Unlocking Continuous Diffusion Language Models via Discrete Stochastic Localization

Yunshu Wu[1] **Jiayi Cheng**[2] **Longxuan Yu**[1] **Partha Thakuria**[1]
Rob Brekelmans **Evangelos E. Papalexakis**[1] **Greg Ver Steeg**[1]
[1]University of California Riverside [2]New York University
{ywu380, ylong030, pthakuria, epapalex, greg.versteeg}@ucr.edu
jiayi.cheng@nyu.edu rob.brekelmans@gmail.com

Abstract

Continuous diffusion is a natural framework for non-autoregressive generation but has generally lagged behind masked discrete diffusion models (MDMs) on discrete sequence generation. We argue that the bottleneck is not continuity itself, but a representation in which denoising depends on timestep-indexed noise regimes. We introduce *Discrete Stochastic Localization* (DSL), a continuous-state framework with unit-sphere token embeddings whose Bayes-optimal denoiser is invariant to the nominal signal-to-noise ratio (SNR) under the localization channel. One trained network then supports an entire family of per-token SNR paths, with endpoint masked-diffusion paths as a special case. Fine-tuning a pretrained MDLM checkpoint with DSL substantially improves distributional faithfulness (MAUVE) on OpenWebText across all step budgets from $T{=}128$ to $T{=}1024$, and the same checkpoint supports random-order autoregressive sampling, as well as a hybrid continuous-then-discrete sampler using as few as T=48 total steps—without distillation or retraining.

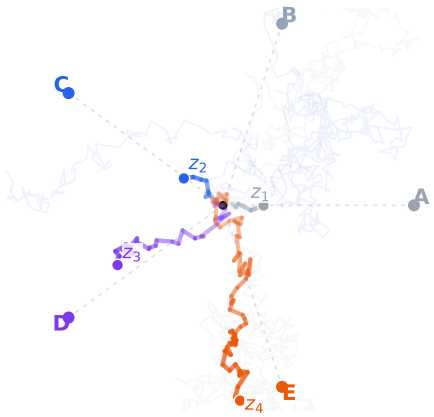
1 Introduction

Continuous diffusion has become a standard framework for high-dimensional generation, with strong results in image, video, and audio synthesis [10, 35, 11, 14, 18]. For language, however, continuous diffusion over token embeddings has not delivered the same advantage: existing continuous diffusion language models have consistently lagged behind masked discrete diffusion models (MDMs) [2, 28, 33]. This gap is puzzling. Continuous diffusion exposes the model to a continuum of finite-SNR states between the fully uninformative and nearly clean limits, which should in principle provide richer supervision than endpoint-like masked corruptions. Why, then, has this extra continuity not translated into stronger language generation?

We argue that the missing ingredient is not continuity alone, but also representation choice. Standard embedding-space diffusion typically adopts Variance-Preserving (VP) or Variance-Exploding (VE) Gaussian noising. These parameterizations are probabilistically valid, but they represent the zero-information limit as isotropic Gaussian noise rather than as a mask-like token state. Moreover, the raw noisy embedding is generally not sufficient for clean-token prediction: the denoiser must also receive a timestep or noise-level label to calibrate how much token evidence the embedding contains. As a result, denoisers in this representation are highly sensitive to time conditioning to distinguish between masked versus uncertain states.

This suggests a simple design goal: parameterize the corruption so that the noisy token state alone determines the clean-token posterior. We introduce *Discrete Stochastic Localization* (DSL), a continuous diffusion language model based on stochastic localization over unit-norm token embeddings. In DSL, the zero-SNR state is the zero vector, which can be identified with a `[MASK]` token, while the high-SNR limit recovers the clean token embedding. More importantly, we show that unit-norm

40th Conference on Neural Information Processing Systems (NeurIPS 2026).



(a) DSL dynamics: each $\boldsymbol{z}_i(t)$ localizes toward a symbol anchor over time.

	Token 1	Token 2	Token 3	Token 4	Token 5
State	z_1	z_5	z_2	z_3	z_4
SNR	0.0	1.0	1.4	9.3	24.3
Decoded	B (mask)	A (soft)	C (soft)	D (soft)	E (hard)

(b) State summary: tokens span SNR 1.0–24.3, from masked to hard decoding.

Figure 1: Discrete Stochastic Localization (DSL) dynamically "localizes" to sample discrete tokens. *(Left)* DSL dynamics on a cyclic toy sequence (ABCDE, BCDEA, . . .); details in Appendix D. *(Right)* State summary at one snapshot: each token occupies a different SNR regime, transitioning from masked to soft to hard decoding as SNR increases.

stochastic-localization parameterization makes the Bayes-optimal clean-token posterior depend only on the noisy token state, not on the nominal SNR. DSL therefore uses a time-invariant denoiser: masked, uncertain finite-SNR, and near-clean token states are all denoised to clean-token posteriors without explicit time conditioning.

This time-invariant denoiser gives DSL a simple inference-level consequence. Because the model is not tied to a global diffusion time, different positions can occupy different SNR levels and follow different update schedules. Masked refinement, random-order autoregressive decoding, and hybrid continuous-then-discrete sampling are therefore shown to be different ways of querying the same posterior estimator, rather than separate sampler-specific models. Thus DSL keeps the endpoint behavior and checkpoint compatibility of masked diffusion while adding continuous finite-SNR states for richer refinement.

We instantiate DSL by fine-tuning from a pretrained MDLM checkpoint and evaluate multiple inference paths through the same model on OpenWebText. On a masked-refinement path with a ReMDM-family decoder, DSL substantially improves few-step MAUVE across sampling budgets from $T = 128$ to $T = 1024$. The same checkpoint also supports random-order autoregressive sampling without retraining and a hybrid continuous-then-discrete sampler that produces coherent generations in as few as $T = 48$ steps. Together, these results support the central claim: endpoint revealing, masked refinement, and continuous/hybrid denoising can be served by one posterior estimator rather than by separate sampler-specific models.

Contributions.

- **A continuous-state framework for discrete generation.** We introduce DSL, a stochastic-localization channel over unit-sphere token embeddings whose Bayes-optimal denoiser is, by construction, a function of the noisy state alone rather than the nominal SNR. A single time-agnostic network can therefore support an entire family of per-token SNR paths.

- **A practical recipe.** Mixed-support SNR sampling trains the same denoiser on endpoint and finite-SNR states, while a mixture-of-tokens converter turns the framework into a trainable objective over a standard Transformer/DiT backbone.

- **Evidence across paths.** A single DSL checkpoint improves few-step OpenWebText generation under masked refinement, supports random-order AR and hybrid sampling without retraining, and is competitive on Text8 likelihood.

2 Discrete Stochastic Localization

This section develops DSL in three steps. We first set up the simplest version of the framework—a single Gaussian observation channel applied to the whole sentence at one global signal-to-noise ratio (§2.1). We then introduce DSL's key design choice, placing clean token embeddings on the unit sphere, and show that under this geometry the Bayes-optimal denoiser depends only on the noisy token and not on the nominal SNR (§2.2). Because a single denoiser then suffices across all SNR regimes, we can let each token position evolve under its own SNR; this generalizes the channel to a continuous family of per-token SNR paths (§2.3). Finally, two exact NLL views justify the state support used by the training objective (§2.4).

2.1 A sentence-level continuous starting point

We start from the simplest version of DSL: a Gaussian observation channel applied to the whole sentence at a single global signal-to-noise ratio. This setup is the natural continuous analogue of standard diffusion and lets us derive the MMSE drift form that the rest of the section will build on.

Let $\boldsymbol{s} = (s_1, \ldots, s_L) \in \mathcal{V}^L$ be a sentence of length L, and let each token be embedded as $\boldsymbol{x}_i = \mathrm{enc}(s_i) \in \mathbb{R}^d$. We write the full sentence embedding as $\boldsymbol{x} = [\boldsymbol{x}_1; \ldots; \boldsymbol{x}_L] \in \mathbb{R}^{Ld}$. The sentence-level localization channel is

$$\boldsymbol{z}_t = t\,\boldsymbol{x} + \sqrt{t}\,\boldsymbol{\epsilon}, \qquad \boldsymbol{x} \sim P(\boldsymbol{x}),\ \boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}).$$

Dividing by t gives $\boldsymbol{z}_t/t = \boldsymbol{x} + \boldsymbol{\epsilon}/\sqrt{t}$, so $\boldsymbol{z}_t$ is equivalent in information to observing $\boldsymbol{x}$ through additive Gaussian noise of variance $1/t$. The signal-to-noise ratio is therefore exactly t: at $t = 0$ the observation carries no information, and as t grows $\boldsymbol{z}_t$ becomes progressively more informative about $\boldsymbol{x}$. This channel admits a simple SDE realization,

$$d\boldsymbol{z}_t = \boldsymbol{x}\,dt + d\mathbf{w}, \qquad \boldsymbol{x} \sim P(\boldsymbol{x}), \tag{1}$$

which conditions on the unknown clean sentence and so cannot be used directly for generation. We seek an equivalent unconditional SDE—equivalent in the sense that both dynamics induce the same marginal on $\boldsymbol{z}_t$ —whose drift depends only on $\boldsymbol{z}_t$:

$$d\boldsymbol{z}_t = \hat{\boldsymbol{x}}(\boldsymbol{z}_t, t)\,dt + d\mathbf{w}, \qquad \hat{\boldsymbol{x}}(\boldsymbol{z}_t, t) := \mathbb{E}_{P_t(\boldsymbol{x}|\boldsymbol{z}_t)}[\boldsymbol{x}]. \tag{2}$$

The drift that achieves this equivalence is the MMSE denoiser, the posterior mean of $\boldsymbol{x}$ given $\boldsymbol{z}_t$ (proof in Appendix A.4). At this point DSL still resembles a standard continuous-denoising construction: one global SNR, one denoiser that nominally depends on t.

2.2 Unit-norm token geometry gives a time-invariant denoiser

The key design choice in DSL is to constrain every clean token embedding to lie on the unit sphere, $\|\boldsymbol{x}_i\|_2 = 1$ for all i. This is a structural constraint built into the embedding geometry, not a regularizer on the loss. As we now show, the benefit of this construction is to remove the nominal SNR t from the Bayes-optimal denoiser.

Under the localization channel Eq. 1, $P_t(\boldsymbol{x} \mid \boldsymbol{z}_t) \propto P(\boldsymbol{x}) \exp\left(\boldsymbol{x} \cdot \boldsymbol{z}_t - \frac{t}{2}\|\boldsymbol{x}\|_2^2\right)$ is the Bayes posterior factorization. The term $\frac{t}{2}\|\boldsymbol{x}\|_2^2$ is the only place the nominal SNR enters. Under unit-norm token embeddings, $\|\boldsymbol{x}\|_2^2 = L$ for every length-L sentence, so this term is sentence-independent and cancels after normalization. The posterior, and therefore the MMSE denoiser, depends only on $\boldsymbol{z}_t$:

$$\hat{\boldsymbol{x}}(\boldsymbol{z}, t) = \mathbb{E}_{P_t(\boldsymbol{x}|\boldsymbol{z})}[\boldsymbol{x}] = \mathbb{E}_{P(\boldsymbol{x})}[\boldsymbol{x} e^{\boldsymbol{x}\cdot\boldsymbol{z}}] \Big/ \mathbb{E}_{P(\boldsymbol{x})}[e^{\boldsymbol{x}\cdot\boldsymbol{z}}] = \hat{\boldsymbol{x}}(\boldsymbol{z}). \tag{3}$$

The SNR-invariance in Eq. 3 requires both the localization parameterization, which makes z a posterior natural coordinate with $a_\tau/\sigma_\tau^2 = 1$, and unit-sphere geometry, which makes $\|x_i\|^2$ vocabulary-constant. Either condition alone is insufficient; their composition produces the structural cancellation.

This SNR-invariance is the structural property that drives the rest of the paper. The Bayes denoiser does not need to know which t generated $\boldsymbol{z}_t$ to denoise it optimally; it only needs the noisy token. A single network can therefore in principle serve every SNR regime at once. We give the full derivation in Appendix A.2; geometric consequences—an entropy envelope on the induced token posterior $p(\boldsymbol{x}_i \mid \boldsymbol{z}_t)$ and Lipschitz smoothness—are deferred to Section 4.

This connection between spherical geometry and autonomous posterior-mean dynamics also appears in Sekimoto's independent work on classical spins [30, 31]; see Section 6 for the correspondence discussed in [32].

2.3 Arbitrary per-token SNR paths

Need to introduce $t = \gamma$, otherwise ppl don't know SNR=t in DSL's setting.

The SNR-invariance from §2.2 lets us drop the assumption that all tokens share a single SNR. We can let each position i evolve under its own SNR:

$$\boldsymbol{z}_i = \gamma_i \, \boldsymbol{x}_i + \sqrt{\gamma_i} \, \boldsymbol{\epsilon}_i, \qquad \boldsymbol{\epsilon}_i \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}),$$

and let each γ_i follow a piecewise-smooth path from 0 to ∞. Eq. 6 holds on any such path. Eqs. 4–5 describe the segments where γ_i increases; Proposition 2.1 describes the segments where it decreases.

For these per-token paths, the data-conditioned and unconditional dynamics are

$$d\boldsymbol{z}_i = \boldsymbol{x}_i \, \dot{\gamma}_i \, dt + \sqrt{\dot{\gamma}_i} \, d\mathbf{w}_i, \qquad \boldsymbol{x} \sim P(\boldsymbol{x}), \tag{4}$$

$$d\boldsymbol{z}_i = \hat{\boldsymbol{x}}_i(\boldsymbol{z}) \, \dot{\gamma}_i \, dt + \sqrt{\dot{\gamma}_i} \, d\mathbf{w}_i. \tag{5}$$

We show in Appendix A.4 that these two SDEs induce the same marginal on $\boldsymbol{z}_i$ along any admissible per-token path. Combined with the SNR-invariance of $\hat{\boldsymbol{x}}_i$, this means a *single* trained denoiser can be sampled along arbitrary per-token SNR schedules without changing the target distribution.

Lowering SNR. Eq. 4 and Eq. 5 describe increasing SNR. The same localization process also admits an exact reverse transition. This supplies the SNR-lowering operation underlying remasking in the DSL framework, which we use to interpret the masked samplers in Section **??**.

Proposition 2.1 (Reverse localization) *Let $\boldsymbol{z}_i$ and $\boldsymbol{z}_i'$ denote the states at SNRs γ_i and γ_i', respectively, coupled by Eq. 4. For $0 \le \gamma_i' < \gamma_i$,*

$$\boldsymbol{z}_i' \mid \boldsymbol{z}_i, \boldsymbol{x} \sim \mathcal{N}\left(\frac{\gamma_i'}{\gamma_i} \boldsymbol{z}_i, \; \gamma_i' \left(1 - \frac{\gamma_i'}{\gamma_i} \right) \boldsymbol{I} \right).$$

This reverse kernel is independent of $\boldsymbol{x}$. Setting $\gamma_i' = 0$ recovers the mask state $\mathbf{0}$, while $0 < \gamma_i' < \gamma_i$ gives a partial remask.

Generating toward higher SNR uses SNR-invariant denoiser $\hat{\boldsymbol{x}}(\boldsymbol{z})$; lowering SNR requires no denoiser. Together, these transitions allow a token to accumulate evidence, discard it through remasking, and be refined again within the same localization framework. In SNR time, the reverse process is a Brownian bridge toward $\mathbf{0}$ (Appendix A.5).

The path interpretation is then immediate. Driving one token from $\gamma{=}0$ to ∞ while holding others fixed corresponds to revealing that token in random-order autoregressive (ROAR) style; lowering selected positions to $\gamma{=}0$ (Proposition 2.1) and raising them again corresponds to masked diffusion with remasking; jointly increasing all γ_i corresponds to standard continuous diffusion. **Random-order AR generation, masked refinement, and continuous diffusion sampling are different paths through one shared family of per-token SNR configurations**—and, by the argument above, can all be served by one trained DSL model.

2.4 Exact NLL views that motivate the training support

The DSL training objective is not chosen as a sampler heuristic. It is derived from two exact likelihood views of the same per-token SNR formalism: a continuous path-integral NLL over finite-SNR denoising states, and an endpoint ROAR NLL over mask/reveal states. These identities are the likelihood-level reason that the mixed-support loss in Section 3.1 places training mass on both state families.

Continuous path-integral view. For any admissible per-token SNR path C, DSL admits the path-integral identity

$$-\log P(\boldsymbol{x}) \;=\; \tfrac{1}{2} \int_C \mathbf{E}(\boldsymbol{x}, \gamma) \cdot d\gamma, \qquad E_i(\boldsymbol{x}, \gamma) := \mathbb{E}_{p_\gamma(\boldsymbol{z}|\boldsymbol{x})}\big[\|\boldsymbol{x}_i - \hat{\boldsymbol{x}}_i(\boldsymbol{z})\|_2^2 \big] \tag{6}$$

(derivation in Appendix A.3). Under the Bayes-optimal denoiser, this error field is conservative, so the integral is path-independent: every admissible path gives the same $-\log P(\boldsymbol{x})$. Intermediate continuous-SNR states are therefore not heuristic interpolants—they are the integration variables of an exact likelihood.

Endpoint (ROAR) view. A second exact view comes from configurations in which some positions are already fully revealed ($\gamma{=}\infty$) and the rest remain fully masked ($\gamma{=}0$). In this regime the

Algorithm 1 DSL training

Require: Sequence length L; mixing weight $\lambda \in [0,1]$; ROAR endpoint values $\gamma_{\min}, \gamma_{\max}$; log-normal parameters (μ, σ); backbone f_θ with converter; learning rate η.

for each training step **do**
  Sample sentence $\boldsymbol{s} = (s_1, \ldots, s_L) \sim P_{\text{data}}$; embed $\boldsymbol{x}_i = \text{enc}(s_i)$.
  Draw $u \sim \text{Unif}[0,1]$.
  if $u < 1 - \lambda$ **then** ▷ ROAR branch
    Sample reveal-set size $k \sim \text{Unif}\{0, \ldots, L-1\}$, subset $A \subseteq [L]$ with $|A|{=}k$.
    Set $\gamma_i \leftarrow \gamma_{\max}$ for $i \in A$, and $\gamma_i \leftarrow \gamma_{\min}$ otherwise.
  else ▷ Continuous path branch
    Sample $\gamma_i \sim \text{LogNormal}(\mu, \sigma^2)$ independently for each $i \in [L]$.
  end if
  Sample $\boldsymbol{\epsilon}_i \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$; form noisy tokens $\boldsymbol{z}_i \leftarrow \gamma_i \boldsymbol{x}_i + \sqrt{\gamma_i}\,\boldsymbol{\epsilon}_i$.
  Compute converter outputs $\boldsymbol{m}_i^{\text{conv}}$ from each $\boldsymbol{z}_i$ via Eq. 11.
  Run backbone: $p_\theta(\cdot \mid \boldsymbol{z}) \leftarrow f_\theta(\boldsymbol{m}_1^{\text{conv}}, \ldots, \boldsymbol{m}_L^{\text{conv}})$.
  Update $\theta \leftarrow \theta - \eta \nabla_\theta \sum_{i=1}^{L} -\log p_\theta(s_i \mid \boldsymbol{z})$.
end for

sentence likelihood reduces to an autoregressive-style decomposition, and averaging over the size of the revealed subset, the choice of revealed positions, and the next index to predict yields the random-order autoregressive estimator

$$-\tfrac{1}{L} \log P(\boldsymbol{s}) \;=\; -\mathbb{E}_{(k,A,i)}\big[\log P(s_i \mid s_A)\big] \tag{7}$$

(derivation in Appendix A.7). This is the subset-conditioned objective used by masked-diffusion language models [28, 23]; here, it arises directly from the same DSL path formalism.

Bridge to mixed-support training. Eq. 6 and Eq. 7 are the two likelihood sources of the practical objective in Section 3.1. The former contributes intermediate finite-SNR states; the latter contributes endpoint mask/reveal states. Section 3 then implements this NLL-derived support with a single mixed cross-entropy objective.

3 DSL Training and Architecture

Section 2 identifies two exact likelihood views and their corresponding SNR paths. Turning them into a trainable recipe involves two design decisions: a mixed-support SNR sampler that covers both families under a single loss (§3.1), and a posterior-view converter that presents the corrupted state to the backbone neural network architecture in a form that matches the underlying geometry (§3.2).

3.1 Mixed-SNR training

To cover both state families in a single model, the DSL objective should place training mass on both, written as a convex combination

$$\mathcal{L}_{\text{theory}} \;=\; \lambda\, \mathcal{L}_{\text{cont-NLL}} \;+\; (1-\lambda)\, \mathcal{L}_{\text{ROAR-NLL}}. \tag{8}$$

One CE loss for both branches. $\mathcal{L}_{\text{ROAR-NLL}}$ is already a token-level categorical log-likelihood, so cross-entropy on $P_\theta(s_i \mid \boldsymbol{z})$ is its exact form. $\mathcal{L}_{\text{cont-NLL}}$ comes from the path-integral identity Eq. 6 and is naturally an embedding-space MSE on the Bayes denoiser $\hat{\boldsymbol{x}}_i(\boldsymbol{z}) = \mathbb{E}[\boldsymbol{x}_i \mid \boldsymbol{z}]$. We parameterize this denoiser through a categorical token posterior,

$$\hat{\boldsymbol{x}}_{\theta,i}(\boldsymbol{z}) = \sum_{v \in \mathcal{V}} P_\theta(s_i = v \mid \boldsymbol{z})\, \boldsymbol{x}_v. \tag{9}$$

Minimizing token-level cross-entropy makes $P_\theta(\cdot \mid \boldsymbol{z})$ match the true posterior $P(\cdot \mid \boldsymbol{z})$; at this optimum, the posterior mean above recovers the same Bayes denoiser that minimizes the path-integral MSE. Thus, for the continuous branch, CE is a posterior-matching surrogate for the exact MSE NLL rather than a separate heuristic. It also avoids embedding-space optimization pathologies known from prior continuous diffusion language models [8]. Full details are in Appendix A.8.

Implementation as mixed SNR sampling. Since both branches share the same CE loss, the λ-weighted mixture of losses equals, in expectation, a single CE loss under a λ-mixture of SNR distributions. We implement the latter directly: with probability λ we draw token-wise SNRs from a

continuous-path log-normal distribution, and with probability $1-\lambda$ from a ROAR-style endpoint distribution concentrated near the low- and high-SNR extremes. Both branches are trained under

$$\mathcal{L}_{\mathrm{CE}} \;=\; -\,\mathbb{E}_{(\boldsymbol{s},\boldsymbol{z})\sim q_{\mathrm{train}}} \sum_{i=1}^{L} \log p_\theta(s_i \mid \boldsymbol{z}), \tag{10}$$

realizing Eq. 8 as a single CE objective over a λ-mixed SNR support. We summarize the full training procedure in Algorithm 1; hyperparameters are listed in Appendix C.2.

3.2 Posterior-view converter and backbone

The DSL posterior lives in token-embedding geometry, while a Transformer backbone expects token-like inputs that interact stably under self-attention. We introduce a converter that maps each noisy token $\boldsymbol{z}_i$ to a mixture-of-tokens representation,

$$q_i^{\mathrm{conv}}(v \mid \boldsymbol{z}_i) \;=\; \mathrm{softmax}_\tau\big(\langle \boldsymbol{z}_i, \boldsymbol{x}_v\rangle + b_v\big), \qquad \boldsymbol{m}_i^{\mathrm{conv}} \;=\; \sum_{v\in\mathcal{V}\cup\{[\mathrm{MASK}]\}} q_i^{\mathrm{conv}}(v \mid \boldsymbol{z}_i)\, \boldsymbol{x}_v, \tag{11}$$

Here v indexes candidate token values, $\boldsymbol{x}_v = \mathrm{enc}(v)$, and $\tau > 0$ is a learnable converter temperature hyperparameter. We feed the sequence $(\boldsymbol{m}_1^{\mathrm{conv}}, \ldots, \boldsymbol{m}_L^{\mathrm{conv}})$ to a standard Transformer encoder / DiT-style denoiser. Intuitively, $\boldsymbol{m}_i^{\mathrm{conv}}$ is a vocabulary-weighted combination of unit-norm token embeddings: when $\boldsymbol{z}_i$ is very noisy the mixture is broad, and as the SNR grows it concentrates on a single token.

The converter has two properties required for the analysis in Section 4: $\|\boldsymbol{m}_i^{\mathrm{conv}}\|_2 \le 1$ for every $\boldsymbol{z}_i$ regardless of SNR, and $\boldsymbol{m}_i^{\mathrm{conv}}$ concentrates smoothly as SNR grows. Together these make the backbone see a stationary input geometry in which semantic preference is carried by direction and confidence grows by controlled radial motion. This mirrors the norm–direction posterior decomposition made explicit later in Equation (12).

Finally, DSL uses no informative timestep input: for compatibility with pretrained masked-diffusion DiTs [28], we keep the timestep embedding module but feed $t = 0$ throughout training and sampling, consistent with Eq. 3 since $\boldsymbol{z}_i$ and $\boldsymbol{m}_i^{\mathrm{conv}}$ already encode the corruption regime.

4 Posterior Geometry and Pathwise Error Control

Sections 2–3 define DSL as a posterior-state framework: exact NLL views identify the continuous-SNR and endpoint state families, and mixed-SNR training places support on both. This section explains what DSL geometry buys us for finite models. In the unit-sphere localization channel, the Bayes target is not a family of time-indexed denoisers $\{\hat{\boldsymbol{x}}(\cdot,t)\}_t$, but a single posterior-mean map $\hat{\boldsymbol{x}}(\boldsymbol{z}) = \mathbb{E}[\boldsymbol{x} \mid \boldsymbol{z}]$. Different inference paths therefore differ only in which realized states $\boldsymbol{z}$ they visit, and all query the same denoising map. We analyze this shared map below: its geometry separates token identity from confidence and yields bounded, smooth posterior motion, which helps control path-dependent approximation error.

Norm–direction posterior geometry. Fix a token position. Written as a function of the noisy token $\boldsymbol{z}$, the induced token posterior is

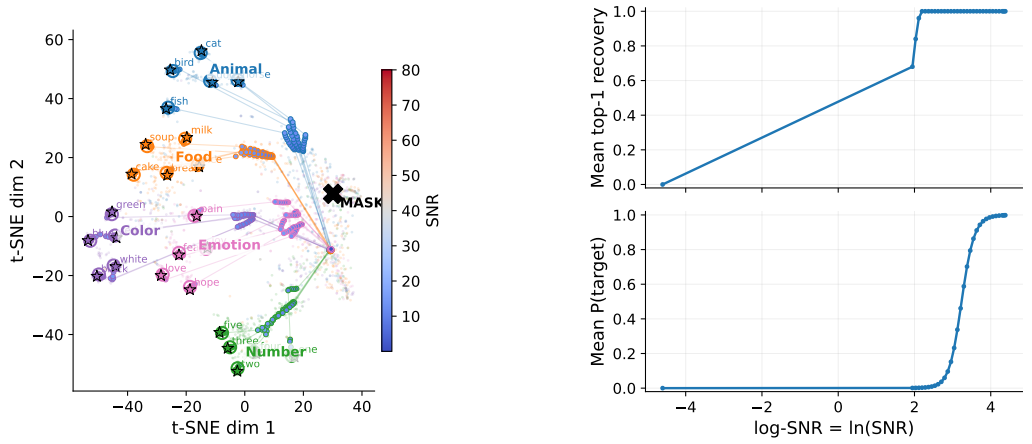
$$p(v \mid \boldsymbol{z}) \;\propto\; \pi_v \exp\big(\langle \boldsymbol{z}, \boldsymbol{x}_v\rangle\big) \;=\; \pi_v \exp\big(\|\boldsymbol{z}\|_2 \cos\theta_v\big), \tag{12}$$

where $\boldsymbol{x}_v := \mathrm{enc}(v)$ is the unit-norm embedding of token v, π_v is its prior, and θ_v is the angle between $\boldsymbol{z}$ and $\boldsymbol{x}_v$. Thus direction controls token preference while norm controls posterior concentration; this is a property of the unit-sphere channel, not a calibration objective. Figure 2 verifies that the trained converter follows this decomposition: trajectories organize by direction, and increasing SNR primarily increases concentration.

Bounded and Lipschitz posterior motion. The converter maps each noisy state to a bounded mixture-of-tokens input, with $\|\boldsymbol{m}_i^{\mathrm{conv}}\|_2 \le 1$. At fixed converter temperature, this bounds the achievable logit range and gives an entropy envelope for the token posterior. Moreover, for each position i and fixed $\boldsymbol{z}_{-i}$, the Bayes posterior-mean map $\boldsymbol{m}_i(\boldsymbol{z}_i) := \mathbb{E}[\boldsymbol{x}_i \mid \boldsymbol{z}]$ is 1-Lipschitz:

$$\|\boldsymbol{m}_i(\boldsymbol{z}_i) - \boldsymbol{m}_i(\boldsymbol{z}_i')\|_2 \le \|\boldsymbol{z}_i - \boldsymbol{z}_i'\|_2. \tag{13}$$

Thus a local perturbation at one position cannot be amplified by the Bayes denoising field at that position. Full entropy bounds are in Appendix A.10, and the proof of Eq. 13 is in Appendix A.11.



(a) (b)

Figure 2: **The DSL posterior decomposes into direction and norm axes. (a)** t-SNE projection of converter outputs for 25 probe tokens spanning five semantic classes. Trajectories sweep $\boldsymbol{z}_i = \gamma\, \boldsymbol{e}_v$ from `[MASK]` across $\gamma \in [10^{-2}, 80]$ and are colored by SNR. **(b)** Mean top-1 token recovery and target-token probability as γ increases.

Table 1: OWT ($L = 1024$) unconditional generation. We report MAUVE↑ as the primary metric; the full table with MAUVE / GenPPL / SentEnt is deferred to Appendix F. Bold marks the best MAUVE in each column among non-reference rows. Rows marked ‡ are taken from [40]; rows marked § from [27]. All DSL-FT rows use the same DSL-finetuned checkpoint. DSL-FT + ROAR (naive) and DSL-FT + MDLM are non-refinement decoder ablations: tokens are committed once and never remasked. The ReMDM-family rows add iterative remasking refinement.

Method	$T = 128$	$T = 256$	$T = 512$	$T = 1024$
Data (reference)	1.000	1.000	1.000	1.000
AR (reference)	–	–	–	0.760
SEDD (absorb)	0.007	0.008	0.009	0.008
MDLM‡	0.015	0.023	0.031	0.042
MDLM+FB‡	0.064	0.086	0.103	0.133
MDLM+DFM‡	0.041	0.098	0.168	0.254
ReMDM‡	0.057	0.216	0.350	0.403
PRISM‡	0.175	0.268	0.281	0.260
PRISM-loop‡	0.118	0.294	0.423	0.527
ADLM§	0.140	0.349	0.573	0.699
DSL-FT + ROAR (naive)	–	–	–	0.551
DSL-FT + MDLM	0.402	0.481	0.506	0.495
DSL-FT + ReMDM (confidence-based)	0.615	0.622	**0.707**	0.610
DSL-FT + ReMDM-loop (principled η_{cap})	**0.639**	**0.661**	0.651	**0.722**

5 Experiments

On OpenWebText (OWT), we evaluate the gains from DSL training with masked diffusion sampling, the additional benefits of ReMDM samplers [40], and the composition of continuous and remasking samplers through hybrid sampling. All three evaluations use the same DSL-finetuned checkpoint, connecting the per-token SNR paths of Sec. 2.3 to practical sampling procedures. We additionally evaluate likelihood on Text8 using a model trained from scratch.

5.1 Setup

All OWT DSL results use the same checkpoint fine-tuned from the official MDLM model. Following the MDLM/ReMDM protocol [28, 40], we generate 5k sequences of length $L = 1024$ with GPT-2 BPE tokenization and use nucleus sampling with top-$p = 0.9$ where applicable. Our primary metric is MAUVE, computed with GPT-2 Large embeddings and $K = 500$ clusters; GenPPL and sentence entropy are reported as diagnostics in Appendix F. For DSL-FT + ReMDM-loop at $T = 128$, bootstrap resampling of generated texts and variation of the clustering seed give MAUVE standard deviations of 0.020 and 0.017, respectively; these are not training-seed error bars.

Algorithm 2 DSL sampling with ReMDM sampler. Adapted from [40, Algorithm 1]; in DSL, the ReMDM mask state is represented by zero SNR.

Input: DSL denoiser $\boldsymbol{x}_\theta$, sampling steps T, noise schedule α_t, clean endpoint $\mathrm{SNR}_{\max}$, remask schedule σ_t.
Initialize: set every token embedding to the mask endpoint, i.e., $\mathrm{SNR}(\boldsymbol{z}_T) = 0$.
for $i = T, T-1, \ldots, 1$ **do**
  $t = i/T, \quad s = (i-1)/T$; set α_t, α_s.
  Set $\sigma_t \in [0, \sigma_t^{\max}]$, where $\sigma_t^{\max} = \min\{1, (1-\alpha_s)/\alpha_t\}$.
  Predict clean-token posteriors $\widehat{\boldsymbol{x}} = \boldsymbol{x}_\theta(\boldsymbol{z}_t)$.
  Form the ReMDM endpoint posterior $p_\theta(\boldsymbol{z}_s \mid \boldsymbol{z}_t) = q_\sigma^{\mathrm{DSL}}\big(\boldsymbol{z}_s \mid \boldsymbol{z}_t, \boldsymbol{x} = \widehat{\boldsymbol{x}}\big)$, where ReMDM's mask atom is interpreted as $\mathrm{SNR} = 0$ and its clean-token atom as $\mathrm{SNR} = \mathrm{SNR}_{\max}$.
  Sample $\boldsymbol{z}_s \sim p_\theta(\boldsymbol{z}_s \mid \boldsymbol{z}_t)$.
  Realize sampled clean tokens at $\mathrm{SNR}_{\max}$; realize remasked tokens by resetting their SNR to 0.
  Set $\boldsymbol{z}_t \leftarrow \boldsymbol{z}_s$.
end for
Output: decoded tokens from the final clean-endpoint latents.

For masked diffusion sampling in DSL, masked positions have $\boldsymbol{z}_i = \mathbf{0}$ (SNR 0), while sampled tokens s_i are represented by $\boldsymbol{z}_i = \gamma_{\max}\,\mathrm{enc}(s_i)$, with $\gamma_{\max} = 100$ approximating the clean endpoint. All samplers use the same per-token posteriors $p_\theta(s_i \mid \boldsymbol{z})$ without timestep conditioning.

5.2 Masked diffusion sampling without remasking

We first evaluate the same DSL-finetuned checkpoint using MDLM and random order AR (ROAR) sampling without remasking. The vanilla MDLM sampler follows the official implementation of [28], progressively converting masked positions into sampled tokens. ROAR instead reveals one token at a time in a uniformly random order. In both samplers, masked positions have $\boldsymbol{z}_i = \mathbf{0}$ (SNR 0). After sampling token s_i, we set $\boldsymbol{z}_i = \gamma_{\max}\,\mathrm{enc}(s_i)$ (SNR $= \gamma_{\max}$). Full protocols are given in Appendices C.4 and C.5.

Under the same vanilla MDLM sampler, replacing the original checkpoint with DSL-FT raises MAUVE from $0.015/0.023/0.031/0.042$ to $0.402/0.481/0.506/0.495$ at $T = 128, 256, 512, 1024$, respectively (Table 1). This shows that DSL improves generation quality across all evaluated budgets without changing the sampling rule or introducing remasking. Matched-step continued training of MDLM does not produce comparable gains, supporting the benefit of DSL adaptation beyond additional training alone (Appendix F.2).

ROAR shows that the smae checkpoint also supports random-order autoregressive generation without retraining. It performs a single reveal pass with $T = L = 1024$, committing each token exactly once, and performs no confidence-based ordering or remasking. Despite this simple sampling procedure, DSL-FT + ROAR reaches MAUVE 0.551, exceeding DSL-FT + MDLM at the same budget and several prior baselines. Together, these results demonstrate strong generation quality under both MDLM and ROAR sampling using one DSL checkpoint. Training-component ablations are reported in Appendix F.2, Table 9.

5.3 Masked diffusion sampling with ReMDM

We next evaluate the same DSL checkpoint with ReMDM samplers [40], extending our evaluation to masked diffusion sampling with remasking. Algorithm 2 summarizes the sampling method. Starting from the zero-SNR mask state, the sampler repeatedly predicts clean-token posteriors and reveals masked positions by sampling tokens and assigning their high-SNR representations. Remasking resets previously generated positions to zero-SNR, allowing them to be sampled again conditioned on the remaining tokens. We evaluate two schedules in [40] – confidence-based schedule ReMDM-conf and ReMDM-loop; the latter activates remasking within a loop window using the step-budgeted capped schedule $\eta_{\mathrm{cap}}(T)$. Full settings are given in Appendix C.4. Remasking implements the decreasing-SNR direction of Sec. 2.3: resetting a position to $\boldsymbol{z}_i = \mathbf{0}$ is the $\gamma_i' = 0$ case of reverse localization (Proposition 2.1). The same DSL denoiser then predicts from the updated state. The reverse kernel specifies the SNR-lowering operation, while the remasking schedule determines when and which positions are revisited.

Table 1 compares these samplers against published diffusion baselines and the preceding DSL results without remasking. Both DSL remasking variants outperform ReMDM baseline across all four step budgets. ReMDM-conf gives the best DSL result at $T = 512$, with MAUVE 0.707; ReMDM-

loop performs best at $T = 128$, 256, and 1024, reaching 0.639, 0.661, and 0.722, respectively. At $T = 1024$, the same checkpoint achieves 0.495 with MDLM sampling and 0.551 with ROAR, compared with 0.722 with ReMDM-loop. These results show that DSL achieves strong generation quality with ReMDM samplers, extending its performance acros different sampling strategies without retraining.

5.4 Hybrid continuous–discrete sampling

We also evaluate a hybrid sampler that combines continuous sampling and masked diffusion sampling with remasking using the same DSL checkpoint. The hybrid sampler first runs a short continuous denoising stage, here we adopt an EDM-style sampler [16] which denoises until σ_{switch}. And then we continue with confidence-based remasking samplin: selected tokens are reset to SNR$= 0$ and resampled. This tests whether one checkpoint can combine continuous and masked sampling within a single generation run. Appendix C.6 gives the full procedure.

Table 2: Few-step OWT generation with the DSL hybrid continuous-then-MDM sampler. All rows use the same DSL-finetuned checkpoint. For each $(T_{\text{cont}}, T_{\text{MDM}})$, we report a balanced σ_{switch} from a small handoff sweep. The full sweep with MAUVE / GenPPL / SentEnt is in Appendix F.

T_{cont}	T_{MDM}	NFE	σ_{switch}	MAUVE↑
16	16	32	0.49	0.501
16	32	48	0.49	0.662
16	48	64	0.49	0.702
32	16	48	0.46	0.587
32	32	64	0.46	0.589
32	48	80	0.46	0.724

Table 2 reports strong generation quality with short continuous and masked sampling stages. With $(T_{\text{cont}}, T_{\text{disc}}) = (16, 32)$, MAUVE reaches 0.662; with $(32, 48)$, it reaches 0.724, comparable to ReMDM-loop at $T = 1024$. The budget allocation and switching point jointly affect quality: $(16, 48)$ achieves 0.702, whereas $(32, 32)$ gives 0.589 despite the same nominal total step count. Thus, allocating more steps to continuous sampling does not necessarily improve the final result. These configurations demonstrate the potential of combining samplers; systematic optimization of the switching point and compute allocation remains future work.

5.5 Likelihood-based evaluation on Text8

The DSL framework admits two likelihood estimators following Section 2.4: the path-integral identity Eq. 6 and the ROAR estimator Eq. 7. Table 3 reports BPC on Text8. DSL achieves ≤ 1.45 BPC, the first continuous-state NLL estimate to approach the masked-discrete-diffusion range (MDLM ≤ 1.40, MD4 ≤ 1.37). Continuous-state estimates are upper bounds via ELBO and so cannot match the exact NLL accessible to masked and AR models; the relevant comparison is to other continuous-state baselines, where DSL improves on Plaid by ≥ 0.03 BPC. Setup details in Appendix F.

Table 3: Bits Per Character (BPC) on Text8 test set. Continuous diffusion and any-order entries are variational upper bounds; AR and flow entries are exact NLL. DSL is the only continuous-state entry without time conditioning, and is trained for 0.5M steps versus 1M for the diffusion baselines.

Method	BPC (↓)
Continuous Diffusion	
Plaid (MD4 impl.)	≤ 1.48
DSL (Ours)	≤ 1.45
Discrete Diffusion	
Mult. Diffusion	≤ 1.72
D3PM Uniform	≤ 1.61
D3PM Absorb	≤ 1.45
SEDD Absorb	≤ 1.41
MDLM	≤ 1.40
MD4	≤ 1.37
Autoregressive	
Transformer AR	**1.23**
IAF/SCF	1.88
AR Argmax Flow	1.39
AR Discrete Flow	**1.23**
Any-order Autoregressive	
ARDM	≤ 1.43
MAC	≤ 1.40

6 Related Work

Continuous and masked diffusion language models. Continuous diffusion LMs generate text by denoising token embeddings, typically with timestep or noise-level conditioning [19, 36, 4, 8]. Bayesian Flow Networks instead update distribution parameters through Bayesian inference and condition their network on both these parameters and time [6]. Token geometry has also been explored: CDCD normalizes token embeddings [4], while RDLM defines diffusion on a hypersphere [15]. Masked diffusion models generate tokens by unmasking [2, 28, 33], and ReMDM permits previously generated tokens to be remasked [40]. Hybrid approaches also connect discrete and Gaussian diffusion [29, 25]. DSL combines unit-norm clean token embeddings with stochastic localization, while allowing noisy tokens to evolve in $\mathbb{R}^d$. This combination makes the Bayes-optimal denoiser SNR-invariant

and allows one network to support continuous and masked diffusion sampling (Appendix B). Our companion work scales DSL to an 8B masked diffusion LM DSL-LLaDA [42].

Stochastic localization and classical spins. DSL builds on stochastic localization [5, 22] and its use in sampling algorithms [1, 7]. Sekimoto [30, 31] studies autonomous Langevin drifts that are martingales along their own dynamics, obtaining, in the isotropic case, a drift expressed as the mean of a classical spin on the unit sphere. DSL arrives at a corresponding structure from the other direction: starting with **constant-norm token embeddings**, stochastic localization yields an autonomous posterior-mean drift that is also a martingale along the localization process. The two constructions differ in their base measures: Sekimoto's isotropic formulation uses the uniform measure on the sphere, whereas DSL uses the generally non-uniform discrete data distribution $P(\boldsymbol{x})$ over token sequences. Sekimoto's subsequent work [32] discusses the connection between these approaches.

Diffusion and autoregressive generation. Dieleman [3] interprets image diffusion as approximate autoregression over spatial frequencies. This perspective motivated us to investigate how continuous denoising and token-by-token generation could be understood within a common framework for language. DSL provides such a framework through per-token SNR paths: continuous denoising gradually accumulates evidence across positions, while autoregressive generation reveals tokens one at a time. Under the Bayes-optimal denoiser, these paths admit an exact likelihood correspondence. This connection also guides model design: DSL supports both sampling procedures with one SNR-invariant denoiser.

7 Conclusion

We introduced Discrete Stochastic Localization (DSL), a continuous-state framework for discrete sequence generation with unit-sphere token embeddings. Combining this geometry with stochastic localization makes the Bayes-optimal clean-token posterior depend only on the noisy state, without additional SNR conditioning. DSL implements this posterior with a time-invariance denoiser and a mixture-of-tokens interface for standard Transformer/DiT backbones. A single trained denoiser supports continuous denoising, masked diffusion with remasking, random-order autoregression, and hybrid sampling, with strong OpenWebText generation results and Text8 likelihood bounds approaching those of masked diffusion models. These results show how token geometry and the choice of noisy-state parameterization can bring continuous and discrete generation together within one model.

Limitations. Our evaluation is limited to OpenWebText generation and Text8 likelihood. The reported ReMDM-family and hybrid samplers are not the result of an exhaustive decoding-policy search; they are intended to test whether DSL training improves the underlying posterior and whether one checkpoint can be queried along multiple SNR paths. More optimized remasking schedules, hybrid handoff rules, larger models, longer contexts, and conditional generation remain important directions for future work.

Acknowledgments and Disclosure of Funding

Research was supported by National Science Foundation CAREER grant no. IIS-2046086.

References

[1] Ahmed El Alaoui, Andrea Montanari, and Mark Sellke. Sampling from the sherrington-kirkpatrick gibbs measure via algorithmic stochastic localization, 2024.

[2] Jacob Austin, Daniel D Johnson, Jonathan Ho, Daniel Tarlow, and Rianne van den Berg. Structured denoising diffusion models in discrete state-spaces. *Advances in Neural Information Processing Systems*, 34:17981–17993, 2021.

[3] Sander Dieleman. Diffusion is spectral autoregression, 2024.

[4] 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.

[5] Ronen Eldan. Thin shell implies spectral gap up to polylog via a stochastic localization scheme. *Geometric and Functional Analysis*, 23(2):532–569, 2013.

[6] Alex Graves, Rupesh Kumar Srivastava, Timothy Atkinson, and Faustino Gomez. Bayesian flow networks, 2025.

[7] Louis Grenioux, Maxence Noble, Marylou Gabrié, and Alain Oliviero Durmus. Stochastic localization via iterative posterior sampling, 2026.

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

[9] Dongning Guo, Shlomo Shamai, and Sergio Verdú. Mutual information and minimum mean-square error in gaussian channels. *IEEE transactions on information theory*, 51(4):1261–1282, 2005.

[10] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020.

[11] Jonathan Ho, Tim Salimans, Alexey Gritsenko, William Chan, Mohammad Norouzi, and David J Fleet. Video diffusion models. *Advances in neural information processing systems*, 35:8633–8646, 2022.

[12] Emiel Hoogeboom, Alexey A Gritsenko, Jasmijn Bastings, Ben Poole, Rianne van den Berg, and Tim Salimans. Autoregressive diffusion models. *arXiv preprint arXiv:2110.02037*, 2021.

[13] Emiel Hoogeboom, Didrik Nielsen, Priyank Jaini, Patrick Forré, and Max Welling. Argmax flows and multinomial diffusion: Learning categorical distributions. *Advances in neural information processing systems*, 34:12454–12465, 2021.

[14] Xun Huang, Zhengqi Li, Guande He, Mingyuan Zhou, and Eli Shechtman. Self forcing: Bridging the train-test gap in autoregressive video diffusion. *arXiv preprint arXiv:2506.08009*, 2025.

[15] Jaehyeong Jo and Sung Ju Hwang. Continuous diffusion model for language modeling, 2025.

[16] Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. *arXiv preprint arXiv:2206.00364*, 2022.

[17] Xianghao Kong, Rob Brekelmans, and Greg Ver Steeg. Information-theoretic diffusion. In *International Conference on Learning Representations*, 2023.

[18] Zhifeng Kong, Wei Ping, Jiaji Huang, Kexin Zhao, and Bryan Catanzaro. Diffwave: A versatile diffusion model for audio synthesis. *arXiv preprint arXiv:2009.09761*, 2020.

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

[20] Aaron Lou, Chenlin Meng, and Stefano Ermon. Discrete diffusion modeling by estimating the ratios of the data distribution. In *Forty-first International Conference on Machine Learning*, 2024.

[21] Matt Mahoney. Large Text Compression Benchmark. `https://www.mattmahoney.net/dc/text.html`, 2006. Accessed: 2025-05-11.

[22] Andrea Montanari. Sampling, diffusions, and stochastic localization. *arXiv preprint arXiv:2305.10690*, 2023.

[23] 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. *arXiv preprint arXiv:2502.09992*, 2025.

[24] Daniel P Palomar and Sergio Verdú. Gradient of mutual information in linear vector gaussian channels. *IEEE Transactions on Information Theory*, 52(1):141–154, 2005.

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

[26] Hubert Ramsauer, Bernhard Schäfl, Johannes Lehner, Philipp Seidl, Michael Widrich, Thomas Adler, Lukas Gruber, Markus Holzleitner, Milena Pavlović, Geir Kjetil Sandve, et al. Hopfield networks is all you need. *arXiv preprint arXiv:2008.02217*, 2020.

[27] Litu Rout, Constantine Caramanis, and Sanjay Shakkottai. Anchored diffusion language model. *arXiv preprint arXiv:2505.18456*, 2025.

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

[29] Subham Sekhar Sahoo, Justin Deschenaux, Aaron Gokaslan, Guanghan Wang, Justin T Chiu, and Volodymyr Kuleshov. The diffusion duality. In *ICLR 2025 Workshop on Deep Generative Model in Machine Learning: Theory, Principle and Efficacy*.

[30] Ken Sekimoto. Martingale drift of langevin dynamics and classical canonical spin statistics. *Phys. Rev. E*, 109:014106, Jan 2024.

[31] Ken Sekimoto. Martingale drift of langevin dynamics and classical canonical spin statistics. ii. *Phys. Rev. E*, 111:044102, Apr 2025.

[32] Ken Sekimoto. Autonomous self-harmonic drift in langevin dynamics causes a compaction of underlying domain for the density of stochastic localization. *arXiv preprint arXiv:2609.36232*, 2026.

[33] Jiaxin Shi, Kehang Han, Zhe Wang, Arnaud Doucet, and Michalis Titsias. Simplified and generalized masked diffusion for discrete data. *Advances in neural information processing systems*, 37:103131–103167, 2024.

[34] Andy Shih, Dorsa Sadigh, and Stefano Ermon. Training and inference on any-order autoregressive models the right way. *Advances in Neural Information Processing Systems*, 35:2762–2775, 2022.

[35] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020.

[36] 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.

[37] Qiao Sun, Zhicheng Jiang, Hanhong Zhao, and Kaiming He. Is noise conditioning necessary for denoising generative models? *arXiv preprint arXiv:2502.13129*, 2025.

[38] Dustin Tran, Keyon Vafa, Kumar Agrawal, Laurent Dinh, and Ben Poole. Discrete flows: Invertible generative models of discrete data. *Advances in Neural Information Processing Systems*, 32, 2019.

[39] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[40] Guanghan Wang, Yair Schiff, Subham Sekhar Sahoo, and Volodymyr Kuleshov. Remasking discrete diffusion models with inference-time scaling. *arXiv preprint arXiv:2503.00307*, 2025.

[41] Yunshu Wu, Yingtao Luo, Xianghao Kong, Evangelos E Papalexakis, and Greg V Steeg. Your diffusion model is secretly a noise classifier and benefits from contrastive training. *Advances in Neural Information Processing Systems*, 37:32370–32399, 2024.

[42] 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. In Hung-Yi Lee, Sunipa Dev, Jessy Li, and Desmond Elliott, editors, *Findings of the Association for Computational Linguistics: EMNLP 2026*, pages 44727–44752, Budapest, Hungary, October 2026. Association for Computational Linguistics.

[43] Zachary Ziegler and Alexander Rush. Latent normalizing flows for discrete sequences. In *International Conference on Machine Learning*, pages 7673–7682. PMLR, 2019.

Appendix Overview

Appendix A: Mathematical Derivations

§A.1 Notation and Summary Table
§A.2 Optimal Denoiser is SNR-Invariant
§A.3 Exact Likelihood over Arbitrary SNR Paths
§A.4 Equivalence of Conditional and Unconditional Dynamics
§A.5 Autoregressive Contours as a Special Case
§A.6 Exact ROAR Likelihood Decomposition and Estimator
§A.7 Cross-Entropy as a Surrogate for the Path-Integral MSE
§A.8 Posterior-State Matching and Inference-Time Error Correction
§A.9 Confidence Geometry: Norm–Direction Decomposition and Bounded Entropy
§A.10 Lipschitz Continuity of the Induced Posterior-Mean Map

Appendix B: Continuous States Are Not Enough

Posterior Coordinates vs. Timestep-Conditioned Denoising

Appendix C: Training and Sampling Details

§C.1 OWT Finetuning Settings
§C.2 Mixed SNR Support Used in Training
§C.3 Masked Refinement with a DSL Checkpoint
§C.4 Sampling Settings on OWT
§C.5 ROAR Sampler Details
§C.6 Hybrid Sampler Details

Appendix D: Mechanistic Analysis

§D.1 Cyclic Toy Setup and Correction Example
§D.2 Why Targeted Remasking Matters
§D.3 Robustness to Self-Generated Intermediate Drafts
§D.4 Sampling Diagnostics and Over-Refinement

Appendix E: Calibration and Endpoint-Smoothing Ablation

§E.1 Atomic vs. Smoothed ROAR Endpoints
§E.2 Near-Clean Calibration
§E.3 Downstream Step–Quality Tradeoff

Appendix F: Additional Experiments

§F.1 OpenWebText Generation
§F.2 Text8 Likelihood Evaluation

Appendix G: Limitations and Broader Impacts

Appendix H: Assets and Licenses

Appendix I: Additional Qualitative Samples

§I.1 Text8 Samples

§I.2 OpenWebText Samples

A Mathematical Derivations

This appendix collects the derivations underlying the DSL method in Section 2. The main text only uses three consequences of these results: (i) the denoiser is realized-state invariant under unit-sphere token geometry, (ii) exact likelihood views motivate both continuous and ROAR-style support in training, and (iii) the induced posterior-mean map is smooth. Here we provide the full proofs and auxiliary technical remarks.

A.1 Notation and Summary Table

Symbol	Description
$i = 1, \dots, L$	Index for sequences of length L
$t \in [0, \infty)$	Index for continuous time dynamics, equal to SNR
$\gamma_i \in [0, \infty)$	Per token SNR, defined on a contour or as function of t
$s_i \in \mathcal{V}$	i-th symbol/token from the vocabulary $\mathcal{V}$
$\boldsymbol{x}_i = \text{enc}(s_i) \in \mathbb{R}^d$	Embed token as vector on unit hyper-sphere surface ($\|\boldsymbol{x}\| = 1$)
$\boldsymbol{z}_i \in \mathbb{R}^d$	Noisy embedding of the i-th token at noise level t
$d\boldsymbol{z} = \boldsymbol{x}\, dt + dW$	Conditional SDE for dynamics
$\boldsymbol{z} = t\, \boldsymbol{x} + \sqrt{t}\, \boldsymbol{\epsilon}$	Sample conditional marginal
$\boldsymbol{z} \sim \mathcal{N}(t\, \boldsymbol{x}, t)$	Alternate form to sample conditional marginal
$d\boldsymbol{z} = \hat{\boldsymbol{x}}(\boldsymbol{z})\, dt + dW$	Unconditional SDE with equivalent dynamics to conditional SDE
$\hat{\boldsymbol{x}}(\boldsymbol{z}, t) = \mathbb{E}[\boldsymbol{x} \vert \boldsymbol{z}(t)]$	Optimal denoiser would generally depend on time or noise level
$\hat{\boldsymbol{x}}(\boldsymbol{z}) = \mathbb{E}_{\boldsymbol{x}}[\boldsymbol{x} e^{\boldsymbol{x}\cdot\boldsymbol{z}}] / \mathbb{E}_{\boldsymbol{x}}[e^{\boldsymbol{x}\cdot\boldsymbol{z}}]$	Key result: optimal denoiser doesn't depend on time for spherical embeddings
$\hat{\boldsymbol{x}}(\boldsymbol{z}) = \nabla_{\boldsymbol{z}}\, \text{LSE}_{\boldsymbol{x}\in\mathcal{D}}(\boldsymbol{z}\cdot\boldsymbol{x})$	For empirical distribution, relates to gradient of cumulant generating function, or modern Hopfield energy[26]
$-\log P(\boldsymbol{s}) = -\log P(\boldsymbol{x})$ $= {}^1\!/_2 \int_0^\infty dt\, \mathbb{E}_{\boldsymbol{z}(t)\vert\boldsymbol{x}}[\|\boldsymbol{x} - \hat{\boldsymbol{x}}(\boldsymbol{z})\|^2]$	Probability relates to MMSE, for any one-to-one embedding [9, 17, 41]

Table 4: Summary of notation and key relations.

A.2 Optimal Denoiser is SNR-Invariant

We now derive the optimal denoiser for the noise channel with per token SNR described in the main text. The denoiser is as follows, where we first re-write with Bayes rule, then expand the Gaussian noise channel.

$$\begin{aligned}
\hat{\boldsymbol{x}}(\boldsymbol{z}, \boldsymbol{\gamma}) \equiv \mathbb{E}_{p_{\boldsymbol{\gamma}}(\boldsymbol{x}\vert\boldsymbol{z})}[\boldsymbol{x}] &= \frac{\sum_{\boldsymbol{x}} p_{\boldsymbol{\gamma}}(\boldsymbol{z}\vert\boldsymbol{x}) P(\boldsymbol{x}) \boldsymbol{x}}{p_{\boldsymbol{\gamma}}(\boldsymbol{z})} = \frac{\sum_{\boldsymbol{x}} p_{\boldsymbol{\gamma}}(\boldsymbol{z}\vert\boldsymbol{x}) P(\boldsymbol{x}) \boldsymbol{x}}{\sum_{\bar{\boldsymbol{x}}} p_{\boldsymbol{\gamma}}(\boldsymbol{z}\vert\bar{\boldsymbol{x}}) P(\bar{\boldsymbol{x}})} \\
&= \frac{\sum_{\boldsymbol{x}} \exp(-{}^1\!/_2 \|\boldsymbol{z} - \boldsymbol{\gamma}\boldsymbol{x}\|^2 / \boldsymbol{\gamma}) / \mathcal{Z}(\boldsymbol{\gamma}) P(\boldsymbol{x}) \boldsymbol{x}}{\sum_{\bar{\boldsymbol{x}}} \exp(-{}^1\!/_2 \|\boldsymbol{z} - \boldsymbol{\gamma}\bar{\boldsymbol{x}}\|^2 / \boldsymbol{\gamma}) / \mathcal{Z}(\boldsymbol{\gamma}) P(\bar{\boldsymbol{x}})} \\
&= \frac{\sum_{\boldsymbol{x}} \exp(\boldsymbol{z}\cdot\boldsymbol{x}) P(\boldsymbol{x}) \boldsymbol{x}}{\sum_{\bar{\boldsymbol{x}}} \exp(\boldsymbol{z}\cdot\bar{\boldsymbol{x}}) P(\bar{\boldsymbol{x}})} \\
\hat{\boldsymbol{x}}(\boldsymbol{z}, \boldsymbol{\gamma}) = \hat{\boldsymbol{x}}(\boldsymbol{z}) &= \mathbb{E}_{\boldsymbol{x}}[\boldsymbol{x} e^{\boldsymbol{x}\cdot\boldsymbol{z}}] / \mathbb{E}_{\boldsymbol{x}}[e^{\boldsymbol{x}\cdot\boldsymbol{z}}]
\end{aligned}$$

The division and multiplication between $\boldsymbol{\gamma}$ and $\boldsymbol{z}$ should be understood to be applied component-wise, per token. We canceled out the $\|\boldsymbol{z}\|$ term, and due to the normed embedding, $\exp(-1/2\boldsymbol{\gamma}\|\boldsymbol{x}\|^2)$ terms become constant and cancel out, leaving us with no dependence on $\boldsymbol{\gamma}$ at all. The distribution $P(\boldsymbol{x})e^{\boldsymbol{z}\cdot\boldsymbol{x}}/\mathcal{Z}(\boldsymbol{z})$ is sometimes referred to as an exponentially tilted distribution of $P(\boldsymbol{x})$. This form also can be written as $\nabla_z \log \mathbb{E}_{\boldsymbol{x}}[e^{\boldsymbol{x}\cdot\boldsymbol{z}}]$, which is a gradient of the cumulant generating function.

The results above also hold for the simpler case where $\gamma_i(t) = t$, leading to a denoiser which is invariant to t. Recent work has also empirically noted that even for traditional diffusion models, noise conditioning is not always necessary or useful[37].

Constant norm is also necessary. For any two support points $\boldsymbol{x}, \boldsymbol{x}'$ of P,

$$\log \frac{P_{\boldsymbol{\gamma}}(\boldsymbol{x} \mid \boldsymbol{z})}{P_{\boldsymbol{\gamma}}(\boldsymbol{x}' \mid \boldsymbol{z})} = \log \frac{P(\boldsymbol{x})}{P(\boldsymbol{x}')} + \langle \boldsymbol{z}, \boldsymbol{x} - \boldsymbol{x}' \rangle - \tfrac{1}{2}\sum_{i=1}^{L} \gamma_i \big(\|\boldsymbol{x}_i\|^2 - \|\boldsymbol{x}_i'\|^2\big).$$

The first two terms do not involve $\boldsymbol{\gamma}$. The posterior is therefore $\boldsymbol{\gamma}$-invariant for all $\boldsymbol{z}$ and all $\boldsymbol{\gamma} \in \mathbb{R}_+^L$ if and only if $\|\boldsymbol{x}_i\| = \|\boldsymbol{x}_i'\|$ for every position i and every pair of support points. Under a single global SNR $\gamma_i \equiv t$ the weaker condition $\|\boldsymbol{x}\|^2 = \text{const}$ suffices. Unit-norm token embeddings satisfy both.

Note that constant norm is enough, people normally use unit-norm is because it's natural.

A.3 Exact Likelihood over Arbitrary SNR Paths

We make use of the following theorem, re-stated with our setting and notation, from [24], to prove Equation (6).

Theorem A.1 ([24], Thm. 5) *Consider the signal model*

$$\boldsymbol{z}_i = \gamma_i \boldsymbol{x}_i + \sqrt{\gamma_i}\boldsymbol{\epsilon}_i,$$

where $\boldsymbol{x}$ is arbitrarily distributed with finite second-order moments, and $\boldsymbol{\epsilon}_i \sim \mathcal{N}(0, \mathbf{I})$ is independent Gaussian noise.

Then the gradient of the Kullback–Leibler divergence between the conditional output distribution $p_{\boldsymbol{\gamma}}(\boldsymbol{z}|\boldsymbol{x})$ and the unconditional output distribution $p_{\boldsymbol{\gamma}}(\boldsymbol{z})$ is

$$\nabla_{\boldsymbol{\gamma}} D\big(p_{\boldsymbol{\gamma}}(\boldsymbol{z}|\boldsymbol{x}) \,\|\, p_{\boldsymbol{\gamma}}(\boldsymbol{z})\big) = 1/2 \mathbf{E}(\boldsymbol{x}, \boldsymbol{\gamma}),$$

where $E_i(\boldsymbol{x}, \boldsymbol{\gamma}) \equiv \mathbb{E}_{p_{\boldsymbol{\gamma}}(\boldsymbol{z}|\boldsymbol{x})}[\|\boldsymbol{x}_i - \hat{\boldsymbol{x}}_i(\boldsymbol{z})\|^2]$, and $\hat{\boldsymbol{x}}(\boldsymbol{z}) = \mathbb{E}_{p_{\boldsymbol{\gamma}}(\boldsymbol{x}|\boldsymbol{z})}[\boldsymbol{x}]$ is the MMSE estimator.

We use this theorem in conjunction with the fundamental theorem of calculus for line integrals to obtain the following.

$$\int_C d\boldsymbol{\gamma} \cdot \nabla_{\boldsymbol{\gamma}} D\big(p_{\boldsymbol{\gamma}}(\boldsymbol{z}|\boldsymbol{x}) \,\|\, p_{\boldsymbol{\gamma}}(\boldsymbol{z})\big) = 1/2 \int_C d\boldsymbol{\gamma} \cdot \mathbf{E}(\boldsymbol{x}, \boldsymbol{\gamma}) = D\big(p_{\boldsymbol{\gamma}}(\boldsymbol{z}|\boldsymbol{x}) \,\|\, p_{\boldsymbol{\gamma}}(\boldsymbol{z})\big)\Big|_{\boldsymbol{\gamma}_0}^{\boldsymbol{\gamma}_1} \tag{14}$$

Here, $\boldsymbol{\gamma}_0, \boldsymbol{\gamma}_1$ define the endpoints of the contour, which must be piecewise smooth. Next, let's consider some of the limits.

$$\lim_{\boldsymbol{\gamma}\to 0} D\big(p_{\boldsymbol{\gamma}}(\boldsymbol{z}|\boldsymbol{x}) \,\|\, p_{\boldsymbol{\gamma}}(\boldsymbol{z})\big) = 0 \tag{15}$$

$$\lim_{\boldsymbol{\gamma}\to \infty} D\big(p_{\boldsymbol{\gamma}}(\boldsymbol{z}|\boldsymbol{x}) \,\|\, p_{\boldsymbol{\gamma}}(\boldsymbol{z})\big) = -\log P(\boldsymbol{x}) \tag{16}$$

The second limit can be observed as follows.

$$\begin{aligned} D\big(p_{\boldsymbol{\gamma}}(\boldsymbol{z}|\boldsymbol{x}) \,\|\, p_{\boldsymbol{\gamma}}(\boldsymbol{z})\big) &= \mathbb{E}_{p_{\boldsymbol{\gamma}}(\boldsymbol{z}|\boldsymbol{x})}\left[\log \frac{p(\boldsymbol{z}|\boldsymbol{x})}{p(\boldsymbol{z})}\right] \\ &= \mathbb{E}_{p_{\boldsymbol{\gamma}}(\boldsymbol{z}|\boldsymbol{x})}\left[\log \frac{P(\boldsymbol{x}|\boldsymbol{z})}{P(\boldsymbol{x})}\right] \\ &= \mathbb{E}_{p_{\boldsymbol{\gamma}}(\boldsymbol{z}|\boldsymbol{x})}\left[\log P(\boldsymbol{x}|\boldsymbol{z})\right] - \log P(\boldsymbol{x}) \end{aligned} \tag{17}$$

Because $P(\boldsymbol{x})$ is discrete, $\lim_{\boldsymbol{\gamma}\to\infty} P(\mathbf{x} \mid \mathbf{z}) = 1$ almost surely.

Combining Equation (17) and Equation (14), we obtain the following.

$$-\log P(\boldsymbol{x}) = 1/2 \int_{C_{\boldsymbol{\gamma}}} d\bar{\boldsymbol{\gamma}} \cdot \mathbf{E}(\boldsymbol{x}, \bar{\boldsymbol{\gamma}}) - \mathbb{E}_{p_{\boldsymbol{\gamma}}(\boldsymbol{z}|\boldsymbol{x})}\left[\log P(\boldsymbol{x}|\boldsymbol{z})\right] \tag{18}$$

Where the curve $C_{\boldsymbol{\gamma}}$ is piecewise smooth, begins at $\boldsymbol{\gamma} = 0$ and ends at $\boldsymbol{\gamma}$. We can use this result to obtain a path that starts at zero and ends at t, and Equation (6) when we let the end of the contour go to infinity. ■

A.4 Equivalence of Conditional and Unconditional Dynamics

Consider the marginals generated conditioned on the data with the following noise channel.

$$p_{\boldsymbol{\gamma}}(\boldsymbol{z}) \equiv \sum_{\boldsymbol{x}} p_{\boldsymbol{\gamma}}(\boldsymbol{z}|\boldsymbol{x})P(\boldsymbol{x})$$

Where $p_{\boldsymbol{\gamma}}(\boldsymbol{z}|\boldsymbol{x})$ is a Gaussian noise channel, which could be defined as follows,

$$\boldsymbol{z}_i = \gamma_i\, \boldsymbol{x}_i + \sqrt{\gamma_i}\, \boldsymbol{\epsilon}_i, \qquad \boldsymbol{\epsilon} \sim \mathcal{N}(0, I).$$

This is equivalent to the conditional SDE dynamics,

$$d\boldsymbol{z}_i = \boldsymbol{x}_i \dot{\gamma}_i\, dt + \sqrt{\dot{\gamma}_i} dW_i, \;\; \boldsymbol{x} \sim P(\boldsymbol{x}), \tag{19}$$

as can be seen from direct integration. We can use the Fokker-Planck equation to understand the marginal dynamics of this process. Consider a particular contour, $\boldsymbol{\gamma}(t)$, as defined in the text, which goes from zero to infinity. At $\boldsymbol{\gamma}(0) = 0$, we have

$$p_{\boldsymbol{\gamma}(0)}(\boldsymbol{z}) = \delta(\boldsymbol{z}).$$

The Fokker-Planck equation gives the following for the conditional dynamics.

$$\frac{\partial}{\partial t} p_{\boldsymbol{\gamma}(t)}(\boldsymbol{z} \mid \boldsymbol{x}) = -\sum_i \nabla_{\boldsymbol{z}_i} \cdot (\dot{\gamma}_i\, \boldsymbol{x}\, p_{\boldsymbol{\gamma}}(\boldsymbol{z} \mid \boldsymbol{x})) + \sum_i \tfrac{1}{2}\dot{\gamma}_i \Delta_{\boldsymbol{z}_i}\, p_{\boldsymbol{\gamma}}(\boldsymbol{z} \mid \boldsymbol{x})$$

By marginalizing over $P(\boldsymbol{x})$, get the following.

$$\frac{\partial}{\partial t} p_{\boldsymbol{\gamma}(t)}(\boldsymbol{z}) = \sum_{\boldsymbol{x}} P(\boldsymbol{x}) \frac{\partial}{\partial t} p_{\boldsymbol{\gamma}(t)}(\boldsymbol{z} \mid \boldsymbol{x}) \tag{20}$$

$$= -\sum_i \nabla_{\boldsymbol{z}_i} \cdot \left(\dot{\gamma}_i \sum_{\boldsymbol{x}} P(\boldsymbol{x}) \boldsymbol{x}_i\, p_{\boldsymbol{\gamma}}(\boldsymbol{z} \mid \boldsymbol{x}) \right) + \sum_i \tfrac{1}{2}\dot{\gamma}_i \Delta_{\boldsymbol{z}_i}\, p_{\boldsymbol{\gamma}}(\boldsymbol{z}) \tag{21}$$

$$= -\sum_i \nabla_{\boldsymbol{z}_i} \cdot (\dot{\gamma}_i\, \hat{\boldsymbol{x}}_i(\boldsymbol{z})\, p_{\boldsymbol{\gamma}}(\boldsymbol{z})) + \sum_i \tfrac{1}{2}\dot{\gamma}_i \Delta_{\boldsymbol{z}_i}\, p_{\boldsymbol{\gamma}}(\boldsymbol{z}) \tag{22}$$

In the last line, we used the definition of the optimal denoiser in terms of the posterior mean. The diffusion term is identical in the conditional and marginal equations and is untouched by marginalization over $P(\boldsymbol{x})$; the equivalence rests on matching the drift terms.

The goal is to design an unconditional dynamics that obeys the same evolution of the marginal. The unconditional dynamics are defined as follows.

$$d\boldsymbol{z}_i = \hat{\boldsymbol{x}}_i(\boldsymbol{z}) \dot{\gamma}_i\, dt + \sqrt{\dot{\gamma}_i} dW_i, \tag{23}$$

Applying the Fokker-Planck equation to this expression directly matches Equation (22). ■

A.5 Reverse Localization

Revise this part.

The main text uses only the kernel. Here we also record its continuous-time form and one consequence for resampling.

Proof of the kernel. Write $\boldsymbol{z}_i(\gamma) = \gamma \boldsymbol{x}_i + \mathbf{b}_i(\gamma)$. For a Brownian motion and $0 \le a < b$, $\mathbf{b}_i(a) \mid \mathbf{b}_i(b) \sim \mathcal{N}\big(\frac{a}{b}\mathbf{b}_i(b), \frac{a(b-a)}{b}\boldsymbol{I}\big)$. Substituting $\mathbf{b}_i(b) = \boldsymbol{z}_i(b) - b\boldsymbol{x}_i$ gives mean $a\boldsymbol{x}_i + \frac{a}{b}(\boldsymbol{z}_i(b) - b\boldsymbol{x}_i) = \frac{a}{b}\boldsymbol{z}_i(b)$; the clean token cancels. As a check, $\mathbb{E}[\boldsymbol{z}_i(a) \mid \boldsymbol{x}] = a\boldsymbol{x}_i$ and $\mathrm{Var} = \frac{a^2}{b^2}b + \frac{a(b-a)}{b} = a$, which is the channel at SNR a. Because the kernel does not depend on $\boldsymbol{x}$, it acts term by term on the mixture $p_{\boldsymbol{\gamma}}(\boldsymbol{z}) = \sum_{\boldsymbol{x}} P(\boldsymbol{x}) p_{\boldsymbol{\gamma}}(\boldsymbol{z} \mid \boldsymbol{x})$ and so maps conditional and unconditional marginals alike. Letting $b \to \infty$ with $\boldsymbol{z}_i(b)/b \to \boldsymbol{x}_i$ gives $\mathcal{N}(a\boldsymbol{x}_i, a\boldsymbol{I})$.

Continuous-time form. Along a segment with $\dot{\gamma}_i < 0$, the bridge satisfies

$$d\boldsymbol{z}_i = \frac{\dot{\gamma}_i}{\gamma_i}\, \boldsymbol{z}_i\, du + \sqrt{-\dot{\gamma}_i}\, d\mathbf{w}_i.$$

The same drift follows from time reversal of Eq. 5. The marginal score is $\nabla_{\boldsymbol{z}_i} \log p_{\boldsymbol{\gamma}}(\boldsymbol{z}) = \hat{\boldsymbol{x}}_i(\boldsymbol{z}) - \boldsymbol{z}_i/\gamma_i$, so the reverse drift is $\hat{\boldsymbol{x}}_i(\boldsymbol{z}) - \nabla_{\boldsymbol{z}_i} \log p_{\boldsymbol{\gamma}}(\boldsymbol{z}) = \boldsymbol{z}_i/\gamma_i$ per unit of $d\gamma_i$. The denoiser cancels. The conditional SDE Eq. 4 gives the same drift, since $\nabla_{\boldsymbol{z}_i} \log p_{\boldsymbol{\gamma}}(\boldsymbol{z} \mid \boldsymbol{x}) = \boldsymbol{x}_i - \boldsymbol{z}_i/\gamma_i$. The drift contracts toward the origin at the time-varying rate $1/\gamma_i$, so this is a Brownian bridge to $\mathbf{0}$, not an Ornstein–Uhlenbeck process.

Proof of invariance (part 2). Start from $(\boldsymbol{x}, \boldsymbol{z}) \sim P(\boldsymbol{x})\, p_{\boldsymbol{\gamma}}(\boldsymbol{z} \mid \boldsymbol{x})$. Applying the kernel to the positions in S gives $(\boldsymbol{x}, \boldsymbol{z}') \sim P(\boldsymbol{x})\, p_{\boldsymbol{\gamma}'}(\boldsymbol{z}' \mid \boldsymbol{x})$ by part 1, since S does not depend on $(\boldsymbol{x}, \boldsymbol{z})$. Redrawing $\boldsymbol{x}' \sim P_{\boldsymbol{\gamma}'}(\boldsymbol{x} \mid \boldsymbol{z}')$ preserves this joint law, so $\boldsymbol{x}' \sim P$. This is a data-augmentation Gibbs step. When the positions outside S sit at the clean endpoint, the posterior reduces to $P(\boldsymbol{x}_S \mid \boldsymbol{x}_{-S}, \boldsymbol{z}'_S)$; for $|S| = 1$ it is one categorical draw, i.e. one network evaluation. The step interpolates between the identity ($a_i \to \gamma_i$) and a Gibbs update ($a_i = 0$). For $|S| > 1$, drawing positions independently from per-token marginals introduces the usual factorization error of parallel decoding.

Non-monotone paths and the likelihood identity. $D\big(p_{\boldsymbol{\gamma}}(\boldsymbol{z} \mid \boldsymbol{x}) \,\|\, p_{\boldsymbol{\gamma}}(\boldsymbol{z})\big)$ is a function of $\boldsymbol{\gamma}$, so Eq. 14 holds for any piecewise-smooth contour, including decreasing segments. Monotonicity is needed only to write the dynamics in the form Eq. 4–Eq. 5, where $\sqrt{\dot{\gamma}_i}$ must be real. Path-independence is a property of the Bayes-optimal denoiser and does not transfer to a trained network.

A.6 Autoregressive Contours as a Special Case

The continuous SNR path formulation also recovers the standard autoregressive factorization as a special contour choice. In particular, when the contour reveals one token at a time while treating previously revealed tokens as effectively clean and future tokens as maximally corrupted, the path-integral likelihood reduces to the standard chain rule. For completeness, we isolate this argument here, since the main text only needs the high-level consequence.

Conditional denoisers Replacing $P(\boldsymbol{x})$ above with $P(\boldsymbol{x}|\boldsymbol{y})$ gives the conditional denoiser, and we still get a result invariant to SNR.

$$\hat{\boldsymbol{x}}(\boldsymbol{z}, y, \boldsymbol{\gamma}) = \hat{\boldsymbol{x}}(\boldsymbol{z}, y) = \mathbb{E}_{P(\boldsymbol{x}|y)}[\boldsymbol{x} e^{\boldsymbol{x}\cdot\boldsymbol{z}}] / \mathbb{E}_{P(\boldsymbol{x}|y)}[e^{\boldsymbol{x}\cdot\boldsymbol{z}}]$$

We used the following result to show the connection between probability estimates using certain contours and autoregressive models.

$$\begin{aligned} -\log P(\boldsymbol{x}) &= \frac{1}{2}\int_{C_{AR}} \mathbf{E}(\boldsymbol{x}, \boldsymbol{\gamma}) \cdot d\boldsymbol{\gamma} = \frac{1}{2}\sum_{i=1}^{L}\int_{C_i} \mathbf{E}(\boldsymbol{x}, \boldsymbol{\gamma}) \cdot d\boldsymbol{\gamma} \\ &= \sum_{i=1}^{L} \frac{1}{2}\int_0^\infty d\gamma_i E_i(\boldsymbol{x}, (\infty, \dots, \gamma_i, 0, \dots)) = -\sum_{i=1}^{L} \log P(\boldsymbol{x}_i|\boldsymbol{x}_{<i}) \end{aligned}$$

To show this, we need to demonstrate the last equality explicitly.

$$-\log P(\boldsymbol{x}_i \mid \boldsymbol{x}_{<i}) = \frac{1}{2}\int_0^\infty d\gamma_i \; E_i\left(\boldsymbol{x}, \boldsymbol{\gamma} = (\infty, \dots, \gamma_i, 0, \dots)\right) \tag{24}$$

First, we rewrite the left hand side.

$$\begin{aligned} \frac{1}{2}\int_0^\infty d\gamma_i \; E_i\left(\boldsymbol{x}, \boldsymbol{\gamma} = (\infty, \dots, \gamma_i, 0, \dots)\right) &= \frac{1}{2}\int_0^\infty d\gamma_i \; \mathbb{E}_{p_{\boldsymbol{\gamma}}(\boldsymbol{z}|\boldsymbol{x})}\left[\|\boldsymbol{x}_i - \hat{\boldsymbol{x}}_i(\boldsymbol{z})\|^2\right] \\ &= \frac{1}{2}\int_0^\infty d\gamma_i \; \mathbb{E}_{p_{\gamma_i}(\boldsymbol{z}_i|\boldsymbol{x}_i)}\left[\|\boldsymbol{x}_i - \hat{\boldsymbol{x}}_i(\boldsymbol{x}_{<i}, \boldsymbol{z}_i, 0, \dots)\|^2\right] \end{aligned}$$

In the last line, we note that for tokens at SNR 0, there is no information, so the optimal denoiser will be conditioned on a constant, zero. For tokens with infinite SNR, $\boldsymbol{z}_{<i}$ is equivalent in distribution to $\boldsymbol{x}_{<i}$, which we replace in the argument, with a slight abuse of notation.

Next, we see what happens when we try to write $P(\boldsymbol{x}_i|\boldsymbol{x}_{<i})$, in terms of the optimal *conditional denoiser* for denoising only $\boldsymbol{z}_i$, using Equation (6), where the contours simplify to 1-d integrals as we only have one SNR, γ_i. In that case we know we can write the probability in terms of the conditional denoiser, which we will call $\tilde{\boldsymbol{x}}(\boldsymbol{z}_i, \boldsymbol{x}_{<i})$ to distinguish from $\hat{\boldsymbol{x}}$.

$$-\log P(\boldsymbol{x}_i|\boldsymbol{x}_{<i}) = \frac{1}{2}\int_0^\infty d\gamma_i \; \mathbb{E}_{p_{\gamma_i}(\boldsymbol{z}_i|\boldsymbol{x}_i)}[\|\boldsymbol{x}_i - \tilde{\boldsymbol{x}}_i(\boldsymbol{z}_i, \boldsymbol{x}_{<i})\|^2]$$

We can see that this matches the expression above, except for the denoisers. These denoisers are conditioned on the same information, and have to be MMSE denoisers for the same denoising problem, and therefore they must be equivalent, and equality of Equation (24) must hold. **Conditional denoisers** Replacing $P(\boldsymbol{x})$ above with $P(\boldsymbol{x}|\boldsymbol{y})$ gives the conditional denoiser, and we still get a result invariant to SNR.

$$\hat{\boldsymbol{x}}(\boldsymbol{z}, y, \boldsymbol{\gamma}) = \hat{\boldsymbol{x}}(\boldsymbol{z}, y) = \mathbb{E}_{P(\boldsymbol{x}|y)}[\boldsymbol{x} e^{\boldsymbol{x}\cdot\boldsymbol{z}}] / \mathbb{E}_{P(\boldsymbol{x}|y)}[e^{\boldsymbol{x}\cdot\boldsymbol{z}}]$$

A.7 Exact ROAR Likelihood Decomposition and Estimator

This subsection derives the exact random-order conditional likelihood that motivates the ROAR-style endpoint branch in DSL training. The purpose is not to claim that the main experiments directly optimize this objective. Rather, it provides a second exact likelihood source, complementary to the continuous path-integral likelihood, and justifies including masking/reveal-like endpoint states in the training support.

In DSL, we point out that masked diffusion modeling is equivalent to paths in SNR space where we completely denoise one variable (going from zero SNR to infinite SNR), conditioned on zero noise (infinite SNR) for the already "unmasked" variables and infinite noise (zero SNR) for the "masked" variables. If we choose our SNR paths to be the ROAR paths, then we just need to model conditional probabilities of the next variable, given previously unmasked variables, similar to masked diffusion models. However, this idea doesn't exactly match up with practical implementations of masked diffusion models like [23] because they write their objective (Eq. 3 in [23] for example) in terms of probability of an unmasked set, conditioned on a masked set. Below, we derive a similar expression starting from the perspective that we are trying to model ROAR SNR paths. Unlike the result in [23], our derivation is path-based and makes no explicit mention of time.

Consider a multivariate probability distribution, $p(x_1, \dots, x_n)$. We can always decompose this distribution autoregressively as $p(x_1, \dots, x_n) = \prod_{i=1}^{n} p(x_i | x_{1:i})$. We could in principle use any ordering of the variables for this decomposition. Let π be a permutation of $\{1, \dots, n\}$, indexed with vector notation. For $n = 3$ we could have $\pi_{1:3} = (3, 1, 2)$, for example. For a particular permutation, π, we could write the AR decomposition as follows.

$$p(x_1, \dots, x_n) = \prod_{i=1}^{n} p(x_{\pi_i} | x_{\pi_{1:i-1}})$$

There are $n!$ permutations, so we can take the average over the set of all permutations, $\mathcal{S}_n$, and switch to log probabilities for convenience.

$$\log p(x_1, \dots, x_n) = \frac{1}{n!} \sum_{\pi \in \mathcal{S}_n} \sum_{i=1}^{n} \log p(x_{\pi_i} | x_{\pi_{1:i-1}}) \tag{25}$$

Now, note that this expression has many equivalent terms that can be grouped together. We are summing over a total of $n2^n$ log probability terms, but terms like $\log p(x_3|x_1, x_2) = \log p(x_3|x_2, x_1)$, i.e., the order of the conditioned information doesn't matter. This suggests that we should re-write this sum in terms of sets. Let $[n] = \{1, \dots, n\}$ and, for $A \subseteq [n]$, write $x_A = \{x_i : i \in A\}$. We will have terms of the form $\log p(x_i|x_A)$ (where i is not in the subset A) and we need to count how many such terms appear in the original sum. The number of permutations where we have the $|A|$ indices in A followed by i, followed by the remaining $(n - |A| - 1)$ indices in arbitrary order is $|A|!(n - |A| - 1)!$.

$$\begin{aligned}
\log p(x_1, \dots, x_n) &= \sum_{A \subseteq [n]} \frac{|A|!(n - |A| - 1)!}{n!} \sum_{i \in [n] \setminus A} \log p(x_i | x_A) \\
&= \sum_{k=0}^{n-1} \frac{n}{n} \sum_{A \subseteq [n] : |A| = k} \frac{k!(n-k-1)!}{n!} \frac{n-k}{n-k} \sum_{i \in [n] \setminus A} \log p(x_i | x_A) \\
&= n \sum_{k=0}^{n-1} \frac{1}{n} \sum_{A \subseteq [n] : |A| = k} \frac{k!(n-k)!}{n!} \sum_{i \in [n] \setminus A} \frac{1}{n-k} \log p(x_i | x_A)
\end{aligned} \tag{26}$$

In the second line, we rewrite the sum over sets to a sum over subsets of size k, then a sum of k. We also introduce some factors of 1 which are re-arranged in the third line. The goal is to re-write this expression in terms of expectations to get a convenient Monte Carlo estimator.

$$\frac{1}{n} \log p(x_1, \dots, x_n) = \underbrace{\sum_{k=0}^{n-1} \frac{1}{n}}_{\text{Uniform over k}} \underbrace{\sum_{A \subseteq [n] : |A| = k} \frac{1}{\binom{n}{k}}}_{\text{Uniform over sets } |A| = k} \underbrace{\sum_{i \in [n] \setminus A} \frac{1}{n-k}}_{\text{Uniform over } i \in [n] \setminus A} \log p(x_i | x_A) \tag{27}$$

At this point, we can recognize that each sum corresponds to a sample over a uniform distribution. The number of indices i in the set $[n] \setminus A$ is $n - k$, the number of subsets of $[n]$ of size k is $\binom{n}{k}$, the number of subset sizes is n.

$$\begin{aligned} \frac{1}{n} \log p(x_1, \ldots, x_n) &= \mathbb{E}_{P(k,A,i)}[\log p(x_i | x_A)] \\ P(k) &= \text{Unif}\{0, 1, \ldots, n-1\} \\ P(A|k) &= \text{Unif}\{A \subseteq [n] : |A| = k\} \\ P(i|A,k) &= \text{Unif}\{[n] \setminus A\} \end{aligned} \tag{28}$$

This gives us a concrete procedure for obtaining an unbiased estimator of the log likelihood that only requires evaluating probabilities of marginals, conditioned on sets of variables. This is exactly what our denoiser is trained to do. For discrete random variables, the inner expectation is just the mean cross entropy per variable of unmasked variables conditioned on masked ones, as seen in the LLADA estimator. We construct a Monte Carlo estimate by sampling over subsets of A. Training with this objective is simple as each Monte Carlo sample is a simple cross entropy. We summarize the procedure in Alg. 3.

Algorithm 3 ROAR NLL Estimator

Require: Sample, x. Model giving marginal probabilities, $p(x_i|x_A)$ for any subset of unmasked tokens, A.

1: Sample k and set A of size k uniformly
2: Average cross entropy per variable of masked tokens conditioned on unmasked token set, A
3: **return** NLL estimate (bits per token)

A.8 Cross-entropy as a surrogate for the path-integral MSE

This appendix justifies the choice of token-level cross-entropy on the continuous-SNR branch of DSL training. The path-integral identity Eq. 6 expresses $-\log P(\boldsymbol{x})$ as a contour integral of the squared embedding error

$$E_i(\boldsymbol{x}, \gamma) \;=\; \mathbb{E}_{p_\gamma(\boldsymbol{z}|\boldsymbol{x})}\big[\|\boldsymbol{x}_i - \hat{\boldsymbol{x}}_i(\boldsymbol{z})\|_2^2\big],$$

where $\hat{\boldsymbol{x}}_i = \mathbb{E}[\boldsymbol{x}_i \mid \boldsymbol{z}]$ is the Bayes-optimal denoiser. A direct continuous-NLL training scheme would parameterize a denoiser $\hat{\boldsymbol{x}}_{i,\theta}$ in embedding space and minimize this MSE. We argue that DSL should instead minimize a token-level cross-entropy objective that has the same Bayes-optimal solution but avoids three concrete pathologies of embedding-space MSE.

Same Bayes-optimal solution. Under the unit-sphere constraint, the Bayes denoiser admits the closed form

$$\hat{\boldsymbol{x}}(\boldsymbol{z}) \;=\; \mathbb{E}_{P(\boldsymbol{x})}\big[\boldsymbol{x}\, e^{\boldsymbol{x}\cdot\boldsymbol{z}}\big] \,/\, \mathbb{E}_{P(\boldsymbol{x})}\big[e^{\boldsymbol{x}\cdot\boldsymbol{z}}\big] \;=\; \sum_{v \in \mathcal{V}} p(\boldsymbol{x}{=}v \mid \boldsymbol{z})\, \boldsymbol{e}_v,$$

i.e., a vocabulary-weighted average of unit-norm embeddings under the induced token posterior $p(\boldsymbol{x}{=}v \mid \boldsymbol{z})$. Embedding-space MSE is minimized when $\hat{\boldsymbol{x}}_\theta(\boldsymbol{z})$ matches this posterior mean. Token-level cross-entropy on $p_\theta(\boldsymbol{x}{=}v \mid \boldsymbol{z})$ is minimized when $p_\theta(\cdot \mid \boldsymbol{z}) = p(\cdot \mid \boldsymbol{z})$. Both objectives therefore identify the same Bayes-optimal solution: at the global minimum, the CE-trained model recovers $p(\cdot \mid \boldsymbol{z})$, and reading off its posterior mean recovers $\hat{\boldsymbol{x}}(\boldsymbol{z})$. We use cross-entropy because, away from the global minimum, embedding-space MSE has structurally worse optimization geometry, as we explain next.

Embedding collapse under jointly learned embeddings. [8] observe that direct ℓ_2 regression in embedding space is ill-posed when the embedding matrix W_{emb} is jointly trained with the denoiser: setting $W_{\text{emb}} \to 0$ together with $\hat{\boldsymbol{x}}_\theta \to 0$ achieves zero loss for any noisy input $\boldsymbol{z}$, yielding a degenerate solution. Prior continuous diffusion language models address this only with handcrafted embeddings or auxiliary regularizers [19, 4]. DSL constrains every clean embedding to lie on the unit sphere ($\|\boldsymbol{x}_i\|_2 = 1$), which already rules out exact collapse, but the corresponding low-rank near-collapsed configurations remain reachable by gradient descent on embedding-space MSE.

Cross-entropy on a softmax over fixed unit-norm embeddings does not exhibit this pathology: the cross-entropy minimum requires the predicted distribution to actually concentrate on the correct token, not merely shrink the predicted embedding.

Low-SNR memorization burden. CDCD [4] first trained a continuous language diffusion model with cross-entropy on a softmax whose expectation over (normalized) embeddings gives the denoiser (score interpolation). Plaid [8] also note that a denoiser trained directly with embedding MSE must, at low noise, output the clean embedding vectors to high precision—effectively memorizing W_{emb} inside the network parameters. They mitigate this by reparameterizing the denoiser as a softmax over token logits, then averaging embeddings weighted by that softmax, an architectural change motivated by exactly this observation. The DSL converter Eq. 11 adopts the same architectural shape, and CE training is the natural objective for the resulting categorical head: the network's job is to assign probability mass over $\mathcal{V}$, not to reconstruct embedding coordinates.

Mean-seeking shortcuts on multi-modal posteriors. At intermediate SNR, the token posterior $p(\cdot \mid \boldsymbol{z})$ is generically multi-modal: several tokens have non-trivial probability. The MMSE estimator $\hat{\boldsymbol{x}} = \sum_v p(v \mid \boldsymbol{z})\boldsymbol{e}_v$ is the posterior mean—an average of the candidate embeddings—which under unit-sphere geometry is itself non-unit-norm and lies in the interior of the embedding simplex. A finite-capacity denoiser trained on embedding MSE can approach this mean by outputting a generic interior vector that does not commit to any particular token, lowering raw MSE without producing useful posterior structure. The cross-entropy objective penalizes such non-committal outputs because they correspond to entropic predicted distributions; the optimizer is pushed toward distributions that match the actual posterior shape.

In summary, cross-entropy and embedding MSE share the same Bayes-optimal solution under Eq. 6, but cross-entropy is preferred for three operational reasons rooted in prior work [8]: it eliminates joint-embedding collapse, eliminates the low-SNR memorization burden, and avoids mean-seeking shortcuts on multi-modal posteriors. This is what makes cross-entropy a practical surrogate for the path-integral MSE on the continuous-SNR branch of DSL training.

A.9 Posterior-State Matching and Inference-Time Error Correction

A useful way to understand why DSL can open a *correction window* is through a minimal two-bit manifold example. Consider sequences $(x_1, x_2) \in \{0,1\}^2$ whose data distribution is supported only on

$$\mathcal{M} = \{(0,0),(1,1)\}, \qquad P\big((0,0)\big) = P\big((1,1)\big) = 1/2. \tag{29}$$

Thus, inconsistent states such as $(0,1)$ or $(1,0)$ are off-manifold: they may arise as intermediate draft errors, but they are not valid solutions.

Embed the two symbols on the unit sphere by

$$\mathrm{enc}(0) = -1, \qquad \mathrm{enc}(1) = +1, \tag{30}$$

so the two valid clean sequence embeddings are

$$\boldsymbol{x}^{(-)} := (-1,-1), \qquad \boldsymbol{x}^{(+)} := (+1,+1). \tag{31}$$

Under the DSL localization channel, a corrupted state is

$$\boldsymbol{z} = \gamma \boldsymbol{x} + \sqrt{\gamma}\,\boldsymbol{\epsilon}, \qquad \boldsymbol{\epsilon} \sim \mathcal{N}(0, \boldsymbol{I}_2), \qquad \boldsymbol{x} \in \{\boldsymbol{x}^{(-)}, \boldsymbol{x}^{(+)}\}. \tag{32}$$

Because the prior is uniform and the clean embeddings are unit norm, the posterior over the two valid hypotheses is

$$P\big(\boldsymbol{x} = \boldsymbol{x}^{(+)} \mid \boldsymbol{z}\big) = \frac{\exp(\boldsymbol{z}^\top \boldsymbol{x}^{(+)})}{\exp(\boldsymbol{z}^\top \boldsymbol{x}^{(+)}) + \exp(\boldsymbol{z}^\top \boldsymbol{x}^{(-)})} = \frac{\exp(z_1+z_2)}{\exp(z_1+z_2) + \exp(-(z_1+z_2))}, \tag{33}$$

$$P\big(\boldsymbol{x} = \boldsymbol{x}^{(-)} \mid \boldsymbol{z}\big) = \frac{\exp(\boldsymbol{z}^\top \boldsymbol{x}^{(-)})}{\exp(\boldsymbol{z}^\top \boldsymbol{x}^{(+)}) + \exp(\boldsymbol{z}^\top \boldsymbol{x}^{(-)})} = \frac{\exp(-(z_1+z_2))}{\exp(z_1+z_2) + \exp(-(z_1+z_2))}. \tag{34}$$

Equivalently,

$$P\big(\boldsymbol{x} = \boldsymbol{x}^{(+)} \mid \boldsymbol{z}\big) = \sigma\big(2(z_1+z_2)\big), \qquad P\big(\boldsymbol{x} = \boldsymbol{x}^{(-)} \mid \boldsymbol{z}\big) = 1 - \sigma\big(2(z_1+z_2)\big). \tag{35}$$

Hence the decision boundary is not determined token-by-token, but by the *joint* statistic

$$z_1 + z_2 = 0. \tag{36}$$

The posterior-mean denoiser is

$$\begin{aligned}\hat{\boldsymbol{x}}(\boldsymbol{z}) = \mathbb{E}[\boldsymbol{x} \mid \boldsymbol{z}] &= P\big(\boldsymbol{x} = \boldsymbol{x}^{(+)} \mid \boldsymbol{z}\big)\boldsymbol{x}^{(+)} + P\big(\boldsymbol{x} = \boldsymbol{x}^{(-)} \mid \boldsymbol{z}\big)\boldsymbol{x}^{(-)} \\ &= \Big(P\big(\boldsymbol{x} = \boldsymbol{x}^{(+)} \mid \boldsymbol{z}\big) - P\big(\boldsymbol{x} = \boldsymbol{x}^{(-)} \mid \boldsymbol{z}\big)\Big)(1,1) \\ &= \tanh(z_1 + z_2)\,(1,1). \end{aligned} \tag{37}$$

Equation 37 is the key point: the denoising field points toward the correct manifold as long as the corrupted state remains on the correct side of the joint decision boundary.

To make this concrete, suppose the ground-truth sequence is $(1,1)$, but an intermediate draft contains a local inconsistency resembling $(1,0)$. In continuous state space, this corresponds to a region such as

$$\mathcal{R}_{11}^{\mathrm{corr}} := \{\boldsymbol{z} \in \mathbb{R}^2 :\ z_1 > 0,\ z_2 < 0,\ z_1 + z_2 > 0\}. \tag{38}$$

Inside $\mathcal{R}_{11}^{\mathrm{corr}}$, the second coordinate looks locally wrong, but the joint posterior still favors $\boldsymbol{x}^{(+)} = (1,1)$ because $z_1 + z_2 > 0$. Therefore, the denoising update in Eq. 37 continues to point toward $(1,1)$ rather than amplifying the local error. This is the *correction window*: an off-manifold draft can still be revised if its induced posterior state remains inside the attraction region of the correct solution.

This example also clarifies what DSL does *not* claim. DSL does not guarantee correction for every erroneous draft. If the state crosses the wrong side of the boundary, i.e.

$$z_1 + z_2 < 0, \tag{39}$$

then the posterior collapses toward $\boldsymbol{x}^{(-)} = (0,0)$ and the denoising field points to the wrong manifold. Thus, posterior-state matching opens a correction window, but does not guarantee that inference remains inside it.

This toy case explains why mixed-corruption DSL training can help. Training does not need to reproduce the exact symbolic error $(1,0)$ as a raw state. Instead, it is sufficient that the corruption distribution exposes the model to continuous states whose induced posterior geometry overlaps with revisable regions such as Eq. 38. In that case, some imperfect self-generated drafts at inference time are less out-of-distribution, and the model is more likely to produce a denoising field that still points back toward a valid solution manifold.

By contrast, standard autoregressive decoding and plain absorbing masked diffusion do not naturally realize this mechanism. For autoregressive decoding,

$$\hat{x}_1 = \arg\max_{x \in \{0,1\}} p_\theta(x_1 = x), \qquad \hat{x}_2 = \arg\max_{x \in \{0,1\}} p_\theta(x_2 = x \mid \hat{x}_1), \tag{40}$$

so once $\hat{x}_1$ is emitted, later steps condition on a hard prefix and do not revise it. Likewise, in plain absorbing masked diffusion, visible tokens are effectively frozen once unmasked, so a visible-but-wrong token is not naturally moved back toward the correct manifold unless an explicit remasking mechanism is introduced. DSL is therefore complementary to remasking-based samplers: remasking reopens a visible token for revision, while DSL increases the chance that the reopened state still lies in a posterior region where the denoising field points back toward the correct solution.

The lesson of this example is that correction is not explained by "many possible contexts" alone. What matters is whether these possibilities are organized into a posterior geometry whose mass still overlaps the correct solution and whose denoising field still points toward it. DSL helps precisely by broadening the posterior states seen during training, thereby making some refinement-time erroneous drafts less out-of-distribution and more revisable.

A.10 Confidence Geometry: Norm–Direction Decomposition and Bounded Entropy

With the posterior written as a function of the realized state alone, its geometry becomes transparent. Throughout this subsection we use the unit-sphere token geometry of DSL: each clean token embedding satisfies $\|\boldsymbol{e}_v\|_2 = 1$ for $v \in \mathcal{V}$.

Proposition A.1 (Norm–direction decomposition) *For any nonzero state $\boldsymbol{z}$ and any unit-norm token embedding $\boldsymbol{e}_i$, the score of token i induced by $\boldsymbol{z}$ satisfies*

$$s_i(\boldsymbol{z}) := \langle \boldsymbol{z}, \boldsymbol{e}_i \rangle = \|\boldsymbol{z}\|_2 \cos\theta_i, \tag{41}$$

where θ_i is the angle between $\boldsymbol{z}$ and $\boldsymbol{e}_i$. At $\boldsymbol{z} = 0$, all scores are zero and the posterior reduces to the base measure.

Proof A.1 *For $\boldsymbol{z} \neq 0$, the cosine identity gives*

$$\cos\theta_i = \frac{\langle \boldsymbol{z}, \boldsymbol{e}_i \rangle}{\|\boldsymbol{z}\|_2 \|\boldsymbol{e}_i\|_2}.$$

Since $\|\boldsymbol{e}_i\|_2 = 1$, this implies $\langle \boldsymbol{z}, \boldsymbol{e}_i \rangle = \|\boldsymbol{z}\|_2 \cos\theta_i$.

Proposition A.1 separates two roles that are entangled in unconstrained logits: the angular term determines *which token* is preferred, whereas the radial term determines *how concentrated* the posterior is around that preference. This yields the discrete hyperspherical posterior

$$p(x = i \mid \boldsymbol{z}) = \frac{\pi_i \exp\big(\|\boldsymbol{z}\|_2 \cos\theta_i\big)}{\sum_{j \in \mathcal{V}} \pi_j \exp\big(\|\boldsymbol{z}\|_2 \cos\theta_j\big)}. \tag{42}$$

Entropy envelope at bounded radius. For any finite radius cap $\beta < \infty$, write $\boldsymbol{z} = \beta \boldsymbol{u}$ with $\|\boldsymbol{u}\|_2 \leq 1$. The corresponding bounded-radius posterior family is

$$q_\beta(v \mid \boldsymbol{u}) = \frac{\pi_v \exp\big(\beta \langle \boldsymbol{u}, \boldsymbol{e}_v \rangle\big)}{\sum_{j \in \mathcal{V}} \pi_j \exp\big(\beta \langle \boldsymbol{u}, \boldsymbol{e}_j \rangle\big)}, \qquad \|\boldsymbol{u}\|_2 \leq 1,\ \|\boldsymbol{e}_v\|_2 = 1. \tag{43}$$

Since $\langle \boldsymbol{u}, \boldsymbol{e}_v \rangle \in [-1, 1]$, the logits can only tilt away from the base measure π within a finite band controlled by β. Hence the achievable posterior entropy over this bounded-radius family is confined to the interval

$$H_{\min}(\beta, \pi) \ \leq\ H(q_\beta(\cdot \mid \boldsymbol{u})) \ \leq\ H_{\max}(\beta, \pi), \tag{44}$$

where

$$H_{\min}(\beta, \pi) := \min_{\|\boldsymbol{u}\|_2 \leq 1} H(q_\beta(\cdot \mid \boldsymbol{u})), \qquad H_{\max}(\beta, \pi) := \max_{\|\boldsymbol{u}\|_2 \leq 1} H(q_\beta(\cdot \mid \boldsymbol{u})). \tag{45}$$

Thus, within a finite-SNR or bounded-converter regime, β controls how sharp the posterior may become, while π sets the baseline non-uniformity of token probabilities. Without such a finite radius cap, the Bayes posterior can become arbitrarily sharp as $\|\boldsymbol{z}\|_2$ grows.

Smoothness as a supporting property. The same hyperspherical geometry also induces a smooth Bayes posterior-mean map

$$\boldsymbol{m}(\boldsymbol{z}) := \mathbb{E}[\boldsymbol{e}_x \mid \boldsymbol{z}]. \tag{46}$$

When the clean-token distribution is supported on unit-norm embeddings, this Bayes posterior-mean map is 1-Lipschitz:

$$\|\boldsymbol{m}(\boldsymbol{z}_1) - \boldsymbol{m}(\boldsymbol{z}_2)\|_2 \leq \|\boldsymbol{z}_1 - \boldsymbol{z}_2\|_2. \tag{47}$$

We treat this as a supporting geometric property: it suggests that DSL refinement is well-behaved by construction, but in practice the decisive mechanism is whether confidence remains informative on refinement-time draft errors.

A.11 Lipschitz Continuity of the Induced Posterior-Mean Map

The main text treats smoothness as a supporting geometric property rather than a core mechanism. We therefore defer the full proof here. The result shows that the posterior-mean refinement map induced by DSL is 1-Lipschitz under unit-sphere token geometry.

Lemma A.1 *Let the induced posterior-mean map (MMSE denoiser) of the unconditional SDE be defined by*

$$\hat{\boldsymbol{x}}(\boldsymbol{z}) = \mathbb{E}_{P(\boldsymbol{x})}[\boldsymbol{x} e^{\boldsymbol{x} \cdot \boldsymbol{z}}] / \mathbb{E}_{P(\boldsymbol{x})}[e^{\boldsymbol{x} \cdot \boldsymbol{z}}] \qquad \boldsymbol{z} \in \mathbb{R}^d,$$

where the data distribution $p(\boldsymbol{x})$ is supported on the unit sphere $\|\boldsymbol{x}\| = 1$. Then $\hat{\boldsymbol{x}}(\cdot)$ is 1-Lipschitz; that is,

$$\forall \boldsymbol{z}_1, \boldsymbol{z}_2 \in \mathbb{R}^d : \quad \|\hat{\boldsymbol{x}}(\boldsymbol{z}_1) - \hat{\boldsymbol{x}}(\boldsymbol{z}_2)\| \ \leq\ \|\boldsymbol{z}_1 - \boldsymbol{z}_2\|.$$

Define the log–partition function

$$\phi(\boldsymbol{z}) \;=\; \log \mathbb{E}_{p(\boldsymbol{x})}\big[e^{\boldsymbol{x}\cdot\boldsymbol{z}}\big],$$

so that $\hat{\boldsymbol{x}}(\boldsymbol{z}) = \nabla_{\boldsymbol{z}}\phi(\boldsymbol{z})$.

Because $e^{\boldsymbol{x}\cdot\boldsymbol{z}}$ is convex in $\boldsymbol{z}$ and the expectation preserves convexity, ϕ is convex. Its Hessian equals the covariance of $\boldsymbol{x}$ under the Gibbs posterior $p(\boldsymbol{x} \mid \boldsymbol{z}) \propto p(\boldsymbol{x})e^{\boldsymbol{x}\cdot\boldsymbol{z}}$:

$$H(\boldsymbol{z}) \;=\; \nabla^2_{\boldsymbol{z}}\phi(\boldsymbol{z}) \;=\; \mathrm{Cov}_{p(\boldsymbol{x}|\boldsymbol{z})}\big(\boldsymbol{x}\big).$$

For any unit vector $v \in \mathbb{R}^d$,

$$v^\top H(\boldsymbol{z})\, v \;=\; \mathbb{E}_{p(\boldsymbol{x}|\boldsymbol{z})}\big[(v\cdot\boldsymbol{x})^2\big] - \Big(\mathbb{E}_{p(\boldsymbol{x}|\boldsymbol{z})}[v\cdot\boldsymbol{x}]\Big)^2 \;\le\; \mathbb{E}_{p(\boldsymbol{x}|\boldsymbol{z})}\big[(v\cdot\boldsymbol{x})^2\big] \;\le\; \|v\|^2 \;=\; 1,$$

where the last inequality uses $\|\boldsymbol{x}\| = 1$, therefore, spectral norm satisfies $\|H(\boldsymbol{z})\| \le 1$.

Since $\nabla^2_{\boldsymbol{z}}\phi$ is bounded by 1 everywhere, the gradient $\nabla_{\boldsymbol{z}}\phi = \hat{\boldsymbol{x}}$ is 1-Lipschitz:

$$\|\hat{\boldsymbol{x}}(\boldsymbol{z}_1) - \hat{\boldsymbol{x}}(\boldsymbol{z}_2)\| \;\le\; \int_0^1 \big\|H\big(\boldsymbol{z}_2 + t(\boldsymbol{z}_1 - \boldsymbol{z}_2)\big)\big\|\, \|\boldsymbol{z}_1 - \boldsymbol{z}_2\|\, dt \;\le\; \|\boldsymbol{z}_1 - \boldsymbol{z}_2\|.$$

This proves non-expansiveness of the Bayes posterior-mean map. Empirical estimates with K < 1 are consistent with this bound, but are stronger than what the lemma guarantees. ■

B Continuous States Are Not Enough: Posterior Coordinates vs. Timestep-Conditioned Denoising

Prior continuous diffusion LMs already use finite-SNR states between noise and clean embeddings [19, 4, 8, 36]. The difference in DSL is not continuity itself. It is how the noisy token is parameterized and what the denoiser needs to read it.

Generic Gaussian corruption. Consider

$$\boldsymbol{y}_\tau = a_\tau \boldsymbol{x} + \sigma_\tau \boldsymbol{\epsilon}, \qquad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}), \tag{48}$$

with nominal time τ. Let $\boldsymbol{x}_v = \mathrm{enc}(v)$ and let π_v be the context-dependent prior of token v. The Bayes posterior is

$$P(\boldsymbol{x} = \boldsymbol{x}_v \mid \boldsymbol{y}_\tau, \tau) \;\propto\; \pi_v \exp\Big(\tfrac{a_\tau}{\sigma_\tau^2}\langle \boldsymbol{y}_\tau, \boldsymbol{x}_v\rangle - \tfrac{a_\tau^2}{2\sigma_\tau^2}\|\boldsymbol{x}_v\|^2\Big). \tag{49}$$

Time enters in two places: the norm term and the scale a_τ/σ_τ^2. Normalizing the embeddings removes the first, but not the second. The natural coordinate is

$$\tilde{\boldsymbol{z}}_\tau = \tfrac{a_\tau}{\sigma_\tau^2}\, \boldsymbol{y}_\tau, \tag{50}$$

so a denoiser that reads $\boldsymbol{y}_\tau$ must also be told τ:

$$f_\theta(\boldsymbol{y}_\tau, \tau). \tag{51}$$

This is a representational point, not a probabilistic one: the pair $(\boldsymbol{y}_\tau, \tau)$ carries the full posterior.

DSL uses the natural coordinate. DSL sets

$$\boldsymbol{z}_i = \gamma_i \boldsymbol{x}_i + \sqrt{\gamma_i}\, \boldsymbol{\epsilon}_i, \qquad \frac{a_{\gamma_i}}{\sigma^2_{\gamma_i}} = \frac{\gamma_i}{\gamma_i} = 1, \tag{52}$$

so the noisy token is already the natural coordinate. With $\|\boldsymbol{x}_v\| = 1$ the norm term also cancels:

$$P(\boldsymbol{x}_i = \boldsymbol{x}_v \mid \boldsymbol{z}, \boldsymbol{\gamma}) = P(\boldsymbol{x}_i = \boldsymbol{x}_v \mid \boldsymbol{z}) \;\propto\; \pi_v \exp\langle \boldsymbol{z}_i, \boldsymbol{x}_v\rangle, \tag{53}$$

$$\hat{\boldsymbol{x}}(\boldsymbol{z}, \boldsymbol{\gamma}) = \mathbb{E}_{P(\boldsymbol{x})}[\boldsymbol{x} e^{\boldsymbol{x}\cdot\boldsymbol{z}}] / \mathbb{E}_{P(\boldsymbol{x})}[e^{\boldsymbol{x}\cdot\boldsymbol{z}}] = \hat{\boldsymbol{x}}(\boldsymbol{z}). \tag{54}$$

DSL therefore learns one map,

$$f_\theta(\boldsymbol{z}) \approx P(\boldsymbol{x}_i \mid \boldsymbol{z}), \tag{55}$$

and different SNRs are different regions of the same posterior space. The corruption level is still visible: $\|\boldsymbol{z}_i\|$ plays the role that time plays elsewhere. It is not a missing input, because $\boldsymbol{z}$ is a sufficient statistic for $\boldsymbol{x}$.

Table 5: **Prior continuous diffusion LMs vs. DSL.** Both use finite-SNR states; they differ in how those states are parameterized and read by the denoiser.

	Prior continuous diffusion LMs	**DSL**
Posterior depends on	The pair $(\boldsymbol{y}_\tau, \tau)$.	The noisy token $\boldsymbol{z}$ alone.
Denoiser	$f_\theta(\boldsymbol{y}_\tau, \tau)$; noise levels are separate, time-indexed tasks.	$f_\theta(\boldsymbol{z})$; noise levels are regions of one posterior space.
Token geometry	Often unnormalized; CDCD normalizes, but the scale a_τ/σ_τ^2 still requires τ.	Unit sphere and $a/\sigma^2 = 1$ together remove the SNR exactly.
Confidence	Learned; read relative to τ.	$\|\boldsymbol{z}_i\|$ sets concentration; direction sets the token (Eq. Eq. 12).
Training support	One global noise schedule.	Mixed: finite-SNR states and endpoint mask/reveal states.
Network input	Noisy embedding plus time.	Bounded mixture of tokens (converter).
Sampling paths	One global schedule.	Per-token SNR paths; ROAR, masked diffusion and remasking are special cases.

Relation to specific models. CDCD normalizes embeddings and trains with cross-entropy [4]; under VE noise its posterior depends on $\boldsymbol{y}/\sigma^2$, so σ remains an input. Bayesian Flow Networks update a categorical belief from Gaussian evidence and feed the network that belief together with t [**?**]. RDLM places the noisy state on the hypersphere and builds diffusion bridges there, with a time-conditioned network [**?**]. DSL keeps the noisy token in $\mathbb{R}^d$ and places only the clean embeddings on the sphere.

Noise-level-free denoising. (author?) [37] observe that many image diffusion models work without noise conditioning. In DSL this holds exactly: the Bayes denoiser has no SNR argument (Eq. Eq. 54), and constant-norm embeddings are required for this (Appendix A.2).

The converter does not discard information. The converter output in Eq. 11 is the gradient of a log-sum-exp in $\boldsymbol{z}_i$. When the embeddings span $\mathbb{R}^d$, this function is strictly convex, so the map $\boldsymbol{z}_i \mapsto \boldsymbol{m}_i^{\text{conv}}$ is injective. The converter is therefore a reparameterization of the noisy token, not a normalization trick. It presents the posterior state to the Transformer as a bounded mixture of token embeddings ($\|\boldsymbol{m}_i^{\text{conv}}\| \leq 1$), in the same geometry as ordinary token inputs.

Why not add a per-token time input? A global-time denoiser cannot represent states in which positions have different SNRs. A denoiser with a per-token γ_i input can, but it needs position-wise conditioning in place of the sequence-level adaLN of DiT. Pretrained masked-diffusion weights then no longer map onto the backbone. DSL removes the need for this input, keeps the timestep module fixed at $t = 0$, and stays compatible with the MDLM checkpoint.

Comparison summary. Table 5 summarizes the differences.

What this comparison does not claim. We do not claim that prior continuous diffusion LMs lack posterior meaning or are numerically unstable. Any Gaussian corruption defines a posterior $P(\boldsymbol{x} \mid \boldsymbol{y}_\tau, \tau)$. The claim is narrower: DSL chooses a coordinate in which the noisy token alone determines the posterior, and builds the training support and the network interface around it.

C Training and Sampling Details

This appendix documents the practical choices used in the main experiments. The main text focuses on the method-level design: mixed-support CE posterior matching and a posterior-view converter. Here we give the concrete training and decoding settings needed for reproducibility.

C.1 OWT Finetuning Settings

We exactly follow MDLM's [28] training configurations.

Data and sequence length. We fine-tune on OpenWebText using the `openwebtext-split` dataset. All training samples are truncated/padded to a fixed sequence length $L = 1024$ tokens.

Backbone and parameterization. We use the DiT backbone in official MDLM GitHub repository [28] with token embedding dimension 64. Training is conducted in full precision (FP32).

Optimization and batching. We train for a maximum of 100,000 optimizer steps with no learning-rate warmup (`num_warmup_steps=0`). Everything else in training setting is the same as MDLM training setting.

Compute resources. All experiments were run on NVIDIA GPUs. Text8 experiments were run on 2 H100s, and sampling/evaluation took approximately 1 hour. OpenWebText training and sampling evaluation used 2 H100s, with 5000 generated samples per configuration. The hybrid-sampler sweeps required approximately 2 hours in total, including preliminary hyperparameter searches. No additional large-scale pretraining was performed beyond fine-tuning/evaluating the checkpoints described above.

C.2 Mixed SNR Support Used in Training

This subsection specifies the practical mixed-support training distribution used in DSL. A subset of token positions is sampled from a ROAR-style endpoint branch, while the remainder is sampled from a continuous log-normal SNR branch. This construction is the practical instantiation of the two exact likelihood views in the main text.

We denote by γ the signal-to-noise ratio (SNR) used in the training corruption process. We use a *mixed* SNR path: with probability

$$p_{\mathrm{ROAR}} \;=\; \frac{1}{k} \;=\; 0.1, \tag{56}$$

we draw γ from the ROAR path sampler, where mask is represented by $\gamma = 0$, and clean token is approximated by $\gamma_{\text{max}} = 100$; with probability $1 - p_{\mathrm{ROAR}}$ we draw γ from a lognormal sampler. Concretely, for the lognormal continuous SNR branch we sample

$$\gamma \sim \mathrm{LogNormal}(\mu, \sigma^2), \qquad \mu = 1.65, \;\; \sigma = 0.9, \tag{57}$$

and there is no $\gamma_{\max}$ cut-off for this lognormal continuous training schedule. Throughout training, γ **denotes SNR and is distinct from the decoding progress variable τ used in sampling.**

We summarize DSL training in Algorithm 1. Each step samples a sentence from the data, draws a per-token SNR vector γ from one of two branches (ROAR with probability $1-\lambda$, continuous-path with probability λ), produces noisy embeddings $\boldsymbol{z}$, runs the converter and backbone to obtain token posteriors $p_\theta(\cdot \mid \boldsymbol{z})$, and updates parameters via cross-entropy against the clean tokens.

C.3 Masked Refinement with a DSL Checkpoint

The first decoding family used in the paper pairs a trained DSL checkpoint with a ReMDM-style masked-refinement schedule. Its role is to test the primary empirical claim of the paper: whether mixed-support DSL training improves refinement robustness and step-efficiency under a fixed masked decoder family. Algorithm 4 gives the high-level procedure, while all schedule hyperparameters are listed in Section C.4.

C.4 Sampling Settings on OWT

This section fully specifies the decoding schedules used in Section 5 and defines the rewrite-count statistics and diagnostics reported throughout the paper.

Common evaluation protocol (all rows in Table 1). Unless otherwise stated, all methods are evaluated with the same tokenizer (GPT-2 BPE), sequence length $L = 1024$, nucleus sampling (top-$p = 0.9$), sample count (5k), and MAUVE configuration (GPT-2 Large embeddings with $K = 500$ buckets) on the same held-out OWT split. GenPPL is computed as perplexity under the same GPT-2

Algorithm 4 DSL sampling with ReMDM sampler. Adapted from [40, Algorithm 1]; in DSL, the ReMDM mask state is represented by zero SNR.

Input: DSL denoiser $\boldsymbol{x}_\theta$, sampling steps T, noise schedule α_t, clean endpoint $\mathrm{SNR}_{\max}$, remask schedule σ_t.
Initialize: set every token embedding to the mask endpoint, i.e., $\mathrm{SNR}(\boldsymbol{z}_T) = 0$.
for $i = T, T-1, \ldots, 1$ **do**
  $t = i/T, \quad s = (i-1)/T$; set α_t, α_s.
  Set $\sigma_t \in [0, \sigma_t^{\max}]$, where $\sigma_t^{\max} = \min\{1, (1-\alpha_s)/\alpha_t\}$.
  Predict clean-token posteriors $\widehat{\boldsymbol{x}} = \boldsymbol{x}_\theta(\boldsymbol{z}_t)$.
  Form the ReMDM endpoint posterior $p_\theta(\boldsymbol{z}_s \mid \boldsymbol{z}_t) = q_\sigma^{\mathrm{DSL}}(\boldsymbol{z}_s \mid \boldsymbol{z}_t, \boldsymbol{x} = \widehat{\boldsymbol{x}})$, where ReMDM's mask atom is interpreted as $\mathrm{SNR} = 0$ and its clean-token atom as $\mathrm{SNR} = \mathrm{SNR}_{\max}$.
  Sample $\boldsymbol{z}_s \sim p_\theta(\boldsymbol{z}_s \mid \boldsymbol{z}_t)$.
  Realize sampled clean tokens at $\mathrm{SNR}_{\max}$; realize remasked tokens by resetting their SNR to 0.
  Set $\boldsymbol{z}_t \leftarrow \boldsymbol{z}_s$.
end for
Output: decoded tokens from the final clean-endpoint latents.

Large autoregressive oracle LM. Sentence entropy is computed as the average Shannon entropy of the token-ID histogram within each generated sequence (prior to text decoding). This matches the MDLM/ReMDM evaluation protocol and ensures baseline parity.

Time parameterization (matches logged t). We index a T-step decode by a *0-based* step counter $k = 0, \ldots, T-1$ and use normalized time

$$t_k \triangleq 1 - \frac{k}{T} \in \left[\frac{1}{T}, 1\right]. \tag{58}$$

Thus decoding progresses from $t_0 = 1$ (fully masked/noisy) to $t_{T-1} = 1/T$ (near-clean). This convention matches the sampler logs (e.g., for $T = 128$, the final logged $t \approx 0.0078 \approx 1/128$).

Mask ratio schedule and reveal fraction. Let $M_k \subseteq [L]$ denote the masked positions at step k (i.e., positions whose visible token is `[MASK]`). We log the *realized* masked ratio

$$r(t_k) \triangleq \frac{|M_k|}{L}. \tag{59}$$

We additionally report the *reveal fraction* (newly unmasked mass) per step,

$$\Delta r(t_k) \triangleq r(t_k) - r(t_{k+1}) \quad (k < T-1), \tag{60}$$

which indicates how aggressively the schedule transitions from global drafting (large Δr early) to local refinement (small Δr late).

ReMDM-loop and the loop window. Our main step-budgeted protocol uses the ReMDM-loop variant (for all T) with the official loop defaults:

$$t_{\mathrm{on}} = 0.55, \qquad t_{\mathrm{off}} = 0.05, \qquad \alpha_{\mathrm{loop}} = 0.9, \qquad \texttt{refresh_unmasked} = \texttt{true}. \tag{61}$$

Conceptually, decoding consists of (i) a standard MDLM-style reveal phase, (ii) a *loop phase* over the time window $[t_{\mathrm{on}}, t_{\mathrm{off}})$ with $t_{\mathrm{on}} > t_{\mathrm{off}}$, and (iii) a final reveal phase that anneals to the near-clean endpoint. During the loop phase, the noise level is held fixed at α_{loop} so that repeated self-correction operates under a stable context quality.

Capped remasking intensity (ReMDM-cap inside the loop). When remasking is active (i.e., within the loop window), ReMDM remasks a subset of currently unmasked tokens to avoid early lock-in. We use ReMDM's capped schedule:

$$q(t) \;=\; \min\left(1,\; \frac{\eta(t)}{1 - r(t)}\right), \qquad \eta(t) \;=\; \eta_{\mathrm{cap}} \frac{\alpha(t)}{1 - \alpha(t)}, \tag{62}$$

where $q(t)$ is the per-token remask probability among unmasked positions and $\alpha(t)$ is ReMDM's noise schedule. Outside the loop window, we set $q(t) = 0$ (no remasking).

Step budget T	128	256	512	1024
$\eta_{\text{cap}}(T)$	0.010	0.008	0.008	0.002
top-p nucleus	0.9			
Tokenizer / L	GPT-2 BPE / 1024			
MAUVE embed / K	GPT-2 Large / 500			
GenPPL oracle	GPT-2 Large			
SNR schedule	linear, $\texttt{snr_min} = 10^{-3}$, $\texttt{snr_max} = 100$, $\rho = 7$, $\gamma = 4$			
σ schedule	Karras, $\sigma_{\min} = 0.01$, $\sigma_{\max} = 1.0$, $\rho = 7$			
Loop window	$[t_{\text{on}}, t_{\text{off}}) = [0.55, 0.05)$			
α_{loop}	0.9			

Table 6: Decoding hyperparameters for OWT experiments under our step-budgeted ReMDM-loop protocol. All settings are identical across T except $\eta_{\text{cap}}(T)$.

T	$\eta_{\text{cap}}(T)$	mean rewrites-per-token ($\mathbb{E}[R_i]$)
128	0.010	0.579834
256	0.008	0.924500
512	0.008	1.610600
1024	0.002	1.848940

Table 7: Rewrite statistics for the step-budgeted ReMDM-loop protocol (computed from Eq. 64, averaged over 5k samples).

Step-budget-aware choice of $\eta_{\text{cap}}(T)$ (as used in our logs). For the step-budgeted schedule used in Table 1 (DSL-FT + ReMDM-loop), we set

$$\eta_{\text{cap}}(128) = 0.010, \quad \eta_{\text{cap}}(256) = 0.008, \quad \eta_{\text{cap}}(512) = 0.007, \quad \eta_{\text{cap}}(1024) = 0.004, \tag{63}$$

and keep all other ReMDM-loop hyperparameters identical to the official setting. This is a minimal, reproducible adaptation that controls rewrite counts while front-loading corrections.

Rewrite-count statistics. Let $x^{(k)} \in \mathcal{V}^L$ be the sampled sequence after step k (with `[MASK]` treated as a valid symbol during decoding). We define the per-position rewrite count

$$R_i \triangleq \sum_{k=1}^{T-1} \mathbf{1}\left[x_i^{(k)} \neq x_i^{(k-1)}\right], \qquad i \in [L], \tag{64}$$

and report the mean rewrite count $\frac{1}{L}\sum_{i=1}^{L} R_i$ (equivalently "Rewrites-per-token" in our logs), averaged over the 5k generated samples.

Logged posterior sharpness checkpoints (sanity check for Section D.4). To connect the sampler to the diagnostics in §D.4, we also log (i) the average per-token maximum probability ("mean_max_prob") and (ii) the average top-p nucleus size ("nucleus size") at selected steps. A representative subset of checkpoints is shown below; the strong monotone sharpening (early diffuse $\rightarrow$ late near-deterministic) is consistent across step budgets.

C.5 ROAR Sampler Details

The random-order autoregressive (ROAR) sampler instantiates the ROAR endpoint configurations of Eq. 7 as a single-pass inference path on the same DSL-finetuned checkpoint. The sampler reveals one token at a time in a uniformly random order; each position is denoised exactly once, with no remasking, revisits, or self-correction. It requires no retraining or sampler-specific finetuning—it is an alternative decoding rule applied to the same checkpoint used for iterative refinement. Algorithm 5 gives the procedure.

State representation. At each step, the input $\boldsymbol{z} \in \mathbb{R}^{L \times d}$ is composed position-by-position according to the current reveal status. Unrevealed positions $i \notin A$ are set to $\boldsymbol{z}_i = \mathbf{0}$, corresponding to $\gamma_i = 0$ in the DSL channel. Revealed positions $i \in A$ are set to $\boldsymbol{z}_i = \gamma_{\max}\, \text{enc}(s_i)$ with $\gamma_{\max} = 100$, providing the denoiser with the same near-clean signal it sees at training time on the ROAR endpoint

T	step k	t_k	mean_max_prob ↑	nucleus size (top-p=0.9) ↓
128	0	1.0000	0.0398	10268.59
128	56	0.5625	0.9391	12.17
128	71	0.4453	0.9585	8.72
128	127	0.0078	0.9928	1.74
256	0	1.0000	0.0398	10268.59
256	113	0.5586	0.9455	8.72
256	142	0.4453	0.9603	7.06
256	255	0.0039	0.9952	1.33
512	0	1.0000	0.0398	10268.59
512	227	0.5566	0.9466	9.85
512	284	0.4453	0.9632	6.21
512	511	0.0020	0.9973	1.04
1024	0	1.0000	0.0398	10268.59
1024	455	0.5557	0.9482	8.75
1024	568	0.4453	0.9640	4.83
1024	1023	0.0010	0.9980	1.17

Table 8: Logged posterior sharpness checkpoints from the sampler logs ("ReMDM logits stats" and "ReMDM nucleus size").

branch (§3.1, Appendix C.2). The sampler thus operates on the exact endpoint support the model was explicitly trained to cover; revealed positions act as conditioning context, unrevealed positions act as targets for the next prediction.

Reveal order. For each generation batch, we draw a single permutation π of $[L]$ uniformly at random and process positions in the order $\pi_1, \pi_2, \ldots, \pi_L$. The same permutation is shared across all sequences in the batch but resampled across batches. We do not use confidence-based or content-aware reveal orders, preserving the unbiased random-order structure of the ROAR estimator Eq. 7. The sampler also exposes a `causal=True` option that fixes π to the identity ($\pi_k = k$), recovering strict left-to-right autoregressive decoding; our reported results use the random-order setting.

Per-step prediction. At step k, the sampler queries the converter and backbone on the current partial state $\boldsymbol{z}$ to obtain token posteriors $p_\theta(\cdot \mid \boldsymbol{z})$ at every position, reads the marginal at position π_k, and samples s_{π_k} from this marginal using nucleus sampling. Position π_k is then added to A, and $\boldsymbol{z}_{\pi_k}$ is updated to $\gamma_{\max}\ \mathrm{enc}(s_{\pi_k})$ for use in subsequent steps. The remaining $L-1$ positions are recomputed in the next forward pass under the updated $\boldsymbol{z}$, but each position i is itself only *committed* (sampled) once, namely at step k such that $\pi_k = i$.

Hyperparameters. For OWT we use $L = 1024$, $\gamma_{\max} = 100$, top-$p = 0.9$, and run $T = L = 1024$ reveal steps per sample.

Algorithm 5 DSL ROAR sampler (single pass, random-order)

Require: Sequence length L; reveal-time SNR $\gamma_{\max}$; nucleus parameter top-p; `causal` flag (default `False`).

$A \leftarrow \emptyset$
$\boldsymbol{z}_i \leftarrow \mathbf{0}$ for all $i \in [L]$ ▷ All positions start at $\gamma_i = 0$
if `causal` **then**
  $\pi \leftarrow (1, 2, \ldots, L)$
else
  $\pi \sim \mathrm{Unif}(\mathfrak{S}_L)$ ▷ Single permutation shared across the batch
end if
for $k = 1, \ldots, L$ **do**
  Compute token posteriors $p_\theta(\cdot \mid \boldsymbol{z})$ via converter and backbone
  $s_{\pi_k} \sim \mathrm{nucleus}(p_\theta(\cdot \mid \boldsymbol{z})_{\pi_k};\ \text{top-}p)$
  $\boldsymbol{z}_{\pi_k} \leftarrow \gamma_{\max}\ \mathrm{enc}(s_{\pi_k})$
  $A \leftarrow A \cup \{\pi_k\}$
end for
return $\boldsymbol{s} = (s_1, \ldots, s_L)$

C.6 Hybrid Sampler Details

The hybrid sampler combines continuous-state EDM-style denoising with discrete masked refinement, both executed on the same DSL-finetuned checkpoint. The high-level idea is that continuous denoising provides a low-cost global initialization across all L token positions in parallel, after which a short discrete refinement stage commits the final tokens. Algorithm 6 gives the procedure; we describe the three stages below.

Stage 1: Continuous EDM denoising to σ_{switch}. We follow the standard EDM [16] parameterization with a Karras σ schedule, but truncate the schedule at an intermediate switching point σ_{switch} rather than running it all the way to $\sigma_{\min}$. Throughout the continuous stage we maintain two views of the state: the physical state $\boldsymbol{y}_t \in \mathbb{R}^{L\times d}$ and its rescaled form $\boldsymbol{z}_t = \boldsymbol{y}_t/\sigma_t^2$, which matches the SNR-parameterized input the DSL denoiser was trained on (§2.1). Initialization is $\boldsymbol{y}_0 = \sigma_{\max}\boldsymbol{\epsilon}$ with $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$. At each step we form $\boldsymbol{z}_t = \boldsymbol{y}_t/\sigma_t^2$, query the converter and backbone to obtain $\hat{\boldsymbol{x}}(\boldsymbol{z}_t)$, and update $\boldsymbol{y}_t$ via either an Euler or Heun step,

$$\boldsymbol{y}_{t+1} = \boldsymbol{y}_t + (\sigma_{t+1} - \sigma_t)\,\frac{\boldsymbol{y}_t - \hat{\boldsymbol{x}}(\boldsymbol{z}_t)}{\sigma_t} \quad \text{(Euler)}, \tag{65}$$

with the Heun second-order correction applied analogously. We optionally inject EDM-style stochastic churn [16] within a configurable σ band to improve sample diversity.

Stage 2: Projection to discrete tokens at σ_{switch}. At the switching point, we read out token logits from the backbone applied to $\boldsymbol{z}_T = \boldsymbol{y}_T/\sigma_T^2$ and sample an initial discrete sequence using nucleus sampling with parameters top-p and temperature. This produces an initial token sequence $\boldsymbol{s}^{(0)} = (s_1^{(0)}, \ldots, s_L^{(0)})$ that captures the global structure resolved by the continuous stage but may contain locally inconsistent tokens.

Stage 3: Discrete masked refinement. We refine $\boldsymbol{s}^{(0)}$ with T_{mdlm} steps of MDLM-style remasking. Tokens are mapped to high-SNR continuous states $\boldsymbol{z}_i = \gamma_{\max}\,\text{enc}(s_i)$ with $\gamma_{\max} = 100$ to provide the denoiser with a near-clean input at unmasked positions. At each refinement step, the most uncertain tokens—ranked by $1 - \max_v p_\theta(v \mid \boldsymbol{z}_i)$—are remasked (their $\boldsymbol{z}_i$ set to $\mathbf{0}$) and resampled conditioned on the remaining tokens. The mask ratio decays from an adaptive initial value r_0 to a target r_{end} via a cosine schedule. The initial ratio r_0 is set from the mean per-token uncertainty at the switching point, clipped to $[r_{\min}, r_{\max}]$, so sequences with sharper switching-point posteriors begin refinement with less aggressive remasking.

Hyperparameters. The configurations reported in Table 2 use the Karras σ schedule with $\sigma_{\max} = 10$, $\sigma_{\min} = 0.01$, $\rho = 7$, Heun solver, EDM churn $= 1.41$ within the band $[\sigma_{\min}, \sigma_{\max}]$, and $\sigma_{\text{switch}} = 0.3$. The discrete stage uses top-$p = 0.9$, temperature 0.8, $\gamma_{\max} = 100$, mask ratio bounds $[r_{\min}, r_{\max}] = [0.2, 0.8]$, and $r_{\text{end}} = 0$.

Algorithm 6 DSL hybrid continuous-then-discrete sampler

Require: Sequence length L; continuous step count T_{cont}; discrete step count T_{mdlm}; sigma schedule $\{\sigma_t\}_{t=0}^{T_{\text{cont}}}$ with $\sigma_0 = \sigma_{\max}$ and $\sigma_{T_{\text{cont}}} = \sigma_{\text{switch}}$; nucleus parameters (top-p, τ); refinement parameters $(\gamma_{\max}, r_{\min}, r_{\max}, r_{\text{end}})$.

$\boldsymbol{y}_0 \leftarrow \sigma_{\max}\,\boldsymbol{\epsilon}, \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}_{L\times d})$ ▷ Stage 1: continuous EDM
for $t = 0, \ldots, T_{\text{cont}} - 1$ **do**
  $\boldsymbol{z}_t \leftarrow \boldsymbol{y}_t/\sigma_t^2$
  $\hat{\boldsymbol{x}}_t \leftarrow$ DSL denoiser$(\boldsymbol{z}_t)$ ▷ converter → backbone
  Update $\boldsymbol{y}_{t+1}$ via Heun step on $(\boldsymbol{y}_t - \hat{\boldsymbol{x}}_t)/\sigma_t$ (with optional churn)
end for
$\boldsymbol{z}_{T_{\text{cont}}} \leftarrow \boldsymbol{y}_{T_{\text{cont}}}/\sigma_{T_{\text{cont}}}^2$ ▷ Stage 2: project to tokens
Compute logits ℓ_i at each position; set $u_i \leftarrow 1 - \max_v \text{softmax}(\ell_i)_v$
Sample $s_i^{(0)} \sim$ nucleus$(\ell_i; \text{top-}p, \tau)$ for all i
$r_0 \leftarrow \text{clip}(\bar{u}, r_{\min}, r_{\max})$ where $\bar{u}$ is the mean of $\{u_i\}$
for $k = 0, \ldots, T_{\text{mdlm}} - 1$ **do** ▷ Stage 3: discrete refinement
  $r_k \leftarrow r_{\text{end}} + \frac{1}{2}(r_0 - r_{\text{end}})(1 + \cos(\pi k/T_{\text{mdlm}}))$
  Select $\lceil r_k L \rceil$ positions with highest current uncertainty; mask them
  Set $\boldsymbol{z}_i \leftarrow \mathbf{0}$ at masked positions, $\boldsymbol{z}_i \leftarrow \gamma_{\max}\,\text{enc}(s_i^{(k)})$ otherwise
  Resample $s_i^{(k+1)}$ at masked positions from DSL denoiser logits
end for
return $\boldsymbol{s}^{(T_{\text{mdlm}})}$

D Mechanistic Analysis

The main text only briefly summarizes why DSL works in practice. This appendix provides supporting mechanistic evidence. The central message is that DSL improves refinement when training exposes the denoiser to the posterior states encountered under self-generated drafts, and that useful uncertainty is more important than mere local smoothness.

D.1 Cyclic Toy Setup and Correction Example

We use a simple synthetic discrete dataset to illustrate how DSL handles both masked tokens and visible-but-wrong tokens within a single framework. Fix an integer K and consider the dataset consisting of all cyclic shifts of a base sequence:

$$x^{(0)} = [0, 1, \ldots, K-1], \qquad x^{(i)} = \text{roll}(x^{(0)}, i), \quad i = 0, \ldots, K-1.$$

For visualization, each token is embedded in $\mathbb{R}^2$ on the unit circle with equal angular spacing, yielding a symmetric geometry in which posterior trajectories can be plotted directly.

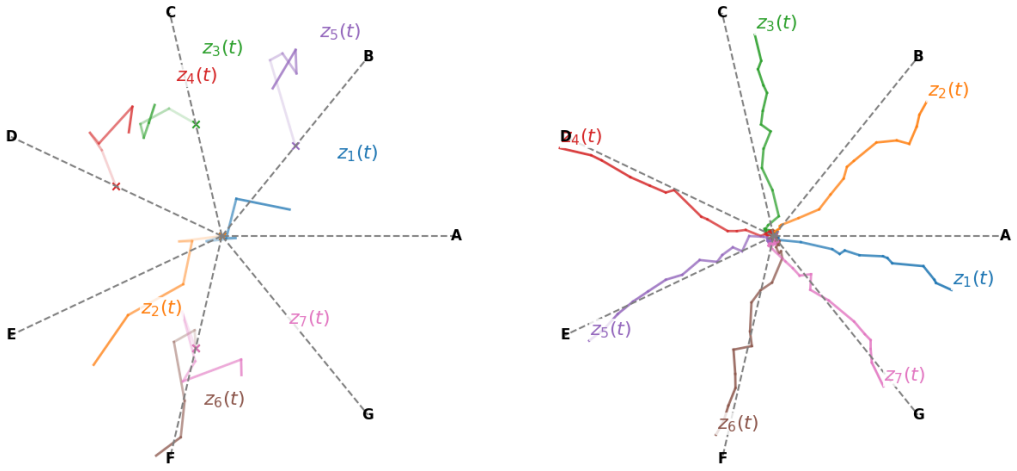
To illustrate correction, suppose the true sequence is `ABCDEFG`, but the input contains both masked and garbled positions:

Original:	A	B	C	D	E	F	G
Input:	$\underbrace{_}_{\text{masked}}$	$\underbrace{_}_{\text{masked}}$	$\underbrace{C}_{\text{correct}}$	$\underbrace{D}_{\text{correct}}$	$\underbrace{B}_{\text{garbled}}$	$\underbrace{F}_{\text{correct}}$	$\underbrace{F}_{\text{garbled}}$

In DSL, masked positions can be assigned $SNR_i = 0$, while visible but uncertain positions can be assigned small positive SNR. This is Proposition 2.1 with $0 < a < \gamma_i$ (Appendix A.5). This allows the model to use partial signal without forcing visible tokens to remain fixed. In our cyclic toy, this lets the same dynamics both fill masked positions and revise garbled visible ones.

D.2 Why Targeted Remasking Matters

The toy reveals a key asymmetry: masked positions are already mutable, whereas visible-but-wrong positions must first be remasked before they can be corrected. This is exactly why refinement-time uncertainty must remain informative. If the model cannot identify which visible tokens should become mutable again, then self-correction stalls. Conversely, once posteriors have already sharpened, late remasking tends to induce churn rather than useful revision.



(a) Masked + garbled input

(b) Reconstruction

Figure 3: **DSL correction under masked and garbled inputs.** The input contains both masked positions and visible-but-wrong tokens. DSL can assign zero SNR to masked tokens and small positive SNR to uncertain visible tokens, allowing the same refinement dynamics to both fill missing values and correct garbled ones.

D.3 Robustness to Self-Generated Intermediate Drafts

Two complementary ingredients explain DSL's robustness to imperfect self-generated drafts.

(1) Exposure to rollout-like partially corrupted contexts. Refinement-time drafts mix masked gaps with model-made mistakes that remain visible. DSL trains the denoiser under a mixed-support corruption distribution that covers both endpoint-like masked contexts and intermediate partially corrupted drafts. This broadens the support of training contexts and reduces brittleness when the model conditions on imperfect self-generated states.

(2) Smoothness can help, but remasking requires informative confidence. A plausible supporting factor is geometric smoothness in continuous embedding space: a denoiser that behaves smoothly across nearby contexts may generalize better across partially-correct drafts. However, in a ReMDM-style discrete sampler, correction is bottlenecked by *which tokens get remasked*. Thus the decisive property is not just smooth interpolation, but whether confidence remains informative enough to identify low-confidence visible tokens that should be revised.

D.4 Sampling Diagnostics and Over-Refinement

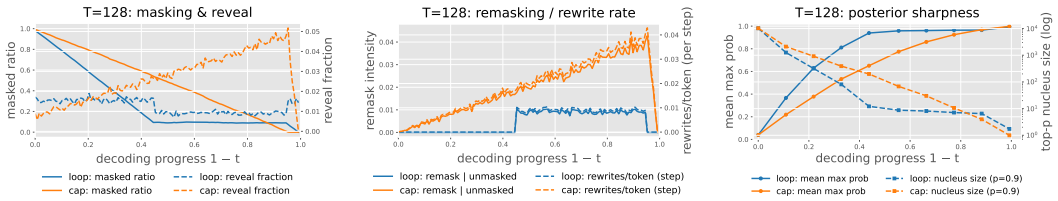
We track uncertainty, posterior sharpness, and repair workload in order to diagnose when iterative refinement remains productive and when it begins to over-refine.

Token uncertainty and entropy. Let $p_{\theta,t}(x_i)$ be the categorical distribution at token position i and sampling step t. Define average uncertainty and entropy by

$$u_t := \frac{1}{L}\sum_{i=1}^{L}\Big(1 - \max_v p_{\theta,t}(x_i = v)\Big), \tag{66}$$

$$H_t := \frac{1}{L}\sum_{i=1}^{L}\Big(-\sum_v p_{\theta,t}(x_i = v)\log p_{\theta,t}(x_i = v)\Big). \tag{67}$$

A rapid drop in (u_t, H_t) indicates posterior sharpening and the onset of a local-repair regime.



(a) Masking & reveal (b) Remasking / rewrite rate (c) Posterior sharpness

Figure 4: **Sampling diagnostics under a fixed step budget.** (a) Masking and reveal schedule. (b) Remasking intensity and realized rewrites per token. (c) Posterior sharpening measured by mean max-probability and top-p nucleus size.

Nucleus size as a sharpness proxy. For a fixed top-p threshold (e.g. $p = 0.9$), define

$$k_t := \frac{1}{L} \sum_i |\mathrm{TopP}(p_{\theta,t}(x_i), p)|. \tag{68}$$

Large k_t corresponds to broad uncertainty; small k_t indicates sharp, near-deterministic posteriors.

Repair workload and diminishing returns. Let $\mathcal{M}_t$ be the set of positions remasked or rewritten at step t. We track the remask ratio and realized token-change rate:

$$r_t := |\mathcal{M}_t|/L, \qquad \Delta_t := \frac{1}{L} \sum_{i=1}^{L} \mathbb{I}[x_i^{(t)} \neq x_i^{(t-1)}]. \tag{69}$$

Over-refinement is characterized by non-trivial r_t but tiny Δ_t, indicating diminishing returns.

Over-refinement hurts coverage. After posteriors sharpen, the sampler may still apply non-trivial remasking while realizing little actual change. This corresponds to diminishing returns: local likelihood proxies may keep improving while distributional coverage degrades. This explains the sweet spot observed in the main OWT experiments.

E Calibration and Endpoint-Smoothing Ablation

This appendix supports one of the paper's main practical conclusions: endpoint-like support is necessary, but endpoint-only training is insufficient. To obtain usable refinement-time uncertainty, the ROAR-style endpoint branch should be smoothed rather than concentrated entirely on atomic extremes.

E.1 Atomic vs. Smoothed ROAR Endpoints

DSL uses a ROAR-style endpoint branch to expose the denoiser to masking/reveal-like states. A natural baseline is *atomic* endpoint sampling, where ROAR tokens are assigned only the two extreme values

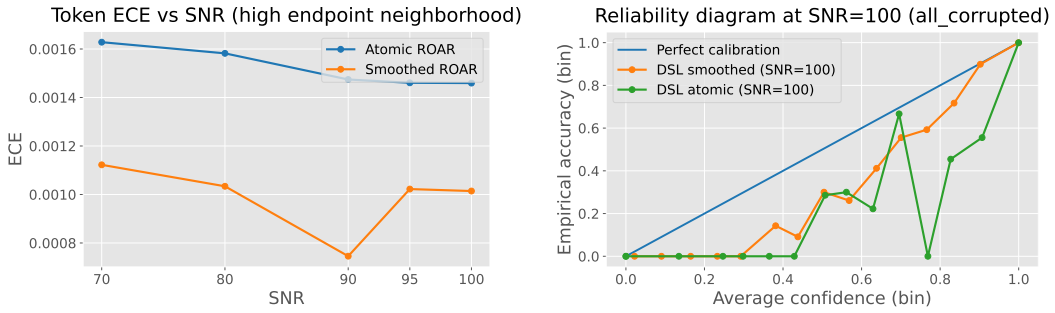
$$\gamma \in \{0, \gamma_{\max}\}.$$

In contrast, our *smoothed* endpoint variant samples from two narrow endpoint ranges,

$$\gamma \sim \mathrm{Unif}(0, \gamma_{\min}) \qquad \text{or} \qquad \gamma \sim \mathrm{Unif}(c\gamma_{\max}, \gamma_{\max}),$$

with equal probability.

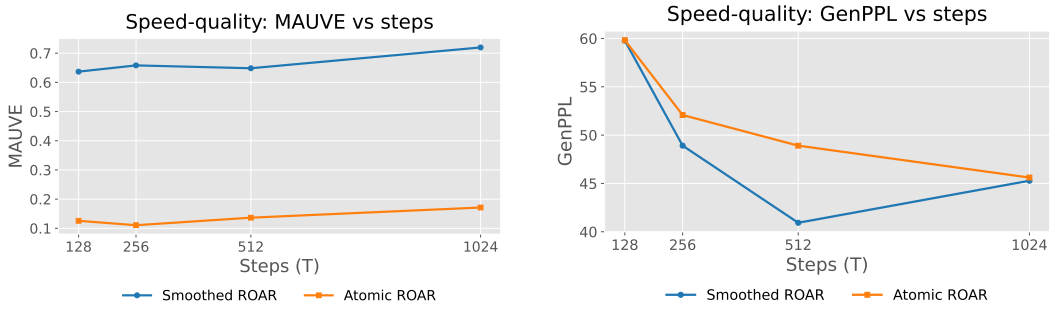
The motivation is practical. Atomic endpoints preserve the masking/reveal semantics, but make the endpoint branch too degenerate: near the fully masked and near-clean limits, corruption becomes almost deterministic, reducing local variation in training contexts. Smoothed endpoints preserve the same semantics while broadening the support near both extremes.



(a) ECE at large SNRs

(b) Reliability at SNR $= 100$

Figure 5: **Endpoint smoothing improves near-clean calibration.** We compare atomic ROAR endpoints ($\gamma \in \{0, \gamma_{\max}\}$) to smoothed endpoint ranges. Smoothing reduces ECE at large SNR and yields reliability curves closer to the diagonal near the clean limit.



(a) MAUVE vs. sampling steps

(b) GenPPL vs. sampling steps

Figure 6: **Endpoint smoothing improves the step–quality tradeoff under fixed decoding.** Using the same ReMDM-style sampler with identical schedules, smoothed-endpoint checkpoints achieve stronger MAUVE across step budgets while maintaining comparable or better GenPPL.

E.2 Near-Clean Calibration

We evaluate calibration under teacher forcing on held-out corrupted inputs. Compared with atomic ROAR endpoints, smoothed endpoints improve calibration precisely in the near-clean regime where refinement decisions are made.

These results indicate that endpoint-only support is insufficient. A denoiser trained only on atomic extremes can become poorly calibrated near the clean limit, even though those are exactly the states where refinement-time confidence must be trusted.

E.3 Downstream Step–Quality Tradeoff

Improved near-clean calibration translates into better downstream refinement under the same decoder family.

Together, these results support the practical point made in the main text: endpoint coverage is necessary because masking/reveal states are part of the likelihood family used by refinement, but collapsing training entirely onto atomic endpoints hurts calibration on intermediate and near-clean draft states. Smoothing the endpoint branch improves the quality of refinement-time uncertainty and thereby improves downstream step-efficient decoding.

F Additional Experiments

This section provides additional experimental details and diagnostics omitted from the main text for space. We report full OpenWebText generation results for masked-refinement decoders and hybrid continuous-then-discrete sampling, including GenPPL and sentence-entropy diagnostics in addition

Table 9: Training-side controls on OWT (MAUVE↑, 5k protocol). (a) Matched-compute continued pretraining (CPT) of MDLM under the official samplers. (b) Training variants under one fixed ReMDM-loop schedule.

		T=128	256	512	1024
(a) Official samplers					
MDLM (published)	MDLM sampler	0.015	0.023	0.031	0.042
MDLM + CPT	MDLM sampler	0.016	0.019	0.030	0.043
DSL-FT	MDLM sampler	0.402	0.481	0.506	0.495
MDLM (published)	ReMDM sampler	0.057	0.216	0.350	0.403
MDLM + CPT	ReMDM sampler	0.031	0.108	0.223	0.354
(b) Fixed ReMDM-loop schedule					
MDLM + CPT		0.011	0.030	0.069	0.111
DSL w/o continuous branch (λ=0)		0.005	0.004	0.005	0.004
DSL, atomic endpoints		0.234	0.222	0.277	0.307
DSL, full		**0.639**	**0.661**	**0.651**	**0.722**

Table 10: OWT ($L = 1024$) unconditional generation under masked-refinement decoders. Each entry is MAUVE↑ / GenPPL↓ / SentEnt↑; bold marks per-column best MAUVE among non-reference rows. Rows marked ‡ are taken from [40]; rows marked § from [27]. All other rows use our protocol. The two DSL rows use the same DSL-finetuned checkpoint with different ReMDM-family remasking schedules.

Method	$T = 128$	$T = 256$	$T = 512$	$T = 1024$
Data (reference)	1.000 / 14.8 / 5.44	1.000 / 14.8 / 5.44	1.000 / 14.8 / 5.44	1.000 / 14.8 / 5.44
AR (reference)	–	–	–	0.760 / 12.1 / 5.22
SEDD (absorb)	0.007 / 119.2 / 5.65	0.008 / 112.9 / 5.63	0.009 / 107.8 / 5.62	0.008 / 104.7 / 5.62
MDLM‡	0.015 / 61.5 / 5.52	0.023 / 55.8 / 5.49	0.031 / 53.0 / 5.48	0.042 / 51.3 / 5.46
MDLM+FB‡	0.064 / 42.8 / 5.44	0.086 / 39.0 / 5.41	0.103 / 37.0 / 5.40	0.133 / 33.8 / 5.35
MDLM+DFM‡	0.041 / 37.9 / 5.31	0.098 / 31.2 / 5.30	0.168 / 24.2 / 5.26	0.254 / 21.7 / 5.20
ReMDM‡	0.057 / 42.5 / 5.43	0.216 / 30.5 / 5.34	0.350 / 21.1 / 5.21	0.403 / 28.6 / 5.38
PRISM‡	0.175 / 17.9 / 5.10	0.268 / 16.2 / 5.07	0.281 / 15.4 / 5.04	0.260 / 15.0 / 5.02
PRISM-loop‡	0.118 / 21.5 / 5.18	0.294 / 18.0 / 5.15	0.423 / 16.4 / 5.12	0.527 / 15.3 / 5.10
ADLM§	0.140 / 52.5 / 5.52	0.349 / 39.9 / 5.46	0.573 / 31.6 / 5.40	0.699 / 25.4 / 5.35
DSL-FT + ROAR (naive)	–	–	–	0.551 / 47.4 / 5.19
DSL-FT + MDLM	0.402 / 54.2 / 5.25	0.481 / 50.5 / 5.22	0.506 / 49.4 / 5.21	0.495 / 48.3 / 5.20
DSL-FT + ReMDM (confidence-based)	0.615 / 68.0 / 5.36	0.622 / 61.6 / 5.32	**0.707** / 57.2 / 5.28	0.610 / 55.3 / 5.25
DSL-FT + ReMDM-loop (principled η_{cap})	**0.639** / 59.7 / 5.34	**0.661** / 49.0 / 5.27	0.651 / 41.0 / 5.18	**0.722** / 45.3 / 5.20

to MAUVE. We also provide supplementary Text8 likelihood results and implementation details for reproducibility.

F.1 OWT

Here we report the full OpenWebText generation tables. Table 10 compares DSL-finetuned checkpoints with masked-refinement baselines across sampling budgets. Table 11 reports diagnostics for the selected DSL hybrid continuous-then-MDM handoff configurations used in the main text.

F.2 Matched-compute and component controls

To separate DSL from extra OWT training, we continue the MDLM checkpoint under its own objective for the same number of steps (MDLM+CPT). CPT does not improve MDLM under either official sampler (Table 9), while DSL-FT reaches 0.40–0.51 under the same vanilla sampler. We then fix the ReMDM-loop schedule and vary only training (Table 9b). Removing the continuous branch (λ=0) drops MAUVE to 0.004–0.005 under all four samplers. GenPPL rises to $1{,}802$–$2{,}985$ while entropy stays at the MDLM level, so the output is incoherent rather than narrowed. Atomic endpoints reach only 0.22–0.31. Both the continuous branch and endpoint smoothing are needed, and the gain comes from training, not the decoder.

F.3 Text8

Dataset We test DSL on Text8 dataset [21], a relatively small-scale, character-level text modeling benchmark extracted from English Wikipedia.

Table 11: Full diagnostics for the selected DSL hybrid continuous-then-MDM sampler configurations on OWT. All rows use the same DSL-finetuned checkpoint. The main text reports MAUVE only; here we include GenPPL and sentence entropy diagnostics for the selected handoff configurations.

T_{cont}	T_{MDM}	NFE	σ_{switch}	GenPPL↓	MAUVE↑	SentEnt↑
16	16	32	0.49	60.3545	0.5008	5.0521
16	32	48	0.49	52.4820	0.6619	5.0664
16	48	64	0.49	49.0116	0.7017	5.0604
32	16	48	0.45	62.3669	0.5872	5.1147
32	32	64	0.46	53.7585	0.5891	5.1061
32	48	80	0.46	51.4659	0.7240	5.1033

Training Setups Following the previous work [8, 2, 20, 33], we evaluated all models on short text chunks of length 256, and also follow the same dataset split and transformer model size to parameterize the denoising models. For all the models including our method and baselines, we follow the common practice of using 12-layer transformers similar to GPT2-small scale [33]. Our transformer has the same number of heads (12) and hidden dimension (784) as in [28]. Note that all baseline diffusion language models are trained for a million steps, except for our model is trained for half a million.

Baselines We compare DSL against state-of-the-art continuous and discrete diffusion models, and autoregressive models [39]. Continuous diffusion baselines include Plaid [8], CDCD [4]. Discrete diffusion baselines include Discrete Diffusion Model (D3PM) [2], Score Entropy Discrete Diffusion (SEDD) [20], Masked Diffusion Language Model (MDLM) [28] and MD4 [33]. For autoregressive models, we choose Any-order Autoregressive Models ARDM [12] and MAC [34], and flow-based methods IAF/SCF [43], AR Argmax Flow [13], Discrete Flow [38], and Multinomial Diffusion.

G Limitations and Broader Impacts

DSL is a foundational method for diffusion-based language generation. Potential positive impacts include improving the efficiency, controllability, and flexibility of non-autoregressive text generation systems. At the same time, improvements in text generation quality may increase risks associated with synthetic text, including spam, misinformation, impersonation, or other harmful content generation. We do not release a deployed model or user-facing system, and we encourage future applications of DSL-style methods to follow standard safeguards for generative models, including misuse monitoring, content filtering, and careful release practices.

H Assets and Licenses

Table 12 summarizes the existing assets used in this work. We use these assets only for research purposes and follow their stated licenses and terms of use.

Table 12: Existing assets used in this work.

Asset	License / terms of use
OpenWebText	Public research dataset used for open-ended text generation evaluation; we cite the original release and follow its terms.
Text8	Public benchmark dataset used for character-level likelihood evaluation.
LLaDA / MDLM checkpoints	Pretrained checkpoints used for initialization and/or baseline comparison; we cite the original releases and follow their model-use terms.
MAUVE	Evaluation metric for open-ended text generation; we cite the original paper/package and follow the package license.
GPT-2 tokenizer / evaluator	Used for tokenization and GenPPL evaluation; we cite the original release and follow the corresponding terms.

I Additional Qualitative Samples

I.1 Text8

Example 1: [BOS]s to some anarchists to view the betrayal of destiay and show that the essential character of dut theme survived felled to before nine zero bc and were replaced most of all knightnote have changed dramatically basing on edmund lynn s name death on body f[EOS]

Example 2: [BOS]ics at a grant university s trade psychology marketing a statement us by saddam is an example of the abortion of aids most modern reputation says the body agrees to the pope whose arm is considered one of the most intrigued piecses of his body see also c[EOS]

Example 3: [BOS]h the irish protectors caused margraves and other powerful men and hence to present a confusingly successful tour the quotes more interpreted by mary newton puts the rejection of religion lewis and wats honored having led radcliffe men to question the la[EOS]

I.2 OWT

The following examples are generated from DSL finetuned checkpoint, with ReMDM-loop (principled $\eta_{\text{cap}}(T)$).

Example with $T = 128$.
BIP that only need the hard press. Now I think it works very well against your blindness.

At RT11

Here's how it works.

I've adjusted the fast slider to go above the 60's. Eventually, myshoot rate is now 4 percent. It works all on the T-Mobile bands. If I'm too low, I can cross it up the corner, but if it's with a stick hand (Carl Rasmus could cut), I infrotm a little near the edge of the box. If I'm going too low, I'll have the ability to adjust target angles and will still only know what it looks like.

The perfect, though disappointing feeling.

Links:

Lettered September 29st, 2014

are your no planning?

no planning is a natural lifestyle

@khovgeksriumki

forgo where the night of day breaks -go to Eavriumki is going to start the week.

hahahaha no a bet on the day. Daily is on the off right right now. From that I look at the page of his resume and ask if you can break the week, so if the week is fast, I don't have real hassle asking if he's going to break it. So I think he's going to start to go slow, maybe a couple may be there.

Signed on September 29st, 2014

and he's not going to go after that week.

(out/of-in his catalog)

On this week's 2nd, I had a lot of green green warning about my start of the weekend so I knew I'm going to lose. I didn't want to lose any extra weight or anything else, I just wanted to the rest time. At a point in the day I got nothing, I could get anything, but the end of the 60's was not what I wanted to go. I got myself on track so let's do a routine at 20am. Before I could wake 20am, I realized I needed to wake up at 20am. It didn't seem a good day.

Before I had it the next day, I bought a new SIM card, and I spent this week backwards when I was fixed. It was just not a good thing, but I kept the bell circle off for both days.

So it took about 7 minutes to go to work, then 5 minutes on the field between 8 and 7, and had a weight roll out around 5 minutes later. Then the clock on the card still constantly waiting for a delay in the middle of the weight world.

A couple of times, people say, “You don’t have time for this” or “well.” I’m able to stream people and online, listening to radio stations and live. I’m able to tell a lot of people, “Yes, you were able to track people down in front of you today. And I’m thankful I had told myself to do a few things a year and a half.” For me, I get hundreds of texts and messages every day, so it’s right to get to people.

NOLOM feel good

It kind of feels better having spoken to someone now. I’m hungry, but he’s going to do it already. He, I found him happy but I’m halfway through it right now. I’ll bet upon everything from I could buy a couple PTTs from a Gdigy and a T-Mobile, and I don’t want to buy.

The coolest thing now is the ability to fly, long distances, making swimmers with taking pictures. I could do not shoot a flyw if I would like to let it shoot a kmm for me, still out there, still at least where I am. I’ll have got something to show. I don’t worry about a flyw that I shoot for when it’s nowhere.

I could have shoot a group that’s not certain I can shoot, but I come to the angle I have to shoot and where my camera moves it. When I do shoot, I’m going to search, so I can see where a wave’s coming and see if it’s coming on right. Writing and writing is a way to be free from ideas and scrubbing from thought.

I drive a slow car. I’ll take the drive over it and drag it into a right. If I can take it it, I bend my head, and I’m going to take it. And

Example with $T = 256$.
NASts from age 19 to Cuba for Cuba and Cuba for Cuba with the exception of the house in Havana, a tourism official says.

Fiesta Fiaza is missing from ages 11 and 12 at the time she was conceived, but she happily adopted her younger daughter and adopted her at age. “The best question we have right now is how much she wants to give away,” says Arushal Chomam, Managing Director of Case & Privacy with the International Investigations Center. “We will give a guess and as soon as we are certified certified, please check for us for some time.”

See also: Cruise Cruise Cruise of Banks Search for MissingFiesta Fiaza – At $15. Ft Ft : $15. Ft Ft - Offer Nov. Nov. (1601) Grand Competition Winner Contest Winner New Original Siftth-Officile by an Italian Pararonous Unit

Generally, Fume does handcrafted, and she takes away from her collection. Even though she has not used her collection’s collection or design, she says she’ll put on any of her first set Grashinaza, something that won’t be used again with her collection for years. Not too much is said and given what she will be buying in the amount of time. Fume’s collection is also in an eye of local cultural heritage. “It’s disappointing and whether there are more or more examples of local heritage, but I just said there is something kind of African heritage there that is indicative of people who love Bayay Surplitia,” she says.

In addition to an appreciation of culturals and historical traditions, Fume has an appreciation of indigenous cultures as a part of her collection. She, too, will showcase their cultures across color, ethnicity, art, design, and indigenous traditions. “When I’ll give out, I hope my collection will be just as unique,” she says. “You know, when I come out and when I give out, my collection is just as strong,” she continues. “I’m surprised we have a green color, a dark color, or a black color, or something. It’s just what I hope is a positive experience.”

Fume currently holds about 100 pieces of Tokara Kirara Suguiaiko just out of $12,002 in Albany, N.Y. “It’s a mixed-together piece, there are a few strokes of stroke or the nib brush that give an impression, but it’s also so beautiful, it’s so cute and I think is so deeply related. My style is the inner hybrid of a stylistic mixed-together piece, not that or anything. It makes me feel comfortable and relaxed. It really feels like a marriage of something and the nuance of something you’ve been doing and done. It feels like fun and it feels like a touch in the room. It’s not very dark when you’re trying to make this touch with a different color palette as it does and sometimes you’re so distracted when you want to wake up and enjoy it. It’s fun and it inspires me as much as I want to do, and I am proud of my time and my work.”

All artwork which comes from a detailed dark coloring color tones to a subtle dark colored background scheme.

Estimated 1 12,013

Estimated 1 12,n1 12,013 (approximately $100)

Approximately 100 pieces of Tokara Suguiaiko Grashinaza hand-crafted of Tokara Suguiaiko

Details: As follows

Price: All sales, and pricing may vary and is a requirement for normal times. As soon as this site is on lockdown we will do an FAQ tour in stores for up to $50.

Cost: Available online (approx $8 + $9) when you can order order online. For more details what we can think we know by which orders ship on eBay. There's a web form in which you can just print the details, grab something else you can use to sign or keep one of your items on eBay page. You'll see some really nice zilzali, Mugabe, Picasso, and a stock, but you can have a strategy of ordering or keeping one of the items shipped directly to eBay.

• CY CYV Custom Design (rounded by

This is where you'll get sketches, drawings, and your own delivery information. What went wrong? This is where we say I'll go see my own artist drawings.

This is where CY CYV Custom Design/aka delivery info here for an overview of what you can choose for whom you can use to construct a shop of the

Example with $T = 512$.
See also: California is not trying to impose taxes on other states. Instead, states actually adopt the California tax code.

The best form of arguments already makes the great news when comparing taxes, local payouts, local profits, and national profits. But those arguments are too far from explaining how it works and are too far from explaining basic science of how well it works.

I have heard a lot stories about the Corporate Sales Tax. It is that big smaller corporations were forced to tax because they had no enough to pay taxes, other big corporations did not pay taxes because they deduct the taxes for other things like sales.

It is called no taxation: tax, corporate tax, the income is more than a dollar, more than a ‘ with a few multimillionaires and no place in society. It is that you don't have to pay your taxes all of the time.

Hawking Taxes in the US:

Corporations (usually big corporations) have learned how no taxation is the best way to live life. Big companies don't do things that kind of, people, have to pay the taxes, just by converting them to the US. They do other things that kind of: big corporations like being bailed out of the US. Instead of having to make a profit, they get paid.

There is a way in the world, a baker, not baker at all, where they are in trouble. It is because people have to think big corporations can get paid, because those companies hate the system and are trying to hide it.

If you become an investor, you get paid. Think of 0.8

Raising is the most important thing in the economy. It is it it that you aren't buying for $1.00 but if you pay the taxes only for the second one, it becomes 25% of your income. If you pay the taxes only, but according to the law, the rest will not stay on your income until it becomes 25% of your income. If you deduct a couple of deductions for that number, it won't stay on your taxable income, or the total, until the income tax is paid.

You don't see any gains from the tax until $1 million. By that time you get 25% of your income, the tax that now has reinvested into your. You can spend the dividends, save your taxes, and use it to retirement.

And of course, the money used to pay the tax doesn't change it needs to go up or higher. But the more the tax is paid, the more money it needs to spend.

Then the car gets paid off for the insurance, what the $2.50 gets and the $2.50 gets are not additional taxes.

But the amount that gets paid by less than the person gets charged. Tax taxes are taxed on some of the earns a person.

The big difference is that this is how the real income gets taxed. Corporate income is one of the highest levels of taxation. The real income income can be easily derived from the excise tax and the chained tax.

Compensation Ductes: Say you spend money on on a business and you earn the capital to ship it. You pay taxes on 'the corporate owner (or local company)' and you earn the same amount of capital costs to ship it.

You say your industry is money capital which consists of lots of money from other companies. First you own the profits. Then you give it to the government and use it to others. This is not true. You owe billions of dollars in income tax taxes to them. No profit government for 'gas makers', no taxes for 'gas makers.'

The capital dividend is not something that they use. If they trade with the people of capital, they pay them not, but that capital equals it.

Digid Benefit Ductes: America float these profits from the coffers of corporations to pay their taxes. They can and collect at the same time the rate of interest rates that they do. Let's wait until we don't realize how much of a nuisance we don't need to pay our taxes. Do we need to pay our taxes with a 2.5 trillion hole?

A couple of months ago we were actually paying taxes on a company that would provide to certain people id any ill. Another company had fixed higher tax rates for individual hospitals and other hospitals and hospitals, while another company had fixed 40 times the rate of federal tax rates for individual insurers, services and other health care providers, while another company had fixed more than

Example with $T = 1024$**.**
ongs of Africans played a role in determining population status, family structure, political etc. Traditionally, Africa was political symbol of each person of their own, ethnic or ethnic status. Instead, Africa was political symbol of goodwill for U.S. citizens of all races and cultures. In this case, African Americans have not looked back to the 1950s.

The split isn't because very few Africans (both present and later) arrived in the U.S. History does not necessarily fix the split. An analysis of "African Remnantss" discovered that far fewer African Americans arrived in the U.S than they did in the mid-9.01% of today's population from today. The split is primarily due to the influx of Chinese people.

The Chinese people from the past are not migrants who traveled to coast and coast. It may indeed have been allowed to enter the U.S. under open borders. This is probably partially, since the Chinese people are nowhere fully integrated into the U.S., and the borders for the past 25 years are still porous. Under open borders, the nation still belong to the Republic of Somalia. I would suggest that it was the nation of its own or its neighbors who made the decision.

Certainly the nation's productivity has not been a positive. One scholar whose family was Awaisan Ahmed told Al-Sal Al we have a "culture of immigrants originating from the founders of the Somali Republic of Somalia." Another former head of Oshala College concluded that the nation's productivity fell by an average 3.3% in 2013. That's what I'd guess from a fellow Al-Sahala college scholar and the Al-Qaeda fundamentalist.

That's not to say that the nation is a good neighbor — that the U.S. is a state that cares for all. As adjusted capita employment levels and as indexed adjusted income, productivity has fallen 15% since then. In addition, the new state provides health assistance for women, the elderly and all except for the state basic health care. How has the income fallen by nearly 15% in the rest of the country? So has the working class.

Let me begin with the opportunity of success in the U.S. as a nation nation – just from being considered a great nation and having a heart, but being able to get the heart that can go anywhere else. I'm a proud American born in the U.S., my life has gone through periods of war, occupation, occupation, employment as a citizen, as a city, elected as a citizen and a citizen of government. In World War II, America had only due to Russia and Russia in an effort to preserve, build and expand its bases to bring it up as a nation with a rich tradition of generosity and generosity. In the U.S., I

salute the great fathers as an instrumental to bring the world again. One that truly belongs to all Americans.

Today, I must formally recognize my power that U.S.-centric solutions must not simply solve the world's problems. As many southern American countries where I am live, I must recognize what I am doing – as a citizen – and our God Spirit to bring us closer to America. The U.S. Federal Government (Fed-Fed)

Everything Doesn't (CNN Talk Anything)

Fox News News

If statements from the 9/11/11 and 13/11 attacks are accurate, the hijackers or any false theories are simply going to blow up.

Shortly after attempting 9/9/11 to obtain documents from the CIA, they began questioning whether the hijackers, or if they wanted the documents.

The Pentagon 9/11acking conspiracy was a rum being featured on a CNN show. Why have America's world stature shift people's attention to credible answers?

Some of the stories are simply nuts.

One 9/9/11 theory that has been spun on to be "fabged" by CIA's secret secret operations (something U.S. authorities were able to track back), in theory it may have been planted by CIA. This story remains kept wraps ever after leaked 9/9/11 video Video Goes Be Allenged - Message to President Obama.

The narrative of the ongoing 9/9/11 is anything but false and remains as the mainstream media reporting has been consistent.

On Friday, a statement from the White House said that U.S. intelligence personnel were being intercepted by a computer used to collect them between January and October in 9/9/11.

The statement said that data collected when retrieving them from US intelligence operations operating in Moscow is classified because no records exist. Why don't we know that there are so many no records? i.e. the CIA's no records?

It seems like President Obama himself

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction clearly state the main contributions of DSL: the continuous-state framework, the practical training/interface recipe, and empirical evidence across the evaluated decoding paths. These claims are supported by the theory and experiments in the paper, and are scoped to the evaluated OpenWebText and Text8 settings rather than overstated as universal guarantees.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The paper discusses the main limitations of DSL, including the empirical scope of the evaluation, the reliance on appropriate training support over token-wise SNR states, the use of practical surrogate objectives, and the open challenge of scaling mixed-support training to more complex SNR configurations.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: The paper states the assumptions and notation needed for each theoretical result in the corresponding main-text sections, and provides full derivations and proofs in the appendix. The main text cross-references these appendix results where appropriate, so informal explanations in the core paper are backed by complete proofs.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: The paper provides the information needed to reproduce the main experiments, including datasets, model/backbone details, training objectives, sampling algorithms, evaluation protocols, and hyperparameter settings in the experimental sections and appendix. We also provide implementation details and will release anonymized code/configuration files to support reproduction of the reported results.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.

(c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: We provide an anonymized code repository with implementation details, configuration files, environment requirements, and scripts for training, sampling, and evaluating the main reported experiments. The paper and supplemental material also describe the datasets and corresponding data preparation/evaluation protocols.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: The paper describes the experimental setup for the main results, including datasets, model/backbone choices, training objectives, sampling procedures, evaluation metrics, and the key hyperparameters needed to interpret the results. Further implementation details and configuration files are provided in the appendix and anonymized code repository.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [No]

Justification: We do not report formal error bars or confidence intervals for the main large-scale experiments, since repeating the full training and sampling sweeps across multiple seeds is computationally expensive. To improve reliability, we use fixed evaluation protocols, large generation/evaluation sets, and additional ablations to support the empirical trends.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [Yes]

Justification: The appendix reports the compute resources used for the main experiments, including GPU type/count and approximate training, sampling, and evaluation time. We also describe the additional compute used for hyperparameter sweeps where relevant.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

Answer: [Yes]

Justification: We have reviewed the NeurIPS Code of Ethics and believe that the research conducted in this paper conforms to it. The work uses standard public datasets/evaluation protocols and does not involve human subjects, private data, or deceptive data collection.

Guidelines:

- The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: The paper discusses potential positive impacts from improving efficient and flexible diffusion-based language generation, as well as potential negative impacts from stronger synthetic text generation, including misuse for spam, misinformation, or other harmful content. We also note that responsible release practices and standard safeguards for generative models remain important when applying such methods.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: If model checkpoints are released, they will be released for research use with accompanying usage notes and limitations, and without any deployed user-facing generation service. The paper also discusses potential misuse risks of stronger synthetic text generation and encourages standard safeguards such as misuse monitoring and content filtering for downstream applications.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring

that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.

- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: The paper credits the original creators of the datasets, pretrained checkpoints, evaluation metrics, and code packages used in the experiments. We provide asset/license information in the appendix and respect the corresponding terms of use.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: We release anonymized research code, scripts, and configuration files as new assets to support reproducibility. Documentation is provided alongside the code, including environment setup and instructions for running the main training, sampling, and evaluation pipelines.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [N/A]

Justification: The paper does not involve crowdsourcing, human-subject studies, or human annotation experiments.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [N/A]

Justification: The paper does not involve crowdsourcing, human-subject studies, or human annotation experiments, so IRB approval or equivalent review is not applicable.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: The core method development and experimental methodology do not use LLMs as an important, original, or non-standard component. Any LLM use, if applicable, was limited to writing, editing, or formatting assistance and did not affect the scientific content, methodology, or originality of the work.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.