SELF-CONDITIONED FLOW MAP LANGUAGE MODELS VIA FIXED-POINT FLOWS

Anonymous authors
Paper under double-blind review

ABSTRACT

Self-conditioning is a core technique that enhances continuous flow-based language models, where the model learns to denoise generated text by conditioning on its own denoising estimate. While empirically successful, its performance improvements are poorly understood. Moreover, there is growing interest in the use of few-step generators based on flow maps, for which how to leverage self-conditioning is unclear. Here, we show that flow language models with self-conditioning perform a fixed-point iteration that improves generation through iterative refinement. We use this viewpoint to formulate **fixed-point flows**, a two-dimensional class of self-conditioned flows, where the first dimension represents the flow process and the second represents the fixed-point iteration. We show that fixed-point flows define valid flow maps, and show that they can be distilled from self-conditioned flow models by compressing both fixed-point iterations and the flow process, the former with fixed-point distillation and the latter with flow map distillation. Our resulting flow map language model, FMLM*, outperforms state-of-the-art self-conditioned models and few-step models in one- and few-step generation on OpenWebText.

1 INTRODUCTION

Language models (LMs) based on continuous flows have recently emerged as a promising paradigm for non-autoregressive text generation (Lee et al., 2026; Chemseddine et al., 2026; Deschenaux & Gulcehre, 2026). By learning to denoise tokens in a continuous space, these models perform parallel iterative generation through a deterministic evolution driven by a velocity field. Importantly, such processes define a unique flow map, the solution operator that directly transports noise to data in as few as one function evaluation. This advantage has sparked recent breakthroughs on distillation of flow language models into flow map language models that are capable of generating text in one to few inference steps (Lee et al., 2026; Roos et al., 2026; Potaptchik et al., 2026).

Recently, the performance of flow language models has improved through the incorporation of a self-conditioning mechanism (Chen et al., 2022). Unlike in usual flow training, self-conditioned flow models learn to denoise an input by conditioning on its own denoising prediction. Then, during generation, they perform denoising at each flow timestep by conditioning on the previously denoised outcome. Self-conditioning has empirically proven very effective (Dieleman et al., 2022; Strudel et al., 2022), and has been widely adopted in the latest state-of-the-art flow language models (Chen et al., 2026; Hu et al., 2026; Batzolis et al., 2026; Meshchaninov et al., 2026; Yang et al., 2026). Yet, despite its success, why self-conditioning leads to improvements remains poorly understood. Furthermore, it is unclear how to distill flows into flow maps under self-conditioning, as it introduces additional dependencies across generation timesteps. For further context, we discuss related work in Appendix A.

In this work, we introduce **fixed-point flows**, a mathematical framework for self-conditioned flows and their associated flow maps (Figure 1). Our key insight is that self-conditioning in flow language models can be viewed as a fixed-point iteration. With this insight, we characterize a two-dimensional class of self-conditioned flows, where the first dimension represents the original flow, and the second represents fixed-point iterations, which empirically improve generation quality. We use this view to distill self-conditioned flow language models into flow maps by compressing both fixed-point iterations and flow, achieving state-of-the-art one- and few-step language modeling.

Our main contributions are:

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

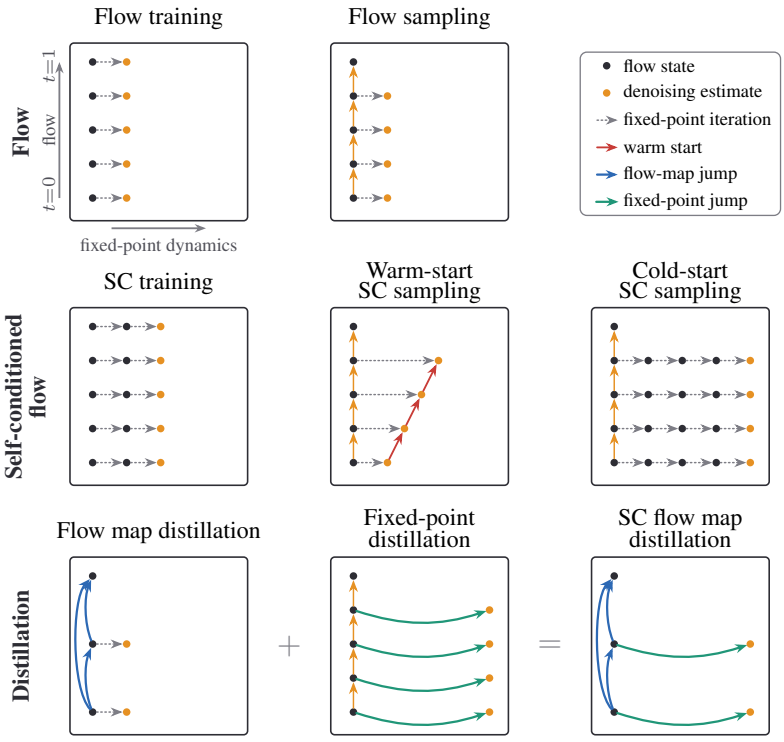


Figure 1: **Overview.** We show that flow language models with self-conditioning perform a fixed-point iteration that improves generation quality. We leverage this insight to formulate fixed-point flows, a class of self-conditioned flows that run fixed-point iterations at each flow timestep. Fixed-point flows yield valid flow maps, which we learn by compressing both the flow and the fixed-point iterations.

- **Fixed-point view of self-conditioning.** We show that flow language models with self-conditioning implicitly learn a fixed-point iteration. We theoretically establish convergence of this iteration under a contractivity assumption.
- **Fixed-point flows and their flow maps.** We introduce fixed-point flows, a formalization of self-conditioned flows. We show that fixed-point flows define a valid flow map, which can be learned by compressing both fixed-point iterations and flow with the respective few-step distillation objectives.
- **Empirical results.** We empirically show that refining predictions toward the learned fixed point improves generation quality. We distill self-conditioned flow language models into flow map language models, achieving state-of-the-art one- and few-step generation on OpenWebText. We also show that it is possible to distill self-conditioned models into self-conditioning-free ones without degradation.

2 Preliminary

Flow language models. Let V be a vocabulary of tokens, and denote text data with length L by $\mathbf{y} = (\mathbf{y}^l)_{l=1}^L \in V^L$. The goal of language modeling is to learn the data distribution $p(\mathbf{y})$ so that we can draw a new text sample efficiently. Flow language models choose a continuous embedding $\mathbf{y} \mapsto \mathbf{x} \in \mathbb{R}^{L\times d}$ and a decoder $\mathbf{x} \mapsto \mathbf{y}$, and model the induced distribution $p(\mathbf{x})$ on the embedding space, generating $\hat{\mathbf{x}} \sim p(\mathbf{x})$ and then rounding into discrete language through $\hat{\mathbf{x}} \mapsto \hat{\mathbf{y}}$.

To model the now-continuous data distribution $p(\mathbf{x})$, flow language models use flow matching over a stochastic interpolant (Lipman et al., 2022; Albergo et al., 2025). They specify a probability path $p_t(\mathbf{x}_t)$ over $t \in [0, 1]$ as the density of an interpolant $I_t := (1-t)\mathbf{x}_0 + t\mathbf{x}_1$ between noise $\mathbf{x}_0 \sim p_0$ and data $\mathbf{x}_1 \sim p_1$. The path p_t admits a deterministic evolution equation that can be used to draw a sample $\mathbf{x}_t \sim p_t$ at inference time:

$$\dot{\mathbf{x}}_t = b_t(\mathbf{x}_t), \quad b_t(\mathbf{x}) = \mathbb{E}[\mathbf{x}_1 - \mathbf{x}_0 \mid I_t = \mathbf{x}], \quad \mathbf{x}_0 \sim p_0, \quad t \in [0, 1], \tag{1}$$

where b_t is the velocity field of the flow. If a model $\hat{b}$ of the velocity is available, a sample $\hat{\mathbf{x}}_1 \sim p_1$ can be approximately drawn by numerically integrating (1) across a time grid $0 = t_0 < ... < t_N = 1$. A simple choice for the integration is the forward Euler scheme:

$$\hat{\mathbf{x}}_{t_{i+1}} = \hat{\mathbf{x}}_{t_i} + (t_{i+1} - t_i)\hat{b}_{t_i}(\hat{\mathbf{x}}_{t_i}), \quad \hat{\mathbf{x}}_0 \sim p_0. \tag{2}$$

Instead of learning the velocity directly, it is more common in language modeling to learn the denoiser D_t (Lee et al., 2026; Hu et al., 2026) which outputs the mean of clean data, and from which one can recover the velocity (Albergo et al., 2025; Li & He, 2026):

$$D_t(\mathbf{x}) \coloneqq \mathbb{E}[\mathbf{x}_1 \mid I_t = \mathbf{x}], \quad b_t(\mathbf{x}) = \frac{D_t(\mathbf{x}) - \mathbf{x}}{1 - t}. \tag{3}$$

The ideal denoiser D can be learned by solving a regression problem $D = \operatorname{argmin}_{\hat{D}} \mathcal{L}(\hat{D})$ that predicts clean data by minimizing the following loss:

$$\mathcal{L}(\hat{D}) \coloneqq \int_0^1 \mathbb{E}|\hat{D}_t(I_t) - \mathbf{x}_1|^2 \mathrm{d}t. \tag{4}$$

While we have used square loss above, our discussions and results readily transfer to the cross-entropy setting of Dieleman et al. (2022); Eijkelboom et al. (2024); Lee et al. (2026). With a learned denoiser $\hat{D}$, generation can be done by turning it into an estimation of the velocity $\hat{b}$ through (3) and then leveraging the forward Euler scheme (2).

Flow map language models. A key advantage of continuous flow for language modeling is that its deterministic generative process (1) driven by the velocity b_t defines a unique flow map $X_{s,t} : \mathbb{R}^{L \times d} \to \mathbb{R}^{L \times d}$, the solution operator that transports a sample directly $\mathbf{x}_t = X_{s,t}(\mathbf{x}_s)$ between any timesteps $(s, t) \in [0, 1]^2$. It satisfies the following integral equation:

$$X_{s,t}(\mathbf{x}_s) = \mathbf{x}_s + \int_s^t b_\tau(\mathbf{x}_\tau)\, \mathrm{d}\tau. \tag{5}$$

If available, a flow map allows for efficient few-step generation by sequentially evaluating $\hat{\mathbf{x}}_{t_{i+1}} = X_{t_i,t_{i+1}}(\hat{\mathbf{x}}_{t_i})$ upon any grid $0 = t_0 < ... < t_N = 1$, even one-step generation via $\hat{\mathbf{x}}_1 = X_{0,1}(\hat{\mathbf{x}}_0)$.

Recent works have demonstrated that flow map language models can be learned by distilling the flow velocity learned by flow language models (Lee et al., 2026; Roos et al., 2026; Potaptchik et al., 2026). These methods leverage a set of mathematical relations between the flow velocity and the flow map (Boffi et al., 2026; Lee et al., 2026), which are derivable from the integral equation in (5), and learn the flow map with respect to a given flow velocity to satisfy these relations.

3 Theoretical Framework

In this section, we formalize self-conditioning and propose that it learns a fixed-point iteration which refines the model's prediction. We then develop fixed-point flows, a framework for self-conditioned flows, along with the associated flow maps for few-step generation. Finally, we introduce distillation methods for turning self-conditioned flow language models into few-step language models.

3.1 Self-conditioned flow language models

Self-conditioning is a technique introduced in Chen et al. (2022) where the denoiser learns to predict conditioned on its own denoising estimates. In these approaches, the model of the denoiser (3) takes an additional conditioning, $\hat{D}_t(\mathbf{x}, \mathbf{z})$, which effectively reduces to a usual denoiser model when $\mathbf{z} = \mathbf{0}$. The denoiser is trained using the following loss, where μ is a choice of training-time distribution of $\mathbf{z}$:

$$\mathcal{L}_\mu(\hat{D}) \coloneqq \int_0^1 \mathbb{E}_{\mathbf{x}_0,\mathbf{x}_1} \mathbb{E}_\mu |\hat{D}_t(I_t, \mathbf{z}) - \mathbf{x}_1|^2 \mathrm{d}t. \tag{6}$$

A common choice of μ is the mixture of delta peaks at zero $\mathbf{z} = \mathbf{0}$ and the model's own denoising prediction $\mathbf{z} = \mathsf{sg}(\hat{D}_t(I_t, \mathbf{0}))$ where sg is the stop-gradient operator. This leads to the following:

$$\mathcal{L}_\mu(\hat{D}) = \frac{1}{2}\int_0^1 \mathbb{E}|\hat{D}_t(I_t, \mathbf{0}) - \mathbf{x}_1|^2 \mathrm{d}t + \frac{1}{2}\int_0^1 \mathbb{E}|\hat{D}_t(I_t, \mathsf{sg}(\hat{D}_t(I_t, \mathbf{0}))) - \mathbf{x}_1|^2 \mathrm{d}t, \tag{7}$$

where the first term is the usual denoising loss (4), whereas the second term is a self-conditioned loss which encourages correcting an initial estimate by the model.

In general, the conditioning variable $\mathbf{z}$ in the training loss (6) does not change the learning target of the denoiser. This is because the Bayes-optimal prediction of $\mathbf{x}_1$ given an interpolant I_t is specified by the ideal denoiser, $D_t(I_t)$ (3), regardless of $\mathbf{z}$, assuming $\mathbf{z}$ does not leak additional information about $\mathbf{x}_1$ beyond what is available from I_t, as in (7). Thus, the objective trains the model to map every on-distribution conditioning $\mathbf{z}$ to the ideal denoising target. We provide a formal proof:

Proposition 3.1. *Assume that $\mathbf{z} \perp \mathbf{x}_1 \mid I_t$ for almost every t, and assume all second moments are finite. Then every unrestricted population minimizer $\bar{D}$ of (6) satisfies*

$$\bar{D}_t(\mathbf{x}, \mathbf{z}) = D_t(\mathbf{x}) \tag{8}$$

almost surely under the law of $(I_t, \mathbf{z})$, for almost every t.

A proof is in Appendix B.1. In practice, training would not attain the minimum loss and the model would have dependence on $\mathbf{z}$. With the self-conditioned loss (7), this can encourage a self-correcting behavior that produces an initial prediction and then pulls it closer to the Bayes-optimal prediction.

The generative process under self-conditioning is similar to that of (1), evolving a sample $\hat{\mathbf{x}}_{t_i}$ across a flow-time grid $0 = t_0 < ... < t_N = 1$. Yet, the denoising estimate is also maintained and updated as $\hat{\mathbf{z}}_{t_i}$ to condition and bootstrap the denoiser, altering the numerical integration scheme (2) as follows:

$$\begin{aligned} \hat{\mathbf{x}}_{t_{i+1}} &= \hat{\mathbf{x}}_{t_i} + (t_{i+1} - t_i)\hat{b}_{t_i}, \quad \hat{b}_{t_i} = \frac{\hat{\mathbf{z}}_{t_{i+1}} - \hat{\mathbf{x}}_{t_i}}{1 - t_i}, \quad \hat{\mathbf{x}}_0 \sim p_0, \\ \hat{\mathbf{z}}_{t_{i+1}} &= \hat{D}_{t_i}(\hat{\mathbf{x}}_{t_i}, \hat{\mathbf{z}}_{t_i}), \quad \hat{\mathbf{z}}_0 = \mathbf{0}. \end{aligned} \tag{9}$$

Intuitively, this creates an information sharing across flow timesteps via $\hat{\mathbf{z}}$ on top of the flow state $\hat{\mathbf{x}}$.

Although self-conditioning empirically leads to substantial improvements of flow language models (Chen et al., 2026; Hu et al., 2026; Batzolis et al., 2026; Meshchaninov et al., 2026; Yang et al., 2026), understanding of its mechanism is limited (Chen et al., 2022; Dieleman et al., 2022; Shabalin et al., 2025; 2026). Furthermore, since the self-conditioned velocity (9) is also coupled with the updates of the denoising estimates, it is unclear how to use it for few-step distillation, which relies on a flow map defined by integrating the velocity field as in (5).

3.2 Self-conditioning induces a fixed-point iteration

We present our key intuition that a self-conditioned denoiser, trained with (7), refines its own predictions through a fixed-point iteration. Formally, a self-conditioned denoiser $\hat{D}(\mathbf{x}, \mathbf{z})$, given a choice of flow timestep t and flow state $\mathbf{x}$, can be used to define the following iteration $\mathbf{z}^j$ for $j = 0, 1, ...$ that starts at some initialization $\mathbf{z}^0$:

$$\mathbf{z}^{j+1} = \hat{D}_t(\mathbf{x}, \mathbf{z}^j). \tag{10}$$

As the denoiser learns to correct its own prediction (7), we posit that it characterizes a self-correcting process, eventually reaching a fixed point $\mathbf{z}^\star = \hat{D}_t(\mathbf{x}, \mathbf{z}^\star)$ (Gulrajani & Hashimoto, 2023). We formalize this intuition below and show that the fixed point coincides with the Bayes-optimal prediction in the ideal case.

First, we recall the notion of a contraction, which suffices for convergence to a unique fixed point.

Definition 3.2 (Contraction). A function $f : \mathbb{R}^d \to \mathbb{R}^d$ is a contraction on a closed set $O \subseteq \mathbb{R}^d$ with factor $0 \leq \eta < 1$ if it satisfies $f(O) \subseteq O$ and $|f(\mathbf{z}) - f(\mathbf{z}')| \leq \eta|\mathbf{z} - \mathbf{z}'|$ for every $\mathbf{z}, \mathbf{z}' \in O$.

By Proposition 3.1, a perfect denoiser $\bar{D}_t(\mathbf{x}, \cdot)$ with minimum loss maps directly to $D_t(\mathbf{x})$, making it a contraction with $\eta = 0$. When the denoiser $\hat{D}$ is learned, it is not always guaranteed to be contractive. Yet, in Appendix B.2, we show that optimizing self-conditioned loss leads to approximate contractivity. To simplify our analysis, we shall henceforth assume that the learned denoiser is contractive. This is in line with common assumptions in looped models (Romano et al., 2017; Ryu et al., 2019; Bai et al., 2019; Fung et al., 2022; Movahedi et al., 2026) and supported by our results that self-conditioned

models ELF (Hu et al., 2026) and LangFlow (Chen et al., 2026) approximate fixed-point iterations. We show:

Proposition 3.3. *Fix t, $\mathbf{x}$, and $0 \le \eta < 1$. Assume that $\hat{D}_t(\mathbf{x}, \cdot)$ is a contraction on a nonempty closed set $O \subseteq \mathbb{R}^{L\times d}$ with factor η. Then, the iteration $\mathbf{z}^{j+1} = \hat{D}_t(\mathbf{x}, \mathbf{z}^j)$ (10) from any $\mathbf{z}^0 \in O$ satisfies the following properties:*

(i) It converges exponentially to a unique fixed point $\mathbf{z}^\star \in O$,

(ii) Its denoising error at any iteration $j \ge 1$ is bounded as

$$|\mathbf{z}^j - D_t(\mathbf{x})| \le |\hat{D}_t(\mathbf{x}, \mathbf{z}^0) - D_t(\mathbf{x})| + \frac{\eta - \eta^j}{1-\eta}|\mathbf{z}^1 - \mathbf{z}^0|. \tag{11}$$

(iii) Its denoising error at the fixed point is bounded as

$$|\mathbf{z}^\star - D_t(\mathbf{x})| \le |\hat{D}_t(\mathbf{x}, \mathbf{z}^0) - D_t(\mathbf{x})| + \frac{\eta}{1-\eta}|\mathbf{z}^1 - \mathbf{z}^0|, \tag{12}$$

A proof is in Appendix B.3. The result shows that the initial prediction error, together with the magnitude of the first iteration, controls the denoising error bound up to the fixed point. It further guarantees convergence to this fixed point under contractivity.

Having formulated the relationship between a self-conditioned denoiser $\hat{D}$ and its fixed points for each pair of t and $\mathbf{x}$, it is convenient to think of a function that directly predicts the fixed point. We call such a function the **fixed-point denoiser** $D^\star$, defined as follows for every $t \in [0,1], \mathbf{x} \in \mathbb{R}^{L\times d}$:

$$D_t^\star(\mathbf{x}) \coloneqq \mathbf{z}^\star = \hat{D}_t(\mathbf{x}, \mathbf{z}^\star). \tag{13}$$

where $\mathbf{z}^\star$ is the unique fixed point of $\hat{D}_t(\mathbf{x}, \cdot)$. The fixed-point denoiser is a self-conditioning-free denoiser that behaves the same as the given self-conditioned denoiser run until convergence. Its ideal target under Proposition 3.1 is the Bayes-optimal prediction $D_t(\mathbf{x})$, in which case we have $D^\star = D$.

3.3 Fixed-point flows

In Section 3.1, we have seen that self-conditioned flow appears to involve two states: the flow state and the self-conditioning state. In Section 3.2, we have established that a self-conditioned denoiser induces a fixed-point iteration. Here, we propose to replace the self-conditioning state by its fixed point. Once this is done, the velocity depends only on the flow state, so it becomes an ordinary flow. We call this a **fixed-point flow**, and prove its fundamental properties as follows.

Proposition 3.4. *Assume that, for every $t \in [0,1)$ and $\mathbf{x}$, the self-conditioned denoiser $\hat{D}_t(\mathbf{x}, \cdot)$ has a unique fixed point $D_t^\star(\mathbf{x}) = \hat{D}_t(\mathbf{x}, D_t^\star(\mathbf{x}))$. Define the **fixed-point velocity***

$$b_t^\star(\mathbf{x}) \coloneqq \frac{D_t^\star(\mathbf{x}) - \mathbf{x}}{1-t}, \quad t \in [0,1). \tag{14}$$

Then $b_t^\star$ depends only on t and $\mathbf{x}$, and therefore the dynamics

$$\dot{\mathbf{x}}_t = b_t^\star(\mathbf{x}_t) \tag{15}$$

define an ordinary time-dependent ODE. Furthermore, when the fixed-point denoiser is Bayes optimal, $D^\star = D$, the fixed-point velocity recovers the true velocity, $b^\star = b$.

A proof is in Appendix B.4. Equation (14) says that at each flow time, the self-conditioned denoiser $\hat{D}$ runs fixed-point iteration (10) until convergence, and the fixed point modulates the flow via (14). This suggests an Euler scheme for sampling $\hat{\mathbf{x}}_1 \sim p_1$ over a flow-time grid $0 = t_0 < \ldots < t_N = 1$:

$$\hat{\mathbf{x}}_{t_{i+1}} = \hat{\mathbf{x}}_{t_i} + (t_{i+1} - t_i)\frac{\hat{\mathbf{z}}_{t_i}^\star - \hat{\mathbf{x}}_{t_i}}{1-t_i}, \tag{16}$$

where each $\hat{\mathbf{z}}_{t_i}^\star$ is obtained by solving an inner fixed-point iteration from an initialization $\hat{\mathbf{z}}_{t_i}^0$:

$$\hat{\mathbf{z}}_{t_i}^{j+1} = \hat{D}_{t_i}(\hat{\mathbf{x}}_{t_i}, \hat{\mathbf{z}}_{t_i}^j). \tag{17}$$

Under the contractivity assumption, the fixed-point iteration converges to a unique fixed point regardless of its initialization within the contraction domain. This suggests a simple approach: always initialize with zero, $\hat{\mathbf{z}}^0_{t_i} = \mathbf{0}$, which we call **cold-start sampling** of a fixed-point flow. On the other hand, choosing the right initialization can improve the efficiency of finding the fixed point. For instance, if the fixed points are correlated locally in t, the fixed-point estimate of the previous flow timestep can provide a good initialization, $\hat{\mathbf{z}}^0_{t_i} = \hat{\mathbf{z}}^\star_{t_{i-1}}$. We call this **warm-start sampling**. The potential benefit of a better initialization is formally quantified under contractivity in Appendix B.5.

From this initialization perspective, we can view conventional self-conditioned generation in (9) as warm-start sampling that approximates fixed points with a single iteration, $\hat{\mathbf{z}}^\star_{t_i} \approx \hat{D}_{t_i}(\hat{\mathbf{x}}_{t_i}, \hat{\mathbf{z}}^\star_{t_{i-1}})$. This implies that the coupling of flow state and self-conditioning state in (9) is not a defining property of self-conditioned flows, as it is merely a byproduct of warm starts, a sampling heuristic. We empirically find that cold-start sampling with sufficient fixed-point iterations can replace warm starts in ELF (Hu et al., 2026) and LangFlow (Chen et al., 2026).

Taking this idea further, we find that it is possible to take a self-conditioned flow model and turn it into a self-conditioning-free model by distilling the self-conditioned denoiser $\hat{D}$ into the fixed-point denoiser $D^\star$. This can be done, for instance, via the following fixed-point distillation loss:

$$\mathcal{L}(D^\star) = \int_0^1 \mathbb{E}|D^\star_t(I_t) - \mathbf{z}^\star|^2 \,\mathrm{d}t, \quad \mathbf{z}^\star = \hat{D}_t(I_t, \mathbf{z}^\star), \tag{18}$$

where the target $\mathbf{z}^\star$ is estimated by iterating $\mathbf{z}^{j+1} = \hat{D}_t(I_t, \mathbf{z}^j)$ from $\mathbf{z}^0 = \mathbf{0}$. Any other fixed-point distillation method can be used instead of (18), such as consistency distillation (Lin et al., 2026). The resulting fixed-point velocity $b^\star$ depends only on the current time and flow state, so its generation can use the usual Euler scheme (2).

3.4 Fixed-point flow maps

In Section 3.3, we have formalized fixed-point flows, which express self-conditioned flows via ordinary velocity fields. We leverage this insight to define their associated flow maps, which allows us to design few-step distillation methods for self-conditioned flow language models.

Proposition 3.5. *On any interval where the fixed-point flow ODE (15) has unique solutions, the **fixed-point flow map**, its solution operator $X^\star_{s,t}(\mathbf{x}_s) = \mathbf{x}_t$, satisfies*

$$X^\star_{s,t}(\mathbf{x}) = \mathbf{x} + \int_s^t b^\star_\tau(\mathbf{x}_\tau)\,\mathrm{d}\tau. \tag{19}$$

Moreover, for $0 \le s \le u \le t < 1$,

$$X^\star_{s,t} = X^\star_{u,t} \circ X^\star_{s,u}. \tag{20}$$

A proof is in Appendix B.6. The result shows that $X^\star$ is a valid flow map and admits conditions such as (20) that can be leveraged to learn it through distillation from the velocity $b^\star_t$. In order to learn the flow map, we parameterize it with a **two-time denoiser** $\delta_{s,t}$ defined as follows (Lee et al., 2026; Lu et al., 2026):

$$\delta_{s,t}(\mathbf{x}) \coloneqq \mathbf{x} + (1-s)v_{s,t}(\mathbf{x}), \tag{21}$$

where $v_{s,t}$ is the average velocity from s to t, defined as follows (Boffi et al., 2026):

$$v_{s,t}(\mathbf{x}) \coloneqq \frac{X^\star_{s,t}(\mathbf{x}) - \mathbf{x}}{t-s}, \quad v_{t,t}(\mathbf{x}) \coloneqq b^\star_t(\mathbf{x}). \tag{22}$$

Analogous to the relationship between the single-time denoiser and velocity, $D^\star_t(\mathbf{x}) = \mathbf{x} + (1-t)b^\star_t(\mathbf{x})$, the two-time denoiser can be viewed as a single denoising step to the clean data domain. Its outputs are known to lie in a low-dimensional manifold (Lee et al., 2026; Lu et al., 2026), making learning easier. We now show:

Proposition 3.6. *The two-time denoiser $\delta_{s,t}$ satisfies the following properties:*

(i) For $0 \le s < t < 1$, the flow map is recovered from $\delta_{s,t}$ by

$$X^\star_{s,t}(\mathbf{x}) = \frac{1-t}{1-s}\mathbf{x} + \frac{t-s}{1-s}\delta_{s,t}(\mathbf{x}). \tag{23}$$

(ii) On the diagonal,

$$\delta_{t,t}(\mathbf{x}) = D^\star_t(\mathbf{x}). \tag{24}$$

(iii) For $0 \le s < u < t < 1$, the semigroup condition (20) is equivalent to

$$\delta_{s,t}(\mathbf{x}) = \gamma\delta_{s,u}(\mathbf{x}) + (1-\gamma)\delta_{u,t}(X^\star_{s,u}(\mathbf{x})), \tag{25}$$

where $\gamma = \frac{(1-t)(u-s)}{(1-u)(t-s)} \in [0,1]$.

A proof is in Appendix B.7. The result shows that learning the two-time denoiser yields the flow map, and that the two-time denoiser matches the fixed-point denoiser $D^\star$ (13) on the diagonal $s = t$, while satisfying a self-consistency criterion off the diagonal. These conditions lead to a learning objective:

Proposition 3.7. *Consider the population objective*

$$\mathcal{L}(\delta) = \mathbb{E}_{s,u,t,\mathbf{x}}|\delta_{s,t}(\mathbf{x}) - \mathsf{sg}(\mathcal{T}(\delta)_{s,u,t}(\mathbf{x}))|^2 + \mathbb{E}_{t,\mathbf{x}}|\delta_{t,t}(\mathbf{x}) - D^\star_t(\mathbf{x})|^2 \tag{26}$$

where

$$\mathcal{T}(\delta)_{s,u,t}(\mathbf{x}) := \gamma\delta_{s,u}(\mathbf{x}) + (1-\gamma)\delta_{u,t}\big(X^\star_{s,u}(\mathbf{x})\big). \tag{27}$$

Then the true two-time denoiser has zero loss. Conversely, any zero-loss solution satisfies the diagonal (24) and semigroup (25) conditions almost surely under the training distribution.

A proof is given in Appendix B.8. The first loss term enforces the semigroup condition, and the second loss term anchors the diagonal to the fixed point $D^\star_t(I_t) = \mathbf{z}^\star = \hat{D}_t(I_t, \mathbf{z}^\star)$. Evaluating the latter requires fixed points $\mathbf{z}^\star$, which we may make available in one of two ways: an offline route that first distills the fixed-point denoiser $D^\star$ as a separate model via (18), or an online route that performs two compressions in (18) and (26) jointly, estimating $\mathbf{z}^\star$ with a few iterations of $\mathbf{z}^{j+1} = \hat{D}_t(I_t, \mathbf{z}^j)$ from $\mathbf{z}^0 = \mathbf{0}$. The online approach ultimately achieves $\delta_{t,t} \approx D^\star_t$ without training a separate model.

4 Experiments

Through our experiments, we aim to answer the following key questions:

Q1. Do self-conditioned flow language models learn fixed-point iterations?

Q2. Does self-conditioned generation provide a better initialization for fixed-point iteration?

Q3. Can we train self-conditioning-free flows competitive with self-conditioned ones?

Q4. Can we train a flow map language model that leverages self-conditioning?

Setup. We conduct all experiments on the OpenWebText dataset (Gokaslan et al., 2019), a standard corpus for language modeling, with a sequence length of 1024 tokens. We assess generation quality with generative perplexity (gPPL) (Dieleman et al., 2022; Sahoo et al., 2025) measured with pretrained GPT-2 Large (Radford et al., 2019), together with average per-sample unigram entropy. We regard a model as strong only when its generations attain low gPPL while remaining close to the data entropy of 5.44 nats (Lee et al., 2026). We use 256 samples for analysis (Section 4.1) and 1024 for distillation (Section 4.2). See Appendix C for more experimental details.

4.1 Analyzing self-conditioned flow language models

To answer whether self-conditioned denoisers induce fixed-point iterations (**Q1**) and whether the gains of self-conditioned sampling come from improved initialization of fixed-point iterations (**Q2**), we study ELF (Hu et al., 2026) and LangFlow (Chen et al., 2026) as representatives of self-conditioned flow language models on learned embeddings and one-hot token embeddings, respectively. We draw

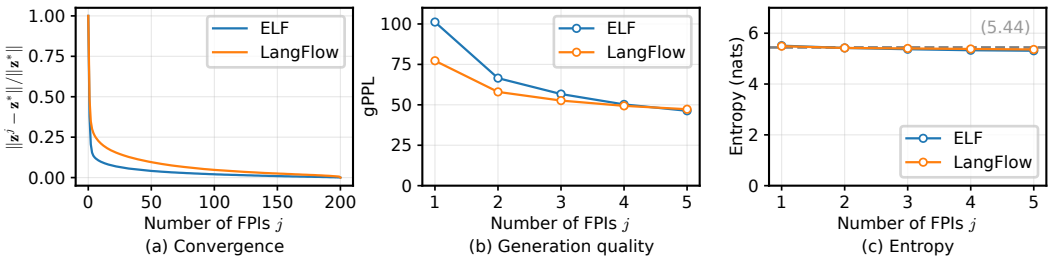


Figure 2: Convergence and generation quality across fixed-point iterations for ELF and LangFlow. (a) Relative distance to the fixed point. (b) gPPL and (c) entropy with 32 sampling steps for ELF and 64 for LangFlow. The dashed gray line indicates data entropy (5.44 nats).

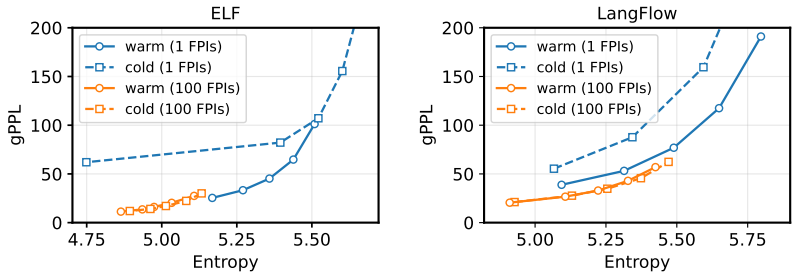


Figure 3: Warm-start and cold-start sampling with 1 and 100 fixed-point iterations.

gPPL-entropy frontier curves (Pynadath et al., 2025) by sweeping sampling parameter γ for ELF and softmax temperature for LangFlow.

Self-conditioning induces a fixed-point iteration (Q1). We first ask whether self-conditioned denoisers learn fixed-point iteration (10). Following Lin et al. (2026), we track relative distance to the fixed point, $|\mathbf{z}^j - \mathbf{z}^\star|/|\mathbf{z}^\star|$, which equals one at $j = 0$ when cold-start sampling is used ($\mathbf{z}^0 = \mathbf{0}$). To approximate $\mathbf{z}^\star$, we run 200 damped Picard iterations (Lauriere, 2021) with a fixed damping parameter 0.3 (see Appendix C.1 for details). As shown in Figure 2(a), the relative distance decays across fixed-point iterations j for both models. This supports that the models have approximately learned a fixed point iteration as posited in Section 3.2.

We now ask whether taking more steps toward the fixed point improves generation quality. As illustrated in Figure 2(b,c), gPPL decreases as the number of fixed-point iterations increases on both ELF and LangFlow, whereas entropy remains stable near the data entropy, showing that taking more steps toward the fixed point produces a better denoiser for generation. In Appendix E.1, we further show qualitatively that iterative refinement improves the consistency and coherence of the model's predictions.

Self-conditioned generation provides a better initialization for fixed-point iteration (Q2). In Section 3.3, we viewed conventional generation of self-conditioned flows as a warm-start sampling scheme that attempts to use a better initialization for the fixed-point iteration. Accordingly, we conjectured that cold-start sampling would reach a similar performance if sufficient iterations are used as it would eventually find the fixed point regardless of initialization. To validate this, we draw the gPPL–entropy frontier of each sampling scheme in Figure 3. When we use a single fixed-point iteration, warm-start outperforms cold-start sampling, suggesting that warm-start provides an effective initialization for fixed-point iteration. With 100 iterations, however, warm- and cold-start attain the same frontier, indicating that the choice of initialization has a negligible effect on generation once enough fixed-point iterations are run.

4.2 Distilling self-conditioning into fixed-point flow maps

To test self-conditioning-free distillation (**Q3**) and self-conditioned flow map distillation (**Q4**), we train ELF (Hu et al., 2026) as a teacher model, replacing its encoder with GPT-2 Large (Radford et al., 2019) for tokenizer-matched comparison with baselines.

Table 1: Comparison with few-step language models on OpenWebText. Offline FMLM* outperforms all baselines that preserve data-level entropy (5.44 nats).

Step	Metric	Duo		MDLM		FMLM	DFM		FMLM*	
		DCD	Di4C	SDTT	Di4C		PSD	ESD	Offline	Online
1	gPPL ↓	5743.29	370.51	1260.86	1298.80	168.30	180.29	5.33	**113.91**	117.89
	entropy	6.02	3.92	5.26	5.29	5.17	4.91	0.26	5.37	5.39
2	gPPL ↓	891.16	210.22	877.22	758.23	133.29	152.83	108.91	**93.66**	98.96
	entropy	5.41	4.63	5.34	5.35	5.25	5.03	5.15	5.46	5.47
4	gPPL ↓	250.86	154.67	339.73	239.27	111.31	122.32	77.08	**73.86**	80.26
	entropy	5.37	4.85	5.38	5.40	5.26	5.10	5.27	5.35	5.44

Self-conditioning is removable (Q3). We find that, given a self-conditioned denoiser $\hat{D}$ (ELF in our case), we can learn its associated fixed-point denoiser $D^\star$ (13) through fixed-point distillation, which yields fixed-point velocity (14), a self-conditioning-free model. Herein, we consider CDEQ (Lin et al., 2026), an existing fixed-point distillation method. Using the method, we distill the ELF teacher into a self-conditioning-free model **ELF***, with implementation details provided in Appendix C. We compare 32-step generation performance of ELF* against 32-step generation of the teacher with gPPL-entropy frontier curves. As in Figure 4, ELF* matches the teacher's frontier, confirming that self-conditioning is removable via fixed-point distillation without degrading quality.

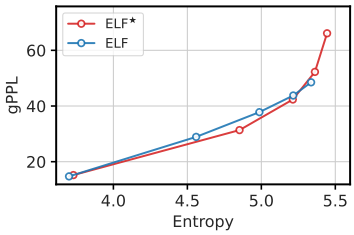


Figure 4: Comparison between ELF* and its teacher model ELF.

Self-conditioning is distillable into a flow map (Q4). Finally, we ask whether self-conditioning can be leveraged to train a few-step flow map. To this end, we distill the self-conditioned ELF teacher into a self-conditioned flow map language model **FMLM***, parameterized by the two-time denoiser $\delta_{s,t}$ (21). Following the offline route proposed in Section 3.4, we take the fixed-point denoiser ELF* as teacher and learn its associated fixed-point flow map with the semigroup distillation objective (26). As illustrated in Figure 5, FMLM* with 8 generation steps approaches the 32-step frontier of the self-conditioned teacher ELF.

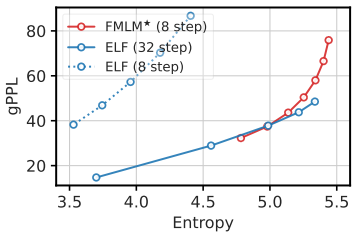


Figure 5: Comparison between FMLM* and teacher model ELF.

We compare FMLM* with discrete diffusion baselines Duo with DCD (Sahoo et al., 2025), MDLM with SDTT (Deschenaux & Gulcehre, 2024), both with Di4C (Hayakawa et al., 2024), and continuous flow map models FMLM (Lee et al., 2026) and DFM (Potaptchik et al., 2026). In Table 1, offline FMLM* achieves the best gPPL among baselines preserving data-level entropy under matched sampling budgets. Online distillation with 8 FPIs avoids training a separate fixed-point denoiser, achieving competitive quality while reducing training time from 82 to 23 hours. Additional FPI results and generation samples appear in Appendices D and E.2.

5 Conclusion

We demonstrated that self-conditioned flow language models implicitly learn a fixed-point iteration that refines their predictions. Under contractivity, the iteration converges to a fixed point; empirically, additional iterations improve generation quality. Based on this viewpoint, we proposed fixed-point flows, a two-dimensional class of self-conditioned flows, and interpreted conventional self-conditioned sampling as a single-step warm-started fixed-point iteration. Building on this framework, we distilled both fixed-point iterations and flow to obtain FMLM*, which achieves state-of-the-art one- and few-step generation on OpenWebText under matched sampling budgets. While these results demonstrate the effectiveness of our approach, our flow map learning requires a separate teacher, motivating further research into self-distillation that jointly learns the self-conditioned, fixed-point, and two-time denoisers from data supervision. Our evaluation is also limited to text, leaving the applicability of the framework to other modalities an open question.

AI USE STATEMENT

We used generative AI tools to assist with polishing the text, literature search, code implementation and debugging, and checking mathematical derivations. The authors take full responsibility for the final manuscript and accompanying artifacts, including all AI-assisted content.

ETHICS STATEMENT

We do not anticipate ethical concerns specific to this work beyond those generally associated with language generation models.

REPRODUCIBILITY STATEMENT

Proofs of the theoretical results are provided in Appendix B, and implementation and experimental details are documented in Appendix C. We plan to release our code to facilitate reproduction of the experimental results.

REFERENCES

Michael Albergo, Nicholas M Boffi, and Eric Vanden-Eijnden. Stochastic interpolants: A unifying framework for flows and diffusions. *Journal of Machine Learning Research*, 26(209):1–80, 2025.

Shaojie Bai, J Zico Kolter, and Vladlen Koltun. Deep equilibrium models. *Advances in neural information processing systems*, 32, 2019.

Xingjian Bai and Luke Melas-Kyriazi. Fixed point diffusion models. In *CVPR*, 2024.

Georgios Batzolis, Mark Girolami, and Luca Ambrogioni. Towards closing the autoregressive gap in language modeling via entropy-gated continuous bitstream diffusion. *arXiv preprint arXiv:2605.07013*, 2026.

Nicholas Boffi, Michael Albergo, and Eric Vanden-Eijnden. How to build a consistency model: Learning flow maps via self-distillation. *Advances in Neural Information Processing Systems*, 38: 33346–33382, 2026.

Jannis Chemseddine, Gregor Kornhardt, and Gabriele Steidl. Spherical flows for sampling categorical data. *arXiv preprint arXiv:2605.05629*, 2026.

Ting Chen, Ruixiang Zhang, and Geoffrey Hinton. Analog bits: Generating discrete data using diffusion models with self-conditioning. *arXiv preprint arXiv:2208.04202*, 2022.

Yuxin Chen, Chumeng Liang, Hangke Sui, Ruihan Guo, Chaoran Cheng, Jiaxuan You, and Ge Liu. Langflow: Continuous diffusion rivals discrete in language modeling. *arXiv preprint arXiv:2604.11748*, 2026.

Justin Deschenaux and Caglar Gulcehre. Beyond autoregression: Fast llms via self-distillation through time. *arXiv preprint arXiv:2410.21035*, 2024.

Justin Deschenaux and Caglar Gulcehre. Language modeling with hyperspherical flows. *arXiv preprint arXiv:2605.11125*, 2026.

Sander Dieleman, Laurent Sartran, Arman Roshannai, Nikolay Savinov, Yaroslav Ganin, Pierre H Richemond, Arnaud Doucet, Robin Strudel, Chris Dyer, Conor Durkan, et al. Continuous diffusion for categorical data. *arXiv preprint arXiv:2211.15089*, 2022.

Floor Eijkelboom, Grigory Bartosh, Christian A Naesseth, Max Welling, and Jan-Willem van de Meent. Variational flow matching for graph generation. *Advances in Neural Information Processing Systems*, 37:11735–11764, 2024.

Samy Wu Fung, Howard Heaton, Qiuwei Li, Daniel McKenzie, Stanley Osher, and Wotao Yin. Jfb: Jacobian-free backpropagation for implicit networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pp. 6648–6656, 2022.

Aaron Gokaslan, Vanya Cohen, Ellie Pavlick, and Stefanie Tellex. Openwebtext corpus. http://Skylion007.github.io/OpenWebTextCorpus, 2019.

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

Satoshi Hayakawa, Yuhta Takida, Masaaki Imaizumi, Hiromi Wakaki, and Yuki Mitsufuji. Distillation of discrete diffusion through dimensional correlations. *arXiv preprint arXiv:2410.08709*, 2024.

Keya Hu, Linlu Qiu, Yiyang Lu, Hanhong Zhao, Tianhong Li, Yoon Kim, Jacob Andreas, and Kaiming He. Elf: Embedded language flows. *arXiv preprint arXiv:2605.10938*, 2026.

Dongjun Kim, Chieh-Hsin Lai, WeiHsiang Liao, Naoki Murata, Yuhta Takida, Toshimitsu Uesaka, Yutong He, Yuki Mitsufuji, and Stefano Ermon. Consistency trajectory models: Learning probability flow ode trajectory of diffusion. In *ICLR*, 2024.

Mathieu Lauriere. Numerical methods for mean field games and mean field type control. *arXiv preprint arXiv:2106.06231*, 2021.

Chanhyuk Lee, Jaehoon Yoo, Manan Agarwal, Sheel Shah, Jerry Huang, Aditi Raghunathan, Seunghoon Hong, Nicholas M Boffi, and Jinwoo Kim. Flow map language models: One-step language modeling via continuous denoising. *arXiv preprint arXiv:2602.16813*, 2026.

Tianhong Li and Kaiming He. Back to basics: Let denoising generative models denoise. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 36115–36125, 2026.

Junchao Lin, Zenan Ling, Jingwen Xu, and Robert C Qiu. Consistency deep equilibrium models. *arXiv preprint arXiv:2602.03024*, 2026.

Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.

Yiyang Lu, Susie Lu, Qiao Sun, Hanhong Zhao, Zhicheng Jiang, Xianbang Wang, Tianhong Li, Zhengyang Geng, and Kaiming He. One-step latent-free image generation with pixel mean flows. *arXiv preprint arXiv:2601.22158*, 2026.

Viacheslav Meshchaninov, Alexander Shabalin, Egor Chimbulatov, Nikita Gushchin, Ilya Koziev, Alexander Korotin, and Dmitry Vetrov. How to train your latent diffusion language model jointly with the latent space. *arXiv preprint arXiv:2605.07933*, 2026.

Andrea Miele, Yiming Qin, Alba Carballo-Castro, Justin Deschenaux, and Pascal Frossard. Fixed-point masked generative modeling. *arXiv preprint arXiv:2605.31215*, 2026.

Sajad Movahedi, Vera Milovanović, Shlomo Libo Feigin, Alexander Theus, Thomas Hofmann, Valentina Boeva, T Konstantin Rusch, and Antonio Orvieto. Fixed-point reasoners: Stable and adaptive deep looped transformers. *arXiv preprint arXiv:2606.18206*, 2026.

Peter Potaptchik, Jason Yim, Adhi Saravanan, Peter Holderrieth, Eric Vanden-Eijnden, and Michael S Albergo. Discrete flow maps. *arXiv preprint arXiv:2604.09784*, 2026.

Patrick Pynadath, Jiaxin Shi, and Ruqi Zhang. Candi: Hybrid discrete-continuous diffusion models. *arXiv preprint arXiv:2510.22510*, 2025.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *JMLR*, 21(140):1–67, 2020.

Yaniv Romano, Michael Elad, and Peyman Milanfar. The little engine that could: Regularization by denoising (red). *SIAM journal on imaging sciences*, 10(4):1804–1844, 2017.

Daan Roos, Oscar Davis, Floor Eijkelboom, Michael Bronstein, Max Welling, Ismail Ilkan Ceylan, Luca Ambrogioni, and Jan-Willem van de Meent. Categorical flow maps. *arXiv preprint arXiv:2602.12233*, 2026.

Ernest Ryu, Jialin Liu, Sicheng Wang, Xiaohan Chen, Zhangyang Wang, and Wotao Yin. Plug-and-play methods provably converge with properly trained denoisers. In *International Conference on Machine Learning*, pp. 5546–5557. PMLR, 2019.

Amirmojtaba Sabour, Sanja Fidler, and Karsten Kreis. Align your flow: Scaling continuous-time flow map distillation. *NeurIPS*, 2025.

Subham Sekhar Sahoo, Justin Deschenaux, Aaron Gokaslan, Guanghan Wang, Justin Chiu, and Volodymyr Kuleshov. The diffusion duality. *arXiv preprint arXiv:2506.10892*, 2025.

Alexander Shabalin, Viacheslav Meshchaninov, Egor Chimbulatov, Vladislav Lapikov, Roman Kim, Grigory Bartosh, Dmitry Molchanov, Sergey Markov, and Dmitry Vetrov. Tencdm: Understanding the properties of the diffusion model in the space of language model encodings. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 25110–25118, 2025.

Alexander Shabalin, Simon Elistratov, Viacheslav Meshchaninov, Ildus Sadrtdinov, and Dmitry Vetrov. Why gaussian diffusion models fail on discrete data? *arXiv preprint arXiv:2604.02028*, 2026.

Robin Strudel, Corentin Tallec, Florent Altché, Yilun Du, Yaroslav Ganin, Arthur Mensch, Will Grathwohl, Nikolay Savinov, Sander Dieleman, Laurent Sifre, et al. Self-conditioned embedding diffusion for text generation. *arXiv preprint arXiv:2211.04236*, 2022.

Zhihan Yang, Wei Guo, Shuibai Zhang, Subham Sekhar Sahoo, Yongxin Chen, Arash Vahdat, Morteza Mardani, and John Thickstun. Continuous diffusion scales competitively with discrete diffusion for language. *arXiv preprint arXiv:2605.18530*, 2026.

A Related Work

Self-conditioning in continuous language models. Self-conditioning reuses denoising predictions as model inputs (Chen et al., 2022) and has been applied to continuous text diffusion (Strudel et al., 2022; Dieleman et al., 2022). Gulrajani & Hashimoto (2023) interpret the self-conditioned denoiser as a fixed point of prediction updates. We develop this perspective by analyzing convergence under contractivity and separating prediction refinement from flow evolution to define fixed-point flows and their flow maps.

Fixed-point computation in neural architectures. Fixed-point iterations enable neural networks to refine their representations repeatedly using shared parameters. Bai et al. (2019) define hidden representations as fixed points and compute them via root-finding, while Movahedi et al. (2026) use fixed-point convergence to adapt computation in looped transformers. For generative modeling, FPDM (Bai & Melas-Kyriazi, 2024) and FPMGM (Miele et al., 2026) incorporate such iterations into diffusion and masked generative denoisers to improve generation quality and computational efficiency. We use the fixed-point iteration perspective to formalize and analyze self-conditioning in flow language models. This analysis allows us to define fixed-point flows and their associated flow maps, enabling flow map distillation of self-conditioned models.

Flow map language models. Flow map language models build on the flow map framework of Boffi et al. (2026) to enable one- and few-step text generation. FMLM (Lee et al., 2026) formulates language generation as a continuous flow and distills its associated flow map. CFM (Roos et al., 2026) develops a simplex-constrained parameterization for categorical data and learns flow maps through self-distillation, including an endpoint-consistency objective. DFM (Potaptchik et al., 2026) further studies flow map training objectives adapted to the geometry of the probability simplex. Integrating self-conditioning into flow map learning remains challenging because the denoiser depends on an additional prediction state that evolves across sampling steps.

B Proofs and Theoretical Results

B.1 Proof of Proposition 3.1

Proposition 3.1. *Assume that $\mathbf{z} \perp \mathbf{x}_1 \mid I_t$ for almost every t, and assume all second moments are finite. Then every unrestricted population minimizer $\bar{D}$ of (6) satisfies*

$$\bar{D}_t(\mathbf{x}, \mathbf{z}) = D_t(\mathbf{x}) \tag{8}$$

almost surely under the law of $(I_t, \mathbf{z})$, for almost every t.

Proof. Fix t for which the conditional independence assumption holds, and write

$$D := D_t(I_t), \qquad U := \hat{D}_t(I_t, \mathbf{z}).$$

Then

$$\mathbf{x}_1 - U = (\mathbf{x}_1 - D) + (D - U).$$

Squaring and taking expectations gives

$$\mathbb{E}|\mathbf{x}_1 - U|^2 = \mathbb{E}|\mathbf{x}_1 - D|^2 + \mathbb{E}|U - D|^2 + 2\mathbb{E}\langle \mathbf{x}_1 - D, D - U\rangle.$$

The cross term vanishes. Indeed,

$$\mathbb{E}[\mathbf{x}_1 - D \mid I_t, \mathbf{z}] = \mathbb{E}[\mathbf{x}_1 \mid I_t, \mathbf{z}] - D_t(I_t).$$

By the conditional independence assumption,

$$\mathbb{E}[\mathbf{x}_1 \mid I_t, \mathbf{z}] = \mathbb{E}[\mathbf{x}_1 \mid I_t] = D_t(I_t).$$

Hence

$$\mathbb{E}[\mathbf{x}_1 - D \mid I_t, \mathbf{z}] = 0,$$

and therefore

$$\mathbb{E}\langle \mathbf{x}_1 - D, D - U\rangle = 0.$$

Thus

$$\mathbb{E}|\hat{D}_t(I_t, \mathbf{z}) - \mathbf{x}_1|^2 = \mathbb{E}|\mathbf{x}_1 - D_t(I_t)|^2 + \mathbb{E}|\hat{D}_t(I_t, \mathbf{z}) - D_t(I_t)|^2.$$

The first term does not depend on $\hat{D}_t$. The second term is minimized exactly when

$$\hat{D}_t(I_t, \mathbf{z}) = D_t(I_t)$$

almost surely under the law of $(I_t, \mathbf{z})$. Integrating over t gives the claim. □

B.2 Proof of approximate contractivity

We show that learning the self-conditioned objective leads to an approximate notion of contractivity.

Definition B.1 (A simplified self-conditioned regression.). Let $O \subseteq \mathbb{R}^d$ be a nonempty and closed set, let $Z \sim \mu$ with $\mu(O) = 1$, let $X \in \mathbb{R}^d$ be fixed, and let $f : \mathbb{R}^d \to \mathbb{R}^d$ be a continuous function. For some $0 < p < 1$, we define the following self-conditioned regression objective:

$$\mathcal{L}(f) \coloneqq p\mathbb{E}|f(Z) - X|^2 + (1-p)\mathbb{E}|f(f(Z)) - X|^2. \tag{28}$$

This is a simplification of our problem setting in Section 3.1. Taking $p = 1/2$, $f = \hat{D}_t(I_t, \cdot)$, and $X = D_t(\mathbf{x})$ (the Bayes-optimal prediction target; Proposition 3.1), for fixed $t \in [0, 1]$ and $\mathbf{x}$, we recover a practical learning scenario for self-conditioned language models similar to (7).

Let us fix

- a scale $w > 0$,
- a factor $0 \leq \eta < 1$,
- a tolerance $r > 0$,
- and a failure probability $0 < \varepsilon < 1$.

Let us assume:

$$\bar{B}(X, r) \subseteq O, \quad 2r \leq \eta w, \quad \mathcal{L}(f) \leq \min\{p, 1-p\}\varepsilon r^2. \tag{29}$$

Define

$$A \coloneqq \{\mathbf{z} \in O : |f(\mathbf{z}) - X| \leq r\}. \tag{30}$$

The following lemmas show that a sufficiently small loss yields a closed, high-probability set A that is contractive with factor η above the scale w, and approximately forward-invariant under the training distribution with leakage at most ε. Intuitively, the first term in the self-conditioned loss makes the set A high-probability under μ, whereas the second term makes $f(Z)$ remain in A for most $Z \sim \mu$.

Lemma B.2 (Closedness and approximate forward invariance). *A is a closed subset of $\mathbb{R}^d$. Furthermore, $f(A) \subseteq O$, and it holds that*

$$\mu(A \cap f^{-1}(A)) \geq 1 - \varepsilon. \tag{31}$$

In particular, $\mu(A) \geq 1 - \varepsilon$ and

$$\mu(\{\mathbf{z} \in A : f(\mathbf{z}) \notin A\}) \leq \varepsilon. \tag{32}$$

Proof. The closed Euclidean ball $\bar{B}(X, r)$ is closed. Since f is continuous,

$$f^{-1}(\bar{B}(X, r))$$

is closed. Therefore,

$$A = O \cap f^{-1}(\bar{B}(X, r))$$

is closed because O is closed.

For every $\mathbf{z} \in A$, we have, by definition,

$$|f(\mathbf{z}) - X| \leq r.$$

Hence

$$f(\mathbf{z}) \in \bar{B}(X, r).$$

By assumption (29),

$$f(A) \subseteq O.$$

Define the initial and self-conditioned errors

$$e_1(\mathbf{z}) \coloneqq |f(\mathbf{z}) - X|, \quad e_2(\mathbf{z}) \coloneqq |f(f(\mathbf{z})) - X|.$$

and define

$$G \coloneqq \{\mathbf{z} \in O : e_1(\mathbf{z}) \leq r \text{ and } e_2(\mathbf{z}) \leq r\}.$$

The condition $e_1(\mathbf{z}) \leq r$ is exactly $\mathbf{z} \in A$.

Moreover, as we have seen, $e_1(\mathbf{z}) \leq r$ implies $f(\mathbf{z}) \in O$. Therefore,

$$e_2(\mathbf{z}) \leq r$$

is equivalent to

$$f(\mathbf{z}) \in A.$$

Consequently,

$$G = A \cap f^{-1}(A).$$

On the complement G^c, at least one of the inequalities

$$e_1(\mathbf{z}) \leq r, \quad e_2(\mathbf{z}) \leq r$$

fails.

If $e_1(\mathbf{z}) > r$, then

$$pe_1(\mathbf{z})^2 > pr^2 \geq \min\{p, 1-p\}r^2.$$

If $e_2(\mathbf{z}) > r$, then

$$(1-p)e_2(\mathbf{z})^2 > (1-p)r^2 \geq \min\{p, 1-p\}r^2.$$

Thus, for every $\mathbf{z} \notin G$,

$$pe_1(\mathbf{z})^2 + (1-p)e_2(\mathbf{z})^2 \geq \min\{p, 1-p\}r^2.$$

Equivalently,

$$\mathbf{1}(\mathbf{z} \notin G) \leq \frac{pe_1(\mathbf{z})^2 + (1-p)e_2(\mathbf{z})^2}{\min\{p, 1-p\}r^2}.$$

Taking expectation with respect to $Z \sim \mu$,

$$\mu(G^c) \leq \frac{\mathbb{E}[pe_1(\mathbf{z})^2 + (1-p)e_2(\mathbf{z})^2]}{\min\{p, 1-p\}r^2} = \frac{\mathcal{L}(f)}{\min\{p, 1-p\}r^2}.$$

By assumption (29),

$$\mu(G^c) \leq \varepsilon.$$

Therefore,

$$\mu(G) \geq 1 - \varepsilon.$$

Since $G = A \cap f^{-1}(A)$,

$$\mu(A \cap f^{-1}(A)) \geq 1 - \varepsilon.$$

This implies $\mu(A) \geq 1 - \varepsilon$ because $A \cap f^{-1}(A) \subseteq A$.

It also implies

$$\mu(A \setminus f^{-1}(A)) \leq \varepsilon,$$

because the event

$$A \setminus f^{-1}(A) = \{\mathbf{z} \in A : f(\mathbf{z}) \notin A\}$$

is contained in G^c. □

Lemma B.3 (Approximate contractivity). *For every $\mathbf{z}, \mathbf{z}' \in A$ satisfying $|\mathbf{z} - \mathbf{z}'| \geq w$, one has*

$$|f(\mathbf{z}) - f(\mathbf{z}')| \leq \eta |\mathbf{z} - \mathbf{z}'|. \tag{33}$$

Proof. Take arbitrary $\mathbf{z}, \mathbf{z}' \in A$. By definition of A,

$$|f(\mathbf{z}) - X| \leq r, \quad |f(\mathbf{z}') - X| \leq r.$$

By the triangle inequality,

$$|f(\mathbf{z}) - f(\mathbf{z}')| \leq |f(\mathbf{z}) - X| + |f(\mathbf{z}') - X| \leq 2r.$$

By assumption,

$$2r \leq \eta w.$$

Therefore, whenever $|\mathbf{z} - \mathbf{z}'| \geq w$,

$$|f(\mathbf{z}) - f(\mathbf{z}')| \leq 2r \leq \eta w \leq \eta |\mathbf{z} - \mathbf{z}'|.$$

Hence

$$|f(\mathbf{z}) - f(\mathbf{z}')| \leq \eta |\mathbf{z} - \mathbf{z}'|$$

for every pair $\mathbf{z}, \mathbf{z}' \in A$ separated by at least w. □

B.3 Proof of Proposition 3.3

We first show a useful lemma.

Lemma B.4. *Let $f : \mathbb{R}^d \to \mathbb{R}^d$ be a contraction on a nonempty closed set $O \subseteq \mathbb{R}^d$ with factor $0 \leq \eta < 1$. Then, there is a unique fixed point $\mathbf{z}^\star \in O$, to which the iteration*

$$\mathbf{z}^{j+1} = f(\mathbf{z}^j)$$

from any $\mathbf{z}^0 \in O$ converges exponentially. Furthermore, the following holds for any finite $j \geq k \geq 0$:

$$|\mathbf{z}^j - \mathbf{z}^k| \leq \frac{\eta^k - \eta^j}{1 - \eta} |\mathbf{z}^1 - \mathbf{z}^0|.$$

Similarly, the following holds:

$$|\mathbf{z}^\star - \mathbf{z}^k| \leq \frac{\eta^k}{1 - \eta} |\mathbf{z}^1 - \mathbf{z}^0|.$$

Proof. Because O is a closed subset of $\mathbb{R}^d$, it is complete. The map f maps O into itself and is a contraction on O. Therefore, by the Banach fixed-point theorem, f has a unique fixed point $\mathbf{z}^\star \in O$, and the iterates converge to it.

A geometric convergence bound follows directly from the contraction property:

$$|\mathbf{z}^{j+1} - \mathbf{z}^\star| = |f(\mathbf{z}^j) - f(\mathbf{z}^\star)| \leq \eta |\mathbf{z}^j - \mathbf{z}^\star|.$$

Repeating this inequality gives

$$|\mathbf{z}^j - \mathbf{z}^\star| \leq \eta^j |\mathbf{z}^0 - \mathbf{z}^\star|. \tag{34}$$

For $j \geq k \geq 0$, telescoping gives

$$\mathbf{z}^j - \mathbf{z}^k = \sum_{i=k}^{j-1} (\mathbf{z}^{i+1} - \mathbf{z}^i).$$

Taking norms and applying the triangle inequality,

$$|\mathbf{z}^j - \mathbf{z}^k| = \left| \sum_{i=k}^{j-1} (\mathbf{z}^{i+1} - \mathbf{z}^i) \right| \leq \sum_{i=k}^{j-1} |\mathbf{z}^{i+1} - \mathbf{z}^i|. \tag{35}$$

Each increment can be bounded using the contraction property:

$$|\mathbf{z}^{i+1} - \mathbf{z}^i| = |f(\mathbf{z}^i) - f(\mathbf{z}^{i-1})| \leq \eta |\mathbf{z}^i - \mathbf{z}^{i-1}|.$$

Repeating this inequality gives

$$|\mathbf{z}^{i+1} - \mathbf{z}^i| \leq \eta^i |\mathbf{z}^1 - \mathbf{z}^0|.$$

Applying this in (35),

$$|\mathbf{z}^j - \mathbf{z}^k| \leq \sum_{i=k}^{j-1} \eta^i |\mathbf{z}^1 - \mathbf{z}^0|.$$

The geometric sum is

$$\sum_{i=k}^{j-1} \eta^i = \frac{\eta^k - \eta^j}{1-\eta}.$$

Thus,

$$|\mathbf{z}^j - \mathbf{z}^k| \leq \frac{\eta^k - \eta^j}{1-\eta} |\mathbf{z}^1 - \mathbf{z}^0|.$$

Finally, since $\mathbf{z}^j \to \mathbf{z}^\star$, continuity of the norm gives

$$|\mathbf{z}^\star - \mathbf{z}^k| = \lim_{j \to \infty} |\mathbf{z}^j - \mathbf{z}^k|.$$

Therefore,

$$|\mathbf{z}^\star - \mathbf{z}^k| \leq \lim_{j \to \infty} \frac{\eta^k - \eta^j}{1-\eta} |\mathbf{z}^1 - \mathbf{z}^0| = \frac{\eta^k}{1-\eta} |\mathbf{z}^1 - \mathbf{z}^0|.$$

□

Proposition 3.3. *Fix t, $\mathbf{x}$, and $0 \leq \eta < 1$. Assume that $\hat{D}_t(\mathbf{x}, \cdot)$ is a contraction on a nonempty closed set $O \subseteq \mathbb{R}^{L \times d}$ with factor η. Then, the iteration $\mathbf{z}^{j+1} = \hat{D}_t(\mathbf{x}, \mathbf{z}^j)$ (10) from any $\mathbf{z}^0 \in O$ satisfies the following properties:*

(i) It converges exponentially to a unique fixed point $\mathbf{z}^\star \in O$,

(ii) Its denoising error at any iteration $j \geq 1$ is bounded as

$$|\mathbf{z}^j - D_t(\mathbf{x})| \leq |\hat{D}_t(\mathbf{x}, \mathbf{z}^0) - D_t(\mathbf{x})| + \frac{\eta - \eta^j}{1-\eta} |\mathbf{z}^1 - \mathbf{z}^0|. \tag{11}$$

(iii) Its denoising error at the fixed point is bounded as

$$|\mathbf{z}^\star - D_t(\mathbf{x})| \leq |\hat{D}_t(\mathbf{x}, \mathbf{z}^0) - D_t(\mathbf{x})| + \frac{\eta}{1-\eta} |\mathbf{z}^1 - \mathbf{z}^0|, \tag{12}$$

Proof. Since $\hat{D}_t(\mathbf{x}, \cdot)$ is a contraction on O, by applying Lemma B.4 to it, we have that $\mathbf{z}^j$ converges exponentially to a unique fixed point $\mathbf{z}^\star \in O$. Furthermore, we have

$$|\mathbf{z}^j - \mathbf{z}^k| \leq \frac{\eta^k - \eta^j}{1-\eta} |\mathbf{z}^1 - \mathbf{z}^0|, \quad |\mathbf{z}^\star - \mathbf{z}^k| \leq \frac{\eta^k}{1-\eta} |\mathbf{z}^1 - \mathbf{z}^0|. \tag{36}$$

By the triangle inequality,

$$|\mathbf{z}^j - D_t(\mathbf{x})| \leq |\mathbf{z}^j - \hat{D}_t(\mathbf{x}, \mathbf{z}_0)| + |\hat{D}_t(\mathbf{x}, \mathbf{z}_0) - D_t(\mathbf{x})|.$$

By the definition of the iteration,

$$|\mathbf{z}^j - D_t(\mathbf{x})| \leq |\mathbf{z}^j - \mathbf{z}^1| + |\hat{D}_t(\mathbf{x}, \mathbf{z}_0) - D_t(\mathbf{x})|.$$

Using the first part of (36) with $k = 1$,

$$|\mathbf{z}^j - D_t(\mathbf{x})| \leq |\hat{D}_t(\mathbf{x}, \mathbf{z}_0) - D_t(\mathbf{x})| + \frac{\eta - \eta^j}{1-\eta} |\mathbf{z}^1 - \mathbf{z}^0|.$$

Applying the same argument, with the second part of (36),

$$|\mathbf{z}^\star - D_t(\mathbf{x})| \leq |\hat{D}_t(\mathbf{x}, \mathbf{z}_0) - D_t(\mathbf{x})| + \frac{\eta}{1-\eta} |\mathbf{z}^1 - \mathbf{z}^0|.$$

□

B.4 Proof of Proposition 3.4

Proposition 3.4. *Assume that, for every $t \in [0,1)$ and $\mathbf{x}$, the self-conditioned denoiser $\hat{D}_t(\mathbf{x},\cdot)$ has a unique fixed point $D_t^\star(\mathbf{x}) = \hat{D}_t(\mathbf{x}, D_t^\star(\mathbf{x}))$. Define the* ***fixed-point velocity***

$$b_t^\star(\mathbf{x}) := \frac{D_t^\star(\mathbf{x}) - \mathbf{x}}{1-t}, \quad t \in [0,1). \tag{14}$$

Then $b_t^\star$ depends only on t and $\mathbf{x}$, and therefore the dynamics

$$\dot{\mathbf{x}}_t = b_t^\star(\mathbf{x}_t) \tag{15}$$

define an ordinary time-dependent ODE. Furthermore, when the fixed-point denoiser is Bayes optimal, $D^\star = D$, the fixed-point velocity recovers the true velocity, $b^\star = b$.

Proof. By definition, $D_t^\star(\mathbf{x})$ is obtained by solving the inner self-conditioning fixed-point problem at the pair $(t,\mathbf{x})$. Hence, after taking the fixed point, there is no remaining conditioning variable $\mathbf{z}$. Therefore

$$b_t^\star(\mathbf{x}) = \frac{D_t^\star(\mathbf{x}) - \mathbf{x}}{1-t}$$

is a function only of t and $\mathbf{x}$.

The sampling dynamics are therefore the ordinary ODE

$$\dot{\mathbf{x}}_t = b_t^\star(\mathbf{x}_t).$$

When the fixed point matches the Bayes-optimal denoiser, $D^\star = D$, then by (1), we have that

$$b_t^\star(\mathbf{x}) = \frac{D_t(\mathbf{x}) - \mathbf{x}}{1-t} = b_t(\mathbf{x}).$$

Therefore, the fixed-point velocity recovers the true velocity. □

B.5 Iteration bound under contractivity

We formally state and prove the iteration bound underlying the initialization discussion in Section 3.3.

Proposition B.5. *In the setting of Proposition 3.3, let $\varepsilon > 0$. If $0 < \eta < 1$ and $|\mathbf{z}^0 - \mathbf{z}^\star| > \varepsilon$, then any integer*

$$j \geq \left\lceil \frac{\log\big(|\mathbf{z}^0 - \mathbf{z}^\star|/\varepsilon\big)}{\log(1/\eta)} \right\rceil \tag{37}$$

guarantees $|\mathbf{z}^j - \mathbf{z}^\star| \leq \varepsilon$. If $|\mathbf{z}^0 - \mathbf{z}^\star| \leq \varepsilon$, then $j = 0$ suffices. If $\eta = 0$, then $j = 1$ suffices.

Proof. If the initial error is at most ε, no iteration is needed. If $\eta = 0$, contractivity gives

$$|\mathbf{z}^1 - \mathbf{z}^\star| = |f(\mathbf{z}^0) - f(\mathbf{z}^\star)| = 0,$$

where $f = \hat{D}_t(\mathbf{x},\cdot)$. Otherwise, $0 < \eta < 1$ and $|\mathbf{z}^0 - \mathbf{z}^\star| > \varepsilon$. The geometric convergence bound (34) in Lemma B.4 gives

$$|\mathbf{z}^j - \mathbf{z}^\star| \leq \eta^j |\mathbf{z}^0 - \mathbf{z}^\star|.$$

It therefore suffices to require

$$\eta^j |\mathbf{z}^0 - \mathbf{z}^\star| \leq \varepsilon.$$

Taking logarithms and using $\log \eta < 0$ gives

$$j \geq \frac{\log\big(|\mathbf{z}^0 - \mathbf{z}^\star|/\varepsilon\big)}{\log(1/\eta)}.$$

Taking the ceiling yields the stated integer bound. □

B.6 Proof of Proposition 3.5

Proposition 3.5. *On any interval where the fixed-point flow ODE (15) has unique solutions, the **fixed-point flow map**, its solution operator $X^\star_{s,t}(\mathbf{x}_s) = \mathbf{x}_t$, satisfies*

$$X^\star_{s,t}(\mathbf{x}) = \mathbf{x} + \int_s^t b^\star_\tau(\mathbf{x}_\tau)\,\mathrm{d}\tau. \tag{19}$$

Moreover, for $0 \le s \le u \le t < 1$,

$$X^\star_{s,t} = X^\star_{u,t} \circ X^\star_{s,u}. \tag{20}$$

Proof. Since (15) is an ODE, it defines the flow map $X^\star_{s,t}(\mathbf{x}_s) = \mathbf{x}_t$ wherever the ODE solution is unique. Uniqueness also gives the composition law: evolving from s to t is the same as first evolving from s to u, and then from u to t. Thus

$$X^\star_{s,t} = X^\star_{u,t} \circ X^\star_{s,u}.$$

□

B.7 Proof of Proposition 3.6

Proposition 3.6. *The two-time denoiser $\delta_{s,t}$ satisfies the following properties:*

(i) For $0 \le s < t < 1$, the flow map is recovered from $\delta_{s,t}$ by

$$X^\star_{s,t}(\mathbf{x}) = \frac{1-t}{1-s}\mathbf{x} + \frac{t-s}{1-s}\delta_{s,t}(\mathbf{x}). \tag{23}$$

(ii) On the diagonal,

$$\delta_{t,t}(\mathbf{x}) = D^\star_t(\mathbf{x}). \tag{24}$$

(iii) For $0 \le s < u < t < 1$, the semigroup condition (20) is equivalent to

$$\delta_{s,t}(\mathbf{x}) = \gamma\delta_{s,u}(\mathbf{x}) + (1-\gamma)\delta_{u,t}(X^\star_{s,u}(\mathbf{x})), \tag{25}$$

where $\gamma = \frac{(1-t)(u-s)}{(1-u)(t-s)} \in [0,1]$.

Proof. First, by definition,

$$\delta_{s,t}(\mathbf{x}) = \mathbf{x} + (1-s)v_{s,t}(\mathbf{x}).$$

Therefore

$$v_{s,t}(\mathbf{x}) = \frac{\delta_{s,t}(\mathbf{x}) - \mathbf{x}}{1-s}.$$

Substituting this into

$$X^\star_{s,t}(\mathbf{x}) = \mathbf{x} + (t-s)v_{s,t}(\mathbf{x})$$

gives

$$X^\star_{s,t}(\mathbf{x}) = \mathbf{x} + \frac{t-s}{1-s}\big(\delta_{s,t}(\mathbf{x}) - \mathbf{x}\big).$$

Rearranging gives

$$X^\star_{s,t}(\mathbf{x}) = \frac{1-t}{1-s}\mathbf{x} + \frac{t-s}{1-s}\delta_{s,t}(\mathbf{x}),$$

which is (23).

For the diagonal identity,

$$\delta_{t,t}(\mathbf{x}) = \mathbf{x} + (1-t)b^\star_t(\mathbf{x}).$$

Since

$$b^\star_t(\mathbf{x}) = \frac{D^\star_t(\mathbf{x}) - \mathbf{x}}{1-t},$$

we get

$$\delta_{t,t}(\mathbf{x}) = D^\star_t(\mathbf{x}),$$

which is (24).

It remains to prove the semigroup identity. Let

$$\mathbf{z} := X^\star_{s,u}(\mathbf{x}) = \frac{1-u}{1-s}\mathbf{x} + \frac{u-s}{1-s}\delta_{s,u}(\mathbf{x}).$$

Using (23),

$$X^\star_{s,t}(\mathbf{x}) = \frac{1-t}{1-s}\mathbf{x} + \frac{t-s}{1-s}\delta_{s,t}(\mathbf{x}),$$

$$X^\star_{u,t}(\mathbf{z}) = \frac{1-t}{1-u}\mathbf{z} + \frac{t-u}{1-u}\delta_{u,t}(\mathbf{z}).$$

Since the semigroup property (20) holds,

$$X^\star_{s,t}(\mathbf{x}) = X^\star_{u,t}(X^\star_{s,u}(\mathbf{x})) = X^\star_{u,t}(\mathbf{z}).$$

Substituting the expression for $\mathbf{z} = X^\star_{s,u}(\mathbf{x})$ into the right-hand side gives

$$\begin{aligned} X^\star_{s,t}(\mathbf{x}) &= \frac{1-t}{1-u}\left(\frac{1-u}{1-s}\mathbf{x} + \frac{u-s}{1-s}\delta_{s,u}(\mathbf{x})\right) + \frac{t-u}{1-u}\delta_{u,t}(\mathbf{z}) \\ &= \frac{1-t}{1-s}\mathbf{x} + \frac{(1-t)(u-s)}{(1-u)(1-s)}\delta_{s,u}(\mathbf{x}) + \frac{t-u}{1-u}\delta_{u,t}(\mathbf{z}). \end{aligned}$$

and canceling the common $\mathbf{x}$-term gives

$$\frac{t-s}{1-s}\delta_{s,t}(\mathbf{x}) = \frac{(1-t)(u-s)}{(1-u)(1-s)}\delta_{s,u}(\mathbf{x}) + \frac{t-u}{1-u}\delta_{u,t}(\mathbf{z}).$$

Multiplying by $(1-s)/(t-s)$, we obtain

$$\delta_{s,t}(\mathbf{x}) = \frac{(1-t)(u-s)}{(1-u)(t-s)}\delta_{s,u}(\mathbf{x}) + \frac{(1-s)(t-u)}{(1-u)(t-s)}\delta_{u,t}(\mathbf{z}).$$

Define

$$\gamma = \frac{(1-t)(u-s)}{(1-u)(t-s)}.$$

Then

$$1-\gamma = \frac{(1-s)(t-u)}{(1-u)(t-s)}.$$

Therefore

$$\delta_{s,t}(\mathbf{x}) = \gamma\delta_{s,u}(\mathbf{x}) + (1-\gamma)\delta_{u,t}\big(X^\star_{s,u}(\mathbf{x})\big).$$

Conversely, if this identity for δ holds, then applying (23) to both sides gives

$$X^\star_{s,t}(\mathbf{x}) = X^\star_{u,t}(X^\star_{s,u}(\mathbf{x})).$$

Thus the two identities are equivalent.

□

B.8 Proof of Proposition 3.7

Proposition 3.7. *Consider the population objective*

$$\mathcal{L}(\delta) = \mathbb{E}_{s,u,t,\mathbf{x}}|\delta_{s,t}(\mathbf{x}) - \mathsf{sg}(\mathcal{T}(\delta)_{s,u,t}(\mathbf{x}))|^2 + \mathbb{E}_{t,\mathbf{x}}|\delta_{t,t}(\mathbf{x}) - D^\star_t(\mathbf{x})|^2 \tag{26}$$

where

$$\mathcal{T}(\delta)_{s,u,t}(\mathbf{x}) := \gamma\delta_{s,u}(\mathbf{x}) + (1-\gamma)\delta_{u,t}\big(X^\star_{s,u}(\mathbf{x})\big). \tag{27}$$

Then the true two-time denoiser has zero loss. Conversely, any zero-loss solution satisfies the diagonal (24) and semigroup (25) conditions almost surely under the training distribution.

Proof. By the two-time denoiser identities (24) and (25), the true two-time denoiser δ satisfies

$$\delta_{t,t}(\mathbf{x}) = D_t^\star(\mathbf{x})$$

and

$$\delta_{s,t}(\mathbf{x}) = \gamma\delta_{s,u}(\mathbf{x}) + (1-\gamma)\delta_{u,t}\big(X_{s,u}^\star(\mathbf{x})\big).$$

Therefore both squared-error terms in $\mathcal{L}(\delta)$ are zero, and hence

$$\mathcal{L}(\delta) = 0.$$

Conversely, suppose $\mathcal{L}(\delta) = 0$. Since $\mathcal{L}$ is a sum of nonnegative squared norms, each squared norm must vanish almost surely under its corresponding training distribution. The diagonal term gives

$$\delta_{t,t}(\mathbf{x}) = D_t^\star(\mathbf{x})$$

almost surely. The semigroup term gives

$$\delta_{s,t}(\mathbf{x}) = \mathcal{T}(\delta)_{s,u,t}(\mathbf{x})$$

almost surely. Substituting the definition of $\mathcal{T}$, this becomes

$$\delta_{s,t}(\mathbf{x}) = \gamma\delta_{s,u}(\mathbf{x}) + (1-\gamma)\delta_{u,t}\big(X_{s,u}^\star(\mathbf{x})\big)$$

almost surely. Thus any zero-loss solution satisfies the diagonal and semigroup consistency identities on the training distributions. □

C Experiment Details

C.1 Analysis details

Fixed-point convergence. For the convergence analysis in Figure 2(a), we construct $\mathbf{x}_t$ from 256 training examples at $t \in \{0.0, 0.1, \ldots, 0.9\}$ and average the relative L2 distances over both the examples and the ten time points. We approximate the fixed point using 200 damped Picard iterations (Lauriere, 2021):

$$\mathbf{z}^{j+1} = \alpha\mathbf{z}^j + (1-\alpha)\hat{D}_t(\mathbf{x}_t, \mathbf{z}^j),$$

where $\mathbf{z}^0 = \mathbf{0}$ and $\alpha = 0.3$. The reference $\mathbf{z}^\star$ is the final iterate $\mathbf{z}^{200}$ from the same run.

Generation quality. For the iteration sweep in Figure 2(b,c), we use warm-start ODE sampling with undamped Picard iterations ($\alpha = 0$), setting $\gamma = 0$ for ELF and softmax temperature 1.0 for LangFlow.

Warm-start and cold-start sampling. For the comparison in Figure 3, we use 32 sampling steps for ELF and 64 for LangFlow, as in Figure 2(b,c). In both schemes, we use $\alpha = 0$ with one iteration and $\alpha = 0.3$ with 100 iterations per flow timestep; warm starts use $\hat{\mathbf{z}}_{t_0}^0 = \mathbf{0}$ at the first flow timestep. For the frontiers, we sweep the sampling parameter $\gamma \in \{0.0, 0.5, 1.0, 1.5, 2.0\}$ for ELF and the softmax temperature over $\{0.8, 0.9, 1.0, 1.1, 1.2\}$ for LangFlow.

Sampling settings. We use the PyTorch reproduction of ELF[1] and the LangFlow implementation[2] with their default time grids, solvers, and token decoding. We set the self-conditioning guidance weight of ELF to $w = 1$ and the guidance scale of LangFlow to 1.

C.2 Self-conditioning-free model details

We obtain ELF* by distilling the frozen self-conditioned ELF teacher $\hat{D}$ into $D^\star$ (13) with consistency deep equilibrium (CDEQ) distillation (Lin et al., 2026).

[1] https://github.com/Ugness/ELF-pytorch
[2] https://github.com/nealchen2003/LangFlow

Teacher. For the teacher $\hat{D}$ in Section 4.2, we train a variant of ELF-B (Hu et al., 2026) whose T5 text encoder (Raffel et al., 2020) is replaced by the last hidden states of a pretrained GPT-2 Large model (Radford et al., 2019), so that it shares the GPT-2 tokenizer with the baselines; otherwise, it follows the original ELF architecture and hyperparameters.

Architecture. ELF (Hu et al., 2026) conditions on the flow time t and the self-conditioning guidance weight w through two sets of four learned prefix tokens, to which embeddings of t and w are added, respectively. The student is initialized from the teacher, retains its decoder head, and adds a third such set for the consistency time τ (defined below); these tokens and the τ embedding are zero-initialized, so the student initially nearly reproduces the teacher.

Training. Following Lin et al. (2026), at each training step, we run the teacher's fixed-point iteration (10) at the training interpolant from $\mathbf{z}^0 = \mathbf{0}$ for 20 Anderson-accelerated iterations (history 3, mixing 0.9, and regularization 10^{-4}) and keep the iterates $\mathbf{z}^0, \dots, \mathbf{z}^{20}$, computed without gradients, taking $\mathbf{z}^{20}$ as $\mathbf{z}^\star$. We assign iterate $\mathbf{z}^j$ a consistency time $\tau_j = \varepsilon + (1 - e^{-\rho j})(T - \varepsilon)$ with $\tau_0 = \varepsilon$ ($\varepsilon = 0.002$, $T = 5.0$, $\rho = 0.3$), and let $c_{\text{skip}}(\tau) = (\tau - \varepsilon)/(T - \varepsilon)$ and $c_{\text{out}}(\tau) = 1 - c_{\text{skip}}(\tau)$. The student predicts a consistency function $g_j = c_{\text{skip}}(\tau_j)\,\mathbf{z}^j + c_{\text{out}}(\tau_j)\,P_j$, where P_j is a closed-form two-point Anderson combination (mixing 1.0) of two τ-conditioned student evaluations. We minimize $\mathcal{L} = \lambda_1 |g_j - \mathbf{z}^\star|^2 + (1 - \lambda_1)|g_j - \text{sg}(g_{j-1})|^2$ with $\lambda_1 = 0.8$. With probability 0.25, we instead regress g_0 directly onto $\mathbf{z}^\star$.

Inference. ELF* generates with the forward Euler scheme (2), using one evaluation per flow timestep at $\tau = \tau_0$ with the self-conditioning input set to $\mathbf{0}$.

C.3 Self-conditioned flow map details

Architecture. The two-time denoiser $\delta_{s,t}$ (21) uses the ELF backbone, with its four flow-time prefix tokens split evenly between the target time t and the source time s, both embedded by the shared time embedder.

Training. Following Lee et al. (2026), each training example is diagonal ($s = t$) with probability 0.4 for offline FMLM* or 0.5 for online FMLM*, and off-diagonal otherwise; $1/32$ of the off-diagonal examples are fixed at $(s, t) = (0, 1)$. For the semigroup target $\mathcal{T}(\delta)_{s,u,t}$ of (26), we evaluate $\delta_{s,u}$ and $\delta_{u,t}$ of the current student under stop-gradient, obtain $X^\star_{s,u}$ from $\delta_{s,u}$ via (23), and set u to the midpoint of s and t in percentile (CDF) coordinates of the logit-normal time distribution below. The times follow the teacher's training-time marginal, a logit-normal distribution with $(\mu, \nu) = (-1.5, 0.8)$. We compute the mean-squared error of each term of (26) separately and sum the two with equal weights.

Per example, we draw one self-conditioning guidance weight w log-uniformly from $[0.5, 5.0]$, the range used in ELF training (Hu et al., 2026), and feed it to every teacher and student evaluation in that example's loss; at inference, w is fixed across all steps.

Offline FMLM* obtains $D^\star_t$ from a single evaluation of ELF* (Appendix C.2); online FMLM* instead estimates it from the self-conditioned ELF teacher. For a budget of K fixed-point iterations (FPIs), we apply $\mathbf{z}^{j+1} = \hat{D}_t(\mathbf{x}, \mathbf{z}^j)$ for $j = 0, \dots, K - 1$ from $\mathbf{z}^0 = \mathbf{0}$ and use $\mathbf{z}^K$ as an approximation of $D^\star_t(\mathbf{x})$, so "# FPIs" in Table 3 counts all K teacher denoiser evaluations, including the first one with zero conditioning.

ELF* and FMLM* follow the original ELF hyperparameters and are each trained for 5 epochs with a global batch size of 512 using 8 NVIDIA B200 GPUs. We use a combination of Muon optimizers (peak learning rate 0.002, warmup over the first 0.5 epochs) and use the exponential moving average (EMA) of the weights with decay 0.9999 for inference.

Inference. To report the gPPL and entropy for the two- and four-step FMLM* (Tables 1 and 3), we utilize γ-sampling (Lee et al., 2026; Sabour et al., 2025; Kim et al., 2024) with $\gamma = 0.75$ and 1.0, respectively. We sweep this γ for the eight-step frontier.

D Additional Results

D.1 Generation quality across fixed-point iterations

We report the numerical results for Figure 2(b,c) in Table 2, following the setup described in Appendix C.1. As the number of fixed-point iterations increases, gPPL decreases for both ELF and LangFlow, whereas entropy remains stable near the data entropy of 5.44 nats. This supports our finding that taking more steps toward the fixed point improves generation quality.

Table 2: Effect of the number of fixed-point iterations on generation quality.

ELF (32 steps), # FPIs	1	2	3	4	5
gPPL ↓	101.19	66.50	56.67	50.25	46.31
entropy	5.51	5.42	5.37	5.33	5.31
LangFlow (64 steps), # FPIs	1	2	3	4	5
gPPL ↓	77.26	57.99	52.63	49.37	47.27
entropy	5.49	5.42	5.40	5.38	5.36

D.2 Online fixed-point distillation

We extend the online distillation results in Section 4.2 by varying the number of fixed-point iterations used during training, with results reported in Table 3. Increasing the number of FPIs improves gPPL up to 8 FPIs. With 16 FPIs, however, gPPL improves only slightly for one- and two-step generation and worsens for four-step generation, while training time increases from 23 to 32 hours, suggesting diminishing returns from additional FPIs.

Table 3: Comparison of FMLM* with online and offline fixed-point distillation. Online distillation with 8 fixed-point iterations achieves competitive generation quality at substantially lower training cost, while additional iterations provide limited benefits.

Step	Metric	# FPIs for online distillation					Offline
		1	2	4	8	16	
1	gPPL ↓	176.31	141.97	123.28	117.89	116.63	**113.91**
	entropy	5.47	5.42	5.41	5.39	5.37	5.37
2	gPPL ↓	149.51	115.25	101.27	98.96	98.61	**93.66**
	entropy	5.53	5.50	5.48	5.47	5.46	5.46
4	gPPL ↓	120.35	89.32	81.27	80.26	82.50	**73.86**
	entropy	5.46	5.44	5.45	5.44	5.44	5.35
Training time		18h	18h	20h	23h	32h	82h

E Qualitative results

E.1 Iterative refinement at a fixed flow state

We qualitatively examine how fixed-point iteration improves the model's predictions, complementing the quantitative analysis in Section 4.1. To this end, we monitor the predictions of ELF at a fixed flow state. We construct an interpolant at $t = 0.25$ from a training sequence and Gaussian noise, and compare the decoded predictions over three fixed-point iterations.

As shown in Figure 6, fixed-point iterations improve grammatical consistency and contextual coherence of the prediction. Specifically, from $j = 1$ to $j = 2$, the model improves grammatical consistency by changing "they to change" to "they can change" and "the counselors and so like" to "the counselors and the vendors." At $j = 3$, the model improves contextual coherence by changing "their voices and send customers" to "their voices and call people" and "the vendors" to "the staff,"

which fits more naturally with the references to patients and counselors. These improvements are also supported by the quantitative results in Figure 2(b,c), where gPPL decreases as the number of fixed-point iterations increases, while entropy stays close to that of the data.

Fixed-point iterations	ELF $t = 0.25$
$j = 1$	... What? Kunsho's business about helping. Many of them say **they to change their phones and send customers**. That is certainly the one reason they're catching. We get responses from not only **the customers**, but also **the counselors and so like**. ...
$j = 2$	... What? Kursho's is about helping. Many of them say **they can change their voices and send customers**. That is certainly the one reason they're catching. We get responses from not only **the patients**, but also **the counselors and the vendors**. ...
$j = 3$	... What is Kurwah's business about helping? Most of them say **they can change their voices and call people**. That is certainly the big reason they're catching. We get responses from not only **the patients**, but also **the counselors and the staff**. ...

Figure 6: Qualitative samples across fixed-point iterations. Fixed-point iterations improve grammatical consistency and contextual coherence.

E.2 Few-step generation samples

We provide generation samples from the FMLM* model. The one-step, two-step, and four-step samples are shown in Figure 7, Figure 8, and Figure 9, respectively.

Sampling Steps: 1 gPPL: 65.54, Entropy: 5.27

for Michael when he was selected to learn last week. He didn't get out of trouble, a tournament game when he came out of day 10 with a win. That's happened in the college setting versus the atmosphere that he played.
That's pretty clear why I always want to like to go five years, what I'm loving doing. It's just why I have how to feel myself every time the year. I'm going enough with the goals, talents and opportunities and feel happy what it meant me.
"As my thing is, I"t want to come across that, or what I calling it up and do the next thing. I need a few things. I feel, if I practice and go back at the year, I can be a major part of a senior team with the most significant effort I have, as a former veteran players.
I'm gonna work out of injury, and the adversity that has some value will be experienced by me. And it's a tough game every year.
My coach is a great player, and I just have to get at the moment. I'm got enough parts I the know that I'm really happy working this year. I have a good season and good day every year. I have to ask people that way and my people to understand it, to meet each other. And so I think this is hopefully for sometime coming back this year.
Here is a signed video playable on a NBA. website, can be in a digital video below.
Like now of the seasons for "LNBA teams are on freeoff for the Cavaliers through late Friday. The Cavaliers are at a superior shooting position. A team who shoot parts of-op per game, High-Footers (for example) and 3pointers (III, -1 (2-1-2) and -4), and out-inging on's pace is -09-30 (48) to -3.
The Clippers are seated in the playoff race and have shot 19.81 percent per year in the league. But this is the first such showing for the Cleveland Celtics this year. Compared to the 105 games this season:
For others, who has been up to the top of the league, have been:
1/AP / Full Press
Batum
At his opener in October NBA season, rookie Guer Turner shot an all-timer of 11.5 points (2311) in first-three points, and seven more points in advance of the 2014 finale.
Barnas is 25th out of 35 a game in the league and his three-mealers go in 34 points, +21.
He also had 15 points to open the game with the finished with 34 (one point). He had connected as a third-seash-point shot and scored a layup (but he didn't look a friendly with his teammates, but 't being a Pac-Star).
Kevin Elin re-recorded his own first plus-pointer since to shoot e, and when he came in 13 minutes, and finished with points on six of his seven Pac-e attempts, all tying the conference record. But he shot the best shot in the White House at that time, which it was his longest score of career.
Payman stays out of The league in pace and offensive/minus.
Last year's 2013 NBA season
Q20 games totaled 13-11 of the players scored in Houston's first playoff campaign as Houston's 3-3 wins.
The-Gobel, with Kevin-en, began with the Warriors in Michael Sthoel: He also added 10 points in the first five minutes of, which he was went off to win his fifth point of the game.
Meanwhile, he pushed for Michael O-o to stay up for the game over the next five games. Then also lining up with the he added 15 points. Which was -22 out, just the biggest mark in the's career.
He's shot 20 times in pre-season, and missed 15 points. The Lakers forward Quentin Billard in the second game began to move back. His foot was injured after sidelined with a toe injury. He started T. D'Angelo Robinson in the opener of Game USA.
When he stepped up to San Antonio, it wasn't much to get argue with him. But he has multiple touches in the season, however, he also led up his points for 25 points and 17.
This is the result of injury that followed a three-game incident with an Andrew on St. John with a knee injury. He played 20 minutes in 10 games through the end of 2013, which has seen the game improve this year.
"I have been looking to be there because of a a little challenge," Dr. said.

Figure 7: A sample generated by FMLM* with one-step sampling.

Sampling Steps: 2 gPPL: 62.97, Entropy: 5.28

Part of gamemaking is an intense field with more funding being forthcoming (I want to announce more next year). All major players will be making different statements on social media sites, as they may say, and we may then find some way to upget players to get that long way. But it might be good time to post more certain important things during the year.
Drake
The game will provide one of our statements:
Obviously, something here is nothing just that we'll never be all the more involved here; there are others that we're not going to be an Easyway and stop it again.
Darth also has a great way of making people right now, so you donate. You all like, share it, and share it to build your following. If you're the follower.
Darth we offer me the incredible opportunity to win a new game on Jump Stones, and I feel activated. The entire campaign happens everyday as you set up and into traction.
It's like a million thousand followers that requires more more than compliments than a smile, obviously. There are still ticket updates, raffles, newbies, campaigns, thousands and dollars to be spent in helping you earn a campaign goal.
Here is some. Set out the bonus points:
In order to bring donations to new players, Jump Stones naturally expands its functionality as the better it offers the best results.
I'm looking proud to give out to Jason [Dundy back in 2004]. I came back to the game at 7 years old, some kids began playing fast shooting games, became gamers, and played games.
In the newsletter I had buttons titled "Make Good Games Again, I recently completed a senior player" video that took all of these and notes, getting several people in the audience to take the details of how I changed my mindset.
I couldn't give them the kind of reason I just didn't make a jump game, as a challenge after a few years. But it was a very refreshing realization. When my brother had read the first article about the game, I wanted to deal up with new issues around flight and in a civilized way at Jump Times. But for the large gamer living in the first Zelda game Minecraft, everyone stands for them. The promotion for Derrader's Fortune sets where we need to move forward in time. One simple idea justices you to releasing the first game in months. This strategy basically depends on the desire to make a game that works with PC traffic there that are otherwise casual, along with things you were forced to say about. In my years of working, I hope you saw that the game has been on the rise. The game world has changed since it launched. I have never had the ability to appreciate it, but even something that we've experienced since Jump Time, I also needed a healthy amount of gratitude and Star Tennis.
But as a gamer, you have the opportunity to look back at the hardest to do with it, and your feeling that every aspect of our life has been meaningful to all our players. Before we decided to change our mind, we have never done right for us repay our costs and we needed to take responsibility in our game strategy. Even at the same time that our game would lead the company as a teenager, we couldn't.
If you have a goal, you are not well born warriors. After all, you can fight your world's game, and this game game is hope and protect you.
If you come back to the goal you hope you do, and you don't actually want to pay an expense, make yourself smile by it.
1. Overthinking goals
No. When you are involved it isn't the key. Many people don't have knowledge about how to get them out of the next year. Contrary to the catch tip to remember that implies, there are criteria that practice is difficult to accomplish, and mistakes to pursue. The other decisions will also be influenced on the way a game approaches.
2. As a result, you are ready to struggle to make an game. If you call yourself a novice, you'll gain a challenge and there will always the time to start with it.
3. Be sure to focus your time to your time as possible…
Statsicizes how much time your season will take to better see the capacity you require could be compromised. Humping time will be the approximate amount of time, the risk that this situation will become the game's total time will entail when you get out going.
In other words, you are sure to get the game well during the day as you try your problems.
From the next months, you are likely to end up behind. Publisher updates will be the place and display a various steps along your way to promote H

Figure 8: A sample generated by FMLM* with two-step sampling.

Sampling Steps: 4 gPPL: 49.57, Entropy: 5.25

the link and the available guidelines for preparing your boat.
How to Do Your Way
We wanted to make a big device - something like unlike the previous 2, which was a 10 small holder and it would be one of the easiest devices to do your sailing.
We're going to go into an example: by the way if you want to use it, you'll have multiple batteries. You'll also need to have an "open" mount on some specific surface to know exactly how to move.
A Monthing It Up
So out, it was a bit tricky to getting one big idea. I started dealing with all kinds of looking devices lately, and it took a lot. But, with the latest one I tried, I didn't expect it. I just hoped it wouldn't...
We like these kind of baskets. They're awesome. I've been doing it for probably over a long year. There was a story that got it very great attention on a youtube post, which contains a decent kitty explanation:
I also used a million-sacricator called Joodoo, to help me fix the boat. It came out with great. It's also extremely durable, and made with appropriate materials. The time needed for a video holder was too big. I did a little work on non-scrabling, and popped it up a-handed, while trying to ignore all the parts.
Anyway, my idea began to work: I decided to single move the holder to the water to build the new device we were going to need. The first piece has a SuperClipper rod on a modified handle piece; which ensures it will break up if it falls into the water. It has a heel handle that allows you to use the time to let it push properly, although it feels for the gear to break. It takes about five minutes of time to actually break up the boat, even if with any other items on the ground: just holding the handle onto a spoon and wiping it with flups.
Beyond that, I'm starting to consider some better solution: I'll think about sharing it to you, and making this thing better.
While the few options would work well, I've had a good time with my efforts. I've also set up a photo video machine and bought a new version of Didtex that supplies IntX. If you want the opportunity to share it with me think it's a really useful device, but here's the purpose of the technique:
First pads are going to set the path to the boat, and then pull it off. At first, they'll take a few minutes working. But correct the calibrations and nothing starts to break, but they can be needed for a long time.
So that we've got it together: You have a question of what exactly technique you need to get it close to you. By doing this work, it will take one step-two way. First you'll have to modify the bottom of the boat, scoop it out of the container and put it on a plate. The plate has fine fibers. That takes a half as long as any other uses it, but I hope you can find something that you can break in a short time. But it's a big challenge.
To apply it, I also used newproly white and 3T batteries for my water repair device (the size is more than 100 inches). This is a really handy, because it can bend something easily after holding something, so only so long before anyone is able to push it.
Once everything is dried properly out of the container that we have this hardware, I melted it onto my machine, which only took me a decent amount of time and can also be done in a workshop. Round holders are a good option, and they work really good.
They're sturdy, but I don't know them. I didn't know any ones but I was very sure it's just how good it was made. It took a little bit to get working on the ball holder, so I picked it from out. I, however, tested it up with a water cutter and it worked the same. This also allows me able to repair the damage by putting additional elements of the metallic structure into the stone, which is really great what I wanted to do.
Now I have to begin to note that my "kanging" price is $200, but at my money it would rate the best price. People are seeing the sensation of this giant stone coming out instead. The technique is a great demo, but it does have some value underneath: "How do you do to repair the stone?" and then the whole player walks up to the shelf and clips it. After re-kraft techniques like this, I had to go in and modify the respective karning parts. This was definitely my first outing from an extended review of the Ins1 model.
On the Features and Implications: I have

Figure 9: A sample generated by FMLM* with four-step sampling.