ConvergeFlow: Language Flow with Provable Convergence to Token Embeddings

Na Li*† Yuchen Jiao*† Changxiao Cai‡ Gen Li†

August 25, 2026

Abstract

Recent advances in continuous diffusion and flow-based language models (LMs) have achieved performance competitive with discrete LMs. However, existing continuous frameworks still rely on decoders supervised with cross entropy (CE) because the flow trajectories are not guaranteed to terminate at valid token embeddings. Motivated by this limitation, we introduce **ConvergeFlow**, an embedding-space flow-based LM, which constrains the data predictor to the convex hull of token embeddings and trains it solely with the mean squared error objective induced by flow matching. Under suitable regularity conditions, we prove that the resulting flow converges to valid token embeddings despite errors in the data predictor, enabling direct token prediction without a CE-supervised decoder. We further develop three sampling mechanisms for controlling the trade-off between the generative perplexity and entropy. Experiments on OpenWebText demonstrate that ConvergeFlow achieves performance competitive with existing continuous and discrete diffusion LMs. These findings demonstrate the potential of the flow-based paradigm for language modeling. Our code is available at https://github.com/Na-Li66/ConvergeFlow.

Contents

1 Introduction **2**
1.1 Contributions . . . 3
1.2 Related work . . . 4

2 Background **5**
2.1 Flow matching and diffusion models . . . 5
2.2 Evaluation metrics for language modeling . . . 7

3 ConvergeFlow **8**
3.1 Framework and convergence theory . . . 8
3.2 Sampling . . . 11

4 Experiments **13**
4.1 Empirical flow convergence to token embeddings . . . 13
4.2 Effect of the training objective: MSE versus CE . . . 14
4.3 Control of quality-diversity trade-off . . . 15
4.4 Combination of three sampling techniques . . . 16
4.5 Further improvement . . . 17

5 Discussion **18**

*The authors contributed equally. Corresponding author: Gen Li.
†Department of Statistics and Data Science, Chinese University of Hong Kong, Hong Kong; Email: {na.li, yuchenjiao, genli}@cuhk.edu.hk
‡Department of Industrial and Operations Engineering, University of Michigan, Ann Arbor, USA; Email: cxcai@umich.edu.

A Proof of theorem and propositions **23**
A.1 Proof of Theorem 1 . . . 23
A.2 Proof of Proposition 1 . . . 28
A.3 Proof of Proposition 2 . . . 28
A.4 Proof of Proposition 3 . . . 29

B Further experimental results **30**
B.1 Detailed results for quality-diversity control . . . 30
B.2 Detailed results for combinations of sampling techniques . . . 30
B.3 Detailed results for guidance allocation . . . 30
B.4 Qualitative Samples . . . 35

1 Introduction

Diffusion models (Sohl-Dickstein et al., 2015; Song and Ermon, 2019; Ho et al., 2020) and flow matching (FM) (Lipman et al., 2022; Liu et al., 2022) have become the backbone for generative modeling in continuous data domains, with applications spanning image synthesis (Dhariwal and Nichol, 2021; Rombach et al., 2022; Esser et al., 2024), video generation (Ho et al., 2022; Wan et al., 2025), and protein design (Trippe et al., 2022). At a high level, these models learn a transport from a simple noise distribution to the data distribution. Diffusion models construct this transformation by learning to reverse a progressive noise corruption process, while FM learns the velocity field of a prescribed probability path. Once learned, the resulting generative dynamics iteratively transform fresh noise into new samples from the data distribution.

Language modeling, a central task in modern generative modeling, has long been dominated by autoregressive (AR) language models (LMs) (Radford et al., 2019; Brown et al., 2020). Despite their remarkable success in practice, AR models have two inherent drawbacks. First, the left-to-right generation order prevents earlier tokens from being revised using later context, limiting bidirectional reasoning and controllable generation. Second, one-by-one sequential generation inherently restricts parallelism and creates a fundamental bottleneck in sampling speed.

To overcome these limitations, substantial effort has recently been devoted to diffusion and flow-based language modeling (Li et al., 2022; Sahoo et al., 2024). These models offer a fundamentally different generation principle—they iteratively refine all token positions using bidirectional context. This formulation permits parallel token updates and allows global planning, controllable generation, and iterative revision, offering the potential for faster and more flexible generation.

Existing diffusion language models (DLMs) can be broadly categorized into continuous and discrete approaches. Continuous diffusion/flow-based models (Li et al., 2022; Han et al., 2023; Lovelace et al., 2023) map discrete tokens to continuous representations and apply Gaussian diffusion in the resulting continuous space.[1] Discrete DLMs (Sahoo et al., 2024; Shi et al., 2024) tailor the diffusion framework to the discrete nature of text by leveraging discrete diffusion models (Hoogeboom et al., 2021; Austin et al., 2021; Campbell et al., 2022), which define categorical corruption processes directly in token space. Recent scaling efforts have shown that discrete DLMs can achieve performance competitive with AR models (Nie et al., 2025; You et al., 2025; Ye et al., 2025; Song et al., 2025; Labs et al., 2025). Despite these substantial advances, operating in categorical state spaces makes it difficult to apply the extensive toolkit developed for continuous diffusion models, including classifier-free guidance (CFG) (Ho and Salimans, 2022), self-conditioning (Chen et al., 2022b), few-step ODE solvers (Lu et al., 2022a,b), and distillation (Song and Dhariwal, 2024; Yin et al., 2024). Moreover, their reliance on discrete token states may also limit their ability to exploit the rich latent geometry underlying language.

These considerations have motivated renewed interest in continuous flow-based LMs. Notably, recent embedding-space flow-based LMs, including LangFlow (Chen et al., 2026), ELF (Hu et al., 2026), and FLM (Lee et al., 2026), have achieved performance competitive with discrete DLMs. However, these continuous models still rely on token-level cross entropy (CE) supervision during training—LangFlow and FLM apply the CE objective along the flow trajectory, whereas ELF combines the FM objective at intermediate denoising

[1] Because continuous diffusion models are equivalent to FM with linear Gaussian interpolation, we use the terms interchangeably throughout this paper; see Section 2.

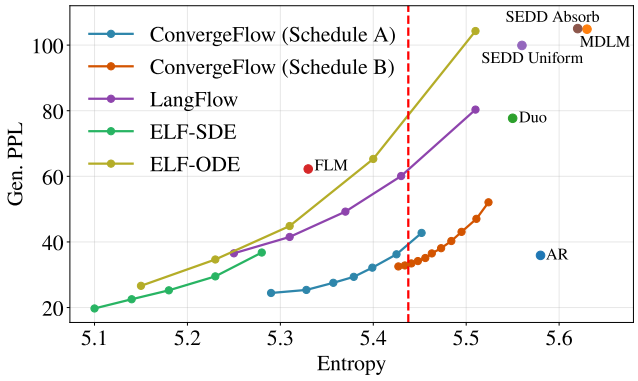


Figure 1: Gen. PPL-entropy trade-off across different models on the OWT dataset (Raffel et al., 2020). Curves show the trade-offs achieved by ConvergeFlow, LangFlow, ELF-SDE, and ELF-ODE, while individual markers indicate the reported results of the remaining baselines. Schedules A and B denote two time-adaptive guidance schedules for ConvergeFlow, defined in Section 3.2. The red dashed line marks the entropy of the OWT dataset, 5.44.

steps with the CE objective at the final decoding step. Crucially, their learned flow trajectories are not guaranteed to terminate at valid token embeddings, because errors in the learned data predictor or velocity can leave the terminal state between vocabulary embeddings. Consequently, these models require a CE-trained decoding mechanism to map such off-embedding states to discrete tokens. Although effective, this reintroduced discrete supervision is inconsistent with the continuous nature of flow-based models, and may limit their full potential.

Consequently, an important question remains unresolved:

> *Can the sampling trajectories of a flow-based LM converge directly to valid token embeddings, enabling discrete token prediction without a CE-supervised decoder?*

1.1 Contributions

In this work, we provide an affirmative answer by introducing **ConvergeFlow**, an embedding-space flow-based LM that retains a fully continuous formulation while incorporating the discrete structure of text.

We parameterize the data predictor as a weighted average of vocabulary embeddings, first introduced in Plaid (Gulrajani and Hashimoto, 2023) and later adopted by LangFlow. Each coefficient is parameterized using a learnable weight and an exact Gaussian kernel induced by the corruption process. The resulting data predictor is trained using the mean squared error (MSE) loss induced by the FM objective. We emphasize that LangFlow directly learns the convex-combination coefficients as token posteriors using CE supervision and subsequently maps them to a continuous data predictor. Instead, ConvergeFlow uses the factorization solely as an architectural parameterization and trains the resulting continuous data predictor directly with the FM objective.

Theoretically, under suitable regularity conditions, we prove that this parameterization ensures the learned flow converges to a valid token embedding despite errors in the learned data predictor. The same data predictor can therefore drive the intermediate continuous flow updates and produce the final discrete token prediction, thereby eliminating the need for a CE-supervised token decoder. We further validate this theoretical guarantee by showing that discrete tokens can be recovered accurately and directly from the terminal flow states.

Our contributions can be summarized as twofold:

- *Flow-based LM with provable convergence to token embeddings.* We introduce **ConvergeFlow**, an embedding-space flow-based LM. We prove that the resulting flow provably converges to valid token embeddings, enabling direct token prediction without a CE-supervised decoder. To our knowledge, ConvergeFlow is the first flow-based LM with provable convergence to token embeddings.

- *Quality–diversity control and strong empirical performance.* We propose three sampling mechanisms that provide explicit control over the trade-off between generation quality, measured by generative perplexity (Gen. PPL), and diversity, measured by entropy. Combined with these mechanisms, ConvergeFlow achieves performance competitive with continuous baselines, including LangFlow and ELF, as well as discrete baselines such as Duo (Sahoo et al., 2025). In particular, on the OpenWebText (OWT) dataset (Raffel et al., 2020), ConvergeFlow achieves a Gen. PPL of 33.17 while maintaining an entropy of 5.44; see Figure 1 and Table 1 for details.

Table 1: Comparison of Gen. PPL and entropy across different models. For ELF and LangFlow, we report the Gen. PPL at the point whose entropy is closest to that of the dataset; their complete Gen. PPL-entropy curves are shown in Figure 1. For ConvergeFlow, we report the Gen. PPL at the dataset entropy using the complete results in Figure 1 and Table 10. Results and model sizes marked with ‡ are taken from Duo. At the dataset entropy of 5.44, our method achieves a Gen. PPL of 33.17, whereas the lowest Gen. PPL among the continuous flow-based LMs is approximately 60, even though these models are evaluated at entropies below the dataset entropy.

Model	Gen. PPL (↓)	Entropy (↑)	Model Size
Autoregressive			
Transformer‡	35.90	5.58	170M
Discrete DLMs			
MDLM‡ (Sahoo et al., 2024)	104.85	5.63	170M
Duo‡ (Sahoo et al., 2025)	77.69	5.55	170M
SEDD Uniform‡ (Lou et al., 2023)	99.90	5.56	170M
SEDD Absorb‡ (Lou et al., 2023)	105.03	5.62	170M
Continuous DLMs			
FLM (Lee et al., 2026)	62.23	5.33	179M
LangFlow (Chen et al., 2026)	60.09	5.43	130M
ELF (Hu et al., 2026)	65.30	5.40	105M
ConvergeFlow	**33.17**	5.44	130M
Dataset	-	5.44	-

1.2 Related work

Continuous DLMs. Continuous DLMs differ primarily in the space over which Gaussian diffusion is performed. At the token level, embedding-space DLMs (Li et al., 2022; Dieleman et al., 2022; Gong et al., 2022; Gulrajani and Hashimoto, 2023) diffuse a sequence of token-level embeddings. Simplex-based DLMs (Han et al., 2023; Mahabadi et al., 2024; Tae et al., 2025; Potaptchik et al., 2026; Roos et al., 2026) instead map each token to a point on the probability simplex over the vocabulary, while Lee et al. (2026) adopts the one-hot encoding. Because such token-level states may not adequately capture contextual semantics, latent DLMs perform diffusion in sequence-level latent spaces constructed from contextual representations. Earlier approaches obtain these features from the outputs of a frozen pre-trained encoder (Lovelace et al., 2023; Zhang et al., 2023; Meshchaninov et al., 2026a), whereas more recent works jointly learn the latent encoder and the diffusion model (Meshchaninov et al., 2026b; Guo et al., 2026).

Discrete DLMs. Two families dominate modern discrete DLMs, distinguished by their forward corruption processes: uniform diffusion models (UDMs) and masked diffusion models (MDMs). UDMs progressively corrupt tokens toward the uniform distribution over the vocabulary (Sahoo et al., 2025, 2026). MDMs instead augment the vocabulary with a special mask token as an absorbing state, and progressively replace tokens with it (Sahoo et al., 2024; Shi et al., 2024). In MDMs, the discrete score (Meng et al., 2022; Lou et al., 2023) is equivalent to the joint conditional distribution of the masked tokens given the unmasked context (Ou et al., 2024; Zheng et al., 2025). In practice, masked DLMs approximate this joint conditional by a product of token-wise conditional marginals (Nie et al., 2025; Arriola et al., 2025). Although this enables

parallel generation, it imposes a conditional independence assumption among tokens revealed in the same iteration and thus introduces an inherent factorization bias. Consequently, the unmasking strategy—which determines the number and positions of tokens to reveal at each step—plays a critical role in the performance of masked DLMs (Kim et al., 2025; Wu et al., 2025; Yu et al., 2025; Ben-Hamu et al., 2026; Fu et al., 2025).

Theory for DLMs. Because masked DLMs have historically outperformed continuous DLMs and uniform DLMs, theoretical analyses for DLMs have largely focused on masked DLMs, particularly on characterizing the accuracy-speed trade-off in parallel generation. Early work studied fixed-size, random-ordering unmasking (Shi et al., 2024; Sahoo et al., 2024), which prescribes the number of sampling steps and tokens revealed per step while selecting their positions at random. Li and Cai (2025) derived the first convergence guarantees for such strategies, which were subsequently sharpened using information-theoretic quantities that capture low-complexity structure in the data distribution (Chen et al., 2025; Lavenant and Zanella, 2025; Zhao and Cai, 2026; Wainwright, 2026; Dmitriev et al., 2026). More recently, Cai and Li (2026) established the provable efficiency of confidence-based unmasking (Ben-Hamu et al., 2026), which adaptively selects both the number and positions of tokens to reveal based on the model's predictive confidence. Parallel to these sampling convergence results, statistical generalization guarantees have been established in Wakasugi and Suzuki (2026); Zhang et al. (2026).

Theory for continuous diffusion models. Recent years have witnessed substantial theoretical progress in continuous diffusion models (Lee et al., 2022; Chen et al., 2023b; Oko et al., 2023; Cai and Li, 2025; Wu and Cai, 2026). In particular, a line of work derives convergence guarantees for sampling from data distributions under mild assumptions, such as bounded moments, without requiring globally Lipschitz score functions (Chen et al., 2022a, 2023a; Benton et al., 2023; Li et al., 2024; Li and Yan, 2024; Jiao and Li, 2024; Jiao et al., 2025). Because token-level embedding distributions have finite support and hence bounded moments, these results naturally apply to continuous DLMs and flow-based LMs. Provably accelerated samplers based on higher-order approximations are further developed in (Li and Cai, 2024; Li et al., 2025b; Jain and Zhang, 2026). Moreover, the discrete nature of text also provides additional structure where Gaussian smoothing of the discrete token-embedding distribution yields a Gaussian mixture. Exploiting this structure, nearly dimension-free convergence guarantees have been established in Li et al. (2025a).

2 Background

2.1 Flow matching and diffusion models

Flow matching (FM) (Lipman et al., 2022) is a continuous-time generative modeling framework that transports a sample from a source distribution p_0 (typically the standard Gaussian) to a target distribution $p_1 = p_{\mathsf{data}}$.

The framework consists of two steps. First, one specifies a probability path $(p_t)_{t\in(0,1)}$ interpolating between the source p_0 and target p_1. A common choice defines p_t as the marginal distribution of

$$x_t = \alpha_t x_\star + \sigma_t z, \quad t \in [0,1], \tag{1}$$

where $x_\star \sim p_{\mathsf{data}}$ and $z \sim \mathcal{N}(0, I)$ is independent Gaussian noise. The differentiable schedules $(\alpha_t, \sigma_t)_{t\in(0,1)}$ are chosen such that $\lim_{t\to 0} \alpha_t/\sigma_t = 0$ and $\lim_{t\to 1} \alpha_t/\sigma_t = \infty$. Under the standard endpoint conditions $(\alpha_0, \sigma_0) = (0, 1)$ and $(\alpha_1, \sigma_1) = (1, 0)$, one has $x_1 = x_\star \sim p_1$ and $x_0 = z \sim p_0$.

Second, one learns a time-dependent velocity field $v : \mathbb{R}^d \times [0,1] \to \mathbb{R}^d$ whose induced flow realizes the prescribed probability path. Specifically, the velocity $v(x,t)$ generates the probability path p_t if the solution to the ordinary differential equation (ODE),

$$\frac{\mathrm{d}x_t}{\mathrm{d}t} = v(x_t, t), \quad t \in (0,1); \qquad x_0 \sim p_0, \tag{2}$$

satisfies $x_t \sim p_t$ for all $t \in [0,1]$.

Training. A natural objective for learning the velocity v_t is the *flow matching loss*

$$\ell_{\mathsf{FM}}(\theta) \coloneqq \mathbb{E}_{t,x_t}\Big[\big\|v_\theta(x_t,t) - v(x_t,t)\big\|_2^2\Big], \tag{3}$$

where $t \sim \mathsf{Unif}(0,1)$ and $x_t \sim p_t$. However, this objective cannot be evaluated directly because the velocity v_t is generally unavailable. Fortunately, one can obtain a tractable objective by conditioning on the target $x_\star$. Under the prescribed probability path (1), consider the conditional distribution $p_t(\cdot \mid x_\star)$ of x_t given $x_\star$. By the path construction in (1) and the ODE in (2), the conditional velocity field is given by

$$v(x_t, t \mid x_\star) = \alpha_t' x_\star + \sigma_t' z = \frac{\sigma_t'}{\sigma_t} x_t + \Big(\alpha_t' - \frac{\sigma_t'}{\sigma_t}\alpha_t\Big) x_\star = \frac{\alpha_t'}{\alpha_t} x_t + \Big(\sigma_t' - \frac{\alpha_t'}{\alpha_t}\sigma_t\Big) z. \tag{4}$$

The key observation underlying conditional flow matching (CFM) is that under mild regularity conditions, the posterior expectation of the conditional velocity,

$$v(x,t) = \mathbb{E}_{x_\star}[v(x_t,t \mid x_\star) \mid x_t = x], \tag{5}$$

yields the marginal velocity v_t that generates the probability path p_t (Lipman et al., 2022). This leads to the *conditional flow matching loss*:

$$\ell_{\mathsf{CFM}}(\theta) \coloneqq \mathbb{E}_{t,x_\star,z}\Big[\big\|v_\theta(x_t,t) - v(x_t,t \mid x_\star)\big\|_2^2\Big], \tag{6}$$

where $t \sim \mathsf{Unif}(0,1)$, $x_\star \sim p_1$, $z \sim \mathcal{N}(0,I)$, and $x_t = \alpha_t x_\star + \sigma_t z$. The minimizer of the objective (6) is given by the conditional expectation

$$v_{\theta^\star}(x,t) = \mathbb{E}_{x_\star}\big[v(x_t,t \mid x_\star) \mid x_t = x\big] = v(x,t).$$

Therefore, the CFM loss (6) shares the same minimizer as the FM loss (3) and thus provides a tractable objective for learning the velocity v_t. For simplicity of presentation, we will refer to the CFM loss as the FM loss in the rest of the paper.

In addition, the velocity can also be expressed via either a data predictor or a noise predictor. Define

$$\mu(x,t) \coloneqq \mathbb{E}[x_\star \mid x_t = x] \quad \text{and} \quad \varepsilon(x,t) \coloneqq \mathbb{E}[z \mid x_t = x]. \tag{7}$$

Combining (4) and (5), we obtain

$$v(x,t) = \frac{\sigma_t'}{\sigma_t} x + \Big(\alpha_t' - \frac{\sigma_t'}{\sigma_t}\alpha_t\Big)\mu(x,t) = \frac{\alpha_t'}{\alpha_t} x + \Big(\sigma_t' - \frac{\alpha_t'}{\alpha_t}\sigma_t\Big)\varepsilon(x,t). \tag{8}$$

This identity shows that learning the velocity is equivalent to predicting the clean data or noise. Therefore, FM is essentially the probability flow ODE in diffusion models (Song et al., 2020). Since we use the linear Gaussian interpolation, we use FM and diffusion models interchangeably in this paper.

Finally, for the linear schedule $\alpha_t = t$ and $\sigma_t = 1 - t$, the FM objective under the data prediction parameterization simplifies to

$$\mathbb{E}_{t,x_\star,z}\Big[(1-t)^{-2}\big\|x_\star - \mu_\theta(x_t,t)\big\|_2^2\Big].$$

For general schedules, the corresponding FM objective has a schedule-dependent weighting that is not invariant across schedules at a fixed signal-to-noise ratio (SNR). To remove this dependence, we instead use the following objective:

$$\mathbb{E}_{t,x_\star,z}\left[\Big(1 + \frac{\alpha_t}{\sigma_t}\Big)^2\big\|x_\star - \mu_\theta(x_t,t)\big\|_2^2\right].$$

Inference. After learning a velocity v_θ, approximate samples from the target distribution p_1 can be generated by drawing $x_0 \sim p_0$ from the source distribution p_0 and numerically solving the ODE

$$\frac{\mathrm{d}x_t}{\mathrm{d}t} = v_\theta(x_t, t), \quad t \in [0, 1]. \tag{9}$$

In practice, one can use a forward Euler method to approximate the one-step update from t to s:

$$x_s - x_t = \int_t^s v_\theta(x_\tau, \tau)\,\mathrm{d}\tau \approx v_\theta(x_t, t)(s - t).$$

The ODE in (9) can be reparameterized using either a data predictor μ_θ or a noise predictor ε_θ. Replacing μ with μ_θ in the data-prediction parameterization of the velocity in (8) gives

$$\frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{x_t}{\sigma_t}\right) = \mu_\theta(x_t, t)\frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{\alpha_t}{\sigma_t}\right).$$

This leads to the data prediction-based inference procedure:

$$\frac{x_s}{\sigma_s} - \frac{x_t}{\sigma_t} = \int_t^s \mu_\theta(x_\tau, \tau)\,\mathrm{d}\left(\frac{\alpha_\tau}{\sigma_\tau}\right) \approx \mu_\theta(x_t, t)\left(\frac{\alpha_s}{\sigma_s} - \frac{\alpha_t}{\sigma_t}\right). \tag{10}$$

Similarly, the noise predictor-based inference is given by

$$\frac{x_s}{\alpha_s} - \frac{x_t}{\alpha_t} = \int_t^s \varepsilon_\theta(x_\tau, \tau)\,\mathrm{d}\left(\frac{\sigma_\tau}{\alpha_\tau}\right) \approx \varepsilon_\theta(x_t, t)\left(\frac{\sigma_s}{\alpha_s} - \frac{\sigma_t}{\alpha_t}\right). \tag{11}$$

Self-conditioning. Self-conditioning (Chen et al., 2022b) is a technique that adds an additional input c to the predictor. During training, the conditional flow matching loss in (6) is modified to

$$\mathbb{E}_{t,x_\star,z,c}\Big[\big\|v_\theta(x_t, t, c) - v(x_t, t \mid x_\star)\big\|_2^2\Big],$$

where the input c is constructed by

$$c = \begin{cases} \varnothing, & \text{with probability } 1 - p, \\ \mathsf{stopgrad}\big(v_\theta(x_t, t, \varnothing)\big), & \text{with probability } p. \end{cases} \tag{12}$$

During training, the model makes an ordinary prediction without self-conditioning with a certain probability. Otherwise, it first produces an ordinary prediction and then uses the resulting prediction as an additional input in a second forward pass. In this way, the model learns to refine a prediction previously produced by itself. During inference, self-conditioning is applied at every step. The self-conditioning input is initialized as empty at the first step, when no previous prediction is available, and is set to the model's prediction from the preceding step thereafter.

2.2 Evaluation metrics for language modeling

We briefly review the metrics commonly used to evaluate language models. Let p_θ denote the distribution induced by a trained language model.

Because the data distribution p_{data} is unknown, the quality of the trained model is often assessed using a reference language model such as the GPT-2 Large model (Radford et al., 2019). Specifically, let p_{ref} denote the distribution of the reference model. *Generative perplexity* (Gen. PPL) is defined as

$$\mathrm{PPL}_{\mathrm{gen}}(p_\theta; p_{\mathsf{ref}}) \coloneqq \exp\left(-\frac{1}{L}\mathbb{E}_{X\sim p_\theta}[\log p_{\mathsf{ref}}(X)]\right), \tag{13}$$

which satisfies the following relationship:

$$\log \mathrm{PPL}_{\mathrm{gen}}(p_\theta; p_{\mathsf{ref}}) = \frac{1}{L}\mathsf{KL}(p_\theta \,\|\, p_{\mathsf{ref}}) + \frac{1}{L}H(p_\theta), \tag{14}$$

where $\mathsf{KL}(p_\theta \,\|\, p_{\mathsf{ref}})$ is the Kullback-Leibler (KL) divergence between the model distribution p_θ and the reference distribution p_{ref}, and $H(p_\theta)$ denotes the entropy of p_θ.

Identity (14) reveals that low Gen. PPL may arise either because the generated samples have high likelihood under the reference model or because the model concentrates its probability mass on a small set of likely sequences. Therefore, it is common to evaluate the entropy of the model distribution alongside its Gen. PPL as a measure of diversity.

In practice, entropy is often approximated using *unigram entropy*. For a sequence $x = (x^{(1)}, \dots, x^{(L)})$, define its empirical unigram distribution by

$$\widehat{p}_x(v) = \frac{1}{L} \sum_{i=1}^{L} \mathbb{1}\{x^{(i)} = v\}, \quad \forall\, v,$$

and let $H(\widehat{p}_x)$ denote the corresponding entropy. The unigram entropy is then defined as

$$H_{\mathsf{uni}}(p_\theta) \coloneqq \mathbb{E}_{X \sim p_\theta}[H(\widehat{p}_X)]. \tag{15}$$

The unigram entropy serves as a proxy for the normalized sequence entropy $L^{-1}H(p_\theta)$ appearing in (14), providing a simple diagnostic of within-sequence diversity.

In practice, the expectations defining Gen. PPL and unigram entropy are approximated by averaging over independent samples.

3 ConvergeFlow

3.1 Framework and convergence theory

Let $s = (s^{(1)}, \dots, s^{(L)})$ be a token sequence of length L drawn from the data distribution p_{data}, where each token $s^{(i)}$ belongs to a vocabulary of size V. Without loss of generality, we assume the vocabulary is $[V] \coloneqq \{1, \dots, V\}$. We map tokens to continuous representations using an embedding matrix $E \in \mathbb{R}^{V \times d}$, where d is the embedding dimension. For each $j \in [V]$, let $e_j^\top \coloneqq E_{j,:} \in \mathbb{R}^d$ denote the j-th row of the embedding matrix E, representing the embedding of the j-th token in the vocabulary. The target $x_\star$ is the continuous representation of the token sequence, given by

$$x_\star = \left[e_{s^{(1)}}, \dots, e_{s^{(L)}}\right]^\top \in \mathbb{R}^{L \times d}. \tag{16}$$

We consider FM with general interpolation schedules (α_t, σ_t):

$$x_t = \alpha_t x_\star + \sigma_t z,$$

where $z \in \mathbb{R}^{L \times d}$ is a standard Gaussian random matrix with i.i.d. entries $z_{ij} \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0,1)$. Using the data-prediction parameterization, we train a data predictor $\mu_\theta : \mathbb{R}^{L \times d} \times [0,1] \to \mathbb{R}^{L \times d}$ using the MSE loss induced by the FM objective:

$$\mathbb{E}_{t, x_\star, z}\left[\left(1 + \frac{\alpha_t}{\sigma_t}\right)^2 \left\|x_\star - \mu_\theta(x_t, t)\right\|_{\mathrm{F}}^2\right]. \tag{17}$$

This objective, also used by ELF, provides purely continuous supervision and does not involve a token-level CE loss.

An important caveat is that the FM objective alone does not sufficiently supervise the joint learning of the embedding matrix and the data predictor. Because the target $x_\star$ itself is defined by the embedding matrix, the FM objective admits degenerate embedding-collapse solutions. We therefore use the pre-trained embedding matrix from LangFlow and keep it fixed throughout training and inference.

We note that for language data, each row of the target $x_\star$ is supported on a finite collection of token embeddings rather than an unrestricted Euclidean space. The MSE loss in (17), however, treats the data predictor $\mu_\theta(x_t, t)$ as an unconstrained regressor and therefore fails to exploit this discrete support. This observation motivates the structured parameterization introduced next.

Embedding-weighted data predictor. Fix a token position $i \in [L]$. The clean embedding of the token at position i is $x_\star^{(i)} = e_{s^{(i)}}$. Observe that its conditional expectation given x_t is a weighted average of all token embeddings according to the posterior token distribution:

$$\mathbb{E}[x_\star^{(i)} \mid x_t] = \sum_{j=1}^{V} \mathbb{P}\{s^{(i)} = j \mid x_t\}\, e_j. \tag{18}$$

Consequently, the Bayes-optimal data predictor under the MSE loss (17) lies in the convex hull of the vocabulary embeddings. Moreover, the following proposition reveals a useful multiplicative structure of the posterior distribution: it can be factored into a context-only posterior and an exact Gaussian kernel. The proof is deferred to Appendix A.2.

Proposition 1. *For $x = [x^{(1)}, \dots, x^{(L)}]^\top \in \mathbb{R}^{L\times d}$, denote*

$$x^{(-i)} := \big[x^{(1)}, \dots, x^{(i-1)}, x^{(i+1)}, \dots, x^{(L)}\big]^\top \in \mathbb{R}^{(L-1)\times d}. \tag{19}$$

The posterior token distribution satisfies

$$\mathbb{P}\{s^{(i)} = j \mid x_t\} \propto \mathbb{P}\{s^{(i)} = j \mid x_t^{(-i)}\} \exp\big(-\|x_t^{(i)} - \alpha_t e_j\|_2^2/(2\sigma_t^2)\big). \tag{20}$$

The identity in (18) establishes an existence result—the posterior probabilities constitute one set of convex weights whose embedding-space barycenter equals the conditional mean of the clean embedding. However, these weights are generally not unique. Because the vocabulary size V is typically much larger than the embedding dimension d, distinct convex weights can produce exactly the same data prediction. Nevertheless, the convex structure in (18) suggests a useful parameterization.

Consequently, our goal is not to learn the posterior distribution itself, but to learn a valid set of convex weights whose embedding-space barycenter accurately predicts the conditional mean. Inspired by the multiplicative form in Proposition 1, we parameterize these convex coefficients using a learned base weight function and the known Gaussian corruption kernel. Specifically, we learn a base weight function $f_\theta^{(i)} : \mathbb{R}^{L\times d} \times [0,1] \mapsto \Delta([V])$, which is optimized using the MSE loss in (17). Importantly, $f_\theta^{(i)}$ is neither supervised nor interpreted as a token posterior. In particular, it is not intended to estimate the distribution $\mathbb{P}\{s^{(i)} = \cdot \mid x_t^{(-i)}\}$ appearing in Proposition 1. Rather, the proposition motivates only the form of the parameterization. We then define the convex weights

$$w_\theta^{(i)}(j \mid x_t, t) = \frac{f_\theta^{(i)}(j \mid x_t, t) \exp\big(-\|x_t^{(i)} - \alpha_t e_j\|_2^2/(2\sigma_t^2)\big)}{\sum_{j'\in[V]} f_\theta^{(i)}(j' \mid x_t, t) \exp\big(-\|x_t^{(i)} - \alpha_t e_{j'}\|_2^2/(2\sigma_t^2)\big)}, \quad j \in [V]. \tag{21}$$

The resulting data predictor for the token at position i is given by

$$\mu_\theta^{(i)}(x_t, t) = \sum_{j=1}^{V} w_\theta^{(i)}(j \mid x_t, t) e_j = E^\top w_\theta^{(i)}(\cdot \mid x_t, t). \tag{22}$$

Applying (22) to each token position i and stacking the outputs yields the full data predictor $\mu_\theta(x_t, t)$.

In summary, our proposed parameterization preserves the target of unconstrained MSE data prediction while explicitly incorporating both the discrete token structure and the known Gaussian corruption.

Provable convergence to token embeddings. Notably, our proposed parameterization for the data predictor guarantees convergence of the sampling trajectory to a valid token embedding. This is formalized below, with the proof deferred to Appendix A.1.

Theorem 1 (Flow convergence to token embeddings)**.** *Assume that, for every token position $i \in [L]$, the learned base weight function satisfies $f_\theta^{(i)}(j \mid x_t, t) > 0$ for any $j \in [V]$, state x_t, and time $t \in (0,1)$. Moreover, assume that the log-weight is Lipschitz continuous along the sampling trajectory: there exists a constant $\widetilde{L}$ such that*

$$\max_{j\in[V]} \big|\log f_\theta^{(i)}(j \mid x_t, t) - \log f_\theta^{(i)}(j \mid x_\tau, \tau)\big| \le \widetilde{L}\,|t - \tau|, \quad \forall\, 0 < t, \tau < 1.$$

Finally, assume that the time grid $0 = t_0 < t_1 < \ldots < t_N < 1$ *satisfies* $t_N \to 1$ *as* $N \to \infty$ *and*

$$\max_{0 \le k < N} \frac{t_{k+1} - t_k}{(1 - t_{k+1})^3} < \delta$$

for a sufficiently small $\delta > 0$*. Then, for each token position* $i \in [L]$*, there exists some* $j_i \in [V]$ *such that*

$$x_{t_N}^{(i)} \to e_{j_i} \quad \textit{in probability} \quad \textit{as} \quad N \to \infty. \tag{23}$$

Theorem 1 shows that every token-level state converges to a valid token embedding. Consequently, provided that the vocabulary embeddings are distinct, nearest-neighbor decoding naturally yields the token prediction. In particular, no separately trained terminal decoder is required. We empirically validate this convergence behavior in Section 4.

We next explain why data prediction accuracy alone does not guarantee convergence to token embeddings, and why additional structure, such as our convex-structured parameterization, is necessary. In particular, an unconstrained data predictor may be asymptotically accurate along any corruption path while its induced flow fails to converge to any token embedding.

The following proposition provides a concrete counterexample. Its proof is deferred to Appendix A.3.

Proposition 2. *There exists a smooth, unconstrained data predictor* μ_θ *such that*

$$\mu_\theta(x_t, t) \to x_\star \quad \textit{in probability} \quad \textit{as} \quad t \to 1,$$

where $x_t = \alpha_t x_\star + \sigma_t z$ *and* $z_{ij} \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0,1)$*. Let* x_{t_N} *denote the flow output induced by this data predictor on a time grid* $0 = t_0 < t_1 < \ldots < t_N < 1$ *with* $t_N \to 1$ *as* $N \to \infty$*. Then there exists a constant* $c_{\mathsf{lb}} > 0$*, independent of* N*, such that, for every token position* $i \in [L]$ *and all sufficiently large* N*,*

$$\mathbb{P}\left\{ \min_{j \in [V]} \left\| \alpha_{t_N}^{-1} x_{t_N}^{(i)} - e_j \right\|_2 \ge 1 \right\} \ge c_{\mathsf{lb}}. \tag{24}$$

This counterexample demonstrates that a smooth, asymptotically accurate data predictor does not by itself ensure convergence to the discrete vocabulary. The convex-structured parameterization provides sufficient structure to guarantee convergence to a valid token embedding.

Moreover, recall that the data predictor is a convex combination of the token embeddings, with weights given by $w_\theta^{(i)}$. The following proposition shows that if the data predictor converges to a token embedding, then the corresponding weight vectors converge to a one-hot vector. The proof is deferred to Appendix A.4.

Proposition 3. *Assume that the token embeddings* $\{e_j\}_{j \in [V]}$ *have the same norm and are pairwise separated, namely,* $\max_{j \ne j'} \left| \frac{\langle e_j, e_{j'} \rangle}{\|e_j\|_2 \|e_{j'}\|_2} \right| \le 1 - \rho$ *for some constant* $\rho > 0$*. If there exists some* $j \in [V]$ *such that* $\mu_\theta^{(i)}(x_t, t) \to e_j$ *as* $t \to 1$*, then one has*

$$w_\theta^{(i)}(\cdot \mid x_t, t) \to \delta_j \quad \textit{as} \quad t \to 1, \tag{25}$$

where $\delta_j \in \mathbb{R}^V$ *denotes the one-hot vector associated with token* j*.*

Comparison with embedding-space flow-based LMs.

- *Comparison with ELF (Hu et al., 2026).* Both ELF and our framework use the MSE objective to train a data predictor. ELF directly learns the data predictor as an unconstrained regressor. At the final step, it invokes a distinct trained decoder. In contrast, we impose additional structure on the data predictor through (22). This guarantees that the flow automatically converges to a token embedding, so the final decoding does not require a separate decoder.
- *Comparison with LangFlow (Chen et al., 2026).* Both LangFlow and our framework produce a data predictor through a convex combination of vocabulary embeddings. LangFlow directly learns the posterior distribution and trains it using the discrete CE objective. In contrast, we parameterize each convex weight using a learned base weight and an exact Gaussian likelihood, and train the resulting data predictor using the continuous MSE objective (17).

3.2 Sampling

Given a trained data predictor μ_θ, we generate samples by solving the data prediction-based ODE in (10), initialized with a standard Gaussian random matrix x_{t_0} with i.i.d. $\mathcal{N}(0,1)$ entries.

Given N sampling steps and a time grid $t_0 < t_1 < \ldots < t_N$, the ODE can be solved using the first-order Euler method, yielding the following update rule:

$$\frac{x_{t_{i+1}}}{\sigma_{t_{i+1}}} - \frac{x_{t_i}}{\sigma_{t_i}} = \mu_{t_i}\left(\frac{\alpha_{t_{i+1}}}{\sigma_{t_{i+1}}} - \frac{\alpha_{t_i}}{\sigma_{t_i}}\right), \tag{26}$$

where $\mu_{t_i} = \mu_\theta(x_{t_i}, t_i)$ denotes the data prediction used at step i. As we will see momentarily, the data prediction μ_{t_i} can be constructed in various ways, leading to different trade-offs between Gen. PPL and entropy.

After the last step, we convert the generated embedding x_{t_N} into a token sequence $\widehat{s} = (\widehat{s}^{(1)}, \ldots, \widehat{s}^{(L)})$ by taking the nearest neighbor in the embedding space for each token position:

$$\widehat{s}^{(i)} = \mathrm{argmin}_{j\in[V]} \|x_{t_N}^{(i)} - e_j\|_2, \quad i \in [L]. \tag{27}$$

Alternatively, we can use the trained weights $w_\theta(x_{t_N}, t_N)$ as the token distribution for decoding, i.e.,

$$\widehat{s}^{(i)} = \mathrm{argmax}_{j\in[V]}\, w_\theta^{(i)}(j \mid x_{t_N}, t_N), \quad i \in [L]. \tag{28}$$

Notably, both token prediction rules are parameter-free and require no separately trained terminal decoder.

Next, we describe sampling with self-conditioning. The data prediction in the ideal two-pass implementation of self-conditioning is given by

$$\mu_{t_i} = \mu_\theta\big(x_{t_i}, t_i, \mu_\theta(x_{t_i}, t_i, \varnothing)\big).$$

To reduce the computational burden, it is common to construct the data prediction μ_{t_i} at step i using that from the preceding step $i-1$ as the self-conditioning input, in place of the same-step unconditional prediction $\mu_\theta(x_{t_i}, t_i, \varnothing)$, i.e.,

$$\mu_{t_0} = \mu_\theta\big(x_{t_0}, t_0, \varnothing\big) \quad \text{and} \quad \mu_{t_i} = \mu_\theta\big(x_{t_i}, t_i, \mu_{t_{i-1}}\big), \;\; i \geq 1. \tag{29}$$

The data prediction μ_{t_i} is then used in the sampling update (26).

One can expect that $\mu_{t_{i-1}} \approx \mu_\theta(x_{t_i}, t_i, \varnothing)$ because t_{i-1} and t_i, as well as $x_{t_{i-1}}$ and x_{t_i} are close when the solver uses sufficiently many steps. However, we observe that such a computational shortcut also introduces a deeper self-conditioning recursion, which will be elaborated later.

Controlling Gen. PPL-entropy trade-off. We introduce three inference mechanisms for controlling the trade-off between Gen. PPL and entropy; see Table 2 for a summary.

- *self-conditioning guidance.* Motivated by classifier-free guidance (CFG) (Ho and Salimans, 2022), we introduce a CFG-type guidance for self-conditioning. At each solver step i, we form the guided data prediction via the unconditional and one-step self-conditioned predictions:

$$\mu_{t_i}^{\mathsf{scg}} = \mu_\theta(x_{t_i}, t_i, \varnothing) + w_{\mathsf{scg}}\big(\mu_\theta(x_{t_i}, t_i, c_{t_i}) - \mu_\theta(x_{t_i}, t_i, \varnothing)\big) \quad \text{with} \quad c_{t_i} = \mu_\theta(x_{t_i}, t_i, \varnothing). \tag{30}$$

  When $w_{\mathsf{scg}} = 0$, this reduces to sampling without self-conditioning; when $w_{\mathsf{scg}} = 1$, it recovers the ordinary sampling with self-conditioning. Values of $w_{\mathsf{scg}} > 1$ extrapolate beyond the self-conditioned prediction and amplify the refinement induced by self-conditioning.

  Although the form in (30) resembles CFG, the condition here is generated by the model itself rather than supplied externally.

- *Iterative self-conditioning refinement.* Recall the computational shortcut for self-conditioning in (29), which reuses the data prediction from the previous step. Unrolling this recursion over K steps shows that the data prediction μ_{t_i} at time t_i satisfies

$$\mu_{t_{i-j}} = \mu_\theta\big(x_{t_{i-j}}, t_{i-j}, \mu_{t_{i-j-1}}\big), \quad j = 0, \ldots, K-1,$$

or equivalently,

$$\mu_{t_i} = \mu_\theta\Big(x_{t_i}, t_i, \mu_\theta\big(x_{t_{i-1}}, t_{i-1}, \dots, \mu_\theta(x_{t_{i-K+1}}, t_{i-K+1}, \mu_{t_{i-K}})\dots\big)\Big).$$

If the time grid is sufficiently fine, then the state and time vary little over these K steps, and $\mu_\theta(x_{t_{i-j}}, t_{i-j}, c) \approx \mu_\theta(x_{t_i}, t_i, c)$ for $j = 0, \dots, K-1$. Consequently, the data prediction μ_{t_i} is approximately given by

$$\mu_{t_i} \approx \mu_\theta\Big(x_{t_i}, t_i, \mu_\theta\big(x_{t_i}, t_i, \dots, \mu_\theta(x_{t_i}, t_i, \mu_{t_{i-K}})\dots\big)\Big),$$

where K recursive evaluations are all applied to the current state x_{t_i} and time t_i. Thus, reusing the previous data prediction in self-conditioning implicitly produces a recursive refinement whose effective depth depends on the number and spacing of solver steps.

Motivated by this observation, we make the self-conditioning refinement explicit. At each sampling step i, we define

$$u_{t_i}^0 \coloneqq \mu_\theta(x_{t_i}, t_i, \varnothing), \qquad u_{t_i}^k \coloneqq \mu_\theta(x_{t_i}, t_i, u_{t_i}^{k-1}), \quad k = 1, \dots, K, \tag{31}$$

and use $u_{t_i}^K$ as the data prediction in the solver update. This construction makes the recursion depth K an explicit hyperparameter, thereby decoupling it from the number of solver steps.

Empirically, we observe that iterative self-conditioning refinement is less sensitive to the solver-step count than the standard self-conditioning shortcut in (29). Moreover, varying the depth K provides an effective means of controlling the trade-off between Gen. PPL and entropy.

- *Unconditional guidance.* We note that improving PPL is equivalent to increasing $\log p(x)$, so the most efficient way is to move in the direction of $\nabla \log p(x)$. By Tweedie's formula, we have

$$\nabla \log p_t(x) = -\frac{\varepsilon(x,t)}{\sigma_t}, \quad \text{with} \quad \varepsilon(x,t) = \frac{x - \alpha_t \mu(x,t)}{\sigma_t}.$$

Recall the standard noise prediction-based update rule from (11):

$$\frac{x_{t_{i+1}}}{\alpha_{t_{i+1}}} - \frac{x_{t_i}}{\alpha_{t_i}} = \left(\frac{\sigma_{t_{i+1}}}{\alpha_{t_{i+1}}} - \frac{\sigma_{t_i}}{\alpha_{t_i}}\right)\varepsilon_\theta(x_{t_i}, t_i).$$

As σ_t/α_t decreases along the sampling process, the coefficient on the right-hand side is negative and the update therefore moves in the direction of $-\varepsilon_\theta$. To strengthen this motion, we multiply the update by a factor of $1 + w_{\mathsf{ug}}$, resulting in the following sampler:

$$\frac{x_{t_{i+1}}}{\alpha_{t_{i+1}}} - \frac{x_{t_i}}{\alpha_{t_i}} = (1 + w_{\mathsf{ug}})\left(\frac{\sigma_{t_{i+1}}}{\alpha_{t_{i+1}}} - \frac{\sigma_{t_i}}{\alpha_{t_i}}\right)\varepsilon_\theta(x_{t_i}, t_i). \tag{32}$$

Table 2: Summary of sampling techniques for quality-diversity trade-offs.

Technique	Control parameter	Intended effect
Self-conditioning guidance	Coefficient w_{scg}	Amplify refinement from self-conditioning
Iterative self-conditioning refinement	Iteration count K	Refine data prediction
Unconditional guidance	Coefficient w_{ug}	Strengthen movement toward gradient of PPL

4 Experiments

Dataset. We follow the experimental setup used in the literature on DLMs (Chen et al., 2026; Hu et al., 2026; Sahoo et al., 2025). We conduct all experiments on the OpenWebText (OWT) dataset (Raffel et al., 2020), which contains approximately 9B tokens, and pack the text into sequences of length $L = 1024$.

Training. We follow the architecture and setup of LangFlow (Chen et al., 2026). We use the same DiT-style Transformer architecture (Peebles and Xie, 2023) as LangFlow, which consists of 12 layers, a hidden dimension of 768, and 12 attention heads, totaling approximately 130M parameters. Self-conditioning is applied during training with probability 0.25.

Because jointly learning the token embeddings and data predictor under the MSE objective admits degenerate embedding-collapse solutions, we use the embedding matrix from the LangFlow checkpoint and keep it fixed throughout training. For all experiments except the convergence study, the remaining trainable parameters are also initialized from the same checkpoint.

For a controlled comparison, we continue training our model using the MSE objective and the LangFlow baseline using its token-level CE objective for additional 200K steps. Both models are trained using AdamW (Loshchilov and Hutter, 2017) with a global batch size of 480 and a learning rate of 10^{-5}. Training is distributed across four or eight NVIDIA A100 40 GB GPUs, depending on availability.

Evaluation. For each sampling configuration, we generate 1024 samples with length $L = 1024$. We measure generation quality using Gen. PPL, evaluated by GPT-2 Large (Radford et al., 2019), and quantify diversity using unigram entropy. We compare sampling methods based on their Gen. PPL–entropy trade-off, where lower Gen. PPL indicates higher quality and higher entropy indicates greater diversity.

We observe that the sampling grid used by LangFlow is highly nonuniform near the two endpoints. Therefore, we use the uniform grid $t_i = (i + 0.5)/N$ for $i = 0, 1, \ldots, N - 1$, where N denotes the number of sampling steps. This grid slightly outperforms the original one in LangFlow; see Figure 9 in Appendix B.

Unless otherwise specified, our default sampling configuration is the standard sampler with one-step self-conditioning, as defined in (26) and (29), without additional techniques introduced in Section 3.2; see Appendix B.4 for generated examples at an entropy of 5.44. This corresponds to $w_{\mathsf{scg}} = 1$, $w_{\mathsf{ug}} = 0$, and $K = 1$. We compare all sampling methods under the same number of function evaluations (NFEs). Every evaluation of the data predictor is counted, including additional evaluations introduced by self-conditioning guidance or iterative self-conditioning refinement.

Unless otherwise stated, the results reported below use the checkpoint obtained after 175K additional training steps, which achieved the best performance among the evaluated checkpoints.

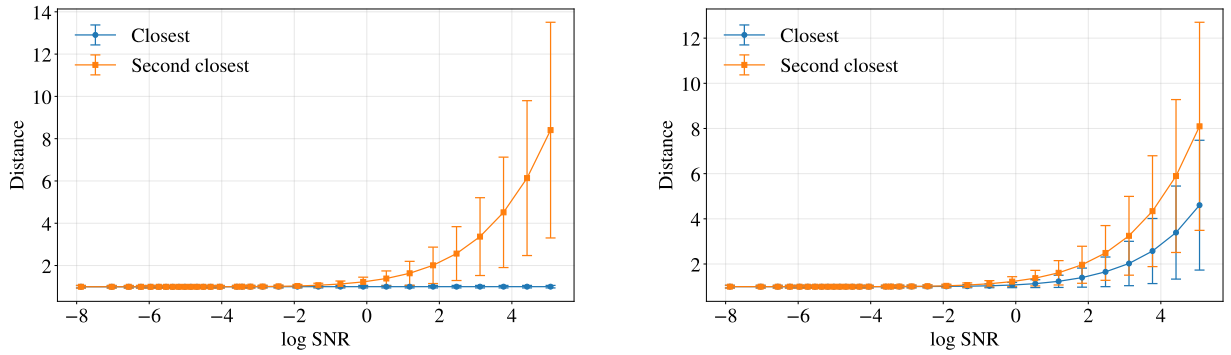
4.1 Empirical flow convergence to token embeddings

We empirically examine how the parameterization of the data predictor affects the convergence of the learned flow. We train two models from random initialization using the same pre-trained LangFlow embedding matrix, network architecture, and training configuration. The models differ only in their output parameterization. The embedding-weighted data predictor expresses its output as a weighted combination of token embeddings, as defined in (22), whereas the direct data predictor outputs an unconstrained vector in the embedding space. This controlled comparison isolates the effect of the structured parameterization: Theorem 1 guarantees the learned flow driven by the embedding-weighted data predictor converges to valid token embeddings, while Proposition 2 shows that such a convergence guarantee may not hold for an unconstrained data predictor.

For each token position $i \in [L]$, we compute the normalized distance between its continuous state $x_t^{(i)}$ and every scaled token embedding $\alpha_t e_j$:

$$\frac{\|x_t^{(i)} - \alpha_t e_j\|_2}{\sigma_t \sqrt{d}}, \qquad j \in [V].$$

For each log-SNR value, we aggregate the smallest and second-smallest distances over all $16 \times 1024 = 16{,}384$ token positions from 16 generated sequences of length $L = 1024$, separately for each data predictor. For each



(a) Embedding-weighted data predictor

(b) Unconstrained direct data predictor

Figure 2: Comparison of flow convergence between the embedding-weighted and direct data predictors. The two models use the same fixed LangFlow embeddings, architecture, and training configuration, differing only in their output parameterization. We plot the smallest and second-smallest normalized distances $\|x_t^{(i)} - \alpha_t e_j\|_2/(\sigma_t\sqrt{d})$ from the flow state associated with each token position $x_t^{(i)}$ to the scaled token embeddings as functions of log-SNR.

distance statistic, we plot the midpoint of its empirical 1st–99th percentile interval, with error bars spanning the full interval.

As shown in Figure 2, the closest and second-closest distances are nearly indistinguishable in the low-SNR regime for both data predictors, but their behaviors diverge as the SNR increases. For the embedding-weighted data predictor in Figure 2(a), the closest normalized distance approaches one and remains tightly concentrated, whereas the second-closest distance increases rapidly, producing a clear separation. In contrast, for the unconstrained direct data predictor in Figure 2(b), both distances increase, exhibit substantially greater variation, and remain poorly separated. These empirical results validate the contrasting convergence behaviors characterized by Theorem 1 and Proposition 2.

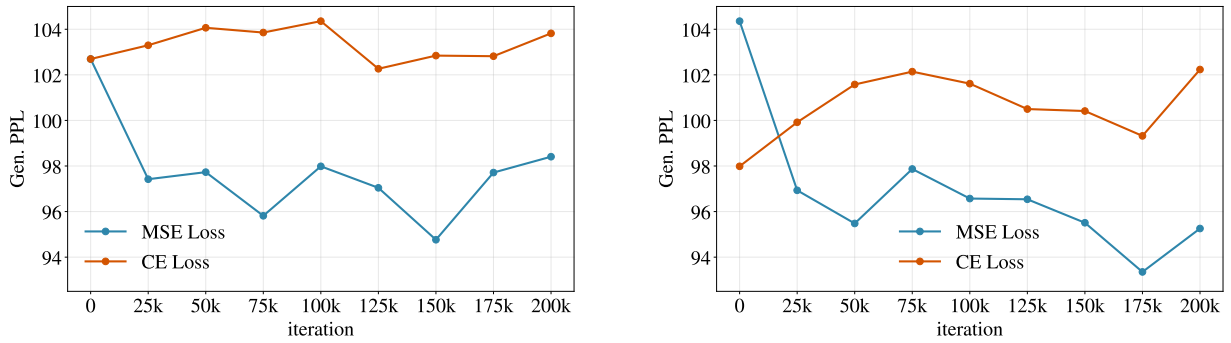
Having empirically examined convergence toward token embeddings, we next examine the consistency of the weight-based and distance-based token prediction rules. Theorem 1 and Proposition 3 together imply their asymptotic equivalence: the flow state converges to a token embedding, while the corresponding weights converge to its one-hot representation. As the number of sampling steps N varies from 32 to 512, the two rules agree at 99.16%–99.82% of token positions, with the agreement increasing as the sampling discretization becomes finer. This near-perfect agreement indicates that the two rules are effectively equivalent in practice, with the remaining discrepancies diminishing as the sampling time discretization becomes finer.

4.2 Effect of the training objective: MSE versus CE

Starting from the pre-trained LangFlow checkpoint, we continue training two models for 200K iterations under identical configurations, using the MSE and CE objectives, respectively. We evaluate Gen. PPL and entropy every 25K steps.

The entropy remains above 5.5 for both objectives throughout training. As for Gen. PPL, Figure 3(a) shows that training with the CE objective provides no consistent improvement over the initial checkpoint and its Gen. PPL remains near its initial value. In contrast, training with the MSE objective steadily reduces Gen. PPL and maintains a clear advantage throughout training, despite some fluctuations across checkpoints.

To further isolate the effect of the training objective, we conduct a crossover experiment using the checkpoints obtained after 100K steps. Specifically, we continue the CE-trained checkpoint using the MSE objective and, conversely, continue the MSE-trained checkpoint using the CE objective. As shown in Figure 3(b), despite starting from the worse-performing CE-trained checkpoint, switching to MSE reduces Gen. PPL consistently. Conversely, switching the better-performing MSE-trained checkpoint to CE degrades Gen. PPL. This crossover experiment confirms that the improvement is attributable to the MSE objective rather than



(a) Continued training from same LangFlow checkpoint

(b) Swapping objectives after 100K training steps

Figure 3: Effect of the training objective on Gen. PPL. (a) Starting from the same pre-trained LangFlow checkpoint, MSE-based training yields an immediate and persistent improvement over CE-based training. (b) In the crossover experiment, the CE-trained checkpoint improves rapidly after switching to MSE, whereas the MSE-trained checkpoint degrades after switching to CE. The horizontal axis denotes additional training steps after the objective switch.

favorable initialization. It further demonstrates that subsequent CE-based training can reverse gains previously obtained through MSE-based training.

4.3 Control of quality-diversity trade-off

Empirically, we find that sampling guidance is most effective near the data endpoint. We therefore adopt time-adaptive variants of self-conditioning guidance, unconditional guidance, and iterative self-conditioning refinement.

Specifically, at each step i, we replace the constant guidance strengths w_{scg} and w_{ug} with $w_{\mathsf{scg}}/(1+\sigma_{t_i}/\alpha_{t_i})$ and $w_{\mathsf{ug}}/(1+\sigma_{t_i}/\alpha_{t_i})$, respectively. Thus, these schedules gradually increase the guidance strength over the sampling trajectory.

For iterative self-conditioning refinement, we analogously adapt the refinement count according to

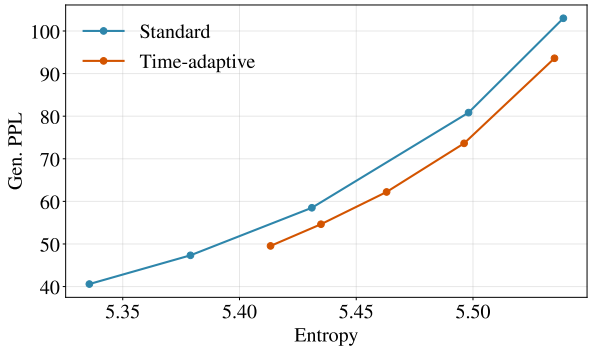
$$K_i = \left\lceil \frac{K_{\mathsf{iscr}}}{1+\sigma_{t_i}/\alpha_{t_i}} \right\rceil.$$

To ensure a fair comparison under a fixed computational budget, we choose the number of sampling steps separately for each configuration. In particular, the NFE for iterative self-conditioning refinement is given by

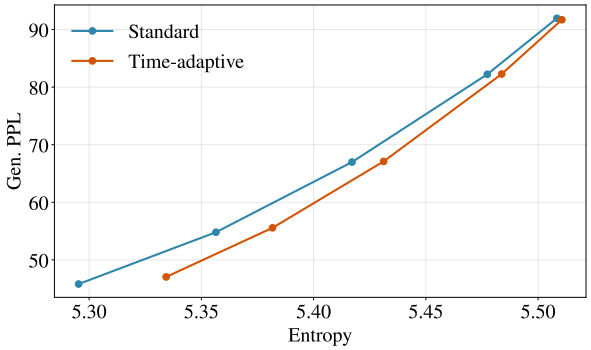
$$\mathsf{NFE} = \sum_{i=0}^{N-1} (K_i + 1).$$

All three sampling mechanisms provide effective control over the quality-diversity trade-off. Figures 4(a)–4(c) present the results for self-conditioning guidance, unconditional guidance, and iterative self-conditioning refinement, respectively. Figure 4(d) compares their time-adaptive variants. Each trade-off curve is obtained by varying the corresponding nominal control parameter, namely w_{scg}, w_{ug}, or K_{iscr}, while holding the total NFE fixed. The complete results for NFE=64 and NFE=128 are summarized in Tables 3 and 4 in Appendix B.1, respectively.

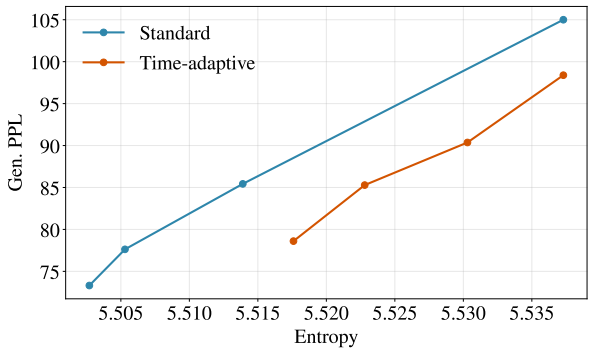
Overall, the time-adaptive variants generally improve the Gen. PPL-entropy frontier relative to their constant-strength counterparts. As shown in Figure 4(d), among the three adaptive techniques, self-conditioning guidance spans the broadest range of operating points and achieves the most favorable trade-off in the low- and medium-entropy regimes. Iterative self-conditioning refinement is particularly effective in the high-entropy regime, although it covers a narrower controllable range. Unconditional guidance yields only modest Gen. PPL reductions, and we investigate this further in Section 4.5.



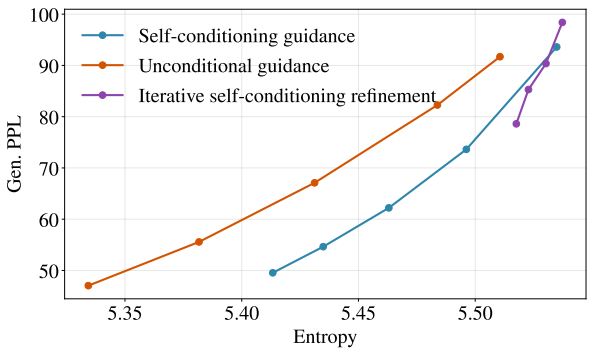
(a) Standard and time-adaptive self-conditioning guidance.



(b) Standard and time-adaptive unconditional guidance.



(c) Standard and time-adaptive iterative self-conditioning refinement.



(d) Comparison of the three time-adaptive samplers.

Figure 4: Gen. PPL-entropy trade-offs of the three sampling techniques at NFE=64. Each curve is obtained by varying the corresponding nominal control parameter. (a)–(c) compare the standard and time-adaptive variants of each sampling technique, where the adaptive variants increase the guidance strength or self-conditioning refinement count toward the data endpoint. (d) compares the time-adaptive variants of the three sampling techniques.

4.4 Combination of three sampling techniques

We next test the performance of combinations of the three sampling techniques, where we use the time-adaptive variant in all cases.

We first combine self-conditioning guidance and iterative self-conditioning refinement. Figure 5 reports the results under a fixed NFE budget of 64. The complete results for NFE=64 and NFE=128 are provided in Tables 5 and 6 in Appendix B.2, respectively.

As illustrated in Figure 5, increasing K_{iscr} generally shifts the trade-off frontier toward lower Gen. PPL at comparable entropy levels. Thus, combining a sufficiently large self-conditioning refinement parameter K_{iscr} with an appropriate self-conditioning guidance strength w_{scg} substantially improves the trade-off.

We next add time-adaptive unconditional guidance to the combined sampler. Figure 6 plots the results, with $K_{\mathsf{iscr}} = 50$ and $K_{\mathsf{iscr}} = 100$ shown separately in Figures 6(a) and 6(b), respectively. For both settings, varying w_{ug} primarily moves the operating point along a similar Gen. PPL-entropy frontier rather than shifting the frontier outward. These results suggest that self-conditioning guidance and iterative self-conditioning refinement capture most of the gain. Nevertheless, unconditional guidance remains useful because it does not require a conditioning signal, which we believe has independent interest. The complete numerical results for different combinations of w_{ug}, K_{iscr}, and w_{scg} are reported in Table 7 in Appendix B.2.

Finally, we compare the two token prediction rules described in Section 3.2. As the flow approaches the data endpoint, these two rules should output the same token. Figure 7 empirically verifies this agreement, with the NFE= 64 and NFE= 128 results shown in Figures 7(a) and 7(b), respectively. The corresponding

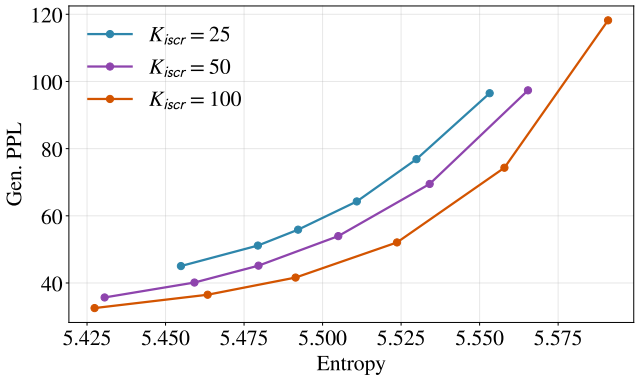
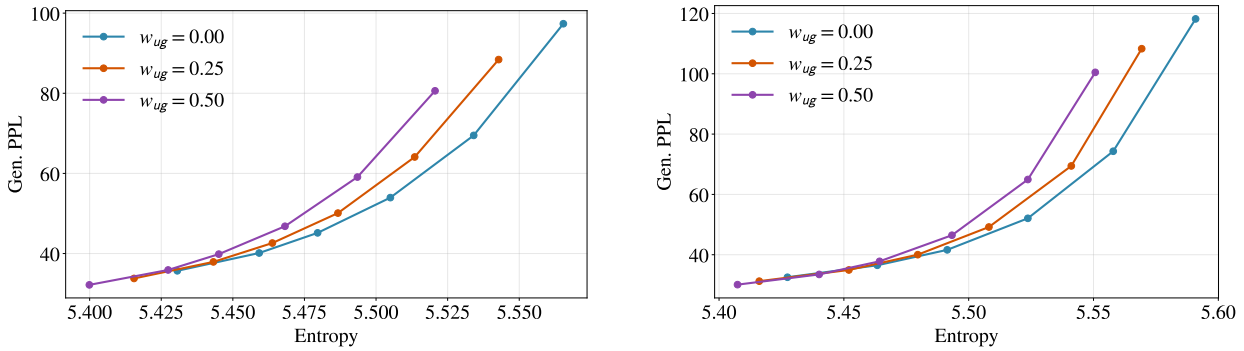


Figure 5: Joint effect of iterative self-conditioning refinement and self-conditioning guidance at NFE=64. Each curve fixes the refinement parameter K_{iscr} and varies the guidance strength w_{scg}.



(a) K_{iscr}=50 (b) K_{iscr}=100

Figure 6: Effect of adding unconditional guidance to a sampler combining iterative self-conditioning refinement and self-conditioning guidance at NFE = 64.

Gen. PPL-entropy curves nearly overlap under both computational budgets. In the high-entropy, high-Gen. PPL regime, distance-based decoding yields a slightly more favorable trade-off. These results support using nearest-neighbor projection to map the generated continuous flow state to a token sequence, without a separately trained terminal decoder.

4.5 Further improvement

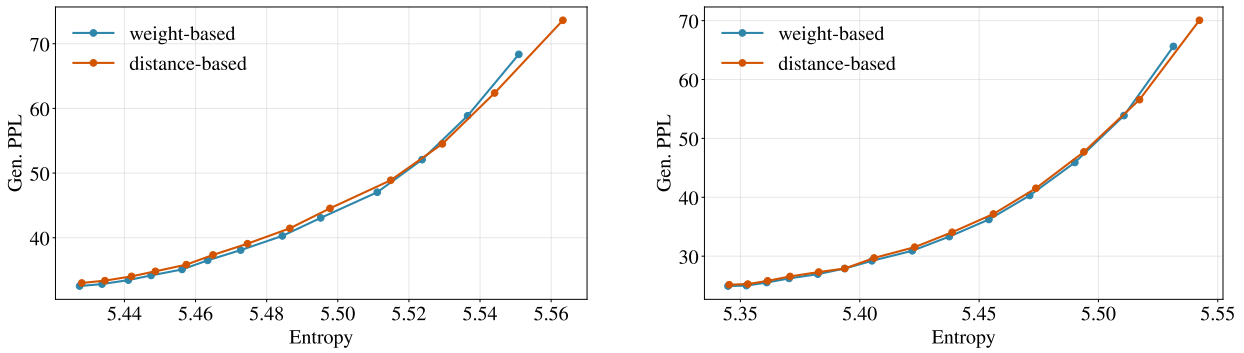
The experiments in Section 4.3 indicate that concentrating sampling guidance near the data endpoint improves the Gen. PPL-entropy trade-off. At sufficiently large guidance strengths, however, data endpoint-focused allocation exhibits diminishing returns. Further reduction in Gen. PPL requires stronger guidance earlier in the trajectory, when the state remains relatively noisy.

To study this effect, we compare the following three configurations under the same NFE budget:

- Configuration A: fixed refinement count $K_i = K$ and $w_{\text{scg}}(t_i) = w_{\text{scg}}$;
- Configuration B: $K_i = \lceil K_{\text{iscr}}/(1 + \sqrt{\sigma_{t_i}/\alpha_{t_i}}) \rceil$ and $w_{\text{scg}}(t_i) = w_{\text{scg}}/(1 + \sqrt{\sigma_{t_i}/\alpha_{t_i}})$;
- Configuration C: $K_i = \lceil K_{\text{iscr}}/(1 + \sigma_{t_i}/\alpha_{t_i}) \rceil$ and $w_{\text{scg}}(t_i) = w_{\text{scg}}/(1 + \sigma_{t_i}/\alpha_{t_i})$.

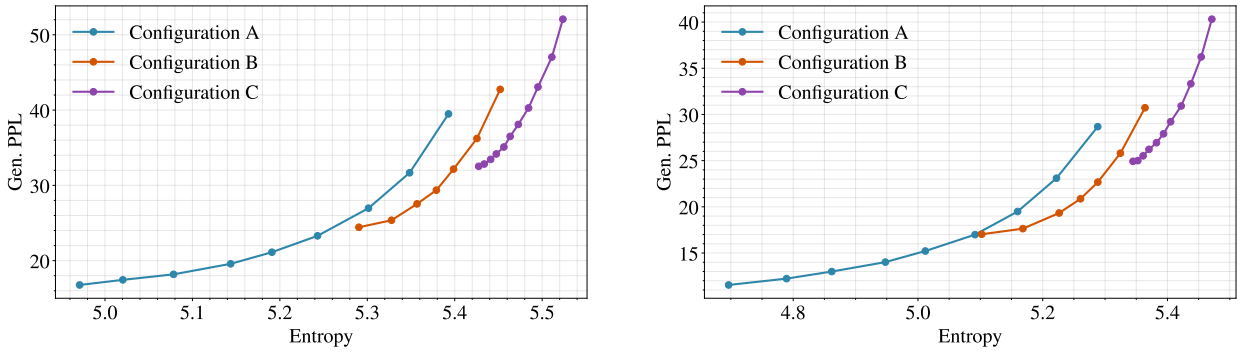
Relative to Configuration C, Configuration B applies stronger guidance in the high-noise regime.

Figure 8 presents the results for NFE=64 and 128. The three configurations are favorable in different regions of the quality-diversity frontier. Configuration A, particularly with $K = 8$, reaches the low-entropy,



(a) NFE=64 (b) NFE=128

Figure 7: Comparison of two token prediction rules for NFE budgets of 64 and 128.



(a) NFE=64 (b) NFE=128

Figure 8: Comparison of the three configurations for NFE budgets of 64 and 128. Each curve is obtained by varying the self-conditioning guidance strength w_{scg}.

low-Gen. PPL regime. Configuration B provides strong intermediate operating points by applying more guidance during the noisier portion of the trajectory, whereas Configuration C preserves the greatest diversity by concentrating guidance near the endpoint.

In regions where the curves overlap, increasing the refinement count generally improves the trade-off. Among the settings considered, $K_{\mathsf{iscr}} = 100$ and $K_{\mathsf{iscr}} = 200$ yield the most favorable frontiers for NFE=64 and 128, respectively. These results show that both the refinement count and its temporal allocation determine the attainable Gen. PPL-entropy regime.

Complete numerical results for Configurations A and B are reported in Tables 8 and 9 in Appendix B.3, respectively. Results for additional guidance configurations under NFE=64 and 128 are provided in Tables 10 and 11 in Appendix B.3.

5 Discussion

ConvergeFlow suggests several directions for future work. First, our formulation keeps the token embeddings fixed to avoid degenerate solutions. It would be interesting to develop fully continuous objectives that support joint learning of the embedding and data predictor. Second, extending the convergence analysis under weaker assumptions and developing non-asymptotic theoretical guarantees would provide a more complete account of practical generation. Third, scaling ConvergeFlow to larger models and evaluating it on conditional generation, instruction following, and reasoning tasks will clarify its broader applicability. Finally, the continuous formulation may facilitate the use of techniques such as distillation and higher-order acceleration, whose effectiveness for language generation remains to be explored.

Acknowledgments

N. Li, Y. Jiao, and G. Li are supported in part by the Chinese University of Hong Kong Direct Grant for Research and the Hong Kong Research Grants Council ECS 24305724 and GRF 14307525. C. Cai is supported in part by the NSF CAREER award CCF-2541600 and grant DMS-2515333.

References

Arriola, M., Gokaslan, A., Chiu, J., Yang, Z., Qi, Z., Han, J., Sahoo, S., and Kuleshov, V. (2025). Block diffusion: Interpolating between autoregressive and diffusion language models. In *International Conference on Learning Representations*, volume 2025, pages 50726–50753.

Austin, J., Johnson, D. D., Ho, J., Tarlow, D., and Van Den Berg, R. (2021). Structured denoising diffusion models in discrete state-spaces. *Advances in Neural Information Processing Systems*, 34:17981–17993.

Ben-Hamu, H., Gat, I., Severo, D., Nolte, N. S., and Karrer, B. (2026). Accelerated sampling from masked diffusion models via entropy bounded unmasking. *Advances in Neural Information Processing Systems*, 38:55981–56007.

Benton, J., De Bortoli, V., Doucet, A., and Deligiannidis, G. (2023). Nearly d-linear convergence bounds for diffusion models via stochastic localization. *arXiv preprint arXiv:2308.03686*.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. (2020). Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Cai, C. and Li, G. (2025). Minimax optimality of the probability flow ode for diffusion models. *arXiv preprint arXiv:2503.09583*.

Cai, C. and Li, G. (2026). Confidence-based decoding is provably efficient for diffusion language models. *arXiv preprint arXiv:2603.22248*.

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

Chen, H., Lee, H., and Lu, J. (2023a). Improved analysis of score-based generative modeling: User-friendly bounds under minimal smoothness assumptions. In *International Conference on Machine Learning*, pages 4735–4763. PMLR.

Chen, S., Chewi, S., Lee, H., Li, Y., Lu, J., and Salim, A. (2023b). The probability flow ode is provably fast. *Advances in Neural Information Processing Systems*, 36:68552–68575.

Chen, S., Chewi, S., Li, J., Li, Y., Salim, A., and Zhang, A. R. (2022a). Sampling is as easy as learning the score: theory for diffusion models with minimal data assumptions. *arXiv preprint arXiv:2209.11215*.

Chen, S., Cong, K., and Li, J. (2025). Optimal inference schedules for masked diffusion models. *arXiv preprint arXiv:2511.04647*.

Chen, T., Zhang, R., and Hinton, G. (2022b). Analog bits: Generating discrete data using diffusion models with self-conditioning. *arXiv preprint arXiv:2208.04202*.

Chen, Y., Liang, C., Sui, H., Guo, R., Cheng, C., You, J., and Liu, G. (2026). Langflow: Continuous diffusion rivals discrete in language modeling. *arXiv preprint arXiv:2604.11748*.

Dhariwal, P. and Nichol, A. (2021). Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794.

Dieleman, S., Sartran, L., Roshannai, A., Savinov, N., Ganin, Y., Richemond, P. H., Doucet, A., Strudel, R., Dyer, C., Durkan, C., et al. (2022). Continuous diffusion for categorical data. *arXiv preprint arXiv:2211.15089*.

Dmitriev, D., Huang, Z., and Wei, Y. (2026). Efficient sampling with discrete diffusion models: Sharp and adaptive guarantees. *arXiv preprint arXiv:2602.15008*.

Esser, P., Kulal, S., Blattmann, A., Entezari, R., Müller, J., Saini, H., Levi, Y., Lorenz, D., Sauer, A., Boesel, F., et al. (2024). Scaling rectified flow transformers for high-resolution image synthesis. In *Forty-first international conference on machine learning*.

Fu, H., Huang, B., Adams, V., Wang, C., Srinivasan, V., and Jiao, J. (2025). From bits to rounds: Parallel decoding with exploration for diffusion language models. *arXiv preprint arXiv:2511.21103*.

Gong, S., Li, M., Feng, J., Wu, Z., and Kong, L. (2022). Diffuseq: Sequence to sequence text generation with diffusion models. *arXiv preprint arXiv:2210.08933*.

Gulrajani, I. and Hashimoto, T. B. (2023). Likelihood-based diffusion language models. *Advances in Neural Information Processing Systems*, 36:16693–16715.

Guo, H., Zhao, Q., Zhao, Y., Nie, S., Zhu, R., Guo, Q., Wang, F., Yang, T., Zhao, H., Wei, G., et al. (2026). Continuous latent diffusion language model. *arXiv preprint arXiv:2605.06548*.

Han, X., Kumar, S., and Tsvetkov, Y. (2023). Ssd-lm: Semi-autoregressive simplex-based diffusion language model for text generation and modular control. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11575–11596.

Ho, J., Jain, A., and Abbeel, P. (2020). Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851.

Ho, J. and Salimans, T. (2022). Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*.

Ho, J., Salimans, T., Gritsenko, A., Chan, W., Norouzi, M., and Fleet, D. J. (2022). Video diffusion models. *Advances in neural information processing systems*, 35:8633–8646.

Hoogeboom, E., Nielsen, D., Jaini, P., Forré, P., and Welling, M. (2021). Argmax flows and multinomial diffusion: Learning categorical distributions. *Advances in neural information processing systems*, 34:12454–12465.

Hu, K., Qiu, L., Lu, Y., Zhao, H., Li, T., Kim, Y., Andreas, J., and He, K. (2026). Elf: Embedded language flows. *arXiv preprint arXiv:2605.10938*.

Jain, N. and Zhang, T. (2026). A sharp kl convergence analysis for diffusion models under minimal assumptions. In *International Conference on Learning Representations*, volume 2026, pages 22706–22735.

Jiao, Y. and Li, G. (2024). Instance-dependent convergence theory for diffusion models. *arXiv preprint arXiv:2410.13738*.

Jiao, Y., Zhou, Y., and Li, G. (2025). Optimal convergence analysis of ddpm for general distributions. *arXiv preprint arXiv:2510.27562*.

Kim, J., Shah, K., Kontonis, V., Kakade, S., and Chen, S. (2025). Train for the worst, plan for the best: Understanding token ordering in masked diffusions. *arXiv preprint arXiv:2502.06768*.

Labs, I., Khanna, S., Kharbanda, S., Li, S., Varma, H., Wang, E., Birnbaum, S., Luo, Z., Miraoui, Y., Palrecha, A., et al. (2025). Mercury: Ultra-fast language models based on diffusion. *arXiv preprint arXiv:2506.17298*.

Lavenant, H. and Zanella, G. (2025). Error bounds and optimal schedules for masked diffusions with factorized approximations. *arXiv preprint arXiv:2510.25544*.

Lee, C., Yoo, J., Agarwal, M., Shah, S., Huang, J., Raghunathan, A., Hong, S., Boffi, N. M., and Kim, J. (2026). Flow map language models: One-step language modeling via continuous denoising. *arXiv preprint arXiv:2602.16813*.

Lee, H., Lu, J., and Tan, Y. (2022). Convergence for score-based generative modeling with polynomial complexity. *Advances in Neural Information Processing Systems*, 35:22870–22882.

Li, G. and Cai, C. (2024). Provable acceleration for diffusion models under minimal assumptions. *arXiv preprint arXiv:2410.23285*.

Li, G. and Cai, C. (2025). Breaking AR's sampling bottleneck: Provable acceleration via diffusion language models. *Advances in Neural Information Processing Systems*, 38:11700–11725.

Li, G., Cai, C., and Wei, Y. (2025a). Dimension-free convergence of diffusion models for approximate gaussian mixtures. *arXiv preprint arXiv:2504.05300*.

Li, G., Wei, Y., Chi, Y., and Chen, Y. (2024). A sharp convergence theory for the probability flow odes of diffusion models. *arXiv preprint arXiv:2408.02320*.

Li, G. and Yan, Y. (2024). $O(d/T)$ convergence theory for diffusion probabilistic models under minimal assumptions. *arXiv preprint arXiv:2409.18959*.

Li, G., Zhou, Y., Wei, Y., and Chen, Y. (2025b). Faster diffusion models via higher-order approximation. *arXiv preprint arXiv:2506.24042*.

Li, X., Thickstun, J., Gulrajani, I., Liang, P. S., and Hashimoto, T. B. (2022). Diffusion-lm improves controllable text generation. *Advances in Neural Information Processing Systems*, 35:4328–4343.

Lipman, Y., Chen, R. T., Ben-Hamu, H., Nickel, M., and Le, M. (2022). Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*.

Liu, X., Gong, C., and Liu, Q. (2022). Flow straight and fast: Learning to generate and transfer data with rectified flow. *arXiv preprint arXiv:2209.03003*.

Loshchilov, I. and Hutter, F. (2017). Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Lou, A., Meng, C., and Ermon, S. (2023). Discrete diffusion modeling by estimating the ratios of the data distribution. *arXiv preprint arXiv:2310.16834*.

Lovelace, J., Kishore, V., Wan, C., Shekhtman, E., and Weinberger, K. Q. (2023). Latent diffusion for language generation. *Advances in Neural Information Processing Systems*, 36:56998–57025.

Lu, C., Zhou, Y., Bao, F., Chen, J., Li, C., and Zhu, J. (2022a). Dpm-solver: A fast ode solver for diffusion probabilistic model sampling in around 10 steps. *Advances in Neural Information Processing Systems*, 35:5775–5787.

Lu, C., Zhou, Y., Bao, F., Chen, J., Li, C., and Zhu, J. (2022b). Dpm-solver++: Fast solver for guided sampling of diffusion probabilistic models. *arXiv preprint arXiv:2211.01095*.

Mahabadi, R. K., Ivison, H., Tae, J., Henderson, J., Beltagy, I., Peters, M. E., and Cohan, A. (2024). Tess: Text-to-text self-conditioned simplex diffusion. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2347–2361.

Meng, C., Choi, K., Song, J., and Ermon, S. (2022). Concrete score matching: Generalized score matching for discrete data. *Advances in Neural Information Processing Systems*, 35:34532–34545.

Meshchaninov, V., Chimbulatov, E., Shabalin, A., Abramov, A., and Vetrov, D. (2026a). Cosmos: Compressed and smooth latent space for text diffusion modeling. *Advances in Neural Information Processing Systems*, 38:14271–14299.

Meshchaninov, V., Shabalin, A., Chimbulatov, E., Gushchin, N., Koziev, I., Korotin, A., and Vetrov, D. (2026b). How to train your latent diffusion language model jointly with the latent space. *arXiv preprint arXiv:2605.07933*.

Nie, S., Zhu, F., You, Z., Zhang, X., Ou, J., Hu, J., Zhou, J., Lin, Y., Wen, J.-R., and Li, C. (2025). Large language diffusion models. *arXiv preprint arXiv:2502.09992*.

Oko, K., Akiyama, S., and Suzuki, T. (2023). Diffusion models are minimax optimal distribution estimators. In *International Conference on Machine Learning*, pages 26517–26582. PMLR.

Ou, J., Nie, S., Xue, K., Zhu, F., Sun, J., Li, Z., and Li, C. (2024). Your absorbing discrete diffusion secretly models the conditional distributions of clean data. *arXiv preprint arXiv:2406.03736*.

Peebles, W. and Xie, S. (2023). Scalable diffusion models with transformers. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 4172–4182. IEEE.

Potaptchik, P., Yim, J., Saravanan, A., Holderrieth, P., Vanden-Eijnden, E., and Albergo, M. S. (2026). Discrete flow maps. *arXiv preprint arXiv:2604.09784*.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I., et al. (2019). Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., and Liu, P. J. (2020). Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67.

Rombach, R., Blattmann, A., Lorenz, D., Esser, P., and Ommer, B. (2022). High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695.

Roos, D., Davis, O., Eijkelboom, F., Bronstein, M., Welling, M., Ceylan, I. I., Ambrogioni, L., and van de Meent, J.-W. (2026). Categorical flow maps. *arXiv preprint arXiv:2602.12233*.

Sahoo, S., Arriola, M., Schiff, Y., Gokaslan, A., Marroquin, E., Chiu, J., Rush, A., and Kuleshov, V. (2024). Simple and effective masked diffusion language models. *Advances in Neural Information Processing Systems*, 37:130136–130184.

Sahoo, S. S., Deschenaux, J., Gokaslan, A., Wang, G., Chiu, J., and Kuleshov, V. (2025). The diffusion duality. *Proceedings of machine learning research*, 267:52584.

Sahoo, S. S., Lemercier, J.-M., Yang, Z., Deschenaux, J., Liu, J., Thickstun, J., and Jukic, A. (2026). Scaling beyond masked diffusion language models. *arXiv preprint arXiv:2602.15014*.

Shi, J., Han, K., Wang, Z., Doucet, A., and Titsias, M. (2024). Simplified and generalized masked diffusion for discrete data. *Advances in neural information processing systems*, 37:103131–103167.

Sohl-Dickstein, J., Weiss, E., Maheswaranathan, N., and Ganguli, S. (2015). Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pages 2256–2265. pmlr.

Song, Y. and Dhariwal, P. (2024). Improved techniques for training consistency models. In *International Conference on Learning Representations*, volume 2024, pages 15078–15097.

Song, Y. and Ermon, S. (2019). Generative modeling by estimating gradients of the data distribution. *Advances in neural information processing systems*, 32.

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

Song, Y., Zhang, Z., Luo, C., Gao, P., Xia, F., Luo, H., Li, Z., Yang, Y., Yu, H., Qu, X., et al. (2025). Seed diffusion: A large-scale diffusion language model with high-speed inference. *arXiv preprint arXiv:2508.02193*.

Tae, J., Ivison, H., Kumar, S., and Cohan, A. (2025). Tess 2: A large-scale generalist diffusion language model. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 21171–21188.

Trippe, B. L., Yim, J., Tischer, D., Baker, D., Broderick, T., Barzilay, R., and Jaakkola, T. (2022). Diffusion probabilistic modeling of protein backbones in 3d for the motif-scaffolding problem. *arXiv preprint arXiv:2206.04119*.

Wainwright, M. J. (2026). The data geometry of masking diffusion: Certified-optimal schedules via unmasking growth complexity. *arXiv preprint arXiv:2608.13520*.

Wakasugi, S. and Suzuki, T. (2026). State size independent statistical error bound for discrete diffusion models. volume 38, pages 138908–138943.

Wan, T., Wang, A., Ai, B., Wen, B., Mao, C., Xie, C.-W., Chen, D., Yu, F., Zhao, H., Yang, J., et al. (2025). Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*.

Wu, C., Zhang, H., Xue, S., Liu, Z., Diao, S., Zhu, L., Luo, P., Han, S., and Xie, E. (2025). Fast-dllm: Training-free acceleration of diffusion llm by enabling kv cache and parallel decoding. *arXiv preprint arXiv:2505.22618*.

Wu, J. and Cai, C. (2026). Diffusion models are statistically optimal for learning low-dimensional multimodal distributions. *arXiv preprint arXiv:2605.30153*.

Ye, J., Xie, Z., Zheng, L., Gao, J., Wu, Z., Jiang, X., Li, Z., and Kong, L. (2025). Dream 7b: Diffusion large language models. *arXiv preprint arXiv:2508.15487*.

Yin, T., Gharbi, M., Zhang, R., Shechtman, E., Durand, F., Freeman, W. T., and Park, T. (2024). One-step diffusion with distribution matching distillation. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6613–6623. IEEE.

You, Z., Nie, S., Zhang, X., Hu, J., Zhou, J., Lu, Z., Wen, J.-R., and Li, C. (2025). Llada-v: Large language diffusion models with visual instruction tuning. *arXiv preprint arXiv:2505.16933*.

Yu, R., Ma, X., and Wang, X. (2025). Dimple: Discrete diffusion multimodal large language model with parallel decoding. *arXiv preprint arXiv:2505.16990*.

Zhang, Y., Gu, J., Wu, Z., Zhai, S., Susskind, J., and Jaitly, N. (2023). Planner: Generating diversified paragraph via latent language diffusion model. *Advances in Neural Information Processing Systems*, 36:80178–80190.

Zhang, Z., Fu, H., Yang, Z., Wang, M., Zhao, T., and Chen, M. (2026). Generalization bounds for discrete diffusion: Statistical advantage of masking. In *International Conference on Machine Learning*.

Zhao, Y. and Cai, C. (2026). Adaptation to intrinsic dependence in diffusion language models. *arXiv preprint arXiv:2602.20126*.

Zheng, K., Chen, Y., Mao, H., Liu, M.-Y., Zhu, J., and Zhang, Q. (2025). Masked diffusion models are secretly time-agnostic masked models and exploit inaccurate categorical sampling. In *International Conference on Learning Representations*, volume 2025, pages 63186–63227.

A Proof of theorem and propositions

A.1 Proof of Theorem 1

In this section, we provide a proof of Theorem 1, organized into four steps.

Step 1: Construction of an auxiliary sequence. Define the probability simplex over the vocabulary as

$$F \coloneqq \Big\{ f \in \mathbb{R}_+^V : \sum_{v\in[V]} f(v) = 1 \Big\}.$$

We first note that given an initial point x_{t_0} and a collection of base weights $\{\widehat{f}_k^{(i)}\}_{0\le k\le N-1, 1\le i\le L} \subset F$ satisfying

$$\max_{j,k,i} \big|\log \widehat{f}_{k+1}^{(i)}(j) - \log \widehat{f}_k^{(i)}(j)\big| \le \widetilde{L}\,|t_{k+1}-t_k|,$$

we can use them to construct an auxiliary flow state sequence $\widehat{x}_{t_k}$ as follows.

Initialize the auxiliary sequence $\widehat{x}_{t_0} = x_{t_0}$. For each token position i and step k, define the auxiliary weights

$$\widehat{w}_k^{(i)}(j \mid \widehat{x}_{t_k}) \coloneqq \frac{\widehat{f}_k^{(i)}(j)\exp\Big(-\frac{\|\widehat{x}_{t_k}^{(i)} - \alpha_{t_k} e_j\|_2^2}{2\sigma_{t_k}^2}\Big)}{\sum_{j=1}^V \widehat{f}_k^{(i)}(j)\exp\Big(-\frac{\|\widehat{x}_{t_k}^{(i)} - \alpha_{t_k} e_j\|_2^2}{2\sigma_{t_k}^2}\Big)}, \quad j\in[V]. \tag{33}$$

Thus, $\widehat{w}_k^{(i)}(j \mid \widehat{x}_{t_k})$ is the counterpart of the learned weight $w_\theta^{(i)}(j|\widehat{x}_{t_k}, t_k)$, obtained by replacing the learned base weight $f_\theta^{(i)}(j \mid x_{t_k}, t_k)$ with the data-independent quantity $\widehat{f}_k^{(i)}(j)$.

The corresponding auxiliary data predictor $\widehat{\mu}_\theta(\widehat{x}_{t_k}, t_k)$ is given by

$$\widehat{\mu}_\theta^{(i)}(\widehat{x}_{t_k}, t_k) \coloneqq \sum_{j=1}^V \widehat{w}_k^{(i)}(j \mid \widehat{x}_{t_k}) e_j. \tag{34}$$

We then iteratively update the auxiliary sequence according to

$$\widehat{x}_{t_{k+1}} = \frac{\sigma_{t_{k+1}}}{\sigma_{t_k}}\widehat{x}_{t_k} + \big(\alpha_{t_{k+1}} - \frac{\sigma_{t_{k+1}}\alpha_{t_k}}{\sigma_{t_k}}\big)\widehat{\mu}_\theta(\widehat{x}_{t_k}, t_k), \qquad k = 0, 1, \ldots, N-1. \tag{35}$$

With this construction procedure in place, we next show that the auxiliary sequence can reproduce the original sampling trajectory. Given the original trajectory $(x_{t_k})_{k=0}^N$, choose

$$\widehat{f}_k^{(i)}(j) \coloneqq f_\theta^{(i)}(j \mid x_{t_k}, t_k), \qquad j\in[V], \tag{36}$$

for each iteration $0 \le k < N$ and token position $i \in [L]$. The log-Lipschitz assumption in Theorem 1 ensures that this sequence satisfies the required regularity condition. We claim that

$$x_{t_k} = \widehat{x}_{t_k}, \qquad 0 \le k \le N. \tag{37}$$

We prove this claim by induction. First, it holds trivially for $k = 0$. Next, suppose that $x_{t_k} = \widehat{x}_{t_k}$ holds for some k. By the choice of $\widehat{f}_k^{(i)}(j)$ in (36) and the construction of $\widehat{w}_k^{(i)}$ in (33), we have

$$\widehat{w}_k^{(i)}(j \mid \widehat{x}_{t_k}) = w_\theta^{(i)}(j|\widehat{x}_{t_k}, t_k) = w_\theta^{(i)}(j \mid x_{t_k}, t_k). \tag{38}$$

It then follows from (34) and (22) that

$$\widehat{\mu}_\theta^{(i)}(\widehat{x}_{t_k}, t_k) = \sum_{j=1}^V w_\theta^{(i)}(j \mid x_{t_k}, t_k) e_j = \mu_\theta^{(i)}(x_{t_k}, t_k).$$

Therefore, the update rule of the auxiliary sequence in (35) coincides with that of the original sequence in (10), thereby yielding $x_{t_{k+1}} = \widehat{x}_{t_{k+1}}$. This completes the induction and proves (37).

Consequently, it remains to analyze the auxiliary sequence for an arbitrary collection of base weights satisfying the stated regularity condition. Concretely, we will show that for any token position $i \in [L]$, there exists some $j_i \in [V]$, such that

$$\widehat{x}_{t_N}^{(i)} \to e_{j_i}, \qquad \text{as} \quad t_N \to 1.$$

Step 2: Construction of reference distributions. For each token position i, let $q_k^{(i)}$ denote the probability density function of $\widehat{x}_{t_k}^{(i)}$ when $\widehat{x}_{t_0}^{(i)}$ is initialized with the standard Gaussian distribution. In addition, for each $0 \le k \le N$ and $i \in [L]$, we define the reference distribution

$$\widehat{p}_k^{(i)}(x) \coloneqq (2\pi\sigma_{t_k}^2)^{-d/2} \sum_{j=1}^{V} \widehat{f}_k^{(i)}(j) \exp\Big(-\frac{\|x - \alpha_{t_k} e_j\|_2^2}{2\sigma_{t_k}^2}\Big), \qquad \forall x \in \mathbb{R}^d.$$

In the remainder of the proof, we focus on a token position i. For notational brevity, we omit the superscript (i) from $\widehat{f}_k^{(i)}$, $\widehat{w}_k^{(i)}$, $\widehat{\mu}_\theta^{(i)}$, $\widehat{p}_k^{(i)}$ and $\widehat{q}_k^{(i)}$, and $\widehat{x}_{t_k}^{(i)}$.

We first compare the reference densities at two consecutive points along the auxiliary trajectory. By the definition of $\widehat{p}_k$, one can derive

$$\begin{aligned}
\frac{\widehat{p}_{k+1}(\widehat{x}_{t_{k+1}})}{\widehat{p}_k(\widehat{x}_{t_k})} &= \frac{1}{\widehat{p}_k(\widehat{x}_{t_k})} (2\pi\sigma_{t_{k+1}}^2)^{-d/2} \sum_{j=1}^{V} \widehat{f}_{k+1}(j) \exp\Big(-\frac{\|\widehat{x}_{t_{k+1}} - \alpha_{t_{k+1}} e_j\|_2^2}{2\sigma_{t_{k+1}}^2}\Big) \\
&= \frac{1}{\widehat{p}_k(\widehat{x}_{t_k})} (2\pi\sigma_{t_k}^2)^{-d/2} \Big(\frac{\sigma_{t_k}}{\sigma_{t_{k+1}}}\Big)^d \sum_{j=1}^{V} \widehat{f}_k(j) \exp\Big(-\frac{\|\widehat{x}_{t_k} - \alpha_{t_k} e_j\|_2^2}{2\sigma_{t_k}^2}\Big) \frac{\widehat{f}_{k+1}(j)}{\widehat{f}_k(j)} \exp\big(\Delta_{t_k}(e_j)\big) \\
&\overset{\text{(i)}}{\ge} \exp\big(-\widetilde{L}(t_{k+1} - t_k)\big) \frac{1}{\widehat{p}_k(\widehat{x}_{t_k})} (2\pi\sigma_{t_k}^2)^{-d/2} \Big(\frac{\sigma_{t_k}}{\sigma_{t_{k+1}}}\Big)^d \sum_{j=1}^{V} \widehat{f}_k(j) \exp\Big(-\frac{\|\widehat{x}_{t_k} - \alpha_{t_k} e_j\|_2^2}{2\sigma_{t_k}^2}\Big) \exp\big(\Delta_{t_k}(e_j)\big) \\
&\overset{\text{(ii)}}{=} \exp\big(-\widetilde{L}(t_{k+1} - t_k)\big) \Big(\frac{\sigma_{t_k}}{\sigma_{t_{k+1}}}\Big)^d \sum_{j=1}^{V} \widehat{w}_k(j \mid \widehat{x}_{t_k}) \exp\big(\Delta_{t_k}(e_j)\big) \\
&\overset{\text{(iii)}}{\ge} \exp\big(-\widetilde{L}(t_{k+1} - t_k)\big) \Big(\frac{\sigma_{t_k}}{\sigma_{t_{k+1}}}\Big)^d \exp\Big(\sum_{j=1}^{V} \widehat{w}_k(j \mid \widehat{x}_{t_k}) \Delta_{t_k}(e_j)\Big).
\end{aligned} \tag{39}$$

Here, $\Delta_{t_k}(e_j)$ records the change in the Gaussian exponent associated with the j-th token embedding:

$$\begin{aligned}
\Delta_{t_k}(e_j) &\coloneqq \frac{\|\widehat{x}_{t_k} - \alpha_{t_k} e_j\|_2^2}{2\sigma_{t_k}^2} - \frac{\|\widehat{x}_{t_{k+1}} - \alpha_{t_{k+1}} e_j\|_2^2}{2\sigma_{t_{k+1}}^2} \\
&= \frac{1}{2}\Big(1 - \frac{\alpha_{t_{k+1}}^2 \sigma_{t_k}^2}{\alpha_{t_k}^2 \sigma_{t_{k+1}}^2}\Big) \frac{\|\widehat{x}_{t_k} - \alpha_{t_k} e_j\|_2^2}{\sigma_{t_k}^2} - \frac{1}{2}\Big(1 - \frac{\alpha_{t_{k+1}}^2 \sigma_{t_k}^2}{\alpha_{t_k}^2 \sigma_{t_{k+1}}^2}\Big) \frac{\|\widehat{x}_{t_k} - \alpha_{t_k} \widehat{\mu}_\theta(\widehat{x}_{t_k}, t_k)\|_2^2}{\sigma_{t_k}^2},
\end{aligned} \tag{40}$$

where the second line follows from the update rule of $\widehat{x}_{t_k}$ given in (35) and the definition of $\widehat{\mu}_\theta(\widehat{x}_{t_k}, t_k)$ from (34); (i) follows from the log-Lipschitz condition on $\widehat{f}_k$; (ii) follows from the identity

$$\widehat{w}_k(j \mid \widehat{x}_{t_k}) = \frac{\widehat{f}_k(j) \exp\Big(-\frac{\|\widehat{x}_{t_k} - \alpha_{t_k} e_j\|_2^2}{2\sigma_{t_k}^2}\Big)}{\sum_{j'=1}^{V} \widehat{f}_k(j') \exp\Big(-\frac{\|\widehat{x}_{t_k} - \alpha_{t_k} e_{j'}\|_2^2}{2\sigma_{t_k}^2}\Big)} = \frac{1}{\widehat{p}_k(\widehat{x}_{t_k})} (2\pi\sigma_{t_k}^2)^{-d/2} \widehat{f}_k(j) \exp\Big(-\frac{\|\widehat{x}_{t_k} - \alpha_{t_k} e_j\|_2^2}{2\sigma_{t_k}^2}\Big);$$

(iii) is a consequence of Jensen's inequality $\mathbb{E}[e^X] \ge e^{\mathbb{E}[X]}$. By summary, we have

$$\begin{aligned}
\sum_{j=1}^{V} \widehat{w}_k(j \mid \widehat{x}_{t_k}) \Delta_{t_k}(e_j) &= \frac{1}{2}\Big(1 - \frac{\alpha_{t_{k+1}}^2 \sigma_{t_k}^2}{\alpha_{t_k}^2 \sigma_{t_{k+1}}^2}\Big) \Bigg(\sum_{j=1}^{V} \widehat{w}_k(j \mid \widehat{x}_{t_k}) \frac{\|\widehat{x}_{t_k} - \alpha_{t_k} e_j\|_2^2}{\sigma_{t_k}^2} - \frac{\|\widehat{x}_{t_k} - \alpha_{t_k} \widehat{\mu}_\theta(\widehat{x}_{t_k}, t_k)\|_2^2}{\sigma_{t_k}^2} \Bigg) \\
&= \frac{\alpha_{t_k}^2}{2\sigma_{t_k}^2}\Big(1 - \frac{\alpha_{t_{k+1}}^2 \sigma_{t_k}^2}{\alpha_{t_k}^2 \sigma_{t_{k+1}}^2}\Big) \Bigg(\sum_{j=1}^{V} \widehat{w}_k(j \mid \widehat{x}_{t_k}) \|e_j\|_2^2 - \|\widehat{\mu}_\theta(\widehat{x}_{t_k}, t_k)\|_2^2 \Bigg).
\end{aligned} \tag{41}$$

Substituting into (39) yields

$$\frac{\widehat{p}_{k+1}(\widehat{x}_{t_{k+1}})}{\widehat{p}_k(\widehat{x}_{t_k})} \ge \exp\big(-\widetilde{L}(t_{k+1} - t_k)\big) \Big(\frac{\sigma_{t_k}}{\sigma_{t_{k+1}}}\Big)^d \exp\Bigg(\frac{\alpha_{t_k}^2}{2\sigma_{t_k}^2}\Big(1 - \frac{\alpha_{t_{k+1}}^2 \sigma_{t_k}^2}{\alpha_{t_k}^2 \sigma_{t_{k+1}}^2}\Big) \Bigg(\sum_{j=1}^{V} \widehat{w}_k(j \mid \widehat{x}_{t_k}) \|e_j\|_2^2 - \|\widehat{\mu}_\theta(\widehat{x}_{t_k}, t_k)\|_2^2 \Bigg) \Bigg). \tag{42}$$

Step 3: Analysis of the distributions of $\widehat{x}_{t_k}$. We now track the density of the auxiliary sequence $\widehat{x}_{t_k}$. Because the update is deterministic, its one-step density ratio is determined by the inverse Jacobian determinant of the update map. From the update rule in (35), the change-of-variable formula yields the following relationship between the densities of $\widehat{x}_{t_k}$ and $\widehat{x}_{t_{k+1}}$:

$$\begin{aligned}\frac{q_{k+1}(\widehat{x}_{t_{k+1}})}{q_k(\widehat{x}_{t_k})} &= \Big|\frac{\sigma_{t_{k+1}}}{\sigma_{t_k}} I + \big(\alpha_{t_{k+1}} - \frac{\sigma_{t_{k+1}}\alpha_{t_k}}{\sigma_{t_k}}\big)\nabla_{\widehat{x}_{t_k}}\widehat{\mu}_\theta(\widehat{x}_{t_k}, t_k)\Big|^{-1} \\ &= \Big(\frac{\sigma_{t_{k+1}}}{\sigma_{t_k}}\Big)^{-d}\Big|I - \alpha_{t_k}\Big(1 - \frac{\alpha_{t_{k+1}}\sigma_{t_k}}{\alpha_{t_k}\sigma_{t_{k+1}}}\Big)\nabla_{\widehat{x}_{t_k}}\widehat{\mu}_\theta(\widehat{x}_{t_k}, t_k)\Big|^{-1}. \end{aligned} \tag{43}$$

where $|\cdot|$ denotes the determinant. By the construction of the embedding-weighted predictor $\widehat{\mu}_\theta(\widehat{x}_{t_k}, t_k)$ in (34), we obtain

$$\begin{aligned}\nabla_{\widehat{x}_{t_k}}\widehat{\mu}_\theta(\widehat{x}_{t_k}, t_k) &= \sum_{j=1}^{V} e_j\big(\nabla_{\widehat{x}_{t_k}}\widehat{w}_k(j \mid \widehat{x}_{t_k})\big)^\top \\ &= -\frac{1}{\sigma_{t_k}^2}\sum_{j=1}^{V}\widehat{w}_k(j \mid \widehat{x}_{t_k})e_j(\widehat{x}_{t_k} - \alpha_{t_k}e_j)^\top + \frac{1}{\sigma_{t_k}^2}\big(\sum_{j=1}^{V}\widehat{w}_k(j \mid \widehat{x}_{t_k})e_j\big)\big(\sum_{j=1}^{V}\widehat{w}_k(j \mid \widehat{x}_{t_k})(\widehat{x}_{t_k} - \alpha_{t_k}e_j)\big)^\top \\ &= \frac{\alpha_{t_k}}{\sigma_{t_k}^2}\left(\sum_{j=1}^{V}\widehat{w}_k(j \mid \widehat{x}_{t_k})e_je_j^\top - \big(\sum_{j=1}^{V}\widehat{w}_k(j \mid \widehat{x}_{t_k})e_j\big)\big(\sum_{j=1}^{V}\widehat{w}_k(j \mid \widehat{x}_{t_k})e_j\big)^\top\right) \\ &\overset{\text{(i)}}{=} \frac{\alpha_{t_k}}{\sigma_{t_k}^2}\left(\sum_{j=1}^{V}\widehat{w}_k(j \mid \widehat{x}_{t_k})e_je_j^\top - \widehat{\mu}_\theta(\widehat{x}_{t_k}, t_k)\widehat{\mu}_\theta(\widehat{x}_{t_k}, t_k)^\top\right) \\ &=: \frac{\alpha_{t_k}}{\sigma_{t_k}^2}\widehat{\Sigma}_k(\widehat{x}_{t_k}),\end{aligned}$$

where (i) applies the definition of $\widehat{\mu}_\theta(\widehat{x}_{t_k}, t_k)$ in (34), and $\widehat{\Sigma}_k(\widehat{x}_{t_k})$ is the covariance matrix of discrete distribution on $\{e_j\}_{j\in[V]}$ associated with the probability $\mathbb{P}\{X = e_j\} = \widehat{w}_k(j \mid \widehat{x}_{t_k})$. This is the same quantity that appears in the reference-density calculation in Step 2; see (41).

Substituting this expression into (43) yields

$$\begin{aligned}\frac{q_{k+1}(\widehat{x}_{t_{k+1}})}{q_k(\widehat{x}_{t_k})} &= \Big(\frac{\sigma_{t_{k+1}}}{\sigma_{t_k}}\Big)^{-d}\Big|I - \frac{\alpha_{t_k}^2}{\sigma_{t_k}^2}\Big(1 - \frac{\alpha_{t_{k+1}}\sigma_{t_k}}{\alpha_{t_k}\sigma_{t_{k+1}}}\Big)\widehat{\Sigma}_k(\widehat{x}_{t_k})\Big|^{-1} \\ &\overset{\text{(i)}}{=} \Big(\frac{\sigma_{t_{k+1}}}{\sigma_{t_k}}\Big)^{-d}\exp\left(\frac{\alpha_{t_k}^2}{\sigma_{t_k}^2}\Big(1 - \frac{\alpha_{t_{k+1}}\sigma_{t_k}}{\alpha_{t_k}\sigma_{t_{k+1}}}\Big)\mathsf{Tr}(\widehat{\Sigma}_k(\widehat{x}_{t_k})) + \frac{\alpha_{t_k}^4}{\sigma_{t_k}^4}\Big(1 - \frac{\alpha_{t_{k+1}}\sigma_{t_k}}{\alpha_{t_k}\sigma_{t_{k+1}}}\Big)^2 O\big(\|\widehat{\Sigma}_k(\widehat{x}_{t_k})\|_F^2\big)\right) \\ &= \Big(\frac{\sigma_{t_{k+1}}}{\sigma_{t_k}}\Big)^{-d}\exp\left(\frac{\alpha_{t_k}^2}{\sigma_{t_k}^2}\Big(1 - \frac{\alpha_{t_{k+1}}\sigma_{t_k}}{\alpha_{t_k}\sigma_{t_{k+1}}}\Big)\Big(\sum_{j=1}^{V}\widehat{w}_k(j \mid \widehat{x}_{t_k})\|e_j\|_2^2 - \|\widehat{\mu}_\theta(\widehat{x}_{t_k}, t_k)\|_2^2\Big) + \mathcal{E}_{t_k}(\widehat{x}_{t_k}).\right)\end{aligned} \tag{44}$$

Here the leading term in the right-hand-side of (i) is obtained from the first-order expansion of the log determinant, while the remainder collects the corresponding higher-order terms, and

$$\mathcal{E}_{t_k}(\widehat{x}_{t_k}) := \frac{\alpha_{t_k}^4}{\sigma_{t_k}^4}\Big(1 - \frac{\alpha_{t_{k+1}}\sigma_{t_k}}{\alpha_{t_k}\sigma_{t_{k+1}}}\Big)^2 O\big(\|\widehat{\Sigma}_k(\widehat{x}_{t_k})\|_F^2\big).$$

Step 4: Combining the estimates. We finally compare the two density evolutions. Combining (42) and (44), and using the identity

$$1 - \frac{\alpha_{t_{k+1}}\sigma_{t_k}}{\alpha_{t_k}\sigma_{t_{k+1}}} - \left(\frac{1}{2} - \frac{\alpha_{t_{k+1}}^2\sigma_{t_k}^2}{2\alpha_{t_k}^2\sigma_{t_{k+1}}^2}\right) = \frac{1}{2}\left(1 - \frac{\alpha_{t_{k+1}}\sigma_{t_k}}{\alpha_{t_k}\sigma_{t_{k+1}}}\right)^2,$$

we obtain

$$\begin{aligned}
&\frac{q_{k+1}(\widehat{x}_{t_{k+1}})\widehat{p}_k(\widehat{x}_{t_k})}{q_k(\widehat{x}_{t_k})\widehat{p}_{k+1}(\widehat{x}_{t_{k+1}})} \\
&\quad\le \exp\big(\widetilde{L}(t_{k+1}-t_k)\big)\exp\left(\frac{\alpha_{t_k}^2}{2\sigma_{t_k}^2}\Big(1-\frac{\alpha_{t_{k+1}}\sigma_{t_k}}{\alpha_{t_k}\sigma_{t_{k+1}}}\Big)^2\Big(\sum_{j=1}^V \widehat{w}_k(j\mid \widehat{x}_{t_k})\|e_j\|_2^2 - \|\widehat{\mu}_\theta(\widehat{x}_{t_k},t_k)\|_2^2\Big) + \mathcal{E}_{t_k}(\widehat{x}_{t_k})\right) \\
&\quad\le \exp\big(\widetilde{L}(t_{k+1}-t_k)\big)\exp\left(\frac{\alpha_{t_k}^2}{2\sigma_{t_k}^2}\Big(1-\frac{\alpha_{t_{k+1}}\sigma_{t_k}}{\alpha_{t_k}\sigma_{t_{k+1}}}\Big)^2\sum_{j=1}^V \widehat{w}_k(j\mid \widehat{x}_{t_k})\|e_j\|_2^2 + \mathcal{E}_{t_k}(\widehat{x}_{t_k})\right).
\end{aligned}$$

To control the right-hand side, recall that $\|e_j\|_2$ is bounded, i.e., $\|e_j\|_2 = B$. Consequently, telescoping the one-step density-ratio bounds gives

$$\begin{aligned}
\frac{q_N(\widehat{x}_{t_N})}{\widehat{p}_N(\widehat{x}_{t_N})} &= \frac{q_0(\widehat{x}_{t_0})}{\widehat{p}_0(\widehat{x}_{t_0})}\prod_{k=0}^{N-1}\frac{q_{k+1}(\widehat{x}_{t_{k+1}})\widehat{p}_k(\widehat{x}_{t_k})}{q_k(\widehat{x}_{t_k})\widehat{p}_{k+1}(\widehat{x}_{t_{k+1}})} \\
&\le \frac{q_0(\widehat{x}_{t_0})}{\widehat{p}_0(\widehat{x}_{t_0})}\exp(\widetilde{L})\exp\Big(\sum_{k=0}^{N-1}\frac{\alpha_{t_k}^2}{2\sigma_{t_k}^2}\Big(1-\frac{\alpha_{t_{k+1}}\sigma_{t_k}}{\alpha_{t_k}\sigma_{t_{k+1}}}\Big)^2 B^2 + \sum_{k=0}^{N-1}\mathcal{E}_{t_k}(\widehat{x}_{t_k})\Big),
\end{aligned}$$

where we have used

$$\begin{aligned}
&\sum_{k=0}^{N-1}\frac{\alpha_{t_k}^2}{2\sigma_{t_k}^2}\Big(1-\frac{\alpha_{t_{k+1}}\sigma_{t_k}}{\alpha_{t_k}\sigma_{t_{k+1}}}\Big)^2\sum_{j=1}^V \widehat{w}_k(j\mid \widehat{x}_{t_k})\|e_j\|_2^2 \\
&= \frac{1}{2}\sum_{k=0}^{N-1}\Big(\frac{\alpha_{t_k}}{\sigma_{t_k}} - \frac{\alpha_{t_{k+1}}}{\sigma_{t_{k+1}}}\Big)^2 B^2 \lesssim N\delta^2,
\end{aligned}$$

and

$$\begin{aligned}
\sum_{k=0}^{N-1}\mathcal{E}_{t_k}(\widehat{x}_{t_k}) &\lesssim \sum_{k=0}^{N-1}\frac{\alpha_{t_k}^4}{\sigma_{t_k}^4}\Big(1-\frac{\alpha_{t_{k+1}}\sigma_{t_k}}{\alpha_{t_k}\sigma_{t_{k+1}}}\Big)^2\mathsf{Tr}(\widehat{\Sigma}_k(\widehat{x}_{t_k}))^2 \\
&\lesssim \sum_{k=0}^{N-1}\frac{\alpha_{t_k}^4}{\sigma_{t_k}^4}\Big(1-\frac{\alpha_{t_{k+1}}\sigma_{t_k}}{\alpha_{t_k}\sigma_{t_{k+1}}}\Big)^2\mathsf{Tr}\Big(\sum_{j=1}^V \widehat{w}_k(j\mid \widehat{x}_{t_k})\|e_j\|_2^2\Big)^2 \\
&= \sum_{k=0}^{N-1}\frac{\alpha_{t_k}^2}{\sigma_{t_k}^2}\Big(\frac{\alpha_{t_k}}{\sigma_{t_k}} - \frac{\alpha_{t_{k+1}}}{\sigma_{t_{k+1}}}\Big)^2 B^4 \lesssim N\delta^2 B^4,
\end{aligned}$$

where the last inequality uses the facts that

$$\begin{aligned}
\Big|\frac{\alpha_{t_k}}{\sigma_{t_k}} - \frac{\alpha_{t_{k+1}}}{\sigma_{t_{k+1}}}\Big| &= \frac{t_{k+1}-t_k}{(1-t_k)(1-t_{k+1})} \le \delta, \\
\frac{\alpha_{t_k}}{\sigma_{t_k}}\Big|\frac{\alpha_{t_k}}{\sigma_{t_k}} - \frac{\alpha_{t_{k+1}}}{\sigma_{t_{k+1}}}\Big| &= \frac{(t_{k+1}-t_k)t_k}{(1-t_k)^2(1-t_{k+1})} \le \delta.
\end{aligned} \tag{45}$$

It remains to control the initial density ratio. Observe that the initial time step satisfies

$$\frac{q_0(\widehat{x}_{t_0})}{\widehat{p}_0(\widehat{x}_{t_0})} = \sigma_{t_0}^d\frac{1}{\sum_{j=1}^V \widehat{f}_k(j)\exp\big(-\frac{\alpha_{t_0}^2\|e_j\|_2^2}{2\sigma_{t_0}^2} + \frac{\alpha_{t_0}e_j^\top\widehat{x}_{t_0}}{2\sigma_{t_0}^2} - \frac{1-\sigma_{t_0}^2}{2\sigma_{t_0}^2}\|\widehat{x}_{t_0}\|_2^2\big)} \to 1 \quad \text{as} \quad t_0\to 0.$$

Under the stated discretization condition, the accumulated remainder remains bounded. In particular,for sufficiently small δ, we have $\delta^2 N = O(1)$. Consequently, as $t_0 \to 0$,

$$\frac{q_N(\widehat{x}_{t_N})}{\widehat{p}_N(\widehat{x}_{t_N})} = O(1).$$

Finally, recall that

$$\widehat{p}_N(\widehat{x}_{t_N}) = (2\pi\sigma_{t_N}^2)^{-d/2} \sum_{j=1}^{V} \widehat{w}_N(e_j) \exp\bigl(-\|\widehat{x}_{t_N} - \alpha_{t_N} e_j\|_2^2/(2\sigma_{t_N}^2)\bigr),$$

which implies that

$$q_N(\widehat{x}_{t_N}) \lesssim (2\pi\sigma_{t_N}^2)^{-d/2} \sum_{j=1}^{V} \widehat{w}_N(e_j) \exp\bigl(-\|\widehat{x}_{t_N} - \alpha_{t_N} e_j\|_2^2/(2\sigma_{t_N}^2)\bigr).$$

As $t_N \to 1$, the variance of every Gaussian component vanishes and its center approaches a vocabulary embedding. Therefore, q_N is asymptotically bounded by $\sum_{j=1}^{V} \widehat{w}_N(e_j)\delta_{e_j}$, where δ_{e_j} denotes the Dirac measure with mass at e_j. This completes the proof.

A.2 Proof of Proposition 1

Define

$$p^{(i)}(j \mid x_t^{(-i)}, t) \coloneqq \mathbb{P}\{s^{(i)} = j \mid x_t^{(-i)}\}, \quad j \in [V]. \tag{46}$$

By Bayes's rule, we can derive

$$\mathbb{P}\{s^{(i)} = j \mid x_t\} = \frac{\mathbb{P}\{s^{(i)} = j, x_t^{(i)} \mid x_t^{(-i)}\}}{\mathbb{P}\{x_t^{(i)} \mid x_t^{(-i)}\}} = \frac{p^{(-i)}(j \mid x_t^{(-i)}, t) p(x_t^{(i)} \mid s^{(i)} = j, x_t^{(-i)})}{\sum_{j' \in [V]} p^{(-i)}(j' \mid x_t^{(-i)}, t) p(x_t^{(i)} \mid s^{(i)} = j', x_t^{(-i)})}. \tag{47}$$

Because the Gaussian corruption is independent across token positions (see the probability path construction in (1)), $x_t^{(i)}$ and $x_t^{(-i)}$ are conditionally independent given $s^{(i)}$. As a result, we have

$$x_t^{(i)} \mid s^{(i)} = j, x_t^{(-i)} \sim \mathcal{N}(\alpha_t e_j, \sigma_t^2 I_d).$$

Plugging this into (47) yields the claim in (20).

A.3 Proof of Proposition 2

Consider a general schedule (α_t, σ_t) such that $\alpha_t \to 1$ and $\sigma_t \to 0$ as $t \to 1$. Define the smooth, unconstrained data predictor

$$\mu_\theta(x, t) = \frac{x}{\alpha_t + \sigma_t}. \tag{48}$$

It is easy to verify that

$$\mu_\theta(x_t, t) = \frac{x_t}{\alpha_t + \sigma_t} = \frac{\alpha_t x_\star + \sigma_t z}{\alpha_t + \sigma_t} \to x_\star \quad \text{as} \quad t \to 1.$$

Thus, the data predictor is asymptotically accurate. However, we will show that the flow induced by this data predictor does not converge to any token embedding with constant probability.

By the ODE in (2) and the velocity identity in (8), the flow induced by μ_θ is given by

$$\frac{\mathrm{d}x_t}{\mathrm{d}t} = \frac{\sigma_t'}{\sigma_t} x_t + \Bigl(\alpha_t' - \frac{\sigma_t'}{\sigma_t}\alpha_t\Bigr) \frac{x_t}{\alpha_t + \sigma_t} = \frac{\alpha_t' + \sigma_t'}{\alpha_t + \sigma_t} x_t.$$

Solving the ODE shows that the flow trajectory initialized at x_{t_0} evolves according to

$$x_t = \frac{\alpha_t + \sigma_t}{\alpha_{t_0} + \sigma_{t_0}} x_{t_0}.$$

As a result, the data predictor remains constant along the flow trajectory:

$$\mu_\theta(x_t, t) = \frac{x_t}{\alpha_t + \sigma_t} = \frac{x_{t_0}}{\alpha_{t_0} + \sigma_{t_0}}.$$

Suppose that $x_{t_0} = \sigma z$ with $z_{ij} \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0,1)$. Then the final iterate is given by

$$\frac{x_{t_N}}{\alpha_{t_N}} = \sigma_N z \qquad \text{with} \qquad \sigma_N \coloneqq \frac{\sigma(\alpha_{t_N} + \sigma_{t_N})}{\alpha_{t_N}(\alpha_{t_0} + \sigma_{t_0})}.$$

Because $\sigma_t/\alpha_t \to 0$ as $t \to 1$, we have

$$\sigma_N \downarrow \sigma_\infty \coloneqq \frac{\sigma}{\alpha_{t_0} + \sigma_{t_0}} > 0 \quad \text{as} \quad N \to \infty.$$

Because $\sigma_\infty z$ has a continuous distribution, its probability of belonging to any finite set is zero. Therefore, the final iterate x_{t_N} fails to converge to any token embedding with constant probability, as formalized in (24).

For example, if $\|e_j\|_2 = \sqrt{d}$ for every $j \in [V]$, then for any token position $i \in [L]$, one has

$$\begin{aligned}\mathbb{P}\Big\{\min_j \big\|\alpha_{t_N}^{-1} x_{t_N}^{(i)} - e_j\big\|_2 \geq 1\Big\} &\geq \mathbb{P}\Big\{\big|\sigma_N \|z^{(i)}\|_2 - \|e_j\|_2\big| \geq 1\Big\} \\ &\geq \mathbb{P}\Big\{\sigma_\infty \|z^{(i)}\|_2 \geq 1 + \sqrt{d}\Big\} \geq c_{\mathsf{lb}}\end{aligned}$$

for some constant $c_{\mathsf{lb}} > 0$ independent of N.

A.4 Proof of Proposition 3

Let us denote by $\bar{e}_j \coloneqq e_j/\|e_j\|_2$ the normalized embedding vector. Fix an arbitrary token position $i \in [L]$. Notice the relation

$$\begin{aligned}\frac{\|\mu_\theta^{(i)}(x_t,t)\|_2^2}{\|e_j\|_2^2} &= \Big\| \sum_{j'=1}^V w_\theta^{(i)}(j' \,|\, x_t, t)\bar{e}_{j'} \Big\|_2^2 \\ &= \sum_{j'=1}^V w_\theta^{(i)}(j' \,|\, x_t, t)^2 + \sum_{j'=1}^V \sum_{\ell \neq j'} w_\theta^{(i)}(j' \,|\, x_t, t) w_\theta^{(i)}(\ell \,|\, x_t, t) \frac{\langle e_{j'}, e_\ell \rangle}{\|e_{j'}\|_2 \|e_\ell\|_2}.\end{aligned}$$

By the separation condition that $\max_{j' \neq \ell} \big|\frac{\langle e_{j'}, e_\ell\rangle}{\|e_{j'}\|_2\|e_\ell\|_2}\big| \leq 1 - \rho$, one can derive

$$\begin{aligned}\frac{\|\mu_\theta^{(i)}(x_t,t)\|_2^2}{\|e_j\|_2^2} &\leq \sum_{j'=1}^V w_\theta^{(i)}(j' \,|\, x_t, t)^2 + (1-\rho) \sum_{j'=1}^V \sum_{\ell \neq j'} w_\theta^{(i)}(j' \,|\, x_t, t) w_\theta^{(i)}(\ell \,|\, x_t, t) \\ &\overset{\text{(i)}}{=} \sum_{j'=1}^V w_\theta^{(i)}(j' \,|\, x_t, t)^2 + (1-\rho) \sum_{j'=1}^V w_\theta^{(i)}(j' \,|\, x_t, t)\big(1 - w_\theta^{(i)}(j' \,|\, x_t, t)\big) \\ &\overset{\text{(ii)}}{=} \sum_{j'=1}^V w_\theta^{(i)}(j' \,|\, x_t, t)^2 + (1-\rho) - (1-\rho) \sum_{j'=1}^V w_\theta^{(i)}(j' \,|\, x_t, t)^2 \\ &= 1 - \rho + \rho \sum_{j'=1}^V w_\theta^{(i)}(j' \,|\, x_t, t)^2, \end{aligned} \tag{49}$$

where (i) and (ii) use the fact that $\sum_{j'=1}^V w_\theta^{(i)}(j' \,|\, x_t, t) = 1$. As $\mu_\theta^{(i)}(x_t, t) \to e_j$, one has $\frac{\|\mu_\theta^{(i)}(x_t,t)\|_2^2}{\|e_j\|_2^2} \to 1$. Combining this with (49), we obtain

$$1 - \rho + \rho \lim_{t \to 1} \sum_{j'=1}^V w_\theta^{(i)}(j' \,|\, x_t, t)^2 \geq 1,$$

which leads to

$$\lim_{t\to 1}\sum_{j'=1}^{V} w_\theta^{(i)}(j' \mid x_t, t)^2 \geq 1.$$

As $w_\theta^{(i)}(j' \mid x_t, t) \in [0,1]$ and $\sum_{j'=1}^{V} w_\theta^{(i)}(j' \mid x_t, t) = 1$, there exists some $\widehat{j} \in [V]$ such as $w_\theta^{(i)}(\widehat{j} \mid x_t, t) \to 1$ and $w_\theta^{(i)}(j' \mid x_t, t) \to 0$ for $j' \neq \widehat{j}$. Since $\mu_\theta^{(i)}(x_t, t) \to e_j$, we conclude that $\widehat{j} = j$, thereby completing the proof.

B Further experimental results

This section provides the complete numerical results underlying the experimental analyses in Section 4, together with additional results under larger NFE budgets. Figure 9 compares the original LangFlow schedule with the adaptive schedule.

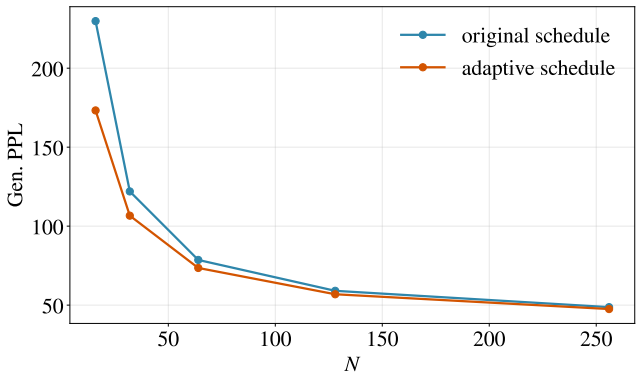


Figure 9: Effect of the sampling schedule on Gen. PPL. We compare the original schedule by LangFlow with the proposed adaptive schedule $t_i = (i + 0.5)/N$, $i = 0, \ldots, N-1$, across different steps N.

B.1 Detailed results for quality-diversity control

Tables 3 and 4 report the complete numerical results for the three sampling techniques under NFE budgets of 64 and 128, respectively. These results underlie the NFE= 64 trade-off curves in Figure 4 and extend the same comparison to NFE= 128. Across both budgets, increasing the corresponding control parameter lowers Gen. PPL at the cost of entropy, while the time-adaptive variants generally retain more entropy at comparable Gen. PPL. This consistent behavior confirms the benefit of concentrating guidance or iterative refinement near the data endpoint.

B.2 Detailed results for combinations of sampling techniques

Tables 5 and 6 provide the complete sweeps for jointly varying iterative self-conditioning refinement K_{iscr} and self-conditioning guidance w_{scg} under NFE budgets of 64 and 128, respectively. Consistent with Figure 5, a sufficiently large K_{iscr} combined with an appropriate w_{scg} shifts the trade-off frontier toward lower Gen. PPL at comparable entropy. Table 7 further reports the three-way combinations that include unconditional guidance. Varying w_{ug} mainly moves the operating point along a similar frontier, providing only a marginal additional improvement after the other two techniques have been combined.

B.3 Detailed results for guidance allocation

Tables 8 and 9 report the complete sweeps for Configurations A and B, respectively, under both NFE budgets. Tables 10 and 11 then collect representative operating points from all three configurations for NFE= 64 and 128. The numerical results support the trends in Figure 8: Configuration A reaches the

Table 3: Gen. PPL-entropy trade-offs produced by the three sampling techniques under a fixed NFE budget of 64. We compare each standard variant with its time-adaptive counterpart. For the time-adaptive variants, "Nominal value" denotes the control parameter before time-dependent scaling.

Sampling technique	**Standard**			**Time-adaptive**		
	Value	Gen. PPL	Entropy	Nominal value	Gen. PPL	Entropy
Iterative self-conditioning refinement	1	105.0006	5.5373	10	98.3887	5.5373
	2	85.4298	5.5139	15	90.3650	5.5303
	3	77.6263	5.5053	20	85.2866	5.5228
	5	73.3118	5.5027	40	78.6051	5.5176
Self-conditioning guidance	1	102.9869	5.5387	20	93.5983	5.5349
	2	80.8420	5.4981	40	73.6271	5.4962
	4	58.4980	5.4310	60	62.2094	5.4630
	6	47.3374	5.3790	80	54.6371	5.4349
	8	40.5905	5.3357	100	49.5442	5.4133
Unconditional guidance	0.01	91.9529	5.5084	0.25	91.6968	5.5106
	0.02	82.2282	5.4774	0.50	82.2702	5.4838
	0.04	66.9820	5.4171	1.00	67.0967	5.4312
	0.06	54.8009	5.3565	1.50	55.5643	5.3817
	0.08	45.8002	5.2953	2.00	47.0382	5.3343

Table 4: Gen. PPL-entropy trade-offs produced by the three sampling techniques under a fixed NFE budget of 128. We compare each standard variant with its time-adaptive counterpart. For the time-adaptive variants, "Nominal value" denotes the control parameter before time-dependent scaling.

Sampling technique	**Standard**			**Time-adaptive**		
	Value	Gen. PPL	Entropy	Nominal value	Gen. PPL	Entropy
Iterative self-conditioning refinement	1	93.3518	5.5078	10	89.2618	5.5052
	2	73.9209	5.4694	15	80.2898	5.4941
	3	64.3410	5.4497	20	74.6253	5.4828
	5	56.5077	5.4300	40	64.9262	5.4636
Self-conditioning guidance	1	93.3518	5.5078	20	84.4239	5.5039
	2	72.7807	5.4623	40	65.6629	5.4601
	4	51.5329	5.3858	60	55.0511	5.4230
	6	41.0542	5.3279	80	48.2367	5.3934
	8	35.0187	5.2822	100	43.8744	5.3719
Unconditional guidance	0.01	81.4945	5.4708	0.25	82.5130	5.4779
	0.02	72.4452	5.4362	0.50	73.6181	5.4501
	0.04	57.1142	5.3643	1.00	59.6448	5.3943
	0.06	46.1258	5.2910	1.50	49.2387	5.3419
	0.08	38.1393	5.2205	2.00	41.3981	5.2915

Table 5: Joint effect of iterative self-conditioning refinement and self-conditioning guidance under NFE = 64.

		Self-conditioning guidance strength w_{scg}					
Refinement parameter K_{iscr}	Metric	10	15	20	25	30	40
25	Gen. PPL	96.5033	76.8682	64.2796	55.8742	51.1444	45.0416
	Entropy	5.5532	5.5299	5.5109	5.4922	5.4794	5.4549
50	Gen. PPL	97.3557	69.4861	53.9569	45.1607	40.1334	35.691
	Entropy	5.5654	5.5341	5.505	5.4796	5.4592	5.4306
100	Gen. PPL	118.1948	74.3155	52.0762	41.6154	36.5026	32.5312
	Entropy	5.5909	5.5579	5.5237	5.4914	5.4634	5.4274

Table 6: Joint effect of iterative self-conditioning refinement and self-conditioning guidance under NFE = 128.

		Self-conditioning guidance strength w_{scg}					
Refinement parameter K_{iscr}	Metric	10	15	20	25	30	40
50	Gen. PPL	79.8379	57.2238	44.7869	37.8605	33.7153	29.8325
	Entropy	5.5191	5.4771	5.443	5.4145	5.3898	5.3587
100	Gen. PPL	85.1575	54.8722	39.9296	32.6275	28.7678	25.5304
	Entropy	5.5417	5.4906	5.4453	5.4049	5.3741	5.3379
200	Gen. PPL	103.3736	59.3922	40.3115	32.0478	27.9034	24.925
	Entropy	5.5743	5.5226	5.4712	5.4286	5.3937	5.3448

Table 7: Joint effect of unconditional guidance, iterative self-conditioning refinement, and self-conditioning guidance under NFE= 64.

			Self-conditioning guidance strength w_{scg}					
w_{ug}	K_{iscr}	Metric	10	15	20	25	30	40
0.0	50	Gen. PPL	97.3557	69.4861	53.9569	45.1607	40.1334	35.6910
		Entropy	5.5654	5.5341	5.5050	5.4796	5.4592	5.4306
	100	Gen. PPL	118.1948	74.3155	52.0762	41.6154	36.5026	32.5312
		Entropy	5.5909	5.5579	5.5237	5.4914	5.4634	5.4274
0.25	50	Gen. PPL	88.4084	64.1003	50.0962	42.6205	37.9129	33.7868
		Entropy	5.5428	5.5135	5.4867	5.4638	5.4432	5.4155
	100	Gen. PPL	108.3142	69.4401	49.1897	40.0159	34.9445	31.2325
		Entropy	5.5693	5.5411	5.5081	5.4796	5.4520	5.4161
0.5	50	Gen. PPL	80.6113	59.0871	46.8084	39.8520	35.8997	32.1812
		Entropy	5.5206	5.4935	5.4682	5.4451	5.4274	5.3999
	100	Gen. PPL	100.4747	64.9446	46.4663	37.8119	33.4694	30.0821
		Entropy	5.5507	5.5237	5.4933	5.4643	5.4401	5.4074

low-entropy, low-Gen. PPL regime, Configuration B provides strong intermediate operating points, and Configuration C preserves the greatest entropy. In overlapping regions, a larger refinement parameter K_{iscr} generally yields a more favorable trade-off, confirming that both the refinement count and its temporal allocation are important.

Table 8: Results for Configuration A that uses a constant refinement count K and self-conditioning guidance strength w_{scg}.

			Self-conditioning guidance strength w_{scg}				
NFE	K	Metric	1.5	2	2.5	3	3.5
64	4	Gen. PPL	45.7076	32.4898	26.1870	22.6047	20.7017
		Entropy	5.4124	5.3412	5.2835	5.2272	5.1831
	6	Gen. PPL	42.8333	29.9203	24.1211	20.8427	19.2514
		Entropy	5.4107	5.3332	5.2610	5.1853	5.1118
	8	Gen. PPL	39.4945	26.9691	21.1209	18.1815	16.7697
		Entropy	5.3929	5.3014	5.1909	5.0784	4.9710
128	4	Gen. PPL	36.0510	25.1186	19.9984	17.1249	15.3786
		Entropy	5.3282	5.2313	5.1521	5.0747	5.0131
	6	Gen. PPL	31.0520	21.3595	17.2039	14.6103	13.3592
		Entropy	5.3032	5.1938	5.0999	4.9871	4.8959
	8	Gen. PPL	28.6775	19.4933	15.2157	12.9933	11.5411
		Entropy	5.2882	5.1598	5.0118	4.8618	4.6964

Table 9: Results for Configuration B, in which both the refinement count and self-conditioning guidance strength are scaled by $1/(1+\sqrt{\sigma_{t_i}/\alpha_{t_i}})$.

			Self-conditioning guidance strength w_{scg}			
NFE	K_{iscr}	Metric	6	8	10	15
64	10	Gen. PPL	54.4313	43.7583	38.1201	32.4360
		Entropy	5.4732	5.4399	5.4151	5.3710
	15	Gen. PPL	48.0552	37.5346	32.1775	27.5590
		Entropy	5.4593	5.4161	5.3818	5.3216
	20	Gen. PPL	44.6766	34.0579	29.1159	25.1861
		Entropy	5.4545	5.4040	5.3662	5.2994
	25	Gen. PPL	42.7540	32.1617	27.5400	24.4351
		Entropy	5.4520	5.3987	5.3569	5.2904
128	10	Gen. PPL	45.4814	36.4717	31.7101	26.5305
		Entropy	5.4142	5.3748	5.3485	5.2930
	20	Gen. PPL	35.2302	26.7910	22.5727	18.6685
		Entropy	5.3741	5.3149	5.2624	5.1597
	30	Gen. PPL	31.6256	23.6132	20.0410	17.2126
		Entropy	5.3633	5.2928	5.2321	5.1147
	40	Gen. PPL	30.7218	22.6761	19.3316	17.0313
		Entropy	5.3640	5.2884	5.2264	5.1022

Table 10: Detailed Gen. PPL and entropy results for Configurations A–C under a fixed NFE budget of 64. Within each configuration, the columns vary the nominal self-conditioning guidance strength w_{scg}.

Configuration	Metric	w_{scg}										
A		1.5	1.75	2	2.25	2.5	2.75	3	3.25	3.5		
	Gen. PPL	39.49	31.67	26.97	23.30	21.12	19.58	18.18	17.45	16.77		
	Entropy	5.393	5.348	5.301	5.243	5.191	5.144	5.078	5.020	4.971		
B		6	7		8	9	10		12	15		
	Gen. PPL	42.75	36.23		32.16	29.36	27.54		25.37	24.44		
	Entropy	5.452	5.425		5.399	5.379	5.357		5.328	5.290		
C		20	22	24	26	28	30	32	34	36	38	40
	Gen. PPL	52.08	47.04	43.07	40.28	38.09	36.50	35.10	34.17	33.45	32.83	32.53
	Entropy	5.524	5.511	5.495	5.484	5.473	5.463	5.456	5.448	5.441	5.434	5.427

Table 11: Detailed Gen. PPL and entropy results for Configurations A–C under a fixed NFE budget of 128. Within each configuration, the columns vary the nominal self-conditioning guidance strength w_{scg}.

Configuration	Metric	w_{scg}										
A		1.5	1.75	2	2.25	2.5	2.75	3	3.25	3.5		
	Gen. PPL	28.68	23.10	19.49	16.99	15.22	14.02	12.99	12.23	11.54		
	Entropy	5.288	5.222	5.160	5.092	5.012	4.948	4.862	4.789	4.696		
B		6	7		8	9	10		12	15		
	Gen. PPL	30.72	25.81		22.68	20.87	19.33		17.64	17.03		
	Entropy	5.364	5.325		5.288	5.261	5.226		5.168	5.102		
C		20	22	24	26	28	30	32	34	36	38	40
	Gen. PPL	40.31	36.23	33.32	30.92	29.21	27.90	26.94	26.22	25.52	25.00	24.93
	Entropy	5.471	5.454	5.438	5.422	5.405	5.394	5.382	5.370	5.361	5.353	5.345

B.4 Qualitative Samples

We present qualitative generation samples from ConvergeFlow under NFE=64. All samplesare generated with a fixed sequence length of 1024 tokens.

ConvergeFlow

NFE: 64; **Gen. PPL:** 33.09; **Entropy:** 5.44

So if capitalism is to be subordinate to the proletariat (from which it must be neither independent, independent, or direct friend), then the state must do its part and let it make no concessions to what it wants, or force it to keep its power. Indeed, as is often the case with major bourgeois parties, the bourgeoisie is more interested in the actual fate of its role than in other parties.<|endoftext|>Yes, I would doubt that this (which has never happened), because Disney was about making an existing non-contained series with one-row characters in the 1930s or early 1940s. Even by this point, I was quite confident that the studio had to submit a schedule (as several months later) of two sold-out Disney-only episodes (and that, today if you haven't made a Disney-only series, you've made it elsewhere). But it's also quite probable now that there are at least one three episodes. It's hard to know what this would amount, but at least right now, it would seem that it would be a stretch to think we could make a lot season (or, for whatever reason, I'd suggest starting at least sooner).
Secondly, what about the current phase of our strategy is the need to work on the groundbreaking books made up to children at Disney. I know Ursula Puig was best known for this idea, and I suppose it's safe to say (with his knowledge) that most of our ongoing work is currently based on works that don't seem familiar enough to tell a live-action story about a man and two orphaned children in a world spanning as 100 or so years. As far as I know (e.g., the comic books), no longer are just 17 children's books in the comic books (such as Alice for Wonderland, King's Child, Aladdin, the Lion Z, and Tooty Duck: Edge of Time) that have appeared over the past 200 years alone. (And there are 17 of these made up more than a hundred years ago.) That supplemental programming has begun to play.
In that respect, I've just offered some detail of one fairly important strategy I would have come waiting to worry about:
Rather than make 12 completely incompatible films with an iconic female lead, our challenge was to make a realistic show, featuring a female lead character. In this area, we could focus on hundreds, or thousands, of a male/female difference, and then we could do it multiple times each year. But we had to pay a $10 fee (at the Disney amusement park, in New York) for every episode of our show that went through a deal with Turner Entertainment (a major partner for the ABC network, now owned by the BBC). Along the way, we went three more years on that goal and built the first Disney network to feature a female lead in every episode. Sailor Moon, Nickelodeon, Snow White, and the ABC's Pirates of the Sea have all been designed with Turner Brothers (with ABC obtaining a fee each) in less than two years.
Today, our vision is very broad. A major part of the mission that we've been working toward over the course of the 20 years now is: to make 12 completely films with iconic male characters, and have stand-alone characters with really female monsters and really female characters. And those people are all really talented! We don't really know how progressive we would actually be in taking that initiative, but it certainly makes sense. We wanted to be the first (and, certainly, most lucky, not necessarily the first) that we'd make "way 12" of a one-week movie with a female lead in the 1960s. The one other thing that really sucks, however, is that in the 1950s, we still had both the restriction (and expectations) of a really female character. We realized that the narratives conferred on women in the '60s and '70s were unfair, and didn't work. And then there just seemed to be no way to repurpose a family-friendly show that only had a stand-alone character (such as Professor Norris or Cat in the Jungle or Braisey or Winnie Florid reappearing in the 1930s), or even a stand-alone character such as Bugs Bunny, or even having to have any sort of female cast since the 1950s. (There's only a that one now as it stands by now.)

Once the proper process of gender-level work is completed (and then on more heavily done), then I think we can have the confidence that there are many other shows we can focus on being produced by women in the media, too. But we'll ask for commitment until we can see that we absolutely need to focus on female characters everywhere. And of course, any sexism within our departments.
Finally, let's start by asking what is the necessary plan in these 20 years to try to make a completely consistent entertainment with a cartoon female first? This is what Walt Disney Animation and other

ConvergeFlow

NFE: 64; **Gen. PPL:** 33.17; **Entropy:** 5.44

Thousands of women have taken steps to prevent and stop such substance abuse. This is the first federal legislation in the country, changing the constitutional rights of women. Activists and women of all administration should be able to have the courage to speak up on this issue," said Ayesem Housem, executive director for UNARAD in Washington, in a statement. "This legislation makes rape and violence a permanent target for our justice system."
Miranda Moore, president and co-founder of the League on Women, State and Girls, said in a release: "This bill is about who cares about fighting sexual assault and what it may mean for women of color."
The drug drug conventions was passed in 1998, many states which have implemented drug laws now follow their federal practice. The District of Columbia is the first state that has banned marijuana, and it is the first time federal laws has passed. President Donald Trump has begun moving with a number of other directives that are attempt to cut down the use of marijuana, reduce the state's opioid consumption and expand the criminal justice system.
During the campaign, President Donald Trump signed the Law Justice Act of 2017, which would decriminalize marijuana for use. The policy was announced in part by Attorney Attorney General Eric Holder, who partnered with federal prosecutors to end the crackdown on non-violent offenders last year. The policy is expected to take effect in July 2018.<|endoftext|>As seems perfectly reasonable, few small businesses make lots of money to run a business. Unfortunately, our customers still get "great money" from our websites and business projects. If we can put it all by ourselves, let's see how we can make it.
In this post I looked at how much money we used on Starbucks customers and how we missed it. In part 1, you'll also see how much we wasted money.
Of course, I plan to update this post today. The video will give you a glimpse of how we worked (i.e., how we failed in terms of great innovation and marketing). I will also explain some of the mistakes we made and break out a brief outline of how our financial failures happened.
There are two main things I've talked about — improved customer experience and better customer success. In this post, I describe a more efficient strategy (i.e., we get financial resources from just one customer while building out only one business). Using these strategies, we can help accomplish a lot more smoothly.
Let's start with the PayPal problem. We had a long and painful battle with PayPal and PayPal Inc, costing us $6 million. The truth is, we kept spending money by using PayPal and all the water in the air was blowing up our financial rate of nowhere. Let us look at some of our (fun) mistakes:
Spending
We spent about $33 million building an business. As the name goes, the company had 22.1 percent of revenue. We borrowed money from other customers, built our first eCommerce store and rented out our unique business with PayPal. We also kept track of our business by spending money.
Spending was consistently inefficient — i.e., 49 percent of revenue came directly from our website and PayPal. We made an increase of 24.6 percent. (While you don't have to spend all that much money, here's how to do that.)
Let's start with four methods:

Dation of debt. This means that money was actually charged for us. We charged $4,500 a month — the average cost of (bighbage). Our charges were big-ranging and the average interest rate was 79 percent, an increase of 17.5 percent.
Investment of cost. This method didn't change much at all. We engaged our e-commerce team to various teams in the development of our e-commerce store so that we could gain a steady flow of revenue while building out new businesses that serve customers. We spent 12 percent of our revenue in our first "4-point" trial. This increased the efficiency of our revenue model and gave us an increase of 1.8 percent. While our revenue was 1.6 or more optimistic than the 4-point trial, we still reduced productivity by 18.5 percent.
Our loans. The fourth method was quite inefficient. We didn't use debt at all. Even rather, we borrowed income from just one customer. (Note the loans from the screen apps via app.) You don't have to make that test to see how it's worked.
Borrow from one customer. When we got financial money from another customer, we earned 23.8 percent.
This last method is a little different. Unlike our first method, it doesn't really count as passive income or debt. Instead, we just take extra income from

ConvergeFlow

NFE: 64; **Gen. PPL:** 33.17; **Entropy:** 5.44

Primarily, it also applies to snakes, kangaroo, trophy-tailed dogs, and so on, while it applies to cows, dogs and chickens, raccours, leopards, wolves, and so on, and many other phenomena. Humanism allows for many different kinds of negative acts toward humans. Without these negative acts, non-humanists have no reason to believe that their own actions are coming from humans, even if they share their true beliefs, values, and so forth. Eugenicists, in stark contrast, have no reason to take anyone's actions, even if they have society's moral authority behind them. It is the responsibilities of humanists to decide what is right for humans and what is best for society in these decisions.
Humanism is a statement of nature's nature, a critique of nature's culture, history, and so forth. It is essential for human people to consistently understand this one way or another, and obviously we will have incredible difficulty in hearing them. Vegetarianism does not view nature as a complicated activity that seeks to set the boundaries to our laws as we approach them toward one another. But its system of veganism is particularly harmful because it can intentionally distort nature's boundaries and it is demeaning. Bulled animal abuse is so ubiquitous as it has little official use among animals. The failure to apply it, instead of treating it as a weapon, gives a death eye to ethical weakness. This is why the literature showing that animal abuse in animals is widespread, well-documented, and widespread.
Animalism is especially dangerous because it is often used as an excuse to stir a criticism of nature's boundaries, but almost always by literally calling for animals that can be captured and domesticated. This incapacity to ignore it, instead of treating it as a deterrent, puts a temporary end to ethical weakness. Whether humans aim to "strictly exploit the natural environment" or "endrictly eat livestock," nonhumanists often claim that their laws are incompatible with their moral duties against humans and animals we use, and even make the case that society ought to work with them just to make their lives better.
The key to improving the ethical treatment of victims of animal animal abuse is the lack of empathy by individuals who care nothing about their behavior as they understand their attitudes toward it. They cannot even talk about the boundaries or the acts that justify one's moral duty towards animals. The most precise example used in this article is when lions and tigers were allegedly shot down an amusement park in Zimbabwe.
Reason and Peace: Nonhumanists

Phumanists are not necessarily bad people. They often celebrate a state's constitutional right to a man to keep arms, but not a state's constitutional right to bear arms. For some, the loss of such a powerful fellow is an extremely useful ethical choice given that it encaps in just a few of the same core principles and principles that historically entangled them with the wide range of moral issues. Nonhumanists generally seek to accept the "crime" of aggression with the right to be taken away from others, even though they distinguish between humans and all humans, rather than by acting on the application of moral relativism. Even when using the words "gender heteronamally," an abdication of a bogeyman or patriarchal society, or a woman's legal right to have children, or the ability to have children and them, these are thoroughly debunked.

Nonhumanists often seek to understand the "naturalization" of all humans, but how they seek to justify it in order to acknowledge the absolutism of nature. For example, they object to the term "theetine Earth," and talk about keeping rabbits, gorillas or chimpanzees living in open spaces and engaging in other forms of transhumanization. Nonhumanists do not accept the purely contestal moral choice of a human being, unless they harm others with their own choices, even when pushing for creating laws or regulations that clearly violate our laws.

It is important to understand that physical violence—even intentional and reprehensible conduct—is central to our ethical practice. All acts of actions are religiously defined by these principles, and are adjudicated with a quick response to everyone who wants them. Nonhumanists areently act on animals without their direct legal responsibility, and rely exclusively on animals to try to destroy human beings lest they flaunt their rights.

Legal Considerations

You can find an authoritative list of dozens of cases that are central to our ethical practice. However, these are generally the laws that should be discussed in practice.

Human beings are well-organized in performing their moral duties towards us and should not be subject to any use by those with moral authority or responsibility. If you believe in truly equal liberty and equality, accept the fact that those with good moral responsibility are being only

ConvergeFlow

NFE: 64; **Gen. PPL:** 33.17; **Entropy:** 5.44

She looked at her very beautiful young sister, who looked up, but well-liked feeling as though she was making a living for her.

Kazuki came over to the restaurant without question, and walked into it with a sigh of relief. "There's a serious problem, mother, don't trust me if anyone asks that you can help."

"Thanks the best!" Murayumi said. Misuki closed his eyes and glanced at him, defiantly. "Now, try not to distract them, are you going to stay home, or are you planning next tomorrow to take care of your daughter altogether?"

Murayumi looked down at her feet to see if that would do anything, but that was what Misuki expected of her. She leaned on being worried about government interference, which seemed to be being the root cause of the situation. Misuki didn't know when to talk or reassure her. His partner gave her a nod up next, but the blond's two went alingter as she turned around, noticing that Murayumi felt like she had more control. Maybe she would, but she started to calm down, and then started to talk to bed.

"Looks better here. I'll talk to you next week as we should. Any questions, I suppose." Kazuki groaned and glapped slightly.

"That was not working out." Murayumi let out a whimper. "Anyway, I'll get you back here soon. I'm willing to do something else for you if you can help me do my very best! Dugging." Misuki glanced over at Murayumi's young sister, angry that she wasn't pustering. He tumbled to his feet and walked around, kneeling in search of food and swinging around on a jet driven midboard. Murayumi may have wanted him to, but he didn't know. The young Yang seemed even more concerned, not with her

new clothes, but her new body and anything else that could hold on. Why did it be so hot?
"What do you feel concerned about?" Murayumi began, blinking clearly. "I need a couple more minutes if I can discuss it with you."
Shinuki's response was interesting. He was had much to think about, but it was perhaps not by the best part. She was calm and one of the most muscleless people that Murayumi had ever had to offer, and he was certainly glad she would have been able to carry on with him.
"That's it." Murayumi said as he walked around back to the table and waited for some leftover pancakes to prepare. "Get it yourself if you can."
"Mother." She was ready, lifting her head open and peering her hands back into the pancakes. The menu was okay as well, but the opportunity to see Misuki get on herself was relatively easy. The thinness of the legs didn't help by much at that time, especially for her. They were the tinnest parts of Murayumi's body and covered them perfectly, so she turned them around to make a skie. Though she didn't have the same level of difficulty as caused by the abundance of clothes she was wearing, she instead appeared to be extremely active and motivated to get dressed.
So, what happened next? The initial conversation was seemingly awkward at first, but it did offer an interesting perspective. Misuki already had a good understanding of Murayumi's condition, but she was still not in fact very concerned of her health. She was definitely very concerned about the weight that she had put on her body, so it was actually by any means worrying. Misuki turned his eyes back to her mother and wondered how things had held up. "Was that one kind of meat? Were you ever worried about it?" Murayumi asked. "I wasn't so much concerned, either. Had it not gone far enough?"
"It has been a very busy week, I suppose." Murayumi said, moving her mother's hands back to the table and sliding her face up at the table. "It's been fun, though, mother. Like I said before I ate most of your meals, but when you woke up, I always took them away right away."
"Alright." Misuki said, turning back to the table and hunching her hands, while still trying to force himself away from herself. "Alright, mother. I'll talk with you as I plan a meal within the week."
Asakura shook her head and shook her head. "Again, great." She said. "Thank you so much. I hope you can join me for some wonderful times. You're the one that I hope to eventually meet."
"Haha." Murayumi said with a hint of relief. "Well, you're the one I would use the most. Later on, I can't serve you. Don't let