Large Language Continuous Diffusion Models

Zhihan Yang[*,†,1,2] **Wei Guo**[*,†,1,3] **Jean-Marie Lemercier**[*,1] **Simon Welker**[*,1]
Mohammad Mahdi Kamani[*,1] **Yonggan Fu**[1] **Sajad Norouzi**[1] **Julius Berner**[1]
Tomas Geffner[1] **Karsten Kreis**[1] **Yongxin Chen**[3] **Molei Tao**[3] **John Thickstun**[2]
Pavlo Molchanov[1] **Ante Jukić**[1] **Arash Vahdat**[⋆,1] **Morteza Mardani**[⋆,1]
[1]NVIDIA [2]Cornell University [3]Georgia Institute of Technology [†]Work done during internship at NVIDIA.
[*]Core contributors, equal contribution. [⋆]Equal supervision. Correspondence to: mmardani@nvidia.com

arXiv:2610.02665v2 [cs.CL] 12 Nov 2026

Despite the success of discrete diffusion language models (dLMs) for fast parallel decoding, their non-smooth, high-dimensional space hinders trajectory steering for reasoning and inference acceleration. To overcome this, we present *Nemotron-Labs-Sigma*, the first family of large-scale (3B/8B) continuous dLMs built on steerable, low-dimensional ODE/SDE latent trajectories. Trained blockwise via likelihood optimization, *Nemotron-Labs-Sigma* jointly denoises Gaussian-corrupted token embeddings while learning an optimal embedding geometry, and leverages pretrained autoregressive (AR) weights for warm-starting. At inference, we identify classifier-free guidance and score temperature as essential for high-fidelity reasoning and coding. Against state-of-the-art discrete counterparts (masked dLMs and AR baselines), *Nemotron-Labs-Sigma* outperforms them on standard benchmarks after pretraining (e.g., GSM8K, Minerva, HumanEval, MBPP) and on challenging reasoning tasks after supervised fine-tuning (e.g., MATH-500, AIME), with the largest consistent gains on coding. Beyond accuracy, we exploit two structural properties unique to continuous dLMs. First, smooth trajectories degrade gracefully at low NFEs and enable efficient few-step distillation: distribution matching proves strongest among the methods we study, giving roughly 8× acceleration on math and 4× on coding (8 and 4 tokens per forward pass) with negligible accuracy loss and preserved diversity. In addition, the one-step speculative decoding on the (pre-SFT) distilled 8B model yields a further ∼3.5× throughput gain over the AR counterpart (on GB200), outperforming masked dLM baselines. Second, embedding-space steering controls the quality–diversity trade-off, yielding strong pass@k that keeps improving as k grows. Together, these results establish continuous dLMs as a promising paradigm for efficient language generation. Model family weights are publicly available at 🤗 *Nemotron-Labs-Sigma*.

1. Introduction

While diffusion language models (dLMs) enable fast parallel decoding compared to autoregressive (AR) baselines (Inception Labs et al., 2025; Song et al., 2025; Fu et al., 2026; Zhu et al., 2026; DiffusionGemma Team et al., 2026), dominant discrete dLMs operate in high-dimensional, categorical state spaces (e.g., vocabulary size $V \geq 100$k) that suffer under limited step budgets and impede trajectory control for complex reasoning and coding tasks. Continuous diffusion offers a principled way to overcome these structural limitations through smooth latent geometries that natively support structured editing (Meng et al., 2022), flexible ODE trajectory steering (Zhang & Chen, 2023; Lu et al., 2022; Zhang et al., 2023), and efficient few-step distillation (Salimans & Ho, 2022; Song et al., 2023).

Recently, there has been growing interest in continuous diffusion and flow matching for text generation (Gulrajani & Hashimoto, 2023; Dieleman et al., 2022; Chen et al., 2026; Hu et al., 2026; Lee et al., 2026; Guo et al., 2026; Liang et al., 2026; Yang et al., 2026); see App. A for a review. However, existing methods remain limited in scale (< 2B parameters) or focus on *unconditional* generation, lacking evaluations on complex reasoning tasks. Whether continuous dLMs scale to complex conditional reasoning remains open. This is a non-trivial hurdle because continuous models update block embeddings globally, in contrast to the factorized categorical denoising of discrete dLMs that leads to semi-autoregressive token commitment per block.

To overcome these limitations, we present *Nemotron-Labs-Sigma* (3B/8B): **s**cal**i**ng up **G**aussian diffusion **m**odels for l**a**nguage. For brevity, we refer to the model family simply as *Sigma* throughout the remainder of the paper. We propose a family of continuous dLMs operating over a 16-dimensional embedding space trained blockwise via likelihood optimization. Within each block, the model learns to jointly denoise Gaussian-corrupted token embeddings and to structure an optimal embedding geometry. To scale training, we warm-start

© 2026 NVIDIA. All rights reserved.

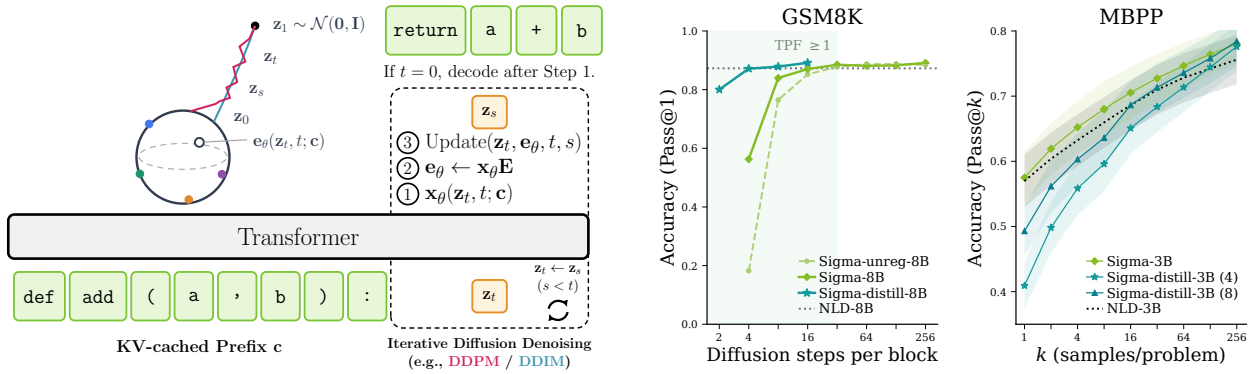


Figure 1 | **Left: Inference pipeline.** *Sigma* generates text blockwise via iterative diffusion denoising over 16D continuous latents ($\mathbf{z}_t$) conditioned on a KV-cached prefix ($\mathbf{c}$), followed by categorical readout. Predicted vocabulary probabilities $\mathbf{x}_\theta(\mathbf{z}_t, t; \mathbf{c})$ map to mean predictions $\mathbf{e}_\theta(\mathbf{z}_t, t; \mathbf{c})$ in 16D space via categorical reparameterization. Clean embeddings (dots) lie on the 16D unit sphere, while $\mathbf{e}_\theta$ lies on or within it. **Middle: Smooth trajectory enables low-NFE reasoning.** GSM8K accuracy vs. NFEs using DDIM sampler for *Sigma*(*-unreg*)-8B, and sampler (17) for *Sigma-distill*-8B. **Right: Scaling inference-time compute.** pass@k vs. k on MBPP (solid curves: fixed config across k; thin curves: alternative tested configs).

the model from existing pretrained AR weights. For inference, we explore diffusion trajectory sampling, identifying embedding-space Classifier-Free Guidance (CFG) and Score Temperature (ST) as crucial ingredients for high-fidelity reasoning and coding.

Our key contributions are summarized as follows:

- **First Large-Scale Continuous dLM:** We present *Nemotron-Labs-Sigma* (3B/8B), efficiently trained by warm-starting from pretrained AR weights, and optimizing a regularized continuous diffusion ELBO.
- **Competitive Math & Coding:** Across six base and ten instruct benchmarks, *Sigma* outperforms state-of-the-art discrete dLMs/ARs on reasoning and coding.
- **Inference Steering (CFG & ST):** By elucidating the inference design space, we show that embedding-space CFG and ST effectively govern the quality-diversity trade-off with minimal latency overhead via single-pass block-causal attention packing.
- **Distillation:** We study trajectory- and distribution-matching distillation (DMD) methods and find that phased DMD delivers the strongest acceleration, speeding up inference by 8× for math and 4× for coding, reaching the equivalent of 8 and 4 tokens per forward (TPFs), respectively.
- **Speculative Decoding:** We apply speculative decoding to both distilled and non-distilled *Sigma* variants, and find that combining distillation with one-step speculative decoding is highly effective, yielding a ∼3× throughput improvement over AR on an NVIDIA GB200 GPU.

2. Background

Notation. Let $\mathbf{x}$ be a sequence of length L over a vocabulary $\mathcal{V}$ of size V. We identify $\mathbf{x}$ with a matrix in $\{0,1\}^{L\times V}$ whose rows $\mathbf{x}^1, ..., \mathbf{x}^L$ are one-hot vectors in $\mathbb{R}^V$. Δ^V denotes the probability simplex over $\mathcal{V}$. $\langle \cdot, \cdot \rangle$ denotes the inner product. NELBO stands for negative evidence lower bound.

Plaid. Continuous dLMs lift discrete tokens into a low-dimensional Euclidean space and run Gaussian diffusion. There have been multiple proposed design choices, see App. A for an overview. We adopt the embedding and diffusion process from Plaid (Gulrajani & Hashimoto, 2023), a Variational Diffusion Model (VDM, Kingma et al. (2021)) over token embeddings. Let $\mathbf{E} \in \mathbb{R}^{V\times d_e}$ be a learnable embedding matrix whose rows lie on the unit sphere, with $d_e \ll V$. The embedding of $\mathbf{x}$ is given by $\mathbf{e} := \mathbf{x}\mathbf{E} \in \mathbb{R}^{L\times d_e}$ (i.e., standard embedding lookup). The forward process applies variance-preserving Gaussian noise on $\mathbf{e}$:

$$q(\mathbf{z}_t \mid \mathbf{x}) = \mathcal{N}(\alpha_t \mathbf{e},\ \sigma_t^2 \mathbf{I}), \qquad \alpha_t, \sigma_t > 0, \qquad \alpha_t^2 + \sigma_t^2 = 1, \qquad t \in [0,1], \tag{1}$$

with $\mathrm{SNR}(t) = \alpha_t^2/\sigma_t^2$ decreasing monotonically over $t \in [0,1]$. A denoiser $\mathbf{x}_\theta(\mathbf{z}_t, t) \in (\Delta^V)^L \subset \mathbb{R}^{L\times V}$ with output prior returns a categorical distribution over the vocabulary at every position. The reverse process is the exact Gaussian posterior $q(\mathbf{z}_s \mid \mathbf{z}_t, \mathbf{x})$ (Kingma et al., 2021, App. A.2) with $\mathbf{x}$ replaced by $\mathbf{x}_\theta(\mathbf{z}_t, t)$. Its NELBO splits into 3 terms: prior KL $\mathcal{L}_{\text{prior}}$, reconstruction loss $\mathcal{L}_{\text{rec}}$, and diffusion loss $\mathcal{L}_{\text{diff}}$:

$$\begin{aligned} -\log p_\theta(\mathbf{x}) \le \mathcal{L}_{\text{NELBO}}(\mathbf{x}) := \mathrm{KL}(q(\mathbf{z}_1 \mid \mathbf{x}) \,\|\, p(\mathbf{z}_1)) + \mathbb{E}_{\mathbf{z}_0\sim q(\mathbf{z}_0\mid\mathbf{x})}\left[\textstyle\sum_{l=1}^{L} -\log\langle \mathbf{x}_\theta^l(\mathbf{z}_0, 0), \mathbf{x}^l\rangle\right] \\ - \tfrac{1}{2}\ \mathbb{E}_{t\sim\mathrm{Unif}[0,1]}\, \mathbb{E}_{\mathbf{z}_t\sim q(\mathbf{z}_t\mid\mathbf{x})}[\mathrm{SNR}'(t)\|\mathbf{e}_\theta(\mathbf{z}_t, t) - \mathbf{e}\|^2], \end{aligned} \tag{2}$$

where $p(\mathbf{z}_1) = \mathcal{N}(\mathbf{0}, \mathbf{I})$ and the *categorical reparameterization* $\mathbf{e}_\theta(\mathbf{z}_t, t) := \mathbf{x}_\theta(\mathbf{z}_t, t)\mathbf{E} \in \mathbb{R}^{L\times d_e}$ predicts the posterior mean embedding $\mathbb{E}[\mathbf{e} \mid \mathbf{z}_t, t]$ corresponding to $\mathbf{x}_\theta(\mathbf{z}_t, t)$.

Learned noise schedule. The schedule admits $\gamma(t) = \gamma_0 + (\gamma_1 - \gamma_0)\tilde{\gamma}(t)$ with learnable endpoints $\gamma_0 < \gamma_1$ and a monotone interior shape $\tilde{\gamma} \in [0,1]$, giving $\alpha_t^2 = \sigma(-\gamma(t))$, $\sigma_t^2 = \sigma(\gamma(t))$ and $\mathrm{SNR}(t) = \mathrm{e}^{-\gamma(t)}$.[1] The diffusion term depends on the endpoints alone and is invariant to $\tilde{\gamma}(t)$, which instead controls its Monte-Carlo (MC) variance (Kingma et al., 2021), so they are learned for different reasons. Learning the interior-shape distributes denoising difficulty over t (Yang et al., 2026).

Learned embeddings. The NELBO (2) depends on $\mathbf{E}$ via (1), where embeddings become likelihood-trained parameters, which separates Plaid from continuous dLMs built on fixed (e.g., one-hot) encodings. Not learning embeddings significantly worsens Plaid's performance (Yang et al., 2026).

Self-conditioning (SC). Proposed in Chen et al. (2023), SC lets the denoiser see its own previous estimate. Let $\mathbf{x}_\theta(\mathbf{z}_t, t; \mathbf{e}_{\text{sc}})$ denote a denoiser taking an additional predicted clean embedding $\mathbf{e}_{\text{sc}} \in \mathbb{R}^{L\times d_e}$, and $\mathbf{0}$ for the empty condition (omitted by default in our notation). During training, 25% of training samples are self-conditioned: a first pass without gradient tracking produces $\mathbf{e}_{\text{sc}} := \mathbf{x}_\theta(\mathbf{z}_t, t; \mathbf{0})\mathbf{E}$, and only the second pass, $\mathbf{x}_\theta(\mathbf{z}_t, t; \mathrm{sg}(\mathbf{e}_{\text{sc}}))\mathbf{E}$, receives gradients. The remaining samples use $\mathbf{0}$ as SC input, so the denoiser learns both branches. During validation, all samples enable SC. Since $\mathbf{e}_{\text{sc}}$ is itself a function of $(\mathbf{z}_t, t)$, the NELBO remains valid. Notice that during training, 25% SC increases the non-embedding FLOPs by only about 8%, which is minimal. Furthermore, during inference, one directly feeds the previous-step's prediction as SC, which incurs no overhead.

Scaling. RePlaid (Yang et al., 2026) realigns Plaid with the architectures of modern discrete dLMs/ARs and finds that, once aligned, continuous diffusion scales competitively with masked diffusion.

3. Large Language Continuous Diffusion Models

3.1. Blockwise Probabilistic Model

Inspired by Nemotron-Labs-Diffusion (NLD, Fu et al. (2026)), a state-of-the-art open-sourced masked dLM, we adopt a blockwise formulation for our continuous diffusion language model: the sequence $\mathbf{x}$ is partitioned into B contiguous blocks $\{\mathbf{x}^{(b)}\}_{b=1}^{B}$ of size $L' = L/B$, where the model denoises one block at a time given its clean prefix. As in Arriola et al. (2025), we factorize the likelihood autoregressively *across* blocks and then bound each factor:

$$-\log p_\theta(\mathbf{x}) = \textstyle\sum_{b=1}^{B} -\log p_\theta(\mathbf{x}^{(b)} \mid \mathbf{x}^{(<b)}) \le \sum_{b=1}^{B} \mathcal{L}_{\text{NELBO}}(\mathbf{x}^{(b)}; \mathbf{x}^{(<b)}) =: \mathcal{L}_{\text{BD}}(\mathbf{x}), \tag{3}$$

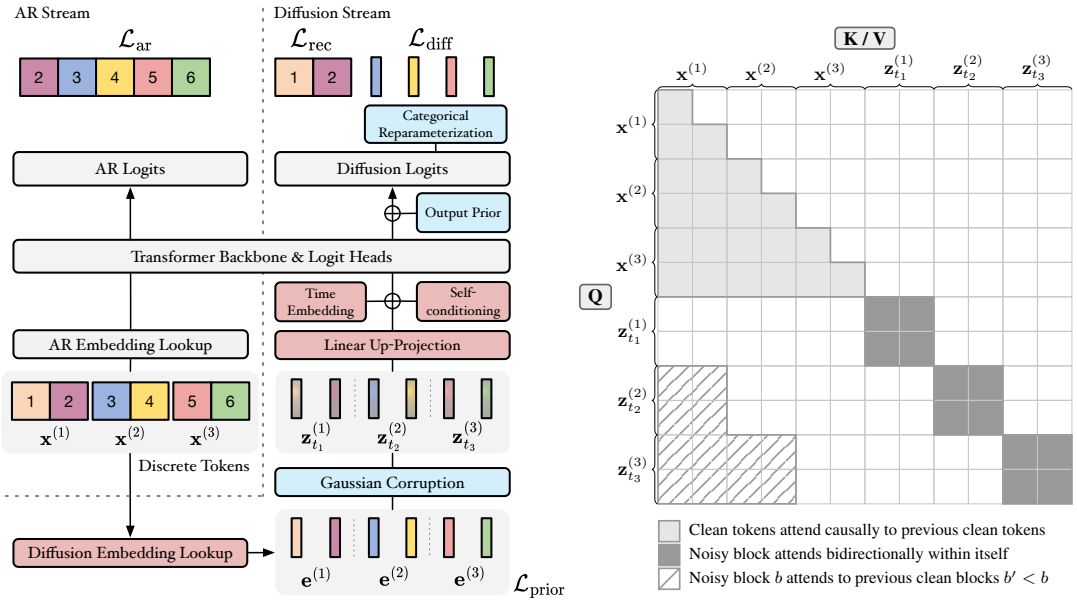
where we obtain $\mathcal{L}_{\text{NELBO}}(\mathbf{x}^{(b)}; \mathbf{x}^{(<b)})$ by conditioning the denoiser on $\mathbf{x}^{(<b)}$ as follows:

$$\begin{aligned} \mathcal{L}_{\text{NELBO}}(\mathbf{x}^{(b)}; \mathbf{x}^{(<b)}) &= \mathrm{KL}(q(\mathbf{z}_1^{(b)} \mid \mathbf{x}^{(b)}) \,\|\, p(\mathbf{z}_1^{(b)})) \\ &\quad + \mathbb{E}_{q(\mathbf{z}_0^{(b)}\mid\mathbf{x}^{(b)})}\left[\textstyle\sum_{l=1}^{L'} -\log\langle \mathbf{x}_\theta^l(\mathbf{z}_0^{(b)}, 0; \mathbf{x}^{(<b)}), \mathbf{x}^{(b,l)}\rangle\right] \\ &\quad - \tfrac{1}{2}\, \mathbb{E}_{t\sim\mathrm{Unif}[0,1]}\, \mathbb{E}_{q(\mathbf{z}_t^{(b)}\mid\mathbf{x}^{(b)})}[\mathrm{SNR}'(t)\|\mathbf{e}_\theta(\mathbf{z}_t^{(b)}, t; \mathbf{x}^{(<b)}) - \mathbf{e}^{(b)}\|^2]. \end{aligned} \tag{4}$$

Training. The reconstruction loss at $t = 0$ and diffusion loss at $t \in (0,1)$ do not share a single forward pass. RePlaid evaluates one of them per $\mathbf{x}^{(b)}$ via Horvitz–Thompson estimator (Horvitz & Thompson, 1952) with $R \sim \mathrm{Ber}(p_{\text{rec}})$, i.e., $\mathbf{x}^{(b)}$ is assigned to $\mathcal{L}_{\text{rec}}$ w.p. $p_{\text{rec}} \in (0,1)$:

$$\mathcal{L}_{\text{NELBO}}(\mathbf{x}^{(b)}; \mathbf{x}^{(<b)}) = \mathbb{E}_{R\sim\mathrm{Ber}(p_{\text{rec}})}\left[\mathcal{L}_{\text{prior}}(\mathbf{x}^{(b)}) + \tfrac{R}{p_{\text{rec}}}\mathcal{L}_{\text{rec}}(\mathbf{x}^{(b)}; \mathbf{x}^{(<b)}) + \tfrac{1-R}{1-p_{\text{rec}}}\mathcal{L}_{\text{diff}}(\mathbf{x}^{(b)}; \mathbf{x}^{(<b)})\right]. \tag{5}$$

[1] $\sigma(t) = 1/(1+\mathrm{e}^{-t})$ is the sigmoid function.



(a) Training procedure of *Sigma*. (b) Attention mask for training.

Figure 2 | **Training procedure of *Sigma* (a) and its transformer attention mask (b)** for an example sequence $\mathbf{x}$ of length $L = 6$ with block size $L' = 2$. During pretraining, we feed the clean discrete tokens $\mathbf{x}$ to the AR stream (a, left part) and the noisy continuous embeddings $\mathbf{z}$ to the diffusion stream (a, right part). For the diffusion stream, each block is randomly chosen to either compute the reconstruction loss (1st block, $t = 0$) or the diffusion loss (2nd & 3rd blocks, $t \sim \text{Unif}[0, 1]$); see (5). Reconstruction loss is the cross entropy between the logits and the ground truth tokens, while for diffusion loss we use categorical reparameterization to obtain the mean embedding prediction and compute the mean squared error against the ground truth diffusion embeddings.

Notice that this is an unbiased estimator if $\mathcal{L}_{\text{diff}}$ and $\mathcal{L}_{\text{rec}}$ are unbiased. We set $p_{\text{rec}} = 0.1$, empirically found to minimize the training objective variance of RePlaid.

To construct the unbiased estimator for $\mathcal{L}_{\text{BD}}(\mathbf{x})$ in (3) for training, we sample $R^{(b)}$ i.i.d. for each block. If $R^{(b)} = 0$, we sample $t^{(b)}$ and $\mathbf{z}^{(b)}_{t^{(b)}}$; otherwise, we set $t = 0$ and sample $\mathbf{z}^{(b)}_0$. The noise schedule endpoints are learned to optimize $\mathcal{L}_{\text{NELBO}}(\mathbf{x}^{(b)}; \mathbf{x}^{(<b)})$ while the schedule interior shape is learned to optimize its MC variance, with both shared across blocks. As mentioned before, we apply SC only to 25% of samples (full sequence with all blocks). See Alg. 1 for the training algorithm.

3.2. Pretraining

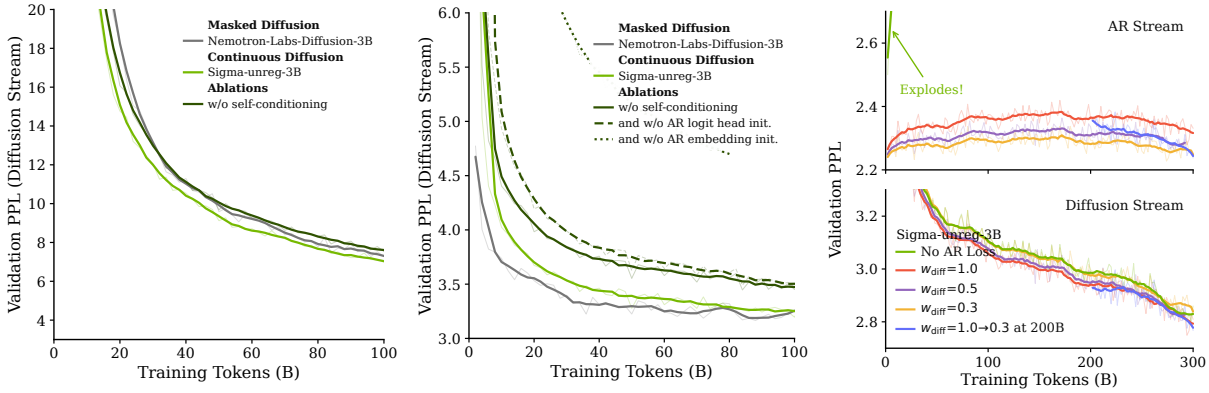
NLD utilizes AR initialization and regularization to enhance downstream performance (Fu et al., 2026). However, adapting these techniques to continuous dLMs like *Sigma* is non-trivial, as the activation dynamics of continuous and discrete models differ fundamentally; it is thus unclear whether discrete AR initialization remains as effective for continuous representation space. To establish a rigorous baseline, we first align *Sigma*'s data, architecture, and optimization with discrete dLMs for a controlled comparison at scale (Sec. 3.2.1). In Sec. 3.2.2, we evaluate the impact of AR initialization when pretraining *Sigma-unreg*-3B[2] with (3), followed by an investigation of AR regularization for training our flagship 3B/8B models.

3.2.1. Unified Pretraining of Continuous and Discrete dLMs

Data. As with NLD, we adopt the open-sourced pretraining data from Basant et al. (2025). We use the Mistral-NeMo tokenizer ($V = 131{,}072$) (Mistral AI, 2024), batch size 512, and $L = 4096$.

Architecture. *Sigma* adopts the transformer backbone of NLD and its AR baseline, ensuring the three families are parameter- and FLOP-matched. As with NLD, *Sigma* uses a block causal attention mask (Fig. 2b).

[2] "*unreg*" means no regularization loss for the diffusion embedding $\mathbf{E}$. This additional loss will be introduced in Sec. 4.3.



(a) From scratch + no AR reg. Block size: 64

(b) From AR init. + no AR reg. Block size: 64

(c) From AR init. + AR reg. Block size: 32

Figure 3 | **Pretraining *Sigma-unreg*-3B**: (a) from scratch, (b) from AR initialization, and (c) from AR initialization with AR regularization. PPL for AR and diffusion streams is computed via $\mathcal{L}_{\text{ar}}(\mathbf{x})$ in (6) and $\mathcal{L}_{\text{BD}}(\mathbf{x})$ in (3), respectively. Early investigations (a, b) use block size 64; our flagship models (c) use block size 32 to match the inference block size of NLD. Proper AR initialization/regularization leads to the best *Sigma* model.

Sigma uniquely adds a unit-norm 16-dimensional diffusion embedding table with a linear up-projection, a categorical reparameterization mapping V-dimensional probabilities to low-dimensional soft embeddings, a self-conditioning linear projection, Fourier time conditioning, and a learnable noise schedule (see Sec. 2). Together, these account for less than 1% of the total parameters. We provide thorough details in App. C.

Training. We train our models for 150k steps, following the optimization procedure detailed in App. C. Given a batch size of 512, 150k steps correspond to over 300B tokens. Training consumed ∼9,600 and ∼19,200 NVIDIA GB200 GPU hours for the 3B and 8B models, respectively.

3.2.2. AR Initialization and Regularization

AR Initialization. To evaluate the impact of initialization, we compare pretraining *Sigma* (*i*) from scratch and (*ii*) from an AR baseline pretrained on 1T tokens. Due to the dimension mismatch between high-dimensional AR embeddings and *Sigma*'s 16-dimensional space, we randomly initialize the diffusion embedding layer. As shown in Figs. 3a and 3b, *Sigma-unreg*-3B ($L' = 64$) with self-conditioning achieves competitive perplexity relative to NLD-3B ($L' = 64$) under both regimes (*i*) and (*ii*). However, AR initialization significantly accelerates convergence and yields lower final training loss.

AR Regularization. Continuous diffusion training inherently suffers from high gradient variance and slow convergence. To mitigate variance and accelerate optimization, we follow NLD by adding AR regularization, optimizing the joint objective $\mathcal{L}_{\text{ar}} + w_{\text{diff}}\mathcal{L}_{\text{BD}}$ (Fig. 3c; see App. C for the w_{diff} weighting schedule), where

$$\mathcal{L}_{\text{ar}} = \sum_{l=2}^{L} -\log\langle \mathbf{x}_\theta^l(\mathbf{x}^{<l}), \mathbf{x}^l \rangle \tag{6}$$

is computed over the AR stream using causal attention (Fig. 2). This architecture employs two distinct readout heads, with the AR regularization computed exclusively from the AR head. While this dual-head formulation improves auxiliary AR decoding capability, our primary objective remains maximizing diffusion decoding performance.

Finally, input sequences utilize AR embeddings for the AR stream and 16-dimensional diffusion embeddings for the diffusion stream (Fig. 2), facilitating AR initialization and efficient KV-caching. As shown in Fig. 3c, AR regularization significantly improves training stability; with a properly tuned w_{diff}, it even outperforms pure perplexity optimization for the diffusion stream. This confirms that AR regularization stabilizes training while yielding a higher-quality model. Consequently, we adopt AR initialization, AR regularization, and a block size of $L' = 32$ for all subsequent *Sigma* experiments.

4. Language Generation via Continuous Diffusion Sampling

Drawing on the success of continuous diffusion samplers using SDE/ODE solvers (Song et al., 2021b), we systematically explore decoding samplers for continuous dLMs, evaluating performance across standard mathematical reasoning and coding benchmarks.

4.1. Block Diffusion Sampling

To generate a response $\mathbf{r}$ given a prompt $\mathbf{p}$, we denoise one block at a time. Let $\mathbf{c} := (\mathbf{p}, \mathbf{r}^{(<b)})$ be the clean context available when generating block b, stored in an attention key-value (KV) cache using the AR embeddings. An offset block-causal mask determines who can see whom (Fig. 5a). Positions in $\mathbf{c}$ use causal attention, so their KV can be cached and reused across diffusion steps, regardless of whether the prompt length is a multiple of L'. Positions in the noisy block $\mathbf{z}_t^{(b)}$ attend bidirectionally within the block and causally to the entire prefix $\mathbf{c}$. After a block is decoded, the AR embeddings of its committed tokens are appended to the causal KV cache for the next block during the next block's first diffusion step, at no extra NFE.

Sampling block b starts from $\mathbf{z}_1^{(b)} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ and steps from t to $s < t$ along a discretization of $[1, 0]$. At each step the denoiser yields a predicted clean embedding $\hat{\mathbf{e}}_\theta := \mathbf{e}_\theta(\mathbf{z}_t^{(b)}, t; \mathbf{c}) = \mathbf{x}_\theta(\mathbf{z}_t^{(b)}, t; \mathbf{c})\mathbf{E}$ and the implied noise prediction $\hat{\boldsymbol{\epsilon}}_\theta := \frac{1}{\sigma_t}(\mathbf{z}_t^{(b)} - \alpha_t \hat{\mathbf{e}}_\theta)$. We introduce two canonical samplers for the reverse process. The *DDPM* sampler uses the ancestral kernel of VDM (Kingma et al., 2021). It draws $\mathbf{z}_s^{(b)}$ from the closed-form Gaussian posterior $q(\mathbf{z}_s^{(b)} \mid \mathbf{z}_t^{(b)}, \mathbf{x}^{(b)})$ with the clean embedding replaced by $\hat{\mathbf{e}}_\theta$. The *DDIM* sampler (Song et al., 2021a) leverages the predictions $\hat{\mathbf{e}}_\theta$ and $\hat{\boldsymbol{\epsilon}}_\theta$ to deterministically update the state as $\mathbf{z}_s^{(b)} = \alpha_s \hat{\mathbf{e}}_\theta + \sigma_s \hat{\boldsymbol{\epsilon}}_\theta$. For both samplers, after the final step, tokens are read out as argmax of the conditional logit output $\mathbf{x}_\theta(\mathbf{z}_0^{(b)}, 0; \mathbf{c})$ and appended to $\mathbf{c}$, so NFEs = steps + 1. See Alg. 2 for the complete sampling algorithms and App. E.1 for more details.

4.2. Trajectory Steering

To sharpen model predictions, we incorporate two reverse-process steering methods to our blockwise continuous diffusion model with no impact to its training efficiency and minimal impact to its inference efficiency.

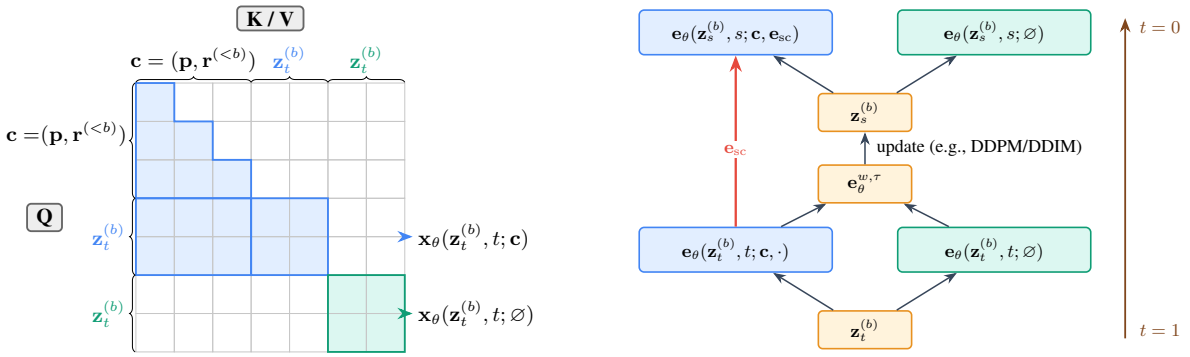
Classifier-free guidance (CFG). CFG (Ho & Salimans, 2022) is the standard tool for class-conditional or text-to-image diffusion models. In CFG, one queries the model twice to obtain a conditional and an unconditional prediction and then linearly extrapolates them. For our block diffusion model, the clean context $\mathbf{c}$ serves as the natural conditioning information, and CFG extrapolates away from the unconditional prediction for the current block b:

$$\mathbf{e}_\theta^w(\mathbf{z}_t^{(b)}, t; \mathbf{c}) := \mathbf{e}_\theta(\mathbf{z}_t^{(b)}, t; \varnothing) + w\left[\mathbf{e}_\theta(\mathbf{z}_t^{(b)}, t; \mathbf{c}) - \mathbf{e}_\theta(\mathbf{z}_t^{(b)}, t; \varnothing)\right], \tag{CFG}$$

where $\varnothing$ drops the entire clean context $\mathbf{c}$ (so $\mathbf{z}_t^{(b)}$ attends only within its own block) and $w \geq 1$ is the guidance strength ($w = 1$ falls back to the plain conditional prediction). We bring two technical considerations. First, at batch size 1 transformer inference is memory-bound, so we pack conditional and unconditional predictions into a single forward pass at almost no wall-clock cost (Liu et al., 2026b), using the packed mask of Fig. 5a. We measure this overhead to be less than 5% per forward pass at block size 32 and 64 (App. E.9). Second, unlike in image diffusion models and existing dLMs that require train-time context dropout (Potaptchik et al., 2026; Hu et al., 2026), **our CFG is a pure inference-time trick**: due to sequence packing in pretraining, the first block typically starts in the middle of a document and naturally sees no context. We discuss existing approaches in App. A.

Score temperature (ST). ST (Dieleman et al., 2022) is a heuristic that sharpens the conditional score itself, rescaling the score $\nabla \log p_t(\mathbf{z}_t^{(b)} \mid \mathbf{c})$ (or equivalently, the noise prediction) by $\frac{1}{\tau}$ for $\tau \in (0, 1]$. One can show that it corresponds to the following operation on the mean prediction:

$$\mathbf{e}_\theta^\tau(\mathbf{z}_t^{(b)}, t; \mathbf{c}) := \tfrac{\mathbf{z}_t^{(b)}}{\alpha_t} + \tfrac{1}{\tau}\left[\mathbf{e}_\theta(\mathbf{z}_t^{(b)}, t; \mathbf{c}) - \tfrac{\mathbf{z}_t^{(b)}}{\alpha_t}\right]. \tag{ST}$$



(a) Packed attention mask while generating block b ($L' = 2$). Prefix $\mathbf{c}$ is causal (KV-cached). Conditional $\mathbf{z}_t^{(b)}$ attends to all of $\mathbf{c}$ and bidirectionally within the block; unconditional copy attends only to itself.

(b) One reverse step $t \to s$. CFG and ST mix the two denoisers into $\mathbf{e}_\theta^{w,\tau}$; an update yields $\mathbf{z}_s^{(b)}$. SC (red) feeds the previous conditional mean (not sharpened by CFG or ST) to the next conditional branch.

Figure 5 | **Inference** for blockwise continuous diffusion: packed CFG attention (a) and one reverse step (b).

Intuitively, since $\mathbb{E}[\mathbf{z}_t^{(b)}|\mathbf{x}^{(b)}] = \alpha_t \mathbf{e}^{(b)}$, $\frac{1}{\alpha_t}\mathbf{z}_t^{(b)}$ serves as the naive bad prediction to extrapolate from. ST does not incur further model queries, and $\tau = 1$ leaves the sampler unchanged. We also recognize ST as a special case of Temporal Score Rescaling proposed in Xu et al. (2026).

Finally, we remark that due to the commutativity of (CFG) and (ST), we can combine them into a single operation $\mathbf{e}_\theta^{w,\tau}(\mathbf{z}_t^{(b)}, t; \mathbf{c})$, which is essentially using the score of the composed distribution $(p_t(\cdot \mid \mathbf{c})^w p_t(\cdot \mid \varnothing)^{1-w})^{1/\tau}$. We observed that using either knob is weaker than using both of them, see Fig. 4.

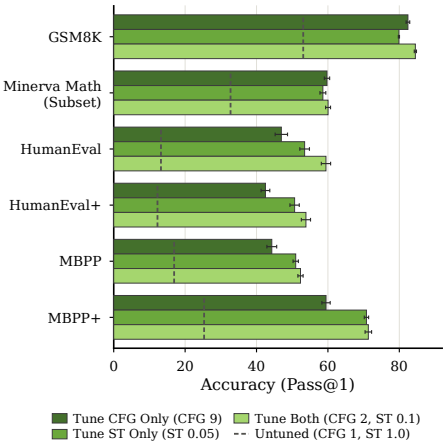


Figure 4 | **CFG and ST tuning** on *Sigma-unreg*-3B for DDPM sampling. Combining both techniques yields larger gains over applying either in isolation, and markedly outperforms sampling without CFG or ST. See Fig. 15 for the full tuning grid.

Self-conditioning (SC). At inference, SC is applied only to the conditional prediction: when querying the model at time s, the previous step's conditional prediction $\mathbf{e}_\theta(\mathbf{z}_t^{(b)}, t; \mathbf{c})$ (before applying CFG and ST) serves as the SC input for the conditional prediction, while the unconditional prediction receives $\mathbf{0}$ as the SC input (Fig. 5b).

Sharpening in logit space. While in the above, we apply guidance directly in the embedding domain, we can also sharpen in logit space. In App. E.6, we show that the logit-space CFG and temperature do provide sharpening, but the embedding-space knobs provide stronger performance, hence we keep (CFG) and (ST) as the default option.

4.3. Embedding Regularization

We observe that during training, embeddings collapse to approximately 5 effective dimensions (i.e., >90% explained variance in the first 5 principal components), a behavior typically not observed in AR language model embeddings but consistent with observations in RePlaid (Yang et al., 2026). This may arise because constant embeddings are a trivial solution for the denoising loss. We hypothesize that this embedding collapse exacerbates token ambiguity under prediction errors from coarse numerical precision and low-NFE sampling, which motivates regularizing the embeddings toward a uniform distribution on the unit sphere. For our regularized model *Sigma*, we therefore apply a multi-scale extension of *Sketched Uniform Spherical Regularization* (SUSReg, Nicollier et al. (2026)) to the embedding matrix $\mathbf{E}$ during pretraining, penalizing deviations from the uniform distribution on the unit sphere. We also add a local repulsion loss to push apart very close embedding vectors, and make other minor training changes, see App. C.1 for full details. We ablate a simpler variant as in the original SUSReg (Nicollier et al., 2026), denoted *Sigma-preg*, which also avoids

Table 1 | **Pass@1 accuracy of *Sigma-unreg*-3B and *Sigma*-3B.** The *Sigma* diffusion rows are reported as mean (over 10 seeds) ± two standard errors. **Bold**: the best value across all methods; underline: the best value within each mode.

	Math		Coding				
Method	GSM8K	Minerva Math	HumanEval	HumanEval+	MBPP	MBPP+	Avg.
AR initialization	83.70	**66.28**	51.22	48.78	60.60	77.78	64.73
AR mode							
Nemotron-Labs-Diffusion-3B	83.78	65.06	53.05	47.56	**60.80**	**78.84**	64.85
***Sigma-unreg*-3B (Ours)**	83.40	65.10	57.32	51.83	59.00	76.19	65.47
***Sigma*-3B (Ours)**	81.20	63.98	51.83	48.17	56.80	75.66	62.94
Diffusion mode							
LLaDA-8B	70.96	27.30	32.32	27.44	40.80	51.85	41.78
Dream-7B	77.18	39.60	54.88	49.39	56.80	74.60	58.74
Nemotron-Labs-Diffusion-3B	83.40	63.96	56.10	53.05	57.60	74.34	64.74
***Sigma-unreg*-3B (Ours)**							
DDPM 256 steps/block (w=2, τ=0.1)	84.50 (± 0.31)	64.60 (± 0.28)	**59.45** (± 1.30)	**53.84** (± 1.29)	52.32 (± 0.73)	71.32 (± 0.91)	64.34
DDIM 32 steps/block (w=3, τ=0.3)	81.91 (± 0.32)	62.76 (± 0.18)	56.59 (± 1.31)	50.67 (± 1.00)	47.26 (± 0.46)	63.49 (± 0.77)	60.45
***Sigma*-3B (Ours)**							
DDPM 256 steps/block (w=3, τ=0.1)	**85.57** (± 0.29)	64.98 (± 0.19)	57.26 (± 0.82)	**53.84** (± 0.55)	58.00 (± 0.43)	73.62 (± 0.51)	**65.55**
DDIM 32 steps/block (w=3, τ=0.3)	84.48 (± 0.21)	64.52 (± 0.21)	55.37 (± 1.28)	51.71 (± 1.23)	53.00 (± 0.34)	71.30 (± 0.45)	63.40

embedding collapse but is less effective at improving low-NFE quality and distillation (Tabs. 10 and 11); we report its full benchmark results in Tab. 6.

Results. As shown in Tabs. 1 and 2, the resulting regularized models *Sigma*-3B/8B achieve the highest average reasoning scores, outperforming NLD (Fu et al., 2026). Furthermore, they significantly improve few-step sampling (see Tab. 10) and distillation (see App. F.3) compared to *Sigma-unreg*-3B/8B, supporting our hypotheses. Despite exhibiting substantially different embedding geometries from the unregularized models (Fig. 10), interestingly we find that regularization incurs almost no validation PPL penalty.

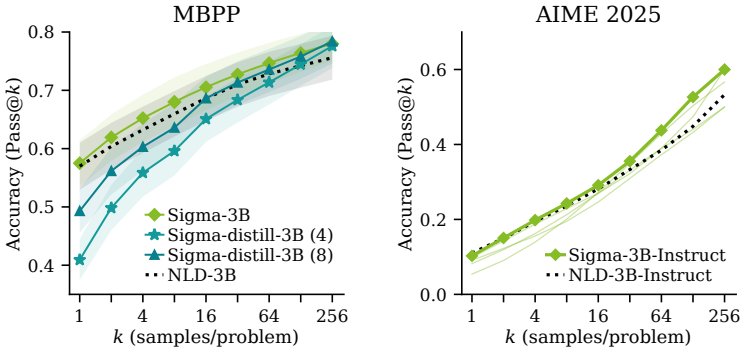


Figure 6 | **Left: Pass@k accuracy on MBPP for *Sigma*-3B and *Sigma-distill*-3B**. All models use pass@1 selected sampler. **Right: Pass@k accuracy on AIME 2025 for *Sigma*-3B-Instruct**, highlighting best CFG/ST pair chosen from five predefined combinations.

4.4. Mathematical Reasoning and Coding Performance

We evaluate on two math and four coding benchmarks: GSM8K, Minerva Math, HumanEval, HumanEval+, MBPP, and MBPP+. Results are reported in Tab. 1. For diffusion generation, *Sigma* models always use SC, and we report performance on DDPM sampling with 256 steps/block with EDM $\rho = 7$ time discretization, and DDIM sampling with 32 steps/block with uniform-γ time discretization. See App. E.3 for details on the harness, scoring pipeline, error bar conventions, and the samplers for baselines, and App. E.2 for time discretization schemes.

Results. In AR mode, *Sigma-unreg*-3B attains the best six-task average across all methods; in diffusion mode, our DDPM-256 runs are comparable and out-perform NLD-3B on average and sets the best base-model numbers on GSM8K, HumanEval, and HumanEval+; they also outperform LLaDA-8B and Dream-7B by a noticeable margin, despite being $\sim$2$\times$ smaller in parameters. Empirically, we observe *less* left-to-right bias within a block for *Sigma* compared with the confidence-based masked dLM sampling (Fig. 19). To confirm the benefit of AR regularization (Sec. 3.2.2), we also trained *Sigma-unreg*-3B with $\mathcal{L}_{\text{BD}}$ alone (3) without $\mathcal{L}_{\text{ar}}$ (6); results are uniformly worse across all tasks (Fig. 16). We refer readers to Tab. 2 for the same evaluation on 8B base models.

Pass@k analysis. While pass@1 accuracy in Tab. 1 captures single-draw quality, test-time search and RL draw multiple rollouts for the same prompt, where diverse rollouts are needed for richer learning signals and

Table 2 | **Pass@1 accuracy of *Sigma-unreg*-8B and *Sigma*-8B.** The *Sigma* diffusion rows are reported as mean (over 10 seeds) ± two standard errors. **Bold**: the best value across all methods; underline: the best value within each mode.

	Math		Coding				
Method	GSM8K	Minerva Math	HumanEval	HumanEval+	MBPP	MBPP+	Avg.
AR initialization	85.90	**69.18**	59.15	53.05	68.20	**83.33**	69.80
AR mode							
Llama-3.1-8B	54.06	18.22	35.37	28.66	48.80	61.90	41.17
Ministral3-8B	80.21	44.58	42.68	38.41	61.60	76.98	57.41
Qwen3-8B	86.73	52.94	64.63	56.71	**69.40**	83.07	68.91
Nemotron-Labs-Diffusion-8B	88.25	66.00	60.37	53.05	68.20	82.54	69.73
***Sigma-unreg*-8B (Ours)**	86.88	66.72	51.83	48.78	64.00	80.95	66.53
***Sigma*-8B (Ours)**	84.84	65.18	56.10	52.44	63.80	82.54	67.48
Diffusion mode							
LLaDA-8B	70.96	27.30	32.32	27.44	40.80	51.85	41.78
Dream-7B	77.18	39.60	54.88	49.39	56.80	74.60	58.74
Nemotron-Labs-Diffusion-8B	87.26	65.16	62.80	57.32	67.00	81.75	70.22
***Sigma-unreg*-8B (Ours)**							
DDPM 256 steps/block (w=2, τ=0.1)	**89.07** (± 0.28)	67.47 (± 0.20)	61.40 (± 1.73)	57.44 (± 1.62)	64.78 (± 0.54)	77.83 (± 0.72)	69.67
DDIM 32 steps/block (w=3, τ=0.3)	87.78 (± 0.33)	67.13 (± 0.15)	58.54 (± 1.04)	53.17 (± 1.27)	59.54 (± 0.57)	74.15 (± 0.43)	66.72
***Sigma*-8B (Ours)**							
DDPM 256 steps/block (w=3, τ=0.1)	88.57 (± 0.12)	66.72 (± 0.16)	65.12 (± 1.22)	60.61 (± 1.15)	65.74 (± 0.44)	80.19 (± 0.76)	**71.16**
DDIM 32 steps/block (w=3, τ=0.3)	88.48 (± 0.14)	65.91 (± 0.15)	**65.37** (± 0.57)	**60.91** (± 0.74)	62.68 (± 0.30)	77.72 (± 0.32)	70.18

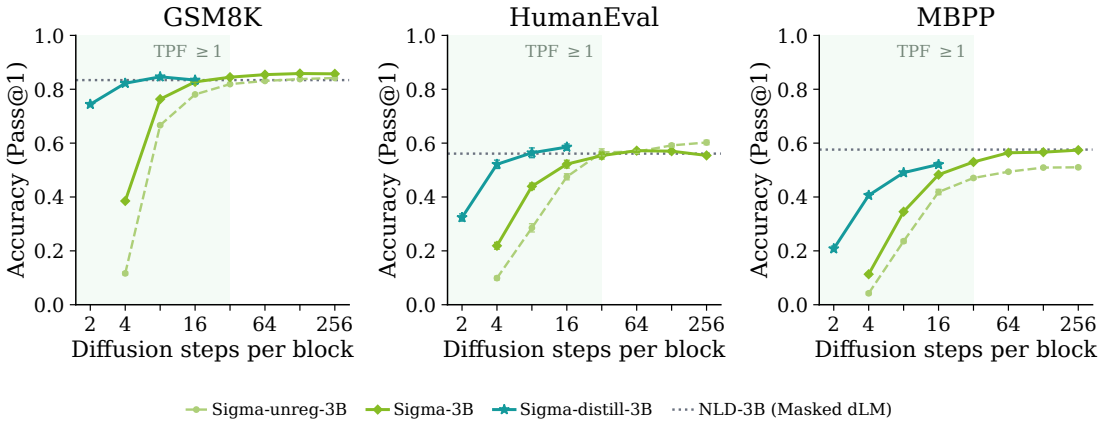


Figure 7 | **Pass@1 accuracy of *Sigma-unreg*-3B, *Sigma*-3B, and *Sigma-distill*-3B** Undistilled models use DDIM with best per-NFE (CFG, ST), shared across the three tasks. Full results on base tasks in Tab. 10.

increase the chance of discovering a correct solution. Pass@k does not directly measure diversity, but tests whether additional samples expand correct-solution coverage. We report results in Fig. 6 on MBPP at the 3B scale, using the same sampler configuration that performs best on pass@1 accuracy with 256 NFEs DDIM sampler. There, *Sigma*-3B increasingly surpasses as more trials are available, showing that the model has a useful output diversity. See App. E.8 for the full study.

5. Few-Step Generation

5.1. ODE Sampler and Time Discretization

In previous results, we mainly run 256 DDPM steps per block to establish the strongest performance. In low-NFE regime, it is generally known that deterministic samplers paired with well-tuned time discretization lead to better performance. To investigate the tradeoffs for *Sigma*, we choose three representative tasks (GSM8K, HumanEval, and MBPP), and vary the diffusion sampling steps from 4 to 256. For each sampler-discretization-budget combination, we sweep CFG and ST; we then report, for each combination, the CFG-ST pair that achieves the best average accuracy over the two tasks; see App. E.3 for more details.

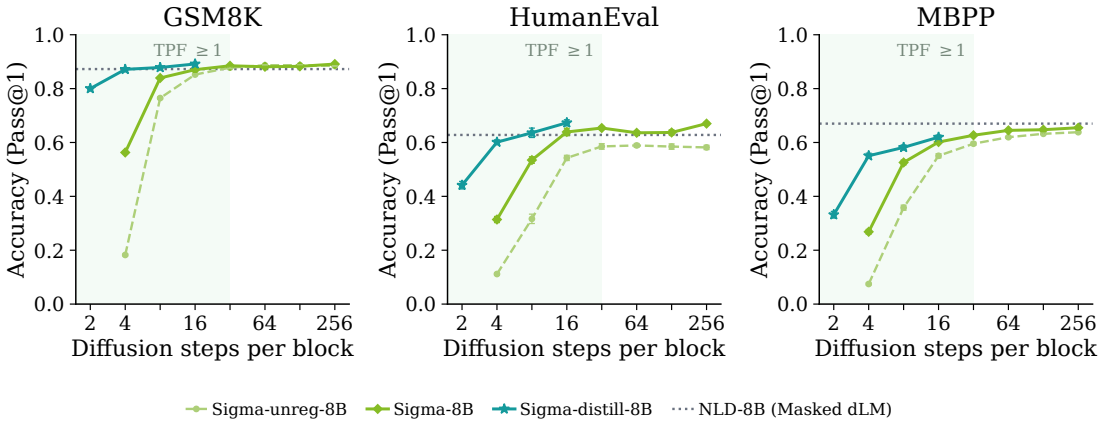


Figure 8 | **Pass@1 accuracy of *Sigma-unreg*-8B, *Sigma*-8B and *Sigma-distill*-8B.** Undistilled models use DDIM with best per-NFE (CFG, ST), shared across the three tasks. Full results on base tasks in Tab. 11.

Results. As shown in Fig. 14, DDIM substantially outperforms DDPM at low NFEs while matching DDPM at high NFEs. The uniform-γ discretization (App. E.2) performs the best for a wide range of budgets. Using the same optimal hyperparameters from the two tasks, we evaluate the 16/32-step performance across all six tasks in Tabs. 1 and 2; Tabs. 10 and 11 show performance at other budgets.

5.2. Distribution Matching Distillation for Few-Step Generation

Although DDIM has better performance than DDPM at a reduced generation budget, the pretrained models still lose quality at very low NFEs (especially $\leq$16 steps, see Fig. 7). We therefore distill *Sigma* into a few-step student *Sigma-distill* using Distribution-Matching Distillation (DMD, Yin et al. (2024b,a)). Notably, the student uses self-conditioning and a purely stochastic sampler that alternates between clean embedding estimation and fresh Gaussian noise injection, which is different from DDIM sampler. Furthermore, our DMD students are trained to match the target distribution without using CFG, which saves inference FLOPs per NFE compared with the guided teacher, and needs no ST rescaling.

Results. As shown in Figs. 7 and 8 and Tabs. 10 and 11, our distillation recipe yields significant acceleration: with only 8 NFEs (TPF = 4), *Sigma-distill* matches both NLD (Fu et al., 2026) and our best-performing 256-NFE DDPM sampler on GSM8K and HumanEval at both 3B and 8B scales. For MBPP, which is a full code generation task, syntactic bugs are introduced at low NFE, but *Sigma-distill*-3B is still within 3% accuracy of the 256-step sampler at 8 NFEs. Interestingly, despite significant acceleration, we observe that *Sigma-distill*-3B does not lose any diversity compared with its 256-step DDPM teacher (see App. F.3). As a result, the distilled model exhibits strong pass@k performance: both 4- and 8-NFE *Sigma-distill*-3B catch up with the 256-NFE DDIM *Sigma*-3B teacher at $k = 256$ (see Fig. 6). Together, this suggests the distilled model is well suited for accelerated RL-based alignment and post-training.

6. Self-Speculative Decoding

Sigma trains a block-diffusion stream and a causal AR stream on one backbone, so it can act as its own drafter. In each round, one denoiser call from pure noise drafts a block of $L' = 32$ tokens. The AR stream then scores the block in one causal pass, keeps the longest prefix that matches its own greedy tokens, and appends one token of its own (Leviathan et al., 2023; Fu et al., 2026). The output is therefore the greedy AR output. A round of two forward passes commits AL + 1 tokens, where the acceptance length (AL) counts the accepted draft tokens that reach the returned answer.

At both 3B and 8B scales, we compare NLD to *Sigma-distill* (Sec. 5.2) to a post-trained model *Sigma-distill-spec*, which pairs the same distillation objective as *Sigma-distill* with an acceptance objective (full details in App. H.4). We serve with SGLang (Zheng et al., 2024) in BF16 on one NVIDIA GB200, one request at a time, and evaluate on GSM8K, HumanEval, and MBPP. AL is pooled over the prompts whose outputs neither loop nor reach the length cap. App. H details the protocol and gives further results for the undistilled *Sigma* with trained drafters.

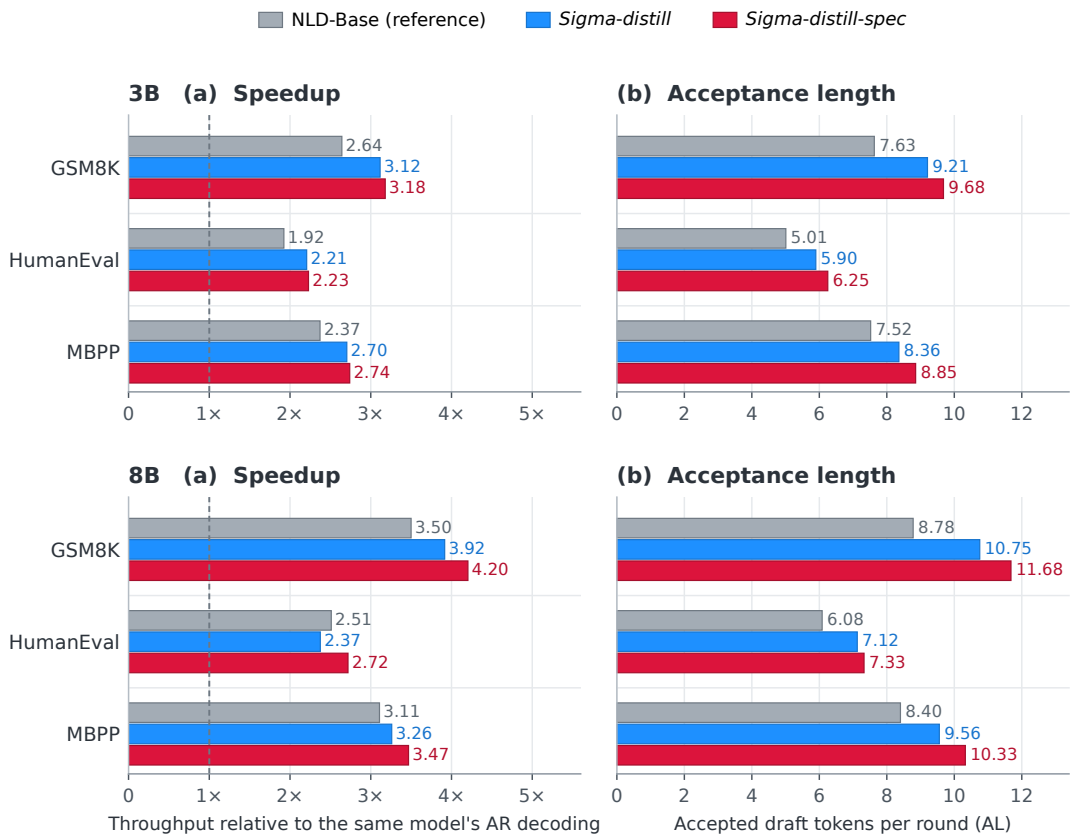


Figure 9 | **Self-speculative decoding at 3B (top) and 8B (bottom).** (a) Throughput relative to the same model's AR decoding. (b) Accepted draft tokens per round (AL).

Results. Fig. 9 shows speedup and AL, and Tab. 3 shows accuracy and the AL gain over NLD. Without any drafter training, *Sigma-distill*-3B accepts 11–21% more draft tokens per round than NLD-3B, and *Sigma-distill*-8B accepts 14–23% more than NLD-8B. We also show in App. H.3 that *Sigma-distill*-3B also accepts 15–22% more tokens than the *Sigma*-3B, more than what an adapter trained specifically for improving drafting adds to that model. *Sigma-distill*-8B gains also more than the adapter on GSM8K and MBPP and as much on HumanEval (App. H.3). Further acceptance training (joint distillation and drafter fine-tuning) yields *Sigma-distill-spec*, which accepts another 5–9% more tokens per round and decodes up to 3.18× (3B) and 4.20× (8B) faster than AR decoding, ahead of NLD on every benchmark and model size. In all these experiments, the accuracy remains that of the AR mode in each model, since verification returns the greedy AR output. Therefore the speedup comes at no cost relative to AR decoding with the same model.

7. Supervised Fine-Tuning

We apply supervised fine-tuning (SFT) to the pretrained *Sigma* models with the joint AR and block-diffusion losses, restricting supervision to assistant responses. The complete data processing, optimization recipe, and continuous-diffusion-specific controls are given in App. G.

Results. As shown in Tabs. 4 and 5, the average performance of *Sigma-instruct* models is on par with NLD at both 3B and 8B scale, and surpasses other open-source masked dLMs like Dream and LLaDA at the 8B scale. For both 3B checkpoints, we partially re-sweep CFG and ST after SFT and observe that the same optima selected on the corresponding pre-SFT models remain statistically tied for best after SFT (Tab. 20); we therefore retain the pre-SFT settings for the full evaluation.

Table 3 | **Accuracy and AL gain over NLD.** Accuracy (%) is that of AR decoding, which speculative decoding reproduces (GSM8K: flexible extract; code: pass@1). AL gain is relative to NLD of the same size, over the prompts both models keep, with 95% bootstrap intervals.

		GSM8K		HumanEval		MBPP	
	Model	Acc.	AL gain	Acc.	AL gain	Acc.	AL gain
3B	NLD-3B-Base	83.8	–	52.4	–	61.6	–
	Sigma-distill-3B	82.9	+20.8% [18.9, 22.5]	56.1	+18.5% [7.4, 30.1]	58.2	+10.5% [7.1, 13.9]
	Sigma-distill-spec-3B	82.2	+26.7% [24.8, 28.7]	55.5	+26.3% [15.6, 36.1]	58.8	+17.4% [14.1, 20.7]
8B	NLD-8B-Base	88.8	–	61.6	–	66.2	–
	Sigma-distill-8B	86.4	+22.6% [20.6, 24.6]	56.1	+15.0% [4.1, 27.6]	64.0	+13.6% [10.3, 17.3]
	Sigma-distill-spec-8B	86.4	+33.0% [31.0, 35.1]	57.3	+21.0% [9.8, 34.9]	63.8	+22.7% [19.1, 26.5]

Table 4 | **Pass@1 accuracy of *Sigma-unreg*-3B-*instruct* and *Sigma*-3B-*instruct*.** The *Sigma* diffusion rows are reported as mean (over 10 seeds) ± two standard errors. The four reference models at the top are AR models. Baseline numbers are from Fu et al. (2026). In *diffusion mode*, underline marks best value per task.

	Math				Coding			Science & Knowledge		IF	
Method	GSM8K	Math500	AIME24	AIME25	HumanEval	MBPP	LCB-CPP	MMLU	GPQA	IFEval	Avg.
Llama-3.2-3B-Instruct	62.09	32.60	0.00	0.00	65.85	66.93	9.25	62.45	27.78	69.69	39.66
Phi-4-mini-Instruct	76.57	70.80	10.00	6.67	77.44	75.13	9.91	53.37	40.91	71.16	49.20
Ministral3-3B-Instruct-2512	87.41	71.90	12.50	12.71	64.79	68.39	12.50	63.21	28.79	57.55	47.98
Qwen3-4B	92.19	68.40	10.63	10.63	75.91	72.49	15.58	77.05	37.18	72.20	53.23
AR mode											
Nemotron-Labs-Diffusion-3B-Instruct	87.87	77.60	23.33	16.67	76.22	71.16	21.37	71.70	39.39	69.39	55.47
***Sigma-unreg*-3B-*instruct* (Ours)**	88.17	71.60	20.00	10.00	71.34	77.51	13.00	71.36	38.38	73.62	53.50
***Sigma*-3B-*instruct* (Ours)**	85.29	71.60	3.33	3.33	73.17	78.04	14.76	69.98	34.34	70.74	50.46
Diffusion mode											
Nemotron-Labs-Diffusion-3B-Instruct	88.40	74.80	16.67	10.00	74.39	73.54	16.30	72.06	33.84	68.93	52.89
***Sigma-unreg*-3B-*instruct* (Ours)**											
DDPM (256 steps, w=2, τ=0.1)	89.32 ± 0.24	73.20 ± 0.45	15.67 ± 2.84	11.00 ± 1.90	77.32 ± 1.05	71.75 ± 0.72	12.97 ± 0.62	72.58 ± 0.08	37.68 ± 1.70	70.11 ± 0.64	53.16
DDIM (32 steps, w=3, τ=0.3)	88.10 ± 0.24	71.42 ± 0.88	14.00 ± 2.46	8.67 ± 1.93	73.60 ± 1.17	68.76 ± 0.69	11.56 ± 0.61	71.37 ± 0.08	34.24 ± 1.42	68.18 ± 0.32	50.99
***Sigma*-3B-*instruct* (Ours)**											
DDPM (256 steps, w=3, τ=0.1)	89.64 ± 0.10	73.08 ± 0.62	13.67 ± 1.99	11.33 ± 1.03	77.01 ± 1.24	74.97 ± 0.59	13.50 ± 0.42	70.64 ± 0.12	36.01 ± 1.46	73.26 ± 0.59	53.31
DDIM (32 steps, w=3, τ=0.3)	88.82 ± 0.15	72.10 ± 0.74	15.33 ± 2.70	13.33 ± 2.11	73.96 ± 0.91	73.31 ± 0.82	12.16 ± 0.56	70.20 ± 0.07	34.80 ± 1.99	69.00 ± 0.49	52.30

8. Conclusions

We introduced *Nemotron-Labs-Sigma*, a family of large-scale blockwise continuous dLMs obtained by efficient continual pretraining from an AR checkpoint, together with a reverse-diffusion sampling recipe that combines DDPM/DDIM samplers with CFG, ST, and SC. In diffusion mode, *Sigma* (3B/8B) outperforms same-size masked dLMs and AR baselines trained under an aligned pipeline, both after pretraining and after SFT, with the most consistent gains on coding. A sphere-normalized embedding regularization further improves low-NFE sampling and makes model distillation markedly more effective.

Beyond accuracy, the smoothness of continuous trajectories buys efficiency. Distribution matching is the strongest of the few-step distillation methods we study: using only 8 NFEs (TPF = 4), *Sigma-distill*-3B almost matches both NLD (Fu et al., 2026) and our best-performing 256-NFE DDPM teacher, and it does so without collapsing the teacher's output distribution, preserving diversity and yielding pass@k that keeps improving as k grows. Because *Sigma* carries a causal AR stream on the same backbone, the *distilled* student can also serve as its own one-call drafter: self-speculative decoding returns the greedy AR output at a 3.46× average speedup over AR decoding at 8B, against 3.04× for NLD.

Together, these results establish continuous dLMs as a practical path to efficient language generation. Looking ahead, the controllable quality–diversity trade-off and the shared diffusion/AR backbone make them particularly attractive for RL-based post-training, which we leave to future work.

Table 5 | **Pass@1 accuracy of *Sigma-unreg*-8B-*instruct* and *Sigma*-8B-*instruct*.** The *Sigma* diffusion rows are reported as mean (over 10 seeds) ± two standard errors. The three reference models at the top are AR models. NLD-8B-Instruct's diffusion mode results are reproduced under aligned protocol, while other baseline numbers are from Fu et al. (2026, Tab. 5). In *diffusion mode*, underline marks best value per task.

	Math				Coding			Science & Knowledge		IF	
Method	GSM8K	Math500	AIME24	AIME25	HumanEval	MBPP	LCB-CPP	MMLU	GPQA	IFEval	Avg.
Qwen2.5-7B-Instruct	91.89	75.10	13.75	6.88	77.44	81.55	12.33	74.86	37.12	74.58	54.55
Qwen3-8B	92.42	84.80	30.21	22.08	81.71	81.88	21.09	76.66	49.24	87.38	62.75
Ministral3-8B-Instruct-2512	92.42	83.60	27.71	24.58	71.04	78.97	20.76	73.90	42.87	64.31	58.02
AR mode											
Nemotron-Labs-Diffusion-8B-Instruct	94.01	88.00	33.33	33.33	80.49	85.19	28.85	79.85	44.44	68.65	63.61
***Sigma-unreg*-8B-*instruct* (Ours)**	91.74	80.80	16.67	20.00	78.05	84.92	21.59	78.22	46.97	79.50	59.85
***Sigma*-8B-*instruct* (Ours)**	91.58	79.40	26.67	16.67	81.10	80.69	12.33	76.27	43.94	78.06	58.67
Diffusion mode											
LLaDA-8B Instruct	79.91	39.20	0.00	0.00	49.40	41.00	4.19	65.50	33.30	59.90	37.24
Dream-7B Instruct	81.00	43.00	0.00	3.33	55.50	58.80	1.25	67.00	33.00	62.50	40.54
SDAR-8B Chat	91.30	78.60	16.67	10.00	78.70	72.00	13.44	78.60	40.20	61.40	54.09
Nemotron-Labs-Diffusion-8B-Instruct	91.58	84.60	30.00	23.33	83.54	82.01	21.15	76.65	33.33	73.02	59.92
***Sigma-unreg*-8B-*instruct* (Ours)**											
DDPM (256 steps, w=2, τ=0.1)	93.24 ± 0.20	79.80 ± 0.55	23.67 ± 3.19	18.00 ± 2.15	77.99 ± 1.06	80.40 ± 0.80	20.79 ± 0.52	77.91 ± 0.11	42.37 ± 1.80	79.83 ± 0.42	59.40
DDIM (32 steps, w=3, τ=0.3)	92.90 ± 0.18	78.20 ± 0.33	19.00 ± 0.97	19.00 ± 2.50	78.35 ± 1.62	79.74 ± 0.98	18.33 ± 0.61	77.60 ± 0.09	41.87 ± 1.74	76.69 ± 0.38	58.17
***Sigma*-8B-*instruct* (Ours)**											
DDPM (256 steps, w=3, τ=0.1)	91.96 ± 0.21	78.40 ± 0.67	23.33 ± 2.67	17.00 ± 1.75	82.50 ± 1.17	82.04 ± 0.67	20.09 ± 0.63	76.57 ± 0.08	42.37 ± 1.17	77.18 ± 0.38	59.14
DDIM (32 steps, w=3, τ=0.3)	91.90 ± 0.17	76.70 ± 0.67	21.67 ± 3.02	14.33 ± 3.53	80.55 ± 0.58	79.47 ± 0.55	19.38 ± 0.48	76.15 ± 0.08	42.93 ± 1.75	75.96 ± 0.32	57.90

References

Niket Agarwal, Arslan Ali, Maciej Bala, Yogesh Balaji, Erik Barker, Tiffany Cai, Prithvijit Chattopadhyay, Yongxin Chen, Yin Cui, Yifan Ding, et al. Cosmos world foundation model platform for physical AI. *arXiv preprint arXiv:2501.03575*, 2025.

Marianne Arriola, Aaron Gokaslan, Justin T Chiu, Zhihan Yang, Zhixuan Qi, Jiaqi Han, Subham Sekhar Sahoo, and Volodymyr Kuleshov. Block diffusion: Interpolating between autoregressive and diffusion language models. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=tyEyYT267x.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, et al. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*, 2021.

Pavel Avdeyev, Chenlai Shi, Yuhao Tan, Kseniia Dudnyk, and Jian Zhou. Dirichlet diffusion score model for biological sequence generation. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 1276–1301. PMLR, 23–29 Jul 2023. URL https://proceedings.mlr.press/v202/avdeyev23a.html.

Aarti Basant, Abhijit Khairnar, Abhijit Paithankar, Abhinav Khattar, Adithya Renduchintala, Aditya Malte, Akhiad Bercovich, Akshay Hazare, Alejandra Rico, Aleksander Ficek, et al. Nvidia nemotron nano 2: An accurate and efficient hybrid mamba-transformer reasoning model. *arXiv preprint arXiv:2508.14444*, 2025.

Georgios Batzolis, Mark Girolami, and Luca Ambrogioni. CoBit: Language modeling with bitstream diffusion. *arXiv preprint arXiv:2605.07013*, 2026.

Gary Becigneul and Octavian-Eugen Ganea. Riemannian adaptive optimization methods. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=r1eiqi09K7.

Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. Piqa: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 7432–7439, 2020.

Aaron Blakeman, Aaron Grattafiori, Aarti Basant, Abhibha Gupta, Abhinav Khattar, Adi Renduchintala, Aditya Vavre, Akanksha Shukla, Akhiad Bercovich, Aleksander Ficek, et al. Nemotron 3 Nano: Open, efficient

mixture-of-experts hybrid mamba-transformer model for agentic reasoning. *arXiv preprint arXiv:2512.20848*, 2025.

Aakshita Chandiramani, Aaron Blakeman, Abdullahi Olaoye, Abhibha Gupta, Abhilash Somasamudramath, Abhinav Khattar, Adeola Adesoba, Adi Renduchintala, Adil Asif, Aditya Agrawal, et al. Nemotron 3 super: Open, efficient mixture-of-experts hybrid mamba-transformer model for agentic reasoning. *arXiv preprint arXiv:2604.12374*, 2026.

Huiwen Chang, Han Zhang, Lu Jiang, Ce Liu, and William T Freeman. MaskGIT: Masked generative image transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022.

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

Boyuan Chen, Diego Martí Monsó, Yilun Du, Max Simchowitz, Russ Tedrake, and Vincent Sitzmann. Diffusion forcing: Next-token prediction meets full-sequence diffusion. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=yDo1ynArjj.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde De Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

Ting Chen, Ruixiang Zhang, and Geoffrey Hinton. Analog bits: Generating discrete data using diffusion models with self-conditioning. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=3itjR9QxFw.

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. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=5fybcQZ0g4.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try ARC, the AI2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Oscar Davis, Samuel Kessler, Mircea Petrache, Ismail Ilkan Ceylan, Michael M. Bronstein, and Joey Bose. Fisher flow matching for generative modeling over discrete data. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=6jOScqwdHU.

Oscar Davis, Anastasiia Filippova, Pierre Ablin, Victor Turrisi, Amitis Shidani, Marco Cuturi, and Louis Béthune. Scaling categorical flow maps. *arXiv preprint arXiv:2605.07820*, 2026.

Justin Deschenaux and Caglar Gulcehre. Language modeling with hyperspherical flows. *Transactions on Machine Learning Research*, 2026. ISSN 2835-8856. URL https://openreview.net/forum?id=jBh2WNssnL.

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

DiffusionGemma Team, Adrien Ali Taïga, James Assiene, Daniele Calandriello, Rahma Chaabouni, João Gante, Tamara von Glehn, Nate Keating, Chris Knutsen, Martin Kukla, et al. DiffusionGemma technical report. *arXiv preprint arXiv:2608.00146*, 2026.

Xiangyu Fan, Zesong Qiu, Zhuguanyu Wu, Fanzhou Wang, Zhiqian Lin, Tianxiang Ren, Dahua Lin, Ruihao Gong, and Lei Yang. Phased DMD: Few-step distribution matching distillation via score matching within subintervals. In *Conference on Computer Vision and Pattern Recognition 2026*, 2026. URL https://openreview.net/forum?id=nwLx41yFhO.

Dan Friedman and Adji Bousso Dieng. The vendi score: A diversity evaluation metric for machine learning. *Transactions on Machine Learning Research*, 2023. ISSN 2835-8856.

Yonggan Fu, Lexington Whalen, Abhinav Garg, Chengyue Wu, Maksim Khadkevich, Nicolai Oswald, Enze Xie, Daniel Egert, Sharath Turuvekere Sreenivas, Shizhe Diao, Chenhan D. Yu, Ye Yu, Weijia Chen, Sajad Norouzi, Jingyu Liu, Shiyi Lan, Ligeng Zhu, Jin Wang, Jindong Jiang, Morteza Mardani, Mehran Maghoumi, Song Han, Ante Jukić, Nima Tajbakhsh, Jan Kautz, and Pavlo Molchanov. Nemotron-Labs-Diffusion: A tri-mode language model unifying autoregressive, diffusion, and self-speculation decoding. In Hung-Yi Lee, Sunipa Dev, Junyi Jessy Li, and Desmond Elliott (eds.), *Proceedings of the 2026 Conference on Empirical Methods in Natural Language Processing*, pp. 54868–54887, Budapest, Hungary, October 2026. Association for Computational Linguistics. ISBN 979-8-89176-459-0. doi: 10.18653/2026.emnlp-main.2416. URL https://aclanthology.org/2026.emnlp-main.2416/.

Ishaan Gulrajani and Tatsunori Hashimoto. Likelihood-based diffusion language models. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=e2MCL6hObn.

Hongcan Guo, Qinyu Zhao, Yian Zhao, Shen Nie, Rui Zhu, Qiushan Guo, Feng Wang, Tao Yang, Hengshuang Zhao, Guoqiang Wei, et al. Continuous latent diffusion language model. *arXiv preprint arXiv:2605.06548*, 2026.

Xiaochuang Han, Sachin Kumar, and Yulia Tsvetkov. SSD-LM: Semi-autoregressive simplex-based diffusion language model for text generation and modular control. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki (eds.), *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 11575–11596, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-long.647. URL https://aclanthology.org/2023.acl-long.647/.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=d7KBjmI3GmQ.

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

Jonathan Ho, Tim Salimans, Alexey Gritsenko, William Chan, Mohammad Norouzi, and David Fleet. Video diffusion models. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 8633–8646. Curran Associates, Inc., 2022. doi: 10.52202/068431-0628. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/39235c56aef13fb05a6adc95eb9d8d66-Paper-Conference.pdf.

Emiel Hoogeboom, Didrik Nielsen, Priyank Jaini, Patrick Forré, and Max Welling. Argmax flows and multinomial diffusion: Learning categorical distributions. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, 2021. URL https://openreview.net/forum?id=6nbpPqUCIi7.

Daniel G Horvitz and Donovan J Thompson. A generalization of sampling without replacement from a finite universe. *Journal of the American statistical Association*, 47(260):663–685, 1952.

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.

Xun Huang, Zhengqi Li, Guande He, Mingyuan Zhou, and Eli Shechtman. Self forcing: Bridging the train-test gap in autoregressive video diffusion. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=mSiN7i0BYH.

Inception Labs, Samar Khanna, Siddhant Kharbanda, Shufan Li, Harshit Varma, Eric Wang, Sawyer Birnbaum, Ziyang Luo, Yanis Miraoui, Akash Palrecha, et al. Mercury: Ultra-fast language models based on diffusion. *arXiv preprint arXiv:2506.17298*, 2025.

Jaehyeong Jo and Sung Ju Hwang. Continuous diffusion model for language modeling. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=VGv5y60sXC.

Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho (eds.), *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=k7FuTOWMOc7.

Tero Karras, Miika Aittala, Tuomas Kynkäänniemi, Jaakko Lehtinen, Timo Aila, and Samuli Laine. Guiding a diffusion model with a bad version of itself. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=bg6fVPVs3s.

Diederik Kingma, Tim Salimans, Ben Poole, and Jonathan Ho. Variational diffusion models. *Advances in neural information processing systems*, 34:21696–21707, 2021. URL https://openreview.net/forum?id=2LdBqxc1Yv.

Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James V. Miranda, Alisa Liu, Nouha Dziri, Shane Lyu, Yuling Gu, Saumya Malik, Victoria Graf, Jena D. Hwang, Jiangjiang Yang, Ronan Le Bras, Oyvind Tafjord, Chris Wilhelm, Luca Soldaini, Noah A. Smith, Yizhong Wang, Pradeep Dasigi, and Hannaneh Hajishirzi. Tülu 3: Pushing frontiers in open language model post-training. *arXiv preprint arXiv:2411.15124*, 2024. URL https://arxiv.org/abs/2411.15124.

Paul Le Van Kiem, Dario Shariatian, Umut Simsekli, and Alain Durmus. Distribution matching distillation for continuous diffusion language models. *arXiv preprint arXiv:2609.40235*, 2026.

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.

Yaniv Leviathan, Matan Kalman, and Yossi Matias. Fast inference from transformers via speculative decoding. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 19274–19286, 2023. URL https://proceedings.mlr.press/v202/leviathan23a.html.

Aitor Lewkowycz, Anders Johan Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Venkatesh Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, Yuhuai Wu, Behnam Neyshabur, Guy Gur-Ari, and Vedant Misra. Solving quantitative reasoning problems with language models. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho (eds.), *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=IFXTZERXdM7.

Pengxiang Li, Shilin Yan, Jiayin Cai, Renrui Zhang, Ruichuan An, Ziyu Guo, and Xiaowei Gao. Adaptive classifier-free guidance via dynamic low-confidence masking. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=0EcomthIKh.

Xiang Lisa Li, John Thickstun, Ishaan Gulrajani, Percy Liang, and Tatsunori Hashimoto. Diffusion-LM improves controllable text generation. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho (eds.), *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=3s9IrEsjLyk.

Jiajun Liang, Yucheng Liao, Yukang Cao, Jiazhe Wei, Ken Li, Wende Tan, Jiankun Zhang, ZY Cui, Jingkang Yang, Liucheng Guo, et al. AURORA-LM: Autoencoding unified representation for continuous-latent diffusion language modeling. *arXiv preprint arXiv:2608.02602*, 2026.

Alexander H Liu, Kartik Khandelwal, Sandeep Subramanian, Victor Jouault, Abhinav Rastogi, Adrien Sadé, Alan Jeffares, Albert Jiang, Alexandre Cahill, Alexandre Gavaudan, et al. Ministral 3. *arXiv preprint arXiv:2601.08584*, 2026a.

Jingyu Liu, Xin Dong, Zhifan Ye, Rishabh Mehta, Yonggan Fu, Vartika Singh, Ce Zhang, and Pavlo Molchanov. TiDAR: Think in diffusion, talk in autoregression. In A. Chowdhery and Z. Jia (eds.), *Proceedings of Machine Learning and Systems*, volume 8, pp. 748–762. MLSys, 2026b. URL https://proceedings.mlsys.org/paper_files/paper/2026/file/1367d856028f65a9555b0274db09e608-Paper-Conference.pdf.

Weiyang Liu, Rongmei Lin, Zhen Liu, Li Xiong, Bernhard Schölkopf, and Adrian Weller. Learning with hyperspherical uniformity. In Arindam Banerjee and Kenji Fukumizu (eds.), *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics*, volume 130 of *Proceedings of Machine Learning Research*, pp. 1180–1188. PMLR, 13–15 Apr 2021. URL https://proceedings.mlr.press/v130/liu21d.html.

Justin Lovelace, Varsha Kishore, Chao Wan, Eliot Seo Shekhtman, and Kilian Q Weinberger. Latent diffusion for language generation. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=NKdtztladR.

Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. DPM-solver: A fast ODE solver for diffusion probabilistic model sampling in around 10 steps. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho (eds.), *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=2uAaGwlP_V.

Chenlin Meng, Yutong He, Yang Song, Jiaming Song, Jiajun Wu, Jun-Yan Zhu, and Stefano Ermon. SDEdit: Guided image synthesis and editing with stochastic differential equations. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=aBsCjcPu_tE.

Viacheslav Meshchaninov, Egor Chimbulatov, Alexander Shabalin, Aleksandr Abramov, and Dmitry Vetrov. Cosmos: Compressed and smooth latent space for text diffusion modeling. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=Rv6Lz84FlZ.

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

Mistral AI. Mistral NeMo. https://mistral.ai/news/mistral-nemo/, July 2024. Accessed: 2026-09-14.

Léo Nicollier, Max Dunitz, Marc Pic, Pablo Musé, Enric Meinhardt-Llopis, and Gabriele Facciolo. SPHERE-JEPA: Spherical prediction with homogeneous embeddings. *arXiv preprint arXiv:2605.26900*, 2026.

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. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=KnqiC0znVF.

Hunter Nisonoff, Junhao Xiong, Stephan Allenspach, and Jennifer Listgarten. Unlocking guidance for discrete state-space diffusion and flow models. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=XsgHl54yO7.

Jingyang Ou, Shen Nie, Kaiwen Xue, Fengqi Zhu, Jiacheng Sun, Zhenguo Li, and Chongxuan Li. Your absorbing discrete diffusion secretly models the conditional distributions of clean data. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=sMyXP8Tanm.

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

Subham Sekhar Sahoo, Marianne Arriola, Yair Schiff, Aaron Gokaslan, Edgar Mariano Marroquin, Justin T Chiu, Alexander M Rush, and Volodymyr Kuleshov. Simple and effective masked diffusion language models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=L4uaAR4ArM.

Subham Sekhar Sahoo, Justin Deschenaux, Aaron Gokaslan, Guanghan Wang, Justin T Chiu, and Volodymyr Kuleshov. The diffusion duality. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=9P9Y8FOSOk.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. Winogrande: An adversarial winograd schema challenge at scale. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 8732–8740, 2020.

Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=TIdIXIpzhoI.

Yair Schiff, Subham Sekhar Sahoo, Hao Phung, Guanghan Wang, Sam Boshar, Hugo Dalla-torre, Bernardo P de Almeida, Alexander M Rush, Thomas Pierrot, and Volodymyr Kuleshov. Simple guidance mechanisms for discrete diffusion models. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=i5MrJ6g5G1.

Jiaxin Shi, Kehang Han, Zhe Wang, Arnaud Doucet, and Michalis Titsias. Simplified and generalized masked diffusion for discrete data. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=xcqSOfHt4g.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2021a. URL https://openreview.net/forum?id=St1giarCHLP.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021b. URL https://openreview.net/forum?id=PxTIG12RRHS.

Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 32211–32252. PMLR, 23–29 Jul 2023. URL https://proceedings.mlr.press/v202/song23a.html.

Yuxuan Song, Zheng Zhang, Cheng Luo, Pengyang Gao, Fan Xia, Hao Luo, Zheng Li, Yuehang Yang, Hongli Yu, Xingwei Qu, et al. Seed diffusion: A large-scale diffusion language model with high-speed inference. *arXiv preprint arXiv:2508.02193*, 2025.

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

Jaesung Tae, Hamish Ivison, Sachin Kumar, and Arman Cohan. TESS 2: A large-scale generalist diffusion language model. pp. 21171–21188, July 2025. doi: 10.18653/v1/2025.acl-long.1029. URL https://aclanthology.org/2025.acl-long.1029/.

Yunshu Wu, Jiayi Cheng, Longxuan Yu, Partha Thakuria, Rob Brekelmans, Evangelos E Papalexakis, and Greg Ver Steeg. Discrete stochastic localization for non-autoregressive generation. *arXiv preprint arXiv:2602.16169*, 2026.

Yanbo Xu, Yu Wu, Sungjae Park, Zhizhuo Zhou, and Shubham Tulsiani. Temporal score rescaling for temperature sampling in diffusion and flow models. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=WbfRe0cheB.

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

Tianwei Yin, Michaël Gharbi, Taesung Park, Richard Zhang, Eli Shechtman, Fredo Durand, and William T. Freeman. Improved distribution matching distillation for fast image synthesis. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024a. URL https://openreview.net/forum?id=tQukGCDaNT.

Tianwei Yin, Nanxin Luan, Prafulla Dhariwal, Yang Chen, Yang Song, and Ivan Lukic. One-step diffusion with distribution matching distillation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024b.

Tianwei Yin, Qiang Zhang, Richard Zhang, William T. Freeman, Fredo Durand, Eli Shechtman, and Xun Huang. From Slow Bidirectional to Fast Autoregressive Video Diffusion Models . In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 22963–22974, Los Alamitos, CA, USA, June 2025. IEEE Computer Society. doi: 10.1109/CVPR52734.2025.02138. URL https://doi.ieeecomputersociety.org/10.1109/CVPR52734.2025.02138.

Longxuan Yu, Yunshu Wu, Yu Fu, Siheng Xiong, Rob Brekelmans, Hui Liu, Yue Dong, and Greg Ver Steeg. DSL-LLaDA: Scaling continuous denoising to 8B masked diffusion LMs. *arXiv preprint arXiv:2606.01024*, 2026.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. HellaSwag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 4791–4800, Florence, Italy, July 2019. Association for Computational Linguistics. URL https://aclanthology.org/P19-1472/.

Qinsheng Zhang and Yongxin Chen. Fast sampling of diffusion models with exponential integrator. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=Loek7hfb46P.

Qinsheng Zhang, Molei Tao, and Yongxin Chen. gDDIM: Generalized denoising diffusion implicit models. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=1hKE9qjvz-.

Yiming Zhang, Harshita Diddee, Susan Holm, Hanchen Liu, Xinyue Liu, Vinay Samuel, Barry Wang, and Daphne Ippolito. Noveltybench: Evaluating language models for humanlike diversity. *arXiv:2504.05228*, 2025.

Lianmin Zheng, Liangsheng Yin, Zhiqiang Xie, Chuyue Sun, Jeff Huang, Cody Hao Yu, Shiyi Cao, Christos Kozyrakis, Ion Stoica, Joseph E. Gonzalez, Clark Barrett, and Ying Sheng. SGLang: Efficient execution of structured language model programs. In *Advances in Neural Information Processing Systems*, 2024. URL https://arxiv.org/abs/2312.07104.

Yuchen Zhu, Jing Shi, Chongjian Ge, Hao Tan, Yiran Xu, Wanrong Zhu, Jason Kuen, Koustava Goswami, Rajiv Jain, Yongxin Chen, et al. FLARE: Diffusion for hybrid language model. *arXiv preprint arXiv:2606.01774*, 2026.

Shaobin Zhuang, Yuang Ai, Jiaming Han, Xiaohui Li, Huaibo Huang, Xiangyu Yue, Xuefeng Hu, Kun Xu, Yali Wang, and Hao Chen. BitLM: Unlocking multi-token language generation with bitwise continuous diffusion. *arXiv preprint arXiv:2605.11577*, 2026.

Contents

A Related Work **21**

B Training and Inference Algorithms **23**

C Implementation Details and Additional Results for Pretraining **25**

C.1 Learning Regularized Embeddings 25

C.2 Ablation Study: MDLM Initialization for Pretraining 28

D Implementation Details and Additional Results for Likelihood-based Benchmarks **31**

E Implementation Details and Additional Results for Inference **33**

E.1 Sampler Updates 35

E.2 Ablation Study: Time Discretization 35

E.3 Reasoning and Coding Benchmark Setup 35

E.4 Sweeping Classifier-Free Guidance and Score Temperature 37

E.5 Ablation Study: Auto-Guidance with Self-Conditioning 37

E.6 Ablation Study: Steering in Logit Space instead of Embedding Space 37

E.7 Ablation Study: Commit-Stride Sampling 40

E.8 Pass@k Evaluation 42

E.9 Wall-Clock Cost of the Packed CFG Forward 43

E.10 Reasoning Trajectory Analysis 44

F Implementation Details and Additional Results for Few-step Distillation **46**

F.1 Method and Training Objective 46

F.2 Implementation Details 47

F.3 Additional Results 48

F.4 Proof: DMD Gradient Surrogate Loss 51

G Implementation Details and Additional Results for Supervised Fine-Tuning **54**

G.1 Supervised Fine-Tuning Setup 54

G.2 Evaluation Setup, CFG–ST Sweep, and Full Results 55

H Implementation Details and Additional Results for Self-Speculative Decoding **58**

H.1 Setup 58

H.2 Main Results in Full 58

H.3 Undistilled Model and Trained Drafters 58

H.4 Acceptance Training of the Distilled Model 59

A. Related Work

[Back to Sec. 1 or Sec. 2 in the main paper.]

This appendix provides a literature review on continuous diffusion language models, the family to which *Sigma* belongs.

Continuous diffusion language models. Continuous dLMs lift discrete tokens into a Euclidean space and run a Gaussian (or Riemannian) diffusion there. A central attraction for doing so is that the machinery developed for image diffusion, such as efficient ODE solvers, guidance, and distillation techniques, transfers with little modification. Existing methods differ mainly in *where* the diffusion process lives:

- Diffusing *token embeddings.* Diffusion-LM (Li et al., 2022) pioneered this approach with Gaussian diffusion over learned word embeddings and a rounding step back to tokens; CDCD (Dieleman et al., 2022) refined the objective with score interpolation and a categorical cross-entropy loss over softmax logits, and self-conditioned embedding diffusion (Strudel et al., 2022) added self-conditioning on the previous denoising estimate. Plaid (Gulrajani & Hashimoto, 2023) tied these ideas together around a likelihood target by pairing low-dimensional sphere-normalized embeddings with a learned noise schedule and categorical reparameterization; RePlaid (Yang et al., 2026) aligns this recipe with modern DiT architectures and shows it scales competitively with discrete dLMs. Meanwhile, LangFlow (Chen et al., 2026) follows the same learnable-embedding but uses cross-entropy training loss, while Wu et al. (2026) adopts a stochastic-localization formulation. $\mathbb{S}$-FLM (Deschenaux & Gulcehre, 2026) and Spherical Flows (Chemseddine et al., 2026) likewise use learnable token embeddings but further constrain the diffusion trajectory itself to the embedding hypersphere. A parallel strand fixes the embedding to be *one-hot* rather than learned: FLM (Lee et al., 2026) and DFM (Potaptchik et al., 2026) run flow matching from a Gaussian source to one-hot vocabulary targets in $\mathbb{R}^V$, and further distill the flow into a few-step flow map for fast generation.
- Diffusing an *autoencoded latent.* Here, a text autoencoder produces a compact continuous latent variable and diffusion is run in that latent space. LD4LG (Lovelace et al., 2023) runs latent diffusion on a frozen encoder–decoder (BART and T5); COSMOS (Meshchaninov et al., 2025), ELF (Hu et al., 2026) Cola-DLM (Guo et al., 2026), and AURORA-LM (Liang et al., 2026) scale this axis with different latent designs and training objectives; LDLM (Meshchaninov et al., 2026) learns the encoder and the diffusion prior jointly rather than sequentially.
- Diffusing on the *simplex or other statistical manifolds.* These methods parameterize a categorical distribution over the vocabulary and diffuse its parameters directly, keeping the noise process on-manifold and avoiding a rounding step. SSD-LM (Han et al., 2023) runs simplex diffusion in a semi-autoregressive block structure, Dirichlet diffusion (Avdeyev et al., 2023) uses a Jacobi diffusion process on the probability simplex, and later variants (Cheng et al., 2024; Davis et al., 2024; Jo & Hwang, 2025) extend the geometry to Fisher–Rao and Riemannian flow matching.
- Reaching discrete data through *dequantization or a Gaussian–discrete duality.* Argmax flows (Hoogeboom et al., 2021) lift categorical data into a continuous space via argmax-based variational dequantization, so continuous flows can be trained under a discrete likelihood; analog bits (Chen et al., 2023), BitLM (Zhuang et al., 2026), and CoBit (Batzolis et al., 2026) instead map each token to a bitstring and diffuse from there; Duo (Sahoo et al., 2025) establishes a duality between continuous Gaussian and uniform-state discrete diffusion, letting the two families share training and sampling machinery.

Sigma sits in the token-embedding class, inheriting the learnable embedding and variational diffusion process of Plaid and RePlaid.

Scaling continuous dLMs to billions of parameters. Recent work pushes continuous dLMs into the billion-parameter regime, but along axes different from ours: for example, Cola-DLM (Guo et al., 2026) (∼2B, hierarchical latent-prior modeling), AURORA-LM (Liang et al., 2026) (1B, flow matching over an autoencoded latent), and Categorical Flow Maps (Davis et al., 2026) (1.7B, self-distilled for few-step generation). A separate line changes the base representation and scales through an existing LM backbone: BitLM (Zhuang et al., 2026) places a bitwise diffusion head atop a causal LM up to 8B, TESS 2 (Tae et al., 2025) adapts an autoregressive Mistral-7B into a generalist diffusion model with a simplex cross-entropy loss, and DSL-LLaDA (Yu et al., 2026) scales continuous denoising via discrete stochastic localization (Wu et al., 2026) on top of the LLaDA-8B masked-diffusion backbone (Nie et al., 2025). In contrast, *Sigma* is, to our knowledge, the first likelihood-trained continuous dLM at 3B/8B, and the first evaluated head-to-head with a matched masked

dLM (Fu et al., 2026) on conditional reasoning and code.

CFG in dLMs. Continuous dLMs realize the guidance condition in two ways. When there is no prompt (e.g., unconditional generation), several works repurpose the self-conditioning signal as the condition: ELF (Hu et al., 2026) and AURORA-LM (Liang et al., 2026) extrapolate between predictions made with and without the previous-step estimate. For conditional generation, DFM (Potaptchik et al., 2026), AURORA-LM (Liang et al., 2026), Cola-DLM (Guo et al., 2026), and BitLM (Zhuang et al., 2026) extrapolate between a prompt-conditioned and an unconditional prediction, as in image diffusion models. A cost for this is that enabling it requires dropping the prompt context during training. FLM (Lee et al., 2026) also leverages autoguidance (Karras et al., 2024) for prediction sharpening. For discrete dLMs, LLaDA (Nie et al., 2025) popularized an "unsupervised" CFG that runs one forward with full prompt and one with the prompt masked, then linearly extrapolates the two logit vectors before renormalizing; this has since been refined by more general discrete guidance mechanisms (Schiff et al., 2025; Nisonoff et al., 2025). Our steering (Sec. 4.2) differs on three points: (*i*) the guidance condition is the clean prefix itself, and the unconditional branch simply drops that prefix so the noisy block attends only within itself; (*ii*) the extrapolation acts on the predicted clean embedding (equivalently the score), not on logits; (*iii*) it is a pure inference-time operation that requires no train-time dropout because sequence packing leaves the first block without context.

Autoregressive and block video diffusion. Our block-causal factorization (i.e., autoregressive across blocks, bidirectional Gaussian denoising within a block over a KV-cacheable clean prefix) closely mirrors recent video generation models that couple causal structure across frames with denoising within them (Chen et al., 2024; Yin et al., 2025; Huang et al., 2025). The bidirectional video-diffusion lineage (Ho et al., 2022) was the original target of this conversion, and the autoregressive arm of Cosmos (Agarwal et al., 2025) likewise structures video generation causally across frames. Diffusion Forcing (Chen et al., 2024) trains a causal model with independent per-frame noise levels, unifying next-token prediction with full-sequence diffusion and enabling stable rollouts past the training horizon. CausVid (Yin et al., 2025) and Self Forcing (Huang et al., 2025) convert bidirectional video diffusion into causal, KV-cached autoregressive generators for fast streaming synthesis while curbing error accumulation along the rollout. These works share our central design tension (i.e., coupling causal, autoregressive structure with within-block diffusion) and several of their techniques (per-block noise schedules, rolling KV caches, distribution-matching distillation) transfer naturally to the language setting.

B. Training and Inference Algorithms

[Back to Sec. 3.2 or Sec. 4 in the main paper.]

We collect the two procedures described in the main paper:

- Alg. 1 is one optimization step of the blockwise NELBO (3) and (4) under the Horvitz–Thompson estimator (5) (Sec. 3.1);
- Alg. 2 is blockwise reverse sampling (Sec. 4.1) with CFG, ST, SC, and an optional commit stride (Sec. 4.2).

Algorithm 1 *Sigma* training step for one sequence

Require: sequence $\mathbf{x} \in \{0,1\}^{L\times V}$; block size L', number of blocks $B = L/L'$; denoiser parameters θ (transformer, heads, and up-projections); embedding matrix $\mathbf{E} \in \mathbb{R}^{V\times d_e}$; noise schedule endpoints $\gamma_0 < \gamma_1$ and monotone interior shape $\tilde{\gamma}_\psi : [0,1] \to [0,1]$, giving $\gamma(t) = \gamma_0 + (\gamma_1 - \gamma_0)\tilde{\gamma}_\psi(t)$; reconstruction probability $p_{\text{rec}} \in (0,1)$ (default 0.1); self-conditioning probability $p_{\text{sc}} \in (0,1)$ (default 0.25); diffusion loss weight $w_{\text{diff}} > 0$ and optional AR-loss weight $w_{\text{ar}} \geq 0$; optional embedding regularizer $s_\star \in \{s, s_{\text{MS}}\}$ with weight $\lambda_{\text{susreg}} \geq 0$; optional repulsion weight $\lambda_{\text{rep}} \geq 0$ with radius r computed from $\text{sg}(\gamma_0)$ (settings in App. C.1)

$\mathbf{e} \leftarrow \mathbf{x}\mathbf{E} \in \mathbb{R}^{L\times d_e}$ ▷ Embedding representation, each row has unit-norm
for $b = 1, \ldots, B$ **do** ▷ independent noise level and estimator branch per block
  $R^{(b)} \sim \text{Ber}(p_{\text{rec}})$
  $t^{(b)} \leftarrow 0$ **if** $R^{(b)} = 1$ **else** $t^{(b)} \sim \text{Unif}[0,1]$
  $\boldsymbol{\epsilon}^{(b)} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, $\mathbf{z}^{(b)} \leftarrow \alpha_{t^{(b)}}\mathbf{e}^{(b)} + \sigma_{t^{(b)}}\boldsymbol{\epsilon}^{(b)}$ ▷ Gaussian noising (1)
end for
$S \sim \text{Ber}(p_{\text{sc}})$, $\mathbf{e}_{\text{sc}} \leftarrow \mathbf{0}$
if $S = 1$ **then** ▷ detached SC pass
  $\mathbf{e}_{\text{sc}} \leftarrow \text{sg}\left(\mathbf{x}_\theta(\{\mathbf{z}^{(b)}\}, \{t^{(b)}\}; \mathbf{x}, \mathbf{0})\,\mathbf{E}\right)$
end if
$\mathbf{x}_\theta \leftarrow \mathbf{x}_\theta(\{\mathbf{z}^{(b)}\}, \{t^{(b)}\}; \mathbf{x}, \mathbf{e}_{\text{sc}})$, $\mathbf{e}_\theta \leftarrow \mathbf{x}_\theta\mathbf{E}$ ▷ one forward for all blocks; block-causal mask
for $b = 1, \ldots, B$ **do**
  $\mathcal{L}^{(b)}_{\text{rec}} \leftarrow -\sum_{l=1}^{L'} \log\langle \mathbf{x}_\theta^{(b,l)}, \mathbf{x}^{(b,l)}\rangle$ ▷ used only when $R^{(b)} = 1$, i.e., $t^{(b)} = 0$
  $\mathcal{L}^{(b)}_{\text{diff}} \leftarrow -\frac{1}{2}\,\text{SNR}'(t^{(b)})\,\|\mathbf{e}_\theta^{(b)} - \mathbf{e}^{(b)}\|^2$ ▷ used only when $R^{(b)} = 0$
  $\mathcal{L}^{(b)}_{\text{prior}} \leftarrow \frac{L'}{2}\left[d_e(\sigma_1^2 - 1 - \log\sigma_1^2) + \alpha_1^2\right]$ ▷ closed form of $\text{KL}(q(\mathbf{z}_1^{(b)} \mid \mathbf{x}^{(b)}) \,\|\, p(\mathbf{z}_1^{(b)}))$
  $\mathcal{L}^{(b)} \leftarrow \frac{1-R^{(b)}}{1-p_{\text{rec}}}\mathcal{L}^{(b)}_{\text{diff}} + \frac{R^{(b)}}{p_{\text{rec}}}\mathcal{L}^{(b)}_{\text{rec}} + \mathcal{L}^{(b)}_{\text{prior}}$ ▷ Horvitz–Thompson estimator (5)
end for
$\mathcal{L}_{\text{BD}} \leftarrow \sum_{b=1}^{B}\mathcal{L}^{(b)}$ ▷ block-diffusion NELBO estimator (3)
if $\lambda_{\text{susreg}} > 0$ **then**
  $\mathcal{L}_{\text{BD}} \leftarrow \mathcal{L}_{\text{BD}} + \lambda_{\text{susreg}}\,L\,s_\star(\mathbf{E})$ ▷ *Sigma-preg*: $\lambda_{\text{susreg}} = 0.005$, $s_\star = s$; *Sigma*: $\lambda_{\text{susreg}} = 1$, $s_\star = s_{\text{MS}}$
end if
if $\lambda_{\text{rep}} > 0$ **then**
  $\mathcal{L}_{\text{BD}} \leftarrow \mathcal{L}_{\text{BD}} + \lambda_{\text{rep}}\,L\,s_{\text{rep}}(\mathbf{E})$ ▷ local repulsion loss (11), used by *Sigma* only
end if
$\mathcal{L} \leftarrow w_{\text{diff}}\,\mathcal{L}_{\text{BD}} + w_{\text{ar}}\,\mathcal{L}_{\text{ar}}(\mathbf{x})$ ▷ joint objective (Sec. 3.2.2)
$\mathcal{L}_{\text{var}} \leftarrow \frac{(1-S)w_{\text{diff}}}{(1-p_{\text{rec}})L'}\sum_{b\,:\,R^{(b)}=0}(\mathcal{L}^{(b)}_{\text{diff}})^2$ ▷ Noise schedule interior shape learned to minimize MC variance
update θ with $\nabla_\theta\mathcal{L}$ ▷ backbone group, peak lr 10^{-5}
update $\mathbf{E}$ with $\nabla_{\mathbf{E}}\mathcal{L}$ ▷ Riemannian Adam for *Sigma*, AdamW otherwise; no weight decay
update (γ_0, γ_1) with $\nabla_{(\gamma_0,\gamma_1)}\mathcal{L}$ ▷ AdamW, weight decay 0.1; ordering projection for *Sigma*
update ψ with $\nabla_\psi\mathcal{L}_{\text{var}}$ ▷ projected Adam for *Sigma*, AdamW otherwise; no weight decay

At training, all B blocks are denoised in a *single* forward pass: the offset block-causal mask of Fig. 2b makes block b's noisy latent attend to the clean prefix $\mathbf{x}^{(<b)}$ and bidirectionally within itself, so a per-block time $t^{(b)}$ and a per-block Horvitz–Thompson coin $R^{(b)}$ cost no extra compute. At inference, the conditional and unconditional branches are likewise packed into one forward as in Fig. 5a, so the guidance computation costs slightly more wall-clock time per step (Tab. 19) at batch size one.

The losses are sequence sums, so per-token embedding penalties carry a factor L. For MS-SUSReg, $\lambda_{\text{susreg}} = 1$ because s_{MS} already includes the scale weights. SC-selected sequences stop NELBO gradients to the diffusion

embedding and schedule while retaining regularizer gradients. MS-SUSReg balances the 25% SC fraction across each global batch.

We further clarify on the learning of the noise schedule, since its two parts are optimized for two different objectives. The endpoints γ_0, γ_1 enter $\mathcal{L}_{\text{diff}}$ through $\text{SNR}'(t)$ and (α_t, σ_t), so they receive an ordinary NELBO gradient and are trained to tighten the bound while the interior shape $\tilde{\gamma}_\psi$ does not. As noted in Sec. 2, the diffusion term is invariant to it in expectation, so $\nabla_\psi \mathcal{L}$ is zero on average and carries only noise. Following Kingma et al. (2021), ψ is instead trained to minimize the Monte-Carlo variance of the diffusion estimator, which for a mean-invariant estimator reduces to minimizing its second moment $\mathcal{L}_{\text{var}}$ in Alg. 1. In code, each block's schedule gradient is scaled by twice its detached mean diffusion loss; reconstruction blocks and SC-selected sequences do not update the shape. Variant-specific optimizer and learning-rate settings are given in App. C.1.

Algorithm 2 Blockwise reverse sampling of *Sigma* with CFG, ST, SC, and commit stride

Require: prompt $\mathbf{p}$; block size L'; commit stride $S \in \{1, \dots, L'\}$ ($S = L'$ recovers plain block decoding, App. E.7); total generation length T_{gen}; steps N with discretization $1 = t_0 > t_1 > \cdots > t_N = 0$ (App. E.2); guidance weight $w \geq 1$; score temperature $\tau \in (0, 1]$; sampler $\in \{\text{DDPM}, \text{DDIM}\}$

$\mathbf{c} \leftarrow \mathbf{p}$; populate the causal KV cache from $\mathbf{p}$; $b \leftarrow 0$
while $|\mathbf{r}| < T_{\text{gen}}$ **do** ▷ $|\mathbf{r}|$: number of tokens committed so far
  $b \leftarrow b + 1$
  $\mathbf{z}_{t_0}^{(b)} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}) \in \mathbb{R}^{L' \times d_e}$, $\mathbf{e}_{\text{sc}} \leftarrow \mathbf{0}$
  for $i = 1, \dots, N$ **do**
    $t \leftarrow t_{i-1}$, $s \leftarrow t_i$
    $(\mathbf{e}_{\text{c}}, \mathbf{e}_{\varnothing}) \leftarrow (\mathbf{e}_\theta(\mathbf{z}_t^{(b)}, t; \mathbf{c}, \mathbf{e}_{\text{sc}}),\ \mathbf{e}_\theta(\mathbf{z}_t^{(b)}, t; \varnothing, \mathbf{0}))$ ▷ one packed forward, Fig. 5a
    $\mathbf{e}_{\text{sc}} \leftarrow \mathbf{e}_{\text{c}}$ ▷ SC reuses the *unsteered* conditional mean, Fig. 5b
    $\hat{\mathbf{e}}_\theta \leftarrow \frac{w}{\tau}\mathbf{e}_{\text{c}} + \frac{1-w}{\tau}\mathbf{e}_{\varnothing} + \left(1 - \frac{1}{\tau}\right)\frac{\mathbf{z}_t^{(b)}}{\alpha_t}$ ▷ Embedding mean prediction after CFG & ST
    if sampler is DDIM **then**
      $\hat{\boldsymbol{\epsilon}}_\theta \leftarrow \frac{1}{\sigma_t}(\mathbf{z}_t^{(b)} - \alpha_t \hat{\mathbf{e}}_\theta)$, $\mathbf{z}_s^{(b)} \leftarrow \alpha_s \hat{\mathbf{e}}_\theta + \sigma_s \hat{\boldsymbol{\epsilon}}_\theta$ ▷ (16)
    else
      $\kappa \leftarrow 1 - \mathrm{e}^{\gamma(s) - \gamma(t)}$, $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
      $\mathbf{z}_s^{(b)} \leftarrow (1 - \kappa)\frac{\alpha_s}{\alpha_t}\mathbf{z}_t^{(b)} + \kappa\, \alpha_s \hat{\mathbf{e}}_\theta + \sqrt{\kappa(1 - \alpha_s^2)}\, \boldsymbol{\epsilon}$ ▷ (15)
    end if
  end for
  $k_b \leftarrow \min(S,\ T_{\text{gen}} - |\mathbf{r}|)$ ▷ tokens to commit: the last window may commit fewer than S
  $\mathbf{r}^{(b,l)} \leftarrow \text{argmax}_v\, \mathbf{x}_\theta^{l,v}(\mathbf{z}_{t_N}^{(b)}, t_N; \mathbf{c}, \mathbf{e}_{\text{sc}})$, $l \in \{1, ..., k_b\}$ ▷ argmax decoding
  $\mathbf{c} \leftarrow (\mathbf{c}, \mathbf{r}^{(b)})$; append the KV of $\mathbf{r}^{(b)}$ to the causal cache; discard the tail ▷ the $L' - k_b$ tail positions are re-drawn as fresh noise in the next window
end while
return $\mathbf{r} = (\mathbf{r}^{(1)}, \dots, \mathbf{r}^{(b)})$

The outer loop of Alg. 2 runs $B := \lceil T_{\text{gen}}/S \rceil$ times, and each window costs $N + 1$ denoiser forward passes (N diffusion steps plus one decoding readout), for a total of $B(N + 1)$ forward passes. The effective NFE per committed token is thus $\approx (N + 1)/S$, see App. E.7 for a full compute-vs-quality sweep. Setting $w = 1$ disables CFG, $\tau = 1$ disables ST, and never updating $\mathbf{e}_{\text{sc}}$ (i.e., leaving it as $\mathbf{0}$) disables SC; each of these recovers a strictly weaker sampler, as shown in Fig. 15.

C. Implementation Details and Additional Results for Pretraining

[Back to Sec. 3.2 in the main paper.]

Architecture. Following NLD (Fu et al., 2026), we adopt the transformer architecture of Ministral (Liu et al., 2026a):

- 3B: 26 layers, width 3072, FFN 9216, 32 query and 8 key/value heads (head dimension 128), SwiGLU, RMSNorm, YaRN-scaled rotary embeddings ($\theta = 10^6$), and untied embeddings. This adds up to 3.0B non-embedding and 3.8B total parameters.
- 8B: 34 layers, width 4096, FFN 14336, same 32 query and 8 key/value heads (head dimension 128) and same SwiGLU, RMSNorm, YaRN rotary embeddings and untied embeddings. This adds up to 7.4B non-embedding and 8.5B total parameters.

Optimization. For *Sigma-unreg*/*Sigma-preg*, $\tilde{\gamma}$ and $\mathbf{E}$ use AdamW with $\beta_1 = 0.9$, $\beta_2 = 0.95$, no weight decay (rows of $\mathbf{E}$ unit-norm), and peak learning rate $\eta_{\text{peak}} = 1 \times 10^{-2}$; γ_0, γ_1 use the same setting but with weight decay 0.1 (Yang et al., 2026). For other parameters, we follow NLD and use AdamW with $\beta_1 = 0.9$, $\beta_2 = 0.95$, weight decay 0.1, and $\eta_{\text{peak}} = 1 \times 10^{-5}$. For all parameters, we use gradient clipping value 1.0. The learning rate schedule is Warmup-Stable-Decay over 150k steps: 1% linear warmup to η_{peak}, constant through step 120k, then exponential decay to 10% of η_{peak}.

Self-conditioning (SC). SC injects the model's own detached clean-embedding prediction into the projected noised input stream with a bias-free, zero-initialized linear layer mapping $\mathbb{R}^{d_e}$ to the hidden width $\mathbb{R}^d$: a first pass with $\mathbf{e}_{\text{sc}} = \mathbf{0}$, then a second pass with $\sqrt{d_e}\ \text{sg}(\mathbf{e}_{\text{sc}})$ through that linear layer.

Time conditioning. For diffusion stream, a single γ is sampled per block and shared across its tokens. We encode γ using fixed sinusoidal features with frequencies $\exp(\text{linspace}(-5, 5, 32))$, concatenating $[\sin, \cos]$ into 64 dimensions. A bias-free, zero-initialized linear layer projects this encoding to the hidden width, where it is added to the projected noised input.

Numerical precision. Transformer weights and matrix multiplications, including the linear logit heads, use `BF16`; gradient accumulation and communication use `FP32`. The diffusion embedding, noise schedule, gamma endpoints, and input/time/SC projections are stored in `FP32` for MS-SUSReg and `BF16` for *Sigma-unreg*/*Sigma-preg*. Embedding normalization, corruption, input/time/SC projections, and logit postprocessing compute in `FP32`; projected inputs are cast to `BF16` for the transformer. Schedule arithmetic and NELBO accumulation use `FP64`, following Gulrajani & Hashimoto (2023); Yang et al. (2026). Adam moments and master copies of `BF16` weights are `FP32`. Optimizer state is sharded across data-parallel ranks, except for MS-SUSReg's replicated schedule and endpoint states.

AR regularization. For training *Sigma-unreg*-3B, we searched over $w_{\text{diff}} \in \{0.3, 1.0\}$. Interestingly, when $w_{\text{diff}} = 1.0$, adding the AR regularization even improved diffusion PPL (Fig. 3c). However, we found that switching to $w_{\text{diff}} = 0.3$ at 200B tokens (equivalently, 100k steps) offers the best convergence for both losses (Fig. 3c). We thus adopted this schedule ($w_{\text{diff}} = 1.0$ for the first 100k steps and $w_{\text{diff}} = 0.3$ for the last 50k steps) for all of our *Sigma-unreg*-3B/8B and *Sigma-preg*-3B/8B models (see App. C.1). For *Sigma*-3B/8B, we keep $w_{\text{diff}} = 1.0$ all the way through 150k steps for simplicity of the training objective.

C.1. Learning Regularized Embeddings

[Back to Sec. 4.3 in the main paper.]

The central goal for our embedding-regularized model variants is to improve the low-NFE performance and distillability by improving the structure of the embeddings. To this end, we develop the toolbox described in the following: A multiscale variant of Sketched Uniform Spherical Regularization (SUSReg) and a local repulsion loss to combat global and local embedding collapse; a modified Riemannian Adam optimizer step for the embedding table that respects the per-row hyperspherical geometry; a simpler noise schedule parameterization based on Bernstein polynomials that can be trivially initialized to a linear (log-SNR) shape, together with a custom projected optimizer step; and a cosine learning rate schedule for the embeddings to aid their stable learning.

Sketched Uniform Spherical Regularization (SUSReg). To avoid the $V = 131072$ embedding vectors in $d_e = 16$ dimensional unit sphere being too tightly clustered for accurate decoding (as observed in their

low-rank structure in Yang et al. (2026)), we propose to add an optional regularization term based on SUSReg (Nicollier et al., 2026) to the embeddings, which regularizes their distribution $p_{\mathbf{E}} := \frac{1}{V}\sum_{v=1}^{V}\delta_{\mathbf{e}_v}$ toward $\mathrm{Unif}(\mathbb{S}^{d_e-1})$, the uniform distribution on the unit sphere in $\mathbb{R}^{d_e}$.

For M directions $\mathbf{a}_m \sim \mathrm{Unif}(\mathbb{S}^{d_e-1})$, we compare the empirical characteristic function of each one-dimensional projection,

$$\phi_{\mathbf{a}_m}(t) = \frac{1}{V}\sum_{v} \mathrm{e}^{\mathrm{i}t\langle \mathbf{a}_m, \mathbf{e}_v\rangle},$$

to the target characteristic function of a one-dimensional projection of $\mathrm{Unif}(\mathbb{S}^{d_e-1})$, whose marginal density is known in closed form:

$$\phi_{d_e}(t) = \int_{-1}^{1} \cos(tx)\rho_{d_e}(x)\,\mathrm{d}x, \quad \rho_{d_e}(x) = \frac{\Gamma(\frac{d_e}{2})}{\sqrt{\pi}\Gamma(\frac{d_e-1}{2})}(1-x^2)^{\frac{d_e-3}{2}}\mathbf{1}_{|x|\le 1},$$

under an Epps–Pulley Gaussian-weighted L^2 metric:

$$s(\mathbf{E}) := \frac{V}{M}\sum_{m=1}^{M}\int_{-t_{\max}}^{t_{\max}} \left|\phi_{\mathbf{a}_m}(t) - \phi_{d_e}(t)\right|^2 \mathrm{e}^{-t^2/2}\,\mathrm{d}t. \tag{7}$$

The multiplier V compensates for the $1/V$ scaling of the expected squared characteristic-function discrepancy under i.i.d. uniform embeddings. We add $\lambda_{\text{susreg}}s(\mathbf{E})$ per token on the diffusion arm, contributing $w_{\text{diff}}\lambda_{\text{susreg}}s(\mathbf{E})$ to the joint objective (Alg. 1). Plain SUSReg-3B/8B uses $\lambda_{\text{susreg}} = 0.005$, $M = 512$ freshly sampled directions, and $P = 17$ trapezoidal points over $[-5, 5]$. The target characteristic function is cached using 512-point numerical quadrature. The $O(VM(d_e + P))$ cost is negligible relative to the transformer.

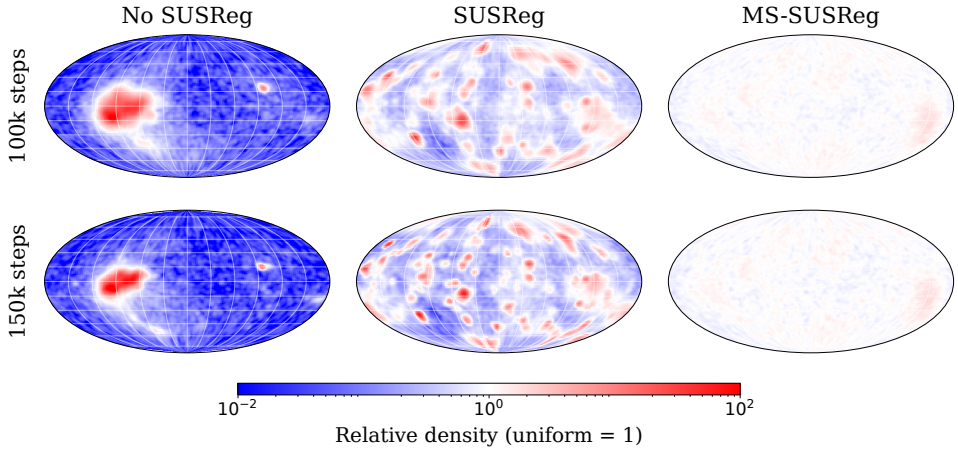


Figure 10 | Density plot of a random $\mathbb{S}^{15} \to \mathbb{S}^2$ Mollweide projection of all learned embeddings living on $\mathbb{S}^{15}$. The baseline collapses to a small subregion of the 16D space, while SUSReg distributes density more evently, but still contains many concentrated clusters. MS-SUSReg achieves a comparatively much more uniform distribution. Note that while the MS-SUSReg cloud looks uniform in density, it is far from simple random initialization, see the conceptual hierarchical clustering shown in Fig. 11. The PCA effective dimensions of three embedding matrices at 150k steps are $3.4008, 15.9593, 15.9986$, respectively.

Multiscale-SUSReg (MS-SUSReg). The SUSReg regularizer (7) contains the Gaussian kernel $\mathrm{e}^{-t^2/2}$, which corresponds to a specific low-pass weighting of the frequencies in the compared characteristic functions. This means the loss is more sensitive to larger-scale structural mismatches (global uniformity) than to the

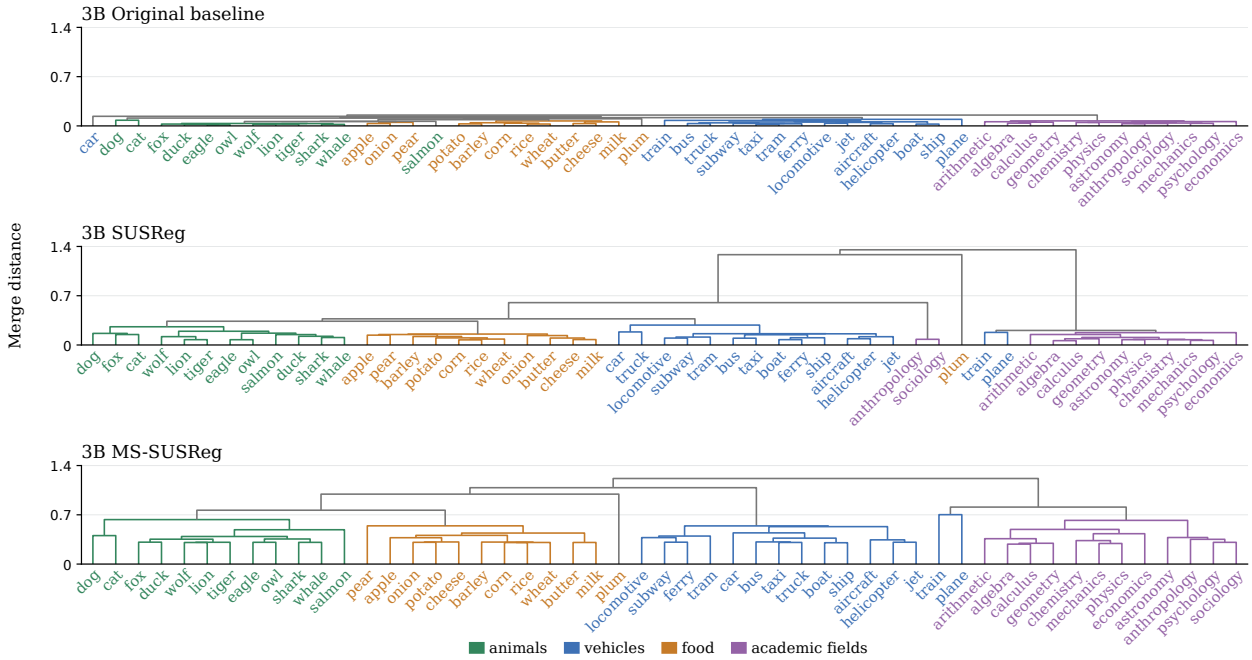


Figure 11 | Dendrogram on a set of single-token example words, constructed by bottom-up average-linkage clustering of the selected unit-normalized embedding rows in the 3B model variants (Unregularized baseline, SUSReg, MS-SUSReg), using Euclidean distances, with the cluster merge distance shown on the y-axis. Each shown word represents the lowercase token form prefixed by a single space, e.g., `" dog"`. **Top:** The unregularized embedding table exhibits very low clustering distances, consistent with subspace collapse, and also contains some counterintuitive clusterings that mix categories. **Middle:** The SUSReg run greatly improves upon both issues, but most merges still occur at low distances and some unusual groupings (train, plum) remain. **Bottom:** The MS-SUSReg run exhibits the cleanest clustering in this example, corresponding well to natural categories, and also shows larger merge distances already at lower levels of the hierarchy.

finer scales (local clustering). We generalize this by introducing a kernel width parameter $\beta > 0$:

$$s_\beta(\mathbf{E}) := \frac{V}{M}\sum_{m=1}^{M}\int_{-\beta t_{\max}}^{\beta t_{\max}} \frac{1}{\beta}\left|\phi_{\mathbf{a}_m}(t) - \phi_{d_e}(t)\right|^2 \mathrm{e}^{-t^2/(2\beta^2)}\,\mathrm{d}t \tag{8}$$

$$= \frac{V}{M}\sum_{m=1}^{M}\int_{-t_{\max}}^{t_{\max}} \left|\phi_{\mathbf{a}_m}(\beta u) - \phi_{d_e}(\beta u)\right|^2 \mathrm{e}^{-u^2/2}\,\mathrm{d}u \tag{9}$$

where the second form is from a simple change of variables with $u = t/\beta$. The original SUSReg loss (7) corresponds to $\beta = 1$. To regularize both the global and the local structure, we can then sum K loss terms like (8) with different weights w_k and kernel widths β_k:

$$s_{\mathrm{MS}}(\mathbf{E}) := \frac{V}{M}\sum_{m=1}^{M}\sum_{k=1}^{K} w_k \int_{-t_{\max}}^{t_{\max}} \left|\phi_{\mathbf{a}_m}(\beta_k t) - \phi_{d_e}(\beta_k t)\right|^2 \mathrm{e}^{-t^2/2}\,\mathrm{d}t \tag{10}$$

which we call the *Multiscale-SUSReg* (MS-SUSReg) loss. Its scale weights w_k are absolute coefficients: the per-token objective contains $w_{\mathrm{diff}} s_{\mathrm{MS}}(\mathbf{E})$, without an additional SUSReg multiplier. Each bandwidth uses $P_k = 1 + \lceil \beta_k (P-1) \rceil$ quadrature points to preserve frequency resolution: 17 and 257 points for $\beta = 1$ and 16. The scales share $M = 512$ random directions and embedding projections, use $t_{\max} = 5$, and are active from step zero. Cached targets give a total cost of $O(VM(d_e + \sum_k P_k))$.

Local repulsion regularization. MS-SUSReg encourages distributional uniformity through sampled projections, but does not explicitly enforce separation between individual embedding pairs. Nearly coincident embeddings can therefore remain difficult to distinguish under diffusion noise. To address this directly, we

add a local separation penalty related to maximum hyperspherical separation (Liu et al., 2021). Namely, we add a per-token weighted loss term $w_{\text{diff}}\lambda_{\text{rep}}s_{\text{rep}}(\mathbf{E})$, where

$$s_{\text{rep}}(\mathbf{E}) := \frac{1}{2V}\sum_{u<v}\max\{0, r - \|\mathbf{e}_u - \mathbf{e}_v\|_2\}^2 \tag{11}$$

penalizes pairwise chord distances below an exclusion radius r. We set $r = \max\{\sqrt{2(1-c)}, 2\Phi^{-1}(1-p)\mathrm{e}^{\gamma_0/2}\}$, where Φ is the standard normal CDF, and $c = 0.9995$, $p = 0.01$, and $\lambda_{\text{rep}} = 20$. Unlike maximum hyperspherical separation, which maximizes the smallest pairwise distance, our penalty acts on all pairs within r. The adaptive term depending on γ_0 defines a target separation corresponding to a per-competitor confusion probability of p under the diffusion forward marginal distribution at the minimum noise level γ_0. We disable gradients through r here so this loss only affects the embeddings and not γ_0. We evaluate all unordered pairs in tiles across data-parallel workers without storing the full pairwise matrix. Both embedding regularizers act from step zero, including on SC-selected sequences, and are excluded from reported NELBO/perplexity.

Riemannian Adam step. Since the effective embeddings have unit norm, their geometry depends only on their directions. The naive embedding representation, which normalizes each unconstrained row on the fly when used, creates a per-row dependence of the effective learning rate on arbitrary underlying pre-normalization row norms. We find that these row norms grow over training, resulting in an uncontrolled LR decay on the embedding rows. With the following Riemannian Adam step, we optimize the embedding rows directly on the sphere to make every update respect the on-sphere constraint and avoid this uncontrolled influence on the embedding LR. For each row $\mathbf{e}_v$, we project its total training-loss gradient $\mathbf{g}_v$ onto the tangent space at $\mathbf{e}_v$ as $\mathbf{g}_v^\perp = \mathbf{g}_v - \langle \mathbf{g}_v, \mathbf{e}_v\rangle \mathbf{e}_v$. Similar to Becigneul & Ganea (2019), we adapt the step size separately for each embedding vector. We apply a bias-corrected on-manifold Adam variant using a tangent first moment and one scalar second moment per row, estimated from $\|\mathbf{g}_v^\perp\|_2^2/d_e$. This scalar second moment rescales all tangent directions equally. After the update, we project the row back onto the sphere by normalization and parallel-transport the first moment along the shortest spherical geodesic to the new tangent space, preserving momentum as the embedding direction changes.

Projected noise-schedule updates. Our main regularized model (*Sigma*-3B/8B) uses a degree-32 Bernstein polynomial for the noise schedule shape with increments $\psi_i \geq 10^{-3}/32$, $\sum_i \psi_i = 1$:

$$\tilde{\gamma}_\psi(t) = \sum_{j=0}^{32} c_j \binom{32}{j} t^j (1-t)^{32-j}, \qquad c_0 = 0, \quad c_j = \sum_{i=1}^{j}\psi_i.$$

Initializing $\psi_i = 1/32$ gives a linear schedule. Adam updates on the variance objective are projected onto this simplex; endpoint AdamW updates use weight decay 0.1 and enforce $\gamma_1 - \gamma_0 \geq 10^{-3}$. The embedding and shape have no weight decay. All three groups use Adam coefficients $(\beta_1, \beta_2) = (0.9, 0.95)$ and $\epsilon = 10^{-8}$.

Training details for the regularized models. *For the plain SUSReg ablation models (Sigma-preg-3B/8B):*[3] we add the described SUSReg loss in (7) and freeze the embeddings and schedule parameters after 100k steps, but otherwise follow the original model's training recipe including the change to $w_{\text{diff}} = 0.3$ at 100k steps. *For MS-SUSReg (Sigma-3B/8B):* we use $(\beta_1, w_1) = (1, 10^{-2})$, $(\beta_2, w_2) = (16, 10^{-4})$, and repulsion weight $\lambda_{\text{rep}} = 20$. The embedding uses Riemannian Adam; the schedule uses the projected updates above. Their learning rates warm up to 3×10^{-3} over 1.5k steps, decay by cosine to 10^{-4} at 30k, remain constant through 120k, then follow WSD decay to 10^{-5} at 150k. Embeddings and schedules remain trainable throughout, with $w_{\text{diff}} = 1$. Global gradient-norm clipping is 3 for 3B and 1 for 8B.

Full pretrained results for all three variants. Tab. 6 reports the six generative reasoning tasks of Tabs. 1 and 2 for all three pretrained variants at both scales. In diffusion mode, plain SUSReg is roughly neutral against *Sigma-unreg* at DDPM-256 but gains 1.40 (3B) and 0.79 (8B) average points at DDIM-32, consistent with our hypothesis that uniform embeddings mainly buy robustness to coarse discretization; MS-SUSReg improves on both at every operating point and scale.

C.2. Ablation Study: MDLM Initialization for Pretraining

[Back to Sec. 3.2 in the main paper.]

[3] "*preg*" stands for plain SUSReg.

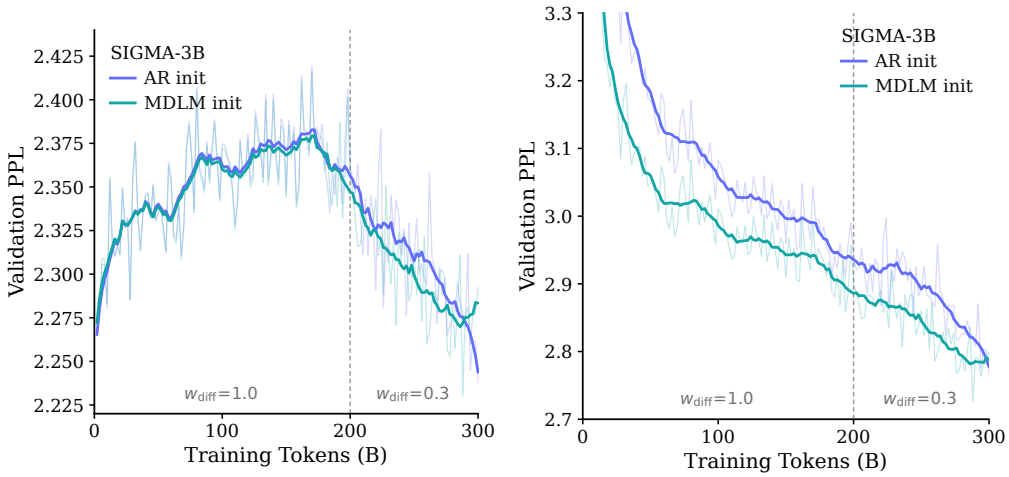
Table 6 | **Pass@1 accuracy of all three pretrained *Sigma* variants at 3B and 8B.** Same six generative reasoning tasks, harness, and operating points as Tabs. 1 and 2. *Sigma* diffusion rows are mean (over 10 sampling lanes) ± two standard errors. **Bold** marks the best value per task within each mode and model size.

	Math		Coding				
Method	GSM8K	Minerva Math	HumanEval	HumanEval+	MBPP	MBPP+	Avg.
3B models							
AR initialization	83.70	66.28	51.22	48.78	60.60	77.78	64.73
AR mode							
Nemotron-Labs-Diffusion-3B	**83.78**	65.06	53.05	47.56	**60.80**	**78.84**	64.85
Sigma-unreg-3B	83.40	65.10	**57.32**	**51.83**	59.00	76.19	**65.47**
Sigma-preg-3B	82.18	**65.64**	53.66	49.39	60.00	77.78	64.77
Sigma-3B	81.20	63.98	51.83	48.17	56.80	75.66	62.94
Diffusion mode							
Nemotron-Labs-Diffusion-3B	83.40	63.96	56.10	53.05	57.60	**74.34**	64.74
Sigma-unreg-3B							
DDPM 256 steps/block (w=2, τ=0.1)	84.50 (± 0.31)	64.60 (± 0.28)	**59.45** (± 1.30)	**53.84** (± 1.29)	52.32 (± 0.73)	71.32 (± 0.91)	64.34
DDIM 32 steps/block (w=3, τ=0.3)	81.91 (± 0.32)	62.76 (± 0.18)	56.59 (± 1.31)	50.67 (± 1.00)	47.26 (± 0.46)	63.49 (± 0.77)	60.45
Sigma-preg-3B							
DDPM 256 steps/block (w=3, τ=0.2)	84.61 (± 0.20)	63.82 (± 0.39)	56.59 (± 1.38)	50.43 (± 1.02)	56.34 (± 0.70)	73.52 (± 0.99)	64.22
DDIM 32 steps/block (w=4, τ=0.4)	83.17 (± 0.30)	63.28 (± 0.15)	55.24 (± 1.36)	50.91 (± 0.99)	51.64 (± 0.57)	66.88 (± 0.29)	61.85
Sigma-3B							
DDPM 256 steps/block (w=3, τ=0.1)	**85.57** (± 0.29)	**64.98** (± 0.19)	57.26 (± 0.82)	**53.84** (± 0.55)	**58.00** (± 0.43)	73.62 (± 0.51)	**65.55**
DDIM 32 steps/block (w=3, τ=0.3)	84.48 (± 0.21)	64.52 (± 0.21)	55.37 (± 1.28)	51.71 (± 1.23)	53.00 (± 0.34)	71.30 (± 0.45)	63.40
8B models							
AR initialization	85.90	69.18	59.15	53.05	68.20	83.33	69.80
AR mode							
Nemotron-Labs-Diffusion-8B	**88.25**	66.00	**60.37**	**53.05**	**68.20**	**82.54**	**69.73**
Sigma-unreg-8B	86.88	66.72	51.83	48.78	64.00	80.95	66.53
Sigma-preg-8B	86.66	**66.92**	53.66	50.00	63.80	79.63	66.78
Sigma-8B	84.84	65.18	56.10	52.44	63.80	**82.54**	67.48
Diffusion mode							
Nemotron-Labs-Diffusion-8B	87.26	65.16	62.80	57.32	**67.00**	**81.75**	70.22
Sigma-unreg-8B							
DDPM 256 steps/block (w=2, τ=0.1)	**89.07** (± 0.28)	**67.47** (± 0.20)	61.40 (± 1.73)	57.44 (± 1.62)	64.78 (± 0.54)	77.83 (± 0.72)	69.67
DDIM 32 steps/block (w=3, τ=0.3)	87.78 (± 0.33)	67.13 (± 0.15)	58.54 (± 1.04)	53.17 (± 1.27)	59.54 (± 0.57)	74.15 (± 0.43)	66.72
Sigma-preg-8B							
DDPM 256 steps/block (w=3, τ=0.2)	88.61 (± 0.31)	66.40 (± 0.17)	63.54 (± 1.39)	58.23 (± 1.02)	63.28 (± 0.57)	77.49 (± 0.50)	69.59
DDIM 32 steps/block (w=4, τ=0.4)	87.60 (± 0.25)	66.04 (± 0.21)	61.04 (± 1.04)	55.67 (± 0.88)	60.34 (± 0.37)	74.34 (± 0.46)	67.51
Sigma-8B							
DDPM 256 steps/block (w=3, τ=0.1)	88.57 (± 0.12)	66.72 (± 0.16)	65.12 (± 1.22)	60.61 (± 1.15)	65.74 (± 0.44)	80.19 (± 0.76)	**71.16**
DDIM 32 steps/block (w=3, τ=0.3)	88.48 (± 0.14)	65.91 (± 0.15)	**65.37** (± 0.57)	**60.91** (± 0.74)	62.68 (± 0.30)	77.72 (± 0.32)	70.18

All *Sigma* models featured in our main results are warm-started from an AR checkpoint ("AR initialization" in Tab. 1) and then trained for 300B tokens under the joint diffusion + AR objective (Sec. 3.2.2). We ablate that choice against a masked diffusion language model (MDLM) warm start: identical 300B-token recipe, run instead from the released Nemotron-Labs-Diffusion-3B checkpoint of Fu et al. (2026), which is our main baseline in Tab. 1. All other knobs are held fixed.

Evaluation setup. In Tab. 7, we evaluate the performance of the two trained *Sigma-unreg*-3B models at the two CFG–ST configurations of Tab. 1, both with SC enabled: DDPM (256 steps, EDM $\rho = 7$ schedule, $w = 2$, $\tau = 0.1$) and DDIM (32 steps, uniform-γ schedule, $w = 3$, $\tau = 0.3$). Numbers are mean ± two standard errors over 10 sampling lanes, forming a 95% confidence interval of the mean.

Results. MDLM initialization has a small but consistent win on average at both configurations, but the sign of the change is task-dependent rather than uniform. MBPP and MBPP+ strongly favor MDLM initialization, while HumanEval and HumanEval+ show opposite trends.



(a) Validation Loss (AR)

(b) Validation Loss (Diffusion)

Figure 12 | **Pretraining *Sigma-unreg*-3B** with block size 32 and AR regularization from AR initialization (AR init) and from MDLM initialization (MDLM init). PPL for AR stream (a) and diffusion stream (b) are computed using $\mathcal{L}_{\text{ar}}(\mathbf{x})$ (6) and $\mathcal{L}_{\text{BD}}(\mathbf{x})$ (3), respectively. MDLM initialization has an initial advantage that dies off during learning rate decay.

Table 7 | **Diffusion initialization ablation for *Sigma-unreg*-3B pretraining.** Same six generative reasoning tasks as Tab. 1. The *AR init* rows are copied verbatim from Tab. 1. The *MDLM init* rows use the same 300B-token training recipe but warm-started from the Nemotron-Labs-Diffusion-3B checkpoint of Fu et al. (2026). Δ = MDLM − AR. All metrics are mean ± two standard errors over 10 sampling lanes. **Avg.** is the unweighted mean over the six tasks.

Sampler	Init	GSM8K	Minerva Math	HumanEval	HumanEval+	MBPP	MBPP+	Avg.
DDPM (256 steps, w=2, τ=0.1)	AR	84.50 (± 0.31)	64.60 (± 0.28)	59.45 (± 1.30)	53.84 (± 1.29)	52.32 (± 0.73)	71.32 (± 0.91)	64.34
	MDLM	84.45 (± 0.35)	65.37 (± 0.12)	57.13 (± 1.05)	52.87 (± 1.35)	55.56 (± 0.65)	72.62 (± 0.56)	64.67
	Δ	-0.05	+0.77	-2.32	-0.97	+3.24	+1.30	+0.33
DDIM (32 steps, w=3, τ=0.3)	AR	81.91 (± 0.32)	62.76 (± 0.18)	56.59 (± 1.31)	50.67 (± 1.00)	47.26 (± 0.46)	63.49 (± 0.77)	60.45
	MDLM	83.71 (± 0.17)	64.23 (± 0.28)	52.99 (± 0.59)	50.18 (± 0.68)	51.24 (± 0.51)	67.46 (± 0.58)	61.64
	Δ	+1.80	+1.47	-3.60	-0.49	+3.98	+3.97	+1.19

D. Implementation Details and Additional Results for Likelihood-based Benchmarks

[Back to Sec. 3.2 in the main paper.]

Table 8 | **Likelihood-based multiple-choice benchmarks (3B).** Accuracy (%) per task. *AR mode* ranks candidates with the AR likelihood from the AR stream; *diffusion mode* ranks them by the variational bound of the diffusion stream: (12) for *Sigma*, the count-based masked diffusion NELBO (14) for Nemotron-Labs-Diffusion. Underline: the best within each mode. **Bold**: the best value across all methods.

Method	ARC-e	ARC-c	HellaSwag	PIQA	WinoGrande	MMLU	Avg.
Chance	*25.00*	*25.00*	*25.00*	*50.00*	*50.00*	*25.00*	*33.33*
AR initialization	**80.39**	**57.42**	**71.74**	78.18	**70.64**	**70.74**	**71.52**
AR mode							
Nemotron-Labs-Diffusion-3B	78.70	55.46	70.39	**78.24**	70.32	69.61	70.46
***Sigma-unreg*-3B (Ours)**	79.29	55.29	69.98	77.64	69.46	68.50	70.03
Diffusion mode							
Nemotron-Labs-Diffusion-3B	78.07	52.99	64.90	73.78	68.75	69.70	68.03
***Sigma-unreg*-3B (Ours)**	78.96	54.61	65.64	73.61	69.93	67.66	68.40

Table 9 | **Likelihood-based multiple-choice benchmarks (8B).** Accuracy (%) per task. *AR mode* ranks candidates with the AR likelihood from the AR stream; *diffusion mode* ranks them by the variational bound of the diffusion stream: (12) for *Sigma*, the count-based masked diffusion NELBO (14) for Nemotron-Labs-Diffusion. Underline: the best within each mode. **Bold**: the best value across all methods.

Method	ARC-e	ARC-c	HellaSwag	PIQA	WinoGrande	MMLU	Avg.
Chance	*25.00*	*25.00*	*25.00*	*50.00*	*50.00*	*25.00*	*33.33*
AR initialization	**83.80**	**61.52**	**77.05**	80.03	**76.48**	**76.84**	**75.95**
AR mode							
Nemotron-Labs-Diffusion-8B	82.32	59.73	76.09	79.76	75.93	74.88	74.78
***Sigma-unreg*-8B (Ours)**	82.28	60.67	75.70	**80.41**	**76.48**	74.58	75.02
Diffusion mode							
Nemotron-Labs-Diffusion-8B	80.81	58.62	71.47	77.31	73.88	76.01	73.02
***Sigma-unreg*-8B (Ours)**	80.85	58.19	69.03	76.44	75.22	73.96	72.28

Benchmarks. We evaluate on ARC-e/c (Clark et al., 2018), HellaSwag (Zellers et al., 2019), PIQA (Bisk et al., 2020), WinoGrande (Sakaguchi et al., 2020), and MMLU (Hendrycks et al., 2021). These benchmarks score a candidate response $\mathbf{r}$ against a prompt $\mathbf{p}$ by comparing $\log p_\theta(\mathbf{r} \mid \mathbf{p})$ across candidates. Per Open LLM Leaderboard convention, we report the plain likelihood on WinoGrande and MMLU, and the likelihood normalized by character length on the other four tasks which are length-varying.

Scoring our continuous method. *Sigma* does not give likelihood in closed form, but a surrogate based on the NELBO: taking the prompt as the clean prefix and the candidate as the block to be denoised, and then applying (4) verbatim with $\mathbf{p}$ as $\mathbf{x}^{(<b)}$ and $\mathbf{r}$ as $\mathbf{x}^{(b)}$:

$$-\log p_\theta(\mathbf{r} \mid \mathbf{p}) \le \mathcal{L}_{\text{NELBO}}(\mathbf{r};\mathbf{p}) = \mathrm{KL}(q(\mathbf{z}_1 \mid \mathbf{r}) \,\|\, p(\mathbf{z}_1)) + \mathbb{E}_{q(\mathbf{z}_0|\mathbf{r})}\left[\textstyle\sum_{l=1}^{|\mathbf{r}|} - \log\langle \mathbf{x}_\theta^l(\mathbf{z}_0, 0; \mathbf{p}), \mathbf{r}^l\rangle\right] - \tfrac{1}{2}\ \mathbb{E}_{t\sim\mathrm{Unif}[0,1]}\, \mathbb{E}_{q(\mathbf{z}_t|\mathbf{r})}[\mathrm{SNR}'(t)\|\mathbf{e}_\theta(\mathbf{z}_t, t; \mathbf{p}) - \mathbf{e}\|^2]. \tag{12}$$

In practice, we evaluate (12) with 128 stratified samples $t_k = \frac{k+u}{128}$, $k = 0, \ldots, 127$, $u \sim \mathrm{Unif}[0,1]$ and 8 reconstruction samples.

Scoring the masked baseline via count-based NELBO. We first give a brief introduction of masked diffusion models (Sahoo et al., 2024; Shi et al., 2024; Ou et al., 2025). There, the latent $\mathbf{z}_t \in \mathcal{V}^L$ is a partially

masked token sequence: each position is independently kept clean with probability $1-t$ or replaced by the mask token $\mathbf{m}$ with probability t. The denoiser $\mathbf{x}_\theta(\mathbf{z}_t) \in (\Delta^V)^L \subset \mathbb{R}^{L\times V}$ returns a categorical distribution over clean tokens at each position; one can prove that it needs not to take time argument as t carries no information about $\mathbf{x}$ beyond what $\mathbf{z}_t$ already reveals (Ou et al., 2025). The resulting NELBO is a $\frac{1}{t}$-weighted cross entropy over the masked positions only.

Let $\mathbf{p}$ be the prompt and $\mathbf{r}$ be a candidate response. We generate the masked states by the *mask set*, instead of randomly masking each position with probability t: for $M \subseteq \{1,\dots,L_\mathbf{r}\}$, where $L_\mathbf{r} := |\mathbf{r}|$, we obtain $\mathbf{z}_M$ from masking indices in M of $\mathbf{r}$, and denote $C(M) := \sum_{l\in M} -\log\langle \mathbf{x}_\theta^l(\mathbf{z}_M;\mathbf{p}),\, \mathbf{r}^l\rangle$ their cross entropy. Conditioning that bound on $\mathbf{p}$ and integrating t out yields an average over the *number* of masked tokens (Ou et al., 2025; Nie et al., 2025):

$$-\log p_\theta(\mathbf{r}\mid\mathbf{p}) \;\le\; \mathcal{L}_{\text{MDLM}}(\mathbf{r};\mathbf{p}) \;=\; \mathbb{E}_{t\sim\text{Unif}[0,1]}\left[\frac{1}{t}\,\mathbb{E}_{M\sim\text{Ber}(t)^{\otimes L_\mathbf{r}}}\,C(M)\right] \tag{13}$$

$$=\; \mathbb{E}_{n\sim\text{Unif}\{1,\dots,L_\mathbf{r}\}}\left[\frac{L_\mathbf{r}}{n}\,\mathbb{E}_{M\sim\text{Unif}\binom{[L_\mathbf{r}]}{n}}\,C(M)\right], \tag{14}$$

where $\binom{[L_\mathbf{r}]}{n}$ means the set of all n-element subsets of $[L_\mathbf{r}] := \{1,2,...,L_\mathbf{r}\}$. (14) is a Rao–Blackwellized low-variance estimator of (13), manifested in the following three major differences: *(i)* The $\frac{1}{t}$ weight becomes $\frac{L_\mathbf{r}}{n} \le L_\mathbf{r}$, so rare small-$t$ draws that leave one or two masked tokens would not cause the variance of the estimator to blow up. *(ii)* Bernoulli masking leaves $\mathbf{r}$ wholly unmasked with probability $(1-t)^{L_\mathbf{r}}$, which may be common for short responses and wastes a forward pass, while the $n \ge 1$ constraint avoids this. *(iii)* As n is finite, we can similarly use stratified sampling as the t-grid above, spreading counts evenly over $\{1,\dots,L_\mathbf{r}\}$ from one uniform offset, so every batch covers all masking rates.

In practice, we use (14) with 128 low-discrepancy draws n_k and standard random sampling for M_k for Nemotron-Labs-Diffusion (Fu et al., 2026) in Tabs. 8 and 9. On MMLU candidates are single tokens, so $L_\mathbf{r} = n = 1$ and (14) is exact after one forward pass, while (13) would still be averaging $\frac{1}{t}$. We found (14) to work better than (13) in early experiments.

Length generalization. Diffusion training denoises blocks of exactly L' tokens from packed sequences of length L, whereas candidates are typically short and occasionally longer than L'. For each $(\mathbf{p},\mathbf{r})$ pair, we apply causal attention over $\mathbf{p}$ and bidirectional attention over $\mathbf{r}$, assuming that the bidirectional attention can generalize to response length other than L'. The attention map follows the conditional branch in Fig. 5a.

E. Implementation Details and Additional Results for Inference

[Back to Secs. 4 and 5 in the main paper.]

Table 10 | **All-NFE pass@1 accuracy** of *Sigma-unreg*-3B, *Sigma-preg*-3B, and *Sigma*-3B. 10 seed-runs with average and ± two standard errors reported. **Bold** marks the best value per task among *Sigma* per-NFE rows and NLD; underline marks the better of *Sigma-unreg*-3B, *Sigma-preg*-3B and *Sigma*-3B within each step budget. † we did not conduct distillation on *Sigma-unreg*-3B with the newest distillation recipe, which was historically developed on *Sigma-preg*-3B.

	Math		**Coding**					
Method (Diffusion Mode)	GSM8K	Minerva Math	HumanEval	HumanEval+	MBPP	MBPP+	Avg.	Δ Avg.
Nemotron-Labs-Diffusion-3B	83.40	63.96	56.10	53.05	**57.60**	**74.34**	64.74	
DDIM 256 steps / block								
Sigma-unreg-3B (w=3, τ=0.2)	84.18 (± 0.18)	64.00 (± 0.14)	**60.24** (± 0.84)	54.02 (± 0.80)	51.02 (± 0.42)	69.79 (± 0.16)	63.88	
Sigma-preg-3B (w=3, τ=0.2)	84.12 (± 0.14)	63.68 (± 0.18)	57.32 (± 0.67)	52.20 (± 0.74)	56.08 (± 0.39)	72.65 (± 0.31)	64.34	+0.46
Sigma-3B (w=3, τ=0.1)	85.73 (± 0.10)	**64.74** (± 0.16)	55.43 (± 0.53)	52.87 (± 0.62)	57.40 (± 0.19)	72.04 (± 0.32)	64.70	+0.82
DDIM 128 steps / block								
Sigma-unreg-3B (w=3, τ=0.2)	83.78 (± 0.11)	64.01 (± 0.15)	59.15 (± 0.68)	53.23 (± 0.91)	50.92 (± 0.26)	68.49 (± 0.37)	63.26	
Sigma-preg-3B (w=3, τ=0.1)	84.64 (± 0.16)	63.50 (± 0.11)	57.56 (± 0.39)	51.77 (± 0.58)	55.84 (± 0.17)	72.62 (± 0.38)	64.32	+1.06
Sigma-3B (w=3, τ=0.2)	**85.84** (± 0.14)	64.62 (± 0.21)	57.01 (± 0.50)	54.15 (± 0.69)	56.64 (± 0.32)	72.94 (± 0.27)	**65.20**	+1.94
DDIM 64 steps / block								
Sigma-unreg-3B (w=3, τ=0.2)	83.09 (± 0.27)	63.96 (± 0.16)	56.83 (± 0.62)	51.40 (± 0.60)	49.38 (± 0.39)	66.35 (± 0.35)	61.84	
Sigma-preg-3B (w=4, τ=0.2)	85.31 (± 0.15)	63.68 (± 0.11)	56.71 (± 0.55)	51.95 (± 0.64)	53.34 (± 0.33)	70.61 (± 0.43)	63.60	+1.76
Sigma-3B (w=3, τ=0.2)	85.43 (± 0.22)	64.28 (± 0.09)	57.20 (± 0.66)	53.66 (± 0.62)	56.42 (± 0.37)	72.65 (± 0.28)	64.94	+3.10
DDIM 32 steps / block								
Sigma-unreg-3B (w=3, τ=0.3)	81.91 (± 0.32)	62.76 (± 0.18)	56.59 (± 1.30)	50.67 (± 1.00)	47.06 (± 0.52)	63.49 (± 0.77)	60.41	
Sigma-preg-3B (w=4, τ=0.4)	83.17 (± 0.30)	63.28 (± 0.15)	55.24 (± 1.36)	50.91 (± 0.99)	51.64 (± 0.57)	66.88 (± 0.29)	61.85	+1.44
Sigma-3B (w=3, τ=0.3)	84.48 (± 0.21)	64.52 (± 0.21)	55.37 (± 1.28)	51.71 (± 1.23)	53.00 (± 0.34)	71.30 (± 0.45)	63.40	+2.99
DDIM 16 steps / block								
Sigma-unreg-3B (w=3, τ=0.5)	78.08 (± 0.34)	60.16 (± 0.21)	47.50 (± 1.14)	43.84 (± 0.94)	41.84 (± 1.00)	59.58 (± 0.98)	55.17	
Sigma-preg-3B (w=3, τ=0.4)	80.85 (± 0.31)	61.73 (± 0.24)	53.60 (± 0.80)	48.84 (± 0.90)	45.22 (± 0.57)	60.79 (± 0.66)	58.50	+3.33
Sigma-3B (w=3, τ=0.4)	82.74 (± 0.29)	63.34 (± 0.24)	52.20 (± 1.45)	49.57 (± 1.41)	48.28 (± 0.30)	66.98 (± 0.74)	60.52	+5.35
DDIM 8 steps / block								
Sigma-unreg-3B (w=3, τ=0.8)	66.65 (± 0.40)	45.13 (± 0.23)	28.54 (± 1.53)	25.67 (± 1.35)	23.58 (± 0.65)	32.80 (± 0.83)	37.06	
Sigma-preg-3B (w=3, τ=0.7)	70.88 (± 0.43)	52.12 (± 0.40)	34.45 (± 1.15)	31.52 (± 1.29)	27.98 (± 0.66)	39.63 (± 0.99)	42.76	+5.70
Sigma-3B (w=3, τ=0.6)	76.33 (± 0.45)	57.95 (± 0.18)	43.96 (± 1.20)	40.55 (± 1.12)	34.54 (± 0.64)	51.53 (± 0.86)	50.81	+13.75
DDIM 4 steps / block								
Sigma-unreg-3B (w=2, τ=0.9)	11.64 (± 0.57)	4.94 (± 0.11)	9.88 (± 0.69)	9.15 (± 0.79)	4.24 (± 0.53)	6.59 (± 0.52)	7.74	
Sigma-preg-3B (w=2, τ=0.9)	26.04 (± 0.64)	9.64 (± 0.20)	11.59 (± 1.22)	10.85 (± 1.09)	4.32 (± 0.62)	8.33 (± 0.84)	11.80	+4.06
Sigma-3B (w=2, τ=0.8)	38.52 (± 0.40)	20.67 (± 0.21)	21.89 (± 1.27)	20.24 (± 1.25)	11.32 (± 0.53)	19.39 (± 0.81)	22.01	+14.27
Distilled, 16 steps / block								
Sigma-distill-unreg-3B†	–	–	–	–	–	–	–	
Sigma-distill-preg-3B	83.12 (± 0.31)	60.02 (± 0.24)	57.32 (± 1.63)	51.83 (± 1.57)	49.40 (± 0.55)	69.02 (± 0.87)	61.78	
Sigma-distill-3B	83.42 (± 0.27)	61.70 (± 0.21)	58.54 (± 1.15)	**54.51** (± 1.48)	52.08 (± 0.87)	69.95 (± 0.69)	63.37	
Distilled, 8 steps / block								
Sigma-distill-unreg-3B†	–	–	–	–	–	–	–	
Sigma-distill-preg-3B	80.37 (± 0.33)	58.22 (± 0.15)	51.89 (± 1.49)	47.20 (± 1.49)	42.38 (± 0.84)	59.79 (± 0.76)	56.64	
Sigma-distill-3B	84.62 (± 0.44)	61.17 (± 0.22)	56.40 (± 1.82)	51.71 (± 2.28)	49.04 (± 0.45)	67.70 (± 0.77)	61.77	
Distilled, 4 steps / block								
Sigma-distill-unreg-3B†	–	–	–	–	–	–	–	
Sigma-distill-preg-3B	76.94 (± 0.35)	53.97 (± 0.27)	36.77 (± 1.54)	33.17 (± 1.70)	23.70 (± 1.10)	33.76 (± 0.69)	43.05	
Sigma-distill-3B	82.23 (± 0.34)	59.13 (± 0.19)	52.13 (± 1.56)	48.72 (± 1.20)	40.62 (± 0.91)	58.81 (± 1.22)	56.94	
Distilled, 2 steps / block								
Sigma-distill-unreg-3B†	–	–	–	–	–	–	–	
Sigma-distill-preg-3B	–	–	–	–	–	–	–	
Sigma-distill-3B	74.42 (± 0.46)	49.71 (± 0.30)	32.38 (± 1.34)	30.55 (± 0.86)	20.78 (± 1.00)	30.87 (± 0.79)	39.78	

Table 11 | **All-NFE pass@1 accuracy** of *Sigma-unreg*-3B, *Sigma-preg*-3B, and *Sigma*-3B. 10 seed-runs with average and ± two standard errors reported. **Bold** marks the best value per task among *Sigma* per-NFE rows and NLD; underline marks the better of *Sigma-unreg*-3B, *Sigma-preg*-3B and *Sigma*-3B within each step budget. ; underline marks the better of *Sigma-unreg*-8B, *Sigma-preg*-8B and *Sigma*-8B within each step budget, matching Tab. 10.

	Math		**Coding**					
Method (Diffusion Mode)	GSM8K	Minerva Math	HumanEval	HumanEval+	MBPP	MBPP+	Avg.	Δ Avg.
Nemotron-Labs-Diffusion-8B	87.26	65.16	62.80	57.32	**67.00**	**81.75**	70.22	
DDIM 256 steps / block								
Sigma-unreg-8B (w=3, τ=0.2)	88.41 (± 0.14)	**67.73** (± 0.10)	58.17 (± 0.80)	52.87 (± 0.81)	63.82 (± 0.26)	76.85 (± 0.21)	67.97	
Sigma-preg-8B (w=3, τ=0.2)	88.38 (± 0.11)	66.88 (± 0.11)	64.63 (± 0.30)	58.48 (± 0.40)	62.58 (± 0.24)	78.31 (± 0.24)	69.88	+1.91
Sigma-8B (w=3, τ=0.1)	89.12 (± 0.12)	67.06 (± 0.17)	67.01 (± 0.50)	62.56 (± 0.39)	65.48 (± 0.18)	80.26 (± 0.19)	**71.92**	+3.95
DDIM 128 steps / block								
Sigma-unreg-8B (w=3, τ=0.2)	88.70 (± 0.14)	67.28 (± 0.09)	58.48 (± 1.00)	52.68 (± 1.00)	63.22 (± 0.28)	76.85 (± 0.31)	67.87	
Sigma-preg-8B (w=3, τ=0.1)	88.28 (± 0.12)	66.36 (± 0.11)	64.15 (± 0.86)	58.05 (± 0.82)	62.22 (± 0.21)	78.02 (± 0.22)	69.51	+1.64
Sigma-8B (w=3, τ=0.2)	88.27 (± 0.14)	66.86 (± 0.08)	63.72 (± 0.95)	59.70 (± 0.98)	64.74 (± 0.28)	79.66 (± 0.32)	70.49	+2.62
DDIM 64 steps / block								
Sigma-unreg-8B (w=3, τ=0.2)	88.68 (± 0.19)	67.30 (± 0.17)	58.90 (± 0.71)	52.56 (± 0.54)	61.96 (± 0.36)	76.35 (± 0.28)	67.63	
Sigma-preg-8B (w=4, τ=0.2)	88.66 (± 0.14)	66.40 (± 0.16)	63.54 (± 0.93)	56.65 (± 0.63)	63.02 (± 0.25)	76.46 (± 0.24)	69.12	+1.49
Sigma-8B (w=3, τ=0.2)	88.12 (± 0.15)	66.35 (± 0.12)	63.60 (± 0.60)	59.94 (± 0.52)	64.50 (± 0.28)	80.08 (± 0.24)	70.43	+2.80
DDIM 32 steps / block								
Sigma-unreg-8B (w=3, τ=0.3)	87.78 (± 0.33)	67.13 (± 0.15)	58.54 (± 1.04)	53.17 (± 1.27)	59.54 (± 0.57)	74.15 (± 0.43)	66.72	
Sigma-preg-8B (w=4, τ=0.4)	87.60 (± 0.25)	66.04 (± 0.21)	61.04 (± 1.04)	55.67 (± 0.88)	60.34 (± 0.37)	74.34 (± 0.46)	67.51	+0.79
Sigma-8B (w=3, τ=0.3)	88.48 (± 0.14)	65.91 (± 0.15)	65.37 (± 0.57)	60.91 (± 0.74)	62.68 (± 0.30)	77.72 (± 0.32)	70.18	+3.46
DDIM 16 steps / block								
Sigma-unreg-8B (w=3, τ=0.5)	85.22 (± 0.33)	65.90 (± 0.20)	54.27 (± 0.96)	48.96 (± 1.12)	55.06 (± 0.79)	69.47 (± 0.69)	63.15	
Sigma-preg-8B (w=3, τ=0.4)	86.57 (± 0.28)	65.40 (± 0.18)	55.55 (± 1.41)	50.00 (± 1.29)	56.68 (± 0.57)	69.42 (± 0.52)	63.94	+0.79
Sigma-8B (w=3, τ=0.4)	87.03 (± 0.20)	65.66 (± 0.21)	63.90 (± 1.36)	59.33 (± 1.28)	60.16 (± 0.41)	75.61 (± 0.53)	68.61	+5.46
DDIM 8 steps / block								
Sigma-unreg-8B (w=3, τ=0.8)	76.49 (± 0.42)	56.79 (± 0.23)	31.65 (± 1.74)	29.02 (± 1.50)	35.80 (± 0.94)	46.53 (± 0.93)	46.05	
Sigma-preg-8B (w=3, τ=0.7)	80.40 (± 0.50)	59.90 (± 0.24)	40.98 (± 0.94)	38.05 (± 1.26)	40.24 (± 0.78)	52.51 (± 1.08)	52.01	+5.96
Sigma-8B (w=3, τ=0.6)	83.95 (± 0.36)	62.94 (± 0.22)	53.48 (± 1.23)	49.63 (± 1.41)	52.58 (± 0.88)	66.61 (± 0.79)	61.53	+15.48
DDIM 4 steps / block								
Sigma-unreg-8B (w=2, τ=0.9)	18.24 (± 0.38)	10.65 (± 0.19)	11.16 (± 0.48)	11.04 (± 0.76)	7.46 (± 0.38)	11.46 (± 0.40)	11.67	
Sigma-preg-8B (w=2, τ=0.9)	37.85 (± 0.47)	17.06 (± 0.24)	14.94 (± 1.15)	14.39 (± 1.04)	8.88 (± 0.62)	12.99 (± 0.57)	17.69	+6.02
Sigma-8B (w=2, τ=0.8)	56.29 (± 0.39)	32.43 (± 0.24)	31.40 (± 1.13)	28.41 (± 0.83)	26.84 (± 0.87)	38.57 (± 0.69)	35.66	+23.99
Distilled, 16 steps / block								
Sigma-distill-unreg-8B	87.21 (± 0.21)	66.46 (± 0.21)	62.74 (± 0.80)	59.39 (± 0.73)	61.26 (± 0.53)	75.29 (± 1.00)	68.73	
Sigma-distill-preg-8B	85.78 (± 0.48)	63.80 (± 0.18)	62.93 (± 0.77)	58.72 (± 0.66)	61.02 (± 0.65)	77.28 (± 0.46)	68.25	
Sigma-distill-8B	**89.15** (± 0.28)	65.51 (± 0.23)	**67.32** (± 1.14)	**63.35** (± 1.18)	61.96 (± 0.81)	77.70 (± 0.96)	70.83	
Distilled, 8 steps / block								
Sigma-distill-unreg-8B	87.37 (± 0.22)	64.98 (± 0.24)	63.84 (± 1.16)	60.49 (± 1.37)	60.80 (± 0.67)	74.81 (± 0.41)	68.71	
Sigma-distill-preg-8B	86.62 (± 0.27)	63.51 (± 0.29)	60.91 (± 0.97)	57.20 (± 1.56)	59.34 (± 0.78)	74.95 (± 0.66)	67.09	
Sigma-distill-8B	87.85 (± 0.18)	65.36 (± 0.25)	63.60 (± 1.74)	59.63 (± 2.09)	58.18 (± 0.91)	72.75 (± 1.36)	67.89	
Distilled, 4 steps / block								
Sigma-distill-unreg-8B	85.82 (± 0.28)	62.53 (± 0.27)	51.89 (± 1.32)	49.15 (± 1.06)	47.92 (± 0.78)	62.78 (± 0.94)	60.01	
Sigma-distill-preg-8B	86.90 (± 0.36)	63.20 (± 0.19)	51.40 (± 0.93)	48.48 (± 0.84)	46.66 (± 0.87)	59.10 (± 0.67)	59.29	
Sigma-distill-8B	87.15 (± 0.24)	65.05 (± 0.29)	60.12 (± 0.86)	57.01 (± 0.71)	55.06 (± 0.97)	72.62 (± 0.78)	66.17	
Distilled, 2 steps / block								
Sigma-distill-unreg-8B	–	–	–	–	–	–	–	
Sigma-distill-preg-8B	–	–	–	–	–	–	–	
Sigma-distill-8B	80.02 (± 0.39)	56.34 (± 0.33)	44.15 (± 1.40)	41.52 (± 1.57)	33.18 (± 1.24)	47.35 (± 1.54)	50.43	

E.1. Sampler Updates

Let $1 = t_0 > \cdots > t_N = 0$, with $\alpha_i = \alpha_{t_i}$, $\sigma_i = \sigma_{t_i}$, $\rho_i = \sigma_i/\alpha_i$, and $\lambda_i = \log(\alpha_i/\sigma_i) = -\gamma(t_i)/2$. At step i, the clean prediction $\hat{\mathbf{e}}_i = \mathbf{e}_\theta(\mathbf{z}_i, t_i; \mathbf{c})$ implies $\hat{\boldsymbol{\epsilon}}_i = (\mathbf{z}_i - \alpha_i \hat{\mathbf{e}}_i)/\sigma_i$. Predictions incorporate CFG and ST as in (CFG) and (ST).

DDPM. The ancestral VDM update (Kingma et al., 2021, App. A.4) uses the Gaussian posterior with $\hat{\mathbf{e}}_i$ replacing the clean embedding:

$$\mathbf{z}_{i+1} = (1 - c_i)\frac{\alpha_{i+1}}{\alpha_i}\mathbf{z}_i + c_i \alpha_{i+1}\hat{\mathbf{e}}_i + \sigma_{i+1}\sqrt{c_i}\,\boldsymbol{\epsilon}, \quad c_i = 1 - \mathrm{e}^{\gamma(t_{i+1}) - \gamma(t_i)}, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}). \tag{15}$$

DDIM. DDIM (Song et al., 2021a) deterministically combines the current clean and noise predictions:

$$\mathbf{z}_{i+1}^{\mathrm{DDIM}} = \alpha_{i+1}\hat{\mathbf{e}}_i + \sigma_{i+1}\hat{\boldsymbol{\epsilon}}_i. \tag{16}$$

It is the first-order update obtained by freezing either prediction in the corresponding integrating-factor formulation of the PFODE.

E.2. Ablation Study: Time Discretization

[Back to Sec. 5.1 in the main paper.]

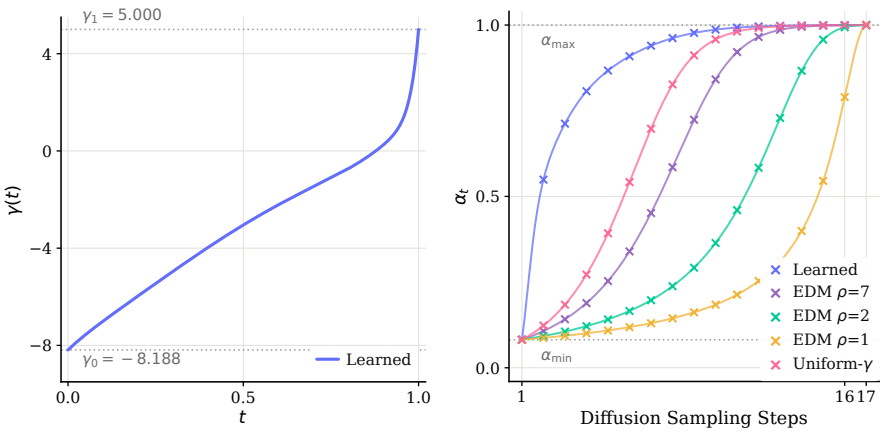


Figure 13 | Left: the **learned noise schedule** $\gamma(t)$ for *Sigma-unreg*-3B. Right: Different **time discretization schemes** for 16-step generation. Note that an extra decoding step is needed at $t = 0$.

We mainly consider the following discretization schemes. Let N be the total number of steps and assume the i-th step ($i = 1, ..., N$) goes from t_{i-1} to $t_i < t_{i-1}$.

- Learned: arithmetic progression in time: $t_i = 1 - \frac{i}{N}$.
- Uniform-γ: arithmetic progression in γ, the minus log-SNR: $\gamma(t_i) = \gamma_{\max} + \frac{i}{N}(\gamma_{\min} - \gamma_{\max})$.
- EDM ρ (Karras et al., 2022): first convert the VP schedule to VE by defining $\tilde{\sigma}_t := \frac{\sigma_t}{\alpha_t}$ as the essential standard deviation at time t. The grid points are chosen so that $\tilde{\sigma}_{t_i}^{1/\rho}$ is an arithmetic progression, i.e., $\tilde{\sigma}_{t_i} = \left(\tilde{\sigma}_{\max}^{1/\rho} + \frac{i}{N}(\tilde{\sigma}_{\min}^{1/\rho} - \tilde{\sigma}_{\max}^{1/\rho})\right)^\rho$. Note that as $\rho \to \infty$, this converges to the uniform-γ scheme.

We visualize the three schemes in Fig. 13. Empirically, we find that for high NFE sampling, both the uniform-γ and EDM $\rho = 7$ schemes perform similarly, while uniform-γ generally performs the best for low NFE sampling. See Fig. 14 for the few-step reasoning accuracy of DDPM and DDIM over GSM8K and HumanEval.

E.3. Reasoning and Coding Benchmark Setup

[Back to Secs. 4 and 5 in the main paper.]

We describe the harness, scoring pipeline, and sampler that produce Tabs. 1, 2, 10 and 11. The scoring pipeline is consistent for all base models and generation modes.

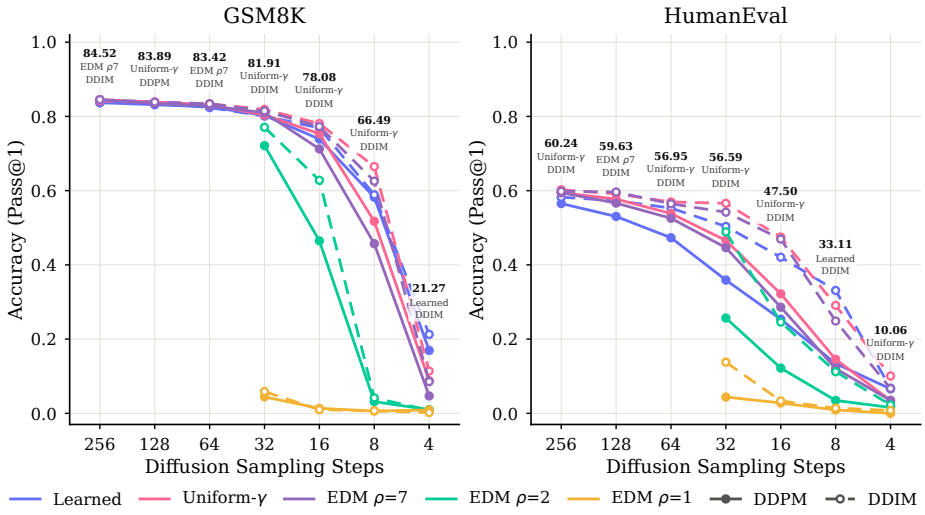


Figure 14 | **DDIM substantially outperforms DDPM at low NFEs** for *Sigma-unreg*-3B, while matching DDPM at high NFEs.

Benchmarks and few-shot budgets. We use six tasks from `lm-evaluation-harness` following Fu et al. (2026): GSM8K (8-shot, 1319 questions) (Cobbe et al., 2021), Minerva Math (4-shot, 5000 questions[4]) (Lewkowycz et al., 2022), HumanEval and HumanEval+ (0-shot, 164 questions) (Chen et al., 2021), and MBPP and MBPP+ (3-shot, 500 and 378 questions, respectively) (Austin et al., 2021). The per-task generation budget is 256 tokens for GSM8K and 512 tokens for the other five tasks.

Scoring. GSM8K is scored by exact-match on the numerical answer, extracted from the model's response by a permissive regex that keeps the last number appearing in the completion (stripping separators and trailing punctuation). Minerva Math is scored by symbolic equivalence after canonicalizing boxed and inline expressions. HumanEval and HumanEval+ report post-processed pass@1: the completion is stripped out of the Markdown code fence, an AST-based longest-valid-code extraction is applied, and the resulting program is executed against the task test cases in the container. MBPP and MBPP+ report the harness pass@1 directly without this post-processing step, matching the convention of prior work.

Decoding configuration. All *Sigma* diffusion-mode runs use block size $L' = 32$, `BF16` model weights at inference time, and operation of SC follows Fig. 5b. The DDPM-256 row uses the ancestral sampler of (15) on the EDM $\rho = 7$ time discretization (App. E.2); the DDIM-32 row uses the deterministic sampler of (16) on the uniform-γ time discretization.

Seed error bars. Each *Sigma* diffusion row aggregates 10 independent sampling lanes as follows. For question i, we independently draw 10 responses under the same model with 10 different seeds. Let $s_{i,j}$ be the score of the j-th response to question i. We first average over questions to obtain 10 per-lane scores $s_j = \text{mean}_i(s_{i,j})$ for $j = 1, \ldots, 10$, and then obtain $\bar{s} = \text{mean}_j(s_j)$ together with the across-lane standard deviation $\hat{\sigma} = \text{std}_j(s_j)$. The half-widths in Tab. 1 are $\frac{2}{\sqrt{10}}\hat{\sigma}$, an approximate 95% confidence interval for the mean.

Baseline decoders. The AR-mode for NLD and *Sigma* row uses greedy argmax at each step; NLD's diffusion-mode row uses the confidence-threshold greedy sampler of Fu et al. (2026) (block size $L' = 32$; argmax readout at every position whose top-1 probability exceeds a threshold; we report the better of thresholds 0.8 and 0.9 per task). For LLaDA-8B and Dream-7B we quote each model's best diffusion-mode number without the confidence-threshold rule: at each denoising step the single position with the highest top-1 confidence is unmasked. Both rules are instances of confidence-based decoding (Chang et al., 2022), and both inherit the absorbing property of masked diffusion: once a position is unmasked, its token is frozen for the rest of the trajectory.

[4]Number of questions in each sub-domain: algebra 1187, counting/prob 474, geometry 479, intermediate algebra 903, number theory 540, prealgebra 871, precalc 546.

E.4. Sweeping Classifier-Free Guidance and Score Temperature

[Back to Sec. 4.2 or Sec. 4.4 in the main paper.]

In Figs. 15 and 16, we present the full tables for CFG and ST sweep for our *Sigma-unreg*-3B models trained with and without AR regularization, respectively.

E.5. Ablation Study: Auto-Guidance with Self-Conditioning

[Back to Sec. 4.2 in the main paper.]

Standard CFG (Ho & Salimans, 2022), also following the idea of auto-guidance (Karras et al., 2024), extrapolates a conditional prediction away from a "bad" prediction to sharpen the model's prediction. Self-conditioning introduces a second source of conditioning, as is used in existing continuous dLMs (e.g., Lee et al. (2026); Potaptchik et al. (2026); Hu et al. (2026); see App. A for detailed discussion). We define the following notations in this subsection:

$$\mathbf{e}_\theta^+ := \mathbf{e}_\theta(\mathbf{z}_t^{(b)}, t; \mathbf{c}, \mathbf{e}_{\text{sc}}), \qquad \mathbf{e}_\theta^{\mathbf{c},\mathbf{0}} := \mathbf{e}_\theta(\mathbf{z}_t^{(b)}, t; \mathbf{c}, \mathbf{0}), \qquad \mathbf{e}_\theta^{\varnothing,\mathbf{0}} := \mathbf{e}_\theta(\mathbf{z}_t^{(b)}, t; \varnothing, \mathbf{0}),$$

where the good prediction $\mathbf{e}_\theta^+$ is the conditional prediction that uses both the clean context and SC; for either bad prediction $\mathbf{e}_\theta^- \in \{\mathbf{e}_\theta^{\mathbf{c},\mathbf{0}}, \mathbf{e}_\theta^{\varnothing,\mathbf{0}}\}$, the guided mean embedding prediction is

$$\mathbf{e}_\theta^w = \mathbf{e}_\theta^+ + (w-1)(\mathbf{e}_\theta^+ - \mathbf{e}_\theta^-).$$

The *context-preserving* baseline keeps the prompt and previously generated blocks in both branches and sets $\mathbf{e}_\theta^- \leftarrow \mathbf{e}_\theta^{\mathbf{c},\mathbf{0}}$; it therefore extrapolates only away from the prediction made without SC. Our proposed method in Sec. 4.2 instead uses the fully unconditional branch $\mathbf{e}_\theta^- \leftarrow \mathbf{e}_\theta^{\varnothing,\mathbf{0}}$, removing both clean context and SC.

Experimental setup. We compare the two choices using the same *Sigma-unreg*-3B checkpoint with block length 32, full GSM8K and HumanEval evaluation sets, and 10 samples per problem. The DDIM sweep uses 64 steps per block, uniform-γ schedule, and ST $\tau = 0.3$; the DDPM sweep uses 256 steps per block, EDM $\rho = 7$ schedule, and ST $\tau = 1$. We remark that the guidance is inactive at $w = 1$ and the two choices coincide.

Table 12 | GSM8K accuracy (%) for self-conditioning auto-guidance. "Context/no-SC" retains clean context but removes SC (i.e., $\mathbf{e}_\theta^- \leftarrow \mathbf{e}_\theta^{\mathbf{c},\mathbf{0}}$); ours removes both (i.e., $\mathbf{e}_\theta^- \leftarrow \mathbf{e}_\theta^{\varnothing,\mathbf{0}}$). Score entries are mean ± two standard errors over the 10 sampling lanes, an approximate 95% confidence interval for the mean; Δ is ours − context/no-SC.

DDIM, 64 steps/block, uniform-γ schedule, $\tau = 0.3$

Guidance w	1	2	3	4	5	6	8	12	16
Context/no-SC	68.13 (± 0.38)	72.48 (± 0.27)	73.99 (± 0.47)	74.37 (± 0.35)	75.12 (± 0.51)	76.13 (± 0.41)	**76.19** (± 0.44)	75.60 (± 0.56)	74.09 (± 0.52)
Ours	68.13 (± 0.38)	82.52 (± 0.21)	82.84 (± 0.28)	83.04 (± 0.20)	**83.19** (± 0.18)	83.01 (± 0.30)	82.78 (± 0.40)	81.26 (± 0.40)	78.00 (± 0.32)
Δ	0.00	+10.05	+8.85	+8.67	+8.07	+6.88	+6.59	+5.66	+3.91

DDPM, 256 steps/block, EDM $\rho = 7$ schedule, $\tau = 1$

Guidance w	1	2	3	4	5	6	7	8	9	10
Context/no-SC	53.09 (± 0.85)	62.43 (± 0.97)	65.44 (± 0.77)	67.53 (± 0.58)	**68.45** (± 0.77)	**68.45** (± 0.56)	67.66 (± 0.61)	67.82 (± 1.17)	67.09 (± 0.48)	66.76 (± 0.56)
Ours	53.09 (± 0.85)	77.12 (± 0.66)	80.04 (± 0.39)	80.99 (± 0.48)	81.30 (± 0.41)	82.00 (± 0.47)	81.54 (± 0.64)	82.05 (± 0.29)	82.41 (± 0.51)	**82.43** (± 0.50)
Δ	0.00	+14.69	+14.60	+13.46	+12.85	+13.55	+13.88	+14.23	+15.32	+15.67

Results. As shown in Tabs. 12 and 13, the fully unconditional bad prediction (ours) is consistently stronger in the useful guidance range: it almost wins every matched CFG weight in sweep on both tasks and both samplers, and, tuned over the sweep, reaches a higher ceiling in each task. Thus, we conclude that the advantage shown here is not an artifact of using different guidance search ranges.

E.6. Ablation Study: Steering in Logit Space instead of Embedding Space

[Back to Sec. 4.2 in the main paper.]

Both steering knobs introduced in Sec. 4.2 act on the predicted clean embedding $\mathbf{e}_\theta \in \mathbb{R}^{L' \times d_e}$, equivalently on the score or velocity. Recall that our denoiser leverages *categorical reparameterization*, i.e., $\mathbf{e}_\theta = \text{softmax}(\boldsymbol{\ell}_\theta)\mathbf{E} = \mathbf{x}_\theta \mathbf{E}$, where $\boldsymbol{\ell}_\theta \in \mathbb{R}^{L' \times V}$ are the predicted logits, and softmax operates in the vocabulary

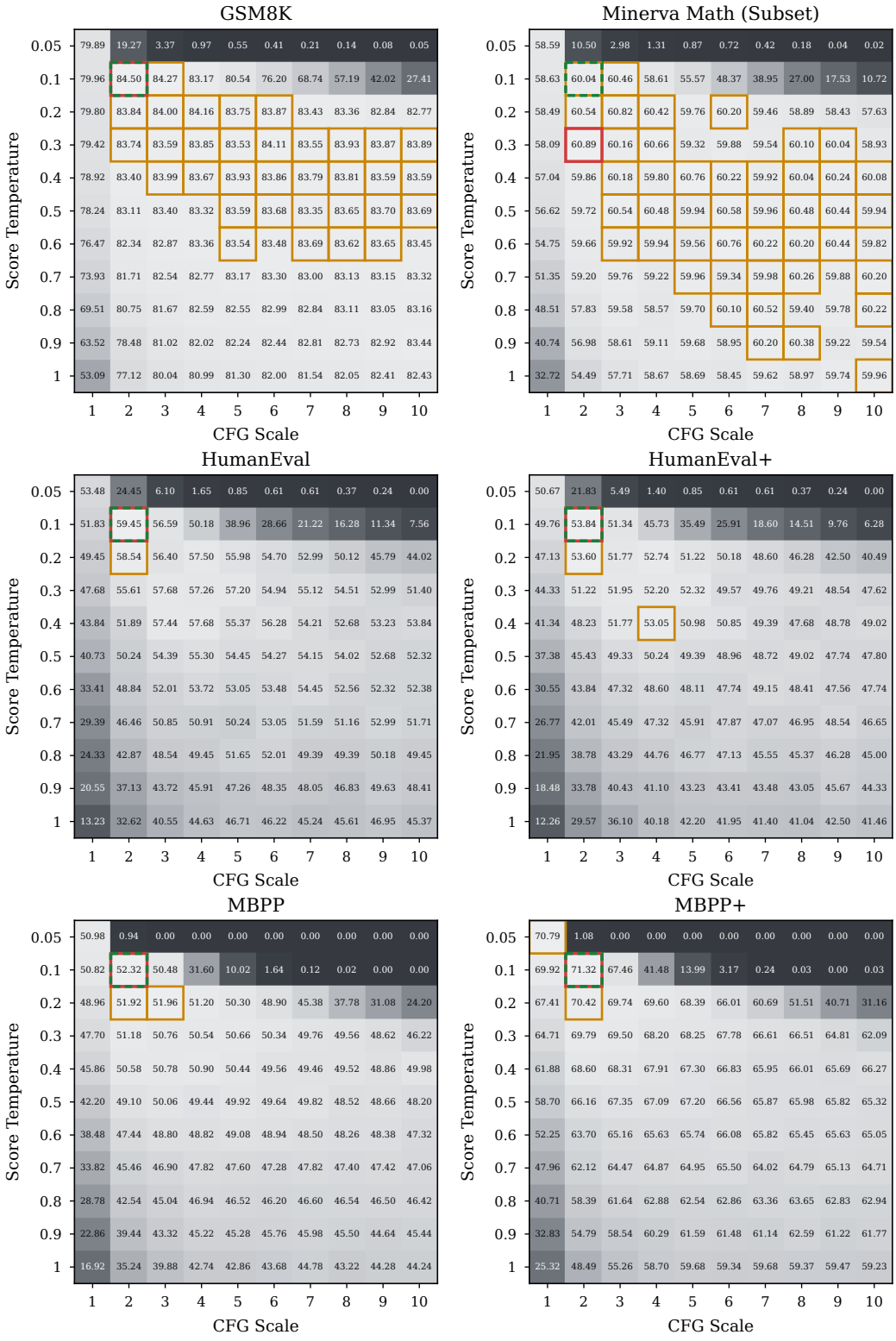


Figure 15 | Sweeping CFG and ST for *Sigma-unreg*-3B trained with AR regularization. Accuracies are averaged over 10 random seeds. Red: per-task best; orange: within 1 point of per-task best; green: best average across 6 tasks. **For this model, per-task best and best average align on 5 out of 6 tasks.** In the exception of Minerva Math (subset; random 10% from each subtask), per-task best is still within 1 point from best average. We report best average ($w = 2$, $\tau = 0.1$) in Fig. 4 and Tab. 1.

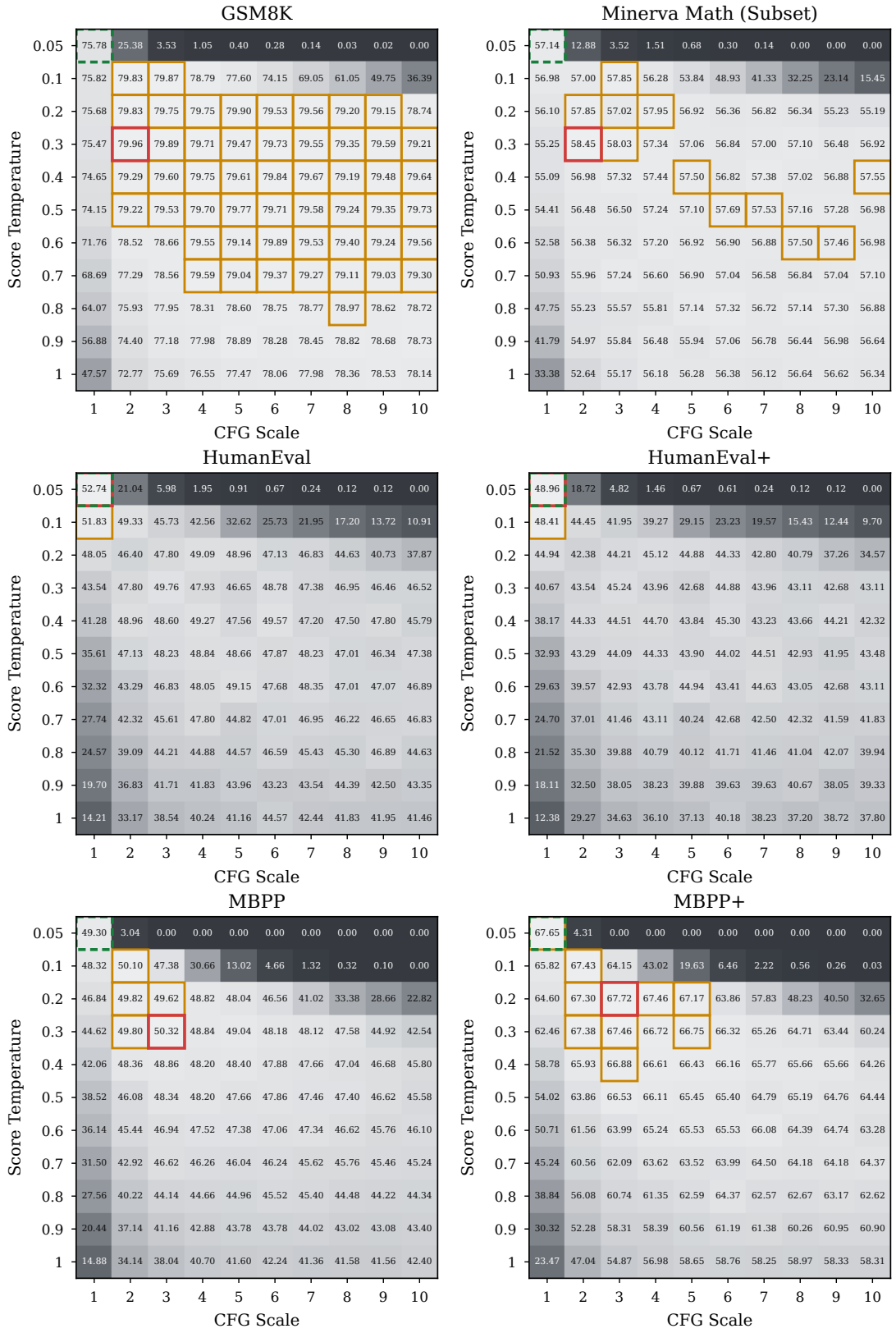


Figure 16 | Sweeping CFG and ST for *Sigma-unreg*-3B trained *without* AR regularization. Accuracies are averaged over 10 random seeds. Red: per-task best; orange: within 1 point of per-task best; green: best average across 6 tasks. Both per-task best and best average are worse than *Sigma-unreg*-3B trained AR regularization (Fig. 15).

Table 13 | HumanEval pass@1 accuracy (%) for self-conditioning auto-guidance. "Context/no-SC" retains clean context but removes SC (i.e., $\mathbf{e}_\theta^- \leftarrow \mathbf{e}_\theta^{\mathbf{c},\mathbf{0}}$); ours removes both (i.e., $\mathbf{e}_\theta^- \leftarrow \mathbf{e}_\theta^{\varnothing,\mathbf{0}}$). Score entries are mean ± two standard errors over the 10 sampling lanes, an approximate 95% confidence interval for the mean; Δ is ours − context/no-SC.

DDIM, 64 steps/block, uniform-γ schedule, $\tau = 0.3$

Guidance w	1	2	3	4	5	6	8	12	16
Context/no-SC	42.93 (± 0.66)	46.65 (± 1.08)	50.00 (± 0.81)	50.85 (± 0.61)	**50.91** (± 0.86)	50.43 (± 0.66)	49.39 (± 1.30)	43.23 (± 1.20)	40.85 (± 2.07)
Ours	42.93 (± 0.66)	53.96 (± 0.95)	56.04 (± 1.16)	**56.22** (± 0.83)	54.09 (± 1.22)	54.51 (± 0.89)	54.02 (± 0.78)	43.66 (± 1.52)	31.22 (± 1.62)
Δ	0.00	+7.31	+6.04	+5.37	+3.18	+4.08	+4.63	+0.43	-9.63

DDPM, 256 steps/block, EDM $\rho = 7$ schedule, $\tau = 1$

Guidance w	1	2	3	4	5	6	7	8	9	10
Context/no-SC	13.23 (± 1.50)	20.85 (± 2.29)	25.73 (± 1.51)	27.93 (± 1.66)	29.57 (± 1.03)	29.76 (± 1.75)	31.40 (± 2.07)	32.38 (± 1.88)	**32.50** (± 2.26)	30.00 (± 1.35)
Ours	13.23 (± 1.50)	32.62 (± 1.98)	40.55 (± 1.79)	44.63 (± 1.07)	46.71 (± 1.82)	46.22 (± 2.12)	45.24 (± 1.49)	45.61 (± 1.03)	**46.95** (± 1.73)	45.37 (± 1.63)
Δ	0.00	+11.77	+14.82	+16.70	+17.14	+16.46	+13.84	+13.23	+14.45	+15.37

dimension. Thus, a natural alternative is to apply the same operations to $\boldsymbol{\ell}_\theta$ before the softmax. This is the form of guidance used by discrete dLMs, which does not have embedding-space prediction to steer (Nie et al., 2025; Li et al., 2025; Schiff et al., 2025) (App. A).

Logit-space CFG (LCFG). Reusing the notations in App. E.5, with the same fully unconditional bad prediction, we do softmax on the extrapolated logits:

$$\mathbf{e}_\theta^{w,\text{logit}} := \text{softmax}(\boldsymbol{\ell}_\theta^{\varnothing,\mathbf{0}} + w\,[\boldsymbol{\ell}_\theta^{+} - \boldsymbol{\ell}_\theta^{\varnothing,\mathbf{0}}])\mathbf{E}, \qquad w \geq 1,$$

against our default $\mathbf{e}_\theta^w = \mathbf{e}_\theta^{\varnothing,\mathbf{0}} + w[\mathbf{e}_\theta^{+} - \mathbf{e}_\theta^{\varnothing,\mathbf{0}}]$ of (CFG), which extrapolates after the softmax. The two coincide at $w = 1$ (i.e., no guidance).

Logit temperature (LT). This is the analogue of ST in logit space, which rescales the conditional logits themselves:

$$\mathbf{e}_\theta^T := \text{softmax}\left(\frac{\boldsymbol{\ell}_\theta^{+}}{T}\right)\mathbf{E}, \qquad T \in [0, 1],$$

where $T = 1$ recovers the conditional mean embedding prediction (i.e., no guidance), and smaller T sharpens. The $T \to 0$ limit is the maximum a posteriori (MAP) denoiser $\mathbf{e}_\theta^0 = \mathbf{E}[\text{argmax}\,\boldsymbol{\ell}_\theta^{+}]$, which snaps the prediction onto a single vocabulary embedding.

We remark that both operations change the trajectory only; tokens are still read out as the argmax of the conditional logits at the final step $t = 0$, exactly as in Sec. 4.1 and Alg. 2. Also, LCFG does not cost extra compute compared with the embedding-space counterpart, since we need to get both conditional and unconditional logit predictions.

Experimental setup. We use the setting identical to the default in App. E.5. We use the same *Sigma-unreg*-3B checkpoint with block length 32, the full GSM8K and HumanEval evaluation sets, and generate 10 responses per problem. DDIM uses 64 steps/block on a uniform-γ time discretization with $\tau = 0.3$, and DDPM uses 256 steps/block on the EDM $\rho = 7$ time discretization and $\tau = 1$.

Results. Logit-space CFG (LCFG) peaks at $w = 2$ in all four settings, whereas embedding-space CFG improves up to $w = 4 \sim 10$; at their respective optima, LCFG gains only 0.30 points on GSM8K/DDIM and loses 2.44 $\sim$ 17.26 points elsewhere (Tab. 14). Logit temperature (LT) provides effective sharpening without a second branch: on DDPM, lowering T from 1 to 1/5 raises GSM8K from 52.44 to 79.11, while the MAP endpoint $T = 0$ retains 78.61 (Tab. 15). Relative to matched sharpening ($T = 1/w$), the unconditional branch in LCFG contributes negligibly at $w = 2$ and hurts at larger weights. LT also complements ST, adding 9.42 points on DDIM with ST = 0.3, but reduces accuracy when combined with both CFG and ST (Tab. 16). We therefore retain embedding-space CFG and ST as the default choice.

E.7. Ablation Study: Commit-Stride Sampling

[Back to Sec. 4 in the main paper.]

Table 14 | Logit-space versus embedding-space CFG. GSM8K is accuracy (%) and HumanEval is execution-based pass@1 (%). Entries are mean ± two standard errors over the 10 generations per prompt, an approximate 95% confidence interval for the mean, and **bold** marks the best entry in each row. The last column is the best value over the *full* sweep of Tabs. 12 and 13, which continues past $w = 5$ for the embedding rows. The logit rows were swept only to $w = 5$ as they already peak at $w = 2$ and fall monotonically after that.

Setting	Guidance space	$w = 1$	$w = 2$	$w = 3$	$w = 4$	$w = 5$	Tuned best
GSM8K, DDIM 64 steps, $\tau = 0.3$	Logit	68.13 (± 0.38)	**83.49** (± 0.22)	81.55 (± 0.41)	77.08 (± 0.41)	67.78 (± 0.53)	**83.49** (± 0.22) ($w = 2$)
	Embedding (ours)	68.13 (± 0.38)	82.52 (± 0.21)	82.84 (± 0.28)	83.04 (± 0.20)	**83.19** (± 0.18)	83.19 (± 0.18) ($w = 5$)
GSM8K, DDPM 256 steps, $\tau = 1$	Logit	53.09 (± 0.85)	**76.76** (± 0.50)	75.16 (± 0.79)	69.48 (± 0.59)	60.77 (± 0.99)	76.76 (± 0.50) ($w = 2$)
	Embedding (ours)	53.09 (± 0.85)	77.12 (± 0.66)	80.04 (± 0.39)	80.99 (± 0.48)	**81.30** (± 0.41)	**82.43** (± 0.50) ($w = 10$)
HumanEval, DDIM 64 steps, $\tau = 0.3$	Logit	42.93 (± 0.66)	**53.78** (± 1.09)	39.15 (± 1.13)	20.67 (± 0.84)	12.68 (± 0.47)	53.78 (± 1.09) ($w = 2$)
	Embedding (ours)	42.93 (± 0.66)	53.96 (± 0.95)	56.04 (± 1.16)	**56.22** (± 0.83)	54.09 (± 1.22)	**56.22** (± 0.83) ($w = 4$)
HumanEval, DDPM 256 steps, $\tau = 1$	Logit	13.23 (± 1.50)	**29.70** (± 2.69)	22.68 (± 1.36)	14.51 (± 2.14)	9.76 (± 0.48)	29.70 (± 2.69) ($w = 2$)
	Embedding (ours)	13.23 (± 1.50)	32.62 (± 1.98)	40.55 (± 1.79)	44.63 (± 1.07)	**46.71** (± 1.82)	**46.95** (± 1.73) ($w = 9$)

Table 15 | Logit temperature, GSM8K accuracy (%) with DDPM, 256 steps/block, EDM $\rho = 7$ schedule, no CFG. Score entries are mean ± two standard errors over the 10 generations per prompt, an approximate 95% confidence interval for the mean. $T = 1$ is the unsharpened MMSE denoiser and is the same configuration as the $w = 1$ logit-space entry of Tab. 14. The bottom row is LCFG at the matched weight $w = \frac{1}{T}$, repeated from Tab. 14; the two are the same operation if the unconditional logits are uniform over the vocabulary, so the gap isolates what logit-space guidance adds over plain sharpening.

T	1	$\frac{1}{2}$	$\frac{1}{3}$	$\frac{1}{4}$	$\frac{1}{5}$	$\frac{1}{10}$	0 (MAP)
LT at T	52.44 (± 1.00)	76.88 (± 0.62)	79.07 (± 0.38)	78.97 (± 0.50)	**79.11** (± 0.32)	78.74 (± 0.23)	78.61 (± 0.33)
LCFG at $w = \frac{1}{T}$	52.44 (± 1.00)	76.76 (± 0.50)	75.16 (± 0.79)	69.48 (± 0.59)	60.77 (± 0.99)	—	—
Δ (LT − LCFG)	0.00	+0.12	+3.91	+9.49	+18.34	—	—

Table 16 | Composing logit temperature with the other two knobs: GSM8K accuracy (%) with DDIM, 64 steps/block, uniform-γ schedule, ST $\tau = 0.3$. Entries are mean ± two standard errors over the 10 generations per prompt, an approximate 95% confidence interval for the mean. **Bold** marks the best entry in each row.

T	1	$\frac{1}{2}$	$\frac{1}{3}$	$\frac{1}{5}$	0 (MAP)
No guidance ($w = 1$)	68.18 (± 0.33)	**77.60** (± 0.23)	77.42 (± 0.35)	76.13 (± 0.31)	71.83 (± 0.24)
Embedding CFG at $w = 5$	**83.19** (± 0.18)	81.44 (± 0.19)	81.49 (± 0.25)	79.96 (± 0.31)	76.84 (± 0.44)

Beyond blockwise decoding of L' tokens per denoising window, we further introduce a *commit stride* $S \in \{1, \ldots, L'\}$. Each denoising window still runs the same blockwise reverse process over a block of fresh Gaussian noise with length $L' = 32$, but only the leading $S \in \{8, 16, 32\}$ tokens are committed back into the causal KV cache; the remaining $L' - S$ tail positions are discarded and re-drawn as fresh noise in the next window (which is shifted forward by S absolute positions). Setting $S \leftarrow L'$ recovers the default block-decoding method. The pseudocode for commit stride can be found in Alg. 2. This is a pure inference-time technique that further improves the output quality even under matched compute budgets, as we will see below.

Setup. We evaluate on the base models of Tabs. 1 and 2: *Sigma-unreg*-3B/8B, both with block size $L' = 32$. Decoding invariants match the diffusion-mode rows of Tab. 1. For each stride, CFG w and ST τ are chosen based on the number of DDIM steps per window (Tab. 10): $(w, \tau) = (3, 0.2)$ at ≥ 64 steps, $(3, 0.3)$ at 32 steps, $(3, 0.5)$ at 16 steps, and $(3, 0.8)$ at 8 steps. All cells generate 10 samples for each prompt and report mean ± two standard errors (App. E.3).

Nominal vs. true NFE. For simplicity, we match compute cost by *nominal* NFE, defined as the number of denoiser forward passes required to commit $L' = 32$ tokens under the given (steps, stride) pair:

$$\text{Nominal NFE} := \frac{NL'}{S},$$

where N is the number of diffusion steps per denoising window. Since every denoising window operates on the *same* L'-wide noisy block appended to the same-length clean prefix, one forward pass costs the same FLOPs

regardless of S.

Table 17 | **Commit-stride ablation on *Sigma-unreg*-3B.** Same six generative reasoning tasks as Tab. 1. *Nom. NFE* is the nominal number of forward passes per 32 committed tokens (i.e., steps/stride × 32). All DDIM rows use the uniform-γ schedule; the DDPM row uses the EDM $\rho = 7$ schedule. Rows within each nominal-NFE bucket are directly compute-comparable. Mean ± two standard errors across 10 sampling lanes. **Bold** marks the best value per column across the stride sweep.

Sampler	Nom. NFE	Steps	Stride	GSM8K	Minerva Math	HumanEval	HumanEval+	MBPP	MBPP+	Avg.
DDPM (w=2, τ=0.1)	256	256	32	84.50 (± 0.31)	64.60 (± 0.28)	59.45 (± 1.30)	53.84 (± 1.29)	52.32 (± 0.73)	71.32 (± 0.91)	64.34
DDIM (w=3, τ=0.2)	256	128	16	85.70 (± 0.15)	64.68 (± 0.20)	60.67 (± 0.80)	54.88 (± 0.84)	55.52 (± 0.36)	72.14 (± 0.34)	65.60
DDIM (w=3, τ=0.2)	256	64	8	84.94 (± 0.17)	**64.73** (± 0.18)	61.52 (± 0.76)	56.10 (± 0.62)	**58.86** (± 0.35)	**75.87** (± 0.40)	**67.00**
DDIM (w=3, τ=0.2)	128	128	32	83.78 (± 0.11)	63.89 (± 0.13)	59.15 (± 0.68)	53.17 (± 0.87)	50.48 (± 0.21)	68.12 (± 0.51)	63.10
DDIM (w=3, τ=0.2)	128	64	16	**85.82** (± 0.09)	64.24 (± 0.14)	60.73 (± 0.83)	54.63 (± 0.85)	54.06 (± 0.30)	71.56 (± 0.44)	65.17
DDIM (w=3, τ=0.3)	128	32	8	84.17 (± 0.29)	64.44 (± 0.22)	**62.13** (± 0.63)	**56.34** (± 0.49)	57.56 (± 0.61)	74.68 (± 0.50)	66.55
DDIM (w=3, τ=0.2)	64	64	32	82.93 (± 0.15)	63.93 (± 0.13)	56.89 (± 0.73)	50.91 (± 0.66)	49.26 (± 0.27)	66.35 (± 0.35)	61.71
DDIM (w=3, τ=0.3)	64	32	16	85.21 (± 0.25)	63.44 (± 0.16)	59.82 (± 0.92)	54.27 (± 1.16)	51.88 (± 0.62)	69.87 (± 0.49)	64.08
DDIM (w=3, τ=0.5)	64	16	8	83.95 (± 0.28)	62.97 (± 0.23)	59.63 (± 1.06)	53.66 (± 0.94)	54.90 (± 0.50)	73.10 (± 0.57)	64.70
DDIM (w=3, τ=0.3)	32	32	32	81.91 (± 0.32)	62.76 (± 0.18)	56.59 (± 1.31)	50.67 (± 1.00)	47.26 (± 0.46)	63.49 (± 0.77)	60.45
DDIM (w=3, τ=0.5)	32	16	16	82.40 (± 0.38)	62.37 (± 0.21)	56.77 (± 0.82)	51.52 (± 0.90)	50.12 (± 0.71)	67.72 (± 0.46)	61.82
DDIM (w=3, τ=0.8)	32	8	8	80.60 (± 0.47)	59.48 (± 0.22)	52.56 (± 1.94)	46.95 (± 1.86)	46.50 (± 0.80)	60.50 (± 1.02)	57.76

Table 18 | **Commit-stride ablation on *Sigma-unreg*-8B.** Same six tasks as Tab. 2; conventions match Tab. 17. **Bold** marks the best value per column across the stride sweep.

Sampler	Nom. NFE	Steps	Stride	GSM8K	Minerva Math	HumanEval	HumanEval+	MBPP	MBPP+	Avg.
DDPM (w=2, τ=0.1)	256	256	32	89.07 (± 0.28)	67.47 (± 0.20)	**61.40** (± 1.73)	**57.44** (± 1.62)	64.78 (± 0.54)	77.83 (± 0.72)	69.66
DDIM (w=3, τ=0.2)	256	128	16	88.64 (± 0.11)	67.59 (± 0.10)	60.91 (± 0.64)	55.67 (± 0.60)	67.44 (± 0.35)	**81.75** (± 0.33)	70.33
DDIM (w=3, τ=0.2)	256	64	8	**90.17** (± 0.15)	**68.82** (± 0.20)	60.49 (± 0.98)	53.96 (± 0.93)	**67.72** (± 0.21)	81.35 (± 0.28)	**70.42**
DDIM (w=3, τ=0.2)	128	128	32	88.70 (± 0.14)	67.28 (± 0.09)	58.48 (± 1.00)	52.68 (± 1.00)	63.22 (± 0.28)	76.85 (± 0.31)	67.87
DDIM (w=3, τ=0.2)	128	64	16	88.76 (± 0.17)	67.63 (± 0.17)	59.33 (± 1.06)	53.54 (± 0.94)	66.62 (± 0.42)	80.90 (± 0.31)	69.46
DDIM (w=3, τ=0.3)	128	32	8	89.61 (± 0.24)	68.11 (± 0.15)	61.22 (± 0.55)	55.18 (± 0.69)	66.22 (± 0.49)	80.19 (± 0.34)	70.09
DDIM (w=3, τ=0.2)	64	64	32	88.68 (± 0.19)	67.30 (± 0.18)	58.90 (± 0.71)	52.56 (± 0.54)	61.96 (± 0.36)	76.35 (± 0.28)	67.62
DDIM (w=3, τ=0.3)	64	32	16	88.15 (± 0.33)	67.58 (± 0.21)	58.05 (± 1.09)	52.80 (± 0.98)	65.32 (± 0.41)	78.10 (± 0.59)	68.33
DDIM (w=3, τ=0.5)	64	16	8	88.45 (± 0.31)	67.21 (± 0.26)	61.40 (± 1.73)	55.67 (± 1.31)	61.92 (± 0.77)	75.21 (± 0.73)	68.31
DDIM (w=3, τ=0.3)	32	32	32	87.78 (± 0.33)	67.13 (± 0.15)	58.54 (± 1.04)	53.17 (± 1.27)	59.54 (± 0.57)	74.15 (± 0.43)	66.72
DDIM (w=3, τ=0.5)	32	16	16	87.04 (± 0.35)	67.14 (± 0.27)	58.41 (± 1.22)	53.29 (± 1.19)	60.46 (± 0.39)	73.41 (± 0.51)	66.63
DDIM (w=3, τ=0.8)	32	8	8	85.91 (± 0.29)	64.57 (± 0.22)	55.98 (± 1.46)	50.61 (± 1.75)	52.74 (± 0.85)	64.44 (± 0.87)	62.38

Results. We present the results for commit-stride sampling for *Sigma-unreg*-3B/8B in Tabs. 17 and 18, respectively. The tables sweep all three strides $S \in \{8, 16, 32\}$ within each nominal-NFE bucket $\{256, 128, 64, 32\}$. At the top NFE (nominal 256), the DDIM 64 steps / stride 8 row is the best on both models: it beats the tuned DDPM-256 baseline of Tabs. 1 and 2 by +2.66 average points on 3B and +0.76 on 8B, with the largest per-task gains on the coding benchmarks such as MBPP. The same ordering holds down to around nominal 64: committing fewer tokens per window (i.e., smaller stride S) improves the average monotonically. At nominal 128 NFE, DDIM 32 steps / stride 8 reaches within a point of the top budget on both sizes, and still above the full-stride DDPM-256 baselines. The benefit of a smaller stride disappears once it forces the per-window step count below roughly 16: at nominal 32, stride 8 leaves only 8 steps per window, and its coarse time grid collapses quality due to larger discretization error, falling below the stride-16 and stride-32 rows at the same budget.

E.8. Pass@k Evaluation

[Back to Sec. 4.4 in the main paper.]

We study inference-time scaling in two distinct evaluation regimes. First, we compare the pretrained 3B models on MBPP; second, we compare the instruct 3B models on AIME 2025.

MBPP for base models. We compare *Sigma*-3B, *Sigma-distill*-3B and NLD-3B (Fu et al., 2026) on MBPP, drawing 256 generations per problem for every configuration. Both models use block size 32. *Sigma*-3B uses DDPM with 256 steps/block and the EDM $\rho = 7$ schedule, sweeping CFG $w \in \{1, 2, 3\}$ and ST $\tau \in \{0.1, 0.4, 0.7, 1.0\}$; NLD-3B uses 32 transfer steps per block, sweeping confidence threshold $\vartheta \in \{0.8, 0.9\}$ and temperature $T \in \{0.3, 0.6, 1.0\}$; *Sigma-distill*-3B use 4 and 8 NFEs and dedicated sampler without any

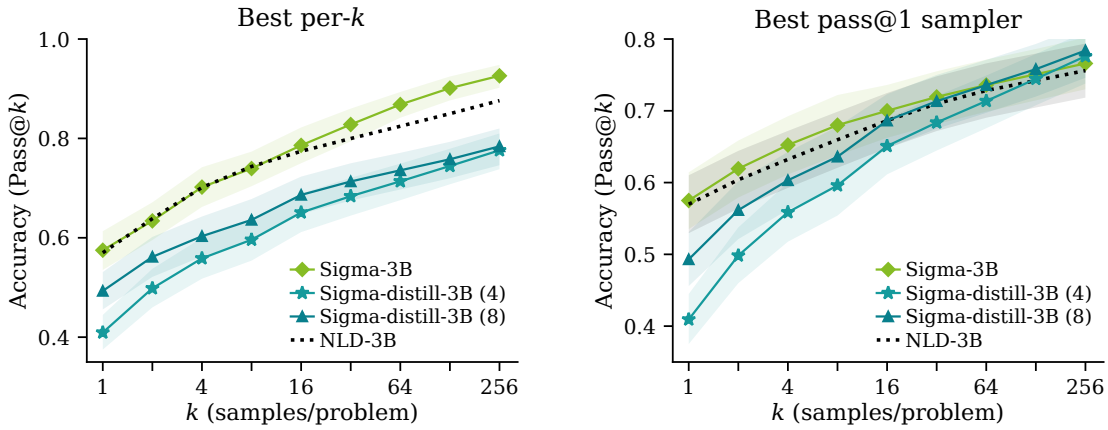


Figure 17 | **Pass@k on MBPP of *Sigma*-3B, *Sigma-distill*-3B and NLD-3B.** *Left:* At each k, each family uses the best configuration in its tested sampler grid; *Right:* Fixed sampler configuration picked by highest pass@1 accuracy: NLD-3B ($\vartheta = 0.9, T = 0.3$), *Sigma*-3B ($DDPM, w = 2, \tau = 0.1$), *Sigma-distill*-3B have a fixed sampler (no CFG/ST).

post-hoc guidance. Candidate draws use different random seeds while preserving identical prompts and few-shot examples across configurations, as in App. E.3.

The left panel of Fig. 17 shows the best performance available in each tested sampler grid at every generation budget. As k grows, the *Sigma*-3B setting carrying this curve moves from $\tau = 0.1$ to 0.4, 0.7, and finally 1.0, shifting from single-sample quality toward broader coverage. the right panel provides the complementary fixed-configuration result tuned on pass@1 (i.e. that in Tab. 1). *Sigma*-3B consistently beats NLD=3B by a small, almost constant margin. However, most interesting is the behavior of *Sigma-distill*-3B which with only 8-NFEs matches the *Sigma*-3B teacher from $k = 32$ onward, and the 4-NFE version closes the gap as k increases, finally matching *Sigma*-3B at $k = 256$. This demonstrates that our distilled models exercise *useful* diversity.

AIME 2025 for instruct models. We repeat the comparison on the *instruct* models, under the same protocol as the benchmark table in Tab. 4 (harness described in App. G.2), on AIME 2025. *Sigma*-3B–*instruct* uses DDIM with 32 steps and the uniform-γ schedule, sweeping the five settings $(w, \tau) \in \{(3, 0.3), (4, 0.3), (4, 0.7), (5, 0.7), (3, 1.0)\}$; NLD-3B-Instruct uses confidence-threshold masked decoding with $\vartheta = 0.9$ and $T \in \{0.2, 0.6, 1.0\}$.

The response budgets are 4096 tokens for *Sigma*-3B–*instruct* and 16384 for NLD-3B-Instruct, but the shorter *Sigma* cap is not binding since the reasonable generations (i.e., the samples that are not degenerate, such as repeating the same pattern without emitting EOS token) are all within this length range for both models. The results are in Fig. 18.

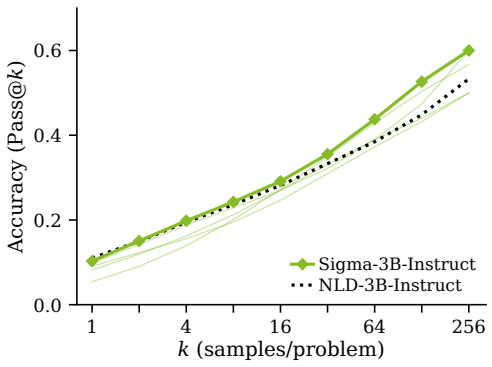


Figure 18 | **Pass@k on AIME 2025 of *Sigma*-3B-Instruct and NLD-3B-Instruct.** At each k, each family uses the best configuration in its tested sampler grid; Best overall *Sigma*-3B-Instruct configuration is highlighted, the other thin lines correspond to the other configs.

E.9. Wall-Clock Cost of the Packed CFG Forward

[Back to Sec. 4.2 in the main paper.]

We measure the wall-clock cost of the packed CFG forward with attention mask (Fig. 5a) at batch size 1 compared with the unguided forward. We decode the same set of 16 GSM8K questions (8-shot, prompt length $729 \sim 814$) with the DDIM sampler at 32 steps per block, on the first block only. Each

Table 19 | Wall-clock cost per denoiser evaluation (NFE) of packing the conditional and unconditional CFG branches into one forward pass, at batch size 1. Times are in milliseconds on H100.

Block Size	Stage	No CFG (ms)	Packed CFG (ms)	Δ	Δ (%)
$L' = 32$	Transformer forward	18.06	18.38	$+0.322 \pm 0.181$	+1.8
	Logit head + cat. reparam.	1.02	1.50	$+0.481 \pm 0.008$	+47.2
	Total per NFE	19.08	19.89	$+0.804 \pm 0.186$	+4.2
$L' = 64$	Transformer forward	18.23	18.21	-0.021 ± 0.159	−0.1
	Logit head + cat. reparam.	1.08	1.60	$+0.519 \pm 0.006$	+48.3
	Total per NFE	19.31	19.80	$+0.498 \pm 0.162$	+2.6

question therefore requires $32+1$ forward passes (Alg. 2). The very first denoiser query of each process involves a one-time CUDA initialization which takes a longer time, so is excluded to avoid biasing the measurement. All runs are performed on one NVIDIA H100 GPU, using *Sigma-unreg*-3B with precision `BF16`.

Results in Tab. 19 show that at $L' = 32$ the packed transformer forward costs +1.8% over the unguided one, and at $L' = 64$ it is free to within measurement noise. The per-NFE overhead is dominated by computing the second set of logits, doing softmax, and multiplying by the embedding matrix rather than by attention.

E.10. Reasoning Trajectory Analysis

[Back to Sec. 4.4 in the main paper.]

As noted in Sec. 4.4, we empirically observe less left-to-right bias within a block for *Sigma* compared with the confidence-based masked dLM sampling. We demonstrate this in Fig. 19, where we show a commit order analysis between continuous and discrete dLMs.

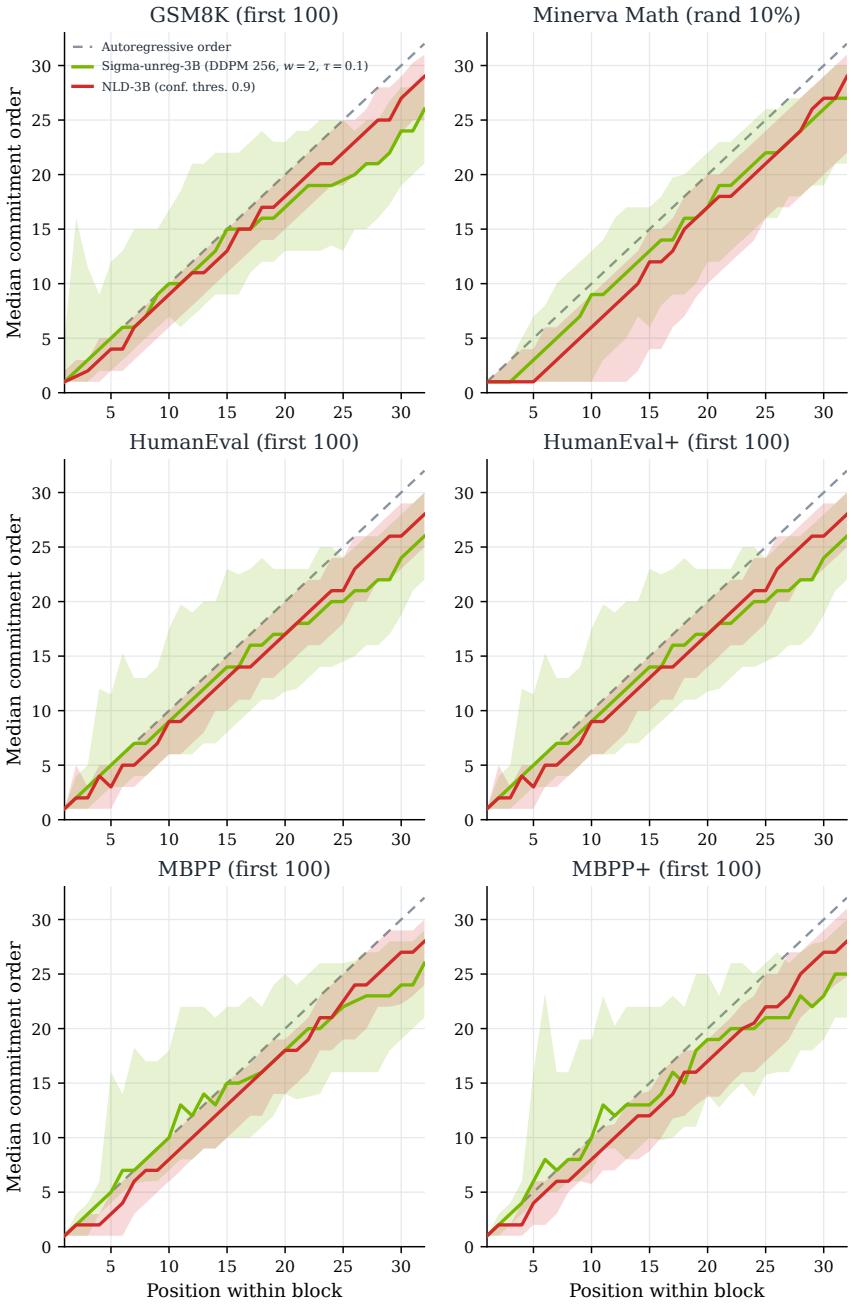


Figure 19 | **Token commitment order vs token position for *Sigma-unreg*-3B** *Sigma* uses 256-step DDPM ($w = 2, \tau = 0.1$) as in Tab. 1. Generation order averaged across multiple documents and blocks. For *Sigma-unreg*-3B, the commitment step t_i of position i is defined as the smallest t such that for every step $t' \geq t$ up to the final readout, (i) the argmax of $\mathbf{x}_\theta$ equals the finally decoded token, and (ii) that argmax is assigned probability at least 0.9. The commitment order of position i is the rank of t_i among $t_1, \ldots, t_{32}$ within its block, in ascending order. Positions sharing a commitment step share the lowest rank of the tie.

F. Implementation Details and Additional Results for Few-step Distillation

F.1. Method and Training Objective

We initialize a student with parameters ϕ from the teacher parameters θ, and devise a continuous relaxation of Distribution Matching Distillation (DMD, Yin et al. (2024b)) for language models, inspired by Le Van Kiem et al. (2026) and Fan et al. (2026), to train the student to estimate a clean-context-conditional teacher-generated data distribution $p_{\text{teacher}}(\cdot \mid \mathbf{c})$. In the following we omit the context conditioning for readability. As detailed in App. F.2, this distribution is obtained using the strongest teacher sampler for optimal distributional quality (high NFE and guidance). We choose p_{teacher} as the target distribution instead of using the pretraining dataset, as we observed that distillation performance was better for the former, likely due to the inductive bias existing in self-distillation since the student is initialized from the teacher. We ablate this choice along with other design decisions in App. F.3.

We train the student on-policy for a specific number of NFEs M, i.e., the student should produce its own multi-step mapping from the Gaussian prior to the target data distribution by matching intermediate noisy marginals, following the principle in Fan et al. (2026). The training procedure is detailed in Alg. 3.

At each training iteration, we pick a clean target sample $\mathbf{e} \sim p_{\text{teacher}}$ and independently initialize the rollout with a near-standard-Gaussian state $\mathbf{e}_1 \sim q_1(\mathbf{0}, \gamma_1)$. The student is rolled out for $M_s \sim \text{Unif}\{0, \dots, M-1\}$ steps from $\mathbf{e}_1$ until a *source state* $\mathbf{e}_s$ with noise level $\gamma_s = \gamma_1 - \frac{M_s}{M}(\gamma_1 - \gamma_0)$ using self-conditioning. The multi-step sampler we use for rollout as well as during inference is the consistency-model-style procedure used in Yin et al. (2024a); Song et al. (2023):

$$\mathbf{e}_{t_{n-1}} = \alpha_{t_{n+1}} \mathbf{e}_0^{\phi,t_n} + \sigma_{t_{n+1}} \boldsymbol{\epsilon} \qquad \mathbf{e}_0^{\phi,t_n} = \mathbf{e}_\phi(\mathbf{e}_{t_n}, \gamma_{t_n}) \qquad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), \tag{17}$$

This means the intermediate states have the exact same noise marginals as the forward variance-preserving process, but the state is centered around the current clean estimate. We detach this prefix rollout and treat its source-state distribution as fixed during the selected student update: the student objective is not differentiated through the earlier rollout steps.

From the given source state $\mathbf{e}_s$, the student is called to produce a clean embedding estimate $\mathbf{e}_0^{\phi,s} := \mathbf{e}_\phi(\mathbf{e}_s, \gamma_s)$, which is re-noised to a *query noise level* sampled uniformly within the segment starting from the source noise level $\gamma_t \sim \text{Unif}\left[\gamma_1 - \frac{M_s+1}{M}(\gamma_1 - \gamma_0), \gamma_1 - \frac{M_s}{M}(\gamma_1 - \gamma_0)\right]$. This together yields the re-noised student sample $\mathbf{e}_t^{\phi,s} \sim q_t(\mathbf{e}_0^{\phi,s} \mid \gamma_t)$ and we denote the resulting marginal as $q_t^{\phi,s}$. Similarly, the target embedding $\mathbf{e}$ is re-noised to the same query level, yielding a sample $\mathbf{e}_t \sim q_t(\mathbf{e}, \gamma_t)$.

These choices of source and query distribution correspond to a specific choice of components described in Fan et al. (2026). Like Fan et al. (2026), we train our students in an on-policy fashion; however, we use fully stochastic transitions as in Yin et al. (2024a); Le Van Kiem et al. (2026) whereas Fan et al. (2026) used deterministic DDIM transitions. Furthermore, we do not use phase experts but only one set of student weights. As we ablate it later, this specific choice of source and query noise level distributions has a large influence on the overall student quality.

We match the marginals $q_t^{\phi,s}$ and q_t using reverse-KL, which smooths out the training objective compared to directly trying to match clean embedding distributions (Yin et al., 2024b):

$$\mathcal{K}_{s,t}(\phi) = \text{KL}\left(q_t^{\phi,s} \,\|\, q_t\right), \qquad \mathcal{L}_{\text{DMD}}^{\phi} = \mathbb{E}_{\gamma_s,\gamma_t}\left[\mathcal{K}_{s,t}(\phi)\right]. \tag{18}$$

Deriving the gradient of $\mathcal{L}_{\text{DMD}}^{\phi}$ (Le Van Kiem et al., 2026) yields the following gradient surrogate loss:

$$\mathcal{L}_{\text{DMD-grad}}^{\phi} = \mathbb{E}_{\gamma_t,\gamma_s,\mathbf{e}_t^{\phi,s} \sim q_t(\mathbf{e}_0^{\phi,s} \mid \gamma_t)}\left[\, \mathrm{e}^{-\gamma_t} \sum_{l=1}^{L'} \left\langle \mathbf{e}_0^{\phi,s,l}, \text{sg}\left[\mathbf{e}_\eta^l(\mathbf{e}_t^{\phi,s}, \gamma_t) - \mathbf{e}_\psi^l(\mathbf{e}_t^{\phi,s}, \gamma_t)\right] \right\rangle \right], \tag{19}$$

with *reference denoiser* $\mathbf{e}_\psi$ (linked to the so-called "real score estimator" in the literature) and the *auxiliary denoiser* $\mathbf{e}_\eta$ (linked to the "fake score estimator" in the literature). The full derivation is presented in App. F.4. The reference denoiser $\mathbf{e}_\psi$ denoises the re-noised student's samples toward the target distribution p_{teacher}, while the auxiliary denoiser $\mathbf{e}_\eta$ denoises the re-noised student's samples toward the student distribution. Steering the student with the detached difference of these two denoisers in (19) matches the student and

target distributions. Following Le Van Kiem et al. (2026), we train the categorical auxiliary denoiser with a soft-target cross-entropy against the detached student-produced probabilities:

$$\mathcal{L}^{\eta}_{\text{aux}} = \mathbb{E}_{\gamma_t,\gamma_s,\mathbf{e}_t^{\phi,s}\sim q_t(\mathbf{e}_0^{\phi,s}|\gamma_t)}\left[-\sum_{l=1}^{L'}\left\langle \text{sg}[\mathbf{x}^{\phi,s,l}],\, \log \mathbf{x}_\eta^l(\text{sg}[\mathbf{e}_t^{\phi,s}],\gamma_t)\right\rangle\right], \tag{20}$$

with $\mathbf{e}_\eta = \mathbf{x}_\eta \mathbf{E}$. Usually the teacher diffusion model serves directly as $\mathbf{e}_\psi$ since it is trained as a denoiser of the pretraining data distribution p_{data}. However, since our target distribution is p_{teacher}, i.e., teacher-generated samples rather than the pretraining distribution, we re-fit the reference denoiser (initialized from the teacher) to p_{teacher} using cross-entropy:

$$\mathcal{L}^{\psi}_{\text{ref}} = \mathbb{E}_{\gamma_u\sim\text{Unif}[\gamma_0,\gamma_1],\mathbf{e}\sim p_{\text{teacher}},\mathbf{e}_u\sim q_t(\mathbf{e}|\gamma_u)}\left[-\sum_{l=1}^{L'}\left\langle \mathbf{x}^l,\, \log \mathbf{x}_\psi^l(\mathbf{e}_u,\gamma_u)\right\rangle\right], \tag{21}$$

with $\mathbf{e} = \mathbf{x}\mathbf{E}$ and $\mathbf{e}_\psi = \mathbf{x}_\psi \mathbf{E}$.

The auxiliary denoiser, the reference denoiser and the main student all use same-state two-call training for self-conditioning (Chen et al., 2023) with probability $p_{\text{SC}} = 0.25$ while inference carries the previous raw clean prediction, thereby mimicking the original teacher's training and inference behaviors. The complete training procedure is summarized in Alg. 3.

F.2. Implementation Details

Target distribution. The target distribution is a teacher-generated bank containing 65,536 clean prefixes of 256 tokens, taken from the pretraining dataset, and four teacher-generated 32-token continuation blocks per prefix. This bank is generated with the empirically optimal teacher sampler, i.e., 256-step DDPM sampling with self-conditioning enabled and optimal guidance: for *Sigma*-3B this is CFG $w = 2$ and ST $\tau = 0.1$, whereas for *Sigma*-8B this is CFG $w = 3$ and ST $\tau = 0.1$.

Parameterization and optimization. All three neural networks are initialized from the same respective teacher checkpoint, with separate trainable parameter copies and optimizers. We copy the embedding matrix, noise schedule and noise endpoints from the teacher and retain them throughout distillation. We detach the source rollout and the SC preliminary prediction, so student gradients pass only through the selected denoising transition, similar to Fan et al. (2026). We prefit the reference denoiser $\mathbf{e}_\psi$ for 50 optimizer updates before starting student/auxiliary updates. Each subsequent student update is preceded by four auxiliary updates and four reference updates. We optimize all networks with AdamW with a global batch size of 256, $\beta_1 = 0.9$, $\beta_2 = 0.999$, zero weight decay, and unit-norm gradient clipping. The learning rates for each network are selected through gradient balancing and are 3.6×10^{-7} for the student, 1.188×10^{-5} for the auxiliary denoiser, and 10^{-6} for the reference denoiser.

In practice, we use the residual-normalization strategy in Yin et al. (2024a) and remove the SNR weight $\mathrm{e}^{-\gamma_t}$, thereby replacing (19) with

$$\mathcal{L}^{\phi}_{\text{DMD2-grad}} = \mathbb{E}_{\gamma_t,\gamma_s,\mathbf{e}_t^{\phi,s}\sim q_t(\mathbf{e}_0^{\phi,s}|\gamma_t)}\left[\frac{1}{R}\sum_{l=1}^{L'}\left\langle \mathbf{e}_0^{\phi,s,l}, \text{sg}\left[\mathbf{e}_\eta^l(\mathbf{e}_t^{\phi,s},\gamma_t) - \mathbf{e}_\psi^l(\mathbf{e}_t^{\phi,s},\gamma_t)\right]\right\rangle\right], \tag{22}$$

with the student-to-reference denoiser residual

$$R = 10^{-6} + \frac{1}{L'd_e}\sum_{l=1}^{L'}\text{sg}\left[\left\|\mathbf{e}_0^{\phi,s,l} - \mathbf{e}_\psi^l(\mathbf{e}_t^{\phi,s},\gamma_t)\right\|_1\right]. \tag{23}$$

We further stabilize the auxiliary denoiser training by multiplying its objective in (20) with

$$h_t = \min(1,\, 10\,\sigma^2(\gamma_t)(1 - \mathrm{e}^{\gamma_r - \gamma_t})), \tag{24}$$

where γ_r is the cleaner endpoint of the selected segment.

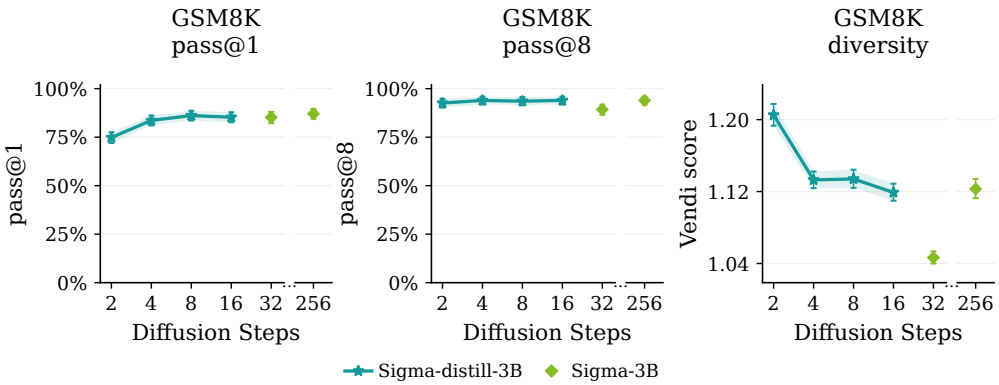


Figure 20 | Diversity and pass@k results on GSM8K 512-prompt subset.

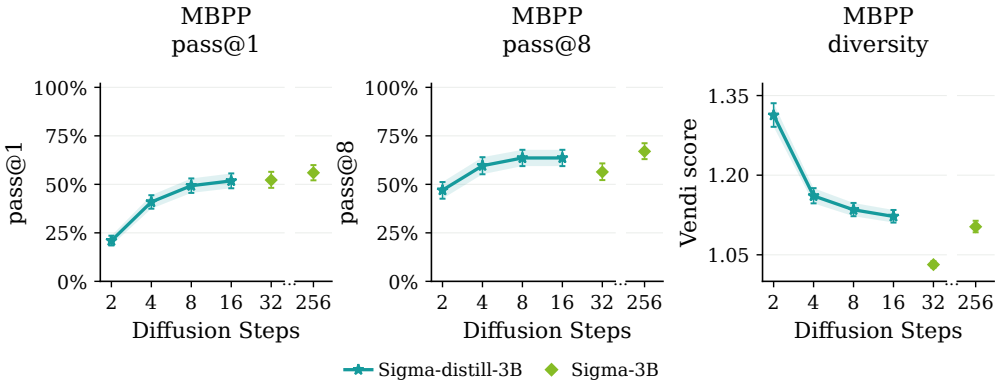


Figure 21 | Diversity and pass@k results on MBPP 500-prompt test set.

F.3. Additional Results

Diversity and pass@k. We demonstrate here that our distribution-matching distillation procedure preserves the diversity of the target teacher distribution, i.e., we do not fall into the pitfall often encountered in DMD methods. We use 512 prompts and sample 8 trajectories per prompt, on 3 benchmarks (GSM8K, MBPP and Novelty-Bench). GSM8K measures math reasoning, MBPP measures coding ability, and Novelty-Bench (Zhang et al., 2025) is an open-ended creative writing benchmark which helps us measure actual text diversity and creativity.

We measure pass@1 and pass@8 accuracies as well as semantic diversity using the Vendi score (Friedman & Dieng, 2023). We compute the Vendi score as the exponentiated Jensen-Shannon entropy of the pairwise similarity matrix between SentenceBERT embeddings of the produced outputs. Results displayed in Figs. 20 to 22 clearly demonstrate that our DMD student recovers the DDPM-256 Vendi score as well as its pass@8 accuracy, with as few as 4 diffusion steps. Furthermore, the DDIM-32 *Sigma*-3B (with self-conditioning) teacher has lower diversity and pass@8 accuracy than both our *Sigma-distill*-3B students and the DDPM-256 *Sigma*-3B, although its pass@1 is somewhat comparable to our 4-NFE *Sigma-distill*-3B. This suggests that our distilled students balance quality and diversity for only a fraction of the inference cost, which is very promising for, e.g., RL-based post-training.

Ablation 1: Teacher. We investigate the influence of the teacher's geometry on the distillation performance. We observed earlier that improving the regularity of the geometry is key for low-NFE performance. Indeed, the unregularized *Sigma-unreg*-3B teacher exhibits local embedding collapse, lowering the global rank participation and creating uncertainty during decoding, which particularly affects coding tasks (see Sec. 4.3 and Tab. 10). This consequence transfers naturally to the distillation quality, as we notice in Fig. 23 that *Sigma*-3B teaches much stronger students at 4- and 8-NFE in particular.

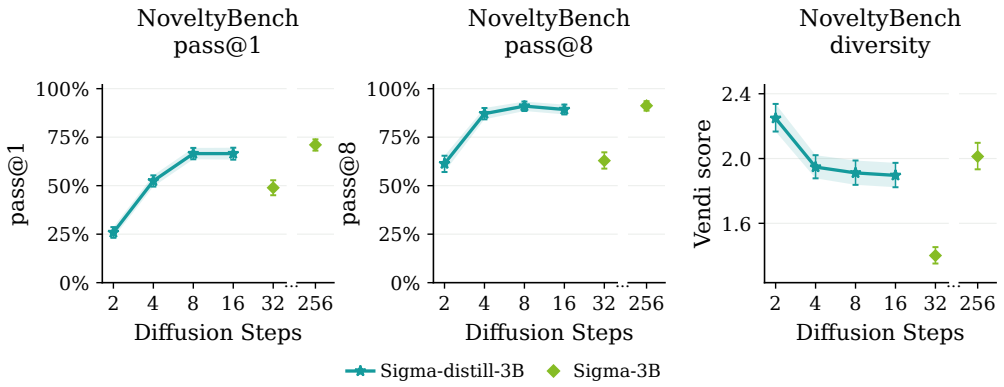


Figure 22 | Diversity and pass@k results on NoveltyBench and NoveltyBench-WildCat 512-prompt test set.

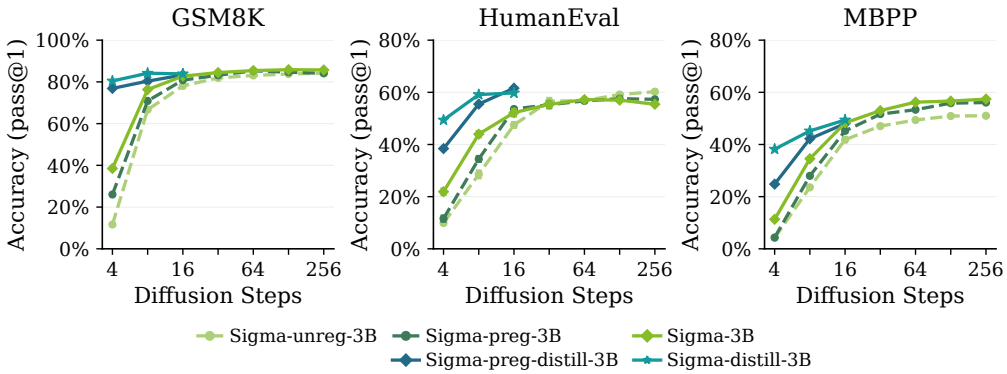


Figure 23 | Teacher ablation for DMD distillation at 3B scale. Global batch size 64.

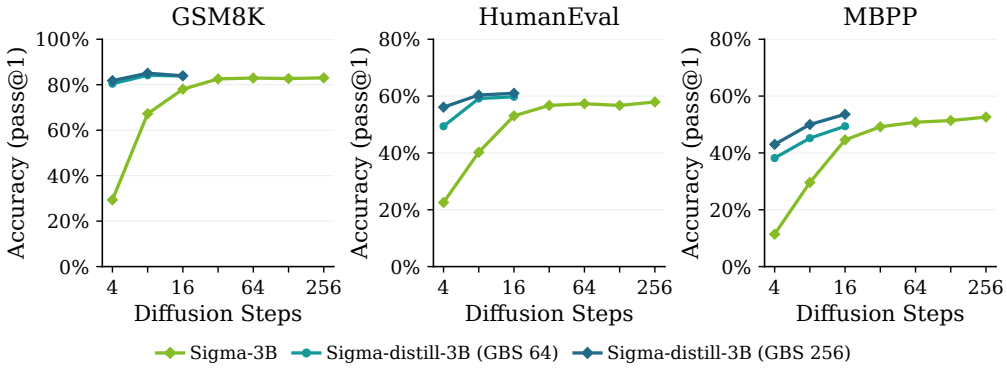


Figure 24 | Global batch size ablation for DMD distillation from *Sigma*-3B teacher.

Ablation 2: Global Batch Size. We ablate the effect of global batch size on distillation quality. Increasing the global batch size from 64 to 256 yields large and consistent improvements across NFEs. Increasing batch size above 256 did not further improve results.

Ablation 3: Query Distributions. Choosing the query distribution is a key design choice in DMD distillation. After obtaining very good early results with Phased DMD (Fan et al., 2026), we ablated the discrepancies between Phased DMD and plain DMD, and noticed that the source states, the transitions and the source and query distributions were the most discriminative factors. As we kept the consistency-style transitions with fresh Gaussian noise and the student rollout for obtaining our input states, the main remaining difference was the query distribution. We choose three candidate distributions: Uniform, Gumbel fitted to

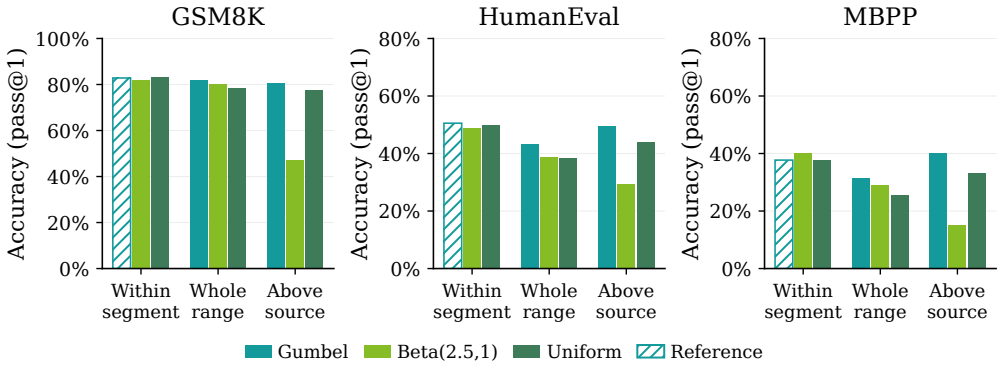


Figure 25 | Query distribution ablation for DMD distillation from *Sigma*-3B teacher. Global batch size 64. $M = 4$ NFE distillation.

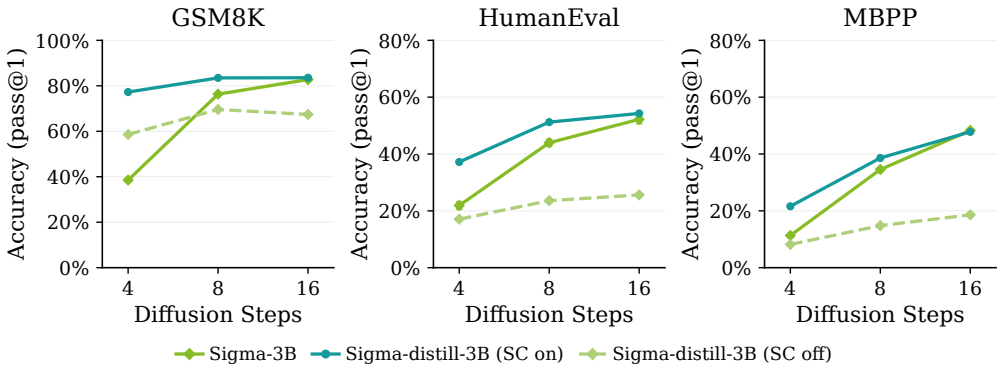


Figure 26 | Self-conditioning ablation for DMD distillation from *Sigma*-3B teacher. Global batch size 64.

the teacher's empirical denoising cross-entropy distribution (skewed toward intermediate noise levels), and $\beta(2.5, 1)$ (skewed toward high noise levels); as well as three ranges for the distribution support: the whole range $[\gamma_0, \gamma_1]$, within segment $[\gamma_r, \gamma_t]$, and above segment $[\gamma_t, \gamma_1]$. We conduct this ablation specifically on the 4-NFE distillation setup. In Fig. 25, we observe that using the current segment as the support consistently obtained the best results, and that the exact distribution did not matter so much there. The teacher-CE-fit Gumbel above segment obtained excellent results too, but we noticed that the resulting distillation procedure was very unstable.

Ablation 4: Self-conditioning and student input. Using self-conditioning in the student is absolutely key to its generation quality. The self-conditioning probability used during distillation hardly influences the result: for simplicity we retain $p_{\text{SC}} = 0.25$. Using $p_{\text{SC}} = 0$ degraded results slightly, but far worse is deactivating self-conditioning at inference, as shown in Fig. 26.

Ablation 5: Training data. We observe in Fig. 27 that teacher-generated data led to better results than directly using the pretraining dataset. We posit this is due to the inductive bias carried by the student about the target distribution, since it is initialized from the teacher that produced this distribution. We conduct this ablation specifically on the 4-NFE distillation setup.

Ablation 6: Student and auxiliary objectives. We ablate the choice of student loss weighting ("Norm off SNR on" (19) vs. "Norm on SNR off" (22)) and the auxiliary denoiser parameterization and objective. "Soft CE" indicates our baseline, i.e., a categorical denoiser (similar to the teacher) trained with cross-entropy on soft probabilities. "MSE" indicates a denoiser augmented with an MLP-parameterized residual connection from Transformer final hidden state to the re-embedded categorical prediction, trained with MSE in the embedding space directly. We conduct this ablation specifically on the 4-NFE distillation setup. The results highlighted in Fig. 28 favor the Soft CE auxiliary objective introduced in Le Van Kiem et al. (2026); the DMD2-type normalization (Yin et al., 2024a) does not play a major role, although we informally observed it

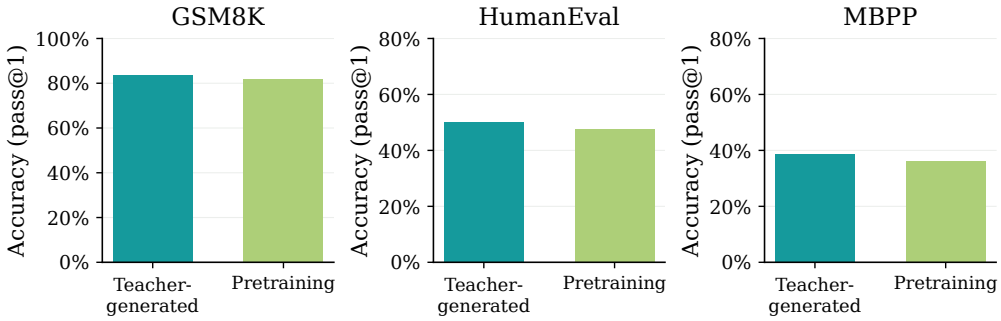


Figure 27 | Data distribution ablation for DMD distillation from *Sigma*-3B teacher. Global batch size 64. $M = 4$ NFE distillation.

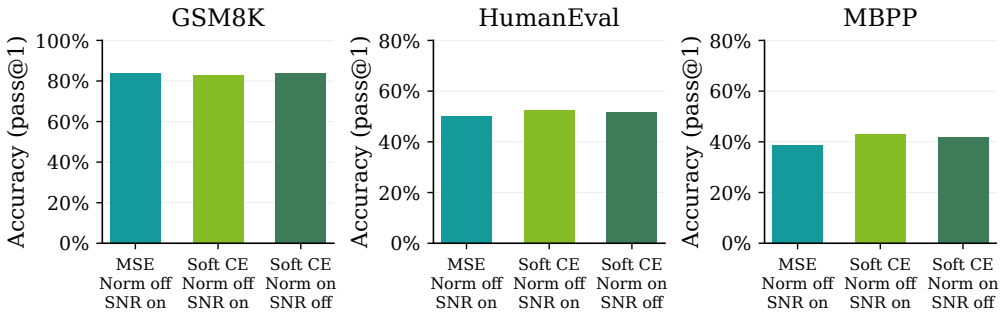


Figure 28 | Student objective ablation for DMD distillation from *Sigma*-3B teacher. Global batch size 64. $M = 4$ NFE distillation.

led to faster and more stable convergence, therefore being favored in our final setup.

F.4. Proof: DMD Gradient Surrogate Loss

In this section, we prove the form of the DMD gradient surrogate loss in (19).

Proof. We first fix the source and query noise levels (γ_s, γ_t). All distributions below are implicitly conditioned on the clean prefix. For this derivation, we hold the source-state sampling distribution fixed, consistently with detaching the on-policy source rollout.

Write $\alpha_t = \alpha(\gamma_t)$ and $\sigma_t = \sigma(\gamma_t)$. The Gaussian corruption admits the reparameterization

$$\mathbf{e}_t^{\phi,s} = \alpha_t \mathbf{e}_0^{\phi,s} + \sigma_t \boldsymbol{\epsilon}, \qquad \mathbf{e}_0^{\phi,s} = \mathbf{e}_\phi(\mathbf{e}_s, \gamma_s), \qquad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}). \tag{25}$$

Our log-noise-to-signal parameterization satisfies

$$\gamma_t = \log \frac{\sigma_t^2}{\alpha_t^2}, \qquad \frac{\alpha_t^2}{\sigma_t^2} = \mathrm{e}^{-\gamma_t}. \tag{26}$$

For fixed (γ_s, γ_t), the per-level objective of (18) reads

$$\mathcal{K}_{s,t}(\phi) = \mathrm{KL}\left(q_t^{\phi,s} \,\|\, q_t\right) = \mathbb{E}_{\mathbf{e}_s, \boldsymbol{\epsilon}}\left[\log q_t^{\phi,s}(\mathbf{e}_t^{\phi,s}) - \log q_t(\mathbf{e}_t^{\phi,s})\right]. \tag{27}$$

There are two contributions when differentiating the student log-density: its explicit dependence on ϕ at a

fixed input, and its dependence through the sampled input $\mathbf{e}_t^{\phi,s}$. Consequently,

$$\nabla_\phi \mathcal{K}_{s,t} = \mathbb{E}_{\mathbf{e}_s,\boldsymbol{\epsilon}}\Big[\nabla_\phi \log q_t^{\phi,s}(\mathbf{e})\Big|_{\mathbf{e}=\mathbf{e}_t^{\phi,s}} + \Big(\frac{\partial \mathbf{e}_t^{\phi,s}}{\partial \phi}\Big)^\top \Big(\nabla_{\mathbf{e}} \log q_t^{\phi,s}(\mathbf{e}_t^{\phi,s}) - \nabla_{\mathbf{e}} \log q_t(\mathbf{e}_t^{\phi,s})\Big)\Big]. \tag{28}$$

The expectation of the explicit density derivative vanishes:

$$\mathbb{E}_{\mathbf{e}\sim q_t^{\phi,s}}\Big[\nabla_\phi \log q_t^{\phi,s}(\mathbf{e})\Big|_{\mathbf{e}\text{ fixed}}\Big] = \int \nabla_\phi q_t^{\phi,s}(\mathbf{e})\, d\mathbf{e} = \nabla_\phi \int q_t^{\phi,s}(\mathbf{e})\, d\mathbf{e} = 0. \tag{29}$$

Thus, introducing the student and target scores

$$\mathbf{s}_{\text{stu}}^{\phi,s}(\mathbf{e},\gamma_t) := \nabla_{\mathbf{e}} \log q_t^{\phi,s}(\mathbf{e}), \qquad \mathbf{s}_{\text{ref}}(\mathbf{e},\gamma_t) := \nabla_{\mathbf{e}} \log q_t(\mathbf{e}), \tag{30}$$

we obtain

$$\nabla_\phi \mathcal{K}_{s,t} = \mathbb{E}_{\mathbf{e}_s,\boldsymbol{\epsilon}}\Big[\Big(\frac{\partial \mathbf{e}_t^{\phi,s}}{\partial \phi}\Big)^\top \Big(\mathbf{s}_{\text{stu}}^{\phi,s}(\mathbf{e}_t^{\phi,s},\gamma_t) - \mathbf{s}_{\text{ref}}(\mathbf{e}_t^{\phi,s},\gamma_t)\Big)\Big]. \tag{31}$$

For Gaussian corruption, the score is related to the posterior mean of the clean embedding by

$$\nabla_{\mathbf{e}_t} \log q_t(\mathbf{e}_t) = \frac{\alpha_t\ \mathbb{E}[\mathbf{e}_0 \mid \mathbf{e}_t] - \mathbf{e}_t}{\sigma_t^2}. \tag{32}$$

Let the ideal auxiliary and reference denoisers be

$$\mathbf{e}_\eta^\star(\mathbf{e}_t,\gamma_t) := \mathbb{E}_{\text{stu}}[\mathbf{e}_0^{\phi,s} \mid \mathbf{e}_t,\gamma_s,\gamma_t], \qquad \mathbf{e}_\psi^\star(\mathbf{e}_t,\gamma_t) := \mathbb{E}_{\text{ref}}[\mathbf{e}_0 \mid \mathbf{e}_t,\gamma_t], \tag{33}$$

where source conditioning of the ideal auxiliary is suppressed in its arguments. The corresponding score difference is therefore

$$\begin{aligned}\mathbf{s}_{\text{stu}}^{\phi,s}(\mathbf{e}_t,\gamma_t) - \mathbf{s}_{\text{ref}}(\mathbf{e}_t,\gamma_t) &= \frac{\alpha_t \mathbf{e}_\eta^\star(\mathbf{e}_t,\gamma_t) - \mathbf{e}_t}{\sigma_t^2} - \frac{\alpha_t \mathbf{e}_\psi^\star(\mathbf{e}_t,\gamma_t) - \mathbf{e}_t}{\sigma_t^2} \\ &= \frac{\alpha_t}{\sigma_t^2}\Big[\mathbf{e}_\eta^\star(\mathbf{e}_t,\gamma_t) - \mathbf{e}_\psi^\star(\mathbf{e}_t,\gamma_t)\Big].\end{aligned} \tag{34}$$

Meanwhile, because the corruption noise and detached source are held fixed under differentiation,

$$\frac{\partial \mathbf{e}_t^{\phi,s}}{\partial \phi} = \alpha_t \frac{\partial \mathbf{e}_0^{\phi,s}}{\partial \phi}. \tag{35}$$

Combining these identities gives

$$\begin{aligned}\nabla_\phi \mathcal{K}_{s,t} &= \mathbb{E}_{\mathbf{e}_s,\boldsymbol{\epsilon}}\Big[\frac{\alpha_t^2}{\sigma_t^2}\Big(\frac{\partial \mathbf{e}_0^{\phi,s}}{\partial \phi}\Big)^\top \Big(\mathbf{e}_\eta^\star(\mathbf{e}_t^{\phi,s},\gamma_t) - \mathbf{e}_\psi^\star(\mathbf{e}_t^{\phi,s},\gamma_t)\Big)\Big] \\ &= \mathbb{E}_{\mathbf{e}_s,\boldsymbol{\epsilon}}\Big[\mathrm{e}^{-\gamma_t}\Big(\frac{\partial \mathbf{e}_0^{\phi,s}}{\partial \phi}\Big)^\top \Big(\mathbf{e}_\eta^\star(\mathbf{e}_t^{\phi,s},\gamma_t) - \mathbf{e}_\psi^\star(\mathbf{e}_t^{\phi,s},\gamma_t)\Big)\Big].\end{aligned} \tag{36}$$

Replacing the ideal posterior means with learned denoisers and averaging over source and query levels as in (18) yields

$$\nabla_\phi \mathcal{L}_{\text{DMD}}^\phi = \mathbb{E}_{\gamma_s,\gamma_t,\mathbf{e}_s,\boldsymbol{\epsilon}}\Big[\mathrm{e}^{-\gamma_t}\Big(\frac{\partial \mathbf{e}_0^{\phi,s}}{\partial \phi}\Big)^\top \Big(\mathbf{e}_\eta(\mathbf{e}_t^{\phi,s},\gamma_t) - \mathbf{e}_\psi(\mathbf{e}_t^{\phi,s},\gamma_t)\Big)\Big]. \tag{37}$$

The Jacobian–vector product above contracts over both the L' token positions of the block and the d_e embedding coordinates, so writing it as a sum of per-token inner products shows that this gradient is obtained by differentiating the scalar surrogate

$$\mathcal{L}_{\text{DMD-grad}}^\phi = \mathbb{E}_{\gamma_s,\gamma_t,\mathbf{e}_s,\boldsymbol{\epsilon}}\Big[\mathrm{e}^{-\gamma_t}\sum_{l=1}^{L'}\Big\langle \mathbf{e}_0^{\phi,s,l}, \mathrm{sg}\left[\mathbf{e}_\eta^l(\mathbf{e}_t^{\phi,s},\gamma_t) - \mathbf{e}_\psi^l(\mathbf{e}_t^{\phi,s},\gamma_t)\right]\Big\rangle\Big], \tag{38}$$

which is (19) and concludes the proof. □

Algorithm 3 On-policy distribution-matching distillation

Require: Teacher θ, teacher-generated data p_{teacher}, M NFE count

Initialize independent parameter copies $\phi, \eta, \psi \leftarrow \theta$

$\Delta\gamma \leftarrow (\gamma_1 - \gamma_0)/M$; $\bar{\gamma}_j \leftarrow \gamma_1 - j\Delta\gamma$ for $j = 0, \dots, M$ ▷ Uniform logNSR discretization

Reference pre-fitting

for $n = 1, \dots, N_{\text{ref-fitting}}$ **do**

  $\gamma_u \sim \text{Unif}[\gamma_0, \gamma_1]$; $\mathbf{e}_u \sim q_u(\mathbf{e} \mid \gamma_u)$

  $\mathcal{L}^{\psi}_{\text{ref}} \leftarrow \text{CE}\big(\mathbf{x}, \mathbf{x}_\psi(\mathbf{e}_u, \gamma_u)\big)$ ▷ Reference loss equation 21

  $\psi \leftarrow \text{AdamW}(\mathcal{L}^{\psi}_{\text{ref}})$ ▷ Reference update

end for

while not converged **do**

  $\mathbf{x} \sim p_{\text{teacher}}$; $\mathbf{e} \leftarrow \mathbf{x}\mathbf{E}$ ▷ Sample target data

Student rollout

  $M_s \sim \text{Unif}\{0, \dots, M-1\}$ ▷ Sample rollout depth

  $\mathbf{e}_s \leftarrow \sigma(\gamma_1)\boldsymbol{\epsilon}$, $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$; $\mathbf{e}_{\text{SC}} \leftarrow \varnothing$ ▷ Initialize rollout states

  for $j = 0, \dots, M_s - 1$ **do**

    $\widehat{\mathbf{e}}_0 \leftarrow \mathbf{e}_\phi(\mathbf{e}_s, \bar{\gamma}_j; \mathbf{e}_{\text{SC}})$ ▷ Rollout transition with carried self-conditioning

    $\mathbf{e}_s \leftarrow \alpha(\bar{\gamma}_{j+1})\widehat{\mathbf{e}}_0 + \sigma(\bar{\gamma}_{j+1})\boldsymbol{\epsilon}$, $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ ▷ Consistency-style transition

    $\mathbf{e}_{\text{SC}} \leftarrow \widehat{\mathbf{e}}_0$ ▷ Update self-conditioning

  end for

  $\gamma_s \leftarrow \bar{\gamma}_{M_s}$; $\gamma_r \leftarrow \bar{\gamma}_{M_s+1}$ ▷ Segment start and end points

  $\mathbf{e}_0^{\phi,s} \leftarrow \mathbf{e}_\phi(\text{sg}[\mathbf{e}_s], \gamma_s)$ ▷ Student prediction

Auxiliary updates

  for $k = 1, \dots, 4$ **do**

    $\gamma_t \sim \text{Unif}[\gamma_r, \gamma_s]$; $\mathbf{e}_t^{\phi,s} \sim q_t(\text{sg}[\mathbf{e}_0^{\phi,s}] \mid \gamma_t)$ ▷ Re-noise student sample to query point

    $\mathcal{L}^{\eta}_{\text{aux}} \leftarrow h_t \,\text{CE}\big(\text{sg}[\mathbf{x}^{\phi,s}], \mathbf{e}_\eta(\mathbf{e}_t^{\phi,s}, \gamma_t)\big)$ ▷ Auxiliary loss equation 20

    $\eta \leftarrow \text{AdamW}(\mathcal{L}^{\eta}_{\text{aux}})$ ▷ Auxiliary update

  end for

Student update

  $\gamma_t \sim \text{Unif}[\gamma_r, \gamma_s]$; $\mathbf{e}_t^{\phi,s} \sim q_t(\text{sg}[\mathbf{e}_0^{\phi,s}] \mid \gamma_t)$ ▷ Re-noise student sample to query point

  $\widehat{\mathbf{e}}_\eta \leftarrow \mathbf{e}_\eta(\mathbf{e}_t^{\phi,s}, \gamma_t)$; $\widehat{\mathbf{e}}_\psi \leftarrow \mathbf{e}_\psi(\mathbf{e}_t^{\phi,s}, \gamma_t)$ ▷ Auxiliary and reference denoising

  $\mathcal{L}^{\phi}_{\text{DMD2-grad}} \leftarrow \frac{1}{R}\sum_{l=1}^{L'} \big\langle \mathbf{e}_0^{\phi,s,l}, \text{sg}\left[\widehat{\mathbf{e}}_\eta - \widehat{\mathbf{e}}_\psi\right] \big\rangle$ ▷ Student loss equation 22

  $\phi \leftarrow \text{AdamW}(\mathcal{L}^{\phi}_{\text{DMD2-grad}})$ ▷ Student update

end while

return ϕ

G. Implementation Details and Additional Results for Supervised Fine-Tuning

[Back to Sec. 7 in the main paper.]

G.1. Supervised Fine-Tuning Setup

Our SFT pipeline uses NeMo-RL with joint diffusion and AR supervision on assistant responses, following Nemotron-Labs-Diffusion (NLD) (Fu et al., 2026). Prompts remain clean, and training and evaluation share the non-thinking chat boundary described below. We freeze the diffusion embedding $\mathbf{E}$, noise-schedule network $\tilde{\gamma}_\psi(t)$, and endpoints γ_0, γ_1. The clean-stream input embedding, transformer, conditioning projections, and both readout heads remain trainable. Therefore, SUSReg is used only during pretraining.

Initialization and objective. We initialize from the 150k-step pretrained *Sigma-unreg*-3B/8B, *Sigma-preg*-3B/8B, and *Sigma*-3B/8B checkpoints. Before SFT of *Sigma-preg*-8B and *Sigma*-8B, we reinitialize the AR embedding rows with ℓ_2 norm below 10^{-6} to the mean direction of the remaining rows, scaled to their median norm. This includes chat-boundary IDs 10 and 11 and prevents the initial gradient-norm transient caused by their near-zero embeddings. The repair is applied once; its benchmark effect is not separately ablated. Let $\mathcal{A}$ denote the supervised answer positions, including the terminal EOS padding described below. The SFT objective is

$$\mathcal{L}_{\text{sft}} = \alpha \frac{\sum_{i\in\mathcal{A}} \mathcal{L}_{\text{BD},i}}{|\mathcal{A}|} + \frac{\sum_{i\in\mathcal{A}} \mathcal{L}_{\text{ar},i}}{|\mathcal{A}|}, \tag{39}$$

where $\mathcal{L}_{\text{BD}}$ is the continuous block NELBO of (3) and (4), including its diffusion, reconstruction, and prior terms, and $\mathcal{L}_{\text{ar}}$ is next-token cross-entropy from the clean stream (6). The ELBO weight is $\alpha = 0.3$ for *Sigma-unreg*-3B/8B and *Sigma-preg*-3B/8B, and $\alpha = 1$ for *Sigma*-3B/8B, following pretraining (App. C). Numerators and token counts are reduced globally across data-parallel ranks. The AR and diffusion denominators coincide under the answer mask because position zero is unsupervised, so α has the same meaning as the pretraining diffusion-loss weight.

As in pretraining (Sec. 3.2), we estimate the reconstruction and diffusion terms with the Horvitz–Thompson estimator in (5): independently for each block, we select reconstruction at $t = 0$ with probability $p_{\text{rec}} = 0.1$ and otherwise draw $t \sim \text{Unif}[0, 1]$. Gaussian noise is applied only in embedding space; the token IDs always remain clean. Prompt and ordinary batch-padding positions have zero loss and are pinned to their clean embeddings, whereas the synthetic EOS padding is supervised. Answer blocks see the clean context $\mathbf{c} = (\mathbf{p}, \mathbf{r}^{(<b)})$ of Sec. 4.1.

Data and preprocessing. We use the same non-thinking, single-turn SFT corpus as NLD (Chandiramani et al., 2026): 848,087 training and 1,200 validation conversations, each with an empty system message, a user prompt, and an assistant response. We use the same Mistral-NeMo (Mistral AI, 2024) tokenizer ($V = 131{,}072$) and the Nemotron Nano chat template (Blakeman et al., 2025), explicitly setting `enable_thinking=false`. Thus, the supervised response follows the intended prompt suffix `<think></think>`; training and evaluation use the same answer boundary. During training, we left-pad prompts with at most 31 unsupervised tokens so answers begin on the 32-token block grid; no padding is inserted between prompt and answer. Inference uses unpadded, fully causal prompts. Supervised EOS tokens complete the final answer block, with one full EOS block appended when already aligned. Unsupervised right-padding then rounds the batch length to a multiple of 1024 to limit FlexAttention compilation shapes. Samples exceeding 8192 tokens after padding are dropped.

Optimization. Both model sizes train for 8000 steps (≈ 2.4 epochs) at sequence length and maximum position 8192, block length 32, and global/micro batch sizes 256/1. We use AdamW with $(\beta_1, \beta_2) = (0.9, 0.98)$, $\epsilon = 10^{-5}$, weight decay 0.1, and learning rate 2.5×10^{-6} after 50 steps of linear warmup, held constant thereafter. Training uses `BF16`, gradient clipping at 1.0, and FSDP2 with TP = CP = 1, activation checkpointing, and no sequence packing. Self-conditioning is enabled for 25% of microbatches on every optimizer step and all validation sequences. The chat template enables the generation prompt; clean-prompt, answer-only supervision and EOS padding follow the preprocessing above.

Setting the maximum position to 8192 also extends the pretrained rotary frequencies: the effective YaRN factor is $8192/16384 = 0.5$, compared with 0.25 at the pretraining length of 4096.

G.2. Evaluation Setup, CFG–ST Sweep, and Full Results

The instruct evaluations in Tabs. 4 and 5 use the base-model harness (App. E.3), with the differences below.

Benchmarks. We evaluate GSM8K (1319 problems), MATH-500 (500), AIME 2024/2025 (30 each), HumanEval+ (164), MBPP (378), LiveCodeBench-CPP (454), MMLU (14042), GPQA (198), and IFEval (541), using the SFT chat template and diffusion block length $L' = 32$.

Scoring. Math tasks use symbolic equivalence on the final `\boxed{}` answer; GSM8K also accepts the last number. HumanEval and MBPP report executed pass@1 on EvalPlus's base tests (`passing_base_tests`), and LiveCodeBench-CPP compiles and tests extracted C++ programs. GPQA and MMLU request reasoning ending in `Answer: \boxed{A/B/C/D}`; grading extracts the last boxed letter, with a regex fallback. IFEval reports instruction-level loose accuracy, matching the reference models in Tabs. 4 and 5.

Baselines. Baseline scores come from Fu et al. (2026, Tabs. 5, 7), except NLD-8B diffusion, which we reproduce under the aligned protocol without format injection. We use SDAR-8B Chat at TPF = 1.00 rather than 1.75: it scores higher on eight tasks, with gaps ≤ 0.6 points on the remaining two (MMLU and HumanEval).

Decoding algorithms. As in the evaluation for base models, we always enable SC; DDPM-256 uses EDM $\rho = 7$ schedule, and DDIM-32 uses uniform-γ schedule. Their respective (w, τ) settings follow the recommended values for base models Tab. 10.

Generation budget. The generation budget is 1024 tokens for GSM8K and HumanEval at 3B, rising to 1536 and 2560, respectively, at 8B; 2560 for MBPP, LiveCodeBench-CPP, and IFEval; and 4096 for AIME 2024/2025, MATH-500, and GPQA. MMLU uses 2048 for 3B DDPM and 2560 otherwise. AR uses 16384 except on MMLU (2560). Caps follow per-lane graded-answer lengths: no correct answer in the smoke runs required more than 4096 tokens, and the longer tail consists mainly of degenerate repetition.

Seed error bars. Each diffusion row averages 10 independent sampling lanes and reports half-width $\frac{2}{\sqrt{10}}\hat{\sigma}$, where $\hat{\sigma}$ is the standard deviation of per-lane benchmark scores. AR greedily decodes the clean stream through its AR head.

Post-SFT CFG–ST sweep. On GSM8K and HumanEval, the pre-SFT guidance and temperature optima remain within sampling uncertainty of the best post-SFT settings for both 3B checkpoints and samplers (Tab. 20). The 8B SUSReg checkpoint uses the corresponding 3B settings without a separate sweep.

Full instruct results for all three variants. Tab. 21 reports the ten instruct benchmarks for all three variants at 3B and 8B.

Table 20 | **Post-SFT CFG–ST sweep and transfer of pre-SFT optima.** Entries are the unweighted GSM8K–HumanEval average (%), reported as mean ± two standard errors over 10 seeds. DDPM-256 uses EDM $\rho = 7$; DDIM-32 uses uniform-γ. † marks the corresponding pre-SFT optimum; **bold** marks the maximum in each model–sampler grid. The pre-SFT settings remain within sampling uncertainty of the post-SFT maxima.

Model and sampler	w	$\tau = 0.1$	$\tau = 0.2$	$\tau = 0.3$	$\tau = 0.4$	$\tau = 0.5$
Sigma-unreg-3B, DDPM-256	1	81.75 (± 0.56)	—	—	—	—
Sigma-unreg-3B, DDPM-256	2	**83.32**† (± 0.52)	—	—	—	—
Sigma-unreg-3B, DDPM-256	3	80.63 (± 0.42)	—	—	—	—
Sigma-unreg-3B, DDIM-32	2	—	80.93 (± 0.44)	79.83 (± 0.76)	78.84 (± 0.60)	78.65 (± 0.85)
Sigma-unreg-3B, DDIM-32	3	—	60.89 (± 0.40)	80.85† (± 0.60)	81.17 (± 0.57)	81.32 (± 0.70)
Sigma-unreg-3B, DDIM-32	4	—	38.88 (± 0.59)	77.70 (± 0.99)	**81.68** (± 0.62)	81.21 (± 0.84)
Sigma-unreg-3B, DDIM-32	5	—	21.21 (± 0.26)	69.86 (± 0.81)	80.94 (± 0.72)	81.05 (± 0.78)
Sigma-preg-3B, DDPM-256	1	—	80.10 (± 0.56)	—	—	—
Sigma-preg-3B, DDPM-256	2	—	80.75 (± 0.50)	—	—	—
Sigma-preg-3B, DDPM-256	3	—	**80.80**† (± 0.57)	—	—	—
Sigma-preg-3B, DDPM-256	4	—	79.75 (± 0.69)	—	—	—
Sigma-preg-3B, DDIM-32	2	—	**79.17** (± 0.55)	78.76 (± 0.90)	78.03 (± 0.56)	77.63 (± 0.84)
Sigma-preg-3B, DDIM-32	3	—	73.81 (± 0.66)	78.29 (± 0.56)	78.70 (± 1.08)	78.59 (± 0.42)
Sigma-preg-3B, DDIM-32	4	—	54.50 (± 0.65)	77.93 (± 0.65)	78.65† (± 0.55)	78.32 (± 0.60)
Sigma-preg-3B, DDIM-32	5	—	43.24 (± 0.32)	74.94 (± 0.81)	79.09 (± 0.89)	78.97 (± 0.51)

Table 21 | **Pass@1 accuracy of all three *Sigma-instruct* variants at 3B and 8B.** Same ten tasks, harness, and operating points as Tabs. 4 and 5. *Sigma* diffusion rows are mean (over 10 sampling lanes) ± two standard errors. **Bold** marks the best value per task within each mode and model size.

	Math				**Coding**			**Science & Knowledge**		**IF**	
Method	GSM8K	Math500	AIME24	AIME25	HumanEval	MBPP	LCB-CPP	MMLU	GPQA	IFEval	Avg.
3B models											
AR mode											
Nemotron-Labs-Diffusion-3B-Instruct	87.87	**77.60**	**23.33**	**16.67**	76.22	71.16	**21.37**	71.70	39.39	69.39	**55.47**
Sigma-unreg-3B-*instruct*	**88.17**	71.60	20.00	10.00	71.34	77.51	13.00	71.36	38.38	**73.62**	53.50
Sigma-preg-3B-*instruct*	86.81	74.60	10.00	13.33	**79.88**	77.78	15.86	**72.87**	**39.90**	**73.62**	54.46
Sigma-3B-*instruct*	85.29	71.60	3.33	3.33	73.17	**78.04**	14.76	69.98	34.34	70.74	50.46
Diffusion mode											
Nemotron-Labs-Diffusion-3B-Instruct	88.40	**74.80**	**16.67**	10.00	74.39	73.54	**16.30**	72.06	33.84	68.93	52.89
Sigma-unreg-3B-*instruct*											
DDPM (256 steps, w=2, τ=0.1)	89.32 ± 0.24	73.20 ± 0.45	15.67 ± 2.84	11.00 ± 1.90	**77.32** ± 1.05	71.75 ± 0.72	12.97 ± 0.62	**72.58** ± 0.08	**37.68** ± 1.70	70.11 ± 0.64	53.16
DDIM (32 steps, w=3, τ=0.3)	88.10 ± 0.24	71.42 ± 0.88	14.00 ± 2.46	8.67 ± 1.93	73.60 ± 1.17	68.76 ± 0.69	11.56 ± 0.61	71.37 ± 0.08	34.24 ± 1.42	68.18 ± 0.32	50.99
Sigma-preg-3B-*instruct*											
DDPM (256 steps, w=3, τ=0.2)	89.46 ± 0.16	72.96 ± 0.99	13.67 ± 1.48	8.67 ± 1.40	72.13 ± 1.09	71.80 ± 0.52	12.58 ± 0.43	72.20 ± 0.07	37.17 ± 1.49	72.49 ± 0.45	52.31
DDIM (32 steps, w=4, τ=0.4)	88.95 ± 0.19	70.88 ± 0.47	13.00 ± 1.48	**14.33** ± 2.32	68.35 ± 1.10	68.54 ± 0.63	11.30 ± 0.37	71.14 ± 0.09	36.31 ± 1.68	70.96 ± 0.77	51.38
Sigma-3B-*instruct*											
DDPM (256 steps, w=3, τ=0.1)	**89.64** ± 0.10	73.08 ± 0.62	13.67 ± 1.99	11.33 ± 1.03	77.01 ± 1.24	**74.97** ± 0.59	13.50 ± 0.42	70.64 ± 0.12	36.01 ± 1.46	**73.26** ± 0.59	**53.31**
DDIM (32 steps, w=3, τ=0.3)	88.82 ± 0.15	72.10 ± 0.74	15.33 ± 2.70	13.33 ± 2.11	73.96 ± 0.91	73.31 ± 0.82	12.16 ± 0.56	70.20 ± 0.07	34.80 ± 1.99	69.00 ± 0.49	52.30
8B models											
AR mode											
Nemotron-Labs-Diffusion-8B-Instruct	**94.01**	**88.00**	**33.33**	**33.33**	80.49	**85.19**	**28.85**	**79.85**	44.44	68.65	**63.61**
Sigma-unreg-8B-*instruct*	91.74	80.80	16.67	20.00	78.05	84.92	21.59	78.22	**46.97**	79.50	59.85
Sigma-preg-8B-*instruct*	92.57	78.60	16.67	6.67	79.88	83.07	15.20	78.32	44.44	**80.70**	57.61
Sigma-8B-*instruct*	91.58	79.40	26.67	16.67	**81.10**	80.69	12.33	76.27	43.94	78.06	58.67
Diffusion mode											
Nemotron-Labs-Diffusion-8B-Instruct	91.58	**84.60**	**30.00**	**23.33**	**83.54**	82.01	**21.15**	76.65	33.33	73.02	**59.92**
Sigma-unreg-8B-*instruct*											
DDPM (256 steps, w=2, τ=0.1)	**93.24** ± 0.20	79.80 ± 0.55	23.67 ± 3.19	18.00 ± 2.15	77.99 ± 1.06	80.40 ± 0.80	20.79 ± 0.52	**77.91** ± 0.11	42.37 ± 1.80	79.83 ± 0.42	59.40
DDIM (32 steps, w=3, τ=0.3)	92.90 ± 0.18	78.20 ± 0.33	19.00 ± 0.97	19.00 ± 2.50	78.35 ± 1.62	79.74 ± 0.98	18.33 ± 0.61	77.60 ± 0.09	41.87 ± 1.74	76.69 ± 0.38	58.17
Sigma-preg-8B-*instruct*											
DDPM (256 steps, w=3, τ=0.2)	92.81 ± 0.30	80.78 ± 0.69	20.33 ± 2.74	18.00 ± 2.15	80.30 ± 0.93	81.72 ± 0.87	19.93 ± 0.34	77.31 ± 0.08	**44.80** ± 1.55	**80.82** ± 0.28	59.68
DDIM (32 steps, w=4, τ=0.4)	92.37 ± 0.15	80.52 ± 0.69	22.33 ± 3.27	17.67 ± 3.41	80.91 ± 1.61	80.24 ± 0.59	18.00 ± 0.66	76.12 ± 0.07	42.58 ± 1.18	76.74 ± 0.64	58.75
Sigma-8B-*instruct*											
DDPM (256 steps, w=3, τ=0.1)	91.96 ± 0.21	78.40 ± 0.67	23.33 ± 2.67	17.00 ± 1.75	82.50 ± 1.17	**82.04** ± 0.67	20.09 ± 0.63	76.57 ± 0.08	42.37 ± 1.17	77.18 ± 0.38	59.14
DDIM (32 steps, w=3, τ=0.3)	91.90 ± 0.17	76.70 ± 0.67	21.67 ± 3.02	14.33 ± 3.53	80.55 ± 0.58	79.47 ± 0.55	19.38 ± 0.48	76.15 ± 0.08	42.93 ± 1.75	75.96 ± 0.32	57.90

H. Implementation Details and Additional Results for Self-Speculative Decoding

This section gives the protocol (App. H.1) and all results: the models of Sec. 6 in full (App. H.2), the undistilled *Sigma* with trained drafters (App. H.3), and acceptance training (App. H.4).

H.1. Setup

Drafting and verification. Following the two-pass framework of NLD (Fu et al., 2026, Sec. 3.3), each round initializes one block of $L' = 32$ draft positions with Gaussian noise in the 16-dimensional latent space and reads out categorical draft tokens from a single denoiser call at $t = 1$. The AR head then scores all drafts causally in one forward pass over the shared backbone and cached prefix, keeps the longest prefix that matches its own greedy tokens, and appends one token of its own. Draft-only parameters are disabled during verification, so the committed text is the model's greedy AR output, up to `BF16` near-ties between the decoding and verification kernels. More denoiser calls per draft raised AL but not tokens per forward pass, so we use one.

Trained drafters. Optionally, we train draft-only parameters on Tulu-3 prompts (Lambert et al., 2024) with NLD's LK-hybrid and verifier cross-entropy objective (Fu et al., 2026, Eqs. 6–9), keeping the backbone and the AR head frozen. We train a rank-128 LoRA on the query and value projections together with the diffusion head (Q/V, 440M parameters at 3B and 595M at 8B). At 8B, the 6 of 1,000 training steps whose gradient norm is not finite make no update.

Benchmarks. GSM8K (1,319 problems, 8-shot chain of thought), HumanEval (164), and MBPP (500) use the prompts of our base-model evaluation. The end-of-text token is ignored, and an answer ends at the first of the task's stop strings, within 256, 512, and 512 new tokens.

Serving. We serve with SGLang (Zheng et al., 2024) in `BF16`, with greedy decoding, TP1, and one request at a time on one NVIDIA GB200. AL and accuracy use the numerics of the training code (FP32 diffusion-stream parameters and eager kernels). Throughput uses FlashAttention-4 and CUDA graphs.

NLD reference. NLD-3B-Base and NLD-8B-Base ship without a drafter adapter, so they draft with their own diffusion head. Every NLD model runs at its released block size: 32 positions holding the last committed token and 31 drafts, one draft fewer per round than *Sigma*. NLD's speculative server searched only the end of each committed block for stop strings and missed earlier ones; we let it search the whole block, as SGLang's own speculative path does, so that it stops in the block that contains the first stop. NLD's runner times AR and speculative decoding once each, concurrently on two GPUs of one node.

Metrics. AL counts the draft tokens that a round adds to the returned answer, i.e., its committed answer tokens minus the verifier's own token, pooled over rounds; drafts accepted past the end of the answer are discarded and not counted. Repetition loops are trivially predictable and would inflate AL. On GSM8K, HumanEval, and MBPP, we therefore drop, per model, every prompt whose AR, speculative, or timing output loops (at least half of its 16-grams repeat) or reaches the length cap; accuracy uses all prompts. Throughput is answer tokens per second of client time, and speedup divides the speculative throughput by the same model's AR throughput on the same GPU and prompts, averaged over two runs on different GPUs. Paired gains use the prompts that both models keep, with 95% percentile bootstrap intervals over prompts.

H.2. Main Results in Full

Tab. 22 lists throughput, speedup, and AL for the models of Sec. 6, and Tab. 23 lists their accuracy and kept prompts. For *Sigma-distill*, a draft–verify round costs 3.0–3.5 AR decoding steps rather than the ideal two. Speedups are relative to each model's own AR decoding, which NLD's serving implementation runs about 20% faster at 3B and 12% faster at 8B.

H.3. Undistilled Model and Trained Drafters

The undistilled *Sigma*-3B and *Sigma*-8B can draft with their own diffusion head or with a trained Q/V adapter (Tab. 24).

Distillation helps more than a trained drafter at 3B. The adapter raises *Sigma*-3B's AL by 13.1%, 5.4%, and 7.2% on GSM8K, HumanEval, and MBPP ([12.2, 14.0], [3.1, 7.6], and [5.8, 8.9]). The distilled

Table 22 | **Self-speculative decoding in full.** Tokens/s of AR and speculative (Spec) decoding, speedup, and AL.

		GSM8K				HumanEval				MBPP			
	Model	AR	Spec	Speedup	AL	AR	Spec	Speedup	AL	AR	Spec	Speedup	AL
3B	NLD-3B-Base	453	1,197	2.64×	7.63	471	906	1.92×	5.01	454	1,078	2.37×	7.52
	Sigma-distill-3B	375	1,168	3.12×	9.21	399	880	2.21×	5.90	374	1,011	2.70×	8.36
	Sigma-distill-spec-3B	379	1,204	3.18×	9.68	399	889	2.23×	6.25	375	1,027	2.74×	8.85
8B	NLD-8B-Base	247	865	3.50×	8.78	259	652	2.51×	6.08	241	750	3.11×	8.40
	Sigma-distill-8B	221	864	3.92×	10.75	229	543	2.37×	7.12	217	707	3.26×	9.56
	Sigma-distill-spec-8B	222	931	4.20×	11.68	228	620	2.72×	7.33	220	762	3.47×	10.33

Table 23 | **Accuracy (%) of AR / speculative decoding, and kept prompts.** GSM8K: flexible / strict extract; code: pass@1. Kept prompts are out of 1,319, 164, and 500.

	Model	GSM8K flexible	GSM8K strict	HumanEval	MBPP	Kept prompts
3B	NLD-3B-Base	83.8 / 84.0	75.4 / 77.0	52.4 / 51.8	61.6 / 61.8	1,306 / 151 / 497
	Sigma-distill-3B	82.9 / 82.9	79.5 / 79.0	56.1 / 56.1	58.2 / 59.2	1,296 / 157 / 498
	Sigma-distill-spec-3B	82.2 / 82.5	78.5 / 78.5	55.5 / 55.5	58.8 / 58.6	1,298 / 160 / 499
8B	NLD-8B-Base	88.8 / 88.1	87.3 / 87.0	61.6 / 62.2	66.2 / 66.8	1,307 / 149 / 497
	Sigma-distill-8B	86.4 / 86.8	85.2 / 85.7	56.1 / 57.9	64.0 / 63.8	1,311 / 158 / 497
	Sigma-distill-spec-8B	86.4 / 86.1	85.1 / 84.8	57.3 / 56.7	63.8 / 64.4	1,310 / 159 / 497

student, with its own head and no adapter, gains 21.8%, 20.1%, and 15.0% over *Sigma*-3B ([20.1, 23.5], [8.6, 31.7], and [11.8, 18.1]).

At 8B, the two are closer. The adapter raises *Sigma*-8B's AL by 16.1%, 9.4%, and 14.3% ([15.1, 17.1], [6.1, 13.0], and [12.4, 16.1]), and distillation by 22.4%, 7.9%, and 19.9% ([20.9, 24.0], [−0.5, 16.8], and [16.7, 23.2]). Compared directly, *Sigma-distill*-8B accepts 5.4% and 4.9% more than *Sigma*-8B with the adapter on GSM8K and MBPP ([4.0, 6.8] and [2.0, 7.7]) and 1.6% fewer on HumanEval ([−9.3, 6.4]), where it also decodes slower (2.37× against 2.73×).

The adapter leaves the output unchanged. It is disabled during verification, so *Sigma*-3B and *Sigma*-8B solve exactly the same problems with and without it.

H.4. Acceptance Training of the Distilled Model

Instead of training a separate drafter, we continue the distillation with an acceptance objective that updates the whole backbone; we call the result *Sigma-distill-spec*. Let ϕ denote the parameters of the student, which starts from *Sigma-distill*, and $\bar{\phi}$ those of a frozen copy of *Sigma-distill*. We keep the distillation objective and add two terms. An LK-hybrid loss aligns the student's one-call draft with the AR head of $\bar{\phi}$, and a KL term holds the student's AR distribution to that of $\bar{\phi}$, so that the verifier, and hence the decoded output, is preserved:

$$\mathcal{L}_{\text{acc}} = \lambda_{\text{DMD}}\,\mathcal{L}_{\text{DMD}} + \lambda_{\text{LK}}\,\mathcal{L}_{\text{LK}} + \hat{\lambda}_{\text{ret}}\,\mathcal{L}_{\text{ret}}. \tag{40}$$

Training updates the shared transformer and the diffusion head; as in distillation (App. F.2), the embedding matrix $\mathbf{E}$, the noise schedule, and the AR head stay frozen.

Distribution matching. $\mathcal{L}_{\text{DMD}}$ is the student objective of App. F (Alg. 3), with the auxiliary and reference denoisers and the setting of the distillation that produced *Sigma-distill*: two phases, no self-conditioning, and segments that start from fresh student rollouts. Acceptance training thus continues that distillation.

Acceptance. Each LK example takes a context $\mathbf{c}$ from the generations of an earlier 3B DMD student on math and code prompts, mostly GSM8K- and HumanEval-style, and drafts the next block with one call from pure noise, as in serving (App. H.1). Both sizes use these contexts; the two runs averaged into each *Sigma-distill-spec* model use two versions of the context set. Test and dev problems are excluded by exact match; a 13-gram scan finds 3, 1, and 54 near-duplicates of GSM8K, HumanEval, and MBPP test problems among the contexts, and excluding those problems leaves the AL gains of acceptance training unchanged. Slot

Table 24 | **Undistilled *Sigma*: own head, trained Q/V adapter, and distillation.** AL gain is over the undistilled model with its own head, on the prompts both keep.

		GSM8K	HumanEval	MBPP
3B	*Sigma*-3B: AL / speedup	7.55 / 2.54×	4.92 / 1.77×	7.24 / 2.38×
	+ Q/V adapter: AL gain / speedup	+13.1% / 2.93×	+5.4% / 1.94×	+7.2% / 2.46×
	Sigma-distill-3B: AL gain / speedup	+21.8% / 3.12×	+20.1% / 2.21×	+15.0% / 2.70×
	AR accuracy (%), *Sigma*-3B / *Sigma-distill*-3B	81.2 / 82.9	49.4 / 56.1	56.2 / 58.2
8B	*Sigma*-8B: AL / speedup	8.80 / 3.38×	6.60 / 2.58×	7.96 / 2.92×
	+ Q/V adapter: AL gain / speedup	+16.1% / 3.79×	+9.4% / 2.73×	+14.3% / 3.27×
	Sigma-distill-8B: AL gain / speedup	+22.4% / 3.92×	+7.9% / 2.37×	+19.9% / 3.26×
	AR accuracy (%), *Sigma*-8B / *Sigma-distill*-8B	84.8 / 86.4	56.1 / 56.1	64.2 / 64.0

$j \in \{1, \dots, L'\}$ has draft distribution $\hat{p}^j := \mathbf{x}^j_\phi(\mathbf{z}_1, 1; \mathbf{c})$ with $\mathbf{z}_1 \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ and draft token $\hat{v}^j := \operatorname{argmax}_v \hat{p}^j(v)$. The verifier reads all slots in one causal pass and gives $p^j := \mathbf{x}^{|\mathbf{c}|+j}_{\bar{\phi}_{\mathrm{AR}}}(\tilde{\mathbf{x}}^{<|\mathbf{c}|+j})$, where $\tilde{\mathbf{x}}$ is $\mathbf{c}$ followed by the draft tokens. If the first a draft tokens equal the verifier's greedy tokens $v^j_\star := \operatorname{argmax}_v p^j(v)$ and slot $a+1$ does not, the loss covers the accepted prefix and the first rejected slot, $\mathcal{A} = \{1, \dots, \min(a+1, L')\}$. Following NLD (Fu et al., 2026, Eqs. 6–9), let U^j be the union of the 200 most probable tokens under p^j and under $\hat{p}^j$; let p^j_τ and $\hat{p}^j_\tau$ be p^j and $\hat{p}^j$ restricted to U^j and renormalized at temperature $\tau_{\mathrm{LK}} = 3$, i.e., $p^j_\tau(v) \propto p^j(v)^{1/\tau_{\mathrm{LK}}}$ for $v \in U^j$; and let $\hat{p}^j_1$ be the same restriction of $\hat{p}^j$ at temperature 1. Then

$$\mathcal{L}_{\mathrm{LK}} = \mathbb{E}\left[\tfrac{1}{|\mathcal{A}|}\textstyle\sum_{j\in\mathcal{A}} \nu^j \ \mathrm{KL}(p^j_\tau \,\|\, \hat{p}^j_\tau) + (1-\nu^j)\,\mathrm{TV}(p^j_\tau, \hat{p}^j_\tau) - \log \hat{p}^j_1(v^j_\star)\right], \qquad \nu^j = \exp\big(-\eta\ \mathrm{sg}(\omega^j)\big), \tag{41}$$

where $\omega^j = \sum_{v\in U^j} \min\big(p^j_\tau(v), \hat{p}^j_\tau(v)\big) = 1 - \mathrm{TV}(p^j_\tau, \hat{p}^j_\tau)$ is the probability that a token sampled from $\hat{p}^j_\tau$ is accepted, and $\eta = 0.5$; the weight thus shifts from the KL term toward total variation as the draft approaches the verifier. The retention term is the forward KL from the AR distribution of $\bar{\phi}$ to that of the student, averaged over the set $\mathcal{R}$ of the last 128 positions of $\tilde{\mathbf{x}}$, i.e., the L' draft slots and the 96 context tokens before them:

$$\mathcal{L}_{\mathrm{ret}} = \mathbb{E}\left[\tfrac{1}{|\mathcal{R}|}\textstyle\sum_{l\in\mathcal{R}} \mathrm{KL}\left(\mathbf{x}^l_{\bar{\phi}_{\mathrm{AR}}}(\tilde{\mathbf{x}}^{<l}) \,\middle\|\, \mathbf{x}^l_{\phi_{\mathrm{AR}}}(\tilde{\mathbf{x}}^{<l})\right)\right]. \tag{42}$$

Weights and optimization. We set $\lambda_{\mathrm{LK}} = 0.25$ and $\lambda_{\mathrm{ret}} = 12$, and cap the retention weight at

$$\hat{\lambda}_{\mathrm{ret}} = \min\left(\lambda_{\mathrm{ret}},\ 3\,\frac{\|\nabla_\phi(\lambda_{\mathrm{DMD}}\mathcal{L}_{\mathrm{DMD}} + \lambda_{\mathrm{LK}}\mathcal{L}_{\mathrm{LK}})\|}{\|\nabla_\phi \mathcal{L}_{\mathrm{ret}}\|}\right), \tag{43}$$

with both gradient norms taken from the previous update. Without the cap, once the AR distribution drifted, the retention gradient grew until it dominated the clipped update, and training diverged. Each update draws 64 DMD examples and 128 drafts. The drafts form chains of four rounds: the first round drafts after a prefix of up to 1,664 tokens that keeps the prompt's BOS token, and each later round drafts after the tokens that the previous round committed, as in serving. The optimizers are those of App. F.2, except that the student learning rate is doubled to 7.2×10^{-7}; the drafts never use self-conditioning, as in serving. Each *Sigma-distill-spec* model is a uniform weight average of two runs with $\lambda_{\mathrm{DMD}} = 1$ and 0.5, each trained for 600 updates from *Sigma-distill* while the learning rate decays linearly to zero. At 8B, the teacher-generated DMD data has 16,384 contexts instead of 65,536, and the fit of the query noise levels and the LK contexts are reused from 3B.

Results. Tab. 25 gives the gains. Acceptance training raises AL by 5.2%, 6.0%, and 6.1% at 3B and by 8.5%, 4.8%, and 8.1% at 8B on GSM8K, HumanEval, and MBPP; the HumanEval intervals include zero at both sizes. It leaves AR accuracy unchanged on all three benchmarks at both sizes (sign tests, $p \geq 0.10$ at 3B and $p \geq 0.69$ at 8B).

Table 25 | **Acceptance training: gain of *Sigma-distill-spec* over *Sigma-distill*.** AL over the prompts both models keep, with 95% bootstrap intervals; speedup as in Tab. 22.

		GSM8K	**HumanEval**	**MBPP**
3B	AL, *Sigma-distill*-3B → *Sigma-distill-spec*-3B	9.21 → 9.68	5.92 → 6.28	8.36 → 8.87
	AL gain	+5.2% [4.0, 6.2]	+6.0% [−0.2, 12.1]	+6.1% [4.1, 8.2]
	Speedup	3.12× → 3.18×	2.21× → 2.23×	2.70× → 2.74×
8B	AL, *Sigma-distill*-8B → *Sigma-distill-spec*-8B	10.76 → 11.68	7.31 → 7.66	9.56 → 10.33
	AL gain	+8.5% [7.3, 9.8]	+4.8% [−0.1, 10.4]	+8.1% [5.8, 10.4]
	Speedup	3.92× → 4.20×	2.37× → 2.72×	3.26× → 3.47×