Diffusion in the Probability Simplex

Griffin Floto, Eric Zhu, Kristin (Xi Yu) Huang
Department of Computer Science
University of Toronto
{griffin.floto, e.zhu, xiyu.huang}@mail.utoronto.ca

Abstract

This paper presents a novel approach to perform diffusion on categorical data with continuous dynamics. Diffusion models have been widely used in machine learning, where they typically operate in the continuous regime. One exception is categorical diffusion models, which use discrete state transition matrices, resulting in discontinuous dynamics. In this work, we propose a method to perform diffusion on the probability simplex, which enables us to operate on categorical data while maintaining continuous dynamics. Furthermore, our method naturally enables the notion of uncertainty over sampled states, unlike previous categorical diffusion models. We demonstrate a proof-of-concept of our method on the MNIST dataset and compare to the discrete transition categorical diffusion model. Our method opens up new avenues for using diffusion models in applications that involve categorical data.

1 Introduction

Diffusion models are a type of generative model that generates data from probability distributions. Sohl-Dickstein et al. (2015) The fundamental idea is to transform input data into noise by systematically adding noise in each iterative forward diffusion process. This transforms a complex distribution into a distribution that is easy to sample from. Afterwards, a parameterized reverse process is learned which defines a generative model. Diffusion models has applications in the image Dhariwal & Nichol (2021), speech Jeong et al. (2021), and video Singer et al. (2022) generation domains, largely due to continuous diffusion model's ability to represent the continuous space of those domains.

Most recent studies are focused on continuous diffusion using Gaussian probability distributions. Categorical diffusion works in the discrete space, and has been shown to be more applicable to language modeling Li et al. (2022), image segmentation Hoogeboom et al. (2021), and reinforcement learning Janner et al. (2022) domains. Diffusion in the discrete space was initially explored as a diffusion process over binary random variables Sohl-Dickstein et al. (2015), and was later expanded to selecting specific forward transition matrices to corrupt data for different tasks Austin et al. (2023). In our work, we propose a simpler and more flexible method by using score matching on the probability simplex.

2 Background

2.1 Score Matching and Diffusion

Score matching as formulated by Song et al. (2021) considers a continuous time diffusion process. Typically, the process does not have parameters and does not depend on the data. The objective of the method is to learn how to reverse the process to create a generative model. In the forward process, data is perturbed using a stochastic differential equation (SDE) of the following form:

Preprint. Under review.

$$d\boldsymbol{x} = \boldsymbol{f}(\boldsymbol{x}, t)dt + g(t)d\boldsymbol{w}. \tag{1}$$

where $\boldsymbol{w}$ is the standard Weiner process (also know as Brownian motion), $\boldsymbol{f}(\boldsymbol{x}, t)$ is called the drift term and $g(t)$ is called the diffusion coefficient. This process maps our data distribution, $p_{t=0}(\boldsymbol{x})$ into some limiting distribution $p_{t=1}(\boldsymbol{x})$. The reverse solution to this SDE is given by

$$d\boldsymbol{x} = \left[\boldsymbol{f}(\boldsymbol{x}, t) - g(t)^2 \nabla_x \log p_t(\boldsymbol{x})\right] dt + g(t)d\boldsymbol{w} \tag{2}$$

where time now flow backwards from $t = 1$ to $t = 0$. Given this equation, we then seek to approximate the score $\nabla_x \log p_t(\boldsymbol{x})$ to create a generative model. The score can be approximated by a score based model $\boldsymbol{s}_\theta(\boldsymbol{x}, t)$ which can be optimized via

$$\theta^* = \text{argmin}_\theta \mathbb{E}_{u\sim U[0,1]} \mathbb{E}_{x_0 \sim p_0(x)} \mathbb{E}_{x_t \sim p_{0t}(x_t|x_0)} \lambda(t) \left[\|\boldsymbol{s}_\theta(\boldsymbol{x}, t) - \nabla_{x_t} \log p_{0t}(\boldsymbol{x}_t|\boldsymbol{x}_0)\|_2^2\right] \tag{3}$$

where $\lambda(t)$ is a weighting function. A common practice when using diffusion models is to discretize time into uniform steps.

2.2 Categorical Diffusion

When working with categorical data, a slightly different approach is taken. Time is discretized into T steps and a Markov transition matrix is used for the forward process Austin et al. (2023). The transition matrix

$$[Q_t]_{ij} = q(x_t = j | x_{t-1} = i)$$

is applied independently to each data dimension to slowly add noise. The goal of categorical diffusion models is to predict the inverse transition matrices, which are no longer independent. The loss reduces to a sum of Kullback–Leibler divergences between categorical distributions at each discrete time step. We use a uniform transition matrix $Q_t = (1 - \beta_t)I + \beta_t / K \mathbb{1}\mathbb{1}^T$ with $\beta_t \in [0, 1]$ and K equal to number of categories.

3 Method

3.1 The Logit-Normal Distribution on the Probability Simplex

The probability simplex can be defined by the set of points $\{x \in \mathbb{R}^k : x^\top \mathbb{1} = 1\}$ which we denote by $\mathcal{S}^k$. By noting the property that $\sum_{i=1}^k x_i = 1$ we can use points in the probability simplex to represent categorical probability distributions.

The logistic-normal distribution is one example of a probability distribution over the probability simplex. It is defined as the probability distribution of a random variable whose multinomial logit is a normal distribution. The probability density function of the logistic normal is:

$$p(\boldsymbol{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{|(2\pi)^{d-1}\boldsymbol{\Sigma}|} \frac{1}{\prod_{i=1}^d x_i} \exp\left(-\frac{1}{2}\left[\log\left(\frac{\bar{\boldsymbol{x}}_d}{x_d}\right) - \boldsymbol{\mu}\right]^\top \boldsymbol{\Sigma}^{-1} \left[\log\left(\frac{\bar{\boldsymbol{x}}_d}{x_d}\right) - \boldsymbol{\mu}\right]\right),$$

where $\boldsymbol{x} \in \mathcal{S}^d$ and $\bar{\boldsymbol{x}}_d = [x_1, \ldots, x_{d-1}]$

To map a point $\boldsymbol{y} \in \mathbb{R}^{d-1}$ to a point in the probability simplex $\boldsymbol{x} \in \mathcal{S}^d$ we can use the additive logistic transformation:

$$x_i = \begin{cases} \frac{e^{y_i}}{1+\sum_{k=1}^{d-1} e^{y_k}}, & \text{if } i \in \{1, \ldots, d-1\} \\ 1 - \sum_{i=1}^{d-1} x_i = \frac{1}{1+\sum_{k=1}^{d-1} e^{y_k}}, & \text{if } i = d \end{cases}$$

Conversely, the unique inverse map from $\mathcal{S}^d$ to $\mathbb{R}^{d-1}$ is

$$y_i = \log\left[\frac{x_i}{x_d}\right], i \in \{1, \ldots, d-1\}.$$

3.2 The Ornstein-Unlenbeck Process

The Ornstein-Unlenbeck (OU) process is real-valued stochastic process used in financial mathematics and physical sciences. Originally, it was developed to model the velocity of a Brownian particle under the force of friction. The process can be described by the following stochastic differential equation:

$$dx_t = -\theta x_t dt + \sigma dW_t$$

where $\theta > 0$ and $\sigma > 0$ are parameters and W_t is the Weiner process. The process had the exact solution

$$O_t := \mathcal{N}\left(O_0 e^{-\theta t}, \frac{1}{2\theta}\left(1 - e^{-2\theta t}\right)\right).$$

In the limit as $t \to \infty$ the process has a distribution of $\mathcal{N}\left(0, \frac{1}{2\theta}\right)$, meaning that θ uniquely determines the limiting distribution.

3.3 Diffusion on the Probability Simplex

We now describe a process that can be used in the score matching framework to do diffusion in the probability simplex. A new process can be defined by taking the sigmoid (or more generally the additive logistic transformation) of an OU process as follows:

$$X_t = \sigma(O_t).$$

In our case we are able to get an exact solution for S_t by pushing forward the solution of the OU process, meaning that $X_t \sim \sigma\left[\mathcal{N}\left(O_0 e^{-\theta t}, \frac{1}{2\theta}\left(1 - e^{-2\theta t}\right)\right)\right]$. At each point t the resulting distribution is a logistic-normal. By assuming independence between variables (meaning $\boldsymbol{\Sigma} = \boldsymbol{I}$) we can sample to get $p_{0t}(\boldsymbol{x}_t|\boldsymbol{x}_0)$, which is required in 3.

In order to train the score-matching model, we must also have a closed form solution of $p_{0t}(\boldsymbol{x}_t|\boldsymbol{x}_0)$, which we show in Appendix A.1. The results of the derivation is that the score of the logistic-normal distribution is

$$\nabla_x \log p(\boldsymbol{x})_i = \begin{cases} \frac{\log\left[\frac{x_i}{x_d}\right] - \mu + v}{v x_i}, i \in \{1, \ldots, d-1\}, & \text{if } i \in \{1, \ldots, d-1\} \\ -\frac{v + (d-1)(\sum_{i=1}^{d-1} \log\left[\frac{x_i}{x_d}\right] - \mu)}{v x_d}, & \text{if } i = d \end{cases}$$

Finally, we require a process of the form 1 in order to train and sample from a score-based model. By using Itô's lemma, we can write a closed form solution for the sigmoid OU process (see Appendix A.2) as follows:

$$dX_t = \left(-\theta \sigma^{-1}(X_t) X_t (1 - X_t) + \frac{1}{2} X_t (1 - X_t)(1 - 2X_t)\right) dt + X_t (1 - X_t) dB_t.$$

4 Results

4.1 Experiment and Settings

We evaluated our score-matching in the probability simplex on the MNIST digit dataset and compared to a categorical diffusion baseline method. For a basic proof of concept we used binary MNIST as the

dataset. We used the same UNet to compare both methods, and ran both models until convergence with identical training parameters. The parameters we chose for the OU diffusion process are: $O_0 = 6, \theta = 8$ and $t \in [0.075, 0.7]$ which result in a well-shaped diffusion process.

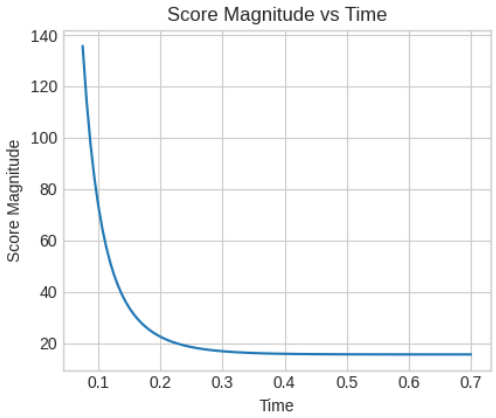

Figure 1: Average Score at $\pm\sigma$ During Process

To implement our score-matching model, we took into consideration the score profile during the diffusion process, see Figure 1. The average score magnitude at one standard deviation varies significantly during the process. To deal with this, we re-scale the target score values, such that they are all on the same order during training. When sampling, we then re-scale to the appropriate value and use a second order Runge-Kutta method to balance accuracy and compute. Example of outputs are shown in Figure 2.

(a) Categorical model (b) Logit-Normal model

Figure 2: Visualization of MNIST samples

4.2 Discussion and Analysis

We observe that the proposed method works to a limited degree on the MNIST dataset, demonstrating a basic proof of concept. The categorical model performs better, however our proposed method has not yet benefited from detailed analysis and parameter tuning.

Our proposed simplex diffusion model suffers from difficulty in the final $\frac{1}{5}$ of the decoding process. The results shown in Figure 2 are the output of $\frac{4}{5}$th of the sampling before the process becomes unstable. An example output of the full process can be visualized in 3. While re-weighting the scores had the benefit of stable training, we suspect that is the cause of the poor performance in this region. Error in the score in regions with high magnitude result in larger errors when solving the backwards SDE, yet this is not reflected in the way we train the model. In the future, we would seek to perform importance sampling Dieleman et al. (2022) such that this critical region is sampled more often during training.

Figure 3: Decoding process comparison

References

Jacob Austin, Daniel D. Johnson, Jonathan Ho, Daniel Tarlow, and Rianne van den Berg. Structured denoising diffusion models in discrete state-spaces, 2023.

Prafulla Dhariwal and Alex Nichol. Diffusion models beat gans on image synthesis, 2021.

Sander Dieleman, Laurent Sartran, Arman Roshannai, Nikolay Savinov, Yaroslav Ganin, Pierre H. Richemond, Arnaud Doucet, Robin Strudel, Chris Dyer, Conor Durkan, Curtis Hawthorne, Rémi Leblond, Will Grathwohl, and Jonas Adler. Continuous diffusion for categorical data, 2022.

Emiel Hoogeboom, Didrik Nielsen, Priyank Jaini, Patrick Forré, and Max Welling. Argmax flows and multinomial diffusion: Learning categorical distributions, 2021.

Michael Janner, Yilun Du, Joshua B. Tenenbaum, and Sergey Levine. Planning with diffusion for flexible behavior synthesis, 2022.

Myeonghun Jeong, Hyeongju Kim, Sung Jun Cheon, Byoung Jin Choi, and Nam Soo Kim. Diff-tts: A denoising diffusion model for text-to-speech, 2021.

Xiang Lisa Li, John Thickstun, Ishaan Gulrajani, Percy Liang, and Tatsunori B. Hashimoto. Diffusion-lm improves controllable text generation, 2022.

Uriel Singer, Adam Polyak, Thomas Hayes, Xi Yin, Jie An, Songyang Zhang, Qiyuan Hu, Harry Yang, Oron Ashual, Oran Gafni, Devi Parikh, Sonal Gupta, and Yaniv Taigman. Make-a-video: Text-to-video generation without text-video data, 2022.

Jascha Sohl-Dickstein, Eric A. Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics, 2015.

Yang Song, Jascha Sohl-Dickstein, Diederik P. Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations, 2021.

5 Contributions

We had three authors in this paper. We all worked on various parts of the project in a collaborative manner, e.g., model building, tuning, writing the report, etc. While all of our contributions were important, Griffin was the team lead, both in terms of heading a lot of the derivations, and initial code of the project. Overall, we consider contributions for the sake of the project to be essentially equal.

A Appendix

A.1 Score Derivation

A.1.1 Score Derivation in 1D

We require the gradient of the log logistic Gaussian distribution

$$p(x) = \frac{1}{\sqrt{2\pi v}} \frac{1}{x(1-x)} \exp\left(-\frac{(\sigma^{-1}(x)-\mu)^2}{2v}\right)$$

where $\sigma^{-1} = \log\left(\frac{x}{1-x}\right)$

We are interested in $\nabla_x \log p(x)$, or for the time being $\frac{\partial}{\partial x} \log p(x)$. After working in 1D, we will then show the general case. First we deal with the log prob:

$$\begin{aligned}\frac{\partial}{\partial x} \log p(x) &= \log\left[\frac{1}{\sqrt{2\pi v}} \frac{1}{x(1-x)} \exp\left(-\frac{(\sigma^{-1}(x)-\mu)^2}{2v}\right)\right] \\ &= C + \log\left[\frac{1}{x(1-x)}\right] - \frac{(\sigma^{-1}(x)-\mu)^2}{2v}\end{aligned}$$

where $C = \log\left[\frac{1}{\sqrt{2\pi v}}\right]$. Next, we can then differentiate each of the components separately. The first can be solved as

$$\begin{aligned}\frac{\partial}{\partial x}\log\left[\frac{1}{x(1-x)}\right] &= \frac{\partial}{\partial x}\log\left[\frac{1}{x}\right] + \frac{\partial}{\partial x}\log\left[\frac{1}{1-x}\right] \\ &= -\frac{\partial}{\partial x}\log[x] - \frac{\partial}{\partial x}\log[1-x] \\ &= -\frac{1}{x} + \frac{1}{1-x}\end{aligned}$$

and the second can be solved as

$$\begin{aligned}\frac{\partial}{\partial x}\frac{(\sigma^{-1}(x)-\mu)^2}{2v} &= -\frac{\partial}{\partial x}\frac{(\log\left[\frac{x}{1-x}\right]-\mu)^2}{2v} \\ &= -\frac{\log\left[\frac{x}{1-x}\right]-\mu}{v}\frac{\partial}{\partial x}\left(\log\left[\frac{x}{1-x}\right]-\mu\right) \\ &= -\frac{\log\left[\frac{x}{1-x}\right]-\mu}{vx(1-x)}\end{aligned}$$

Putting it all together, we get:

$$\begin{aligned}\frac{\partial}{\partial x}\log p(x) &= \frac{1}{1-x} - \frac{1}{x} + \frac{\mu - \log\left[\frac{x}{1-x}\right]}{vx(1-x)} \\ &= \frac{2vx + \mu - v - \log\left[\frac{x}{1-x}\right]}{vx(1-x)} \\ &= \frac{\sigma^{-1}(x) - 2vx - \mu + v}{vx(x-1)}\end{aligned}$$

A.1.2 Score Derivation in General

The general logistic-normal distribution can be written as:

$$p(x) = \frac{1}{Z}\frac{1}{\prod_{i=1}^{d} x_i}\exp\left(-\frac{\|\log\left[\frac{\bar{x}_d}{x_d}\right]-\mu\|_2^2}{2v}\right)$$

where $x \in \mathcal{S}^d$ and $\bar{x}_d = [x_1, \ldots, x_{d-1}]$. We assume that the Gaussian has covariance $\Sigma = \sqrt{v}I$. To bring x from the simplex back to $\mathbb{R}^{d-1}$ we can use:

$$y_i = \log\left[\frac{x_i}{x_d}\right], i \in \{1, \ldots, d-1\}$$

The inverse transformation of this is:

$$x_i = \frac{e^{y_i}}{1 + \sum_{k=1}^{d-1} e^{y_k}}, i \in \{1, \ldots, d-1\}$$

$$x_d = \frac{1}{1 + \sum_{k=1}^{d-1} e^{y_k}} = 1 - \sum_{i=1}^{d-1} x_i$$

Overall, we want to calculate: $\nabla_x \log p(x)$

Following the same process as the 1D case:

$$\log p(x) = -\log [Z] - \log \left[\prod_{i-1}^{d} x_i\right] - \frac{1}{2v}\|\log \left[\frac{\bar{x}_d}{x_d}\right] - \mu\|_2^2$$

We deal with the gradients, starting with the second term (the first one has no gradient). The fact that $\log \left[\prod_{i-1}^{d} x_i\right] = \sum_{i=1}^{d} \log [x_i]$ will be used in the following:

$$g := -\nabla_x \log \left[\prod_{i=1}^{d} x_i\right]$$
$$g_i = -\frac{1}{x_i}$$

Next we look at the rightmost term in the equation

$$\nabla_x \left[-\frac{1}{2v}\|\log \left[\frac{\bar{x}_d}{x_d}\right] - \mu\|_2^2\right] := -\frac{1}{2v}\nabla_x \alpha := h$$

$$\nabla_x \alpha = f$$

Now for $i \in \{1, \ldots, d-1\}$ the following holds:

$$\begin{aligned} f_i &= \frac{\partial}{\partial x_i}\left(\log[\frac{x_i}{x_d}] - \mu\right)^2 \\ &= 2\left(\log[\frac{x_i}{x_d}] - \mu\right)\frac{1}{x_i} \\ h_i &= -\frac{\left(\log[\frac{x_i}{x_d}] - \mu\right)}{v x_i} \end{aligned}$$

The last component is more complex to calculate.

$$\begin{aligned} \frac{\partial}{\partial x_d}\alpha &= \frac{\partial}{\partial x_d}\sum_{i=1}^{d-1}\left(\log\left[\frac{x_i}{x_d}\right] - \mu\right)^2 \\ &= 2\sum_{i=1}^{d-1}\left(\log\left[\frac{x_i}{x_d}\right] - \mu\right)\frac{\partial}{\partial x_d}\left(\log\left[\frac{x_i}{x_d}\right]\right) \\ &= 2\sum_{i=1}^{d-1}\left(\log\left[\frac{x_i}{x_d}\right] - \mu\right)\left(-\frac{1}{x_c}\right) \\ h_d &= \frac{d-1}{v x_d}\left(\sum_{i=1}^{d-1}\log\left[\frac{x_i}{x_d}\right] - \mu\right) \end{aligned}$$

In summary, we can then write:

$$\begin{aligned} \gamma &:= \nabla_x \log p(x) \\ \gamma_i &= -\frac{\log[\frac{x_i}{x_d}] - \mu + v}{v x_i}, i \in \{1, \ldots, d-1\} \\ \gamma_d &= -\frac{v + (d-1)(\sum_{i=1}^{d-1}\log[\frac{x_i}{x_d}] - \mu)}{v x_d} \end{aligned}$$

A.2 Ito's Lemma Application

To review the OU process can be written as $dx_t = -\theta x_t dt + \sigma dW_t$ and the distribution can be written as $O_t := \mathcal{N}\left(O_0 e^{-\theta t}, \frac{1}{2\theta}\left(1 - e^{-2\theta t}\right)\right)$. We are interested in the following stochastic process: $X_t = \sigma(O_t)$.

By using Ito's lemma, we can write the SDE of X_t as:

$$
\begin{aligned}
dX_t &= \sigma'(O_t)dO_t + \frac{1}{2}\sigma''(O_t)(dO_t)^2 \\
&= \sigma'(O_t)(-\theta O_t dt + dB_t) + \frac{1}{2}\sigma''(O_t)dt \\
&= \left(-\theta O_t \sigma'(O_t) + \frac{1}{2}\sigma''(O_t)\right)dt + \sigma'(O_t)dB_t \\
&= \left(-\theta \sigma^{-1}(X_t)X_t(1 - X_t) + \frac{1}{2}X_t(1 - X_t)(1 - 2X_t)\right)dt + X_t(1 - X_t)dB_t
\end{aligned}
$$