Free Lunch for Pass@k? Low Cost Diverse Sampling for Diffusion Language Models

Sean Lamont[1,2], Christian Walder[3], Paul Montague[2], Amir Dezfouli[4], Michael Norrish[1]
[1]Australian National University
[2]Defence Science and Technology Group
[3]Google DeepMind
[4]BIMLOGIQ
sean.lamont@anu.edu.au

Abstract

Diverse outputs in text generation are necessary for effective exploration in complex reasoning tasks, such as code generation and mathematical problem solving. Such Pass@k problems benefit from distinct candidates covering the solution space. However, traditional sampling approaches often waste computational resources on repetitive failure modes. While Diffusion Language Models have emerged as a competitive alternative to the prevailing Autoregressive paradigm, they remain susceptible to this redundancy, with independent samples frequently collapsing into similar modes. To address this, we propose a training free, low cost intervention to enhance generative diversity in Diffusion Language Models. Our approach modifies intermediate samples in a batch sequentially, where each sample is repelled from the feature space of previous samples, actively penalising redundancy. Unlike prior methods that require retraining or beam search, our strategy incurs negligible computational overhead, while ensuring that each sample contributes a unique perspective to the batch. We evaluate our method on HumanEval and GSM8K using the LLaDA-8B-Instruct and Dream-v0-Instruct-7B models, where we demonstrate significantly improved diversity and Pass@k performance across various sampling settings. As a simple modification to the sampling process, our method offers an immediate, low cost improvement for current and future Diffusion Language Models in tasks that benefit from diverse solution search. Our code, experiment logs and data are available at `https://github.com/sean-lamont/odd`.

1 Introduction

Diffusion Language Models (DLMs) such as LLaDA [20] and Dream [30] have recently demonstrated performance comparable to Autoregressive (AR) models while offering distinct advantages, including faster parallel generation, superior constraint adherence [14] and flexible generation controls that allow users to trade off speed and quality. Despite these capabilities, a persistent challenge in generation for both paradigms is *mode collapse* or redundancy. When sampling multiple solutions, standard techniques such as temperature scaling or beam search often produce highly correlated outputs, an example of which is provided in Figure 1. This redundancy diminishes the utility of multiple samples in Pass@k tasks such as coding or mathematics, as they result in repetitive outputs which do not effectively explore the possible solution space.

Diversity has recently become an active topic in Reinforcement Learning (RL) post-training for Language Models [13, 25], where it is often a prerequisite for learning from sparse rewards. In domains such as competition mathematics, a sampling strategy that collapses to a single mode may never generate a correct trajectory, even when the solution lies within the base model's capabilities.

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

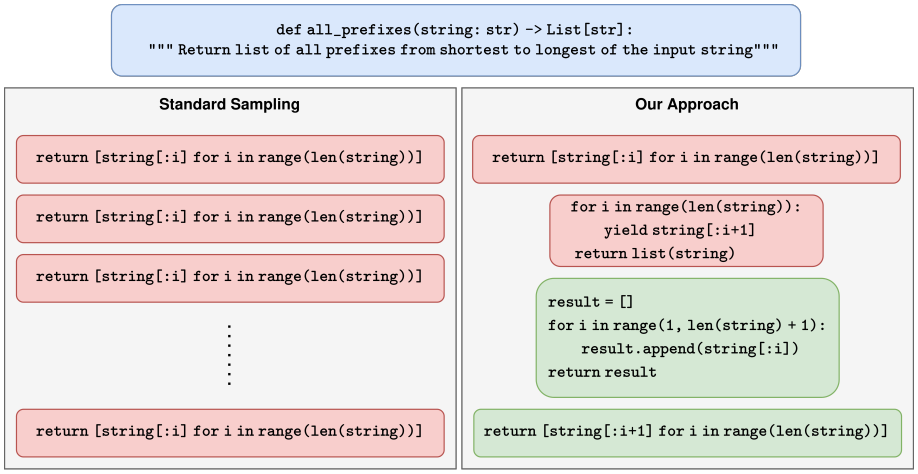

Figure 1: Example outputs from LLaDA with standard sampling (left) and our approach (right), both with a temperature of 1. Standard sampling leads to mode collapse over an incorrect approach, with a Pass@16 of 0. Our method instead guides samples to be mutually diverse, resulting in more distinct samples and greater coverage of the solution space.

To date, the primary research focus has been on post-training interventions incorporating diversity directly into the optimisation objective [13, 25]. Since the initial generation forms the basis of what is selected for downstream tasks or training, effective *inference-time* diversity remains an orthogonal and complementary objective.

Prior work in inference-time diversity has been restricted to AR models, focusing largely on diverse variants of beam search [12, 17, 23, 26]. Unlike AR models, diffusion models possess a global view of the sample at every inference step. This allows for methods which leverage both the current context and the model's predictions for the remaining tokens. This has been exploited in the image and protein synthesis domains, where sample trajectories are modified to optimise a batch-wide diversity term. Until now, this has not been explored in the context of text diffusion models [4, 18].

To address this gap we propose ODD: **O**rthogonal **D**iverse **D**iffusion, which encourages diverse ('odd') outputs to be sampled during inference. We use the intermediate states of the diffusion process to generate feature vectors, for which we apply a lightweight repulsion term during inference. As we generate a batch of N samples, the logits for the i-th sample are actively pushed away from the subspace spanned by samples $\{1, \ldots, i-1\}$. This ensures that the additional computation needed for larger batches effectively contributes to the exploration of solution space. We validate our approach on HumanEval and GSM8K, demonstrating that we can boost Pass@k with minimal overhead. To summarise our contributions:

- We present a training free framework for improving generative diversity in DLMs with minimal time and space overhead. We use this to introduce a diversity improving loss based on maximising the component of the current sample orthogonal to previous samples. In addition, we use our framework to extend a prior image diffusion diversity approach to the text domain, which we further improve upon.
- Testing our approach over HumanEval and GSM8K, we demonstrate significant and consistent improvements in Pass@k across various temperature settings and models. We further demonstrate the improvements transfer with untuned hyperparameters to MATH-500, to the RL post-trained LLaDA-1.5 and to the 30B-parameter RND1. We show that the additional solutions are distinct under Vendi, solution-level and blinded LLM-judge measures, and that they are not recovered by overgeneration and filtering at matched compute.
- We open source our code, giving immediate improvements to Pass@k for current and future DLMs. Our codebase is structured for easy experimentation and evaluation over different models, feature extractors, diversity terms and benchmarks to help encourage further improvements in the area. In addition, we release all experiment logs and data for full transparency and reproducibility, available at `https://github.com/sean-lamont/odd`.

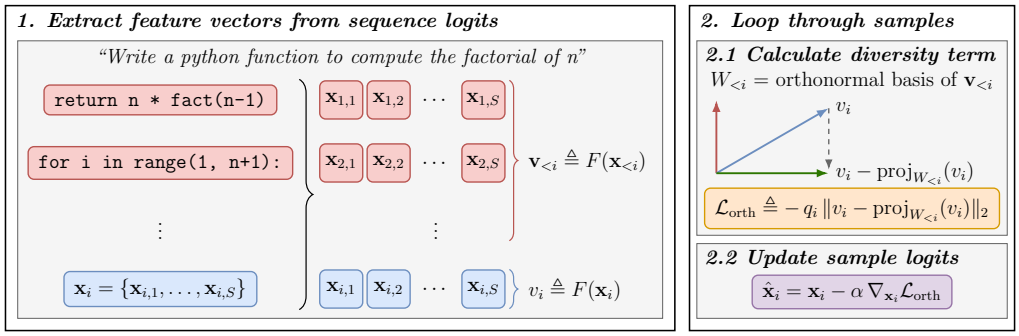

Figure 2: An overview of our proposed diverse sampling framework, using our orthogonal diversity loss. Given a full batch of N samples, feature vectors $F(\mathbf{x}_i)$ are extracted for the logits $\mathbf{x}_i$ for each sample $i \in \{1, \ldots, N\}$. We then loop through each sample, compute an orthonormal basis $W_{<i}$ for all previous sample features, and calculate the projection of the current sample features onto this basis. We then take the quality-weighted (q_i) norm of the residual with respect to this projection as the diversity objective; optimising it moves the sample in a direction distinct from previous samples.

1.1 Related Work

Recent works have integrated diversity directly into RL post-training [9, 13, 25]. By improving inference-time diversity, our method is complementary and applicable to any pre-trained diffusion model without modification. Several inference-time algorithms have been proposed for the AR domain. Diverse variants of beam search [1, 12, 17, 26] have been effective, though can incur significant computational overhead [23]. Other strategies involve conditioning the model on diverse prompts or random seeds, but do not address redundancy over multiple attempts as they treat samples independently [19]. Other work has tried to explicitly condition on past attempts to force exploration, but requires previously generated output before enforcing diversity [16]. Similarly, recent approaches which filter diverse sets from a larger pool are limited by the overhead of generating the initial candidates [11, 25] (compared at matched compute in Section 3.2). In contrast, we enforce diversity during the generation process itself, with minimal overhead.

In the context of Diffusion Models, inference-time diversity has been explored for image and protein synthesis [4, 18]. DiverseFlow [18] and similar methods [4] modify the sampling trajectory by optimising a batch-wide diversity objective. We extend DiverseFlow to the text domain (Section 3.1), where it improves on standard sampling, before showing that ODD improves substantially further. For text diffusion, adding noise to embeddings [29] promotes diversity but treats samples independently, ignoring what a batch has already explored. Our approach (Figure 2) models the dependency between samples, projecting each new sample away from the subspace spanned by previous generations. This results in structured exploration rather than random variance, ensuring that additional compute actively explores areas missed by previous samples.

2 Methodology

2.1 Diffusion Language Models

Unlike AR models which generate tokens sequentially, DLMs such as LLaDA and Dream generate all tokens in a sequence simultaneously through an iterative refinement process [20]. Formally, we define a forward process that gradually corrupts a clean data sample $\mathbf{x}^{(0)} \in \mathbb{R}^{S \times V}$ (where S is sequence length and V is vocabulary size, and sequences are one-hot encoded) into a fully masked state $\mathbf{x}^{(1)}$. This corruption is modelled as a masking process, where each token is independently replaced by a special `[MASK]` token with probability determined by a time-dependent schedule $\gamma(t)$, for $t \in [0, 1]$. In practice t is a normalised discrete index, $t \in \{0, 1/T, \ldots, 1\}$ for T sampling steps. Specifically, the forward transition probability for a token at sequence position s, denoted $x_s^{(t)}$, given the clean

token $x_s^{(0)}$ is defined as:

$$q(x_s^{(t)}|x_s^{(0)}) = \begin{cases} 1-\gamma(t) & \text{if } x_s^{(t)} = x_s^{(0)} \\ \gamma(t) & \text{if } x_s^{(t)} = \texttt{[MASK]} \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

where $\gamma(t)$ is a monotonically increasing function such that $\gamma(0) = 0$ and $\gamma(1) = 1$. The generative reverse process then aims to learn a denoising network to recover the original tokens from their masked initial states. For sampling, the interval $[0, 1]$ is discretised into T steps where T is generally set as a hyperparameter, and has the desirable property of allowing the user to directly control the quality-compute trade-off. Sampling runs from $t = 1$ (fully masked) down to $t = 0$. We focus on the common setting where the number of tokens unmasked at step ℓ is fixed to be the expected number of tokens unmasked over $[\gamma(\frac{\ell-1}{T}), \gamma(\frac{\ell}{T})]$. In deciding the specific tokens to unmask at a given step, it has been found that taking the highest confidence predictions leads to better results than random unmasking, and is the default approach for models like LLaDA or Dream [20, 30]. Regardless of the sampling approach, their bidirectional context gives DLMs a global view of the sequence at every step. This facilitates our inference-time intervention by allowing us to guide the entire remaining sequence based on the current state of the batch.

2.2 Inference-Time Diversity Sampling

For a batch of size N, each step t of the diffusion process yields predicted logits $X^{(t)} \in \mathbb{R}^{N\times S\times V}$, and we write $\mathbf{x}_i^{(t)} \in \mathbb{R}^{S\times V}$ for the logits of sample $i \in \{1, \dots, N\}$. We present a simple inference-time framework for modifying the predicted logits to penalise redundancy. Our method is applied independently for each time step, so we omit the t and use $\mathbf{x}_i$ to refer to the $(S \times V)$-dimensional logits for sample i with assumed timestep t. We assume a sample i with logits $\mathbf{x}_i$, previous samples $\mathbf{x}_{<i} = [\mathbf{x}_1; \cdots; \mathbf{x}_{i-1}]$, and a feature extractor $F : \mathbb{R}^{S\times V} \to \mathbb{R}^d$. For simplicity, we denote the feature vector for the current state as $v_i \triangleq F(\mathbf{x}_i)$, and the concatenation of feature vectors from previous states as $\mathbf{v}_{<i} \triangleq [F(\mathbf{x}_1); \cdots; F(\mathbf{x}_{i-1})]$. We define our diversity term as $\mathcal{L}_{\text{div}}(v_i, \mathbf{v}_{<i})$, which can be any differentiable loss function capturing diversity of features.

For each sample i, our framework returns updated logits $\hat{\mathbf{x}}_i$ which are pushed to optimise the diversity term with respect to previous samples in the batch:

$$\hat{\mathbf{x}}_i = \mathbf{x}_i - \alpha \cdot \nabla_{\mathbf{x}_i} \mathcal{L}_{\text{div}}(v_i, \mathbf{v}_{<i}) \tag{2}$$

where α is a step size controlling the strength of the repulsion. In line with previous work [18], we anneal α with the mask schedule, $\alpha_t = \gamma(t)\,\alpha$, which is linear in t for the linear schedules used by LLaDA and Dream. Intuitively, this gives higher diversity earlier in the generation when the high level structure is being formed, and prevents excessive intervention for later steps where finer details are commonly filled in.

2.3 Feature Extraction and Quality Scoring

The effectiveness of a diversity intervention will depend on the choice of feature extractor F, which must encode the semantic or structural qualities of the generation we wish to diversify. Several prior works have used computationally intensive features, such as pre-trained semantic encoders [23] or learned similarity classifiers [13]. To ensure negligible overhead, we instead opt for a lightweight feature extraction strategy that operates directly on the model's output distribution.

For a sample i at timestep t, we construct a unified probability distribution $P_i \in \mathbb{R}^{S\times V}$ that captures both the model's uncertainty and its committed decisions. For indices s corresponding to currently masked tokens, we compute the softmax over the predicted logits: $P_{i,s} = \text{softmax}(\mathbf{x}_{i,s})$. However, for unmasked tokens which have already materialised, we assign a probability of 1 to the generated token index and 0 elsewhere. We then obtain a fixed-size representation $v_i \in \mathbb{R}^V$ by applying max-pooling over the sequence dimension S:

$$v_i = \max_{s\in\{1,\dots,S\}} P_{i,s}. \tag{3}$$

This vector v_i represents a global confidence profile of the sequence across the vocabulary. By setting realised tokens to 1, we ensure that the feature vector is a weighted combination of the sequence's established history and its intended future trajectory. Appendix F includes experiments with other choices of feature extractor.

Algorithm 1: ODD: Orthogonal Diverse Diffusion

Input : Model logits $X = [\mathbf{x}_1, \ldots, \mathbf{x}_N]$, step size α, current timestep t, diffusion schedule γ, feature extractor F, stability constant ϵ

Output : Updated logits X'

$V, Q \leftarrow F(X)$; // $V = [v_1, \ldots, v_N]$, $Q = [q_1, \ldots, q_N]$

// Initialise orthonormal basis with the first sample

$w_1 \leftarrow \text{sg}\left(\frac{v_1}{||v_1||_2}\right)$; // sg() stops gradient tracking

$W_{<2} \leftarrow [w_1]$

$\alpha_t \leftarrow \gamma(t) \cdot \alpha$

$\mathcal{L}_{\text{total}} \leftarrow 0$

for i *in* $\{2, \ldots, N\}$ **do**

- $p_i \leftarrow \text{proj}_{W_{<i}}(v_i)$
- $\mathcal{L}_{\text{orth}}^{(i)} \leftarrow -q_i \cdot ||v_i - \text{sg}(p_i)||_2$
- $\mathcal{L}_{\text{total}} \leftarrow \mathcal{L}_{\text{total}} + \mathcal{L}_{\text{orth}}^{(i)}$
- // Update orthonormal basis
- $r_i \leftarrow \text{sg}(v_i - p_i)$
- $w_i \leftarrow \frac{r_i}{||r_i||_2 + \epsilon}$
- $W_{<i+1} \leftarrow [W_{<i} : w_i]$

// Update logits simultaneously

$X' \leftarrow X - \alpha_t \cdot \nabla_X \mathcal{L}_{\text{total}}$

return X'

Quality Awareness Strictly enforcing diversity can push the model towards low-probability, incoherent modes. To mitigate this, we also use a scalar *quality score* $q_i \in \mathbb{R}$ as part of the feature extractor. We define q_i as the average maximum confidence of the masked tokens in the sequence:

$$q_i = \frac{1}{|M_i|} \sum_{s \in M_i} \max_v P_{i,s,v} \tag{4}$$

where M_i is the set of masked token indices for sample i. By weighting with q_i, the model is discouraged from excessive diversity in areas it is confident in, preserving generation quality.

2.3.1 Orthogonal Diverse Diffusion

Algorithm 1 and Figure 2 outline our ODD approach. For the i-th sample in the batch with feature vector v_i, we maintain an orthonormal basis for the history vectors $\mathbf{v}_{<i}$. This is done using the Gram-Schmidt algorithm, where we define $W_{<i} \triangleq [w_1, \ldots, w_{i-1}]$ as the orthonormal basis for the history. Each w_j is the normalised residual of v_j with respect to its predecessors, with a small constant ϵ for numerical stability. We then set the diversity term $\mathcal{L}_{\text{div}}$ to be the negative norm of the residual of the projection of current sample features v_i onto the previous samples' subspace $W_{<i}$, scaled by the sample's quality score q_i:

$$\mathcal{L}_{\text{orth}}(v_i, \mathbf{v}_{<i}) \triangleq q_i \cdot \left(-||v_i - \text{proj}_{W_{<i}}(v_i)||_2\right). \tag{5}$$

By examining the gradient, we can see how this enforces an effective quality-diversity trade-off. Let $r_i = v_i - \text{proj}_{W_{<i}}(v_i)$ represent the orthogonal residual.

The total gradient with respect to the sample logits $\mathbf{x}_i$ naturally decomposes into two complementary terms:

$$\nabla_{\mathbf{x}_i} \mathcal{L}_{\text{orth}} = \underbrace{-q_i \nabla_{\mathbf{x}_i} ||r_i||_2}_{\text{Diversity Objective}} - \underbrace{||r_i||_2 \nabla_{\mathbf{x}_i} q_i}_{\text{Quality Objective}}. \tag{6}$$

The diversity objective directs the logits to maximise the magnitude of the residual, effectively forcing the model to explore the null space of previous generations, but scaled by the sample's current quality q_i. Simultaneously, the quality objective guides the sample toward higher confidence regions, scaled by its current diversity $||r_i||_2$. Consequently, if a sample is already highly diverse but of lower quality, the gradient naturally encourages restoring quality. Conversely, if a sample is high quality but near parallel to previous generations, the gradient focuses on orthogonal exploration. Appendix A provides an empirical demonstration of this for the interested reader.

In combination with a lightweight feature extractor F, this approach is extremely cheap as we are only tracking the gradients with respect to the model logits after generation. As outlined in Algorithm 1, we apply stop-gradients to the projection operations to treat the established subspace as a fixed target, restricting backpropagation strictly to the active sample and preventing the formation of an expensive, recursive computation graph. Unlike the optimisation objective of DiverseFlow [18] which optimises the batch globally, our method employs a greedy, sequential projection that forces each sample to independently diverge only from its predecessors. A direct consequence of this formulation is that the generation trajectory of any sample i depends exclusively on the subspace spanned by earlier samples. Therefore the resulting output for sample i will be batch size invariant, giving the same output across runs (given the same logits from the base model) regardless of the total batch size N, for any $N \geq i$.

3 Experiments

We first evaluate our approach using two widely used DLMs: LLaDA-8B-Instruct [20] and Dream-v0-Instruct-7B [30]. Both models are quantised to 4-bit precision to facilitate extensive evaluation on our hardware. We use the default linear scheduler for both models, so $\gamma(t) = t$. We conduct our experiments across two standard generative benchmarks: HumanEval [2] for code generation and GSM8K [3] for mathematical reasoning. For GSM8K, we evaluate on the first 200 problems of the test set to allow for a large scale analysis of our approach across a dense grid of hyperparameters. We also test ODD with a smaller grid on other benchmarks and models, including the full GSM8K, with tables and protocols in Appendix H.

We establish three primary comparative setups, using standard confidence based unmasking across all evaluations:

1. **Baseline**, standard LLaDA and Dream sampling with no logit intervention.
2. **Global Diversity (DPP)**, the Determinantal Point Process loss of DiverseFlow [18] for continuous image diffusion, adapted to the discrete text domain. Implemented as $\mathcal{L}_{\text{div}}$ in Equation 2 (Algorithm 2), this baseline demonstrates the flexibility of our framework and provides a separate global diversity benchmark for comparison.
3. **ODD (Ours)**, sequential orthogonal projection as in Algorithm 1 with the loss of Equation 5 as $\mathcal{L}_{\text{div}}$.

To rigorously evaluate our approach, we first test all three setups over a large parameter grid using LLaDA as the base model. We sweep across temperature settings $\tau \in \{0.0, 0.5, 1.0, 1.5, 2.0\}$ and $\alpha \in \{2, 8, 16, 32, 64, 128\}$, with 8 runs for each configuration for statistical confidence. To test that our approach extends to other models, we run a smaller scale sweep using Dream with just the baseline and ODD setups, testing with a grid of $\alpha \in \{8, 16, 64, 128\}$, with 4 runs for each configuration. Across all experiments, we fix the number of diffusion steps to 32, the generative length to 64 tokens, and the batch size to 16. We report the empirical Pass@16 metric, defined as the percentage of problems where the correct answer was generated in *at least one* of the 16 parallel batch attempts. Curves for all $k \leq 16$ are in Appendix C. Additional analyses, including the sensitivity of our approach to varying diffusion steps and generation lengths, are provided in Appendix G.

3.1 Results

Table 1 presents our main results. We observe a consistent and significant improvement in our approach across both benchmark tasks, and for both models. Surprisingly, the improvements are particularly large for temperature $\tau = 0.0$, which corresponds to the greedy decoding regime where Pass@k = Pass@1. The baseline is deterministic at $\tau = 0$, so in theory we would expect no residual for our approach to amplify. In practice, we found floating point differences between the batch elements' logits leave a small residual that the repulsion amplifies. Hence $\tau = 0$ behaves like a very small positive temperature: identical baseline samples (Appendix K) yet enough noise for ODD to separate them. For $\tau = 2$, the improvements are especially noteworthy in the HumanEval benchmark. HumanEval requires both correct syntax and reasoning over the complete output, so the excessive token-level diversity at $\tau = 2$ severely degrades the baseline, whereas our approach, which optimises quality as well as diversity, corrects for this without tuning.

Interestingly, although the DPP objective of DiverseFlow [18] optimises all samples jointly, our greedy stop-gradient approach gives higher Pass@k. Joint optimisation makes every sample repel every

Table 1: Pass@16 results for GSM8K and HumanEval across varying temperatures (τ) and step sizes (α). Results are reported as mean $\pm$ SE (8 runs for LLaDA, 4 runs for Dream). **Bold** values indicate the best result for a given model and temperature. DPP uses a global diversity objective [18] (Algorithm 2), while ODD uses our orthogonal projection approach (Algorithm 1). For DPP, the 'Best' score aggregates the maximum performance across all tested $\alpha \in \{2, 8, 16, 32, 64, 128\}$.

Step Size (α)	**Temperature** (τ)				
	0.0	0.5	1.0	1.5	2.0
GSM8K (200 PROBLEMS)					
LLaDA-8B-Instruct					
Baseline (Standard Sampling)					
0	61.0 ± 0.0	74.8 ± 0.2	81.3 ± 0.7	83.4 ± 0.6	76.5 ± 0.4
DPP [18]					
Best (all α)	66.5 ± 0.0	75.9 ± 0.7	82.3 ± 0.7	83.9 ± 0.8	80.8 ± 0.7
ODD (Ours)					
2	69.8 ± 0.2	80.8 ± 0.5	83.2 ± 0.7	83.1 ± 0.5	78.2 ± 0.8
8	70.6 ± 0.1	81.6 ± 0.4	84.0 ± 0.5	83.4 ± 0.3	81.3 ± 0.4
16	74.6 ± 0.3	82.8 ± 0.3	84.2 ± 0.6	84.4 ± 0.3	83.1 ± 0.4
32	76.9 ± 0.1	85.4 ± 0.5	86.2 ± 0.6	86.4 ± 0.5	84.9 ± 0.3
64	75.5 ± 0.0	87.3 ± 0.6	$\mathbf{87.9 \pm 0.5}$	86.7 ± 0.4	85.6 ± 0.5
128	$\mathbf{79.0 \pm 0.4}$	$\mathbf{87.8 \pm 0.4}$	87.6 ± 0.6	$\mathbf{88.6 \pm 0.5}$	$\mathbf{87.5 \pm 0.6}$
Dream-v0-Instruct-7B					
Baseline (Standard Sampling)					
0	42.5 ± 0.0	60.4 ± 0.1	74.9 ± 0.6	78.2 ± 0.3	78.1 ± 0.7
ODD (Ours)					
8	65.5 ± 0.0	78.8 ± 0.7	76.9 ± 0.3	79.8 ± 1.2	79.2 ± 1.1
16	67.0 ± 0.0	80.2 ± 0.4	80.4 ± 0.8	76.6 ± 0.5	78.4 ± 1.0
64	71.0 ± 0.0	$\mathbf{83.6 \pm 0.3}$	85.0 ± 0.7	82.8 ± 0.5	79.5 ± 0.3
128	$\mathbf{71.5 \pm 0.0}$	80.9 ± 0.7	$\mathbf{86.2 \pm 1.0}$	$\mathbf{85.2 \pm 0.6}$	$\mathbf{81.9 \pm 0.7}$
HUMANEVAL					
LLaDA-8B-Instruct					
Baseline (Standard Sampling)					
0	19.5 ± 0.0	33.5 ± 0.4	42.7 ± 0.6	41.8 ± 0.8	7.7 ± 0.6
DPP [18]					
Best (all α)	32.9 ± 0.0	37.9 ± 0.5	46.6 ± 0.4	45.3 ± 0.5	33.7 ± 0.7
ODD (Ours)					
2	28.1 ± 0.1	38.2 ± 0.6	44.8 ± 0.5	43.4 ± 0.4	8.1 ± 0.6
8	38.3 ± 0.7	47.2 ± 0.8	$\mathbf{51.3 \pm 0.6}$	$\mathbf{51.6 \pm 0.4}$	17.5 ± 0.5
16	$\mathbf{41.5 \pm 0.1}$	$\mathbf{48.8 \pm 0.7}$	$\mathbf{51.3 \pm 0.7}$	49.8 ± 0.8	33.7 ± 1.1
32	41.3 ± 0.2	44.9 ± 0.6	48.6 ± 0.7	47.3 ± 0.5	$\mathbf{40.2 \pm 0.7}$
64	41.3 ± 0.2	45.6 ± 0.6	46.2 ± 0.4	45.1 ± 0.9	37.7 ± 0.3
128	38.4 ± 0.0	43.7 ± 0.7	46.3 ± 0.8	44.3 ± 0.7	38.5 ± 1.3
Dream-v0-Instruct-7B					
Baseline (Standard Sampling)					
0	23.8 ± 0.0	33.2 ± 0.7	52.1 ± 0.6	56.7 ± 0.9	55.2 ± 0.7
ODD (Ours)					
8	54.3 ± 0.0	56.7 ± 0.4	64.2 ± 1.8	$\mathbf{64.2 \pm 0.6}$	60.2 ± 0.9
16	58.5 ± 0.0	59.0 ± 1.3	$\mathbf{64.5 \pm 1.2}$	63.3 ± 0.7	$\mathbf{64.0 \pm 1.1}$
64	$\mathbf{62.8 \pm 0.0}$	$\mathbf{61.0 \pm 0.5}$	$\mathbf{64.5 \pm 0.8}$	57.9 ± 1.2	57.8 ± 0.8
128	61.0 ± 0.0	58.2 ± 0.7	$\mathbf{64.5 \pm 1.0}$	56.2 ± 0.9	52.7 ± 1.8

other, which can push high-probability samples off the optimal mode, and the dependencies between all samples can produce chaotic optimisation trajectories. Our objective is instead a projection onto the detached subspace of previous samples, which is a static hyperplane.

ODD introduces a batch-aware dependency that trades off exploitation (quality) and exploration (diversity): individual sample quality (Pass@1) can decrease slightly as the model explores less probable paths (Figure 4), while coverage (Pass@16) increases significantly, so additional compute is converted into exploration rather than repeated failure modes.

Choosing α The optimal α is task dependent, but a bad choice costs little: of the 100 (α, τ) cells in Table 1, 96 improve on the baseline (mean $+11.4$, worst cell -2.4). A single fixed value $\alpha = 16$ gains $+6.4$/$+16.0$ points (LLaDA GSM8K/HumanEval) and $+9.7$/$+17.7$ (Dream) without tuning. We also found an α tuned on one model–benchmark pair and transferred to another retains $+12.2$ points on average (Appendix I). Every experiment on additional models and benchmarks in Appendix H

Table 2: Wall time comparison for generation between standard LLaDA/Dream (Baseline) and ODD (Batch Size 16). We report the mean execution time and SE per batch (in seconds) averaged over all α and temperature settings for each approach, and the relative overhead percentage. Our method introduces minimal latency across both benchmarks, and scales independently of model size.

APPROACH	LLADA-8B		DREAM-7B	
	GSM8K	HUMANEVAL	GSM8K	HUMANEVAL
Baseline (s)	22.5 ± 0.3	30.8 ± 0.4	15.9 ± 0.0	17.5 ± 0.0
ODD (s)	23.8 ± 0.5	32.0 ± 0.2	17.0 ± 0.0	18.1 ± 0.0
Overhead (%)	**+5.8%**	**+3.9%**	**+6.7%**	**+3.4%**

Table 3: Cumulative problem coverage across all temperature settings. ODD ($\alpha > 0$) significantly improves coverage on code generation (HumanEval), finding many problems missed by the baseline, while maintaining robust performance on GSM8K. Cumulative scores represent the union of solved instances across all five temperature settings ($\tau \in \{0.0, 0.5, 1.0, 1.5, 2.0\}$).

Dataset	**Step Size (α)**	LLADA-8B		DREAM-7B	
		Solved	**Coverage**	**Solved**	**Coverage**
GSM8K	0.0 (Baseline)	**198**	**99.00%**	188	94.00%
	8.0	192	96.00%	189	94.50%
	16.0	195	97.50%	194	97.00%
	64.0	195	97.50%	**197**	**98.50%**
	128.0	197	98.50%	195	97.50%
HUMANEVAL	0.0 (Baseline)	110	67.07%	128	78.05%
	8.0	124	75.61%	133	81.10%
	16.0	**129**	**78.66%**	**137**	**83.54%**
	64.0	123	75.00%	133	81.10%
	128.0	120	73.17%	131	79.88%

uses this default, showing our approach is effective without tuning. Temperature, the conventional diversity control, is far less forgiving: raising τ buys diversity at a steep quality cost at the extremes, whereas a mis-tuned α approximately recovers the baseline.

Overhead Analysis Table 2 shows the average time overhead of ODD over standard sampling for LLaDA and Dream, for the experiments in Section 3.1. We observe a small overhead for both benchmarks, particularly considering the improvement in Pass@16. As our approach only optimises logits after they are computed, the overhead is independent of the base model, and so will have a lower relative overhead for larger models. We conduct a more detailed overhead analysis in Appendix E.

Cumulative Results Table 3 presents the cumulative problem coverage, calculated as the union of all successfully solved problems across all runs and temperatures for each α regime. With 5 temperature settings, this results in an asymptotic upper bound of 640 total attempts per problem for LLaDA (8 runs $\times$ 16 samples $\times$ 5 temperatures) and 320 for Dream. Coverage is a solution-level correctness statistic: it counts the problems for which a correct solution exists anywhere in the pool, and therefore upper-bounds what any selection or filtering rule applied to that pool could achieve (Section 3.2). On HumanEval, where strict syntactic and logical constraints make the solution space highly sparse, standard sampling achieves a maximum coverage of only 110 problems for LLaDA and 128 for Dream. ODD ($\alpha = 16$) expands this reachable space to 129 problems for LLaDA and 137 for Dream, yielding significant relative improvements of 17.3% and 7.0% (these are problems for which no correct solution appears in any of the baseline's attempts). We also observe this expansion of the solution space is robust across all tested α configurations. The relative gain is larger for LLaDA, whose weaker code prior collapses more often. For Dream, the stronger prior already covers much of the dataset. Finding such rare solutions by standard sampling requires an exponentially growing budget [2, 24, 28], whereas ODD reaches them at a latency overhead of 3.9% (LLaDA) and 3.4% (Dream) (Table 2).

GSM8K presents a more lenient evaluation, where success is given only by the final numerical token. Given a large budget of 640 attempts, standard sampling effectively saturates GSM8K for LLaDA (198 problems, 99.0% coverage). Under this extreme scaling regime, ODD experiences a marginal drop (98.5% at $\alpha = 128$). Conversely, Dream achieves 94.0% coverage with standard sampling, with significant improvements by ODD, reaching up to 98.5%. We hypothesise that the slight reduction in coverage for LLaDA occurs because ODD actively restricts drift into low quality exploratory spaces. As GSM8K evaluates only the final number, standard sampling over many attempts has a higher chance of finding a rare correct answer from a low quality trajectory. At $\tau = 2.0$ we observe standard LLaDA sampling generally produces broken language and irrelevant special tokens, yet maintains 93.5% GSM8K coverage by emitting the correct final token (ODD: 96.0%). On HumanEval, where full algorithmic correctness is required, its coverage at $\tau = 2$ collapses to 21.3% against 62.2% for ODD. Ultimately, any marginal coverage decrease is outweighed by the dramatic gains in practical sample efficiency (Table 1), demonstrating that ODD either strictly expands the reachable solution space or significantly improves sample efficiency depending on the strength of the base model.

3.2 Comparison to Overgeneration and Filtering

Table 4: Overgeneration and filtering, HumanEval, LLaDA, Pass@16 (one run per pool). *First-16*: first 16 samples of the M=17 pool. *Filtered*: 16 of M chosen by k-centre selection on sentence embeddings. *Oracle*: best possible selection of 16. M=17 approximately matches ODD's wall-clock time, M=32 doubles it. ODD is the $\alpha = 16$ row of Table 1 (mean of 8 runs).

		Pool M=17		**Pool** M=32		
τ	First-16	Filtered	Oracle	Filtered	Oracle	ODD (α=16)
0.5	32.3	32.3	32.3	37.2	38.4	**48.8**
1.0	42.1	43.3	43.3	44.5	48.2	**51.3**
1.5	37.8	39.0	39.0	34.8	45.7	**49.8**

A practical alternative to diverse decoding is to generate more samples than needed and filter them [11, 25]. Any such scheme is bounded by an oracle that selects the best 16 samples of the pool with hindsight, and Table 3 already bounds this at an equal 640-attempt budget (110 HumanEval problems solved anywhere in the baseline pool against 129 for ODD). The deficit hence lies in generation, which no selection rule can repair. To expand on this, at matched cost we generate baseline pools of M=17 samples (matching ODD's overhead of at most 6.7%, Table 2) and M=32 (about twice its compute). We then select 16 by greedy k-centre selection [6] over MiniLM sentence embeddings [22, 27], and report each pool's oracle (Table 4, Appendix J). Filtering generally improves slightly on the first 16 samples (the $\tau = 1.5$, M=32 pool is the exception), but even the oracle over the $2\times$ pool trails ODD at every temperature. This shows that twice the compute on independent samples, even with a perfect selection oracle, does not match our low-overhead, dependency-based approach.

3.3 Are the Additional Solutions Distinct?

Pass@k rewards one correct sample, so one may ask whether ODD finds genuinely different solutions or merely nudges near-misses over the threshold. Table 5 measures this on the stored generations of Section 3.1 with the Vendi score [5] and solution-level statistics over the correct samples of each batch. ODD raises both Vendi scores and increases the number of distinct correct solutions per problem by half on GSM8K and twofold on HumanEval. A lack of semantic diversity is clearly apparent in the baseline, whose correct HumanEval samples are mostly copies of one program (74.8% of correct pairs within two tokens, median distance 0). In comparison, ODD's solutions differ by a median of 12 tokens. For problems that ODD solves and the baseline never solves at the same temperature, ODD's solutions lie at a mean minimum normalised edit distance of 0.33 (GSM8K) and 0.62 (HumanEval) from every baseline sample. This demonstrates new solution modes rather than small edits of near-misses.

LLM-Judge Diversity Study As diversity metrics will not capture the full semantics of the solutions, we also ran a blinded pairwise study with an LLM judge (Claude Fable 5) on all 364 problems of the $\tau = 1.0$ runs. For each problem the judge saw both batches with labels removed,

Table 5: Solution-level diversity of the generations (LLaDA, $\tau = 1.0$, $\alpha = 16$); means are over 1,600 GSM8K and 1,312 HumanEval batches of 16. Vendi scores: effective number of distinct samples under a semantic (MiniLM) and a lexical (n-gram) kernel. Distinct correct: clusters of correct samples at normalised edit distance ≤ 0.3, per problem and per solved problem. AST-distinct: syntactically distinct programs. Edit distance is in tokens between correct pairs of a batch. Novelty is a normalised edit distance in $[0, 1]$. Pass@16 from Table 1.

	GSM8K		HUMANEVAL	
	Baseline	ODD	Baseline	ODD
Pass@16 (%)	81.3	**84.2**	42.7	**51.3**
Vendi score, semantic	1.90	**2.17**	2.14	**2.79**
Vendi score, n-gram	4.34	**6.14**	3.52	**6.31**
Distinct correct solutions / problem	3.03	**4.74**	0.65	**1.29**
Distinct correct solutions / solved problem	3.72	**5.61**	1.53	**2.52**
AST-distinct correct programs / problem	–	–	1.09	**1.86**
Problems with ≥ 2 distinct correct (%)	60	**75**	16	**34**
Median edit distance between correct pairs	16	**26**	0	**12**
Correct pairs within ≤ 2 tokens (%)	14.4	**0.7**	74.8	**29.1**
Novelty of ODD-only solutions (normalised distance to nearest baseline sample)	–	0.33	–	0.62

order randomised and in an isolated context. It was instructed to choose the batch with more genuinely distinct solution approaches, or select a tie (details in Appendix K). It preferred ODD on 227 problems and the baseline on 12, with 125 ties. 9 of 10 hidden verbatim duplicates received identical verdicts and 18 of 20 order-swapped duplicates flipped correctly, demonstrating position bias is not driving the result.

4 Conclusion

We presented ODD, a training free inference intervention that significantly enhances the diversity and Pass@k performance of Diffusion Language Models. By sequentially projecting intermediate logits away from the subspace of previous samples, we effectively minimise sampling redundancy at negligible computational cost. Our results demonstrate that this simple geometric repulsion enables more effective exploration of the solution space, particularly in domains requiring complex reasoning where valid solutions are sparse. Our work highlights an advantage unique to the diffusion paradigm, which is the ability to intervene globally on the generation process at low cost. This matters increasingly as inference-time compute becomes a major factor in scaling reasoning.

Limitations and Future Work Our feature extractor is deliberately simple, omitting partial-sequence semantics and positional information. Adapting semantic diversity objectives from AR decoding (e.g. [23]) to diffusion remains open. The repulsion strength α is a hyperparameter: a fixed $\alpha = 16$ transferred to every model and benchmark we tested, but the best value is task dependent (Section 3.1), and $k = 16$ reflects our batch budget rather than being a property of the method (Appendix C). The main grid covers the first 200 GSM8K problems, HumanEval, two 7/8B instruction-tuned models and fixed decoding settings. Appendix H extends it with single runs over a reduced temperature grid. Semi-autoregressive [7] and dynamic unmasking decoders remain untested. Finally, our diversity assessment relies on algorithmic metrics and an LLM judge as proxies for human judgement. Combining our approach with Reinforcement Learning remains unexplored, and is a promising avenue for future work.

Acknowledgments and Disclosure of Funding

This work was supported by the Defence Science and Technology Group (DSTG). The authors declare no competing interests.

References

[1] P. Anderson, B. Fernando, M. Johnson, and S. Gould. Guided Open Vocabulary Image Captioning with Constrained Beam Search. In M. Palmer, R. Hwa, and S. Riedel, editors, *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 936–945, Copenhagen, Denmark, Sept. 2017. Association for Computational Linguistics.

[2] M. Chen, J. Tworek, H. Jun, Q. Yuan, H. Pondé, J. Kaplan, H. Edwards, Y. Burda, N. Joseph, G. Brockman, A. Ray, R. Puri, G. Krueger, M. Petrov, H. Khlaaf, G. Sastry, P. Mishkin, B. Chan, S. Gray, N. Ryder, M. Pavlov, A. Power, L. Kaiser, M. Bavarian, C. Winter, P. Tillet, F. P. Such, D. W. Cummings, M. Plappert, F. Chantzis, E. Barnes, A. Herbert-Voss, W. H. Guss, A. Nichol, I. Babuschkin, S. Balaji, S. Jain, A. Carr, J. Leike, J. Achiam, V. Misra, E. Morikawa, A. Radford, M. M. Knight, M. Brundage, M. Murati, K. Mayer, P. Welinder, B. McGrew, D. Amodei, S. McCandlish, I. Sutskever, and W. Zaremba. Evaluating Large Language Models Trained on Code. *ArXiv*, abs/2107.03374, 2021.

[3] K. Cobbe, V. Kosaraju, M. Bavarian, M. Chen, H. Jun, L. Kaiser, M. Plappert, J. Tworek, J. Hilton, R. Nakano, C. Hesse, and J. Schulman. Training Verifiers to Solve Math Word Problems. *ArXiv*, abs/2110.14168, 2021.

[4] C. Ferragu. Diversity-Guided Sampling for Protein Sequence Design via Iterative Refinement. Master's thesis, Princeton University, 2024.

[5] D. Friedman and A. B. Dieng. The Vendi Score: A Diversity Evaluation Metric for Machine Learning. *Transactions on Machine Learning Research*, 2023.

[6] T. F. Gonzalez. Clustering to minimize the maximum intercluster distance. *Theoretical Computer Science*, 38:293–306, 1985.

[7] X. Han, S. Kumar, and Y. Tsvetkov. SSD-LM: Semi-autoregressive Simplex-based Diffusion Language Model for Text Generation and Modular Control. In A. Rogers, J. Boyd-Graber, and N. Okazaki, editors, *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11575–11596, Toronto, Canada, July 2023. Association for Computational Linguistics.

[8] D. Hendrycks, C. Burns, S. Kadavath, A. Arora, S. Basart, E. Tang, D. Song, and J. Steinhardt. Measuring Mathematical Problem Solving With the MATH Dataset. In J. Vanschoren and S. Yeung, editors, *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, volume 1, 2021.

[9] Z. Hu, S. Zhang, Y. Li, J. Yan, X. Hu, L. Cui, X. Qu, C. Chen, Y. Cheng, and Z. Wang. Diversity-Incentivized Exploration for Versatile Reasoning. *ArXiv*, abs/2509.26209, 2025.

[10] A. Kulesza and B. Taskar. *Determinantal Point Processes for Machine Learning*. Now Publishers Inc, 2012.

[11] S. Lamont, C. Walder, A. Dezfouli, P. Montague, and M. Norrish. 3D-Prover: Diversity Driven Theorem Proving With Determinantal Point Processes. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

[12] J. Li, W. Monroe, and D. Jurafsky. A Simple, Fast Diverse Decoding Algorithm for Neural Generation. *ArXiv*, abs/1611.08562, Nov. 2016.

[13] T. Li, Y. Zhang, P. Yu, S. Saha, D. Khashabi, J. Weston, J. Lanchantin, and T. Wang. Jointly Reinforcing Diversity and Quality in Language Model Generations. *ArXiv*, abs/2509.02534, 2025.

[14] X. L. Li, J. Thickstun, I. Gulrajani, P. Liang, and T. B. Hashimoto. Diffusion-LM improves controllable text generation. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS '22, Red Hook, NY, USA, 2022. Curran Associates Inc.

[15] H. Lightman, V. Kosaraju, Y. Burda, H. Edwards, B. Baker, T. Lee, J. Leike, J. Schulman, I. Sutskever, and K. Cobbe. Let's Verify Step by Step. In *The Twelfth International Conference on Learning Representations*, 2024.

[16] Y. Lu, D. Wang, T. Li, D. Jiang, S. Khudanpur, M. Jiang, and D. Khashabi. Benchmarking Language Model Creativity: A Case Study on Code Generation. In L. Chiruzzo, A. Ritter, and L. Wang, editors, *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 2776–2794, Albuquerque, New Mexico, Apr. 2025. Association for Computational Linguistics.

[17] C. Meister, M. Forster, and R. Cotterell. Determinantal Beam Search. In C. Zong, F. Xia, W. Li, and R. Navigli, editors, *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6551–6562, Online, Aug. 2021. Association for Computational Linguistics.

[18] M. M. Morshed and V. N. Boddeti. DiverseFlow: Sample-Efficient Diverse Mode Coverage in Flows. *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 23303–23312, 2025.

[19] V. Nagarajan, C. H. Wu, C. Ding, and A. Raghunathan. Roll the dice & look before you leap: Going beyond the creative limits of next-token prediction. In *Forty-second International Conference on Machine Learning*, 2025.

[20] S. Nie, F. Zhu, Z. You, X. Zhang, J. Ou, J. Hu, J. Zhou, Y. Lin, J.-R. Wen, and C. Li. Large Language Diffusion Models. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

[21] Radical Numerics. Training Diffusion Language Models at Scale using Autoregressive Models. Technical report, Radical Numerics Inc. `https://www.radicalnumerics.ai/assets/rnd1_report.pdf`, Oct. 2025. Model RND1-Base-0910: `https://huggingface.co/radicalnumerics/RND1-Base-0910`; code: `https://github.com/RadicalNumerics/RND1`.

[22] N. Reimers and I. Gurevych. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In K. Inui, J. Jiang, V. Ng, and X. Wan, editors, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China, Nov. 2019. Association for Computational Linguistics.

[23] W. Shi, Y. Cui, Y. Wu, J. Fang, S. Zhang, M. Li, S. Han, J. Zhu, J. Xu, and X. Zhou. Semantic-guided Diverse Decoding for Large Language Model. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

[24] C. V. Snell, J. Lee, K. Xu, and A. Kumar. Scaling LLM test-time compute optimally can be more effective than scaling parameters for reasoning. In *The Thirteenth International Conference on Learning Representations*, 2025.

[25] J. Tuyls, D. J. Foster, A. Krishnamurthy, and J. T. Ash. Representation-Based Exploration for Language Models: From Test-Time to Post-Training. In *The Fourteenth International Conference on Learning Representations*, 2026.

[26] A. K. Vijayakumar, M. Cogswell, R. R. Selvaraju, Q. Sun, S. Lee, D. J. Crandall, and D. Batra. Diverse Beam Search: Decoding Diverse Solutions from Neural Sequence Models. *ArXiv*, abs/1610.02424, 2016.

[27] W. Wang, F. Wei, L. Dong, H. Bao, N. Yang, and M. Zhou. MiniLM: Deep Self-Attention Distillation for Task-Agnostic Compression of Pre-Trained Transformers. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 5776–5788. Curran Associates, Inc., 2020.

[28] C. H. Wu, S. Goyal, and A. Raghunathan. Mode-conditioning unlocks superior test-time compute scaling. In *The Fourteenth International Conference on Learning Representations*, 2026.

[29] J. Wu, Z. Wan, X. Yu, Y. Yang, Y. Huang, I. Tsang, and Y. You. Time-Annealed Perturbation Sampling: Diverse Generation for Diffusion Language Models. *ArXiv*, abs/2601.22629, 2026.

[30] J. Ye, Z. Xie, L. Zheng, J. Gao, Z. Wu, X. Jiang, Z. Li, and L. Kong. Dream 7B: Diffusion large language models. *arXiv preprint arXiv:2508.15487*, 2025.

[31] F. Zhu, R. Wang, S. Nie, X. Zhang, C. Wu, J. Zhou, Y. Lin, J.-R. Wen, and C. Li. LLaDA 1.5: Variance-Reduced Preference Optimization for Large Language Diffusion Models. In M. Liakata, V. P. Moreira, J. Zhang, and D. Jurgens, editors, *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11425–11460, San Diego, California, United States, July 2026. Association for Computational Linguistics.

A Diversity Dynamics

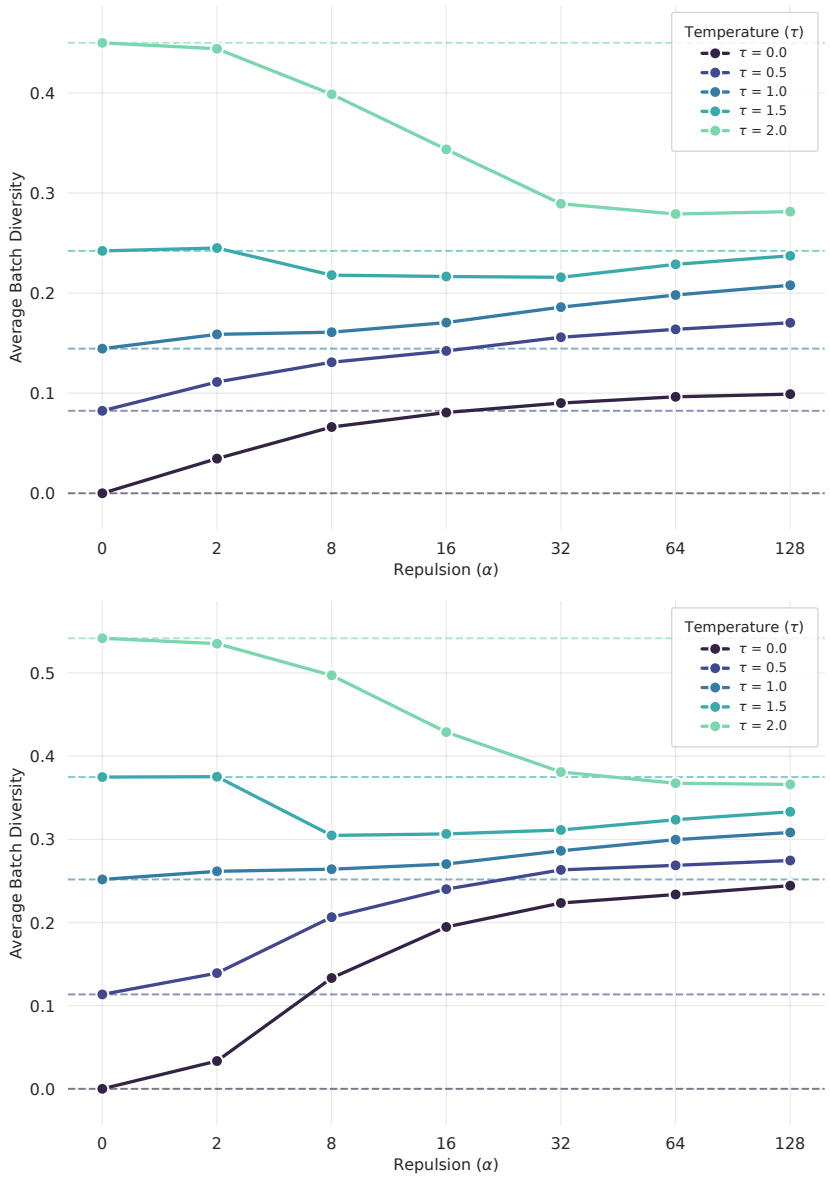

Figure 3: Average batch diversity (1 - cosine similarity of sentence embeddings) for GSM8K (top) and HumanEval (bottom), comparing ODD with LLaDA to the standard sampling ($\alpha = 0$, as indicated by dashed lines). ODD increases diversity at low temperatures and acts as a coherence filter at high temperatures (evidenced by the improved Pass@16 in Table 1), effectively balancing exploration and quality.

In Figure 3, we analyse the semantic diversity of the generated batches using the average cosine similarity between sentence embeddings. We observe an interesting dynamic behaviour in ODD depending on the temperature regime:

- **Low Temperature** ($\tau \leq 1.0$)**:** The baseline exhibits high similarity, with mode collapse in lower temperatures. ODD significantly reduces similarity, forcing the model to explore diverse paths even when the underlying probability distribution is sharp.
- **High Temperature** ($\tau \geq 1.5$)**:** The baseline naturally produces diverse but often incoherent outputs (lower similarity). Interestingly, ODD tends to *reduce* diversity compared to the

baseline in this regime. This suggests our quality-weighted repulsion acts as a filter, guiding the model back towards high-confidence regions when the noise level is excessive.

This demonstrates that ODD naturally trades off diversity and quality, by adaptively promoting diversity when needed and also promoting coherence when diversity is already sufficient.

B Pareto Efficiency and Trade-offs

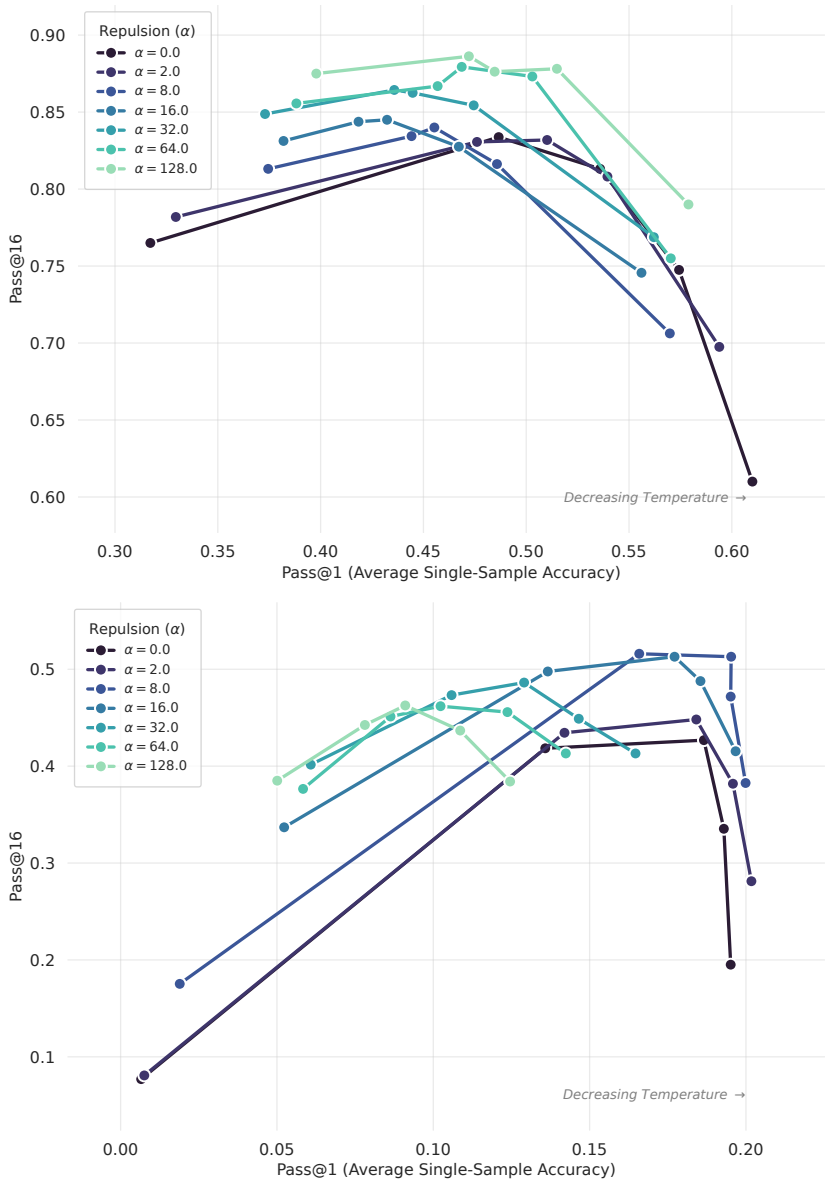

Figure 4: Pass@1 vs Pass@16 Pareto frontiers. $\alpha = 0$ represents standard LLaDA sampling (baseline). Data points represent decreasing temperatures from left ($\tau = 2$) to right ($\tau = 0$). **Top (GSM8K):** ODD trades off individual accuracy (shifting left) for superior batch coverage (shifting up). **Bottom (HumanEval):** For $\alpha \leq 16$, ODD achieves a Pareto improvement, boosting coverage without loss of quality.

Figure 4 visualises the efficiency of our sampling strategy with LLaDA by plotting individual sample quality (Pass@1, as determined by the average correctness in a batch) against batch coverage (Pass@16).

- **HumanEval (Pareto Improvement):** For code generation, moderate repulsion ($\alpha \leq 16$) yields a vertical lift over the baseline. We achieve significantly higher coverage (Pass@16) without degrading the quality of individual samples (Pass@1). This indicates that ODD provides a Pareto improvement over baseline for lower alpha values.
- **GSM8K (Exploration Trade-off):** For reasoning tasks, we observe a shift up and to the left. To maximise batch coverage (Pass@16), ODD sacrifices some individual sample accuracy (Pass@1). This confirms that the model is being forced to explore less probable and riskier reasoning paths. However, the net result is a higher probability of finding the correct answer within a single batch, a capability the baseline cannot match even at high temperatures.

C Empirical Pass@k

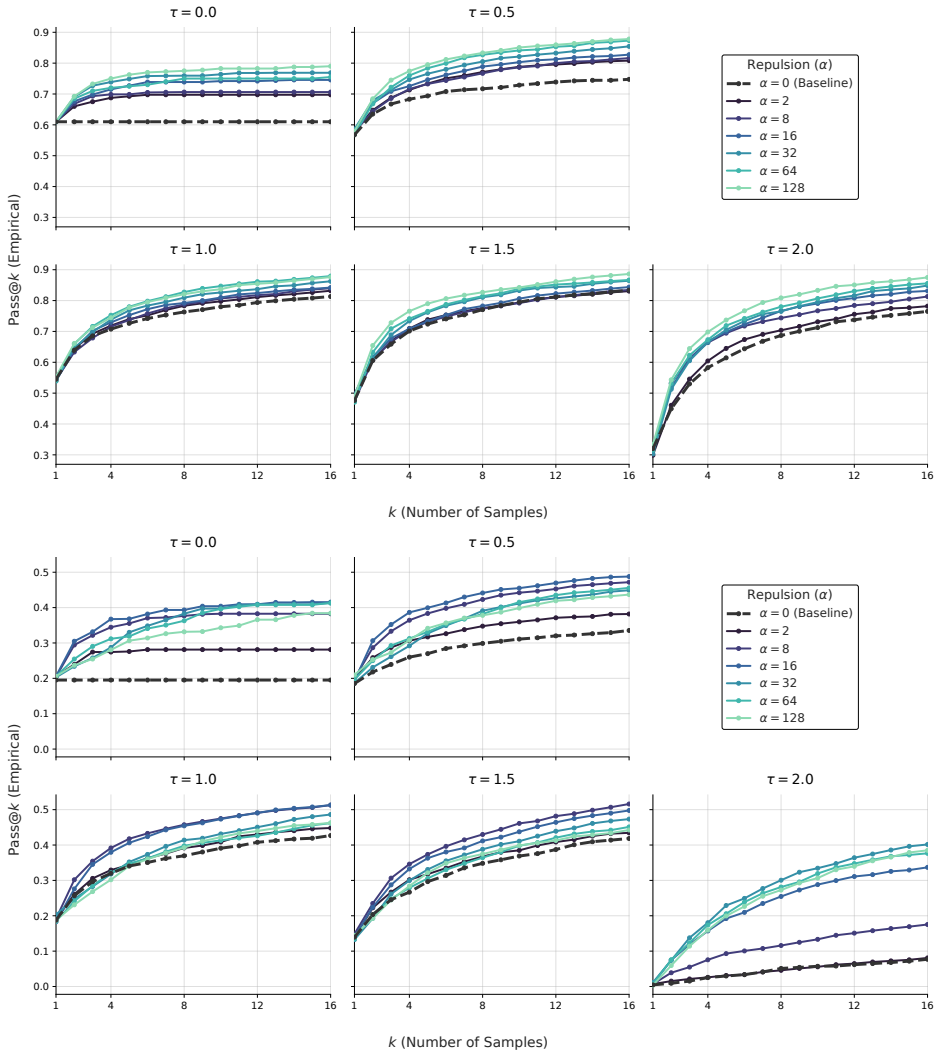

Figure 5: Empirical Pass@k for ODD with LLaDA over GSM8K (top) and HumanEval (bottom). We observe a consistent improvement relative to baseline for ODD. Given the batch size invariance of ODD, we expect this trend would continue for higher batch sizes.

Figure 5 illustrates the empirical Pass@k scaling behaviour of ODD with LLaDA compared to standard baseline sampling across both the GSM8K and HumanEval benchmarks. Empirical Pass@k measures the probability that at least one correct solution is found within a generated subset of k samples, where we take the first k out of the 16 samples from our experiments in Section 3.1.

As observed in the baseline curves, standard independent sampling suffers from severe diminishing returns as k scales. Because the samples are generated independently, the model frequently collapses into identical failure modes, causing the curves to plateau early. In contrast, ODD maintains a significantly steeper trajectory as k increases. By explicitly penalising redundancy and forcing the model into orthogonal subspaces, ODD pushes each subsequent sample towards directions unexplored by its predecessors, which keeps the curve rising.

Given the batch size invariance of ODD, the curves shown for $k \leq 16$ are, in exact arithmetic, identical to the prefix of the curve for any larger batch size. This predictable scaling property suggests that ODD will continue to efficiently expand the explored solution space as compute budgets increase.

D DiverseFlow Baseline

Algorithm 2: Joint Batch Optimisation (DiverseFlow [18])

Input : Model logits $X = [\mathbf{x}_1, \ldots, \mathbf{x}_N]$, step size α, diffusion schedule γ, diffusion timestep t, feature extractor F, numerical jitter ϵ

Output : Updated logits X'

`// Compute normalised features and qualities`

$V, Q \leftarrow F(X)$; `//` $V \in \mathbb{R}^{N \times d}$, $Q \in \mathbb{R}^N$

$\alpha_t \leftarrow \gamma(t) \cdot \alpha$

`// Construct the DPP L-ensemble matrix`

$K \leftarrow VV^\top$; `// Feature similarity kernel`

$Q_{\text{mat}} \leftarrow QQ^\top$; `// Quality outer product,` $Q_{\text{mat}}[i,j] = q_i q_j$

$L \leftarrow K \odot Q_{\text{mat}}$; `// Element-wise scaling`

`// Compute the negative log-likelihood of diversity`

$$\mathcal{L}_{\text{joint}} \leftarrow -\Big(\log\det(L + \epsilon I) - \log\det\big(L + (1+\epsilon)I\big) \Big)$$

`// Update all logits simultaneously via full backprop`

$X' \leftarrow X - \alpha_t \cdot \nabla_X \mathcal{L}_{\text{joint}}$

return X'

To evaluate our approach against previous global diversity methods, we adapt the global optimisation strategy of DiverseFlow [18], originally proposed for continuous image diffusion. The methodology is detailed in Algorithm 2.

This baseline models the diversity of the generated batch using a Determinantal Point Process [10] kernel. Given the extracted feature vectors V and quality scores Q derived from the model logits, the algorithm constructs an L-ensemble Gram matrix L. This matrix is formed via the element-wise multiplication of the pure feature similarity kernel ($K = VV^\top$) and the quality outer product (Q_{mat}), weighting the pairwise similarities by the samples' respective confidence scores. The objective $\mathcal{L}_{\text{joint}}$ then minimises the negative log-determinant of L (stabilised by a numerical jitter ϵ), which geometrically corresponds to maximising the N-dimensional volume enclosed by the batch feature vectors.

E Computational Overhead

To comprehensively evaluate the computational overhead of ODD, we profiled generation using the LLaDA-8B-Instruct model in both standard BF16 and 4-bit quantised precision. We measured latency and VRAM footprint scaling across batch size (N), generated sequence length (S), and diffusion steps (T), with 24 configurations for each problem in HumanEval and the 200 problem subset of GSM8K. The results are presented in Figure 6. Overall, ODD incurs minimal latency, effectively acting as a low cost intervention for scaling Pass@k. Across all configurations, the time overhead consistently remains below 10%. Importantly, the absolute memory and compute introduced by ODD are entirely independent of the base model. Although our approach is model agnostic, the relative latency penalty is slightly higher for the BF16 model. This corresponds to the speed decrease from quantisation overheads, with the base model being approximately 2.4s slower in the quantised setting.

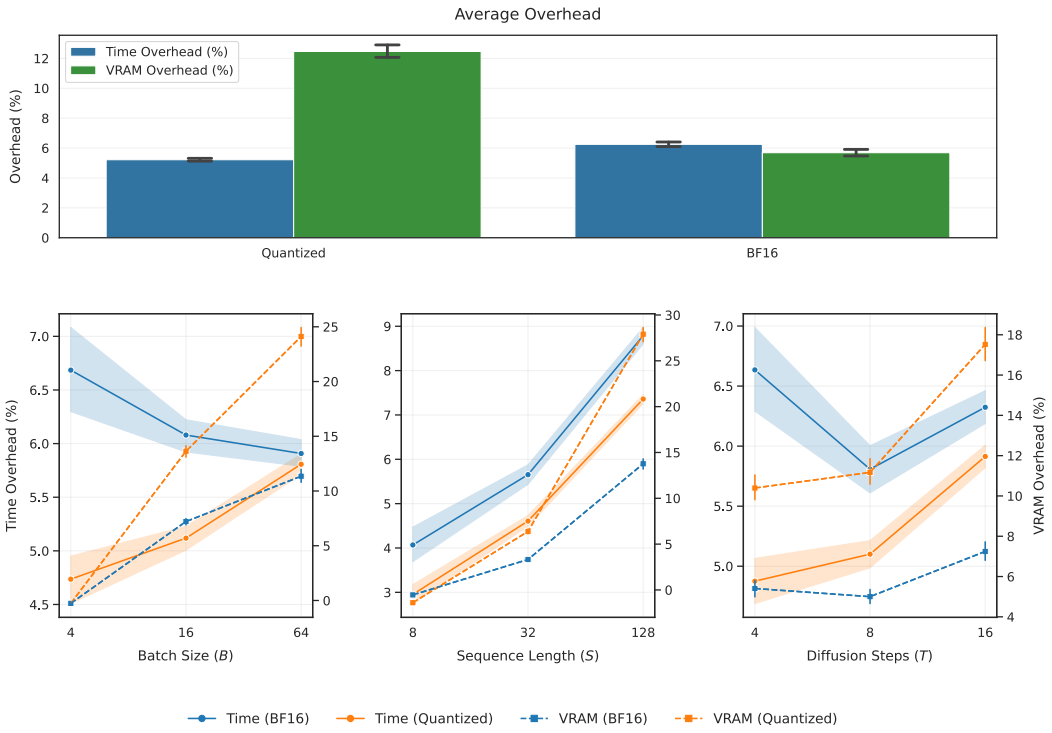

Figure 6: Computational overhead of ODD compared to standard sampling. **Top:** Global averages across all runs. **Bottom:** Scaling behaviour demonstrating time and memory overheads across batch size (N, labelled B in the figure), sequence length (S), and diffusion steps (T). ODD incurs less than a 10% relative time penalty, with memory scaling independently of the base model size.

As expected, the relative memory overhead for the larger BF16 model is below that of the quantised model.

We observe distinct scaling dynamics across the generation parameters. Because ODD sequentially tracks the history of the batch, its memory overhead scales at $O(N^2)$ with batch size (N). The VRAM penalty remains negligible ($5-15\%$) while time overhead remains flat. Scaling the generated sequence length (S) results in a linear increase in both time and memory overhead. This is driven by the requirements of maintaining and backpropagating through the $N \times S \times V$ graph during the feature extraction phase. Despite this linear scaling, the peak overhead at $S = 128$ remains highly manageable. It could be further alleviated by taking a subset of logits (e.g. top-k) rather than the whole vocabulary for each token, with small scale results in Appendix F showing this maintains performance. Finally, the relative time and memory overheads are invariant to the number of diffusion steps (T). Because the intervention is applied locally at each denoising timestep, its computational cost scales in proportion to the forward pass of the base model.

F Ablation of Feature Extraction Mechanisms

Table 6 shows various choices of feature extraction using ODD with LLaDA, fixing the optimal task specific hyperparameters (τ, α) established in Table 1. We compare our primary representation (max-pooled logits) against dense representations formed by taking the logit weighted average of the initial token embeddings. We additionally ablate the effect of restricting the penalty space to the Top-k most probable tokens.

Overall, the performance spread across all logit configurations is remarkably tight, demonstrating that ODD is highly robust to these feature choices. We observe a slight but consistent Pass@16 degradation when using logit-weighted embeddings, suggesting that raw logits provide a sharper, more direct probability signal for optimisation. Notably, restricting the logit space to the Top-k most

Table 6: Feature Extraction Ablation with LLaDA. Pure logit extraction gives a slight gain over logit-weighted embeddings, while restricting to the Top-k logits in extraction maintains performance.

Dataset	**Feature Target**	**Filter (Top-k)**	**Pass@16 (%)**
GSM8K	Logits (Max Pool)	All	88.6 ± 0.5
	Logits (Max Pool)	$k = 256$	88.2 ± 0.3
	Logits (Max Pool)	$k = 128$	87.8 ± 1.1
	Logits (Max Pool)	$k = 64$	88.0 ± 0.4
	Logits (Max Pool)	$k = 32$	$\mathbf{89.0 \pm 0.8}$
	Embeddings (Max Pool)	All	87.2 ± 0.7
HUMANEVAL	Logits (Max Pool)	All	51.6 ± 0.4
	Logits (Max Pool)	$k = 256$	$\mathbf{52.4 \pm 1.2}$
	Logits (Max Pool)	$k = 128$	51.5 ± 0.8
	Logits (Max Pool)	$k = 64$	50.8 ± 0.9
	Logits (Max Pool)	$k = 32$	52.0 ± 1.3
	Embeddings (Max Pool)	All	50.2 ± 1.2

probable tokens maintains peak performance, while reducing the required computational graph size and memory overhead. While the optimal k is mildly task-dependent, the overall minor variance suggests that Top-k filtering can provide an efficiency gain without compromising performance.

Table 7: Generalisation of Pass@16 performance for LLaDA across varying generation lengths and diffusion steps, using the best hyperparameters (τ, α) for each approach and benchmark found in Table 1. We report the mean and standard error across multiple runs.

Dataset	Steps	Approach	Generation Length		
			64	128	256
HUMANEVAL	16	Baseline	36.9 ± 0.3	23.5 ± 0.6	19.8 ± 1.8
		ODD	$\mathbf{45.3 \pm 0.6}$	$\mathbf{43.0 \pm 0.5}$	$\mathbf{23.9 \pm 1.1}$
	32	Baseline	42.7 ± 0.6	32.3 ± 0.9	30.3 ± 1.1
		ODD	$\mathbf{51.6 \pm 0.4}$	$\mathbf{52.0 \pm 1.6}$	$\mathbf{41.2 \pm 1.3}$
	64	Baseline	47.6 ± 0.9	38.2 ± 1.1	49.8 ± 1.0
		ODD	$\mathbf{58.2 \pm 0.3}$	$\mathbf{60.4 \pm 0.7}$	$\mathbf{52.8 \pm 2.0}$
GSM8K	16	Baseline	79.9 ± 1.0	$\mathbf{82.0 \pm 0.6}$	$\mathbf{78.5 \pm 0.9}$
		ODD	$\mathbf{81.1 \pm 0.9}$	78.8 ± 1.2	70.5 ± 0.6
	32	Baseline	83.4 ± 0.6	83.4 ± 1.2	78.2 ± 0.3
		ODD	$\mathbf{88.6 \pm 0.5}$	$\mathbf{90.0 \pm 0.6}$	$\mathbf{79.5 \pm 0.7}$
	64	Baseline	83.9 ± 0.7	80.8 ± 1.2	77.1 ± 0.4
		ODD	$\mathbf{90.1 \pm 0.6}$	$\mathbf{93.6 \pm 0.4}$	$\mathbf{89.9 \pm 0.5}$

G Generalisation Across Diffusion Steps and Generation Lengths

To evaluate the robustness of ODD, we test Pass@16 performance with LLaDA across varying diffusion step counts ($T \in \{16, 32, 64\}$) and generation lengths ($S \in \{64, 128, 256\}$), presented in Table 7. Overall, ODD proves highly robust and largely insensitive to these parameters. As expected, higher step counts yield better baseline and ODD performance. Fewer diffusion steps force larger, coarser token transitions per iteration, which restricts the fine grained denoising capacity. For GSM8K, this results in a drop in ODD performance for longer sequence lengths when $T = 16$, particularly for the $S = 256$ case, where a coarse 16 tokens are modified per iteration.

H Additional Results on New Models and Benchmarks

This appendix reports the experiments on new models and benchmarks referred to in Section 3.1. Unless stated otherwise, all experiments use the untuned default $\alpha = 16$ (Table 8 also reports $\alpha = 128$) and a single run per configuration (the two † cells of Table 8 and the HumanEval rows of Table 10 average two independent runs). We omit $\tau \in \{0, 2\}$, where the baseline degrades heavily, in favour of more configurations, as one full-GSM8K configuration takes ten to twelve GPU-hours on an RTX 3090.

Full GSM8K. Table 8 repeats the main experiment on all 1,319 GSM8K test problems with the prompt, answer extraction and decoding settings of Section 3. The ordering and magnitude of the gains match the 200-problem subset of Table 1: Pass@16 rises by 9.2, 2.8 and 0.9 points at $\alpha = 16$ and by 13.5, 7.0 and 3.9 points at $\alpha = 128$ for $\tau = 0.5$, 1.0 and 1.5, while Pass@1 moves by at most 1.4 points in either direction. Dream improves by 9.2 points at $\tau = 1.0$.

MATH-500. Table 9 uses all 500 problems of the MATH-500 split [15] of the MATH benchmark [8]. The prompt asks the model to reason step by step and place the final answer in `\boxed{}`. Because solutions are longer than for GSM8K, generations are 256 tokens (32 steps, batch 16), where the last boxed expression is checked for equivalence with the gold answer using `math-verify`. Pass@16 improves by 8.2, 7.4 and 3.0 points and Pass@1 also improves at every temperature.

LLaDA-1.5. Table 10 applies the unchanged code and settings (4-bit weights, 64 tokens, 32 steps, batch 16) to LLaDA-1.5 [31]. This is a version of LLaDA-8B-Instruct post-trained with VRPO. The post-trained model is a stronger baseline on GSM8K, yet ODD adds 7.5, 3.0 and 3.5 points of Pass@16 on GSM8K and 10.1, 3.0 and 9.1 points on HumanEval. This is comparable to its gains

Table 8: Full GSM8K test split (1,319 problems). Pass@16 and Pass@1 (%) for standard sampling and ODD with the paper's generation settings (64 tokens, 32 steps, batch 16). One run per cell except the two $\alpha = 16$ cells marked †, which average two independent runs. The Dream rows use its default confidence-based (`maskgit_plus`) unmasking at $\tau = 1.0$.

		Pass@16		
Model	τ	Baseline	ODD $\alpha = 16$	ODD $\alpha = 128$
LLaDA-8B-Instruct	0.5	72.4	81.6	85.9
LLaDA-8B-Instruct	1.0	80.4	83.2†	87.4
LLaDA-8B-Instruct	1.5	82.6	83.5†	86.5
Dream-v0-Instruct-7B	1.0	68.1	77.3	–
		Pass@1		
LLaDA-8B-Instruct	0.5	53.5	52.1	52.5
LLaDA-8B-Instruct	1.0	49.3	50.6†	50.0
LLaDA-8B-Instruct	1.5	45.1	44.6†	45.3
Dream-v0-Instruct-7B	1.0	41.6	40.6	–

Table 9: MATH-500 (all 500 problems), LLaDA-8B-Instruct. Pass@1 and Pass@16 (%) for standard sampling and ODD ($\alpha = 16$, no re-tuning), 256-token generations, 32 steps, batch 16, one run per cell. Answers are the last `\boxed{}` expression, checked for equivalence with `math-verify`.

	Pass@1		Pass@16	
τ	Baseline	ODD	Baseline	ODD
0.5	18.2	20.6	34.8	43.0
1.0	15.6	20.0	40.8	48.2
1.5	15.2	16.2	45.8	48.8

on the base model. RL post-training sharpens the prior, but the samples of a batch are still drawn independently, so the redundancy that ODD removes remains.

RND1-30B. Table 11 applies ODD to RND1-Base-0910 [21]. This is a larger 30B diffusion model converted from an autoregressive Qwen3-30B-A3B checkpoint. We run it in bf16 with its native entropy-based unmasking rule, 64-token generations, 32 steps and batch 16. We apply ODD unchanged to the logits of the generated region. RND1 is far more temperature sensitive than LLaDA: at $\tau \leq 0.1$ a baseline batch of 16 contains one or two distinct generations, at $\tau = 0.5$ nearly all 16 are distinct, and at $\tau \geq 1.0$ both arms produce syntactically degraded text whose Pass@1 roughly halves. We therefore evaluate the usable range $\tau \in \{0.1, 0.3, 0.5\}$ and report $\tau = 1.0$ as the degraded regime, where ODD is neutral on noise-dominated logits. The gains are largest where the baseline collapses and shrink as sampling noise restores baseline diversity, following the pattern of Section 3.1. The bottom block doubles the number of steps and the generation length on the first 50 GSM8K problems at $\tau = 0.3$. The gain persists, showing it is not an artefact of the short decoding budget.

I Choosing α: Robustness and Transfer

Tables 12 and 13 quantify the robustness claims of Section 3.1, computed from the runs behind Table 1 (Dream with its default confidence-based unmasking). Of the 100 (α, τ) cells in Table 1, 96 improve on the baseline. The mean improvement is $+11.4$ Pass@16 points and the worst cell is -2.4 (Dream, HumanEval, $\alpha = 128$, $\tau = 2.0$), where statistics are computed from the run-level results underlying Table 1. The optimal α is larger for GSM8K, whose lenient final-answer evaluation tolerates aggressive exploration, than for HumanEval, where syntactic correctness is required. $\alpha = 16$ lies in the high-gain region of both and is the default we recommend and use for every new model and benchmark in Appendix H.

Table 10: ODD on LLaDA-1.5, the RL (VRPO) post-trained version of LLaDA-8B-Instruct, with the paper's $\alpha = 16$ and no re-tuning. Pass@1 / Pass@16 (%), 64-token generations, 32 steps, batch 16. GSM8K uses the paper's first-200-problem subset (one run). HumanEval values are the mean of two independent runs.

		Baseline		ODD $\alpha = 16$	
Benchmark	τ	Pass@1	Pass@16	Pass@1	Pass@16
GSM8K (200)	0.5	59.5	73.0	55.5	80.5
GSM8K (200)	1.0	56.0	81.5	51.0	84.5
GSM8K (200)	1.5	50.0	81.0	52.0	84.5
HumanEval (2 runs)	0.5	25.0	38.7	24.1	48.8
HumanEval (2 runs)	1.0	23.2	47.0	25.0	50.0
HumanEval (2 runs)	1.5	14.3	42.7	14.0	51.8

Table 11: ODD on RND1-Base-0910, a 30B masked diffusion model, with the paper's $\alpha = 16$ and no re-tuning. We report Pass@1 / Pass@16 (%), with 64-token generations, 32 steps, batch 16. RND1 is far more temperature-sensitive than LLaDA (samples are near-identical at $\tau \leq 0.1$ and syntactically degraded at $\tau \geq 1.0$), so we use $\tau \in \{0.1, 0.3, 0.5\}$. The $\tau = 1.0$ rows show the degraded regime. The bottom block varies the decode budget on the first 50 GSM8K problems at $\tau = 0.3$.

		Baseline		ODD ($\alpha = 16$)	
Benchmark	τ	Pass@1	Pass@16	Pass@1	Pass@16
GSM8K (200)	0.1	44.5	44.5	44.5	63.5
GSM8K (200)	0.3	47.0	56.0	47.5	68.5
GSM8K (200)	0.5	45.0	70.0	42.0	72.0
GSM8K ($n = 50$)	1.0	24.0	68.0	18.0	70.0
HumanEval (164)	0.1	26.2	26.8	25.0	53.7
HumanEval (164)	0.3	28.7	37.8	28.0	60.4
HumanEval (164)	0.5	22.6	50.0	26.8	61.0
HumanEval (164)	1.0	10.4	53.7	9.1	56.7
Decode budget (GSM8K, first 50 problems, $\tau = 0.3$; tokens/steps)					
GSM8K ($n = 50$)	64/32	42.0	44.0	38.0	58.0
GSM8K ($n = 50$)	64/64	50.0	56.0	44.0	78.0
GSM8K ($n = 50$)	128/64	40.0	64.0	52.0	74.0

J Overgeneration and Filtering Protocol

For each HumanEval problem and temperature $\tau \in \{0.5, 1.0, 1.5\}$ we generate a pool of M baseline samples with the decoding settings of Section 3 (LLaDA, 32 steps, 64 tokens), drawn as batches of at most 16 so that every sample is produced under exactly the conditions of Table 1. The *first-16* control is the empirical Pass@16 of the first 16 samples of the pool. The *filtered* score embeds all M samples with the all-MiniLM-L6-v2 sentence encoder [22, 27], selects 16 of them by greedy k-centre (farthest-first) selection [6] on cosine distance, and reports Pass@16 of the selected set. This is the strongest selection rule that uses no information about correctness. The *oracle* counts a problem as solved if any sample in the pool is correct, i.e. Pass@M, and upper-bounds every possible selection rule, including learned rerankers. $M = 17$ matches the wall-clock time of ODD (whose overhead is at most 6.7% at batch size 16, Table 2), and $M = 32$ corresponds to roughly twice its compute. All pools are single runs over the 164 problems.

K Diversity Metrics and the LLM-Judge Study

Metrics. All diversity statistics are computed from the stored generations of the runs in Table 1, with one *problem-batch* (the 16 samples of one run on one problem) as the unit of analysis. LLaDA aggregates 8 runs (1,600 GSM8K and 1,312 HumanEval problem-batches per configuration) and Dream 4 runs. The *Vendi score* [5] of a batch can be read as the effective number of distinct samples in the batch. We use a semantic kernel (cosine similarity of all-MiniLM-L6-v2 sentence embeddings)

Table 12: Robustness of a fixed $\alpha = 16$ (Pass@16 gain over the baseline in points, averaged over the five temperatures of Table 1). *Regret* is the gap to the best α for each temperature (oracle), mean and maximum over temperatures.

Model	Task	Mean gain	Min gain	Regret (mean / max)	Oracle α per τ (0, 0.5, 1, 1.5, 2)
LLaDA	GSM8K	$+6.4$	$+1.0$	4.4 / 5.1	128, 128, 64, 128, 128
LLaDA	HumanEval	$+16.0$	$+7.9$	1.7 / 6.5	16, 16, 16, 8, 32
Dream	GSM8K	$+9.7$	-1.6	5.2 / 8.6	128, 64, 128, 128, 128
Dream	HumanEval	$+17.7$	$+6.6$	1.4 / 4.3	64, 64, 128, 8, 16

Table 13: Cross-setting transfer of a single tuned α. The value α^* maximising the mean gain on the source setting is applied unchanged to the target setting. Gains are Pass@16 points over the baseline, averaged (mean) or minimised (min) over the five temperatures, and regret is relative to the target's per-temperature oracle. Across the 12 pairs the mean gain is $+12.2$ points, the worst pair mean is $+6.4$, the worst single temperature is -2.4 and the mean regret is 3.4 points.

Source (α^*)	Target	Mean gain	Min gain	Mean regret
LLaDA/GSM8K (128)	LLaDA/HumanEval	$+13.2$	$+2.4$	4.5
LLaDA/GSM8K (128)	Dream/GSM8K	$+14.3$	$+3.8$	0.6
LLaDA/GSM8K (128)	Dream/HumanEval	$+14.3$	-2.4	4.8
LLaDA/HumanEval (16)	LLaDA/GSM8K	$+6.4$	$+1.0$	4.4
LLaDA/HumanEval (16)	Dream/GSM8K	$+9.7$	-1.6	5.2
LLaDA/HumanEval (16)	Dream/HumanEval	$+17.7$	$+6.6$	1.4
Dream/GSM8K (128)	LLaDA/GSM8K	$+10.7$	$+5.3$	0.1
Dream/GSM8K (128)	LLaDA/HumanEval	$+13.2$	$+2.4$	4.5
Dream/GSM8K (128)	Dream/HumanEval	$+14.3$	-2.4	4.8
Dream/HumanEval (16)	LLaDA/GSM8K	$+6.4$	$+1.0$	4.4
Dream/HumanEval (16)	LLaDA/HumanEval	$+16.0$	$+7.9$	1.7
Dream/HumanEval (16)	Dream/GSM8K	$+9.7$	-1.6	5.2

and a lexical kernel (cosine similarity of TF-IDF character 2–4-gram vectors). The latter kernel is used to show that gains are not an artefact of the embedding model. *Distinct correct solutions* are the connected components of the correct samples of a batch under normalised token edit distance ≤ 0.3. For HumanEval we additionally parse each correct program, strip comments and docstrings, canonicalise it through its abstract syntax tree and count exact duplicates (*AST-distinct*). *Pairwise edit distances* are token-level Levenshtein distances between pairs of correct samples in a batch. *Novelty* is computed for problems that ODD solves in some run but the baseline solves in none of its runs at the same temperature. It is the mean over such solutions of the minimum normalised edit distance to any baseline sample for that problem.

Table 14 extends Table 5 to all temperatures. Two patterns stand out. First, the Vendi score of the baseline grows with temperature only because high-temperature text is incoherent (its correctness collapses, most visibly on HumanEval at $\tau = 2$). At every temperature up to 1.0 ODD instead raises Vendi scores and the number of distinct correct solutions at the same time. At $\tau = 1.5$ it is acting as the coherence filter seen in Appendix A, lowering the semantic Vendi score while still raising the number of distinct correct solutions. Second, the number of correct samples per batch falls modestly on GSM8K (the Pass@1 trade-off of Figure 4) and is unchanged on HumanEval. Pass@16 and the number of distinct correct solutions rise because the baseline spends its correct samples on copies of one solution to easy problems, while ODD redistributes them over more problems and more solutions.

LLM-judge protocol. The study uses one run pair per benchmark from Table 1 (LLaDA, $\tau = 1.0$, baseline versus $\alpha = 16$), giving 364 items (200 GSM8K, 164 HumanEval). For each item the two 16-sample batches were stripped of any identifying information and assigned to the labels A and B randomly (ODD was A on 200 of 364 items). Each judgment was made by Claude (model `claude-fable-5`) in a fresh, isolated context containing only the prompt below. Hence no item could influence another and the answer key was never available to the judge. As quality controls, 10 items were duplicated verbatim and 20 were duplicated with A and B swapped, giving 394 judgments in total. The prompt was:

```
You are judging solution diversity for a math/coding benchmark.
```

Table 14: Diversity statistics across temperatures for LLaDA and Dream ($\alpha = 16$ versus the baseline), means over problem-batches. Vendi scores use the semantic (sem.) and n-gram kernels; *correct* is the mean number of correct samples per batch of 16; *distinct correct* is the mean number of edit-distance clusters among correct samples (AST-distinct programs for HumanEval in parentheses); ≥ 2 is the fraction of batches with at least two distinct correct solutions. At $\tau = 0$ the baseline Vendi score is exactly 1 (16 identical samples). At high temperature the baseline's Vendi score is inflated by incoherent text while its correctness collapses, whereas ODD adds diversity and correctness together.

		GSM8K				HUMANEVAL			
τ	Method	Vendi (sem. / n-gram)	Correct	Distinct correct	≥ 2	Vendi (sem. / n-gram)	Correct	Distinct correct	≥ 2
LLaDA-8B-Instruct ($\alpha = 16$ versus baseline; 8 runs, 1,600 GSM8K and 1,312 HumanEval problem-batches)									
0.0	Baseline	1.00 / 1.00	9.76	0.61	0.00	1.00 / 1.00	3.12	0.20 (0.20)	0.00
	ODD	1.46 / 3.04	8.90	2.19	0.57	1.88 / 3.16	3.15	0.70 (1.18)	0.20
0.5	Baseline	1.47 / 2.89	9.19	1.92	0.44	1.44 / 2.00	3.09	0.42 (0.62)	0.07
	ODD	1.95 / 5.46	7.48	4.13	0.71	2.33 / 4.82	2.97	0.98 (1.59)	0.28
1.0	Baseline	1.90 / 4.34	8.57	3.03	0.60	2.14 / 3.52	2.98	0.65 (1.09)	0.16
	ODD	2.17 / 6.14	6.92	4.74	0.75	2.79 / 6.31	2.83	1.29 (1.86)	0.34
1.5	Baseline	2.89 / 6.53	7.79	4.61	0.71	3.82 / 7.36	2.17	0.74 (1.02)	0.19
	ODD	2.58 / 6.87	6.70	5.12	0.76	3.23 / 7.34	2.19	1.33 (1.70)	0.34
2.0	Baseline	5.74 / 12.38	5.08	4.96	0.66	7.30 / 13.25	0.11	0.09 (0.09)	0.01
	ODD	4.10 / 9.81	6.11	5.72	0.74	5.11 / 10.57	0.84	0.70 (0.74)	0.19
Dream-v0-Instruct-7B ($\alpha = 16$ versus baseline; 4 runs, 800 GSM8K and 656 HumanEval problem-batches)									
0.0	Baseline	1.00 / 1.00	6.80	0.43	0.00	1.00 / 1.00	3.80	0.24 (0.24)	0.00
	ODD	1.60 / 3.52	6.25	2.06	0.46	1.39 / 3.24	3.74	1.06 (1.81)	0.31
0.5	Baseline	1.36 / 2.39	7.28	1.18	0.27	1.11 / 1.62	3.48	0.36 (0.52)	0.03
	ODD	1.87 / 4.77	7.20	3.56	0.63	1.48 / 3.90	3.48	1.04 (1.80)	0.29
1.0	Baseline	1.68 / 4.04	6.99	2.41	0.48	1.37 / 2.99	3.48	0.70 (1.07)	0.13
	ODD	2.08 / 5.89	5.40	4.00	0.65	1.65 / 5.19	3.65	1.50 (2.58)	0.39
1.5	Baseline	1.97 / 5.47	6.15	3.53	0.59	1.58 / 4.21	3.04	0.95 (1.45)	0.26
	ODD	2.20 / 6.39	4.76	3.78	0.64	1.81 / 5.85	3.38	1.64 (2.64)	0.44
2.0	Baseline	2.28 / 6.78	4.98	3.88	0.61	1.82 / 5.45	2.19	0.96 (1.39)	0.27
	ODD	2.31 / 6.84	4.44	3.75	0.63	1.93 / 6.20	2.88	1.60 (2.30)	0.41

```
Below is one problem and two batches of 16 model-generated attempts,
labelled Batch A and Batch B. The labels are arbitrary and carry no
information.

Question: which batch contains more genuinely DISTINCT solution
approaches? Judge distinctness of approach (different reasoning
paths, different algorithms/structures), not surface rewording,
formatting, or correctness. If the two batches are comparable,
answer TIE.

Reply with ONLY a JSON object on one line, no other text:
{"verdict": "A" | "B" | "TIE", "reason": "<one concise sentence>"}
```

Results. The judge preferred ODD on 227 items and the baseline on 12, with 125 ties, i.e. ODD won 95.0% of the 239 decided comparisons (Wilson 95% interval 91.4–97.1%). Per benchmark the results in favour of ODD were: 90/97 (92.8%) on GSM8K and 137/142 (96.5%) on HumanEval. Decided verdicts were balanced between the labels (129 A versus 110 B), with 9 of the 10 verbatim duplicates receiving identical verdicts. 18 of the 20 swapped duplicates flipped as required, showing position bias was not driving the result. Interestingly, where both the judge and the edit-distance clustering of Table 5 pick a side (126 items) they agree in 75.4% of cases. Most disagreements are items where the judge credits ODD for diversity among all 16 attempts, whereas clustering counts only correct ones. The judge is also stricter, calling 125 ties, because the prompt requires a different reasoning approach rather than a different edit-distance cluster.

L Computational Usage Summary

We ran our experiments on 3 internal machines, each with an Intel Xeon-W223 CPU, and across all machines a total of 4 Ada A6000 and 2 A6000 GPUs with 48 GB of VRAM each. We had two workers for each GPU giving a total of 12 GPU processes. Each run configuration was assigned a

single GPU and CPU worker. Therefore a single 24GB GPU would be sufficient to replicate the individual runs from our results.

For the main results, LLaDA had 8 runs per configuration, 5 temperature settings and 6 alpha values, giving us 40 runs for the baseline and 240 for both the DPP and ODD results, for a total of 520 runs each for GSM8K and HumanEval. For GSM8K, the average time per run was approximately 1.2 hours and for HumanEval was 1.5 hours. This gives around 1404 worker hours, and with 2 workers per GPU gives 702 GPU hours for the LLaDA results. For Dream, we ran 4 runs 5 temperature settings and 4 alpha values, giving 20 runs for the baseline and 80 for ODD, for a total of 100 runs for each benchmark, with an average time per run of 0.9 hours for both benchmarks. This gives 180 worker hours and 90 GPU hours for the Dream results. This gives a total of approximately 792 GPU hours on our hardware. There were additional small scale preliminary experiments in developing the approach, which we do not have a precise estimate for.

The experiments added after review (Appendices H–K) were run on rented cloud GPUs (a node with four RTX 3090 24 GB, a single RTX 4090, an RTX 5880 Ada 48 GB, two RTX PRO 5000 48 GB, and one or two RTX PRO 6000 96 GB for the 30B RND1 model in bf16) and a local RTX 4090. Summing the generation time of the runs that enter the tables gives about 280 GPU-hours. The diversity re-analysis and the LLM-judge study used only the stored generations and no additional GPU time.

M Licenses for Existing Assets

In this work, we use several open models and datasets for our evaluation, respecting their original licenses and terms of use. Specifically, LLaDA, HumanEval, and GSM8K are distributed under the MIT License. The Dream model is released under the Apache License 2.0. Of the assets added after review, the MATH dataset [8] and its MATH-500 split [15] are released under the MIT License, the LLaDA-1.5 weights [31] under the MIT License, the RND1-Base-0910 weights and inference code [21] under the Apache License 2.0, the all-MiniLM-L6-v2 sentence encoder [22, 27] under the Apache License 2.0, and the Vendi Score reference implementation [5] under the MIT License.

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 claims match the experimental results presented in Section 3, demonstrating improved Pass@k with minimal overhead.

   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: Limitations are explicitly discussed in a dedicated paragraph at the end of the Conclusion section.

   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: [N/A]

Justification: The paper proposes an algorithmic intervention and empirical evaluation, it does not include formal theoretical claims or mathematical 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: Full experimental details (models, hyperparameter grids, benchmarks) are provided in Section 3, and all code and run logs are publicly available at `https://github.com/sean-lamont/odd`.

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: Code, configuration files, per-sample generations and metrics for all experiments are publicly available at `https://github.com/sean-lamont/odd`.

   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 experimental setup, including benchmark subsets, hyperparameters, and generation parameters, are outlined in the text and fully detailed in the provided code.

   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: [Yes]

   Justification: Results in Table 1, Table 2, and all Appendix tables report the mean and Standard Error (SE) across multiple runs.

   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: We include a dedicated Appendix section outlining the computer resources used.

   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: The research conforms to the NeurIPS Code of Ethics.

   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: Positive societal impacts are discussed in the introduction (e.g., improving the sample efficiency of complex reasoning and code generation tasks, saving resources). As our approach only augments sampling of existing models, we find no clear direct negative societal impacts beyond those of the base model.

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: The paper does not release new high risk datasets or pretrained models requiring specific safeguards, we only introduce a sampling algorithm for existing open weight models.

    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: All assets are properly credited, with license terms respected and explicitly mentioned in an Appendix section.

    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: The code introduced is documented and provided in the supplementary repository.

    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 research does not involve crowdsourcing or human subjects.

    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 research does not involve human subjects.

    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: [Yes]

    Justification: The usage of LLMs in the core method is fully described (e.g. Section 2.1), with our primary contribution being the improvement of sampling in Diffusion Language Models. An LLM was additionally used as a blinded judge of batch diversity, with the protocol and prompt given in Appendix K.

    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.