1})\right]$$
$$= \mathbb{E}_{p_{\text{data}}(x_0)q_{k+1|0}(x_{k+1}|x_0)}\left[-\log p^\theta_{0|k+1}(x_0|x_{k+1})\right]$$
$$+ \mathbb{E}_{p_{\text{data}}(x_0)q_{k|0}(x_k|x_0)q_{k+1|k}(x_{k+1}|x_k)}\left[-\log\frac{q_{k|k+1}(x_k|x_{k+1})}{q_{0|k+1}(x_0|x_{k+1})}\right]$$

Therefore,

$$
\begin{aligned}
\mathcal{L}_{\text{DT}} \leq & \sum_{k=0}^{K-1} \left\{ \mathbb{E}_{p_{\text{data}}(x_0) q_{k+1|0}(x_{k+1}|x_0)} \left[-\log p^{\theta}_{0|k+1}(x_0|x_{k+1}) \right] \right\} \\
& + \mathbb{E}_{p_{\text{data}}(x_0)} \left[\text{KL}(q_{K|0}(x_K|x_0)||p_K(x_K)) \right] \\
& + \sum_{k=1}^{K-1} \left\{ \mathbb{E}_{p_{\text{data}}(x_0) q_{k+1|0}(x_{k+1}|x_0) q_{k|k+1,0}(x_k|x_{k+1},x_0)} \left[\log q_{k|k+1,0}(x_k|x_{k+1}, x_0) \right] \right. \\
& \qquad \left. + \mathbb{E}_{p_{\text{data}}(x_0) q_{k|0}(x_k|x_0) q_{k+1|k}(x_{k+1}|x_k)} \left[-\log \frac{q_{k|k+1}(x_k|x_{k+1})}{q_{0|k+1}(x_0|x_{k+1})} \right] \right\}
\end{aligned}
$$

We can see that only the first term depends on θ. □

E Choice of Forward Process

We need to choose the structure of R_t such that we can analytically obtain $q_{t|0}$ marginals to enable efficient training.

Proposition 10. *If R_t and $R_{t'}$ commute for all t, t' then $q_{t|0}(x = j|x_0 = i) = \left(exp\left[\int_0^t R_s ds\right]\right)_{ij}$ where exp here is understood to be the matrix exponential function.*

Proof. Let $(P_t)_{ij} = q_{t|0}(x = j|x_0 = i)$. We show that $P_t = \exp\left(\int_0^t R_s ds\right)$ is a solution to the Kolmogorov forward equation, which in matrix form reads, $\partial_t P_t = P_t R_t$. Writing the matrix exponential in sum form

$$
\begin{aligned}
P_t &= \sum_{k=0}^{\infty} \frac{1}{k!} \left(\int_0^t R_s ds \right)^k \\
&= \text{Id} + \int_0^t R_s ds + \frac{1}{2!} \left(\int_0^t R_s ds \right)^2 + \frac{1}{3!} \left(\int_0^t R_s ds \right)^3 + \dots
\end{aligned}
$$

Now, differentiating and using the fact that R_t, $R_{t'}$ commute.

$$
\begin{aligned}
\partial_t P_t &= R_t + \int_0^t R_s ds R_t + \frac{1}{2!} \left(\int_0^t R_s ds \right)^2 R_t + \dots \\
&= \left\{ \sum_{k=0}^{\infty} \frac{1}{k!} \left(\int_0^t R_s ds \right)^k \right\} R_t \\
&= P_t R_t
\end{aligned}
$$

□

As stated in the main text, we achieve the commutative property by selecting $R_t = \beta(t) R_b$ where $\beta(t)$ is a time dependent scalar and R_b is a constant base matrix. We can utilize the eigendecomposition

of $R_b = Q\Lambda Q^{-1}$ to efficiently calculate P_t,

$$\begin{aligned}
P_t &= \exp\left(\int_0^t \beta(s) R_b ds\right) \\
&= \sum_{k=0}^{\infty} \frac{1}{k!}\left(\int_0^t \beta(s) R_b ds\right)^k \\
&= \sum_{k=0}^{\infty} \frac{1}{k!}\left(Q\Lambda Q^{-1}\int_0^t \beta(s) ds\right)^k \\
&= \sum_{k=0}^{\infty} \frac{1}{k!} Q\left(\Lambda\int_0^t \beta(s) ds\right)^k Q^{-1} \\
&= Q\left\{\sum_{k=0}^{\infty} \frac{1}{k!}\left(\Lambda\int_0^t \beta(s) ds\right)^k\right\} Q^{-1} \\
&= Q\exp\left[\Lambda\int_0^t \beta(s) ds\right] Q^{-1}
\end{aligned}$$

Since Λ is a diagonal matrix, the matrix exponential coincides with the element wise exponential making the final expression tractable to compute. We choose $\beta(t) = ab^t \log b$ because this makes the integral which dictates the variance of $q_{t|0}$ have a simple form $\int_0^t \beta(s)ds = ab^t - a$.

For categorical problems, we found a uniform rate matrix works well, $R_t = \beta \mathbb{1}\mathbb{1}^T - \beta S \mathrm{Id}$. This is directly analogous to the discrete time uniform transition matrix: $P = \alpha \mathbb{1}\mathbb{1}^T + (1 - S\alpha)\mathrm{Id}$ with α depending on the time discretization used. Indeed, if one calculates the corresponding discrete transition matrix for the uniform R_t rate through the matrix exponential, the uniform transition matrix is obtained. Another categorical corruption process is the absorbing state process. In discrete time, the transition matrix is given by $P = \alpha \mathbb{1}\mathbf{e}_*^T + (1-\alpha)\mathrm{Id}$ where $\mathbf{e}_*$ is the one-hot encoding of the absorbing state. The corresponding absorbing state continuous time process has transition rate matrix: $R_t = \beta \mathbf{1}\mathbf{e}_*^T - \beta \mathrm{Id}$. The correspondence for more complex transition matrices e.g. the Discretized Gaussian matrix in [8] is much harder to find analytically especially if the time inhomogeneous case is considered. For datasets with an ordinal structure, we construct a new rate matrix that maintains a bias towards nearby states using a similar approach as that taken by [8] to construct the Discretized Gaussian matrix.

We construct this matrix by first picking a desired stationary distribution, p_{ref}, and then filling in matrix entries such that we encourage transitions to nearby states whilst keeping p_{ref} as our stationary distribution. Specifically, we let p_{ref} be a discretized Gaussian over the state space, i.e.

$$p_{\text{ref}}(x) \propto \exp\left[-\frac{(x-\mu_0)^2}{2\sigma_0^2}\right]$$

To find a condition on the rate such that this is the case, recall the Kolmogorov differential equation for the marginals

$$\partial_t q_t(x) = \sum_{\tilde{x}} q_t(\tilde{x}) R_b(\tilde{x}, x)$$

Now, consider a rate that is in detailed balance with p_{ref}

$$p_{\text{ref}}(\tilde{x}) R_b(\tilde{x}, x) = p_{\text{ref}}(x) R_b(x, \tilde{x})$$

Substituting this rate into the Kolmogorov equation, we see that p_{ref} is the stationary distribution

$$\begin{aligned}
\partial_t p_{\text{ref}}(x) &= \sum_{\tilde{x}} p_{\text{ref}}(\tilde{x}) R_b(\tilde{x}, x) \\
&= \sum_{\tilde{x}} p_{\text{ref}}(x) R_b(x, \tilde{x}) \\
&= p_{\text{ref}}(x) \sum_{\tilde{x}} R_b(x, \tilde{x}) \\
&= 0
\end{aligned}$$

where the last line follows from the fact that the row sum of a rate matrix is zero. Note that any $R_t = \beta(t)R_b$ will also have this stationary distribution as the multiplication by $\beta(t)$ can be seen as just a scaling of the time axis. From the detailed balance equation, we gain a condition on R_b such that our desired p_{ref} is the stationary distribution

$$\frac{R_b(\tilde{x}, x)}{R_b(x, \tilde{x})} = \frac{p_{\text{ref}}(x)}{p_{\text{ref}}(\tilde{x})} = \exp\left[\frac{(\tilde{x} - \mu_0)^2}{2\sigma_0^2} - \frac{(x - \mu_0)^2}{2\sigma_0^2}\right]$$

This gives constraints on diagonal elements within R_b but does not fully define the entire matrix. To do this, we first make the assumption that μ is selected to be at the center of the state space. Then we set off diagonal terms to the right of the diagonal in the top half of the rate matrix and off diagonal terms to the left of the diagonal in the bottom half to be 1. Finally, progressing in from the top and bottom of the rate matrix we make definitions of rate matrix values that have not already been defined by the detailed balance condition. For clarity, we provide a pictorial representation of this scheme for an 8×8 rate matrix below

$$\begin{bmatrix} \cdot & 1 & \square & \square & \square & \square & \square & \square \\ \triangle & \cdot & 1 & \square & \square & \square & \square & \triangle \\ \triangle & \triangle & \cdot & 1 & \square & \square & \triangle & \triangle \\ \triangle & \triangle & \triangle & \cdot & 1 & \triangle & \triangle & \triangle \\ \triangle & \triangle & \triangle & \triangle & \cdot & \triangle & \triangle & \triangle \\ \triangle & \triangle & \triangle & \square & 1 & \cdot & \triangle & \triangle \\ \triangle & \triangle & \square & \square & \square & 1 & \cdot & \triangle \\ \triangle & \square & \square & \square & \square & \square & 1 & \cdot \end{bmatrix}$$

where $\square$ represents a value we will define, $\triangle$ represents a value that is defined relative to another entry through the detailed balance condition and $\cdot$ is a diagonal entry that is equal to the negative off diagonal row sum. We could define $\square$ values to be 0 to gain a sparse rate matrix, however, we found in early experiments that allowing transitions to further away states greatly reduces the mixing time and gives better performance. We define $\square$ in each row similarly, by setting it equal to $\exp[-i^2/\sigma_r^2]$ where i is the distance away from the '1' value in that row and σ_r is a hyperparameter defining the length scale in state space of a typical transition. This biases our forward process to make transitions between nearby states, at a length scale of σ_r.

F CTMC Simulation

F.1 Exact CTMC and Tau-Leaping

In this section, we first describe exact CTMC simulation before giving an algorithmic description of tau-leaping.

When a CTMC has a time-homogeneous rate matrix, we can use Gillespie's Algorithm [21, 22, 23] to exactly simulate it. This algorithm is based on the jump chain/holding time definition of the CTMC. It repeats the following two steps:

- Draw a holding time from an exponential distribution with mean $-1/R(x, x)$ and wait in the current state x for that amount of time.
- Sample the next state from $r(\tilde{x}|x) = (1 - \delta_{x,\tilde{x}})\frac{R(x,\tilde{x})}{\sum_{x' \neq x} R(x,x')}$

This Algorithm can be adjusted for the case when we have a time-inhomogeneous rate matrix using the modified next reaction method [33]. However, both algorithms still step through each transition in the CTMC individually and are thus unsuitable in our case because only one dimension would change for each simulation step making it very computationally expensive to produce a sample. Instead we use tau-leaping that allows multiple dimensions to change in a single simulation step. We detail this method in Algorithm 1.

F.2 Predictor-Corrector Discussion

In this section we compare predictor-corrector sampling schemes as applied to continuous state spaces and discrete state spaces.

Algorithm 1: Generative Reverse Process Simulation with Tau-Leaping

$t \leftarrow T$
$\boldsymbol{x}_t^{1:D} \sim p_{\text{ref}}(\boldsymbol{x}_T^{1:D})$
while $t > 0$ **do**
  Compute $p_{0|t}^{\theta}(x_0^d|\boldsymbol{x}_t^{1:D}), d = 1, \ldots, D$ with one forward pass of the denoising network
  for $d = 1, \ldots, D$ **do**
    for $s = 1, \ldots, S \backslash x_t^d$ **do**
      $\hat{R}_t^{\theta\, d}(\boldsymbol{x}_t^{1:D}, s) \leftarrow R_t^d(s, x_t^d) \sum_{x_0^d} p_{0|t}^{\theta}(x_0^d|\boldsymbol{x}_t^{1:D}) \frac{q_{t|0}(s|x_0^d)}{q_{t|0}(x_t^d|x_0^d)}$
      $P_{ds} \leftarrow \text{Poisson}\Big(\tau \hat{R}_t^{\theta\, d}(\boldsymbol{x}_t^{1:D}, s)\Big)$
    end
  end
  for $d = 1, \ldots, D$ **do**
    if *data is categorical AND* $\sum_{s=1}^{S} P_{ds} > 1$ **then**
      $x_{t-\tau}^d \leftarrow x_t^d$ `// reject change`
    else
      $x_{t-\tau}^d \leftarrow x_t^d + \sum_{s=1}^{S} P_{ds} \times (s - x_t^d)$
    end
  end
  $\boldsymbol{x}_{t-\tau}^{1:D} \leftarrow \text{Clamp}(\boldsymbol{x}_{t-\tau}^{1:D}, \text{min} = 1, \text{max} = S)$
  $t \leftarrow t - \tau$
end

The predictor-corrector scheme in continuous state spaces was introduced in [4]. It consists of alternating between a predictor step and a corrector step:

$$\text{Predictor} \quad \boldsymbol{x}_i \leftarrow \boldsymbol{x}_{i+1} + \gamma_i s_\theta(\boldsymbol{x}_{i+1}, i+1) + \sqrt{\gamma_i} z, \quad z \sim \mathcal{N}(0, I)$$
$$\text{Corrector} \quad \boldsymbol{x}_i \leftarrow \boldsymbol{x}_i + \epsilon_i s_\theta(\boldsymbol{x}_i, i) + \sqrt{2\epsilon_i} z, \quad z \sim \mathcal{N}(0, I)$$

where $\boldsymbol{x}_i$ is the state at sampling step i, s_θ is the learned score model approximating $\nabla_{\boldsymbol{x}} \log q_t(\boldsymbol{x})$ and γ_i, ϵ_i are the step sizes for the predictor and corrector respectively. We see that both take similar forms, except the corrector adds in a factor $\sqrt{2}$ more Gaussian noise during the update step.

In discrete state spaces, rather than sampling using gradient guided stochastic steps as in the continuous state space case, we sample by simulating CTMCs with defined rates. When we take a predictor step, we simulate using $\hat{R}_t^\theta$ and when we take a corrector step we simulate using $R_t^{c\,\theta} = \hat{R}_t^\theta + R_t$. If we simulate the CTMC exactly, we have seen in the previous section that this amounts to sampling next states from the categorical distribution defined by normalizing the row of the rate matrix corresponding to the current state. Therefore, corrector sampling can be seen as sampling from a slightly noisier categorical distribution defined through $R_t^{c\,\theta}$ as compared to the predictor categorical distribution defined through $\hat{R}_t^\theta$. This is analogous to the increased Gaussian noise applied during a corrector step in continuous state spaces.

Adding corrector steps brings the marginal of the samples closer to $q_t(\boldsymbol{x})$ and continued application of the corrector will further explore the domain of $q_t(\boldsymbol{x})$. In previous work on continuous state predictor-corrector methods, the number of corrector steps has been small (e.g. 1 or 2 corrector steps per predictor step) or indeed the corrector steps have been removed altogether. In this work we have found that using up to 10 corrector steps per predictor steps can be beneficial during certain regions of the reverse generative process. Additionally, in continuous state spaces, it has been observed that too many corrector steps can result in unwanted noise in the generated data [34].

We hypothesize that corrector steps are better utilized in discrete state spaces to explore the domain of $q_t(\boldsymbol{x})$ than in continuous state spaces. This is because, the corrector update is defined largely through the reverse rate itself, $\hat{R}_t^\theta$, just with the categorical probabilities being annealed slightly more towards uniform through the addition of the forward rate R_t. This may be a more effective update than simply adding extra Gaussian noise in the continuous state space case. Furthermore, the denoising model in continuous state spaces can be seen as outputting a point estimate of $\boldsymbol{x}_0$ of dimension D. However,

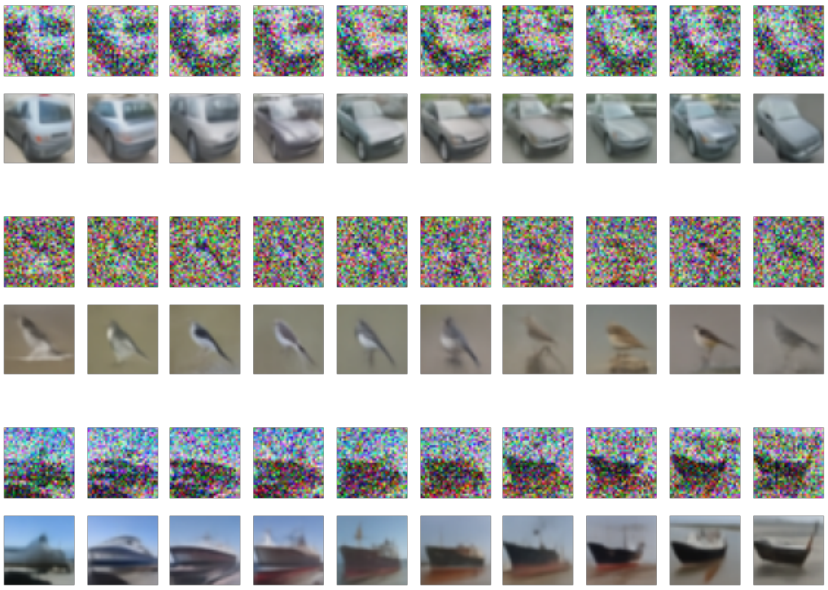

Figure 7: Progression of $\boldsymbol{x}_t$ for $t = 0.4$ by repeated application of corrector steps. In each pair of rows, the top row is $\boldsymbol{x}_t$ whilst the bottom row is the $\boldsymbol{x}_0$ prediction made by $p^{\theta}_{0|t}(\boldsymbol{x}_0|\boldsymbol{x}_t)$ (argmax of the categorical probabilities in each dimension). Each column represents an additional 100 corrector steps.

in discrete state spaces, the denoising model outputs a categorical distribution over every dimension (output dimension $D \times S$) allowing it to express some uncertainty information in the $\boldsymbol{x}_0$ prediction, albeit with conditional independence between the dimensions. Adding corrector steps in discrete state spaces would then allow information to mix between dimensions for the current time step, exploring modes of $q_t(\boldsymbol{x})$.

We explore this idea on the image modelling task in Figure 7. We run the reverse generative process until time $t = 0.4$ at which point we hold the time constant and apply 1000 corrector steps. We see that the resulting progression of $\boldsymbol{x}_t$ states explores potential local modes of $q_t(\boldsymbol{x})$ in the local region of image space.

G Implicit Dimensional Assumptions Made in Discrete Time

In discrete time, the parametric reverse kernel, $p^{\theta}_{k|k+1}$, is commonly defined through a denoising model $p^{\theta}_{0|k+1}$. Here, we examine this definition in the multi-dimensional case where the forward process factorizes, as in Appendix C.3 and previous discrete time work [8]. We begin by writing the true full dimensional reverse kernel, $q_{k|k+1}$, in terms of the true denoising distribution, $q_{0|k+1}$.

$$
\begin{aligned}
q_{k|k+1}(\boldsymbol{x}_k^{1:D}|\boldsymbol{x}_{k+1}^{1:D}) &= \prod_{d=1}^{D} q_{k|k+1}(x_k^d|\boldsymbol{x}_k^{1:d-1}, \boldsymbol{x}_{k+1}^{1:D}) \\
&= \prod_{d=1}^{D} \sum_{x_0^d} q_{k,0|k+1}(x_k^d, x_0^d|\boldsymbol{x}_k^{1:d-1}, \boldsymbol{x}_{k+1}^{1:D}) \\
&= \prod_{d=1}^{D} \sum_{x_0^d} q_{0|k+1}(x_0^d|\boldsymbol{x}_k^{1:d-1}, \boldsymbol{x}_{k+1}^{1:D}) q_{k|0,k+1}(x_k^d|x_0^d, x_{k+1}^d)
\end{aligned}
$$

where on the final line we have used the fact that the forward process is independent across dimensions. To create our approximate reverse kernel, $p^{\theta}_{k|k+1}$, we approximate $q_{0|k+1}(x_0^d|\boldsymbol{x}_k^{1:d-1}, \boldsymbol{x}_{k+1}^{1:D})$ with

$p^\theta_{0|k+1}(x_0^d|\boldsymbol{x}_{k+1}^{1:D})$,

$$p^\theta_{k|k+1}(\boldsymbol{x}_k^{1:D}|\boldsymbol{x}_{k+1}^{1:D}) = \prod_{d=1}^{D} \sum_{x_0^d} p^\theta_{0|k+1}(x_0^d|\boldsymbol{x}_{k+1}^{1:D}) q_{k|0,k+1}(x_k^d|x_0^d, x_{k+1}^d)$$

We throw away the extra $\boldsymbol{x}_k^{1:d-1}$ conditioning because we use a non-autoregressive model that takes in $\boldsymbol{x}_{k+1}^{1:D}$ and in a single forward pass gives conditionally independent probabilities over x_0^d, $d = 1, \ldots, D$. For finite K, this approximation can never match the true kernel because we are not conditioning on all relevant information. Of course, as K gets larger, this approximation becomes more accurate. Since we operate in the continuous regime, we do not have to make this approximation because the conditionally independent denoising model, $q_{0|t}(x_0^d|\boldsymbol{x}^{1:D})$, appears directly in our reverse rate, $\hat{R}_t^{1:D}$, when we factorize the forward process (see Proposition 3).

H Experimental Details

In this section, we provide additional details for the experiments we performed applying our method to practical problems. The code for our models is available at `https://github.com/andrew-cr/tauLDR`. Before describing the specifics for each experiment, we first explain the implementation details common to all.

When we evaluate the objective $\mathcal{L}_{\text{CT}}$ on each minibatch of training datapoints, we must sample a time for each from $t \sim \mathcal{U}(0, T)$ which represents the point in the forward process which we will noise to. Training instabilities can be found if t is sampled very close to 0 because the reverse rate, $\hat{R}_t$, becomes ill-conditioned in this region. This phenomenon is also observed in continuous state space models because the score, $\nabla_x \log q_t(x)$, becomes ill-conditioned close to $t = 0$. The reverse rate and score become ill conditioned close to the start of the forward process because the marginal probability, $q_t(x)$, will be highly peaked around the data manifold and $\log q_t(x)$ will explode in regions that are not close to the data. To avoid these issues, a common trick is to set a minimum time such that $t \sim \mathcal{U}(\epsilon, T)$. ϵ is set such that the level of noising at $t = \epsilon$ is very small and reverse sampling to this point will produce samples very close to p_{data}. In our experiments, we set $\epsilon = 0.01T$.

During reverse sampling, we use tau-leaping to simulate the reverse process from $t = T$ until $t = \epsilon$ because the reverse rate is not trained for $t < \epsilon$. This produces a sample close to p_{data}. We found improved performance in metrics such as FID if we then complete a final step to remove the small amount of noise that may still be present in the sample. Specifically, we pass the sample through the denoising model $p^\theta_{0|t}(x_0|x_t)$ with $t = \epsilon$ to obtain an output of shape $D \times S$ where D is the dimensionality of the problem. This is a probability distribution over the states for each of the dimensions. We set the value of each dimension to the state with the highest probability. This then produces a sample which has all of the noise removed.

The specific value of T within our model is arbitrary because the forward process can be scaled in the time axis to provide the same noising process for any T. Therefore, we simply set $T = 1$.

H.1 Demonstrative Example

Our 2d dataset is created by sampling 1M 2d points from a 32×32 state space with probability proportional to the pixel values of a 32×32 grayscale image of a τ character.

For our forward process, we use a Gaussian rate (see Appendix E) with stationary distribution standard deviation $\sigma_0 = 8$ and rate length scale $\sigma_r = 1$. We use a rate schedule of $\beta(t) = 5 \times 5^t \log(5)$.

To represent $p^\theta_{0|t}$ we use a residual MLP. The architecture consists of an input linear layer to lift the input dimension of 2 to the internal network dimension of 16. Then, there are 2 residual blocks each consisting of: a single hidden layer MLP of hidden dimension 32, a residual connection to the

input of the MLP, a layer norm, and finally a FiLM layer [35] modulated by the time embedding. At the output, there is a single linear layer with output size of $2 \times 32 = 64$ representing state probabilities in each of the 2 dimensions. The time is embedded using the Transformer sinusoidal position embedding [36] creating an embedding of size 32. Then, the embedding is further processed by a single hidden layer MLP with hidden layer size 32 and output size 128. To create the FiLM parameters in each residual block, the time embedding is passed through a linear layer with output of size 32 to provide a multiplicative and additive modulation to the state dimension of 16. We minimize the $\mathcal{L}_{\text{CT}}$ objective using Adam with a learning rate of 0.0001 and batch size of 32 for 1M steps.

For the exact simulation we use the next reaction method with modifications for time dependent transition rates [33]. This method steps through each transition in the exact simulation path individually by calculating the time to the next occurrence of each transition type and applying the transition that occurs soonest. Exact algorithmic details can be found in [33]. To calculate the time to the next occurrence for a transition, we need to integrate the reverse rate matrix (eq (13) in [33]). We do this with euler integration with a step size of 0.001.

H.2 Image Modeling

We train on the CIFAR10 training dataset that contains 50000 images of dimension $3 \times 32 \times 32$. We evaluate the test ELBO on the CIFAR10 test dataset which consists of 10000 images. For the forward noising process, we use the the Gaussian rate (see Appendix E) with stationary distribution standard deviation of $\sigma_0 = 512$ and rate length scale $\sigma_r = 6$. This effectively defines a uniform stationary distribution since the state space is of size 256. We found this performs better than a more concentrated Gaussian. Our β schedule is $\beta(t) = 3 \times 100^t \log 100$. This was selected in accordance with σ_r such that the overall shape of progression of the $q_{t|0}$ variances approximately matches that of the schedule proposed in [3].

Our $p^{\theta}_{0|t}$ model is parameterized with the standard U-net [37] architecture introduced in [3]. The network follows the PixelCNN++ backbone [38] with group normalization layers. There are four feature map resolutions (32×32 to 4×4) in the downsampling/upsampling stacks. At each resolution there are two convolutional residual blocks. There is a self-attention block between the residual blocks at the 16×16 resolution level [39]. The time is input into the network by first embedding with the Transformer sinusoidal position embedding [36]. This time embedding is passed into each residual block by passing it through a SiLU activation [40] and then a linear layer before adding it onto the hidden state within the residual block between the two convolution operations.

The original architecture of [3] has an output of dimension $3 \times 32 \times 32$ as it makes a point prediction of x_0 given x_t. In order for the model to output probabilities over x_0 (i.e. an output dimension of $3 \times 32 \times 32 \times 256$) we make the adjustments suggested in [8]. Specifically, we use their truncated logistic distribution parameterization where the model outputs the mean and log scale of a logistic distribution i.e. an output dimension of $3 \times 32 \times 32 \times 2$. The probability for a state is then the integral of this continuous distribution between this state and the next when mapped onto the real line. To impart a residual inductive bias on the output, the mean of the logistic distribution is taken to be $\tanh(x_t + \mu')$ where x_t is the normalized input into the model and μ' is mean outputted from the network. The normalization operation takes the input in the range $0, \ldots, 255$ and maps it to $[-1, 1]$. In total, our network has approximately 35.7 million parameters.

We optimize with the auxiliary objective described in Appendix D with $\lambda = 0.001$. Within the auxiliary objective, we use the one-forward pass version of the continuous time ELBO, $\mathcal{L}_{\text{eCT}}$. We optimize with Adam for 2M steps with a learning rate of 0.0002 and batch size of 128. We use the standard set of training tricks to improve optimization [3, 4]. Throughout training we maintain an exponential moving average of the parameters with decay factor 0.9999. These average parameters are used during testing. At the start of optimization we use a linear learning rate warm-up for the first 5000 steps. We clip the gradient norm at a norm value of 1.0. We set the dropout rate for the network at 0.1. The skip connections for each residual block are rescaled by a factor of $\frac{1}{\sqrt{2}}$. The input images have random horizontal flips applied to them during training.

For sampling in Table 1 we set $\tau = 0.001$ for τLDR-0 and set $\tau = 0.002$ for τLDR-10. The 10 corrector steps per predictor steps for τLDR-10 are introduced after $t < 0.1T$. We found that introducing the corrector steps near the end of the reverse sampling process had the best improvement in sample quality for the smallest increase in computational cost. When performing tau-leaping with the corrector rate, R_t^c, we have control over what τ we use since we are sampling a different CTMC (with q_t as its stationary distribution) to the original reverse CTMC. We found that setting the corrector rate τ to be 1.5 times the original τ for the reverse CTMC achieves the best performance in this example.

We train using 4 V100 GPUs on an academic research cluster. To calculate Inception and FID values, we use pytorch-fid [41] and a further development [1]. We verified this library produced comparable values to previous work by calculating the Inception and FID scores for the published images from the DDPM [3] method.

We show a large array of unconditional samples from the τLDR-10 model in Figure 8. We now also present statistics from the reverse sampling process with standard tau-leaping with $\tau = 0.001$. Figure 9 shows the proportion of dimensions that transition during a single step of tau-leaping. We see that during the initial stages, every dimensions changes during every tau-leaping step, but nearer the end of the process, more dimensions will have settled in their final positions and the proportion is less. In Figure 10 we show the proportion of dimensions that are clipped due to proposing an out of bounds jump. Overall, the proportion is small. It is largest at the start of the process when we have initially sampled from the approximately uniform p_{ref} and there will be dimensions close to the boundary. As pixel values settle to their final values, the proportion reduces. Figure 11 shows the progression of a selection of dimensions during the reverse sampling process. A similar picture emerges where dimensions eventually settle in a region of the state space. We also note that larger jumps are made in a single tau-leaping step nearer the start of the reverse process and smaller jumps are made nearer the end.

H.3 Monophonic Music

We generate our training dataset from the Lakh pianoroll dataset [29, 30] (license CC By 4.0). This dataset consists of 174,154 multitrack pianorolls. We go through all songs and all tracks within each song and select sequences that match the following criteria: they are monophonic (only one note played at a time), there is not a period longer than one bar in which no note is played, there is more than one type of note played in the sequence and finally there is no one note played for more than 50 time steps out of the total 256 time steps. This removes the uninteresting and trivial sequences present within the dataset. We then remove any duplicates in the result. This leaves us with 6000 training examples and 950 testing examples. Each song consists of 256 time steps (16 per bar) and each time step takes one from 129 values i.e. we have $D = 256$ and $S = 129$. This state value represents either a note from 128 options or a rest. We scramble the ordering of this state space when mapping to the integers from 0 to 128. When we input into the denoising network, we input as one-hot 129 dimensional vectors.

For the forward noising process, we use a uniform rate matrix, $R_b = \mathbb{1}\mathbb{1}^T - S\text{Id}$ and set $\beta(t) = 0.03$. We found a constant in time $\beta(t)$ was sufficient for this dataset. In our comparison, we used a birth/death rate matrix defined as

$$\begin{bmatrix} -\lambda & \lambda & 0 & 0 & \dots & 0 & 0 \\ \lambda & -2\lambda & \lambda & 0 & \dots & 0 & 0 \\ 0 & \lambda & -2\lambda & \lambda & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \dots & \lambda & -\lambda \end{bmatrix}$$

this is the rate matrix for a birth/death process. We set $\lambda = 1$ and $\beta(t) = \frac{1}{2} \times 10000^t \log 10000$. These hyperparameters were selected such that the forward process has a steady rate of noising whilst still having q_T very close to p_{ref}. We chose to compare these types of rate matrix because the birth/death rate is inappropriate for this categorical data as adjacent states have no meaning

[1] https://github.com/w86763777/pytorch-gan-metrics

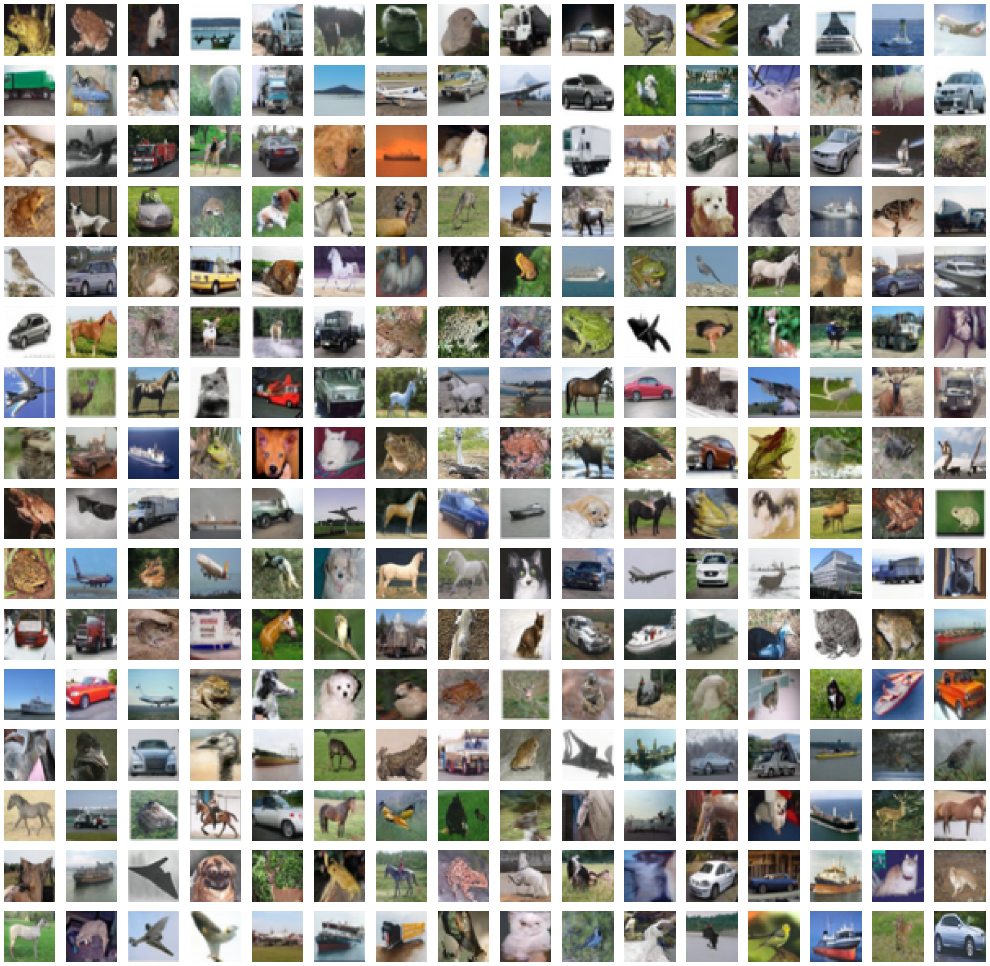

Figure 8: Unconditional CIFAR10 samples from our τLDR-10 model.

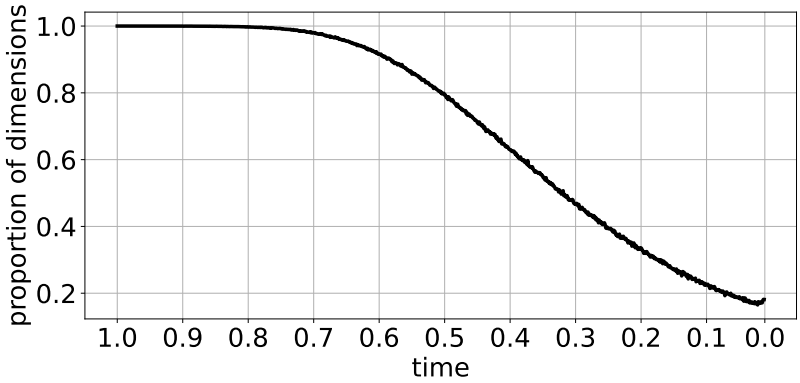

Figure 9: Proportion of dimensions that transition during a single step of tau-leaping during the reverse sampling process.

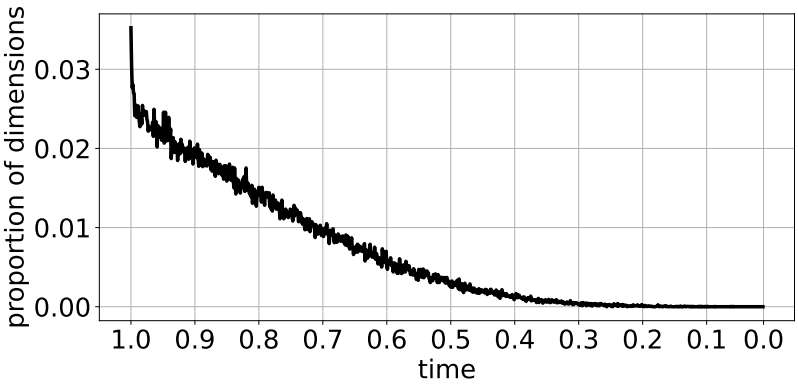

Figure 10: Proportion of dimensions that are clipped during a tau-leaping step due to proposing an out of bounds jump during the reverse sampling process.

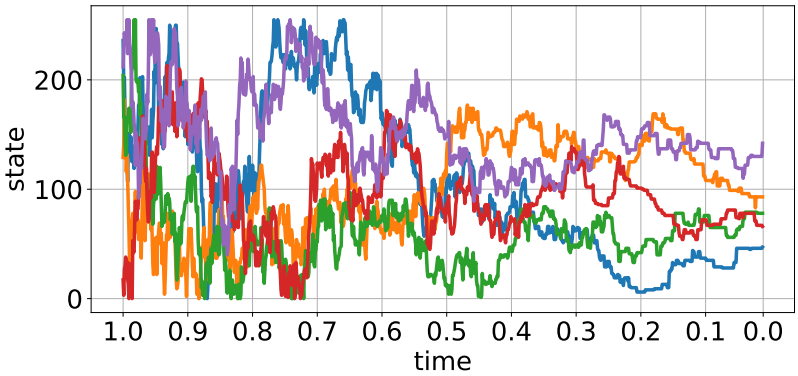

Figure 11: The progression of a selection of dimensions during the reverse sampling process.

since the mapping to the integers was arbitrary. The uniform rate is suitable for this categorical data because, during a time interval, it has a uniform probability to transition to any other state. The D3PM baseline was implemented also with a time homogeneous uniform forward kernel set such that the rate of noising is matched in the discrete and continuous time cases.

We define our conditional denoising network, $p^{\theta}_{0|t}(x_0|x, y)$ using a transformer architecture inspired by [31]. It takes an input of shape (B, D, S) where B is the batch size, D is the dimensionality (256) and S is the state size (129). This final dimension contains the one-hot vectors. The conditioning on the initial bars is achieved by concatenating the conditioning information y with the noisy input x. At the start of the network, there is an input embedding linear layer with output of size 128 which is our model dimension for the transformer. Then a transformer positional embedding is added to the hidden state. Next a stack of 6 transformer encoder layers are applied which consist of a self attention block and a one hidden layer MLP. The self attention block uses 8 heads and the MLP has a hidden layer size of 2048. At the output of each internal block, we apply dropout with rate 0.1. Finally, there is a stack of 2 residual MLP layers. Each consists of a one hidden layer MLP with a hidden dimension of 2048. There is a residual connection between the input and output of the MLP. A layer norm is applied to the output of the block. To create the output of the network, there is an output linear layer with an output shape of (B, D, S) where now the S dimension has logit probabilities. To instill a residual bias into the network, we add the one-hot input to the output logits. All activations are ReLU. The time is input into the network through FiLM layers [35]. First, the time is embedded using the sinusoidal transformer position embedding as in the U-net architecture used for image modeling to create an embedding size of 128. This is then passed into a single hidden layer MLP with hidden size

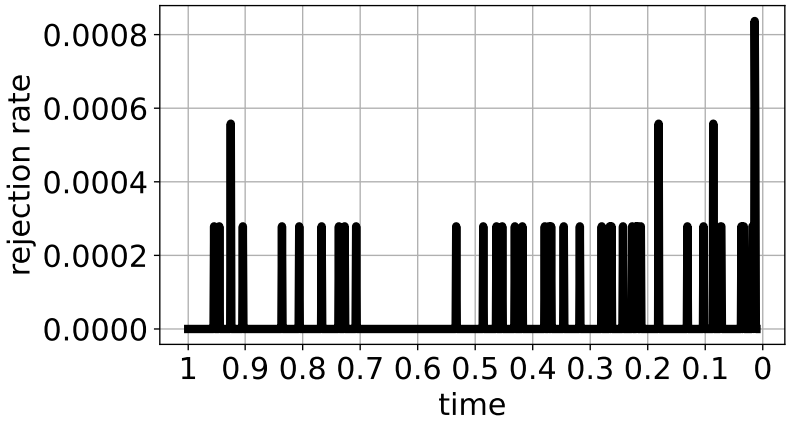

Figure 12: Proportion of jumps rejected during reverse sampling. The rejection rate is calculated as the proportion of dimensions in a tau leaping step that have their jump rejected. The results are averaged over a batch of 16.

2048 and output size 512. Within each encoder and residual MLP block, there is a FiLM linear layer which takes in the 512 time embedding and outputs two FiLM parameters each of size 128. These are the scale and offset applied to the hidden state. In the encoder blocks, this FiLM transform is applied after the self attention block and again after the fully connected block. In the residual MLP blocks, it is applied after the layer norm operation. Our network has approximately 7 million parameters in total.

We optimize using Adam for 1M steps with a batch size of 64 and learning rate of 0.0002. We use the conditional $\bar{\mathcal{L}}_{\text{CT}}$ objective with additional direct $p^{\theta}_{0|t}$ supervision as described in Appendix D with weight $\lambda = 0.001$. We also make the same one forward pass approximation as explained in Appendix C.4. We use the standard set of training tricks to improve optimization [3, 4]. Throughout training we maintain an exponential moving average of the parameters with decay factor 0.9999. These average parameters are used during testing. At the start of optimization we use a linear learning rate warm-up for the first 5000 steps. We clip the gradient norm at a norm value of 1.0. We train on a single V100 GPU on an academic cluster.

For sampling with τLDR-0 we use $\tau = 0.001$ and for sampling with τLDR-2 we include 2 corrector steps per predictor step after $t < 0.9T$. The corrector rate is simulated with $\tau = 0.0001$ which we found to perform best. We reject any dimension in which 2 or more jumps are proposed as this is categorical data. We plot the rejection rate in Figure 12. Most of the time, the rejection rate is zero and there are few steps for which it increases slightly. We show a large batch of samples from the first 10 songs in the test dataset in Figure 13. We see that there is variation between the sampled completions and they consistently follow the style of the conditioning first two bars of the song. Audio samples from the model are available at `https://github.com/andrew-cr/tauLDR`. Finally, we examine the progression of a random selection of dimensions during reverse sampling for the uniform and birth/death rate matrix cases. Figure 14 shows the progression for the uniform case, we see that large jumps through the state space are made throughout the reverse process. Figure 15 shows the progression for the birth/death case. At the start of reverse sampling, no dimensions move as the rejection rate is high in this case because the rate matrix is not suitable for categorical data. Nearer the end, small jumps are made between adjacent states but since large jumps between any category do not occur for this rate matrix, the performance will overall be worse.

I Ethical Considerations

Our work increases our theoretical understanding of denoising generative models and also improves generation capabilities within some discrete datasets. Deep generative models are generic methods

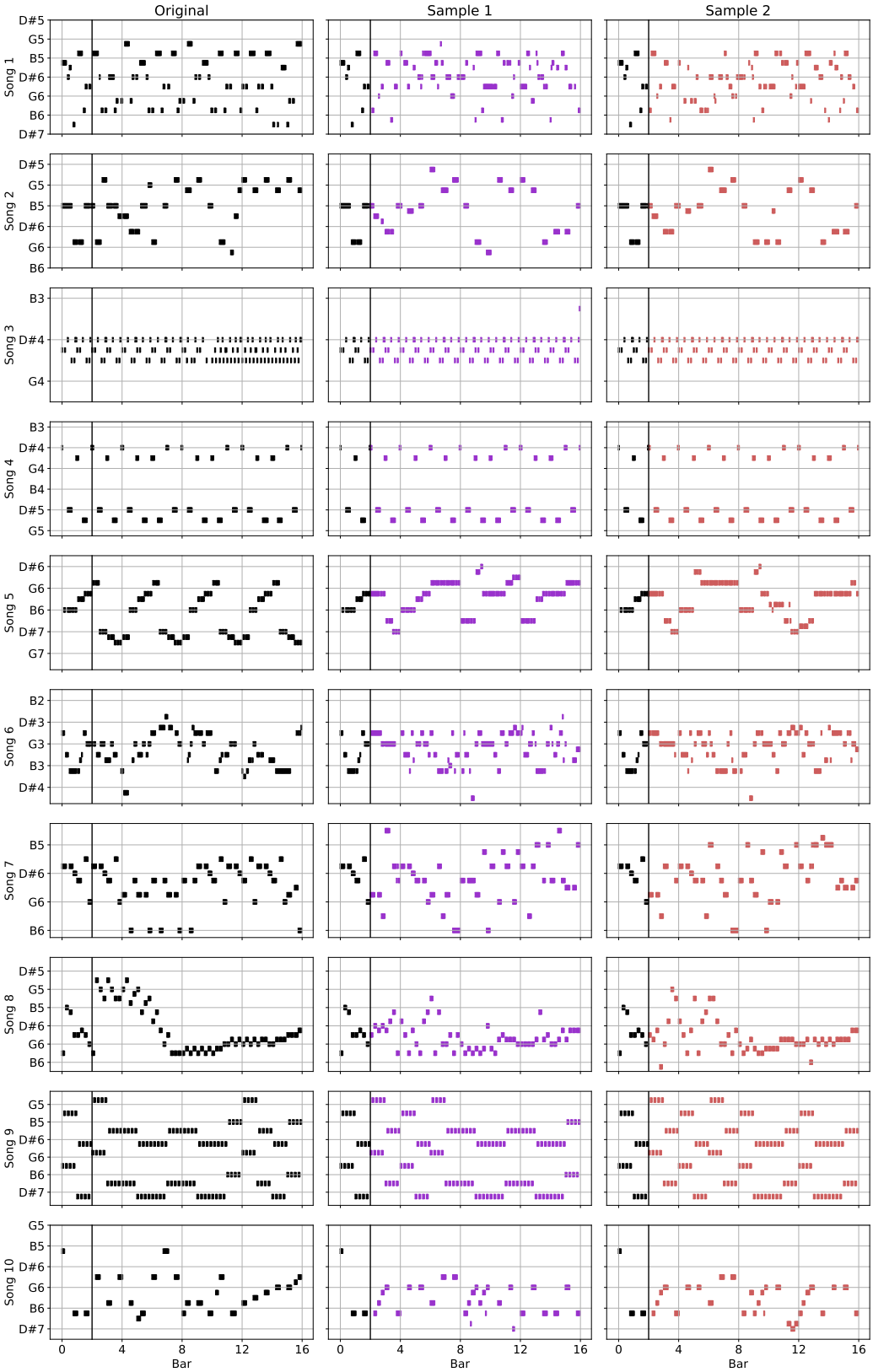

Figure 13: Two conditional samples from the τLDR-0 model for each of the first 10 songs in the test dataset.

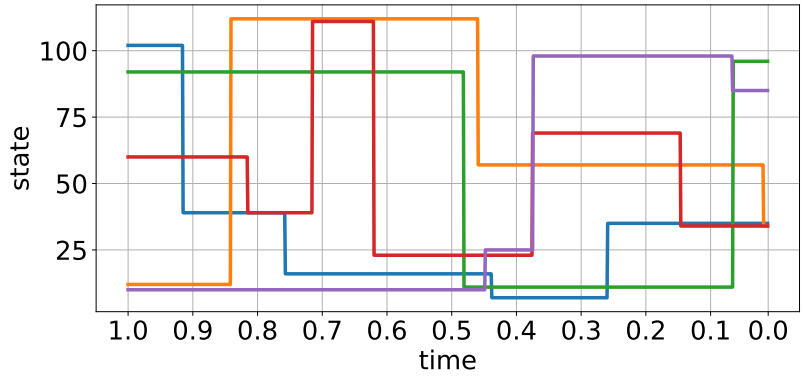

Figure 14: The progression of a selection of dimensions during the reverse sampling process for the uniform rate matrix.

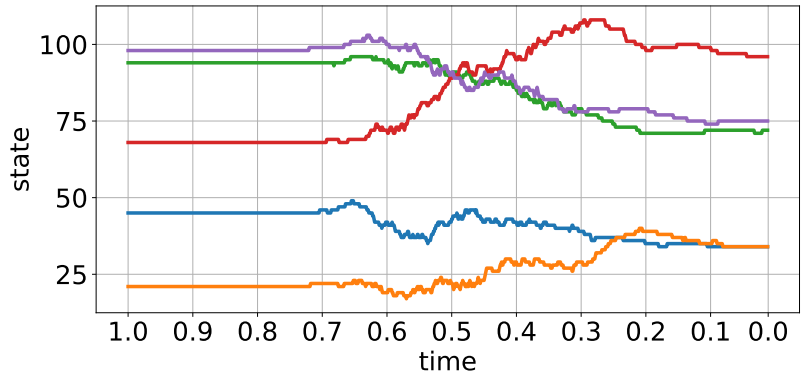

Figure 15: The progression of a selection of dimensions during the reverse sampling process for the birth/death rate matrix.

for learning from unstructured data and can have negative social impacts when misused. For example, they can be used to spread misinformation by reducing the resources required to create realistic fake content. Furthermore, generative models will produce samples that accurately reflect the statistics of their training dataset. Therefore, if samples from these models are interpreted as an objective truth without fully considering the biases present in the original data, then they can perpetuate discrimination against minority groups.

In this work, we train on datasets that contain less sensitive data such as pictures of objects and music samples. The methods we presented, however, could be used to model images of people or text from the internet which will contain biases and potentially harmful content that the model will then learn from and reproduce. Great care must be taken when training these models on real world datasets and when deploying them so as to mitigate and prevent the harms that they can cause.