DISCRETE FEYNMAN-KAC CORRECTORS

Anonymous Authors[1]

Abstract

The performance of Large Language Models (LLMs) directly depends on the size of the context that the model was trained on. Despite significant progress in increasing the context size of the current models, some applications remain bottlenecked by the number of processed tokens at inference time. A particular mathematical problem LLMs can be used for is inferring parameters in a statistical model, given data-points as input. Here we make a case demonstrating that discrete diffusion models offer a promising avenue for scaling such parameter prediction tasks, by combining the outputs of the same model evaluated on different parts of the training data. We propose DISCRETE FEYNMAN-KAC CORRECTORS— a framework that allows for controlling the generated distribution of discrete masked diffusion models at inference time. We derive Sequential Monte Carlo (SMC) algorithms that, given a trained discrete diffusion model, sample from its annealed distribution or the product of distributions with different conditions. Notably, our framework does not require any training, finetuning and external reward functions. Finally, we apply our framework to amortized linear regression using LLaDA and demonstrate that it drastically outperforms the standard inference procedure in terms of accuracy and adherence to prompt format.

1. Introduction

The success of diffusion models in continuous domains, such as the generation of images (Rombach et al., 2022), videos (Wang et al., 2023; Blattmann et al., 2023), or 3D protein structures (Abramson et al., 2024; Watson et al., 2023), has motivated their application to discrete data spaces. Indeed, modeling discrete data such as text or biological sequences using diffusion processes is a promising direction since they do not rely on sequential token generation as with autoregressive models, which can impose arbitrary orderings on data (e.g., molecular structures and protein sequences (Lee et al., 2025b; Alamdari et al., 2023)), or can suffer from exposure biases that limit long-horizon planning or reversal reasoning in natural language domains (Berglund et al., 2023; Nie et al., 2025).

Discrete diffusion models can be formulated as a continuous-time Markov process that progressively transforms data to noise through a series of random transitions, and then learns to reverse the process and recover the original data (Campbell et al., 2022; Lou et al., 2023; Sahoo et al., 2024; Shi et al., 2024). While this process learns the unconditional distribution of the data, it is crucial to be able to control generations based on user desiderata (for example, conditioning on desired target properties of a protein (Gruver et al., 2023)) or optimize outputs based on downstream objectives (for example, sampling from modes). Several notable contributions have developed methods to approximately sample from conditional distributions, requiring access to external classifiers(Vignac et al., 2022; Nisonoff et al., 2024; Tang et al., 2025) or correction schemes (Nisonoff et al., 2024; Gruver et al., 2023). Lee et al. (2025a) derived the exact transition rates needed to sample from the tempered conditional distribution ($p_{t,\beta}^{\text{temp cond}}(x) \propto q_t(x)q_t(y|x)^\beta$) using a Sequential Monte Carlo (SMC) resampling scheme. More recently, methods based on external reward models (Rector-Brooks et al., 2024) have been proposed to improve the quality of generated samples from discrete diffusion models, similar to reinforcement learning from human feedback (RLHF) approaches in Large Language Model (LLM) settings. In general, LLMs an extensive history of improving sample quality with a wide range of approaches. The simplest ones to implement involve in-context methods, which use prompting strategies like chain-of-thought reasoning (Wei et al., 2022; Imani et al., 2023) to improve sample quality. However, these approaches require manual curation, prohibiting generalizability. Recent trends have moved towards automated prompt optimization (Fernando et al., 2023) or fine-tuning the entire model using RL or supervised fine-tuning (DeepSeek-AI, 2025).

The quality of text generated by discrete diffusion models is still quite far from the best modern LLMs (Nie et al., 2025). While the gap will likely close with scaling and fine-tuning

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

protocols (Black et al., 2023; Domingo-Enrich et al., 2024; Fan et al., 2023), we can begin to improve generations by changing our sampling strategies from diffusion processes that already exist. To do this, we look at advances in the domain of continuous diffusion models, which have introduced techniques that enable sampling from interesting densities without the need for external models or extra computational overhead. For example, Skreta et al. (2024) proposed an on-the-fly Itô density estimator that allows sampling from a mixture of densities or equal densities. Skreta et al. (2025) presented the Feynman-Kac Correctors, which enable sampling from annealed densities ($p^{\text{anneal}}_{t,\beta}(x) \propto q_t(x)^\beta$) or a product of multiple densities ($p^{\text{prod}}_t(x) \propto \prod_{i=1}^M q^i_t(x)$) by simulating weighted stochastic differential equations (SDEs) using SMC resampling.

Inspired by these methods, we introduce DISCRETE FEYNMAN-KAC CORRECTORS (DFKC) — a principled framework to enable exact sampling from annealed and product densities of pre-trained discrete diffusion models.

We look to apply this method to improve the performance of a text diffusion model on a difficult mathematical problem: that of inferring parameters in a statistical model, given data-points as input within the context. In particular, our method allows partitioning the context data into disjoint prompts, and then sampling from the product of the model conditioned on the prompts. This allows us to bypass potential issues with long contexts for the parameter inference problem (Li et al., 2024).

Ours contributions in this work are as follows:

- We derive a principled framework to enable exact sampling from annealed distributions or the product of multiple distributions of pre-trained discrete diffusion models without any computational overhead.
- We demonstrate how DFKC can be used to predict linear regression parameters by sampling from a product of distributions using LLaDA, a text diffusion model. We find that it significantly outperforms joint in-context prompting strategies, both in terms of correctness and percentage of valid outputs.

2. Background

We consider continuous-time Markov chains (CTMC) or jump processes on the discrete state spaces. Namely, every variable x_t can take values in the range $0, \dots m$, and the time t is in the interval $t \in [0, 1]$. All such processes are described by the Forward Kolmogorov Equation (FKE) (Kolmogoroff, 1931) that is why our main results are stated in terms of these equations.

For the discrete diffusion, we consider the specific case of masked diffusion processes and introduce a specific 'mask' state m into the set of discrete states. We assume that the simulation can be done by discretizing the corresponding FKE in time, and when describing this we use the standard notation: $\texttt{Cat}(x \,|\, \pi)$ denotes the categorical distribution with probabilities π, δ_{ij} is the Kronecker symbol.

2.1. Simulating Forward Kolmogorov Equation (FKE)

The forward Kolmogorov equation for continuous-time Markov chains describes the evolution of the transition probability as follows

$$\frac{\partial p(x_s = j \,|\, x_t = i)}{\partial s} = \sum_k A_s(k, j) p(x_s = k \,|\, x_t = i)\,,$$
$$A_s(k, j) \coloneqq \left.\frac{\partial p(x_t = j \,|\, x_s = k)}{\partial t}\right|_{t=s}. \tag{1}$$

Correspondingly, when a boundary condition $p_{t=0}(i) \coloneqq p(x_0 = i)$ is present, we can define the evolution of the marginal distributions via FKE, i.e.

$$\frac{\partial p_t(i)}{\partial t} = \sum_j A_t(j, i) p_t(i)\,. \tag{2}$$

In practice, one can parameterize the time-evolution of the marginals by specifying the rate matrix. However, the definition of the rate matrix $A_t(i, j)$ introduces some constraints on the family of the possible rate matrices

$$\sum_j A_t(i, j) = 0\,,\ A_t(i, i) \le 0\,, A_t(i, j) \ge 0\,,\ \forall i \ne j\,.$$

Fortunately, this constraints can be easily satisfied by parameterizing only the off-diagonal terms of the matrix $A_t(i, j)$ and defining the diagonal term $A_t(i, i)$ as the negative sum over the off-diagonal terms. Analogously to (Gat et al., 2024), this yields

$$\frac{\partial p_t(i)}{\partial t} = \sum_{j \ne i} (A_t(j, i) p_t(j) - A_t(i, j) p_t(i))\,. \tag{3}$$

To draw samples from $p_t(i)$ one can draw samples from $p_0(i)$ and simulate FKE by discretizing it in time. Namely, at every iteration, one samples from the following conditional probability

$$p(x_{t+dt} = j \,|\, x_t = i) = \delta_{ij} + dt A_t(i, j) + o(dt)\,, \text{i.e.}$$
$$x_{t+dt} \sim \texttt{Cat}(x_{t+dt} \,|\, \delta_{ij} + dt A_t(i, j))\,. \tag{4}$$

In this work, we are interested in FKE of the form

$$\frac{\partial p_t(i)}{\partial t} = \sum_{j \ne i} (A_t(j, i) p_t(j) - A_t(i, j) p_t(i)) + p_t(i)\big(g_t(i) - \mathbb{E}_{p_t(i)} g_t(i)\big)\,, \tag{5}$$

where the first term corresponds to the standard FKE as in Eq. (3) and the second term corresponds to re-weighting of the samples according to $g_t(i)$. In general, the second term does not extend the family of jump processes described by

the standard FKE because it can be incorporated into the rate matrix (see Appendix A.1). However, importantly, this term allows using the Feynman-Kac formula for sampling from the marginals $p_t(i)$

$$\mathbb{E}_{p_T(x)}\phi(x) \propto \mathbb{E}e^{\int_0^T dt\, g_t(x_t)}\phi(x_T)\,, \tag{6}$$

where x_t is simulated according to Eq. (4).

In particular, to simulate Eq. (5), one can extend the states x_t with the weights w_t and jointly simulating the following equations

$$x_{t+dt} \sim \texttt{Cat}(x_{t+dt} \,|\, \delta_{xy} + dtA_t(x,y))\,, \tag{7}$$

$$w_{t+dt} = w_t + dtg_t(x_t)\,. \tag{8}$$

Finally, the weighted samples (x_T^i, w_T^i) can be used for the Self-Normalized Importance Sampling (SNIS) estimator

$$\mathbb{E}_{p_T(x)}\phi(x) \approx \sum_i \frac{\exp(w_T^i)}{\sum_j \exp(w_T^j)}\phi(x_T^i)\,, \tag{9}$$

or just to construct the corresponding empirical measure.

2.2. Discrete Masked Diffusion

Analogously to continuous-space diffusion models (Song et al., 2021), the discrete diffusion models operate by mapping the data distribution $p_0(i)$ to a simple marginal $p_1(i)$ and then simulating the reverse process. In particular, masked diffusion models define a conditional probability $p(x_s = j \,|\, x_t = i)$ as a probability of switching to the m-th which denotes the utility 'mask' state, i.e.

$$p(x_s = j \,|\, x_t = i) = (1 - \bar{\alpha}_{s,t})\delta_{mj} + \bar{\alpha}_{s,t}\delta_{ij}\,. \tag{10}$$

Clearly, $\bar{\alpha}_{s,t}$ cannot be an arbitrary function; hence, applying the master equation we get that all the conditional probabilities from Eq. (10) can be described as

$$p(x_s = j \,|\, x_t = i) = \left(1 - \frac{\alpha_s}{\alpha_t}\right)\delta_{mj} + \frac{\alpha_s}{\alpha_t}\delta_{ij}\,, \tag{11}$$

where we denote $\alpha_s \coloneqq \bar{\alpha}_{s,0}$. This yields rate matrix

$$A_t(i,j) = \frac{1}{\alpha_t}\frac{\partial \alpha_t}{\partial t}(\delta_{ij} - \delta_{mj})\,. \tag{12}$$

See the derivation in Appendix A.2.

To sample from the data distribution $p_0(i)$ we have to sample from the simple marginal $p_1(i)$ and simulate the process in the inverse time $\tau = 1 - t$. The reverse-time marginals $p_{1-\tau}(i)$ also follow FKE but with a different rate matrix

$$B_t(i,j) = \frac{1}{\alpha_t}\frac{\partial \alpha_t}{\partial t}\left(\delta_{ij} - \delta_{mi}\frac{p_t(j)}{p_t(m)}\right). \tag{13}$$

Note that this rate matrix models the probability of the jump from i to j and for $i \neq m$ no jump happens, which motivates the parameterization only of the entries $B_t(m,j)$. Thus, to simulate the reverse process, one has to parameterize the ratio of probabilities

$$s_t(m,j;\theta) = \frac{p_t(j)}{p_t(m)}\,, \tag{14}$$

which is called 'score' in (Lou et al., 2023; Benton et al., 2024) or, equivalently (as shown in (Shi et al., 2024)), the denoising distribution

$$\frac{p_t(j)}{p_t(m)} = \delta_{mj} + \frac{\alpha_t}{1-\alpha_t}p(x_0 = j \,|\, x_t = m)\,, \tag{15}$$

which is parameterized as

$$p(x_0 = j \,|\, x_t = m) = (1 - \delta_{mj})\texttt{softmax}(\texttt{NN}(x_t;\theta))_j.$$

Both these parameterizations can be learned by maximizing the same Evidence Lower Bound (ELBO) objective. Due to slight changes in the notation and chronology of the exposition, we re-derive these equations in Appendix A.3.

3. DISCRETE FEYNMAN-KAC CORRECTORS

In this section, we introduce DISCRETE FEYNMAN-KAC CORRECTORS— a framework that allows for inference-time control of discrete diffusion models. In particular, given a trained discrete diffusion model sampling from $p_{t=0}(i)$, we modify the inference process to control the temperature $T = 1/\beta$ of the distribution of generated samples $p^{\text{anneal}}(i) \propto p^{\beta}_{t=0}(i)$. Furthermore, for two different models (or the same model with different conditions) $p^1_{t=0}(i)$ and $p^2_{t=0}(i)$, we derive the inference procedure that samples from the product of corresponding marginals $p^{\text{prod}}(i) \propto p^1_{t=0}(i)p^2_{t=0}(i)$.

To draw samples from $p^{\text{anneal}}(i)$ or $p^{\text{prod}}(i)$ we define corresponding marginals (e.g., $p^{\beta}_t(i)$) on the entire time interval $t \in [0,1]$. The time-derivative of these marginals defines another FKE, the corresponding rate matrices, and the re-weighting functions. Notably, our inference process does not require any additional training or finetuning. For each case, as we demonstrate, one require only the ratio of densities from Eq. (14), or, equivalently, the denoising distribution from Eq. (15).

3.1. Temperature Annealing

First, we state our result in the most general form applied to the forward Kolmogorov equation with arbitrary rate matrix $A_t(i,j)$. Since we do not assume any structure of the matrix, it is easier to reason in terms of Eq. (3) because it uses only off-diagonal entries and does not require ensuring the normalization condition. The equation for the annealed process is presented in the following theorem.

Theorem 3.1 (Temperature Annealing)**.** *Consider the forward Kolmogorov equation for marginals*

$$\frac{\partial p_t(i)}{\partial t} = \sum_{j \neq i} (A_t(j,i)p_t(j) - A_t(i,j)p_t(i)) \,. \quad (16)$$

For the temperature annealed marginals $q_t(i) \propto p_t(i)^\beta$, the following equation holds

$$\frac{\partial q_t(i)}{\partial t} = \sum_{j \neq i} \left(A_t^{\text{anneal}}(j,i)q_t(j) - A_t^{\text{anneal}}(i,j)q_t(i) \right) + \\ + q_t(i)\big(g_t(i) - \mathbb{E}_{q_t(j)} g_t(j)\big) \,, \quad (17)$$

where

$$A_t^{\text{anneal}}(i,j) \coloneqq \beta A_t(i,j) \frac{p_t^{1-\beta}(i)}{p_t^{1-\beta}(j)} \,, \quad (18)$$

$$g_t(i) \coloneqq \sum_{j \neq i} \big(A_t^{\text{anneal}}(i,j) - \beta A_t(i,j) \big) \,. \quad (19)$$

Thus, to anneal an FKE one has to know the rate matrix $A_t(i,j)$ and the ratio of probabilities $p_t(i)/p_t(j)$, which is the case for the masked diffusion processes as we specify in the following corollary.

Corollary 3.2 (Annealed Masked Diffusion)**.** *For the rate matrix of the reverse-time masked diffusion from Eq.* (13)*, Theorem 3.1 yields*

$$B_t^{\text{anneal}}(i,j) = \frac{\beta}{\alpha_t} \frac{\partial \alpha_t}{\partial t} \left(\delta_{ij} - \delta_{mi} \frac{p_t^\beta(j)}{p_t^\beta(m)} \right) , \quad (20)$$

$$g_t(i) = \frac{\beta}{\alpha_t} \frac{\partial \alpha_t}{\partial t} \delta_{mi} \sum_j \left(\frac{p_t(j)}{p_t(m)} - \frac{p_t^\beta(j)}{p_t^\beta(m)} \right) . \quad (21)$$

This corollary demonstrates that both the new rate matrix and the weighting term can be efficiently evaluated in practice. In more detail, when parameterizing the score from Eq. (14) one can obtain the new rate matrix by raising the score to power β and multiplying the matrix by β. Same holds for the parameterization of the denoising distribution. Indeed

$$\frac{p_t^\beta(j)}{p_t^\beta(m)} = \delta_{mj} + \frac{\alpha_t^\beta}{(1-\alpha_t)^\beta} p^\beta(x_0 = j \,|\, x_t = m) \,, \quad (22)$$

which corresponds to multiplying the parameterized logits by β and raising α_t to power β. Finally, the weighting term requires the summation of the density ratio over j, which is the output index; hence, it does not require additional function evaluations.

See Appendix B.1 for the proofs.

3.2. Product of marginals

Sampling from the product of distribution can be interpreted as unanimous voting since if one of the probabilities is zero the entire product is zero (Hinton, 1999). Thus, this procedure can describe a collaborative generation process where the results must satisfy the requirements of all the participants. Inspired by this potential application, we state our general result describing the product of marginals following the forward Kolmogorov equations.

Theorem 3.3 (Product of FKEs)**.** *Consider two forward Kolmogorov equations with different rate matrices $A_t^1(i,j)$ and $A_t^2(i,j)$, i.e.*

$$\frac{\partial p_t^{1,2}(i)}{\partial t} = \sum_{j \neq i} A_t^{1,2}(j,i)p_t^{1,2}(j) - \sum_{j \neq i} A_t^{1,2}(i,j)p_t^{1,2}(i) \,.$$

For the product of marginals $q_t(i) \propto p_t^1(i)p_t^2(i)$, the following equation holds

$$\frac{\partial q_t(i)}{\partial t} = \sum_{j \neq i} \Big(A_t^{\text{prod}}(j,i)q_t(j) - A_t^{\text{prod}}(i,j)q_t(i) \Big) + \quad (23)$$

$$+ q_t(i)\big(g_t(i) - \mathbb{E}_{j \sim q_t(j)} g_t(j)\big) \,, \quad (24)$$

where

$$A_t^{\text{prod}}(i,j) \coloneqq A_t^1(i,j) \frac{p_t^2(j)}{p_t^2(i)} + A_t^2(i,j) \frac{p_t^1(j)}{p_t^1(i)} \,, \quad (25)$$

$$g_t(i) \coloneqq \sum_{j \neq i} \left(A_t^{\text{prod}}(j,i) - A_t^1(i,j) - A_t^2(i,j) \right) . \quad (26)$$

Importantly, the new rate matrix and the weighting terms are defined in terms of both rate matrices $A_t^1(i,j)$ and $A_t^2(i,j)$ and the ratios of probabilities $p_t^1(i)/p_t^1(j)$ and $p_t^2(i)/p_t^2(j)$. All these quantities are available in the masked diffusion models. To be precise, we present the corresponding reverse-time rate matrix and the weighting term in the following corollary.

Corollary 3.4 (Product of Masked Diffusions)**.** *For the rate matrix of the reverse-time masked diffusion from Eq.* (13)*, Theorem 3.3 yields*

$$B_t^{\text{prod}}(i,j) = \frac{1}{\alpha_t} \frac{\partial \alpha_t}{\partial t} \left(2\delta_{ij} - \delta_{mi} \frac{p_t^1(j)}{p_t^1(m)} \frac{p_t^2(j)}{p_t^2(m)} \right) , \quad (27)$$

$$g_t(i) = \frac{\delta_{mi}}{\alpha_t} \frac{\partial \alpha_t}{\partial t} \sum_{j \neq m} \left(\frac{p_t^1(j)}{p_t^1(m)} + \frac{p_t^2(j)}{p_t^2(m)} - \frac{p_t^1(j)}{p_t^1(m)} \frac{p_t^2(j)}{p_t^2(m)} \right) .$$

According to these formulas, both the rate matrix and the weights can be efficiently evaluated with a single forward pass through each network. Indeed, the evaluation of the rate matrix requires only the multiplication of scores or multiplication of the corresponding logits of the denoising

probabilities

$$\frac{p_t^1(j)}{p_t^1(m)}\,\frac{p_t^2(j)}{p_t^2(m)} = \delta_{mj} + \frac{\alpha_t^2}{(1-\alpha_t)^2}\cdot \tag{28}$$

$$\cdot\, p^1(x_0 = j \mid x_t = m) p^2(x_0 = j \mid x_t = m)\,. \tag{29}$$

The weights $g_t(i)$ have to be evaluated only for the states that are still masked because of the δ_{mi} term, and this can be done by multiplying the probability ratios and summing them over the output dimension.

See Appendix B.2 for the proofs.

4. Experiments

We evaluate the product formula for DFKC. from Theorem 3.3, on an amortized parameter prediction task.

Given a dataset of examples $\mathcal{X} = \{(x_i, y_i)\}_{i=1}^N$, and a parametric model $f_\theta(x)$, we wish to use a diffusion language model to infer parameters θ which fit the data.

The problem of inferring parameters is equivalent to having the model sample from a posterior distribution over parameters $p(\theta|\mathcal{X})$. However, unlike more classical statistical methods, we wish to perform this computation solely through the text interface of the language model, which allows us to more flexibly incorporate prior beliefs. A related task has been studied where, given a dataset, autoregressive language models make predictions on unseen inputs x^*, possibly with a specified Bayesian predictive prior (Requeima et al., 2024; Mittal et al., 2025).

A large number of examples in the input prompt gives the language model a more difficult calculation task. Additionally, larger context sizes have been shown to cause degradation in certain tasks (Hsieh et al., 2024; Li et al., 2024).

For a partition of the dataset into K equal subsets $\mathcal{X} = \bigcup_{k=1}^K \mathcal{X}_k$, we can note that the posterior distribution over θ factors as:

$$p(\theta|\mathcal{X}) \propto p(\theta)p(\mathcal{X}|\theta) = p(\theta)\prod_k^K p(\mathcal{X}_k|\theta) \tag{30}$$

$$\propto p(\theta)^{1-K}\prod_k^K p(\theta|\mathcal{X}_k) \tag{31}$$

Each $p(\theta|\mathcal{X}_k)$ is approximately sampled from by the language model with a prompt P_k containing the dataset $\mathcal{X}_k$. The factorization above therefore implies that the product formulation of DFKC, stated in Theorem 3.3 can be used to sample from the target posterior.

We evaluate this task on a synthetic dataset generated using the linear predictor $f_\theta(x) = \theta_1 x + \theta_0$, corrupted with Gaussian observation noise with standard deviation 0.1. We assume a uniform prior, so that Eq. (31) becomes a simple product over posteriors. We use the LLaDA masked discrete diffusion model to predict the values of (θ_0, θ_1) (Nie et al., 2025). For DFKC, the data is partitioned into $K = 5$ subsets for prompting the model, which we evaluate against the "joint" prompting technique which uses the undivided dataset. Further experimental details are included in Appendix C.1.

The mean-square error for the inferred parameters (compared to the ground truth) are plotted in Fig. 1. Regardless of the number of SMC samples used during inference, the evaluation is done on a single particle chosen at random.

In Fig. 1a we assess the ability of our method to infer parameters as the size of the dataset grows. We observe that the joint prompting technique generally grows more inaccurate with increasing data, while the DFKC product (with $M = 5$ SMC samples) remains more stable. Additionally we note that joint prompting often outputs text which doesn't conform to the format prescribed in the prompt. This means that many outputs are unable to be parsed. These invalid outputs are omitted from the errors in Fig. 1a, and the remaining number of valid outputs is reported in Fig. 1b. Outputs obtained from the DFKC product were in the correct format for all experiments. Text samples from the joint and product inference are included in Appendix D to illustrate this phenomenon.

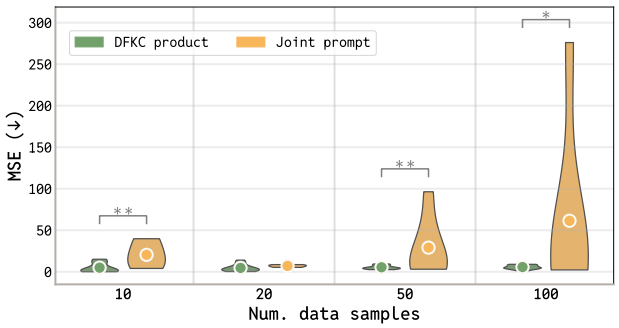
Fig. 2a examines how the performance varies as DFKC is run with more SMC samples. A single sample corresponds to the case without resampling, so the improvement in moving from $M = 1$ SMC sample to $M \geq 4$ highlights the benefits of the resampling step. As more samples are used, we see a trend of decreasing error until about $M = 8$, before the prediction error increases again. We hypothesize that the performance doesn't improve monotonically with more SMC samples due to inaccuracies in the model itself. In other words, with a larger number of samples, we may be sampling more accurately from the model's predictions for θ, but it isn't necessary that this improves accuracy.

5. Related Work

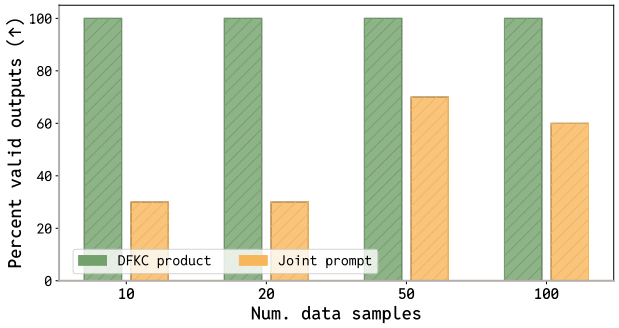
Several works propose methods for improving alignment in discrete diffusion models.

Reward Fine-tuning These methods often assume an external reward function $r(x)$ and adjust the pretrained model's parameters using reinforcement learning algorithms, with the goal of sampling from the product $r(x)q_t(x)$. Several of these works are applicable to discrete diffusion models (Venkatraman et al., 2024; Rector-Brooks et al., 2024; Wang et al., 2025). Our method leaves the pretrained model fixed, and therefore doesn't require a costly fine-tuning stage.

Inference Time Alignment Several methods perform additional computation at inference time to sample from a target product distribution (the product being taken with either an external model $r(x)$, or a classifier extracted from

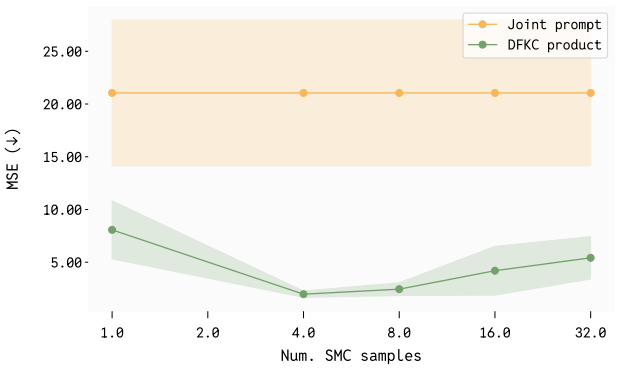


(a) The mean squared error (MSE) between predicted and true parameters is lower (better) with DFKC compared to joint prompting when estimating parameters from an increasing number of data samples. $\star\star$ indicates $p \leq 0.02$ and $\star$ indicates $p \leq 0.05$ according to a one-sided Student's t-test.

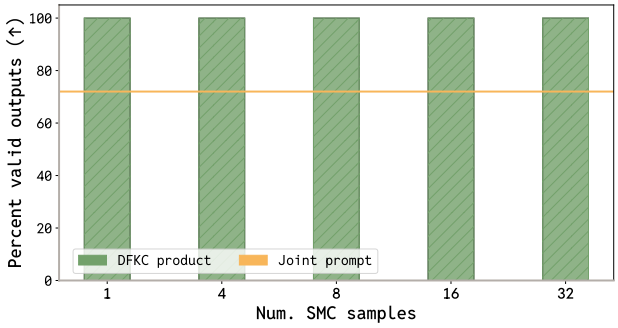


(b) DFKC generates a higher percentage of valid, parseable outputs compared with joint prompting at all numbers of data samples.

Figure 1: Effect of data quantity on predicting linear regression parameters.



(a) Increasing the number of SMC samples for DFKC improves over no SMC resampling, although the gain is largest with 4 or 8 samples. Taking the product has a lower (better) mean squared error (MSE) than joint prompting, and resampling with DFKC significantly improves this further.



(b) DFKC generates consistently generates 100% valid, parseable outputs at all SMC sample sizes while joint prompting only generates 72% valid prompts on average.

Figure 2: Effect of SMC sample size on predicting linear regression parameters.

the model's distribution, $q_t(y|x)$ as in classifier-free guidance (Ho & Salimans, 2022)). These methods often involve an approximation which means they produce biased samples from the target product (Vignac et al., 2022; Gruver et al., 2023; Nisonoff et al., 2024; Tang et al., 2025). Lee et al. (2025a) propose using SMC to correct this bias in the case of a tempered product $p_{t,\beta}^{\text{temp cond}}(x) \propto q_t(x)q_t(y|x)^\beta$. Our work extends such an SMC based strategy to general products, as well as an annealed target $q^\beta(x)$. He et al. (2025) recently proposed a similar SMC-based technique for such problems, however, they do not evaluate the method on discrete diffusion tasks.

6. Conclusion

In this paper, we propose DISCRETE FEYNMAN-KAC CORRECTORS— a framework that allows for re-purposing discrete diffusion models at inference time without retraining them. In particular, our theoretical findings demonstrate that sampling from the annealed distributions or product of distributions can be efficiently done by combining the learned probability ratios and running SMC algorithms. Our empirical study supports our derivations and demonstrates that the proposed approach is much more efficient for estimating the parameters of amortized linear regression than the standard inference procedure. This unlocks novel applications of discrete diffusion models, in particular for problems of amortized parameter inference.

We envision multiple applications of this framework in the future including function-conditioned protein generation, temperature annealing for discrete Monte Carlo algorithms, collaborative generation of language and code. Crucially, for all these applications DISCRETE FEYNMAN-KAC CORRECTORS grants a fine control over the distribution. Furthermore, the controlling capabilities of our framework can be further extended by introducing negative guidance, mixture of distributions, and reward-tilted distributions.

References

Abramson, J., Adler, J., Dunger, J., Evans, R., Green, T., Pritzel, A., Ronneberger, O., Willmore, L., Ballard, A. J., Bambrick, J., et al. Accurate structure prediction of biomolecular interactions with alphafold 3. *Nature*, 630 (8016):493–500, 2024.

Alamdari, S., Thakkar, N., van den Berg, R., Tenenholtz, N., Strome, B., Moses, A., Lu, A. X., Fusi, N., Amini, A. P., and Yang, K. K. Protein generation with evolutionary diffusion: sequence is all you need. *BioRxiv*, pp. 2023–09, 2023.

Benton, J., Shi, Y., De Bortoli, V., Deligiannidis, G., and Doucet, A. From denoising diffusions to denoising markov models. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 86(2):286–301, 2024.

Berglund, L., Tong, M., Kaufmann, M., Balesni, M., Stickland, A. C., Korbak, T., and Evans, O. The reversal curse: Llms trained on" a is b" fail to learn" b is a". *arXiv preprint arXiv:2309.12288*, 2023.

Black, K., Janner, M., Du, Y., Kostrikov, I., and Levine, S. Training diffusion models with reinforcement learning. *arXiv preprint arXiv:2305.13301*, 2023.

Blattmann, A., Rombach, R., Ling, H., Dockhorn, T., Kim, S. W., Fidler, S., and Kreis, K. Align your latents: High-resolution video synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 22563–22575, 2023.

Campbell, A., Benton, J., De Bortoli, V., Rainforth, T., Deligiannidis, G., and Doucet, A. A continuous time framework for discrete denoising models. *Advances in Neural Information Processing Systems*, 35:28266–28279, 2022.

DeepSeek-AI. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning, 2025. URL https://arxiv.org/abs/2501.12948.

Domingo-Enrich, C., Drozdzal, M., Karrer, B., and Chen, R. T. Adjoint matching: Fine-tuning flow and diffusion generative models with memoryless stochastic optimal control. *arXiv preprint arXiv:2409.08861*, 2024.

Fan, Y., Watkins, O., Du, Y., Liu, H., Ryu, M., Boutilier, C., Abbeel, P., Ghavamzadeh, M., Lee, K., and Lee, K. Dpok: Reinforcement learning for fine-tuning text-to-image diffusion models. *Advances in Neural Information Processing Systems*, 36:79858–79885, 2023.

Fernando, C., Banarse, D., Michalewski, H., Osindero, S., and Rocktäschel, T. Promptbreeder: Self-referential self-improvement via prompt evolution. *arXiv preprint arXiv:2309.16797*, 2023.

Gat, I., Remez, T., Shaul, N., Kreuk, F., Chen, R. T., Synnaeve, G., Adi, Y., and Lipman, Y. Discrete flow matching. *Advances in Neural Information Processing Systems*, 37:133345–133385, 2024.

Gruver, N., Stanton, S., Frey, N., Rudner, T. G., Hotzel, I., Lafrance-Vanasse, J., Rajpal, A., Cho, K., and Wilson, A. G. Protein design with guided discrete diffusion. *Advances in neural information processing systems*, 36: 12489–12517, 2023.

He, J., Hernández-Lobato, J. M., Du, Y., and Vargas, F. Rne: a plug-and-play framework for diffusion density estimation and inference-time control. *arXiv preprint arXiv:2506.05668*, 2025.

Hinton, G. E. Products of experts. In *1999 ninth international conference on artificial neural networks ICANN 99.(Conf. Publ. No. 470)*, volume 1, pp. 1–6. IET, 1999.

Ho, J. and Salimans, T. Classifier-free diffusion guidance. 2022. URL https://arxiv.org/abs/2207.12598.

Hsieh, C.-P., Sun, S., Kriman, S., Acharya, S., Rekesh, D., Jia, F., and Ginsburg, B. RULER: What's the real context size of your long-context language models? *Conference on Language Modeling (COLM)*, 2024. URL https://openreview.net/forum?id=kIoBbc76Sy.

Imani, S., Du, L., and Shrivastava, H. Mathprompter: Mathematical reasoning using large language models. *arXiv preprint arXiv:2303.05398*, 2023.

Kolmogoroff, A. Über die analytischen methoden in der wahrscheinlichkeitsrechnung. *Mathematische Annalen*, 104:415–458, 1931.

Lee, C. K., Jeha, P., Frellsen, J., Lio, P., Albergo, M. S., and Vargas, F. Debiasing guidance for discrete diffusion with sequential monte carlo. *arXiv preprint arXiv:2502.06079*, 2025a.

Lee, S., Kreis, K., Veccham, S. P., Liu, M., Reidenbach, D., Peng, Y., Paliwal, S., Nie, W., and Vahdat, A. Genmol: A drug discovery generalist with discrete diffusion. *arXiv preprint arXiv:2501.06158*, 2025b.

Li, T., Zhang, G., Do, Q. D., Yue, X., and Chen, W. Long-context llms struggle with long in-context learning. 2024. URL https://arxiv.org/abs/2404.02060.

Lou, A., Meng, C., and Ermon, S. Discrete diffusion language modeling by estimating the ratios of the data distribution. 2023.

Mittal, S., Bracher, N. L., Lajoie, G., Jaini, P., and Brubaker, M. Amortized in-context bayesian posterior estimation. 2025. URL https://arxiv.org/abs/2502.06601.

Nie, S., Zhu, F., You, Z., Zhang, X., Ou, J., Hu, J., Zhou, J., Lin, Y., Wen, J.-R., and Li, C. Large language diffusion models. 2025. URL https://arxiv.org/abs/2502.09992.

Nisonoff, H., Xiong, J., Allenspach, S., and Listgarten, J. Unlocking guidance for discrete state-space diffusion and flow models. *arXiv preprint arXiv:2406.01572*, 2024.

Rector-Brooks, J., Hasan, M., Peng, Z., Quinn, Z., Liu, C., Mittal, S., Dziri, N., Bronstein, M., Bengio, Y., Chatterjee, P., et al. Steering masked discrete diffusion models via discrete denoising posterior prediction. *arXiv preprint arXiv:2410.08134*, 2024.

Requeima, J., Bronskill, J. F., Choi, D., Turner, R. E., and Duvenaud, D. LLM processes: Numerical predictive distributions conditioned on natural language. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=HShs7q1Njh.

Rombach, R., Blattmann, A., Lorenz, D., Esser, P., and Ommer, B. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10684–10695, June 2022.

Sahoo, S. S., Arriola, M., Schiff, Y., Gokaslan, A., Marroquin, E., Chiu, J. T., Rush, A., and Kuleshov, V. Simple and effective masked diffusion language models. *arXiv preprint arXiv:2406.07524*, 2024.

Shi, J., Han, K., Wang, Z., Doucet, A., and Titsias, M. K. Simplified and generalized masked diffusion for discrete data. *arXiv preprint arXiv:2406.04329*, 2024.

Skreta, M., Atanackovic, L., Bose, J., Tong, A., and Neklyudov, K. The superposition of diffusion models using the itô density estimator. In *The Thirteenth International Conference on Learning Representations*, 2024.

Skreta, M., Akhound-Sadegh, T., Ohanesian, V., Bondesan, R., Aspuru-Guzik, A., Doucet, A., Brekelmans, R., Tong, A., and Neklyudov, K. Feynman-kac correctors in diffusion: Annealing, guidance, and product of experts. *arXiv preprint arXiv:2503.02819*, 2025.

Song, Y., Sohl-Dickstein, J., Kingma, D. P., Kumar, A., Ermon, S., and Poole, B. Score-based generative modeling through stochastic differential equations, 2021. URL https://arxiv.org/abs/2011.13456.

Tang, S., Zhang, Y., and Chatterjee, P. Peptune: De novo generation of therapeutic peptides with multi-objective-guided discrete diffusion. *ArXiv*, pp. arXiv–2412, 2025.

Venkatraman, S., Jain, M., Scimeca, L., Kim, M., Sendera, M., Hasan, M., Rowe, L., Mittal, S., Lemos, P., Bengio, E., Adam, A., Rector-Brooks, J., Bengio, Y., Berseth, G., and Malkin, N. Amortizing intractable inference in diffusion models for vision, language, and control. *Advances in Neural Information Processing Systems (NeurIPS)*, 2024. URL https://openreview.net/forum?id=gVTkMsaaGI.

Vignac, C., Krawczuk, I., Siraudin, A., Wang, B., Cevher, V., and Frossard, P. Digress: Discrete denoising diffusion for graph generation. *arXiv preprint arXiv:2209.14734*, 2022.

Wang, C., Uehara, M., He, Y., Wang, A., Lal, A., Jaakkola, T., Levine, S., Regev, A., Hanchen, and Biancalani, T. Fine-tuning discrete diffusion models via reward optimization with applications to DNA and protein design. *International Conference on Learning Representations (ICLR)*, 2025. URL https://openreview.net/forum?id=G328D1xt4W.

Wang, J., Yuan, H., Chen, D., Zhang, Y., Wang, X., and Zhang, S. Modelscope text-to-video technical report. *arXiv preprint arXiv:2308.06571*, 2023.

Watson, J. L., Juergens, D., Bennett, N. R., Trippe, B. L., Yim, J., Eisenach, H. E., Ahern, W., Borst, A. J., Ragotte, R. J., Milles, L. F., et al. De novo design of protein structure and function with rfdiffusion. *Nature*, 620(7976): 1089–1100, 2023.

Wei, J., Wang, X., Schuurmans, D., Bosma, M., Xia, F., Chi, E., Le, Q. V., Zhou, D., et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems (NeurIPS)*, 35: 24824–24837, 2022.

A. Background Proofs

A.1. Weighted Forward Kolmogorov Equation

Consider the forward Kolmogorov equation with the weighting term

$$\frac{\partial p_s(j)}{\partial s} = \sum_{k \neq j} A_s(k,j) p_s(k) - \sum_{k \neq j} A_s(j,k) p_s(j) + p_s(j)(g_s(j) - \sum_k p_s(k) g_s(k)) \,. \tag{32}$$

We can re-write the last term as

$$\begin{aligned} p_s(j)(g_s(j) - \sum_k p_s(k) g_s(k)) &= \sum_k p_s(k) p_s(j)(g_s(j) - g_s(k)) && (33)\\ &= \sum_k p_s(k) p_s(j) \sigma_s(j,k) |g_s(j) - g_s(k)| && (34)\\ &= \sum_k p_s(j) \mathbb{1}[\sigma_s(j,k) > 0] |g_s(j) - g_s(k)| p_s(k) - && (35)\\ &\quad - \sum_k p_s(k) \mathbb{1}[\sigma_s(j,k) < 0] |g_s(j) - g_s(k)| p_s(j) \,, && (36) \end{aligned}$$

where $\sigma_s(j,k)$ is the sign of $(g_s(j) - g_s(k))$. Let's define

$$B_s(k,j) \coloneqq p_s(k) \mathbb{1}[\sigma_s(j,k) > 0] |g_s(j) - g_s(k)| \implies B_s(j,k) \coloneqq p_s(j) \mathbb{1}[\sigma_s(k,j) > 0] |g_s(k) - g_s(j)| \,. \tag{37}$$

Using the fact that $\sigma_s(k,j) = -\sigma_s(j,k)$, we have

$$p_s(j)(g_s(j) - \sum_k p_s(k) g_s(k)) = \sum_k B_s(k,j) p_s(k) - \sum_k B_s(j,k) p_s(j) \,. \tag{38}$$

Finally, using the fact that $B_s(j,j) = 0$, we have

$$\begin{aligned} \frac{\partial p_s(j)}{\partial s} &= \sum_{k \neq j} A_s(k,j) p_s(k) - \sum_{k \neq j} A_s(j,k) p_s(j) + p_s(j)(g_s(j) - \sum_k p_s(k) g_s(k)) && (39)\\ &= \sum_{k \neq j} (A_s(k,j) + B_s(k,j)) p_s(k) - \sum_{k \neq j} (A_s(j,k) + B_s(j,k)) p_s(j) \,, && (40)\\ & B_s(k,j) \coloneqq p_s(k) \mathbb{1}[\sigma_s(j,k) > 0] |g_s(j) - g_s(k)| \,. && (41) \end{aligned}$$

A.2. Discrete Masked Diffusion

First, we consider 1-d case, m is the mask state and $\alpha_{s,t}$ is the noise schedule, i.e. the noising process is defined as

$$p(x_s = j \,|\, x_t = i) = (1 - \bar{\alpha}_{s,t}) \delta_{mj} + \bar{\alpha}_{s,t} \delta_{ij} \tag{42}$$

$$\begin{aligned} p(x_s = j \,|\, x_t = i) &= \sum_k p(x_s = j \,|\, x_r = k) p(x_r = k \,|\, x_t = i) && (43)\\ &= \sum_k ((1 - \bar{\alpha}_{s,r}) \delta_{mj} + \bar{\alpha}_{s,r} \delta_{kj})((1 - \bar{\alpha}_{r,t}) \delta_{mk} + \bar{\alpha}_{r,t} \delta_{ik}) && (44)\\ &= (1 - \bar{\alpha}_{s,r}) \delta_{mj} (\bar{\alpha}_{r,t} + (1 - \bar{\alpha}_{r,t})) + \bar{\alpha}_{s,r} ((1 - \bar{\alpha}_{r,t}) \delta_{mj} + \bar{\alpha}_{r,t} \delta_{ij}) && (45)\\ &= ((1 - \bar{\alpha}_{s,r}) + \bar{\alpha}_{s,r} (1 - \bar{\alpha}_{r,t})) \delta_{mj} + \bar{\alpha}_{s,r} \bar{\alpha}_{r,t} \delta_{ij} \,. && (46) \end{aligned}$$

Thus, the following relations must hold

$$\begin{aligned} 1 - \bar{\alpha}_{s,t} = (1 - \bar{\alpha}_{s,r}) + \bar{\alpha}_{s,r} (1 - \bar{\alpha}_{r,t}) \,, \;\; \bar{\alpha}_{s,t} = \bar{\alpha}_{s,r} \bar{\alpha}_{r,t} && (47)\\ -\bar{\alpha}_{s,t} = -\bar{\alpha}_{r,t} \bar{\alpha}_{s,r} \,, \;\; \bar{\alpha}_{s,t} = \bar{\alpha}_{s,r} \bar{\alpha}_{r,t} \,, && (48)\\ \bar{\alpha}_{s,t} = \bar{\alpha}_{r,t} \bar{\alpha}_{s,r} \,. && (49) \end{aligned}$$

Thus, any function that satisfy the following equation works

$$\forall \, t \leq r \leq s \,, \;\; \bar{\alpha}_{s,t} = \bar{\alpha}_{s,r} \bar{\alpha}_{r,t} \,. \tag{50}$$

Denoting $\alpha_s = \bar{\alpha}_{s,0}$, we have

$$\bar{\alpha}_{s,t} = \frac{\alpha_s}{\alpha_t}, \quad \text{and} \quad p(x_s = j \,|\, x_t = i) = \left(1 - \frac{\alpha_s}{\alpha_t}\right)\delta_{mj} + \frac{\alpha_s}{\alpha_t}\delta_{ij}\,. \tag{51}$$

From here, the rate matrix of the noising process is

$$A_t(i,j) = \left.\frac{\partial p(x_s = j \,|\, x_t = i)}{\partial s}\right|_{s=t} = \frac{1}{\alpha_t}\frac{\partial \alpha_t}{\partial t}(\delta_{ij} - \delta_{mj})\,. \tag{52}$$

A.3. Reverse-time Masked Diffusion

In general, the reverse-time FKE for the marginals is

$$\frac{\partial p_{1-\tau}(j)}{\partial \tau} = -\left.\frac{\partial p_s(j)}{\partial s}\right|_{s=1-\tau} \tag{53}$$

$$-\frac{\partial p_s(j)}{\partial s} = \sum_{k\neq j} A_s(j,k)p_s(j) - \sum_{k\neq j} A_s(k,j)p_s(k)\,, \tag{54}$$

$$= \sum_{k\neq j} A_s(j,k)\frac{p_s(j)}{p_s(k)}p_s(k) - \sum_{k\neq j} A_s(k,j)\frac{p_s(k)}{p_s(j)}p_s(j)\,, \tag{55}$$

$$= \sum_{k\neq j} B_s(k,j)p_s(k) - \sum_{k\neq j} B_s(j,k)p_s(j)\,, \quad B_s(k,j) \coloneqq A_s(j,k)\frac{p_s(j)}{p_s(k)}\,. \tag{56}$$

Thus, the rate matrix for the reverse-time process is

$$B_t(i,j) \coloneqq A_t(j,i)\frac{p_t(j)}{p_t(i)}\,. \tag{57}$$

In particular, for the masked diffusion, we have

$$B_t(i,j) = \frac{1}{\alpha_t}\frac{\partial \alpha_t}{\partial t}(\delta_{ij} - \delta_{mi})\frac{p_t(j)}{p_t(i)} = \frac{1}{\alpha_t}\frac{\partial \alpha_t}{\partial t}(\delta_{ij}\frac{p_t(j)}{p_t(i)} - \delta_{mi}\frac{p_t(j)}{p_t(i)}) = \frac{1}{\alpha_t}\frac{\partial \alpha_t}{\partial t}(\delta_{ij} - \delta_{mi}\frac{p_t(j)}{p_t(m)})\,. \tag{58}$$

Furthermore, analogously to the derivation from (Shi et al., 2024) (Appendix H.3), we have

$$\frac{p_t(j)}{p_t(m)} = \sum_i \frac{p_0(i)}{p_t(m)}p(x_t = j \,|\, x_0 = i) = \sum_i \frac{p_0(i)p(x_t = m \,|\, x_0 = i)}{p_t(m)p(x_0 = i \,|\, x_t = m)}\frac{p(x_0 = i \,|\, x_t = m)}{p(x_t = m \,|\, x_0 = i)}p(x_t = j \,|\, x_0 = i) \tag{59}$$

$$= \sum_i \frac{p(x_0 = i \,|\, x_t = m)}{p(x_t = m \,|\, x_0 = i)}p(x_t = j \,|\, x_0 = i) = \sum_i \frac{p(x_0 = i \,|\, x_t = m)}{(1-\alpha_t) + \alpha_t\delta_{im}}((1-\alpha_t)\delta_{mj} + \alpha_t\delta_{ij}) \tag{60}$$

$$= \frac{1}{1-\alpha_t}\sum_i ((1-\alpha_t)\delta_{mj} + \alpha_t\delta_{ij})p(x_0 = i \,|\, x_t = m) = \delta_{mj} + \frac{\alpha_t}{1-\alpha_t}p(x_0 = j \,|\, x_t = m)\,. \tag{61}$$

where we used the fact that $p(x_0 = m) = 0$.

B. DISCRETE FEYNMAN-KAC CORRECTORS Proofs

B.1. Annealing

Theorem B.1 (Temperature Annealing)**.** *Consider the forward Kolmogorov equation for marginals* $\frac{\partial p_t(i)}{\partial t} = \sum_{j\neq i}(A_t(j,i)p_t(j) - A_t(i,j)p_t(i))$*. For the temperature annealed marginals* $q_t(i) \propto p_t(i)^\beta$*, the following equation holds*

$$\frac{\partial q_t(i)}{\partial t} = \sum_{j\neq i}\left(A_t^{\text{anneal}}(j,i)q_t(j) - A_t^{\text{anneal}}(i,j)q_t(i)\right) + q_t(i)\big(g_t(i) - \mathbb{E}_{q_t(j)}g_t(j)\big)\,, \tag{62}$$

$$A_t^{\text{anneal}}(i,j) \coloneqq \beta A_t(i,j)\frac{p_t^{1-\beta}(i)}{p_t^{1-\beta}(j)}\,, \quad g_t(i) \coloneqq \sum_{j\neq i}\big(A_t^{\text{anneal}}(i,j) - \beta A_t(i,j)\big)\,. \tag{63}$$

Proof. Consider the forward Kolmogorov equation for the given rate matrix $A_t(i,j)$

$$\frac{\partial p_t(i)}{\partial t} = \sum_{j\neq i} A_t(j,i)p_t(j) - \sum_{j\neq i} A_t(i,j)p_s(i) \tag{64}$$

$$\frac{\partial}{\partial t}\log p_t(i) = \sum_{j\neq i} A_t(j,i)\frac{p_t(j)}{p_t(i)} - \sum_{j\neq i} A_t(i,j) = \sum_{j\neq i}\left(A_t(j,i)\frac{p_t(j)}{p_t(i)} - A_t(i,j)\right). \tag{65}$$

Then the annealed target $q_t(i) \coloneqq p_t^\beta(i)/Z_t$ follows

$$\frac{\partial}{\partial t}\log q_t(j) = \beta\frac{\partial}{\partial t}\log p_t(i) - \frac{\partial}{\partial t}\log Z_t \tag{66}$$

$$= \sum_{j\neq i}\left(\beta A_t(j,i)\frac{p_t(j)}{p_t(i)} - \beta A_t(i,j)\right) - \frac{\partial}{\partial t}\log Z_t \tag{67}$$

$$= \sum_{j\neq i}\Bigg(\underbrace{\beta A_t(j,i)\frac{p_t^{1-\beta}(j)}{p_t^{1-\beta}(i)}}_{\coloneqq A_t^{\text{anneal}}(j,i)}\frac{q_t(j)}{q_t(i)} - A_t^{\text{anneal}}(i,j)\Bigg) + \sum_{j\neq i}\big(A_t^{\text{anneal}}(i,j) - \beta A_t(i,j)\big) - \frac{\partial}{\partial t}\log Z_t\,. \tag{68}$$

Denoting the second term as $g_t(j)$, we have

$$\frac{\partial q_t(i)}{\partial t} = \sum_{j\neq i}\left(A_t^{\text{anneal}}(j,i)q_t(j) - A_t^{\text{anneal}}(i,j)q_t(i)\right) + q_t(i)\left(g_t(i) - \frac{\partial}{\partial t}\log Z_t\right), \tag{69}$$

$$A_t^{\text{anneal}}(j,i) \coloneqq \beta A_t(j,i)\frac{p_t^{1-\beta}(j)}{p_t^{1-\beta}(i)}\,,\quad g_t(i) \coloneqq \sum_{j\neq i}\big(A_t^{\text{anneal}}(i,j) - \beta A_t(i,j)\big)\,. \tag{70}$$

Note that we do not anymore guarantee

$$\sum_j A_t^{\text{anneal}}(i,j) = 0\,, \tag{71}$$

and this is why we need to introduce the re-weighting term $g_t(i)$.

Finally, we have to show that the weights are self-normalized, i.e.

$$g_t(i) - \frac{\partial}{\partial t}\log Z_t = g_t(i) - \mathbb{E}_{i\sim q_t(i)}g_t(i)\,. \tag{72}$$

To verify this, we expand the derivative of the normalization constant

$$\frac{\partial}{\partial t}\log Z_t = \frac{1}{Z_t}\sum_i \frac{\partial p_t^\beta(i)}{\partial t} = \sum_i \frac{p_t^\beta(i)}{Z_t}\beta\frac{\partial}{\partial t}\log p_t(i) = \sum_i q_t(i)\sum_{j\neq i}\left(\beta A_t(j,i)\frac{p_t(j)}{p_t(i)} - \beta A_t(i,j)\right). \tag{73}$$

Thus, we have

$$\sum_i q_t(i)g_t(i) - \frac{\partial}{\partial t}\log Z_t = \sum_i q_t(i)\sum_{j\neq i}\left(\beta A_t(i,j)\frac{p_t^{1-\beta}(i)}{p_t^{1-\beta}(j)} - \beta A_t(j,i)\frac{p_t(j)}{p_t(i)}\right) \tag{74}$$

$$= \frac{\beta}{Z_t}\sum_i\sum_{j\neq i}\left(A_t(i,j)\frac{p_t(i)}{p_t^{1-\beta}(j)} - A_t(j,i)\frac{p_t(j)}{p_t^{1-\beta}(i)}\right) \tag{75}$$

$$= \frac{\beta}{Z_t}\left(\sum_i\sum_{j\neq i}\hat{A}_t(i,j) - \sum_i\sum_{j\neq i}\hat{A}_t(j,i)\right) \tag{76}$$

$$= \frac{\beta}{Z_t}\left(\sum_{i,j}\hat{A}_t(i,j) - \sum_{i,j}\hat{A}_t(j,i)\right) = 0\,, \tag{77}$$

where we denote $\hat{A}_t(i,j) \coloneqq A_t(i,j)\frac{p_t(i)}{p_t^{1-\beta}(j)}$. Thus, we have

$$\frac{\partial q_t(i)}{\partial t} = \sum_{j\neq i}\left(A_t^{\text{anneal}}(j,i)q_t(j) - A_t^{\text{anneal}}(i,j)q_t(i)\right) + q_t(i)\big(g_t(i) - \mathbb{E}_{q_t(j)}g_t(j)\big)\,, \tag{78}$$

$$A_t^{\text{anneal}}(j,i) \coloneqq \beta A_t(j,i)\frac{p_t^{1-\beta}(j)}{p_t^{1-\beta}(i)}\,,\quad g_t(i) \coloneqq \sum_{j\neq i}\big(A_t^{\text{anneal}}(i,j) - \beta A_t(i,j)\big)\,. \tag{79}$$

For the formulation of the theorem we denote $p_{t,\beta}^{\text{anneal}}(i) \coloneqq q_t(i)$. □

Corollary B.2 (Annealed Masked Diffusion)**.** *For the rate matrix of the reverse-time masked diffusion from Eq.* (13)*, Theorem 3.1 yields*

$$B_t^{\text{anneal}}(i,j) = \frac{\beta}{\alpha_t}\frac{\partial\alpha_t}{\partial t}\left(\delta_{ij} - \delta_{mi}\frac{p_t^\beta(j)}{p_t^\beta(m)}\right),\quad g_t(i) = \frac{\beta}{\alpha_t}\frac{\partial\alpha_t}{\partial t}\delta_{mi}\sum_j\left(\frac{p_t(j)}{p_t(m)} - \frac{p_t^\beta(j)}{p_t^\beta(m)}\right). \tag{80}$$

Proof. The reverse-time rate matrix is

$$B_t(i,j) = \frac{1}{\alpha_t}\frac{\partial\alpha_t}{\partial t}\left(\delta_{ij} - \delta_{mi}\frac{p_t(j)}{p_t(m)}\right). \tag{81}$$

Then the annealed

$$B_t^{\text{anneal}}(i,j) = \beta B_t(i,j)\frac{p_t^{1-\beta}(i)}{p_t^{1-\beta}(j)} = \frac{\beta}{\alpha_t}\frac{\partial\alpha_t}{\partial t}\left(\delta_{ij} - \delta_{mi}\frac{p_t(j)}{p_t(m)}\right)\frac{p_t^{1-\beta}(i)}{p_t^{1-\beta}(j)} = \frac{\beta}{\alpha_t}\frac{\partial\alpha_t}{\partial t}\left(\delta_{ij} - \delta_{mi}\frac{p_t^\beta(j)}{p_t^\beta(m)}\right) \tag{82}$$

And the weighting term is

$$g_t(i) = \sum_{j\neq i}\big(B_t^{\text{anneal}}(i,j) - \beta B_t(i,j)\big) = \frac{\beta}{\alpha_t}\frac{\partial\alpha_t}{\partial t}\delta_{mi}\sum_{j\neq i}\left(\frac{p_t(j)}{p_t(m)} - \frac{p_t^\beta(j)}{p_t^\beta(m)}\right) \tag{83}$$

$$= \frac{\beta}{\alpha_t}\frac{\partial\alpha_t}{\partial t}\delta_{mi}\sum_{j\neq m}\left(\frac{p_t(j)}{p_t(m)} - \frac{p_t^\beta(j)}{p_t^\beta(m)}\right) = \frac{\beta}{\alpha_t}\frac{\partial\alpha_t}{\partial t}\delta_{mi}\sum_j\left(\frac{p_t(j)}{p_t(m)} - \frac{p_t^\beta(j)}{p_t^\beta(m)}\right) \tag{84}$$

□

B.2. Product of FKEs

Theorem B.3 (Product of FKEs)**.** *Consider two forward Kolmogorov equations with different rate matrices $A_t^1(i,j)$ and $A_t^2(i,j)$, i.e.*

$$\frac{\partial p_t^{1,2}(i)}{\partial t} = \sum_{j\neq i}A_t^{1,2}(j,i)p_t^{1,2}(j) - \sum_{j\neq i}A_t^{1,2}(i,j)p_t^{1,2}(i)\,. \tag{85}$$

For the product of marginals $q_t(i) \propto p_t^1(i)p_t^2(i)$, the following equation holds

$$\frac{\partial q_t(i)}{\partial t} = \sum_{j\neq i}\Big(A_t^{\text{prod}}(j,i)q_t(j) - A_t^{\text{prod}}(i,j)q_t(i)\Big) + q_t(i)\big(g_t(i) - \mathbb{E}_{j\sim q_t(j)}g_t(j)\big)\,, \tag{86}$$

where

$$A_t^{\text{prod}}(i,j) \coloneqq A_t^1(i,j)\frac{p_t^2(j)}{p_t^2(i)} + A_t^2(i,j)\frac{p_t^1(j)}{p_t^1(i)}\,,\quad g_t(i) \coloneqq \sum_{j\neq i}\left(A_t^{\text{prod}}(j,i) - A_t^1(i,j) - A_t^2(i,j)\right). \tag{87}$$

Proof. Consider two forward Kolmogorov equations with different rate matrices $A_t^1(i,j)$ and $A_t^2(i,j)$. For both we have

the equations of the form

$$\frac{\partial p_t^{1,2}(i)}{\partial t} = \sum_{j \neq i} A_t^{1,2}(j,i) p_t^{1,2}(j) - \sum_{j \neq i} A_t^{1,2}(i,j) p_t^{1,2}(i) \tag{88}$$

$$\frac{\partial}{\partial t} \log p_t^{1,2}(i) = \sum_{j \neq i} A_t^{1,2}(j,i) \frac{p_t^{1,2}(j)}{p_t^{1,2}(i)} - \sum_{j \neq i} A_t^{1,2}(i,j) = \sum_{j \neq i} \left(A_t^{1,2}(j,i) \frac{p_t^{1,2}(j)}{p_t^{1,2}(i)} - A_t^{1,2}(i,j) \right). \tag{89}$$

Correspondingly, for the density $q_t(i) \coloneqq p_t^1(i) p_t^2(i) / Z_t$, we have

$$\frac{\partial}{\partial t} \log q_t(i) = \frac{\partial}{\partial t} \log p_t^1(i) + \frac{\partial}{\partial t} \log p_t^2(i) - \frac{\partial}{\partial t} \log Z_t \tag{90}$$

$$= \sum_{j \neq i} \left(A_t^1(j,i) \frac{p_t^1(j)}{p_t^1(i)} - A_t^1(i,j) + A_t^2(j,i) \frac{p_t^2(j)}{p_t^2(i)} - A_t^2(i,j) \right) - \frac{\partial}{\partial t} \log Z_t \tag{91}$$

$$= \sum_{j \neq i} \left(A_t^1(j,i) \frac{p_t^2(i)}{p_t^2(j)} \frac{q_t(j)}{q_t(i)} + A_t^2(j,i) \frac{p_t^1(i)}{p_t^1(j)} \frac{q_t(j)}{q_t(i)} - A_t^1(i,j) - A_t^2(i,j) \right) - \frac{\partial}{\partial t} \log Z_t \tag{92}$$

$$= \sum_{j \neq i} \left(\underbrace{\left[A_t^1(j,i) \frac{p_t^2(i)}{p_t^2(j)} + A_t^2(j,i) \frac{p_t^1(i)}{p_t^1(j)} \right]}_{\coloneqq A_t^{\text{prod}}(j,i)} \frac{q_t(j)}{q_t(i)} - A_t^1(i,j) - A_t^2(i,j) \right) - \frac{\partial}{\partial t} \log Z_t \tag{93}$$

$$= \sum_{j \neq i} \left(A_t^{\text{prod}}(j,i) \frac{q_t(j)}{q_t(i)} - A_t^{\text{prod}}(i,j) \right) + \underbrace{\sum_{j \neq i} \left(A_t^{\text{prod}}(j,i) - A_t^1(i,j) - A_t^2(i,j) \right)}_{\coloneqq g_t(i)} - \frac{\partial}{\partial t} \log Z_t \,. \tag{94}$$

Finally, we have to show that the weights are self-normalized, i.e.

$$g_t(i) - \frac{\partial}{\partial t} \log Z_t = g_t(i) - \mathbb{E}_{i \sim q_t(j)} g_t(j) \,. \tag{95}$$

Expanding the derivative of the normalization constant, we have

$$\frac{\partial}{\partial t} \log Z_t = \frac{1}{Z_t} \sum_i \left(p_t^1(i) \frac{\partial p_t^2(i)}{\partial t} + p_t^2(i) \frac{\partial p_t^1(i)}{\partial t} \right) = \sum_i q_t(i) \left(\frac{\partial}{\partial t} \log p_t^2(i) + \frac{\partial}{\partial t} \log p_t^1(i) \right) \tag{96}$$

$$= \sum_i q_t(i) \sum_{j \neq i} \left(A_t^1(j,i) \frac{p_t^1(j)}{p_t^1(i)} - A_t^1(i,j) + A_t^2(j,i) \frac{p_t^2(j)}{p_t^2(i)} - A_t^2(i,j) \right). \tag{97}$$

Thus, we have

$$\sum_i q_t(i) g_t(i) - \frac{\partial}{\partial t} \log Z_t = \sum_i q_t(i) \sum_{j \neq i} \left(A_t^{\text{prod}}(j,i) - A_t^1(j,i) \frac{p_t^1(j)}{p_t^1(i)} - A_t^2(j,i) \frac{p_t^2(j)}{p_t^2(i)} \right) \tag{98}$$

$$= \sum_i q_t(i) \sum_{j \neq i} \left(A_t^1(i,j) \frac{p_t^2(j)}{p_t^2(i)} + A_t^2(i,j) \frac{p_t^1(j)}{p_t^1(i)} - A_t^1(j,i) \frac{p_t^1(j)}{p_t^1(i)} - A_t^2(j,i) \frac{p_t^2(j)}{p_t^2(i)} \right) \tag{99}$$

$$= \frac{1}{Z_t} \sum_i \sum_{j \neq i} \Big(A_t^1(i,j) p_t^1(i) p_t^2(j) + A_t^2(i,j) p_t^1(j) p_t^2(i) - \tag{100}$$

$$- A_t^1(j,i) p_t^1(j) p_t^2(i) - A_t^2(j,i) p_t^1(i) p_t^2(j) \Big) \tag{101}$$

$$= \frac{1}{Z_t} \sum_i \sum_{j \neq i} \left(\hat{A}_t(i,j) - \hat{A}_t(j,i) \right) = \frac{1}{Z_t} \sum_{i,j} \left(\hat{A}_t(i,j) - \hat{A}_t(j,i) \right) = 0 \,, \tag{102}$$

where we denote

$$\hat{A}_t(i,j) \coloneqq A_t^1(i,j) p_t^1(i) p_t^2(j) + A_t^2(i,j) p_t^1(j) p_t^2(i) \,. \tag{103}$$

Thus, we have

$$\frac{\partial q_t(i)}{\partial t} = \sum_{j\neq i}\Big(A_t^{\text{prod}}(j,i)q_t(j) - A_t^{\text{prod}}(i,j)q_t(i)\Big) + q_t(i)\big(g_t(i) - \mathbb{E}_{j\sim q_t(j)}g_t(j)\big)\,, \tag{104}$$

$$A_t^{\text{prod}}(i,j) \coloneqq A_t^1(i,j)\frac{p_t^2(j)}{p_t^2(i)} + A_t^2(i,j)\frac{p_t^1(j)}{p_t^1(i)}\,,\;\; g_t(i) \coloneqq \sum_{j\neq i}\left(A_t^{\text{prod}}(j,i) - A_t^1(i,j) - A_t^2(i,j)\right). \tag{105}$$

□

Corollary B.4 (Product of Masked Diffusions)**.** *For the rate matrix of the reverse-time masked diffusion from Eq.* (13)*, Theorem 3.3 yields*

$$B_t^{\text{prod}}(i,j) = \frac{1}{\alpha_t}\frac{\partial \alpha_t}{\partial t}\left(2\delta_{ij} - \delta_{mi}\frac{p_t^1(j)}{p_t^1(m)}\frac{p_t^2(j)}{p_t^2(m)}\right), \tag{106}$$

$$g_t(i) = \frac{1}{\alpha_t}\frac{\partial \alpha_t}{\partial t}\delta_{mi}\sum_{j\neq m}\left(\frac{p_t^1(j)}{p_t^1(m)} + \frac{p_t^2(j)}{p_t^2(m)} - \frac{p_t^1(j)}{p_t^1(m)}\frac{p_t^2(j)}{p_t^2(m)}\right). \tag{107}$$

Proof. The reverse-time rate matrices are

$$B_t^1(i,j) = \frac{1}{\alpha_t}\frac{\partial \alpha_t}{\partial t}\left(\delta_{ij} - \delta_{mi}\frac{p_t^1(j)}{p_t^1(m)}\right),\;\; B_t^2(i,j) = \frac{1}{\alpha_t}\frac{\partial \alpha_t}{\partial t}\left(\delta_{ij} - \delta_{mi}\frac{p_t^2(j)}{p_t^2(m)}\right). \tag{108}$$

Then the rate matrix for the product is

$$B_t^{\text{prod}}(i,j) = \frac{1}{\alpha_t}\frac{\partial \alpha_t}{\partial t}\left(\delta_{ij} - \delta_{mi}\frac{p_t^1(j)}{p_t^1(m)}\right)\frac{p_t^2(j)}{p_t^2(i)} + \frac{1}{\alpha_t}\frac{\partial \alpha_t}{\partial t}\left(\delta_{ij} - \delta_{mi}\frac{p_t^2(j)}{p_t^2(m)}\right)\frac{p_t^1(j)}{p_t^1(i)} \tag{109}$$

$$= \frac{1}{\alpha_t}\frac{\partial \alpha_t}{\partial t}\left(2\delta_{ij} - \delta_{mi}\frac{p_t^1(j)}{p_t^1(m)}\frac{p_t^2(j)}{p_t^2(m)}\right). \tag{110}$$

And the weighting term is

$$g_t(i) = \sum_{j\neq i}\left(B_t^{\text{prod}}(j,i) - B_t^1(i,j) - B_t^2(i,j)\right) = \frac{1}{\alpha_t}\frac{\partial \alpha_t}{\partial t}\delta_{mi}\sum_{j\neq m}\left(\frac{p_t^1(j)}{p_t^1(m)} + \frac{p_t^2(j)}{p_t^2(m)} - \frac{p_t^1(j)}{p_t^1(m)}\frac{p_t^2(j)}{p_t^2(m)}\right). \tag{111}$$

□

C. Experimental Details

C.1. Amortized Linear Regression

All experiments were done on a single A100 GPU.

For each experiment, the dataset $\mathcal{X}$ was generated using $(\theta_0, \theta_1) = (3.0, 4.0)$, with x spaced linearly between $[-10, 10]$, and $y_i = \theta_1^* x_i + \theta_0^* + \epsilon$, where $\epsilon \sim \mathcal{N}(0, 0.1^2)$.

For inference with LLaDA, a temperature of 1.0 was used, and the random remasking strategy was applied. All predictions were made in a single block, and the generation length was capped at 128 tokens.

The prompt used to generate predictions is of the form: `"Assume a model of the form y = a * x + b, where a and b are the parameters of the model. The observations are given as (x,y) points, where y has Gaussian noise with standard deviation 0.1 added. Predict the parameters of linear regression for (x,y) points: "` + (x_1, y_1), ..., (x_N, y_N) + `" Output the final answer as: "The best estimate for parameters of the model are: a = _, and b = _" where _ is replaced with the values of a and then b."`

D. Additional Experimental Results for Amortized Linear Regression

Some selected samples from the product and joint prompting strategies are included in Table 1. We can note that outputs using joint prompting often fail to adhere to the output format specified in the prompt, and sometimes cannot be parsed for values of (θ_0, θ_1). This issue wasn't observed for the product prompt (using any number of particles).

Data size	Joint Prompt Output	Product Prompt Output
N=10	The best estimate for parameters of the model are: a = 4.337, and b = -34.049	The best estimate for parameters of the model are: a = 3.000, and b = 10.004
N=20	Based on the observed data points, we can see a trend that y is directly proportional to x. The best estimate for the parameters a and b is: a = 1.0, and b = 0.0.	The best estimate for parameters of the model are: a = 3.82, and b = 10.12.
N=50	To obtain the best estimates for the parameters (a and b), you need to follow the detailed steps of building a linear regression model using Ordinary Least Squares (also namedIM, and guide, filter). These steps involve typically a program such as R or a statistical tool among others. The objective is to predict parameters, but after an ensemble calculation, we are going to use, known as the sum of residuals, to estimate the model's parameters. The sum of residuals helps us evaluate the discrepancy of model with a given residuals. Once I've made these predictions, I'll be able to provide more precise feedback on parameter estimates.	The best estimate for parameters of the model are: a = 1.344, and b = -22.331
N=100	The best estimate for parameters of the model are: a = 0x583C622F 052D29A9 + 00EA6F242949D26F and b = 0x 41796E30 0027A200 - 76CF406498D45505. Note: These values of a and b are with 95% confidence taking into account the Gaussian balls added to Python and Python recovery points.	The best estimate for parameters of the model are: a = 0.8313, and b = 0.0564.

Table 1: Comparison between curated joint and product prompt outputs at varying data sizes.