EFFICIENT REASONING WITH FLOW LANGUAGE MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Flow Language Models (FLMs) have emerged as a continuous-state alternative to discrete diffusion language models, yet the role of their continuous representations in reasoning remains unclear. We investigate this question by comparing the reasoning efficiency of FLMs and discrete diffusion models, measured by solution accuracy under matched denoising steps. Unlike discrete diffusion, which passes categorical states between denoising steps, FLMs evolve a continuous sequence representation throughout denoising and decodes it into discrete tokens only at the end. Our theoretical analysis shows, from a superposition perspective, how information retained in these continuous states can benefit reasoning. Intermediate-state interventions provide further empirical support for this theoretical account, showing that removing information about alternative candidates reduces subsequent solution recovery. Together, these findings show that FLMs allow evidence for multiple candidates to persist and inform subsequent reasoning before a discrete answer is produced. Furthermore, our experiments on maze planning and Sudoku tasks show that FLMs achieve greater reasoning efficiency in the few-step regime: FLMs achieves higher sequence accuracy than discrete diffusion baselines at matched model sizes and small denoising steps. On maze planning tasks, FLMs can also achieve comparable accuracy with smaller models. For example, on Maze15, FLM reaches the 95% accuracy target at 64 denoising steps with 36.5% fewer parameters than MDLM. These findings point to continuous state spaces as a promising foundation for reasoning models that require fewer refinement steps.

1 INTRODUCTION

Reasoning is essential for many real-world decision-making problems. It requires models to integrate partial information and refine intermediate computational states before reaching a valid solution (Xu et al., 2025; Patil & Jadon, 2025). The reasoning capabilities of autoregressive models have been extensively studied, and their capabilities largely extended through chain-of-thought prompting (Wei et al., 2022) and more recent continuous chain-of-thought variants (Hao et al., 2024; Zhu et al., 2026). These methods allocate additional computation to a sequential chain of intermediate thoughts. While effective on many tasks, this sequential formulation alone may lack the flexibility needed for exploration, planning, or comparing multiple candidate solution paths (Yao et al., 2023). This motivates the study of alternative generation paradigms for reasoning.

Among these alternatives, diffusion language models provide a framework for iterative sequence generation (Austin et al., 2021; Lou et al., 2023; Sahoo et al., 2024). They refine a sequence over multiple denoising steps without requiring a fixed left-to-right generation order. The denoising process allows multiple token positions to be updated together, with some formulations also permitting previously generated tokens to be revised in subsequent steps. This ability to jointly predict undecoded tokens can also benefit reasoning tasks that require planning and global coherence (He et al., 2026).

Since language is typically represented by discrete tokens, many diffusion language models perform denoising directly in a discrete state space (Austin et al., 2021; Lou et al., 2023; Sahoo et al., 2024; Ye et al., 2025). In standard discrete sampling, the denoiser uses interactions across positions to predict token probabilities, but the state passed to the next step contains only a single categorical value per position, which may be a token or a mask. The probabilities assigned to alternative tokens are therefore not explicitly preserved in this state. Early token choices may also influence subsequent

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.

predictions at other positions through bidirectional attention, making refinement sensitive to local decisions made before sufficient information has been integrated. It therefore remains unclear whether discrete states provide a sufficiently expressive medium for iterative reasoning.

Motivated by latent reasoning (Hao et al., 2024), we ask whether reasoning can instead unfold in a continuous space without committing to a categorical state at each denoising step. Flow Language Models (FLMs) (Lee et al., 2026; Roos et al., 2026; Potaptchik et al., 2026) realize this idea by evolving continuous sequence representations throughout denoising and decoding discrete tokens only at the end. Their intermediate states can retain soft representations of multiple candidate tokens, allowing subsequent computation to refine these alternatives before producing a discrete answer.

Recent work has developed continuous diffusion and flow-matching methods for language modeling (Hu et al., 2026; Chen et al., 2026), but their reasoning efficiency remains less understood. Our goal is to compare the reasoning efficiency of continuous FLMs with discrete diffusion models, measured by solution accuracy under a fixed budget of denoiser evaluations. To isolate reasoning capability, we focus on structured, verifiable tasks. Our results indicate that continuous FLMs can offer a better accuracy–computation trade-off than discrete diffusion models in the settings studied, particularly in the few-step regime. Our main contributions are the following.

- We systematically study the reasoning efficiency of continuous FLMs relative to discrete diffusion models, measured by accuracy under a fixed denoising step. Beyond final accuracy alone, we examine the accuracy – computation trade-off throughout iterative refinement, emphasizing the few-step regime.
- From a superposition perspective, we identify a theoretical mechanism by which continuous states preserve useful intermediate information and enable its use in subsequent denoising steps. Candidate-retention measurements, controlled interventions, and decoder probes provide empirical support for this mechanism.
- Experimentally, we demonstrate that FLMs achieve stronger few-step reasoning efficiency than discrete diffusion baselines across both maze planning and Sudoku, attaining higher sequence accuracy under matched model sizes and small denoising steps. On the maze benchmarks specifically, the advantage of FLMs over MDLM becomes more pronounced as the task difficulty increases, and FLMs reach matched accuracy targets with smaller models.

2 Problem Setup

Tokenized sequence representation. Let $\mathcal{V}_{\text{tok}}$ denote the tokenizer vocabulary and $K = |\mathcal{V}_{\text{tok}}|$. Each conditioning input and target output are serialized and tokenized into sequences $\mathbf{c} = (c_1, \ldots, c_{M_c})$ and $\mathbf{x} = (x_1, \ldots, x_{M_x})$. Let $\boldsymbol{e}_a \in \mathbb{R}^K$ denote the one-hot vector associated with token $a \in \mathcal{V}_{\text{tok}}$. For a token sequence $\mathbf{s} = (s_1, \ldots, s_m)$, we write the one-hot representation of $\mathbf{s}$ as $\mathrm{OH}(\mathbf{s}) = [\boldsymbol{e}_{s_1}, \ldots, \boldsymbol{e}_{s_m}]^\top \in \mathbb{R}^{m \times K}$. The conditioning and clean target states are

$$\mathbf{C} = \mathrm{OH}(\mathbf{c}) \in \mathbb{R}^{M_c \times K}, \qquad \boldsymbol{Z}_1 = \mathrm{OH}(\mathbf{x}) \in \mathbb{R}^{M_x \times K}. \tag{1}$$

Conditional flow matching. FLM keeps $\mathbf{C}_{\mathcal{G},\boldsymbol{q}}$ fixed and evolves a continuous answer state $\boldsymbol{Z}_t \in \mathbb{R}^{M_x \times K}$. During training, we sample $\boldsymbol{Z}_0 \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$ and construct the linear interpolation

$$\boldsymbol{Z}_t = (1-t)\boldsymbol{Z}_0 + t\boldsymbol{Z}_1, \qquad t \sim \mathcal{U}[0,1]. \tag{2}$$

Given $(\boldsymbol{Z}_t, t, \mathbf{C}_{\mathcal{G},\boldsymbol{q}})$, the model predicts a distribution over the tokenizer vocabulary at each answer position: $\boldsymbol{P}_\theta(\boldsymbol{Z}_t, t, \mathbf{C}_{\mathcal{G},\boldsymbol{q}}) \in (\Delta^{K-1})^{M_x}$. We train the model to recover clean answer tokens using

$$\mathcal{L}_{\text{FM}}(\theta) = \mathbb{E}\left[-\textstyle\sum_{i=1}^{M_x} \log \boldsymbol{P}_\theta(\boldsymbol{Z}_t, t, \mathbf{C}_{\mathcal{G},\boldsymbol{q}})_{i,x_i}\right]. \tag{3}$$

At inference time, we initialize $\boldsymbol{Z}_0 \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$ and numerically integrate

$$\tfrac{d\boldsymbol{Z}_t}{dt} = \tfrac{\boldsymbol{P}_\theta(\boldsymbol{Z}_t, t, \mathbf{C}_{\mathcal{G},\boldsymbol{q}}) - \boldsymbol{Z}_t}{1-t}, \qquad 0 \le t < 1. \tag{4}$$

Using N integration steps requires N forward passes. After the final update, the answer is discretized once as $\widehat{x}_i = \arg\max_{a \in \mathcal{V}_{\text{tok}}} (\boldsymbol{Z}_1)_{i,a}$, for all $i = 1, \ldots, M_x$. The decoded token sequence is then detokenized and parsed into the predicted path, $\widehat{Y}$.

3 THEORETICAL ANALYSIS

We begin by developing theoretically a representational account of how continuous FLM states preserve and use information not captured by their hard-token projections. Section 3.1 shows that FLM denoising accumulates a temporal superposition of candidate sequences. Section 3.2 characterizes how retained non-top-1 historical support can affect model computation and clean-token margins without changing the current hard state. Finally, Section 3.3 quantifies the task-relevant predictive information available beyond hard projection through decoder-risk bounds.

3.1 TEMPORAL CANDIDATE-SEQUENCE SUPERPOSITION

Let $\mathcal{S}_{M_x} = \mathcal{V}_{\text{tok}}^{M_x}$ denote the set of candidate token sequences. Each $\mathbf{s} = (s_1, \ldots, s_{M_x}) \in \mathcal{S}_{M_x}$ corresponds to the one-hot vertex $\mathbf{E}^{(\mathbf{s})} = \text{OH}(\mathbf{s})$, $\mathcal{H} = \left\{\mathbf{E}^{(\mathbf{s})} : \mathbf{s} \in \mathcal{S}_{M_x}\right\}$, where $\mathcal{H}$ is the set of complete hard-token states. At time t, let $\boldsymbol{P}_t = \boldsymbol{P}_\theta(\boldsymbol{Z}_t, t, \mathbf{C}_{\mathcal{G},\boldsymbol{q}})$ be the position-wise clean-token prediction and define $\pi_{\mathbf{s}}(t) = \prod_{j=1}^{M_x} (\boldsymbol{P}_t)_{j,s_j}$. These weights form a probability distribution and yield the exact decomposition $\boldsymbol{P}_t = \sum_{\mathbf{s}\in\mathcal{S}_{M_x}} \pi_{\mathbf{s}}(t)\mathbf{E}^{(\mathbf{s})}$. Thus, each instantaneous clean prediction is a convex superposition of candidate-sequence vertices.

Theorem 1 (Candidate-sequence superposition in FLM) *Suppose that the FLM state follows*

$$\frac{d\boldsymbol{Z}_t}{dt} = \frac{\boldsymbol{P}_t - \boldsymbol{Z}_t}{1-t}, \qquad 0 \le t < 1, \tag{5}$$

and that $\boldsymbol{P}_t$ is integrable along the trajectory. Then, for every $0 < t < 1$,

$$\boldsymbol{Z}_t = (1-t)\boldsymbol{Z}_0 + t\,\overline{\mathbf{E}}_t, \qquad \overline{\mathbf{E}}_t = \textstyle\sum_{\mathbf{s}\in\mathcal{S}_{M_x}} \overline{\pi}_{\mathbf{s}}(t)\mathbf{E}^{(\mathbf{s})}, \tag{6}$$

where

$$\overline{\pi}_{\mathbf{s}}(t) = \tfrac{1-t}{t}\textstyle\int_0^t \frac{\pi_{\mathbf{s}}(u)}{(1-u)^2}\,du, \qquad \overline{\pi}_{\mathbf{s}}(t) \ge 0, \qquad \textstyle\sum_{\mathbf{s}} \overline{\pi}_{\mathbf{s}}(t) = 1. \tag{7}$$

Therefore, apart from the residual initial noise $(1-t)\boldsymbol{Z}_0$, the FLM state contains a temporally accumulated convex superposition of candidate-sequence vertices. If two distinct sequences have positive integrated weights, then $\overline{\mathbf{E}}_t \notin \mathcal{H}$; hence, this denoised component cannot be represented by any single complete hard-token state.

Theorem 1 shows that the FLM state preserves historical support for non-top-1 candidates through a discounted superposition of past token distributions. Our intermediate-state intervention experiments further suggest that FLM uses those superposed candidate information for subsequent refinement during reasoning (Section 4.4). We next characterize how historical candidate information retained by superposition can affect clean-token predictions without changing the current argmax. Discrete solvers retaining only the current hard-token sequence cannot distinguish such histories at fixed time.

3.2 HISTORICAL INFLUENCE WITHIN A FIXED HARD STATE

We compare the original state with one in which a non-top-1 candidate's historical contribution is removed. Fix the context, time $t \in (0,1)$, parameters, and attention mask. Let Π denote rowwise argmax with fixed tie-breaking, and $\mathbf{x} = \Pi(\boldsymbol{Z}_t)$ the current hard-token sequence. For a strict non-top-1 candidate v at free position j, define

$$h_t = t(\overline{\mathbf{E}}_t)_{j,v} > 0, \qquad \mathbf{U}_t(\lambda) = \boldsymbol{Z}_t - (h_t - \lambda)\mathbf{D}^{jv}, \qquad \mathbf{D}^{jv} = \mathbf{e}_j\mathbf{e}_v^\top,$$

where $\lambda \in [0, h_t]$ is the retained historical support. Let $\boldsymbol{P}_t^{(\lambda)}$ be the full prediction recomputed on $\mathbf{U}_t(\lambda)$, and write $M_t^{(\lambda)} = (\boldsymbol{P}_t^{(\lambda)})_{j,v} - (\boldsymbol{P}_t^{(\lambda)})_{j,x_j}$, with $M_t = M_t^{(h_t)}$.

For first-layer scaled dot-product attention, denote the queries, keys, values, and softmax weights by $\mathbf{q}_i^b, \mathbf{k}_m^b, \mathbf{v}_m^b, a_{im}^b$, respectively, with $\mathbf{o}_i^b = \sum_m a_{im}^b \mathbf{v}_m^b$. Write $M_t^{(\lambda)} = F_t(O(\lambda), \xi(\lambda))$: O collects all head outputs at all positions, ξ collects bypass inputs, and F_t includes all subsequent computation. With overdots denoting derivatives with respect to λ, set

$$\mathbf{g}_i^b = \nabla_{\mathbf{o}_i^b} F_t, \quad \eta_{im}^b = \langle \mathbf{g}_i^b, \mathbf{v}_m^b \rangle, \quad \rho_{im}^{Q,b} = \tfrac{\langle \dot{\mathbf{q}}_i^b, \mathbf{k}_m^b \rangle}{\sqrt{d_b}}, \quad \rho_{im}^{K,b} = \tfrac{\langle \mathbf{q}_i^b, \dot{\mathbf{k}}_m^b \rangle}{\sqrt{d_b}},$$

where d_b is the head dimension. All quantities depend on λ; expectations and covariances below use the attention distribution $m \sim a_i^b$ over visible positions.

Theorem 2 (Historical influence within a fixed hard state) *Assume deterministic evaluation and that the query, key, value, and bypass maps, together with the downstream map F_t, are continuously differentiable on open sets containing their respective inputs along the intervention path. For every $\lambda \in [0, h_t]$,*

$$\Pi(\mathbf{U}_t(\lambda)) = \mathbf{x},\ \partial_\lambda M_t^{(\lambda)} = \mathcal{C}_Q + \mathcal{C}_K + \mathcal{C}_V + \mathcal{C}_R =: \mathcal{C}(\lambda), \text{where} \tag{8}$$

$$\mathcal{C}_X = \textstyle\sum_{b,i} \mathrm{Cov}_{m\sim a_i^b}(\rho_{im}^{X,b}, \eta_{im}^b),\ X \in \{Q, K\};\ \mathcal{C}_V = \textstyle\sum_{b,i} \mathbb{E}_{m\sim a_i^b}\langle \mathbf{g}_i^b, \dot{\mathbf{v}}_m^b\rangle,\ \mathcal{C}_R = \langle \nabla_\xi F_t, \dot{\xi}\rangle. \tag{9}$$

If the maps preceding attention act independently at each position, then for $i \neq j$ with j visible,

$$\dot{\mathbf{o}}_i^b = a_{ij}^b(\mathbf{v}_j^b - \mathbf{o}_i^b)\rho_{ij}^{K,b} + a_{ij}^b \dot{\mathbf{v}}_j^b. \tag{10}$$

Wherever $\mathcal{C}(\lambda) \neq 0$, sufficiently small admissible nonzero perturbations change the margin without changing $\mathbf{x}$. Moreover,

$$\Delta_t := M_t - M_t^{(0)} = \textstyle\int_0^{h_t} \mathcal{C}(\lambda)\, d\lambda. \tag{11}$$

Let $\Omega_t = \max_\lambda M_t^{(\lambda)} - \min_\lambda M_t^{(\lambda)}$ over this path. $\Omega_t > 0$ if and only if $\mathcal{C} \not\equiv 0$. Every scalar readout H using only context, time, and the hard state satisfies

$$\sup_{\lambda\in[0,h_t]} \left| M_t^{(\lambda)} - H(\mathbf{C}_{\mathcal{G},\boldsymbol{q}}, t, \Pi(\mathbf{U}_t(\lambda))) \right| \geq \tfrac{\Omega_t}{2} \geq \tfrac{|\Delta_t|}{2}. \tag{12}$$

The bound is tight.

Equation 9 separates attention reweighting through Q/K, message changes through V, and bypass responses; their signed effects may reinforce or cancel. Equation 10 makes the influence on other positions explicit, while Equation 12 bounds the margin variation that a readout of the hard state cannot reproduce. We experimentally test this mechanism by removing retained non-top-1 historical support while preserving the hard-token sequence and measuring the resulting prediction-margin changes and Q/K/V/bypass contribution in Appendix D.3.

Appendix C.3 further shows how these differences propagate through Euler updates and when the original trajectory strengthens a candidate's relative support without changing the hard-token sequence. We next examine whether these differences help predict the correct solution. The following theorem quantifies this distinction through bounds on prediction risk.

3.3 Task-Relevant Information Beyond Hard Projection

At fixed time t, we predict target-path membership $Y = \mathbf{1}\{X_J = v_{\text{path}}\}$, where X_J is the target token at a uniformly sampled output position J. Inputs A_t, B_t share the context in a window $\mathcal{N}$ around J, using respectively the continuous state and its rowwise one-hot argmax with fixed tie-breaking. Let R_t^Z, R_t^{hard} denote their unweighted population BCE risks, $\Delta_t^{\text{dec}} = R_t^{\text{hard}} - R_t^Z$, and $\varepsilon_t^Z, \varepsilon_t^{\text{hard}}$ their excess risks above the corresponding Bayes risks. For $\mathbf{W}_t = \mathrm{softmax}((\boldsymbol{Z}_t)_{\mathcal{N}})$, define $A_t^{\text{soft}} = (\mathbf{C}_{\mathcal{N}}, \mathbf{W}_t)$ and $A_t^{\text{conf}} = (B_t, \max \mathbf{W}_t)$, with softmax and maxima taken rowwise.

Theorem 3 (Task-relevant information beyond hard projection) *Assume finite decoder risks and let $b_t \geq (\mathbb{E}\|(\boldsymbol{Z}_t)_{J,:} - \mathbf{e}_{X_J}^\top\|_2^2)^{1/2}$. Define the predictive signal beyond winner confidence by*

$$\nu_t = \mathbb{E}\left[\mathrm{Var}\left(\Pr(Y = 1 \mid A_t^{\text{soft}}) \mid A_t^{\text{conf}}\right)\right]. \tag{13}$$

Then, with h_{bin} denoting binary entropy in nats,

$$2\nu_t - \varepsilon_t^Z \leq \Delta_t^{\text{dec}} \leq h_{\text{bin}}\left(\min\{1/2, 2b_t^2\}\right) + \varepsilon_t^{\text{hard}}. \tag{14}$$

Also, $R_t^Z \leq h_{\text{bin}}(\min\{1/2, 2b_t^2\}) + \varepsilon_t^Z$.

The left-hand inequality in Equation 14 guarantees a predictive advantage when $2\nu_t > \varepsilon_t^Z$; the right-hand inequality implies that the Bayes-risk gap vanishes as $b_t \to 0$. We examine this retained information experimentally by using matched decoders on intermediate FLM states and their hard projections, measuring how well each representation supports prediction of the correct solution in Section 4.4. All detailed proofs are provided in the Appendix C.

Table 1: **Sequence accuracy (%) of FLM and baseline methods on Maze10, Maze15, and Maze20 across denoising steps.** + Ent. and + Conf. denote entropy-based and confidence-based remasking, respectively. For each task and method, green marks the first tested step reaching at least 95%; yellow marks the first attaining its observed maximum.

Task	*Params (M)*	*N*	CoT	D3PM	SEDD	SEDD + Ent.	SEDD + Conf.	MDLM	ReMDM	MDLM + Ent.	MDLM + Conf.	**FLM**
Maze10	*59.11*	1	50.5	77.5	94.0	94.0	94.0	87.5	87.5	**87.2**	87.5	**95.7**
		2		82.8	**96.3**	**99.7**	**98.8**	89.0	89.0	66.3	93.1	99.5
		4		85.9	96.7	98.3	*NR*	92.2	92.3	66.2	**95.4**	99.5
		8		91.4	98.8	97.0	99.4	94.0	94.2	71.4	96.2	99.5
		16		94.2	98.9	96.9	**99.7**	**96.3**	**96.4**	69.2	97.7	99.5
		32		**95.7**	99.3	96.7	99.5	97.8	97.9	75.9	98.5	99.5
		64		96.8	99.7	96.5	99.6	98.0	98.4	77.8	99.1	**99.6**
		128		97.5	**99.8**	96.0	98.9	**98.8**	**99.0**	75.9	99.3	99.6
		256		**98.4**	99.8	95.8	99.0	98.3	98.5	84.4	**99.5**	99.6
		512		98.1	99.8	95.0	99.0	98.3	98.9	83.6	99.1	99.6
Maze15	88.25	1	22.3	45.7	52.6	52.6	52.6	49.8	49.8	**49.8**	49.8	**95.9**
		2		52.2	61.2	73.3	77.5	54.9	54.9	32.0	75.7	98.9
		4		58.1	67.5	81.1	80.5	62.5	62.9	33.7	80.4	99.1
		8		66.8	73.3	84.8	85.6	69.6	69.8	35.2	85.8	99.1
		16		74.6	81.1	84.3	90.4	76.2	77.4	38.5	88.2	99.1
		32		82.4	86.8	**87.4**	94.4	83.7	84.4	39.6	93.1	99.1
		64		87.0	90.8	85.4	**96.4**	88.3	88.6	39.4	94.0	**99.2**
		128		90.8	93.2	81.6	**96.6**	91.8	92.1	39.4	**97.8**	99.2
		256		91.5	**95.0**	79.7	96.1	93.7	94.2	39.9	97.8	99.2
		512		**92.6**	**95.4**	78.4	95.9	**95.3**	**96.2**	42.1	97.1	99.2
Maze20	258.81	1	32.2	33.5	33.6	33.6	33.6	35.6	35.6	**35.3**	35.6	71.4
		2		37.3	37.8	41.7	62.4	41.2	42.6	30.8	49.3	88.1
		4		44.0	40.8	68.3	70.1	48.2	48.8	28.7	54.0	91.3
		8		51.0	49.4	84.0	76.5	55.8	54.1	29.5	60.4	92.1
		16		60.8	56.0	89.0	79.9	63.0	64.9	30.1	67.0	92.4
		32		67.5	63.8	90.0	83.3	70.6	71.7	30.0	73.1	92.5
		64		77.3	73.2	**90.4**	85.9	80.2	80.1	29.7	87.4	92.5
		128		81.8	79.4	90.0	88.4	83.7	86.7	31.1	94.7	92.6
		256		**86.5**	82.5	87.2	**92.1**	87.9	89.1	31.2	**98.3**	92.6
		512		86.3	**87.1**	84.1	91.5	**90.1**	**90.6**	32.8	**99.1**	**92.9**

4 EXPERIMENTS

Our experiments are designed to evaluate the *reasoning capability* of continuous flow-based models rather than their general language modeling ability. We therefore focus on structured, non-linguistic tasks in two complementary reasoning domains: maze planning (Section 4.2) and Sudoku (Section 4.3). The maze datasets provide a controlled family of planning problems whose difficulty increases with maze size, allowing us to study not only how performance varies with the number of denoising steps but also how it depends on task difficulty and model capacity. Sudoku-Extreme provides a substantially different constraint-satisfaction problem and allows us to test whether the behavior of FLM extends beyond path planning. Finally, motivated by our theoretical analysis, we conduct a superposition analysis of FLM's intermediate states to examine whether the information they retain contributes to subsequent reasoning (Section 4.4).

4.1 EXPERIMENTAL SETUP

Benchmarks. For maze reasoning, we construct Maze10, Maze15, and Maze20 based on the maze-generation procedure in Zhang et al. (2026). With a consistent task formulation, these datasets provide a controlled setting for studying reasoning as maze size and path-planning difficulty increase.

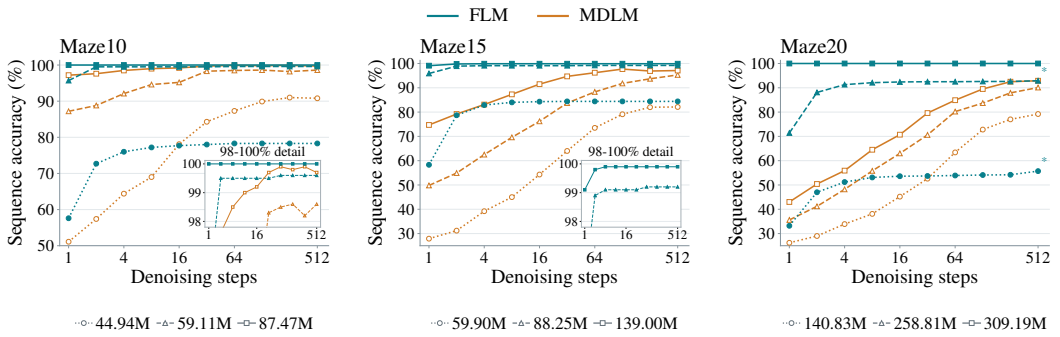


Figure 1: **Sequence accuracy across denoising steps and model capacities.** FLM and MDLM are evaluated on Maze10 (left), Maze15 (center), and Maze20 (right), with three model sizes per task. Colors distinguish methods, while marker shapes and line styles distinguish model sizes; parameter counts are listed below each panel. Insets enlarge the $98 - 100\%$ sequence accuracy range.

We additionally evaluate on Sudoku-Extreme, which requires completing a 9×9 board that satisfies all row, column, and sub-grid constraints. To assess distributional generalization, we further test models trained on Sudoku-Extreme directly on Standard Sudoku without additional fine-tuning. Construction procedures, splits, and detailed statistics for all datasets are provided in Appendix D.1.

Baselines and Evaluation. We compare FLM with representative discrete diffusion models, including D3PM (Austin et al., 2021), SEDD (Lou et al., 2023), MDLM (Sahoo et al., 2024), and ReMDM (Wang et al., 2026). Among these, ReMDM modifies the reverse transition to allow previously generated tokens to be remasked and resampled, thereby permitting error correction. For masked discrete diffusion models, we additionally evaluate confidence-based remasking, which revisits low-confidence tokens, and entropy-ordered decoding, which prioritizes low-entropy positions without revising committed tokens. A Transformer with chain-of-thought (Wei et al., 2022) serves as an additional autoregressive baseline for both maze planning and Sudoku. To ensure a fair comparison, all models use the same backbone and parameter count, with matched denoising steps for FLM and discrete diffusion methods. Across all comparisons, we use sequence accuracy as the primary metric, which counts a prediction as correct only when the complete solution sequence matches the answer.

Implementation Details. For both maze and Sudoku tasks, FLM uses a bidirectional Transformer with fixed input conditions. We train on Gaussian-to-one-hot interpolations using a clean-token cross-entropy objective, optimize with AdamW (Loshchilov et al., 2017), and evaluate EMA weights. Inference uses Euler integration on a uniform time grid followed by tokenwise argmax. All methods are adapted from their corresponding original codebases. Detailed implementations of FLM and all comparison baselines, including architecture configurations, training settings, and sampling procedures, are provided in Appendix D.2.

4.2 Maze Planning: Reasoning Efficiency across Task Difficulties

Maze provides our main controlled benchmark for studying how continuous and discrete generative reasoning behave as planning difficulty increases. We compare against baselines and study the effect of model capacity. Additional results on wall-clock inference efficiency are given in Appendix D.5.

Comprehensive comparison with baselines. Table 1 compares FLM against discrete diffusion, enhanced sampling, and autoregressive baselines. FLM achieves the highest reported one-step sequence accuracy on all three tasks and maintains a clear few-step advantage on Maze15 and Maze20. These results highlight FLM's reasoning efficiency under limited denoising budgets.

Effect of model capacity. We compare FLM and MDLM across three model sizes per maze task (Figure 1). At comparable model sizes, FLM maintains higher accuracy as maze difficulty increases in several evaluated settings, with a larger gap over MDLM on Maze15 than on Maze10 (Figure 2(a)). FLM can also reach the same accuracy target with a smaller model: on Maze15 at 64 denoising steps,

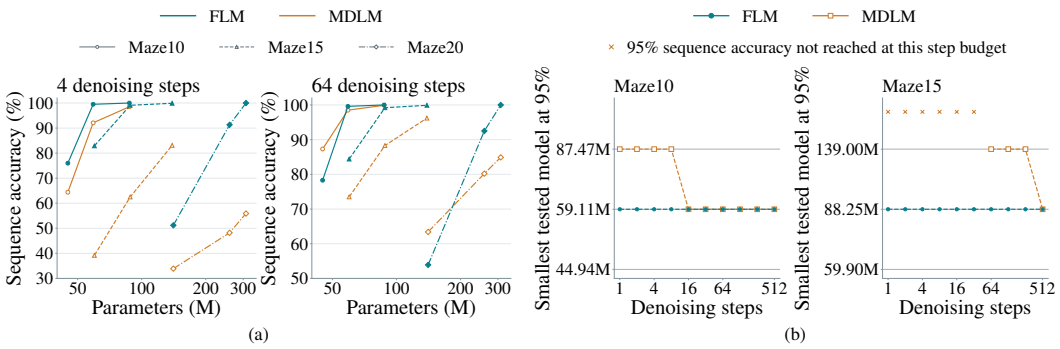


Figure 2: **Model capacity and denoising efficiency across maze difficulties.** (a) Sequence accuracy of FLM and MDLM on Maze10, Maze15, and Maze20 at 4 and 64 denoising steps. (b) Smallest tested model reaching 95% sequence accuracy at each step budget on Maze10 and Maze15. Parameter counts in (a) use a logarithmic axis, while those in (b) are displayed as categorical levels. Crosses indicate that no tested model reaches the target at the corresponding step budget.

Table 2: **Sequence accuracy (%) on Sudoku-Extreme and Standard Sudoku under selected numbers of sampling steps.** Each entry reports Sudoku-Extreme / Standard Sudoku. All models have 86.88M parameters and are trained on Sudoku-Extreme; Standard Sudoku evaluates zero-shot transfer without further fine-tuning. Yellow highlights the highest accuracy for each dataset at each step budget, including ties.

Method	Evaluation Sampling Steps					
	1	2	16	64	256	512
Transformer + CoT	2.9/3.5					
D3PM	11.2/19.5	14.0/22.9	32.0/38.6	40.3/45.7	42.3/46.1	43.1/47.9
SEDD	18.5/25.0	21.2/28.2	37.1/43.3	46.8/50.8	48.7/51.1	47.9/53.1
SEDD + Ent.	18.8/25.5	36.2/41.4	52.4/57.9	53.6/57.8	53.8/58.1	54.3/57.5
SEDD + Conf.	18.8/25.5	26.4/32.1	44.4/50.3	56.2/60.6	57.7/61.8	57.4/62.0
MDLM	19.1/27.1	21.7/29.5	40.0/45.0	47.8/52.0	47.6/53.3	49.2/53.0
MDLM + Ent.	19.2/27.1	33.5/39.5	51.2/55.6	52.5/57.0	51.8/58.3	53.6/57.6
MDLM + Conf.	19.2/27.1	26.5/33.6	47.6/51.5	60.2/64.1	62.3/65.6	62.2/64.3
ReMDM	19.2/27.1	21.7/29.6	38.8/44.4	46.4/52.4	52.0/55.0	53.8/59.2
FLM	21.0/29.5	36.7/41.4	46.3/49.5	43.9/50.5	46.0/51.4	45.5/51.3

the smallest tested models reaching 95% accuracy have 88.25M parameters for FLM and 139.00M for MDLM (Figure 2(b)). These results show an advantage for FLM on harder maze tasks under limited model capacity. We provide full results across all model sizes and denoising steps in Appendix D.4.

4.3 Sudoku: Few-Step Constraint Reasoning and Generalization

We next test whether the same few-step advantage appears in a qualitatively different reasoning domain. Unlike maze planning, Sudoku requires resolving a large collection of mutually interacting row, column, and sub-grid constraints, providing a complementary test of structured reasoning.

Few-step refinement on Sudoku-Extreme. Table 2 shows that FLM improves rapidly within the first two sampling steps. On Sudoku-Extreme, FLM achieves 21.0% sequence accuracy at one step and 36.7% at two steps, compared with 19.2% and 36.2% for the strongest discrete baseline at each budget. Further sampling yields more limited gains for FLM, whereas enhanced discrete samplers achieve higher accuracy from 16 steps onward. These results support FLM's effectiveness under few step denoising.

Generalization to Standard Sudoku. We further evaluate models trained on Sudoku-Extreme directly on Standard Sudoku without additional fine-tuning. At one sampling step, FLM achieves 29.5% sequence accuracy, exceeding the strongest discrete result of 27.1%. The single-step advantage therefore persists under this distribution shift.

4.4 Superposition Analysis

Continuous states retain non-top-1 candidate information. We first examine the accumulated denoising component $\overline{\mathbf{E}}_t$ from Theorem 1 and compare it with the instantaneous clean prediction $\boldsymbol{P}_t$ on Sudoku-Extreme. For each analyzed blank cell, we normalize the mass over the nine digit coordinates and denote the resulting distribution by $p_t(d)$, where $d \in \{1, \ldots, 9\}$. We measure We quantify the concentration of p_t through the effective number of candidates, $\exp(H(p_t))$, where $H(p_t) = -\sum_{d=1}^{9} p_t(d) \log p_t(d)$ is the Shannon entropy computed with natural logarithms. For a uniform distribution over k candidate digits, $H(p_t) = \log k$ and hence $\exp(H(p_t)) = k$, giving this measure a direct interpretation as an effective support size. As shown in Figure 3, $\boldsymbol{P}_t$ quickly concentrates on one digit, whereas $\overline{\mathbf{E}}_t$ retains more mass on secondary candidates. At $t = 0.80$, 23.8% of cells in $\overline{\mathbf{E}}_t$ retain a second candidate with probability at least 0.05, compared with only 4.9% in $\boldsymbol{P}_t$. The larger effective candidate count of $\overline{\mathbf{E}}_t$ further indicates that its mass remains more broadly distributed across candidate digits. These observations are consistent with our theoretical result.

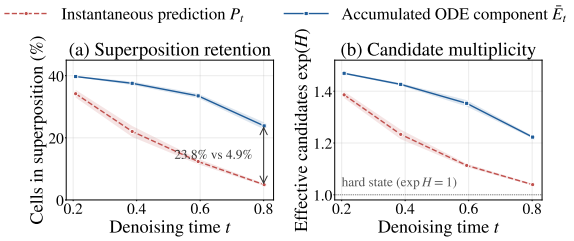


Figure 3: **Candidate retention during FLM denoising.** We compare the instantaneous clean prediction $\boldsymbol{P}_t$ with the accumulated component $\overline{\mathbf{E}}_t$. (a) shows the fraction of ambiguous cells whose second-largest digit probability is at least 0.05. (b) shows the effective number of candidates, $\exp(H)$. The accumulated component retains more non-top-1 information than the instantaneous prediction.

Intermediate-state intervention. Candidate retention alone does not establish that the model uses the retained information. We therefore intervene at an intermediate time $t^\star$ while keeping the input, initial noise, model, and subsequent denoising schedule fixed.

We compare three intermediate-state interventions against an unmodified baseline. *Natural* leaves the accumulated continuous state unchanged. *Digit-soft* masks locally infeasible digit coordinates but retains and renormalizes all locally feasible digits. *Top*-2 and *Top*-1 apply the same candidate filtering operation while retaining only the two or one highest-scoring feasible digits, respectively. Thus, Top-2 and Top-1 progressively remove non-top-1 candidate information from the intermediate state. The Digit-soft condition controls for the effect of the projection and renormalization operation itself: it modifies the state but does not remove any locally feasible candidate. We also compare renormalized and non-renormalized top-k interventions. The same degradation pattern in both settings suggests that the result is not caused solely by rescaling the retained coordinates. We measure *cell-level recovery rate* on ambiguous blank cells whose correct digit ranks second among the nine digit coordinates in the unmodified accumulated state at $t^\star$. Recovery rate measures final cell accuracy on cells whose correct digit ranks second before intervention.

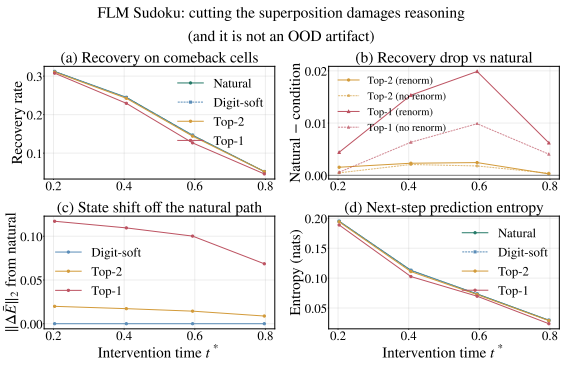


Figure 4: **Intermediate-state intervention.** Removing non-top-1 candidate information reduces recovery, while Digit-soft causes little degradation.

As shown in Figure 4, cell-level recovery rate decreases as more candidate information is removed, $\text{Recovery}_{\text{Natural}} \geq \text{Recovery}_{\text{Top-2}} > \text{Recovery}_{\text{Top-1}}$. Digit-soft causes little degradation, whereas

Top-2 and especially Top-1 reduce recovery. Moreover, Top-1 lowers rather than increases the entropy of the next-step prediction, making increased prediction uncertainty an unlikely explanation for its poorer performance. These controlled interventions provide empirical evidence that non-top-1 candidate information retained in the continuous state contributes to subsequent denoising.

Decoding task information from intermediate states. Motivated by Theorem 3, we study whether FLM states make solution-path information more accessible than hard token assignments during denoising. On Maze15, we train decoders for each representation and step of a frozen FLM's $N = 8$ denoising trajectory. Each decoder receives the maze input and the corresponding state representation and predicts whether each cell belongs to the solution path. We evaluate F1-score from pooled test-cell predictions at a fixed probability threshold of 0.5. Experimental details are in Appendix D.2.3.

Table 3: **Test F1-score for solution-path prediction along an eight-step denoising trajectory.** Bold highlights the step with the largest Raw–Hard gap.

k	Raw	Top-2	Hard	Raw − Hard
0	0.3799	0.4072	0.4090	−0.0291
1	0.3927	0.4066	0.4050	−0.0123
2	0.4765	0.4218	0.4091	+0.0674
3	0.6484	0.4674	0.4565	+0.1919
4	**0.8080**	**0.6614**	**0.5929**	**+0.2151**
5	0.8989	0.8331	0.8172	+0.0817
6	0.9158	0.9109	0.9135	+0.0024
7	0.9189	0.9184	0.9189	≈ 0
8	0.9189	0.9188	0.9189	≈ 0

We compare raw states Z_k, Top-2 representations, and one-hot argmax assignments (Hard). Top-2 retains the two largest entries of $\mathrm{softmax}(Z_k)$ at each position, setting the remaining coordinates to zero. Table 3 shows that the raw-state advantage is concentrated at intermediate steps. The largest Raw–Hard gap occurs at $k = 4$, where Raw achieves an F1-score of **0.8080**, compared with 0.6614 for Top-2 and 0.5929 for Hard. By $k = 7, 8$, Raw and Hard agree to four decimal places. These results suggest that solution-path information is more accessible to the tested decoders in intermediate continuous states, before comparable readout performance is achieved from hard token assignments.

5 Related Work

Continuous diffusion language models. Diffusion-LM (Li et al., 2022) and DiffuSeq (Gong et al., 2022) denoise embeddings, while TextLDM (Jiang et al., 2026) uses autoencoding latents. Plaid (Gulrajani & Hashimoto, 2023), ELF (Hu et al., 2026), and LangFlow (Chen et al., 2026) advance continuous diffusion and flow matching for language modeling. Other representations include relaxed vocabulary states (Han et al., 2023; Mahabadi et al., 2024; Tae et al., 2025), Riemannian categorical distributions (Cheng et al., 2024; Davis et al., 2024), and continuous binary encodings in Analog Bits (Chen et al., 2022) and CoBit (Batzolis et al., 2026).

Reasoning in continuous latent spaces. Continuous latent reasoning performs intermediate computation in hidden representations without expressing every step as text. Coconut (Hao et al., 2024) feeds hidden states back as input embeddings, forming continuous thought chains that support reasoning over multiple possible next steps. Recurrent approaches refine internal representations through repeated updates. The Hierarchical Reasoning Model (HRM) (Wang et al., 2025) couples two recurrent modules at different timescales, while the Tiny Recursive Model (TRM) (Jolicoeur-Martineau, 2025) uses a single small network to recursively update latent reasoning features and answer representations. These methods highlight the potential of iterative continuous computation for reasoning tasks.

6 Conclusion

In this work, we have examined the possible advantages of a continuous state space over a discrete one for diffusion language models. Experiments on maze planning and Sudoku show that FLM can achieve higher sequence accuracy with few denoising steps. Our superposition analysis and intermediate-state experiments support a mechanism through which continuous states preserve candidate information and use it in subsequent reasoning.

Limitations and future work. This work focuses on structured reasoning tasks and does not evaluate general language modeling ability. Further study on text-based reasoning benchmarks is needed to assess whether the observed benefits extend to natural-language reasoning. Another direction is to investigate how these benefits change when scaling FLMs to larger model capacities.

AI USE STATEMENT

In this work, we used generative AI tools to polish the manuscript and assist with code implementation. We reviewed all AI-assisted work, and two authors verified and tested the AI-generated code for correctness. We take responsibility for the final content of this work, including all text, claims, and artifacts produced with the aid of generative AI.

REFERENCES

Jacob Austin, Daniel D Johnson, Jonathan Ho, Daniel Tarlow, and Rianne Van Den Berg. Structured denoising diffusion models in discrete state-spaces. *Advances in neural information processing systems*, 34:17981–17993, 2021.

Georgios Batzolis, Mark Girolami, and Luca Ambrogioni. Cobit: Language modeling with bitstream diffusion. *arXiv preprint arXiv:2605.07013*, 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.

Chaoran Cheng, Jiahan Li, Jian Peng, and Ge Liu. Categorical flow matching on statistical manifolds. *Advances in Neural Information Processing Systems*, 37:54787–54819, 2024.

Oscar Davis, Samuel Kessler, Mircea Petrache, İsmail İ Ceylan, Michael Bronstein, and Avishek J Bose. Fisher flow matching for generative modeling over discrete data. *Advances in Neural Information Processing Systems*, 37:139054–139084, 2024.

Shansan Gong, Mukai Li, Jiangtao Feng, Zhiyong Wu, and LingPeng Kong. Diffuseq: Sequence to sequence text generation with diffusion models. *arXiv preprint arXiv:2210.08933*, 2022.

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

Xiaochuang Han, Sachin Kumar, and Yulia Tsvetkov. 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)*, pp. 11575–11596, 2023.

Shibo Hao, Sainbayar Sukhbaatar, DiJia Su, Xian Li, Zhiting Hu, Jason Weston, and Yuandong Tian. Training large language models to reason in a continuous latent space. *arXiv preprint arXiv:2412.06769*, 2024.

Andre He, Sean Welleck, and Daniel Fried. Reasoning with latent tokens in diffusion language models. *arXiv preprint arXiv:2602.03769*, 2026.

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.

Jiaxiu Jiang, Jingjing Ren, Wenbo Li, Bo Wang, Haoze Sun, Yijun Yang, Jianhui Liu, Yanbing Zhang, Shenghe Zheng, Yuan Zhang, et al. Textldm: Language modeling with continuous latent diffusion. *arXiv preprint arXiv:2605.07748*, 2026.

Alexia Jolicoeur-Martineau. Less is more: Recursive reasoning with tiny networks. *arXiv preprint arXiv:2510.04871*, 2025.

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.

Xiang Li, John Thickstun, Ishaan Gulrajani, Percy S Liang, and Tatsunori B Hashimoto. Diffusion-lm improves controllable text generation. *Advances in neural information processing systems*, 35: 4328–4343, 2022.

Ilya Loshchilov, Frank Hutter, et al. Fixing weight decay regularization in adam. *arXiv preprint arXiv:1711.05101*, 5(5):5, 2017.

Aaron Lou, Chenlin Meng, and Stefano Ermon. Discrete diffusion modeling by estimating the ratios of the data distribution. *arXiv preprint arXiv:2310.16834*, 2023.

Rabeeh Karimi Mahabadi, Hamish Ivison, Jaesung Tae, James Henderson, Iz Beltagy, Matthew E Peters, and Arman Cohan. 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)*, pp. 2347–2361, 2024.

Shen Nie, Fengqi Zhu, Zebin You, Xiaolu Zhang, Jingyang Ou, Jun Hu, Jun Zhou, Yankai Lin, Ji-Rong Wen, and Chongxuan Li. Large language diffusion models. *Advances in Neural Information Processing Systems*, 38:50608–50646, 2026.

Avinash Patil and Aryan Jadon. Advancing reasoning in large language models: Promising methods and approaches. In *International Conference on Computational Intelligence and Soft Computing*, pp. 284–298. Springer, 2025.

Peter Potaptchik, Jason Yim, Adhi Saravanan, Peter Holderrieth, Eric Vanden-Eijnden, and Michael S. Albergo. Discrete flow maps, 2026. URL https://arxiv.org/abs/2604.09784.

Daan Roos, Oscar Davis, Floor Eijkelboom, Michael Bronstein, Max Welling, İsmail İlkan Ceylan, Luca Ambrogioni, and Jan-Willem van de Meent. Categorical Flow Maps, February 2026. URL http://arxiv.org/abs/2602.12233. arXiv:2602.12233 [cs.LG].

Subham S Sahoo, Marianne Arriola, Yair Schiff, Aaron Gokaslan, Edgar Marroquin, Justin T Chiu, Alexander Rush, and Volodymyr Kuleshov. Simple and effective masked diffusion language models. *Advances in Neural Information Processing Systems*, 37:130136–130184, 2024.

Jaesung Tae, Hamish Ivison, Sachin Kumar, and Arman Cohan. 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)*, pp. 21171–21188, 2025.

Guan Wang, Jin Li, Yuhao Sun, Xing Chen, Changling Liu, Yue Wu, Meng Lu, Sen Song, and Yasin Abbasi Yadkori. Hierarchical reasoning model, 2025. URL https://arxiv.org/abs/2506.21734.

Guanghan Wang, Yair Schiff, Subham Sahoo, and Volodymyr Kuleshov. Remasking discrete diffusion models with inference-time scaling. *Advances in Neural Information Processing Systems*, 38: 147282–147339, 2026.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

Fengli Xu, Qianyue Hao, Chenyang Shao, Zefang Zong, Yu Li, Jingwei Wang, Yunke Zhang, Jingyi Wang, Xiaochong Lan, Jiahui Gong, et al. Toward large reasoning models: A survey of reinforced reasoning with large language models. *Patterns*, 6(10), 2025.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. *Advances in neural information processing systems*, 36:11809–11822, 2023.

Jiacheng Ye, Zhihui Xie, Lin Zheng, Jiahui Gao, Zirui Wu, Xin Jiang, Zhenguo Li, and Lingpeng Kong. Dream 7b: Diffusion large language models. *arXiv preprint arXiv:2508.15487*, 2025.

Tao Zhang, Jia-Shu Pan, Ruiqi Feng, and Tailin Wu. Vfscale: Intrinsic reasoning through verifier-free test-time scalable diffusion model. In *International Conference on Learning Representations*, volume 2026, pp. 59758–59787, 2026.

Hanlin Zhu, Shibo Hao, Zhiting Hu, Jiantao Jiao, Stuart J Russell, and Yuandong Tian. Reasoning by superposition: A theoretical perspective on chain of continuous thought. *Advances in Neural Information Processing Systems*, 38:79931–79963, 2026.

APPENDIX OVERVIEW

The appendix provides training and sampling algorithms, theoretical proofs, and additional experimental details and results.

A Training and Sampling Algorithms

B Additional Related Work

C Theoretical Results

- C.1 Proof of Theorem 1
  - C.1.1 Candidate Sequences as Discrete Vertices
  - C.1.2 Geometry of Multi-Candidate Superposition
  - C.1.3 Accessibility of the Complete Graph
  - C.1.4 Proof of Instantaneous Candidate-Sequence Superposition
  - C.1.5 Derivation of the FLM ODE
  - C.1.6 Integral Representation and Proof of the Main Theorem
- C.2 Proof of Theorem 2
- C.3 Historical persistence and continuous refinement
- C.4 Proof of Theorem 3

D Experiments

- D.1 Dataset Construction and Processing
  - D.1.1 Maze Datasets
  - D.1.2 Sudoku Datasets
- D.2 Implementation Details
  - D.2.1 Architecture and Optimization Hyperparameters
  - D.2.2 Inference Details
  - D.2.3 FLM decoder experiments' implementation details
- D.3 Full-state interventions on retained historical support
- D.4 Complete Results across Model Capacities
- D.5 Wall-Clock Inference Efficiency

Algorithm 1 Training the conditional flow matching model

Require: Dataset $\mathcal{D}$, tokenizer tok, model $\boldsymbol{P}_\theta$

repeat
  Sample $(\mathcal{G}, \boldsymbol{q}, Y) \sim \mathcal{D}$
  $\mathbf{c} \leftarrow \mathrm{tok}(\mathrm{ser}_{\mathrm{in}}(\mathcal{G}, \boldsymbol{q})) \,\|\, [\mathrm{SEP}]$
  $\mathbf{x} \leftarrow \mathrm{tok}(\mathrm{ser}_{\mathrm{out}}(Y)) \,\|\, [\mathrm{EOS}]$
  $\mathbf{C}_{\mathcal{G},\boldsymbol{q}} \leftarrow \mathrm{OH}(\mathbf{c})$, $\boldsymbol{Z}_1 \leftarrow \mathrm{OH}(\mathbf{x})$
  Sample $\boldsymbol{Z}_0 \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$ and $t \sim \mathrm{Unif}[0,1]$
  $\boldsymbol{Z}_t \leftarrow (1-t)\boldsymbol{Z}_0 + t\boldsymbol{Z}_1$
  $\boldsymbol{P} \leftarrow \boldsymbol{P}_\theta(\boldsymbol{Z}_t, t, \mathbf{C}_{\mathcal{G},\boldsymbol{q}})$
  Update θ using $\mathrm{CE}(\boldsymbol{Z}_1, \boldsymbol{P})$ over answer positions
until convergence

Algorithm 2 Sampling from the conditional flow matching model

Require: Graph $\mathcal{G}$, query $\boldsymbol{q}$, tokenizer tok, model $\boldsymbol{P}_\theta$, answer length M_x, number of evaluations N

$\mathbf{c} \leftarrow \mathrm{tok}(\mathrm{ser}_{\mathrm{in}}(\mathcal{G}, \boldsymbol{q})) \,\|\, [\mathrm{SEP}]$
$\mathbf{C}_{\mathcal{G},\boldsymbol{q}} \leftarrow \mathrm{OH}(\mathbf{c})$
Sample $\widehat{\boldsymbol{Z}}_{t_0} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$ in $\mathbb{R}^{M_x \times K}$
Set $t_k = k/N$ for $k = 0, \ldots, N$
for $k = 0, \ldots, N-1$ **do**
  $\boldsymbol{P}_k \leftarrow \boldsymbol{P}_\theta(\widehat{\boldsymbol{Z}}_{t_k}, t_k, \mathbf{C}_{\mathcal{G},\boldsymbol{q}})$
  $\widehat{\boldsymbol{v}}_{t_k} \leftarrow (\boldsymbol{P}_k - \widehat{\boldsymbol{Z}}_{t_k})/(1-t_k)$
  $\widehat{\boldsymbol{Z}}_{t_{k+1}} \leftarrow \widehat{\boldsymbol{Z}}_{t_k} + (t_{k+1} - t_k)\widehat{\boldsymbol{v}}_{t_k}$
end for
$\widehat{x}_i \leftarrow \arg\max_{a \in \mathcal{V}_{\mathrm{tok}}} (\widehat{\boldsymbol{Z}}_{t_N})_{i,a}$ for $i = 1, \ldots, M_x$
$\widehat{s}_{\mathrm{out}} \leftarrow \mathrm{detok}(\widehat{x}_1, \ldots, \widehat{x}_{M_x})$
return $\widehat{Y} \leftarrow \mathrm{parse}_{\mathrm{path}}(\widehat{s}_{\mathrm{out}})$

A Training and Sampling Algorithms

Algorithms 1 and 2 summarize the training and sampling procedures used by our conditional FLM. Algorithm 1 constructs a continuous noisy answer state by interpolating between Gaussian noise and the clean tokenized target, and trains the model to predict the clean answer tokens. At inference time, Algorithm 2 starts from Gaussian noise, evolves the continuous answer state using the learned flow for N denoising evaluations, and discretizes the final state into a token sequence and the corresponding path.

B Additional Related Work

Discrete diffusion language models. Discrete diffusion language models have advanced through improved corruption processes, training objectives, and sampling strategies. D3PM (Austin et al., 2021) and SEDD (Lou et al., 2023) introduced discrete transition and score-based formulations, while MDLM (Sahoo et al., 2024) simplified masked-denoising training. LLaDA (Nie et al., 2026) and Dream (Ye et al., 2025) demonstrated competitive language modeling at scale. Sampling strategies based on confidence or entropy guide token selection, while ReMDM (Wang et al., 2026) allows previously generated tokens to be remasked and resampled. These methods perform denoising through categorical intermediate states.

C THEORETICAL RESULTS

C.1 PROOF OF THEOREM 1

Lemma 1 (Instantaneous candidate-sequence superposition) *For each candidate token sequence* $\mathbf{s} \in \mathcal{V}_{\text{tok}}^{M_x}$*, define*

$$\pi_{\mathbf{s}}(t) = \prod_{j=1}^{M_x} (\boldsymbol{P}_t)_{j,s_j}. \tag{15}$$

We call $\mathbf{s}$ *a positive-weight candidate at time* t *if* $\pi_{\mathbf{s}}(t) > 0$*, and define the support of the candidate-weight distribution as*

$$\operatorname{supp}(\pi_t) = \{\mathbf{s} \in \mathcal{S}_{M_x} : \pi_{\mathbf{s}}(t) > 0\}. \tag{16}$$

Then

$$\pi_{\mathbf{s}}(t) \geq 0, \qquad \sum_{\mathbf{s} \in \mathcal{S}_{M_x}} \pi_{\mathbf{s}}(t) = 1, \tag{17}$$

and the following exact decomposition holds:

$$\boldsymbol{P}_t = \sum_{\mathbf{s} \in \mathcal{V}_{\text{tok}}^{M_x}} \pi_{\mathbf{s}}(t) \mathbf{E}^{(\mathbf{s})}. \tag{18}$$

Hence, the instantaneous clean prediction is a convex superposition of candidate token-sequence vertices. If two positive-weight candidates differ at one or more token positions, then $\boldsymbol{P}_t \notin \mathcal{H}$ *and the superposition is nontrivial.*

Lemma 1 is an instantaneous geometric statement about the clean prediction $\boldsymbol{P}_t$. The Theorem 1 establishes the stronger trajectory-level result: the FLM ODE integrates these instantaneous superpositions over time and retains them in its persistent continuous state. We address this question using the FLM dynamics. This appendix provides complete proofs of Lemma 1 and Theorem 1. We first represent complete hard-token sequences as discrete vertices and establish the geometry of their positive-weight superpositions. We then show that global attention provides every answer position with access to the complete input graph. Next, we prove that the position-wise clean prediction of FLM automatically admits a convex decomposition over candidate token sequences. Finally, we derive the FLM ODE and its integral representation, prove the main theorem, and compare the resulting continuous state with the persistent hard state of absorbing discrete diffusion.

C.1.1 CANDIDATE SEQUENCES AS DISCRETE VERTICES

Let the answer contain M_x token positions, and let the tokenizer vocabulary $\mathcal{V}_{\text{tok}}$ have size K. A candidate token sequence

$$\mathbf{s} = (s_1, \ldots, s_{M_x}) \in \mathcal{V}_{\text{tok}}^{M_x}$$

corresponds to the one-hot matrix

$$\mathbf{E}^{(\mathbf{s})} = \begin{bmatrix} \boldsymbol{e}_{s_1}^\top \\ \vdots \\ \boldsymbol{e}_{s_{M_x}}^\top \end{bmatrix} \in \{0,1\}^{M_x \times K}. \tag{19}$$

Each row of $\mathbf{E}^{(\mathbf{s})}$ contains exactly one entry equal to one and all remaining entries equal to zero. Define the set of complete hard-token sequence states as

$$\mathcal{H} = \left\{ \mathbf{E}^{(\mathbf{s})} : \mathbf{s} \in \mathcal{V}_{\text{tok}}^{M_x} \right\}. \tag{20}$$

Also define the product simplex

$$\mathcal{Q} = (\Delta^{K-1})^{M_x} = \left\{ \mathbf{P} \in \mathbb{R}^{M_x \times K} : P_{j,a} \geq 0, \quad \sum_{a=1}^{K} P_{j,a} = 1 \text{ for every } j \right\}. \tag{21}$$

The set $\mathcal{H}$ is precisely the vertex set of $\mathcal{Q}$, and

$$\mathcal{Q} = \operatorname{conv}(\mathcal{H}). \tag{22}$$

The proof of Lemma 1 below also proves Eq. equation 22 by giving an explicit convex decomposition for every $\mathbf{P} \in \mathcal{Q}$. Strictly speaking, $\mathbf{E}^{(\mathbf{s})}$ is a vertex of the product simplex $\mathcal{Q}$, rather than a vertex of the unconstrained vector space $\mathbb{R}^{M_x \times K}$. A complete discrete hard-token state can take only an element of $\mathcal{H}$. Following the notation of Section 2, the FLM state $\boldsymbol{Z}_t \in \mathbb{R}^{M_x \times K}$ lies in the ambient continuous space; the denoised component defined below lies in $\mathcal{Q}$.

C.1.2 Geometry of Multi-Candidate Superposition

Consider $R \geq 2$ distinct candidate vertices $\mathbf{E}^{(1)}, \ldots, \mathbf{E}^{(R)} \in \mathcal{H}$ with strictly positive weights

$$\pi_m > 0, \qquad \sum_{m=1}^{R} \pi_m = 1. \tag{23}$$

Define their continuous superposition as

$$\mathbf{S}(\boldsymbol{\pi}) = \sum_{m=1}^{R} \pi_m \mathbf{E}^{(m)}. \tag{24}$$

Lemma 2 (Nontriviality of a multi-candidate convex combination) *The state $\mathbf{S}(\boldsymbol{\pi})$ belongs to the convex hull of the candidate vertices:*

$$\mathbf{S}(\boldsymbol{\pi}) \in \operatorname{conv}\{\mathbf{E}^{(1)}, \ldots, \mathbf{E}^{(R)}\} \subseteq \mathbb{R}^{M_x \times K}. \tag{25}$$

If two positive-weight candidates differ at one or more token positions, then

$$\mathbf{S}(\boldsymbol{\pi}) \notin \mathcal{H}. \tag{26}$$

Proof. Equation equation 25 follows directly from the definition of a convex hull. Suppose candidates r and s select different tokens at position j:

$$s_j^{(r)} = a, \qquad s_j^{(s)} = b, \qquad a \neq b,$$

with $\pi_r, \pi_s > 0$. Then

$$\mathbf{S}(\boldsymbol{\pi})_{j,a} \geq \pi_r > 0, \qquad \mathbf{S}(\boldsymbol{\pi})_{j,b} \geq \pi_s > 0. \tag{27}$$

Thus, row j of $\mathbf{S}(\boldsymbol{\pi})$ contains at least two positive coordinates and is not one-hot. Since every row of every state in $\mathcal{H}$ must be one-hot, $\mathbf{S}(\boldsymbol{\pi}) \notin \mathcal{H}$. □

Geometrically, the convex hull of two candidate vertices is a line segment, the convex hull of three affinely independent candidate vertices is a triangle, and additional affinely independent vertices generate a higher-dimensional simplex. If the selected candidate vertices are affinely independent and all their weights are strictly positive, the superposition lies in the relative interior of that simplex. If the vertices are not affinely independent, their positive-weight combination still lies in their convex hull and, whenever the candidates disagree at a token position, is not a hard vertex. It need not, however, lie in the relative interior of the convex hull generated by every listed vertex.

C.1.3 Accessibility of the Complete Graph

For a graph node v, let $\mathcal{I}_v$ denote the token positions associated with v in the conditioning sequence. Visibility of the complete graph means that

$$\mathcal{I}_v \neq \varnothing, \qquad \forall v \in \mathcal{V}. \tag{28}$$

In attention layer ℓ and head h, let the attention weight from answer position $\bar{j}$ to an unmasked position i be

$$A_{\bar{j}i}^{(\ell,h)}(t) = \frac{\exp(\ell_{\bar{j}i}^{(\ell,h)}(t))}{\sum_{r \in \mathcal{I}_{\text{real}}} \exp(\ell_{\bar{j}r}^{(\ell,h)}(t))}. \tag{29}$$

Lemma 3 (Global attention covers every input node) *Suppose that all attention logits are finite, attention dropout is disabled at inference time, and only padding positions are masked. Then, for every $i \in \mathcal{I}_v$,*

$$A_{\bar{j}i}^{(\ell,h)}(t) > 0. \tag{30}$$

Consequently, every answer position has an attention route to every graph node.

Proof. The exponential of every finite logit is strictly positive, and the denominator of Eq. equation 29 is a finite sum of strictly positive terms. Hence, every unmasked position receives a strictly positive attention weight. Because $\mathcal{I}_v$ is nonempty for every $v \in \mathcal{V}$, each answer position has a positive-attention route to every graph node. □

This lemma establishes only structural accessibility: the model can use the complete graph to form candidate evidence. It does not guarantee that the learned network uses every node, nor does it guarantee that a candidate token sequence represents a valid graph path.

C.1.4 Proof of Instantaneous Candidate-Sequence Superposition

Along the FLM trajectory, let

$$\boldsymbol{P}_t = \boldsymbol{P}_\theta(\boldsymbol{Z}_t, t, \mathcal{G}, \boldsymbol{q}) \in (\Delta^{K-1})^{M_x} \tag{31}$$

denote the position-wise clean-token distribution. For every $\mathbf{s} = (s_1, \ldots, s_{M_x}) \in \mathcal{V}_{\text{tok}}^{M_x}$, define

$$\pi_{\mathbf{s}}(t) = \prod_{j=1}^{M_x} (\boldsymbol{P}_t)_{j,s_j}. \tag{32}$$

Proof of Lemma 1. Since every row of $\boldsymbol{P}_t$ is a probability distribution,

$$\begin{aligned}\sum_{\mathbf{s}} \pi_{\mathbf{s}}(t) &= \sum_{s_1,\ldots,s_{M_x}} \prod_{j=1}^{M_x} (\boldsymbol{P}_t)_{j,s_j} \\ &= \prod_{j=1}^{M_x} \sum_{a \in \mathcal{V}_{\text{tok}}} (\boldsymbol{P}_t)_{j,a} \\ &= 1.\end{aligned} \tag{33}$$

Now consider coordinate (j, a) on the right-hand side of Eq. equation 3.1:

$$\begin{aligned}&\sum_{\mathbf{s}} \pi_{\mathbf{s}}(t) (\mathbf{E}^{(\mathbf{s})})_{j,a} \\ &\quad= \sum_{\mathbf{s}: s_j = a} \prod_{r=1}^{M_x} (\boldsymbol{P}_t)_{r,s_r} \\ &\quad= (\boldsymbol{P}_t)_{j,a} \prod_{r \neq j} \sum_{b \in \mathcal{V}_{\text{tok}}} (\boldsymbol{P}_t)_{r,b} \\ &\quad= (\boldsymbol{P}_t)_{j,a}.\end{aligned} \tag{34}$$

The equality holds for every j and a, which proves Eq. equation 3.1.

If two positive-weight candidates $\mathbf{s} \neq \mathbf{s}'$ differ at position j, let $s_j = a \neq b = s'_j$. The decomposition gives

$$(\boldsymbol{P}_t)_{j,a} \geq \pi_{\mathbf{s}}(t) > 0, \qquad (\boldsymbol{P}_t)_{j,b} \geq \pi_{\mathbf{s}'}(t) > 0. \tag{35}$$

Thus, row j of $\boldsymbol{P}_t$ is not one-hot, and hence $\boldsymbol{P}_t \notin \mathcal{H}$. This proves the nontrivial-superposition statement. □

Equation equation 32 constructs a factorized candidate-sequence distribution induced by the position-wise token marginals. It proves that $\boldsymbol{P}_t$ automatically belongs to the convex hull of all candidate token-sequence vertices. It does not imply that these sequences are valid graph paths, nor that this factorized distribution captures the general joint dependencies of complete paths. Because the same coordinate-wise construction applies to every $\mathbf{P} \in \mathcal{Q}$, it also proves Eq. equation 22.

C.1.5 Derivation of the FLM ODE

Training uses the linear probability path

$$\boldsymbol{Z}_t = (1-t)\boldsymbol{Z}_0 + t\boldsymbol{Z}_1, \tag{36}$$

where $\boldsymbol{Z}_0$ is Gaussian noise and $\boldsymbol{Z}_1$ is the clean one-hot answer state. Differentiating with respect to t gives

$$\frac{d\boldsymbol{Z}_t}{dt} = \boldsymbol{Z}_1 - \boldsymbol{Z}_0. \tag{37}$$

Rearranging Eq. equation 36 yields

$$\boldsymbol{Z}_0 = \frac{\boldsymbol{Z}_t - t\boldsymbol{Z}_1}{1-t}, \qquad t < 1. \tag{38}$$

Substituting Eq. equation 38 into Eq. equation 37, we obtain

$$\begin{aligned}\frac{d\boldsymbol{Z}_t}{dt} &= \boldsymbol{Z}_1 - \frac{\boldsymbol{Z}_t - t\boldsymbol{Z}_1}{1-t} \\ &= \frac{\boldsymbol{Z}_1 - \boldsymbol{Z}_t}{1-t}.\end{aligned} \tag{39}$$

At sampling time, the true $\boldsymbol{Z}_1$ is unknown. The model therefore replaces it with the clean-state prediction $\boldsymbol{P}_t$, defining the learned velocity field

$$\mathbf{F}_\theta(\boldsymbol{Z}_t, t, \mathbf{C}_{\mathcal{G},\boldsymbol{q}}) = \frac{\boldsymbol{P}_t - \boldsymbol{Z}_t}{1-t}, \tag{40}$$

and the sampling ODE

$$\frac{d\boldsymbol{Z}_t}{dt} = \frac{\boldsymbol{P}_t - \boldsymbol{Z}_t}{1-t}, \qquad 0 \le t < 1. \tag{41}$$

Under an ideal clean-state predictor, $\boldsymbol{P}_t$ approximates

$$\mathbb{E}[\boldsymbol{Z}_1 \mid \boldsymbol{Z}_t, t, \mathcal{G}, \boldsymbol{q}],$$

so Eq. equation 40 corresponds to the conditional mean velocity. For a learned model, Eq. equation 40 is the continuous velocity field defined by its clean-state predictor. Since the denominator vanishes at $t = 1$, the following dynamics and integral identities are stated for $t < 1$.

C.1.6 Integral Representation and Proof of the Main Theorem

Rewrite Eq. equation 41 as

$$\frac{d\boldsymbol{Z}_t}{dt} + \frac{\boldsymbol{Z}_t}{1-t} = \frac{\boldsymbol{P}_t}{1-t}. \tag{42}$$

This is a first-order linear identity along the realized trajectory. Although $\boldsymbol{P}_t$ depends on $\boldsymbol{Z}_t$, the following integral identity remains exact along every solution of the ODE. The integrating factor is

$$\mu(t) = \exp\left(\int \frac{1}{1-t}\, dt\right) = \frac{1}{1-t}. \tag{43}$$

Multiplying Eq. equation 42 by $\mu(t)$ gives

$$\frac{1}{1-t}\frac{d\boldsymbol{Z}_t}{dt} + \frac{\boldsymbol{Z}_t}{(1-t)^2} = \frac{\boldsymbol{P}_t}{(1-t)^2}. \tag{44}$$

By the product rule,

$$\frac{d}{dt}\left(\frac{\boldsymbol{Z}_t}{1-t}\right) = \frac{1}{1-t}\frac{d\boldsymbol{Z}_t}{dt} + \frac{\boldsymbol{Z}_t}{(1-t)^2}.$$

Therefore,

$$\frac{d}{dt}\left(\frac{\boldsymbol{Z}_t}{1-t}\right) = \frac{\boldsymbol{P}_t}{(1-t)^2}. \tag{45}$$

Integrating from 0 to t and applying the fundamental theorem of calculus gives

$$\frac{\boldsymbol{Z}_t}{1-t} - \boldsymbol{Z}_0 = \int_0^t \frac{\boldsymbol{P}_u}{(1-u)^2}\, du. \tag{46}$$

Multiplying by $1-t$ yields

$$\boldsymbol{Z}_t = (1-t)\boldsymbol{Z}_0 + (1-t)\int_0^t \frac{\boldsymbol{P}_u}{(1-u)^2}\, du. \tag{47}$$

Equation equation 47 decomposes the current state into a residual contribution from the initial noise and an accumulated contribution from the preceding clean predictions. Even though $\boldsymbol{P}_u$ depends on $\boldsymbol{Z}_u$ through the neural network, Eq. equation 47 is an exact integral identity along the solution trajectory.

Proof of Theorem 1. Apply Lemma 1 at time u and substitute its decomposition

$$\boldsymbol{P}_u = \sum_{\mathbf{s}} \pi_{\mathbf{s}}(u)\mathbf{E}^{(\mathbf{s})}, \tag{48}$$

into Eq. equation 47:

$$\begin{aligned}\boldsymbol{Z}_t &= (1-t)\boldsymbol{Z}_0 \\ &\quad + \sum_{\mathbf{s}} \left[(1-t)\int_0^t \frac{\pi_{\mathbf{s}}(u)}{(1-u)^2}\,du\right]\mathbf{E}^{(\mathbf{s})}.\end{aligned} \tag{49}$$

Define

$$\lambda_{\mathbf{s}}(t) = (1-t)\int_0^t \frac{\pi_{\mathbf{s}}(u)}{(1-u)^2}\,du. \tag{50}$$

Because $\pi_{\mathbf{s}}(u) \geq 0$ and the integral kernel is strictly positive, $\lambda_{\mathbf{s}}(t) \geq 0$. Furthermore, since $\sum_{\mathbf{s}} \pi_{\mathbf{s}}(u) = 1$,

$$\begin{aligned}\sum_{\mathbf{s}} \lambda_{\mathbf{s}}(t) &= (1-t)\int_0^t \frac{1}{(1-u)^2}\,du \\ &= (1-t)\left(\frac{1}{1-t} - 1\right) \\ &= t.\end{aligned} \tag{51}$$

For $t > 0$, define

$$\overline{\pi}_{\mathbf{s}}(t) = \frac{\lambda_{\mathbf{s}}(t)}{t}. \tag{52}$$

Then $\overline{\pi}_{\mathbf{s}}(t) \geq 0$ and $\sum_{\mathbf{s}} \overline{\pi}_{\mathbf{s}}(t) = 1$. Hence,

$$\boldsymbol{Z}_t = (1-t)\boldsymbol{Z}_0 + t\sum_{\mathbf{s}} \overline{\pi}_{\mathbf{s}}(t)\mathbf{E}^{(\mathbf{s})}. \tag{53}$$

Defining

$$\overline{\mathbf{E}}_t = \sum_{\mathbf{s}} \overline{\pi}_{\mathbf{s}}(t)\mathbf{E}^{(\mathbf{s})} \tag{54}$$

gives the decomposition stated in the theorem.

Suppose that two distinct candidates $\mathbf{s} \neq \mathbf{s}'$ have positive weights on subsets of time of positive measure. Because the integral kernel is strictly positive,

$$\overline{\pi}_{\mathbf{s}}(t) > 0, \qquad \overline{\pi}_{\mathbf{s}'}(t) > 0. \tag{55}$$

Distinct sequences differ at some position j. Let $s_j = a \neq b = s'_j$. Then

$$(\overline{\mathbf{E}}_t)_{j,a} \geq \overline{\pi}_{\mathbf{s}}(t) > 0, \qquad (\overline{\mathbf{E}}_t)_{j,b} \geq \overline{\pi}_{\mathbf{s}'}(t) > 0. \tag{56}$$

Thus, row j of $\overline{\mathbf{E}}_t$ is not one-hot. By Lemma 2,

$$\overline{\mathbf{E}}_t \notin \mathcal{H}. \tag{57}$$

Therefore, contributions from multiple candidate sequences are accumulated through the same FLM ODE trajectory into a nontrivial geometric superposition. □

C.2 Proof of Theorem 2

Proof. Throughout, the context, time t, model parameters, and attention mask are fixed. All derivatives are with respect to λ. We suppress the dependence on λ where convenient.

Hard-state preservation. The intervention modifies only coordinate (j, v). Since v is a strict non-top-1 candidate and $0 \le \lambda \le h_t$,

$$(\mathbf{U}_t(\lambda))_{j,v} = (\boldsymbol{Z}_t)_{j,v} - (h_t - \lambda) \le (\boldsymbol{Z}_t)_{j,v} < (\boldsymbol{Z}_t)_{j,x_j}.$$

Every other coordinate remains unchanged. Hence the set of maximizing coordinates in each row is unchanged, and fixed tie-breaking gives

$$\Pi(\mathbf{U}_t(\lambda)) = \Pi(\boldsymbol{Z}_t) = \mathbf{x}.$$

This proves the first equality in Equation 8.

Exact derivative decomposition. For head b and receiving position i, let $\mathcal{I}_i^b$ be the fixed set of visible positions. Define

$$s_{im}^b = \frac{\langle \mathbf{q}_i^b, \mathbf{k}_m^b \rangle}{\sqrt{d_b}}, \qquad a_{im}^b = \frac{\exp(s_{im}^b)}{\sum_{\ell \in \mathcal{I}_i^b} \exp(s_{i\ell}^b)}, \qquad m \in \mathcal{I}_i^b.$$

The product rule yields

$$\dot{s}_{im}^b = \frac{\langle \dot{\mathbf{q}}_i^b, \mathbf{k}_m^b \rangle}{\sqrt{d_b}} + \frac{\langle \mathbf{q}_i^b, \dot{\mathbf{k}}_m^b \rangle}{\sqrt{d_b}} = \rho_{im}^{Q,b} + \rho_{im}^{K,b}.$$

Differentiating the softmax gives

$$\dot{a}_{im}^b = a_{im}^b \left(\dot{s}_{im}^b - \sum_{\ell \in \mathcal{I}_i^b} a_{i\ell}^b \dot{s}_{i\ell}^b \right). \tag{58}$$

Since $\mathbf{o}_i^b = \sum_{m \in \mathcal{I}_i^b} a_{im}^b \mathbf{v}_m^b$,

$$\dot{\mathbf{o}}_i^b = \sum_{m \in \mathcal{I}_i^b} \dot{a}_{im}^b \mathbf{v}_m^b + \sum_{m \in \mathcal{I}_i^b} a_{im}^b \dot{\mathbf{v}}_m^b. \tag{59}$$

By the complete interface representation $M_t^{(\lambda)} = F_t(O(\lambda), \xi(\lambda))$, the chain rule gives

$$\partial_\lambda M_t^{(\lambda)} = \sum_{b,i} \langle \mathbf{g}_i^b, \dot{\mathbf{o}}_i^b \rangle + \langle \nabla_\xi F_t, \dot{\xi} \rangle. \tag{60}$$

Here all gradients are evaluated at the current intervened state, and include the dependence through all subsequent model layers.

Using $\eta_{im}^b = \langle \mathbf{g}_i^b, \mathbf{v}_m^b \rangle$, the attention-weight contribution at each head and position is

$$\begin{aligned} \sum_{m \in \mathcal{I}_i^b} \dot{a}_{im}^b \eta_{im}^b &= \mathbb{E}_{m \sim a_i^b}[\dot{s}_{im}^b \eta_{im}^b] - \mathbb{E}_{m \sim a_i^b}[\dot{s}_{im}^b]\, \mathbb{E}_{m \sim a_i^b}[\eta_{im}^b] \\ &= \mathrm{Cov}_{m \sim a_i^b}(\rho_{im}^{Q,b}, \eta_{im}^b) + \mathrm{Cov}_{m \sim a_i^b}(\rho_{im}^{K,b}, \eta_{im}^b). \end{aligned}$$

The value contribution is

$$\left\langle \mathbf{g}_i^b, \sum_{m \in \mathcal{I}_i^b} a_{im}^b \dot{\mathbf{v}}_m^b \right\rangle = \mathbb{E}_{m \sim a_i^b} \langle \mathbf{g}_i^b, \dot{\mathbf{v}}_m^b \rangle.$$

Substituting these identities into Equation 60 and summing over all heads and receiving positions gives

$$\partial_\lambda M_t^{(\lambda)} = \mathcal{C}_Q + \mathcal{C}_K + \mathcal{C}_V + \mathcal{C}_R = \mathcal{C}(\lambda),$$

with the contributions stated in Equation 9. This proves the second equality in Equation 8. No sign restriction on the individual contributions is required.

Cross-position influence. Suppose the maps preceding attention act independently at each position. Because the intervention changes only position j, for $i \neq j$ we have

$$\dot{\mathbf{q}}_i^b = 0, \qquad \dot{\mathbf{k}}_m^b = \dot{\mathbf{v}}_m^b = 0 \quad \text{for } m \neq j.$$

For $j \in \mathcal{I}_i^b$, it follows that

$$\dot{s}_{im}^b = \mathbf{1}\{m = j\}\rho_{ij}^{K,b}.$$

Equation 58 therefore becomes

$$\dot{a}_{im}^b = a_{im}^b\big(\mathbf{1}\{m = j\} - a_{ij}^b\big)\rho_{ij}^{K,b}.$$

Substitution into Equation 59 yields

$$\begin{aligned}\dot{\mathbf{o}}_i^b &= a_{ij}^b \mathbf{v}_j^b \rho_{ij}^{K,b} - a_{ij}^b \rho_{ij}^{K,b} \sum_{m \in \mathcal{I}_i^b} a_{im}^b \mathbf{v}_m^b + a_{ij}^b \dot{\mathbf{v}}_j^b \\ &= a_{ij}^b (\mathbf{v}_j^b - \mathbf{o}_i^b)\rho_{ij}^{K,b} + a_{ij}^b \dot{\mathbf{v}}_j^b,\end{aligned}$$

which is Equation 10.

Local sensitivity and the integrated response. The smoothness assumption implies that $\lambda \mapsto M_t^{(\lambda)}$ is C^1, with continuous derivative $\mathcal{C}(\lambda)$. Thus, by the fundamental theorem of calculus,

$$M_t^{(\lambda_1)} - M_t^{(\lambda_0)} = \int_{\lambda_0}^{\lambda_1} \mathcal{C}(s)\, ds \qquad (\lambda_0, \lambda_1 \in [0, h_t]).$$

Taking $\lambda_0 = 0$ and $\lambda_1 = h_t$ proves Equation 11.

If $\mathcal{C}(\lambda_*) \neq 0$, continuity ensures that $\mathcal{C}$ has the same sign as $\mathcal{C}(\lambda_*)$ and magnitude at least $|\mathcal{C}(\lambda_*)|/2$ in a sufficiently small neighborhood of λ_*. Consequently, every sufficiently small nonzero δ with $\lambda_* + \delta \in [0, h_t]$ satisfies

$$\left|M_t^{(\lambda_*+\delta)} - M_t^{(\lambda_*)}\right| = \left|\int_{\lambda_*}^{\lambda_*+\delta} \mathcal{C}(s)\, ds\right| \geq \frac{|\mathcal{C}(\lambda_*)|}{2}|\delta| > 0.$$

The argument also covers one-sided admissible perturbations at the endpoints. Together with hard-state preservation, this proves the claimed local prediction sensitivity at fixed hard state.

Hard-state approximation error and tightness. Continuity on the compact interval $[0, h_t]$ ensures that

$$M_{\min} := \min_{\lambda \in [0,h_t]} M_t^{(\lambda)}, \qquad M_{\max} := \max_{\lambda \in [0,h_t]} M_t^{(\lambda)}$$

are attained, with $\Omega_t = M_{\max} - M_{\min}$. Since $h_t > 0$ and $M_t^{(\lambda)}$ is C^1, it is constant on this interval if and only if $\mathcal{C} \equiv 0$. Hence

$$\Omega_t > 0 \quad \Longleftrightarrow \quad \mathcal{C} \not\equiv 0.$$

For any scalar readout H using only context, time, and the hard state, its input is constant along the intervention path. Write

$$c_H := H(\mathbf{C}_{\mathcal{G},\boldsymbol{q}}, t, \mathbf{x}).$$

The triangle inequality gives

$$\begin{aligned}\Omega_t &= |M_{\max} - M_{\min}| \\ &\leq |M_{\max} - c_H| + |M_{\min} - c_H| \\ &\leq 2 \sup_{\lambda \in [0,h_t]} |M_t^{(\lambda)} - c_H|.\end{aligned}$$

Moreover, both endpoint values lie in $[M_{\min}, M_{\max}]$, so

$$|\Delta_t| = |M_t^{(h_t)} - M_t^{(0)}| \leq \Omega_t.$$

Combining these inequalities establishes Equation 12.

Finally, for this fixed intervention path, choose the constant readout

$$H_*(\cdot) = \frac{M_{\max} + M_{\min}}{2}.$$

Every path value lies between $M_{\min}$ and $M_{\max}$, and both extrema are attained. Therefore,

$$\sup_{\lambda \in [0, h_t]} \left| M_t^{(\lambda)} - H_*(\mathbf{C}_{\mathcal{G},\boldsymbol{q}}, t, \mathbf{x}) \right| = \frac{M_{\max} - M_{\min}}{2} = \frac{\Omega_t}{2}.$$

Thus the first lower bound is tight. □

C.3 Historical persistence and continuous refinement

We use the notation of Theorem 2, with $\boldsymbol{P}_t = \boldsymbol{P}_t^{(h_t)}$ and $\Delta_t = M_t - M_t^{(0)}$. Let $\mathcal{F}$ denote the free positions. Each row of $\boldsymbol{P}_t^{(\lambda)}$ is a probability vector, and conditioning coordinates remain fixed throughout the update.

Corollary 1 (Historical persistence and continuous refinement) *For a fixed coefficient $\alpha_t \in [0, 1]$, define*

$$\mathbf{U}_t^+(\lambda) = (1 - \alpha_t)\mathbf{U}_t(\lambda) + \alpha_t \boldsymbol{P}_t^{(\lambda)}$$

on free coordinates, and let

$$d_t^+(\lambda) = (\mathbf{U}_t^+(\lambda))_{j,v} - (\mathbf{U}_t^+(\lambda))_{j,x_j}.$$

Then

$$\mathbf{U}_t^+(h_t) - \mathbf{U}_t^+(0) = (1 - \alpha_t)h_t \mathbf{D}^{jv} + \alpha_t(\boldsymbol{P}_t - \boldsymbol{P}_t^{(0)}) \tag{61}$$

on free coordinates. Consequently,

$$d_t^+(h_t) - d_t^+(0) = \underbrace{(1 - \alpha_t)h_t}_{\textit{direct persistence}} + \underbrace{\alpha_t \Delta_t}_{\textit{predictor response}}. \tag{62}$$

Moreover, suppose the minimum winner gap is

$$\gamma_t := \min_{\substack{i \in \mathcal{F} \\ b \neq x_i}} \big((\boldsymbol{Z}_t)_{i,x_i} - (\boldsymbol{Z}_t)_{i,b} \big) > 0$$

and $0 < \alpha_t < \gamma_t/(1 + \gamma_t)$. Then

$$\Pi(\mathbf{U}_t^+(\lambda)) = \mathbf{x} \qquad (0 \le \lambda \le h_t). \tag{63}$$

Under the same winner-gap and step-size conditions, the natural update for $d_t = (\boldsymbol{Z}_t)_{j,v} - (\boldsymbol{Z}_t)_{j,x_j} < 0$ satisfies

$$\begin{aligned} d_t^+(h_t) - d_t &= \alpha_t(M_t - d_t), \\ M_t > d_t \quad &\Longleftrightarrow \quad d_t < d_t^+(h_t) < 0. \end{aligned} \tag{64}$$

Proof. By the intervention definition,

$$\mathbf{U}_t(h_t) = \boldsymbol{Z}_t, \qquad \mathbf{U}_t(0) = \boldsymbol{Z}_t - h_t \mathbf{D}^{jv}.$$

Subtracting the two Euler updates on free coordinates gives

$$\begin{aligned} \mathbf{U}_t^+(h_t) - \mathbf{U}_t^+(0) &= (1 - \alpha_t)\big(\mathbf{U}_t(h_t) - \mathbf{U}_t(0)\big) + \alpha_t(\boldsymbol{P}_t - \boldsymbol{P}_t^{(0)}) \\ &= (1 - \alpha_t)h_t \mathbf{D}^{jv} + \alpha_t(\boldsymbol{P}_t - \boldsymbol{P}_t^{(0)}), \end{aligned}$$

proving Equation 61. Since $v \neq x_j$, the matrix $\mathbf{D}^{jv}$ has entry 1 at (j, v) and 0 at (j, x_j). Taking the difference between these two coordinates yields

$$\begin{aligned} d_t^+(h_t) - d_t^+(0) &= (1 - \alpha_t)h_t + \alpha_t\big(M_t - M_t^{(0)}\big) \\ &= (1 - \alpha_t)h_t + \alpha_t \Delta_t, \end{aligned}$$

which proves Equation 62.

Next, the intervention only decreases a nonwinning coordinate. Thus, for every $i \in \mathcal{F}$, $b \neq x_i$, and $\lambda \in [0, h_t]$,

$$\begin{aligned}(\mathbf{U}_t(\lambda))_{i,x_i} - (\mathbf{U}_t(\lambda))_{i,b} &= (\boldsymbol{Z}_t)_{i,x_i} - (\boldsymbol{Z}_t)_{i,b} + (h_t - \lambda)\mathbf{1}\{i = j,\ b = v\} \\ &\geq \gamma_t.\end{aligned}$$

Because predictor rows are probability vectors,

$$(\boldsymbol{P}_t^{(\lambda)})_{i,x_i} - (\boldsymbol{P}_t^{(\lambda)})_{i,b} \geq -1.$$

Combining these inequalities gives

$$\begin{aligned}(\mathbf{U}_t^+(\lambda))_{i,x_i} - (\mathbf{U}_t^+(\lambda))_{i,b} &\geq (1 - \alpha_t)\gamma_t - \alpha_t \\ &= \gamma_t - \alpha_t(1 + \gamma_t) > 0.\end{aligned}$$

Hence x_i remains the unique winner at every free position. Conditioning coordinates are unchanged, so $\Pi(\mathbf{U}_t^+(\lambda)) = \mathbf{x}$ throughout the path, establishing Equation 63.

Finally, $\lambda = h_t$ corresponds to the original, unintervened state. Its updated margin is

$$d_t^+(h_t) = (1 - \alpha_t)d_t + \alpha_t M_t,$$

and therefore

$$d_t^+(h_t) - d_t = \alpha_t(M_t - d_t).$$

Under the stated conditions, $\alpha_t > 0$ and $d_t^+(h_t) < 0$ by hard-state preservation. Consequently,

$$M_t > d_t \quad \Longleftrightarrow \quad d_t^+(h_t) > d_t \quad \Longleftrightarrow \quad d_t < d_t^+(h_t) < 0,$$

proving Equation 64. □

Equation 62 compares the updated states with and without historical support, separating direct persistence from the model's prediction response. Equation 64 instead compares the original trajectory before and after one update: under the stated conditions, $M_t > d_t$ precisely means that the candidate's gap to the current winner shrinks while the hard-token sequence remains unchanged.

C.4 Proof of Theorem 3

We use the representations defined in the main text:

$$\begin{aligned}A_t &= (\mathbf{C}_{\mathcal{N}}, (\boldsymbol{Z}_t)_{\mathcal{N}}), \qquad & B_t &= (\mathbf{C}_{\mathcal{N}}, \mathsf{H}((\boldsymbol{Z}_t)_{\mathcal{N}})), \\ A_t^{\text{soft}} &= (\mathbf{C}_{\mathcal{N}}, \mathbf{W}_t), \qquad & A_t^{\text{conf}} &= (B_t, \max \mathbf{W}_t),\end{aligned}$$

where $\mathsf{H} = \mathrm{OH} \circ \Pi$, $\mathbf{W}_t = \mathrm{softmax}((\boldsymbol{Z}_t)_{\mathcal{N}})$, and all operations are rowwise. The receptive field $\mathcal{N}$ contains J as an identifiable center, and $Y = \mathbf{1}\{X_J = v_{\text{path}}\}$. Fix the trained decoders. All risks and expectations are taken over the same independent test distribution of puzzles, trajectory randomness, and uniformly sampled output positions. All logarithms are natural, and BCE is unweighted.

Proof. Step 1: Decompose the decoder risks. For any representation T, let $p_T = \Pr(Y = 1 \mid T)$ and let $f(T)$ be a decoder's predicted probability. Conditioning the BCE loss on T gives

$$\begin{aligned}R(f; T) &= \mathbb{E}[-Y \log f(T) - (1 - Y) \log(1 - f(T))] \\ &= H(Y \mid T) + \mathbb{E}\,\mathrm{KL}\big(\mathrm{Ber}(p_T) \,\|\, \mathrm{Ber}(f(T))\big).\end{aligned} \tag{65}$$

Thus $\varepsilon_t^Z, \varepsilon_t^{\text{hard}} \geq 0$. Since B_t is a deterministic function of A_t, the Bayes-risk gap $\mathcal{I}_t := H(Y \mid B_t) - H(Y \mid A_t)$ is nonnegative, and

$$\Delta_t^{\text{dec}} = \mathcal{I}_t + \varepsilon_t^{\text{hard}} - \varepsilon_t^Z. \tag{66}$$

Step 2: Lower-bound the signal lost under hard projection. Softmax preserves rowwise argmax, including ties under the fixed rule. Hence the representations generate nested sigma-algebras:

$$\sigma(B_t) \subseteq \sigma(A_t^{\text{conf}}) \subseteq \sigma(A_t^{\text{soft}}) \subseteq \sigma(A_t).$$

Set $p_s = \Pr(Y = 1 \mid A_t^{\text{soft}})$ and $p_c = \Pr(Y = 1 \mid A_t^{\text{conf}})$. The tower property gives $p_c = \mathbb{E}[p_s \mid A_t^{\text{conf}}]$, so

$$\nu_t = \mathbb{E}[(p_s - p_c)^2]. \tag{67}$$

Monotonicity of conditional entropy and the conditional Bernoulli KL identity yield

$$\begin{aligned} \mathcal{I}_t &\geq H(Y \mid A_t^{\text{conf}}) - H(Y \mid A_t^{\text{soft}}) \\ &= \mathbb{E}\,\mathrm{KL}\big(\mathrm{Ber}(p_s) \,\|\, \mathrm{Ber}(p_c)\big) \\ &\geq 2\mathbb{E}[(p_s - p_c)^2] = 2\nu_t. \end{aligned} \tag{68}$$

The last inequality follows from $\mathrm{KL}(\mathrm{Ber}(p)\|\,\mathrm{Ber}(q)) \geq 2(p-q)^2$ in natural logarithms. Indeed, for $q \in (0,1)$, the KL as a function of p has value and first derivative zero at $p = q$, and second derivative $1/[p(1-p)] \geq 4$; boundary cases follow by limits. Combining equation 68 with equation 66 gives

$$\Delta_t^{\text{dec}} \geq 2\nu_t + \varepsilon_t^{\text{hard}} - \varepsilon_t^Z \geq 2\nu_t - \varepsilon_t^Z.$$

Step 3: Upper-bound the remaining benefit by state error. Let $\widehat{X}_J = [\Pi(\boldsymbol{Z}_t)]_J$ and $\widehat{Y} = \mathbf{1}\{\widehat{X}_J = v_{\text{path}}\}$. Since the center J is identifiable, $\widehat{Y}$ is measurable from B_t. A binary-label error implies a token error. On the event $\widehat{X}_J \neq X_J$, write $\mathbf{z} = (\boldsymbol{Z}_t)_{J,:}$ and $w = \widehat{X}_J$. Then $z_w \geq z_{X_J}$, and

$$\begin{aligned} \|\mathbf{z} - \mathbf{e}_{X_J}^\top\|_2^2 &\geq (z_{X_J} - 1)^2 + z_w^2 \\ &\geq \tfrac{1}{2}(1 - z_{X_J} + z_w)^2 \geq \tfrac{1}{2}. \end{aligned} \tag{69}$$

Therefore

$$\begin{aligned} q_t := \Pr(\widehat{Y} \neq Y) &\leq \Pr(\widehat{X}_J \neq X_J) \\ &\leq 2\mathbb{E}\|(\boldsymbol{Z}_t)_{J,:} - \mathbf{e}_{X_J}^\top\|_2^2 \leq 2b_t^2. \end{aligned} \tag{70}$$

Let $E_t = \mathbf{1}\{\widehat{Y} \neq Y\}$. Conditional on B_t, Y and E_t determine one another because $\widehat{Y}$ is known. Consequently,

$$H(Y \mid B_t) = H(E_t \mid B_t) \leq H(E_t) = h_{\text{bin}}(q_t).$$

If $2b_t^2 \leq 1/2$, use the monotonicity of h_{bin} on $[0, 1/2]$; otherwise use $H(Y \mid B_t) \leq \log 2$. In both cases,

$$0 \leq H(Y \mid A_t) \leq H(Y \mid B_t) \leq \chi_t := h_{\text{bin}}(\min\{1/2, 2b_t^2\}). \tag{71}$$

Using equation 66 and nonnegativity of $H(Y \mid A_t)$ and ε_t^Z, we conclude that

$$\Delta_t^{\text{dec}} \leq H(Y \mid B_t) + \varepsilon_t^{\text{hard}} \leq \chi_t + \varepsilon_t^{\text{hard}}.$$

Finally,

$$R_t^Z = H(Y \mid A_t) + \varepsilon_t^Z \leq \chi_t + \varepsilon_t^Z.$$

Together with Step 2, these are all the asserted bounds. □

The proof does not assume that either trained decoder attains its Bayes risk. Matching architecture and training budget therefore does not remove the excess-risk terms.

D Experiments

D.1 Dataset Construction and Processing

D.1.1 Maze Datasets

Each of Maze10, Maze15, and Maze20 contains 10,000 training examples, and 1,000 test examples. The corresponding logical lattices are 10×10, 15×15, and 20×20, respectively. We follow the maze-generation setup of VFScale (Zhang et al., 2026).

For each size, we generate 12,000 candidate mazes with seed 42 and default generator parameters. Each maze is a tree spanning all lattice cells. We sample two distinct nodes uniformly without replacement as the start and goal, and obtain their unique path with the library's A* solver. We retain paths with at least five logical nodes, equivalent to four moves, without an upper length limit.

Dataset	Lattice	Raster	Tokens	Test path length
Maze10	10×10	21×21	884	26.44 ± 14.78
Maze15	15×15	31×31	1,924	50.74 ± 31.43
Maze20	20×20	41×41	3,364	79.85 ± 52.63

Table 4: Maze representation sizes and test-set path lengths (mean $\pm$ population standard deviation, in logical moves). Token counts include `[SEP]` and `[EOS]` and exclude padding and CoT actions.

We deduplicate by binary maze layout, excluding endpoint annotations, then shuffle with NumPy's `default_rng` using seed 123. The first 10,000 retained examples form the training set, followed by 1,000 for testing; remaining examples are unused. Each retained layout has one start–goal pair, and no data augmentation is applied. The fixed splits have no overlapping layouts.

An $n \times n$ logical maze is rasterized into a $(2n+1) \times (2n+1)$ board. Input symbols `0`, `1`, `2`, and `3` denote traversable pixels, walls, the start, and the goal, respectively. The target preserves the board and endpoints and marks intermediate solution pixels with `4`. Both boards are flattened in row-major order and serialized as `[input][SEP][output][EOS]` with character-level tokenization. The resulting length is $2(2n+1)^2 + 2$ tokens before padding. Table 4 summarizes the representation sizes and test-set path lengths. Path length counts logical moves; a path of length L contains $L+1$ logical nodes.

Ground-truth paths are deterministically converted to logical actions `5`, `6`, `7`, and `8` for up, down, left, and right, respectively, with one action per logical move. The sequence format is `[input][SEP][actions]=[output][EOS]`.

D.1.2 Sudoku Datasets

We use Sudoku-Extreme[1], a deduplicated collection of Sudoku puzzles with difficulty ratings based on solver backtracking counts. We take the first 1,000,000 examples for training, the following 1,000 for validation, and the next 1,000 for testing. No additional data augmentation is applied in our experiments.

Each 9×9 puzzle and its solution are flattened in row-major order into strings of length 81. In our representation, `0` denotes an empty cell and `1`–`9` denote filled cells. We serialize each pair as `[puzzle][SEP][solution][EOS]`. The puzzle remains clean conditioning, and only the solution span is generated. With character-level tokenization, each pair contains 164 tokens before padding. A correct completion must preserve the given clues and contain digits 1 through 9 exactly once in every row, column, and 3×3 sub-grid. Our primary metric is sequence accuracy, defined as exact agreement with the reference solution over all 81 cells. We also report per-cell accuracy over the full solution span.

To assess generalization across datasets, we directly evaluate models trained on Sudoku-Extreme on Standard Sudoku from the Kaggle collection[2], without additional fine-tuning.

D.2 Implementation Details

D.2.1 Architecture and Optimization Hyperparameters

FLM and the discrete diffusion baselines use GPT-2-style Transformers with bidirectional self-attention, pre-layer normalization, learned absolute positional embeddings, and GELU feed-forward layers. The feed-forward dimension is four times the hidden dimension. Time conditioning uses a 512-dimensional sinusoidal embedding followed by a three-layer MLP, whose output is added to the input representations. At each model size, FLM and the corresponding discrete baselines share the same backbone configuration.

Both Maze and Sudoku-Extreme use character-level tokenization with a vocabulary of 31 tokens, including special tokens. Each example is serialized as `[input][SEP][output][EOS]`. The

[1] Hugging Face: Sudoku-Extreme.

[2] Kaggle: 3 Million Sudoku Puzzles with Ratings.

Table 5: Backbone configurations for Maze and Sudoku-Extreme. FFN denotes the feed-forward dimension, and maximum length denotes the capacity of the learned positional embedding table. Parameter counts include the input and output projections, positional embeddings, and time-conditioning network. FLM and the discrete baselines have identical parameter counts when using the same configuration.

Dataset	Model	Hidden dim.	Heads	FFN dim.	Max. length	Parameters
Maze10	6 layers	768	6	3072	1024	44.94M
	8 layers	768	8	3072	1024	59.11M
	12 layers	768	12	3072	1024	87.47M
Maze15	8 layers	768	8	3072	2048	59.90M
	12 layers	768	12	3072	2048	88.25M
	14 layers	896	14	3584	2048	139.00M
Maze20	14 layers	896	14	3584	4096	140.83M
	20 layers	1024	16	4096	4096	258.81M
	24 layers	1024	16	4096	4096	309.19M
Sudoku-Extreme	12 layers	768	12	3072	256	86.88M

Table 6: Optimization settings for Maze and Sudoku-Extreme.

Hyperparameter	Maze	Sudoku-Extreme
Optimizer	AdamW	AdamW
AdamW (β_1, β_2)	$(0.9, 0.95)$	$(0.9, 0.95)$
Peak learning rate	4×10^{-4}	1×10^{-4}
Minimum learning rate	4×10^{-6}	1×10^{-6}
Learning-rate schedule	Cosine decay	Cosine decay
Linear warmup (optimizer updates)	125	500
Global batch size (default)	256	1024
Weight decay	0.01	0.01
Training dropout	0.1	0.1
Gradient clipping	1.0	1.0
EMA decay	0.9999	0.9999
FLM training budget (epochs)	1000	300
Discrete baseline training budget (epochs)	2000	300

input and separator remain fixed during training and sampling. The loss is computed only over output positions, including `[EOS]`, while padding positions are excluded. The unpadded sequence lengths are 884, 1,924, and 3,364 for Maze10, Maze15, and Maze20, respectively, and 164 for Sudoku-Extreme.

We optimize all models with AdamW (Loshchilov et al., 2017), using $\beta_1 = 0.9$, $\beta_2 = 0.95$, weight decay 0.01, and gradient clipping at a global norm of 1.0. The training budgets are 1,000 epochs for FLM and 2,000 epochs for the discrete baselines on each Maze dataset, and 300 epochs for all methods on Sudoku-Extreme. Table 6 summarizes the optimization settings. Reported batch sizes are global, accounting for all devices and gradient accumulation. All evaluations use exponential moving average (EMA) weights with decay 0.9999.

FLM is trained on linear interpolations between independent standard Gaussian noise and one-hot target tokens, with the input conditions held fixed. Interpolation times are sampled uniformly on $[0, 1]$ using antithetic pairs. We minimize clean-token cross-entropy over the supervised output positions. The model predicts a clean-token distribution at every time step, from which the continuous velocity field is constructed. D3PM, SEDD, and MDLM retain their respective discrete diffusion objectives: a variational-bound objective, score entropy, and masked denoising, respectively. Their corruption processes use an absorbing mask token and preserve the same input conditions.

D.2.2 Inference Details

We consider sampling budgets $K \in \{1, 2, 4, 8, 16, 32, 64, 128, 256, 512\}$. For FLM, K denotes the number of integration steps on the uniform grid $t_k = k/K$. Sampling starts from standard Gaussian noise at output positions. Given the predicted clean-token probabilities $p_\theta(z_t, t \mid c)$, where c denotes the fixed input conditions, intermediate Euler updates take the form

$$z_{t_{k+1}} = z_{t_k} + \frac{1}{K} \frac{p_\theta(z_{t_k}, t_k \mid c) - z_{t_k}}{1 - t_k + 10^{-5}}. \tag{72}$$

For the final interval, we directly set the state to the predicted clean-token distribution. Conditioning positions are restored after every update, and tokenwise argmax is applied to the final continuous state. This procedure requires exactly K model evaluations.

For the discrete baselines, output positions are initialized with the absorbing mask token. We use a uniform reverse-time grid from 1 to a terminal cutoff ϵ, with ancestral reverse updates for D3PM and MDLM and score-based reverse updates for SEDD. Categorical sampling uses temperature 1.0.

After the K reverse updates, the discrete implementations perform one additional noise-removal model evaluation. Thus, K counts reverse-process updates for these methods, and the corresponding number of model evaluations is $K + 1$. Input conditions remain fixed throughout sampling, and all methods use their EMA weights.

D.2.3 FLM decoder experiments' implementation details

Experimental setup. We freeze a Maze15 FLM with eight layers, eight attention heads, and hidden width 768, using EMA weights, and cache states $Z_0, \ldots, Z_8$ from one $N = 8$ trajectory per maze. Separate local CNN decoders predict the binary solution-path mask from the maze condition and each state representation at each step. All decoders share a 9×9 receptive field, 38,177 parameters, and a 480-update training budget. We use 512/128/128 disjoint training/validation/test mazes and three decoder initialization seeds, matching initialization and minibatch order across conditions. Training uses class-weighted BCE; evaluation reports unweighted BCE and path-F1 at a fixed probability threshold of 0.5.

D.3 Full-state interventions on retained historical support

Experimental setup. We test whether retained historical support affects the next prediction while the current hard-token sequence remains unchanged. For a strict non-top-1 candidate v at position j, we intervene along

$$\mathbf{U}_t(\lambda) = \boldsymbol{Z}_t - (h_t - \lambda)\mathbf{e}_j\mathbf{e}_v^\top, \qquad 0 \le \lambda \le h_t.$$

This removes only the historical contribution at (j, v), leaving context, time, other coordinates, and initial noise fixed. We recompute the complete model without renormalizing the state. The measured effect is $\Delta_t = M_t^{(h_t)} - M_t^{(0)}$, where $M_t^{(\lambda)}$ is the predicted probability of v minus that of the original winner x_j. Thus, $\Delta_t > 0$ means that retaining history increases the candidate's relative support.

The original cohort contains 898 fixed events from Sudoku and Maze20 trajectories with $N \in \{2, 4\}$, including 261 correct-candidate events (v equals the target token). We evaluate 17 points per path. This is a retrospective mechanism analysis. Reported means first average over noise seeds within each problem and then weight problems equally; nominal 95% confidence intervals use 20,000 problem-cluster bootstrap resamples without multiple-comparison correction.

Historical support changes predictions. Retaining history increases the correct-candidate margin in 232/261 original events, with positive means and confidence intervals excluding zero in all four groups (Table 7). An additional validation cohort contains 181 eligible events, including 48 correct-candidate events, of which 47 have positive endpoint effects. These inputs differ from earlier mechanism inputs, but are not claimed to be unseen during checkpoint development; some initial-noise keys are shared with the original cohort. The additional Maze20 $N = 4$ group has only five correct-candidate events, so its result remains limited by sample size. For any nonzero Δ_t, Equation 12 implies a worst-case margin-reproduction error of at least $|\Delta_t|/2$ for a readout using only the fixed context, time, and hard state.

Table 7: Effects of retaining history on correct-candidate probability margins, in percentage points. Means and confidence intervals weight problems equally; the last column counts events with positive effects.

Setting	Δ_t [95% CI]	Positive events
Original cohort		
Sudoku, $N = 2$	$1.456\ [0.767, 2.293]$	60/73
Sudoku, $N = 4$	$1.634\ [0.961, 2.406]$	63/71
Maze20, $N = 2$	$0.188\ [0.045, 0.356]$	66/73
Maze20, $N = 4$	$0.921\ [0.316, 1.641]$	43/44
Additional validation cohort		
Sudoku, $N = 2$	$1.168\ [0.213, 2.361]$	18/19
Sudoku, $N = 4$	$1.439\ [0.231, 3.084]$	10/10
Maze20, $N = 2$	$0.137\ [0.000117, 0.389]$	14/14
Maze20, $N = 4$	$0.345\ [0.017, 0.964]$	5/5

Table 8: Signed path-integral contributions for correct candidates in the original cohort, in probability percentage points. Each cell gives the mean and nominal 95% confidence interval. V-other is the part of V transmitted through first-layer receiving positions $i \neq j$. Negative estimates rounded to zero are shown as -0.000.

Pathway	Sudoku $N = 2$	Sudoku $N = 4$	Maze20 $N = 2$	Maze20 $N = 4$
Q	1.001 $[0.527, 1.576]$	1.114 $[0.655, 1.641]$	0.148 $[0.040, 0.276]$	0.656 $[0.235, 1.151]$
K	-0.002 $[-0.005, 0.001]$	0.000 $[-0.002, 0.002]$	-0.000 $[-0.002, 0.001]$	-0.005 $[-0.012, 0.001]$
V	0.052 $[0.019, 0.096]$	0.050 $[0.017, 0.085]$	0.040 $[0.007, 0.082]$	0.270 $[0.070, 0.522]$
R	0.405 $[0.214, 0.635]$	0.470 $[0.274, 0.696]$	-0.001 $[-0.001, 0.000]$	-0.000 $[-0.002, 0.001]$
V-other	0.093 $[0.043, 0.160]$	0.099 $[0.053, 0.153]$	0.032 $[0.006, 0.064]$	0.223 $[0.049, 0.450]$

How the response propagates. Following Theorem 2, we integrate the four signed contributions along the same full-state path:

$$I_X = \int_0^{h_t} \mathcal{C}_X(\lambda)\, d\lambda, \qquad X \in \{Q, K, V, R\}.$$

The calculation includes all first-layer heads and receiving positions, with gradients through the entire downstream network. Q has the largest mean contribution in each group, and V is positive in all four groups (Table 8). The bypass contribution R is positive in Sudoku and near zero in Maze20; it is distinct from the solver's direct state-persistence term. K intervals include zero, which does not establish that K is unused.

V-other, the contribution through receiving positions $i \neq j$, is positive with intervals excluding zero in every group. This shows that history at j can affect its final prediction through values read by other positions and processed by subsequent layers, as described by Equation 10. In Sudoku, V-other exceeds total V, reflecting cancellation with the contribution at receiver j; these are signed effects, not mechanism percentages.

These results support computational use of retained historical support, with positive effects for most evaluated correct candidates. They concern the next prediction margin, not final puzzle accuracy, and do not establish uniformly beneficial effects or positive derivatives throughout a continuous path.

Table 9: Sequence accuracy (%) of FLM and MDLM across model capacities and denoising steps on Maze10, Maze15, and Maze20. Parameter counts are matched between methods at each model size.

Parameters	Method	Denoising steps									
		1	2	4	8	16	32	64	128	256	512
Maze10											
44.94M	FLM	57.6	72.7	76.0	77.2	77.7	78.0	78.3	78.3	78.3	78.3
	MDLM	51.1	57.4	64.4	69.0	78.1	84.3	87.3	89.9	91.0	90.8
59.11M	FLM	95.7	99.5	99.5	99.5	99.5	99.5	99.6	99.6	99.6	99.6
	MDLM	87.2	88.8	92.1	94.6	95.2	98.3	98.5	98.6	98.2	98.6
87.47M	FLM	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0
	MDLM	97.2	97.6	98.5	99.0	99.2	99.7	99.9	99.8	99.9	99.7
Maze15											
59.90M	FLM	58.3	78.7	82.9	84.0	84.3	84.4	84.4	84.4	84.4	84.4
	MDLM	27.9	31.2	39.2	45.0	54.3	64.0	73.5	79.1	82.0	82.1
88.25M	FLM	95.9	98.9	99.1	99.1	99.1	99.1	99.2	99.2	99.2	99.2
	MDLM	49.8	54.9	62.5	69.6	76.2	83.7	88.3	91.8	93.7	95.3
139.00M	FLM	99.1	99.8	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9
	MDLM	74.7	79.2	83.1	87.3	91.5	94.7	96.2	97.7	96.9	97.1
Maze20											
140.83M	FLM	33.2	47.0	51.2	53.1	53.6	53.7	53.9	54.1	54.2	55.7*
	MDLM	26.2	29.0	33.9	38.1	45.2	52.6	63.4	72.8	77.0	79.2
258.81M	FLM	71.4	88.1	91.3	92.1	92.4	92.5	92.5	92.6	92.6	92.9*
	MDLM	35.6	41.2	48.2	55.8	63.0	70.6	80.2	83.7	87.9	90.1
309.19M	FLM	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0
	MDLM	43.0	50.4	55.9	64.5	70.7	79.6	84.9	89.5	92.5	92.9

D.4 Complete Results across Model Capacities

Table 9 reports the complete model-capacity sweep underlying Figures 1 and 2. For each maze task, we evaluate FLM and MDLM at three matched model sizes across ten denoising step budgets.

D.5 Wall-Clock Inference Efficiency

We measure inference latency on a single A100 80 GB GPU using FP32 with TF32 disabled. Methods use matched Transformer depths and widths within each task. We report mean latency over 32 shared test instances at batch size one, following two warm-up calls. Timing uses GPU synchronization and covers the complete sampler, excluding model loading, tokenization, data transfer, and scoring.

Here, N denotes nominal sampling steps. FLM uses N model evaluations, while standard discrete samplers use $N + 1$, including terminal denoising. Confidence remasking uses 2 and 128 evaluations at $N = 1$ and $N = 64$, respectively. AR+CoT uses cached greedy decoding with fixed output lengths of 480, 1024, and 1760 tokens for Maze10, Maze15, and Maze20, with EOS stopping disabled.

FLM achieves the lowest latency at $N = 1$ on all three tasks. At $N = 64$, its cost is comparable to standard discrete samplers, while confidence remasking approximately doubles latency. Together with Table 1, these results show that FLM combines strong low-step accuracy with low wall-clock inference cost.

Table 10: Mean inference latency in seconds per instance on one A100 (batch size one; lower is better). † marks randomly initialized models used solely for architecture-level timing; other entries use trained EMA weights. Sampling budgets do not enforce matched accuracy.

	Maze10		Maze15		Maze20	
Method	$N = 1$	$N = 64$	$N = 1$	$N = 64$	$N = 1$	$N = 64$
FLM	**0.013**	0.656	**0.033**	2.079	**0.151**	9.740
MDLM	0.023	0.693	0.067	2.144	0.303	9.845
MDLM + Entropy	0.022	0.714	0.067	2.162	0.303	9.902
MDLM + Confidence	0.023	1.371	0.066	4.204	0.303	19.392
SEDD	$0.023^{\dagger}$	$0.705^{\dagger}$	0.067	2.152	0.305	9.950
SEDD + Entropy	$0.026^{\dagger}$	$0.724^{\dagger}$	0.067	2.172	0.307	9.978
SEDD + Confidence	$0.022^{\dagger}$	$1.369^{\dagger}$	0.067	4.225	0.306	19.587
D3PM	$0.025^{\dagger}$	$0.645^{\dagger}$	$0.065^{\dagger}$	$2.082^{\dagger}$	0.301	9.808
ReMDM	0.021	0.651	0.066	2.089	0.301	9.772
AR+CoT	1.424		4.328		$17.384^{\dagger}$	