LATENT-PLAN BLOCK DIFFUSION: SAMPLING A BLOCK-LEVEL INTENT FOR FEW-STEP PARALLEL DECODING

Anonymous authors
Paper under double-blind review

ABSTRACT

Block diffusion language models such as BD3-LM use a masked-diffusion denoiser to decode a block of B tokens in parallel while keeping the key–value (KV) cache of autoregressive models, but every denoising step samples the block from a product of per-token marginals. We show that the least error of drawing a block in one such step is its conditional total correlation given the context, a cost that unmasking schedules can only trade for more steps. *Latent-Plan Block Diffusion* (LP-BD) supplies the missing shared variable: before a block is decoded, a small conditional flow samples a low-dimensional *plan* given the generated prefix, and the denoiser decodes the block conditioned on it. The plan is learned from the denoiser's own loss, without supervision, and LP-BD equals BD3-LM at initialization, so it can be added to a pretrained model. Trained from scratch on TinyStories with 16-token blocks, LP-BD lowers the negative log-likelihood of one-step samples under an autoregressive judge by 31% and reaches 16-step BD3-LM quality with 8 steps. The gains are similar at 28M and 85M parameters. Added to the released 170M BD3-LM checkpoints on OpenWebText, LP-BD lowers generative perplexity at matched entropy by 14–55% at every step budget of a fixed-step sampler, and by 38–66% with guidance. The plan acts as a sampled *intent* shared by the block rather than a prediction from the context, and its gains require steps that decode several tokens jointly.

1 INTRODUCTION

Diffusion language models promise parallel generation, but in practice fewer denoising steps cost quality. Block diffusion (BD3-LM; Arriola et al., 2025a) interpolates between autoregressive (AR) and diffusion language models: a sequence is generated block by block, autoregressively across blocks and by masked diffusion within a block, which restores the key–value (KV) cache and variable-length generation of AR models while still decoding the B tokens of a block in parallel. A growing line of work asks *when* to decode tokens in parallel: adaptive block sizes (Lu et al., 2026a; Huang & Tang, 2025), confidence-based decoding (Wu et al., 2026; Liu et al., 2025b), dilated schedules (Luxembourg et al., 2026), or set-based decoding (Gat et al., 2025).

We start from a different observation. Within a block, every denoising step of a masked diffusion model outputs one distribution per masked position and samples the positions independently. Given the context c (the prefix of blocks generated so far), even an optimal denoiser therefore samples the next block x^b in one step from the product of its marginals, and the best any single parallel step can do, in Kullback–Leibler (KL) divergence, is (Proposition 1)

$$\min_{q \text{ product}} D_{\mathrm{KL}}\big(p(x^b \mid c) \,\|\, q\big) = \mathrm{TC}(x^b \mid c),$$

the conditional *total correlation* (Watanabe, 1960) $\mathrm{TC}(x^b \mid c) = \sum_{i=1}^{B} H(x_i \mid c) - H(x^b \mid c)$ of the block's tokens $x_1, \ldots, x_B$, i.e., the information they share given the context (H is the entropy). This gap is not a scheduling problem but a *missing-variable* problem. If a latent variable carried the shared part, the tokens would be nearly independent given it and could be sampled in one step.

Latent-Plan Block Diffusion (LP-BD) adds exactly such a variable to BD3-LM. Before a block is decoded, a small conditional flow samples a low-dimensional *plan* z_b given the hidden state at

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

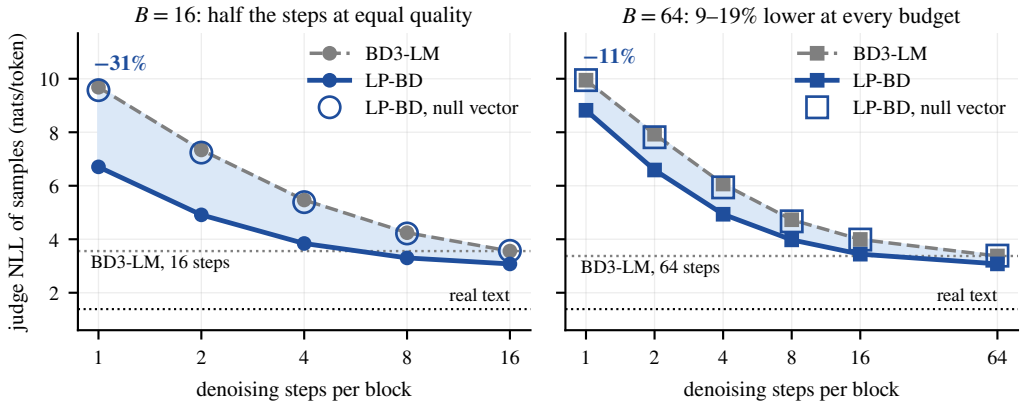

Figure 1: Judge negative log-likelihood (NLL) versus denoising steps on TinyStories, without guidance (lower is better, three-seed means). Hollow markers: sampling with the null vector.

the end of the prefix, and the denoiser then decodes the block conditioned on z_b. The plan is not supervised: an encoder pools the clean block's hidden states into z_b, the denoiser is trained to decode the block from a noised copy of z_b (*generative grounding*), and the flow is trained to match the output of an exponential-moving-average (EMA) target encoder. Two zero-initialized linear maps inject the plan, so LP-BD equals BD3-LM at initialization. Replacing the plan by a learned *null vector* $z_\varnothing$ gives a null path of the same form as BD3-LM, used for guidance and as a control. Unlike latents redrawn at every denoising step (Xie et al., 2026) or generated by a 400M latent diffusion model (Uziel et al., 2026), the plan is drawn once per block by a small flow from the backbone's hidden state, stays fixed while the block is denoised, and, with the KV cache, needs no extra backbone pass.

Intuition. The plan does not *predict* the next block. It is drawn once per block, as "what this block is going to say," and all positions then fill in words that agree with it. After *"Tom found a,"* one plan commits to *"little dog in the garden"* and another to *"shiny coin under the bed."* All positions of the block must agree on one, which independent per-position sampling cannot enforce.

Figure 1 summarizes the outcome on TinyStories (Eldan & Li, 2023): the negative log-likelihood (NLL) that an AR judge assigns to one-step samples is 31% lower for LP-BD than for BD3-LM at B=16, and 8-step LP-BD outperforms 16-step BD3-LM, while the same checkpoints sampled with the null vector nearly recover the BD3-LM curve. Because the plan enters through zero-initialized maps, LP-BD can also be added to a pretrained model: fine-tuned for 20k steps from the released 170M BD3-LM checkpoints on OpenWebText, it lowers generative perplexity at matched entropy by 14–55% at every step budget of a fixed-step sampler (38–66% with guidance). The gains require steps that decode several tokens jointly: with BD3-LM's own sampler, which decodes one token per step, unguided LP-BD is worse than BD3-LM on OpenWebText.

Our contributions are:

- **A characterization.** The least single-step error of block-parallel decoding equals the conditional total correlation of the block. We decompose the expected total correlation given any latent, which upper-bounds the error of one-step sampling with that latent, and give an example in which a latent increases it.
- **LP-BD.** A stochastic block-level plan sampled by a conditional flow and learned by generative grounding with an EMA target and a null vector. It is an additive modification of BD3-LM that equals it at initialization, can be added to a pretrained checkpoint, and has a position-aware sub-block variant for large blocks.
- **An evaluation protocol for parallel decoders.** Generative perplexity compared at matched entropy rather than at a fixed nucleus threshold, which favors less diverse samplers, and a fixed-step sampler without the undefined final step.
- **Results and mechanism.** Gains at every step budget of a fixed-step sampler on TinyStories and text8, and on OpenWebText when added to a pretrained 170M model. The plan must

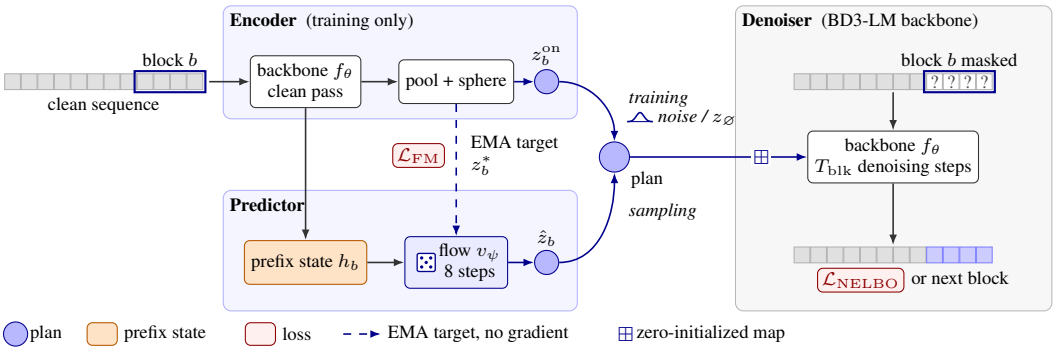

Figure 2: Overview of LP-BD. Training: the denoiser decodes from the noised online plan (or $z_\varnothing$), and the flow fits the EMA-target plan. Sampling: a flow sample replaces the online plan.

be sampled once per block and acts as an intent shared by the block rather than a prediction from the context. The gains are larger where the plan leaves less entropy and require steps that decode several tokens jointly.

2 Block diffusion and the product-of-marginals sampler

Masked diffusion. A masked (absorbing-state) diffusion language model (Austin et al., 2021; Lou et al., 2024; Sahoo et al., 2024; Shi et al., 2024) corrupts a clean sequence x_0 into x_t by masking each token independently with probability $t \in [0, 1]$ and trains a denoiser $p_\theta(x_0 \mid x_t)$ that predicts every masked token. For a linear schedule, the negative evidence lower bound (NELBO) is a $1/t$-weighted cross-entropy on the masked positions. Each reverse step unmasks a subset of masked positions with tokens drawn *independently* from per-position distributions.

Block diffusion. BD3-LM (Arriola et al., 2025a) splits a sequence x of length L into K blocks $x^1, \ldots, x^K$ of B tokens, factorizes $\log p_\theta(x) = \sum_b \log p_\theta(x^b \mid x^{<b})$, and models each factor with masked diffusion inside the block: the noisy block attends to itself and to all clean earlier blocks (a block-causal mask), training is vectorized over the concatenation $[x_t \mid x_0]$ with a $2L \times 2L$ mask, and the noise schedule is clipped to a data-dependent interval to reduce gradient variance. Generation proceeds block by block with a KV cache, and within a block T_{blk} denoising steps decode the B tokens. We keep all of this and build on the official implementation.

The sampler within a block. Fix the context $c = x^{<b}$. At any step, the denoiser's output over the still-masked tokens x_i of the block is a product distribution $\prod_i q_i(x_i)$, and with $T_{\text{blk}}{=}1$ the whole block is one draw from it. By Proposition 1, one step cannot beat the total correlation of the tokens it decodes, so the more strongly a block's tokens depend on each other, the more steps coherent text needs. Word-level blocks are strongly dependent: a subject fixes its verb, a noun its article.

3 Latent-Plan Block Diffusion

3.1 Setting and notation

Figure 2 gives an overview. The backbone f_θ is the block-causal diffusion transformer (DiT; Peebles & Xie, 2023) of BD3-LM. We write $h(x) \in \mathbb{R}^{L \times d_{\text{model}}}$ for its final hidden states on a clean sequence under the block-causal mask (block b sees blocks $\leq b$) and $h(x)_{[b]} \in \mathbb{R}^{B \times d_{\text{model}}}$ for those of block b. Appendix B lists everything LP-BD adds, and nothing in BD3-LM is removed or retuned.

3.2 Encoder, target encoder and plan

Online plan. One clean pass of f_θ gives $h(x)$. For block b, an attention-pooling module Pool_ϕ with m learned queries attends over its B hidden states and maps each pooled vector to d dimensions, and a LayerNorm without affine parameters maps the result onto the sphere of zero-mean vectors

of norm $\sqrt{D}$ ($D{=}md$), giving $z_b^{\text{on}} = \text{Sphere}\big(\text{Pool}_\phi(h(x)_{[b]})\big) \in \mathbb{R}^D$. Gradients from the denoiser pass through z_b^{on} into the pooling and the backbone (*generative grounding*): the plan is whatever helps the denoiser decode the block.

Target plan. The flow fits z_b^*, computed in the same way by a stop-gradient EMA copy $(\bar{\theta}, \bar{\phi})$, so the flow-matching loss cannot directly pull the plan toward what the prefix predicts.

Prefix state. h_b is the hidden state at the last position of block $b{-}1$ in the same clean pass (a learned vector h_{start} for $b{=}1$). At sampling time it is the last hidden state of the pass that stores block $b{-}1$ into the KV cache, so planning costs no additional backbone forward.

3.3 Conditional flow predictor

The plan distribution $p_\psi(z_b \mid h_b)$ is a rectified flow (Liu et al., 2023): with $s \sim \mathcal{U}(0,1)$, $\epsilon \sim \mathcal{N}(0, \mathbf{I}_D)$ and $z^{(s)} = (1-s)\epsilon + s z_b^*$, a three-layer multilayer perceptron (MLP) $v_\psi(z^{(s)}, s, h_b)$ is trained with the flow-matching (FM) loss (Lipman et al., 2023)

$$\mathcal{L}_{\text{FM}} = \mathbb{E}\left\| v_\psi(z^{(s)}, s, h_b) - (z_b^* - \epsilon) \right\|^2,$$

and the flow is sampled with 8 Euler steps from $z^{(0)}{=}\epsilon$ to $s{=}1$, followed by re-projection onto the sphere. The FM gradient also reaches h_b and the backbone. Since h_b is a function of the prefix c, we also write $p_\psi(z_b \mid c)$. Once the encoder is trained, the flow can also be refit on the frozen model.

3.4 Injecting the plan into the denoiser

The DiT backbone already carries a per-position conditioning vector for the noise level through adaptive layer normalization (adaLN; Peebles & Xie, 2023). We add $W_1 z_b$ to that vector and $W_2 z_b$ to the token embeddings, at every noisy position of block b. Clean context positions receive $z{=}0$ and are processed as in BD3-LM. W_1 and W_2 are zero-initialized, so LP-BD is identical to BD3-LM at initialization and the plan bootstraps from the denoiser's gradient. *Sub-block plans.* Broadcasting one vector to 64 positions leaves the plan nearly unused. For $B{=}64$ we use $m{=}4$ slots of $d{=}32$: slot j is injected only into the j-th sub-block of B/m positions, while pooling, target and flow treat the concatenation $(z^1, \ldots, z^4) \in \mathbb{R}^{128}$ as one plan, so sub-block intents are sampled jointly.

3.5 Training objective

For each block in a training batch, the denoiser is conditioned on

$$\tilde{z}_b = \begin{cases} \text{Sphere}\big(z_b^{\text{on}} + \sigma\xi\big), & \text{w.p. } 1 - p_\varnothing, \quad \sigma = \frac{1}{8}\sqrt{D} \ \text{(cosine} \approx 0.8 \text{ with } z_b^{\text{on}} \text{ at } D{=}32), \\ z_\varnothing \text{ (learned null vector)}, & \text{w.p. } p_\varnothing. \end{cases}$$

The noise $\xi \sim \mathcal{N}(0, \mathbf{I}_D)$ makes the denoiser robust to imperfect plans, and training with $z_\varnothing$ gives the null path used for guidance and as a control. A fraction $\rho{=}0.25$ of blocks is fully masked, so the denoiser must decode them from $\tilde{z}_b$ and the context alone. This *full-mask* term keeps the denoiser from ignoring the plan, and the stop-gradient target keeps the plan from collapsing to a constant. The total loss is $\mathcal{L} = \mathcal{L}_{\text{NELBO}}(\theta, \phi; \tilde{z}) + \lambda\, \mathcal{L}_{\text{FM}}(\psi, \theta)$ with $\lambda{=}1$, where $\mathcal{L}_{\text{NELBO}}$ is the BD3-LM NELBO conditioned on $\tilde{z}_b$. Flow samples $\hat{z}_b$, which are independent of the block, are not used as training conditions: training on them makes the plan uninformative (Appendix E).

3.6 Sampling

Algorithm 1 decodes a block in three moves: read h_b from the KV-cache store pass of the previous block, draw $\hat{z}_b$ from the flow, and run T_{blk} masked-diffusion steps on the block conditioned on $\hat{z}_b$. Classifier-free guidance (Ho & Salimans, 2022) combines the logits $\ell_{\hat{z}}$ given the plan and $\ell_\varnothing$ given the null vector as $\ell = \ell_\varnothing + w(\ell_{\hat{z}} - \ell_\varnothing)$ to sharpen the plan's influence, with $w \in [1, 2]$ throughout.

4 Why a plan helps, and when it does not

Fix a context c and write $P = p(x^b \mid c)$ for the block distribution, P_i for its marginals, and $\text{TC}(x^b \mid c) = \sum_i H(x_i \mid c) - H(x^b \mid c)$ for the conditional total correlation, where $x_1, \ldots, x_B$ are the

Algorithm 1 LP-BD sampling of one sequence

$h_1 \leftarrow h_{\text{start}}$; KV cache $\leftarrow \emptyset$
for $b = 1, \dots, K$ **do**
  $\hat{z}_b \leftarrow \text{Sphere}\big(\text{Euler}_8(v_\psi(\cdot, \cdot, h_b))\big)$ *sample the plan*
  $x_t^b \leftarrow [\text{MASK}]^B$
  for t in schedule(T_{blk}) **do**
    $\ell_{\hat{z}} \leftarrow f_\theta(x_t^b \mid \text{KV}, \hat{z}_b)$ and, if $w \neq 1$, $\ell_\varnothing \leftarrow f_\theta(x_t^b \mid \text{KV}, z_\varnothing)$
    $\ell \leftarrow \ell_{\hat{z}}$ if w=1, else $\ell_\varnothing + w(\ell_{\hat{z}} - \ell_\varnothing)$ *denoise given the plan*
    unmask positions of x_t^b from $\text{softmax}(\ell)$ following the masked-diffusion posterior
  end for
  store x^b in the KV cache; $h_{b+1} \leftarrow$ last hidden state of that pass *no extra forward*
end for

tokens of x^b, and H and I denote entropy and mutual information in nats, with entropies given a latent z averaged over z. Proofs are in Appendix A.

Proposition 1 (Single-step irreducible error)**.** $\min_{Q=\prod_i q_i} D_{\text{KL}}(P \,\|\, Q) = D_{\text{KL}}(P \,\|\, \prod_i P_i) = \text{TC}(x^b \mid c)$*, attained at* $q_i = P_i$*.*

Appending a token to the block cannot lower this cost. Methods that choose *which* positions to unmask together can only trade this cost for more steps, down to one token per step.

Proposition 2 (Exact decomposition under a latent)**.** *For any latent* z *(discrete or continuous),* $\mathbb{E}_z \,\text{TC}(x^b \mid z, c) = \big[\sum_i H(x_i \mid z, c) - H(x^b \mid c)\big] + I(x^b; z \mid c)$.

The left side is the expected single-step error given the plan, and by joint convexity it upper-bounds $D_{\text{KL}}\big(P \,\|\, \mathbb{E}_z \prod_i p(x_i \mid z, c)\big)$, the error of one step with a sampled plan. The first bracket is what training minimizes: $\sum_i H(x_i \mid z, c)$ is the smallest cross-entropy, summed over positions, that a product denoiser can reach on a fully masked block given the plan (the full-mask term of the objective), and $H(x^b \mid c)$ is a constant. This expected error is zero if and only if the tokens are conditionally independent given (z, c), in which case one parallel step is exact. A latent that copies the whole block achieves this trivially but leaves the flow to model the block in full, which the bottleneck (small d and noise on the plan) prevents.

Proposition 3 (Sampling consistency)**.** *If* $p_\psi(\cdot \mid c)$ *is the law of* z_b^* *given* c *and the denoiser is exact given* (z, c)*, then sampling* $z \sim p_\psi$ *and then* $x^b \sim p_\theta(\cdot \mid z, c)$ *yields* $x^b \sim p(\cdot \mid c)$*.*

Consistency requires the plan *distribution*, not a point prediction. Both hypotheses hold only approximately, as the denoiser is trained on noised online plans and the flow on EMA-target plans.

Remark 1 (A latent does not help by itself)**.** *Let* x_1, x_2 *be independent fair bits and* $z = x_1 \oplus x_2$*. Then* $I(x_i; z) = 0$ *for each* i*, yet given* z *the two bits are perfectly correlated: the total correlation* rises *from* 0 *to* $\log 2$*. Proposition 2 therefore does not guarantee that a latent lowers* TC*. Training lowers the first bracket and the bottleneck limits* $I(x^b; z \mid c)$*, which favors plans that lower* TC*.*

5 Experiments

We train LP-BD from scratch, add it to a pretrained model, and analyze when and why the plan helps.

5.1 Setup and evaluation protocol

Training from scratch. We train on TinyStories (GPT-2 tokens, length 256; Eldan & Li, 2023) with a 28M backbone for 150k steps (three seeds, plus an 85M backbone for scaling) and on text8 (characters, length 256; Mahoney, 2011) with a 12M backbone for 50k steps. BD3-LM baselines use the official code. Both models share the hyperparameters of Appendix B, including the number of training steps, and LP-BD trains 1.4–1.7× slower. Sample quality is the judge NLL: the negative log-likelihood of generated text under an AR model of the same architecture trained on the same data (held-out text: 1.39 nats/token on TinyStories, 1.02 nats/char on text8), read together with rep_8, the fraction of repeated 8-grams within a sample. Cells with $\text{rep}_8 > 0.02$ count as degenerate.

Table 1: **TinyStories: judge NLL versus denoising steps** (nats/token, lower is better, real text 1.39, mean±std over three seeds). Percentages are relative to BD3-LM.

B	method	T_{blk}=1	2	4	8	16	64
16	BD3-LM	9.68±0.07	7.34±0.02	5.47±0.03	4.24±0.04	3.56±0.04	–
	LP-BD, null vector	9.57±0.04	7.24±0.03	5.39±0.01	4.22±0.02	3.57±0.03	–
	LP-BD, w=1	6.71±0.11 (−31%)	4.91±0.11 (−33%)	3.84±0.05 (−30%)	3.30±0.07 (−22%)	3.08±0.02 (−14%)	–
	LP-BD, best w	**5.96±0.18** (−38%)	**4.41±0.10** (−40%)	**3.58±0.06** (−34%)	**3.17±0.04** (−25%)	**2.96±0.03** (−17%)	–
64	BD3-LM	9.95±0.12	7.93±0.11	6.06±0.09	4.73±0.10	3.99±0.08	3.37±0.12
	LP-BD, null vector	9.95±0.03	7.81±0.02	5.94±0.05	4.67±0.02	3.99±0.04	3.39±0.00
	LP-BD, w=1	8.82±0.03 (−11%)	6.59±0.03 (−17%)	4.93±0.04 (−19%)	3.98±0.03 (−16%)	3.44±0.02 (−14%)	3.08±0.02 (−9%)
	LP-BD, w=2	**8.11±0.08** (−18%)	**5.68±0.02** (−28%)	**4.18±0.02** (−31%)	**3.43±0.03** (−27%)	**3.03±0.02** (−24%)	**2.72±0.02** (−19%)

A pretrained model. To test LP-BD at larger scale, we add it to the released 170M BD3-LM checkpoints for OpenWebText (GPT-2 tokens, length 1024, B=8 and 16; Gokaslan & Cohen, 2019) and fine-tune for 20k steps. The BD3-LM results come from a control fine-tuned as BD3-LM for the same 20k steps. Quality is the generative perplexity (Gen PPL) under GPT-2 Large (Radford et al., 2019), computed as in BD3-LM and reported with the per-sample unigram entropy, a coarse measure of diversity. Gen PPL can be lowered by lowering diversity, and at a common nucleus (top-p) threshold (Holtzman et al., 2020) LP-BD is more diverse than BD3-LM because its plan is drawn before truncation. Systems are therefore compared at matched entropy: BD3-LM's frontier is traced with $p \in \{0.8, 0.9, 0.95, 1\}$ and each LP-BD operating point is compared with BD3-LM interpolated to the same entropy. With BD3-LM's one-token-per-step sampler at p=0.9, our pipeline reproduces the published Gen PPL of these checkpoints within 2% (Appendix D.3).

Sampling and compute. Every cell uses 256 unprompted samples and, unless noted, the fixed-step schedule $t_i = 1 - i/T_{\text{blk}}$ (Appendix C). For LP-BD we sweep $w \in \{1, 1.5, 2\}$ and report w=1 and guided settings. Compute is counted in denoiser forward passes, which guidance (w>1) doubles. The flow adds at most 5% to the floating-point operations (FLOPs), but its sequential steps lower the measured throughput on OpenWebText by 2–19% without guidance (Appendix D.3).

5.2 Models trained from scratch: TinyStories and text8

Table 1 and Figure 1 give the TinyStories results. LP-BD improves the judge NLL at every step budget, most at few steps: at B=16 the one-step judge NLL drops by 31% (38% with guidance), and LP-BD beats BD3-LM with half the steps. Sub-block plans carry the gains to B=64, where 16 guided steps outperform 64-step BD3-LM with half the forward passes. At equal quality, sampling is 1.8× (B=16) and 2.8× (B=64) faster in wall-clock time (Table 5). Sampled with the null vector, the same checkpoints reproduce BD3-LM within 2% (hollow markers in Figure 1), so the gain comes from the plan. Repetition stays below that of held-out text, MAUVE (Pillutla et al., 2021) ranks the systems the same way, with 4 guided steps of LP-BD above 16 steps of BD3-LM (0.64 vs. 0.57 at B=16, Table 6), and where one-step BD3-LM produces a bag of words, LP-BD produces phrases (Figure 6). The relative gains are similar for an 85M backbone (Table 14) and smaller on character-level text8 (21% at one step, Table 7).

Likelihood. With the denoiser conditioned on the null vector, LP-BD has a NELBO of the same form as BD3-LM's (the null-path NELBO). It is at most 0.002 nats above BD3-LM's for every 28M TinyStories model and 0.085 nats/char above on text8 (Table 8). A variant of the plan model with Gaussian-smoothed plans has a rate–distortion bound on its likelihood (Appendix A), 1.55 nats/char on text8 against BD3-LM's 1.34, because the information in the plan is charged as rate.

5.3 A pretrained 170M model: OpenWebText

Added to a pretrained model, the plan helps as well (Table 2, Figure 3). At matched entropy LP-BD lowers Gen PPL at every step budget and both block sizes, by 14–55% without guidance and by 38–66% with w=1.5 (cells marked ‡ excluded), and at B=16 its guided samples stay within 0.1 of the real-text entropy at every step budget, while BD3-LM at p=0.9 falls 0.2–0.5 below it. With guidance, which doubles the denoiser passes per step, LP-BD with half the steps still beats BD3-LM at matched entropy for every budget and both block sizes (by 8–43%), and except at one step with

Table 2: **OpenWebText, LP-BD added to the released 170M BD3-LM checkpoints.** Gen PPL (GPT-2 Large, p=0.9, refit flow). Parentheses: entropy for BD3-LM rows, change against the row above for LP-BD rows. ‡: reference BD3-LM at p=1, still less diverse than LP-BD.

B	method	T_{blk}=1	2	4	8	16
8	BD3-LM, p=0.9	305 (4.48)	240 (4.68)	185 (5.09)	96 (5.26)	–
	BD3-LM at matched entropy	1038 (5.21)	793 (5.40)	416 (5.45)	164 (5.44)	–
	LP-BD, w=1	**586** (−44%)	**361** (−54%)	**186** (−55%)	**116** (−29%)	–
	BD3-LM at matched entropy	1983‡ (5.56)	1631‡ (5.74)	678 (5.66)	297 (5.64)	–
	LP-BD, w=1.5	**1117** (−44%)	**490** (−70%)	**229** (−66%)	**152** (−49%)	–
16	BD3-LM, p=0.9	696 (5.02)	386 (4.92)	284 (5.25)	125 (5.27)	66 (5.28)
	BD3-LM at matched entropy	739 (5.06)	546 (5.12)	307 (5.29)	155 (5.35)	90 (5.38)
	LP-BD, w=1	**635** (−14%)	**353** (−35%)	**204** (−34%)	**108** (−30%)	**75** (−17%)
	BD3-LM at matched entropy	1677 (5.52)	1185 (5.54)	537 (5.55)	225 (5.51)	125 (5.48)
	LP-BD, w=1.5	**900** (−46%)	**484** (−59%)	**218** (−59%)	**113** (−50%)	**78** (−38%)

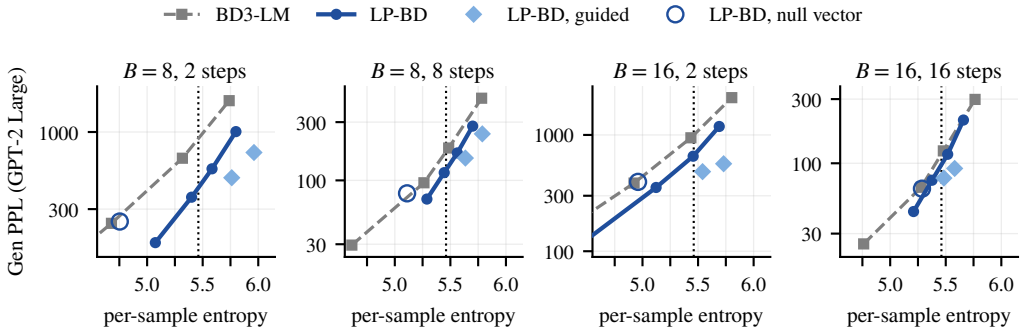

Figure 3: Gen PPL versus per-sample entropy on OpenWebText (lower is better at equal entropy). Lines: nucleus sweeps. Diamonds: guided LP-BD (w=1.5 and 2). Dotted: real-text entropy.

B=8 it is also 1.04–1.23× faster in wall-clock time (Table 11). The 95% bootstrap intervals over samples, which also resample BD3-LM's frontier, exclude zero for every LP-BD cell (Table 10), and sampling BD3-LM directly at the nucleus threshold that matches the entropy of six LP-BD cells moves the reference by at most 5%. Sampled with the null vector, the same checkpoints differ from BD3-LM's frontier by −9% to +17% (Table 13), and the null-path NELBO is 0.04 nats above BD3-LM's. All LP-BD rows use a 12.8M flow refit for 8k steps on the frozen model, which gives larger gains at 8 and 16 steps than the jointly trained 0.7M flow (Appendix D.3).

5.4 When and why the plan helps

The plan is a sampled intent, not a prediction. Shuffling the prefix state across the batch raises the flow's loss by only 4–7% (Figure 4a), and on OpenWebText a larger flow that attends over the whole prefix fits the plans no better than one restricted to h_b. Hence $p_\psi(z \mid c)$ is close to the marginal law $p(z)$ of the plan: the plan encodes the part of the block that the prefix does not determine. Sampling confirms this (Table 9): a flow sample drawn for another sequence's prefix keeps 66–92% of the gain on TinyStories and text8 at up to eight steps, so most of the gain comes from a random intent shared by the whole block. On these datasets, which intents are drawn still matters: a plan drawn uniformly on the sphere keeps only 40–66%.

The plan must be sampled, once per block. Proposition 3 assumes a sampled plan. With a deterministic predictor that regresses z_b^* on h_b, every block is decoded from an average plan, and the samples repeat the corpus's most frequent block ("zero zero zero …" on text8, $\text{rep}_8 = 0.42$–0.63, Table 3 and Figure 5). Pushing the plan toward predictability, by letting the flow-matching gradient reach the encoder, collapses it at full strength and, scaled by 0.1, removes the gain of sub-block plans (Table 8). A plan redrawn at every denoising step fails in the opposite way: quality barely

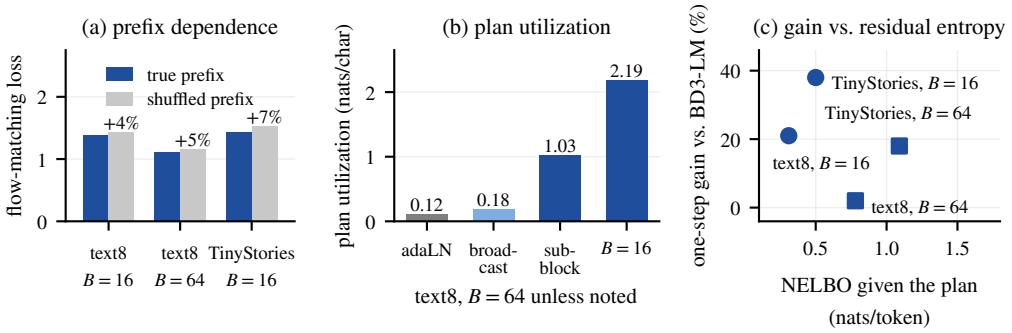

Figure 4: (a) Flow-matching loss with the true or a shuffled prefix. (b) Plan utilization (adaLN: injection into adaLN only). (c) One-step gain (best w) versus NELBO given the plan.

Table 3: **Ablations on text8,** B=16**.** Judge NLL (nats/char, best w per cell) and null-path NELBO. †: degenerate ($\mathrm{rep}_8 = 0.42$–0.63).

variant	T_{blk}=1	2	4	8	16	null-path NELBO
BD3-LM	3.69	3.37	3.03	2.64	2.33	1.341
LP-BD	2.93	2.62	2.38	2.21	2.13	1.426
adaLN injection only	3.19	2.81	2.45	2.21	2.06	1.482
d=8	3.31	2.89	2.46	2.17	2.00	1.432
no noise on the training plan	3.29	3.24	3.22	3.20	3.19	1.590
online target instead of EMA	2.92	2.61	2.38	2.22	2.12	1.435
deterministic plan	1.14†	1.18†	1.11†	1.07†	1.06†	1.423

improves with more steps and falls behind BD3-LM from four steps on (Table 9), since the tokens unmasked at different steps follow different plans.

Gains are larger where the plan leaves less entropy. A plan that leaves little entropy in the block also leaves little total correlation: since $H(x_i \mid z, c) \le H(x^b \mid z, c)$ for every position, $\mathbb{E}_z \mathrm{TC}(x^b \mid z, c) \le (B-1)\, H(x^b \mid z, c)$. Figure 4c plots the one-step gain in four settings against the denoiser's NELBO given the plan, an upper bound on this entropy per token. Within each dataset the gain is larger at B=16, where the plan determines most of the block (21–38% at one step), than at B=64, where it does not (2–18%), while on TinyStories the guided multi-step gains are similar across block sizes because later steps resolve what the plan left open. Word-level tokens put more of a block's entropy into correlations a plan can remove, hence the larger TinyStories gains.

The gain needs joint decoding. Proposition 1 also says when the plan stops helping: a step that decodes a single token has no total-correlation cost. BD3-LM's own sampler unmasks exactly one token per step, whereas the fixed-step schedule unmasks a random number and so decodes jointly even at T_{blk}=B. With the former, unguided LP-BD is 30% (B=8) and 11% (B=16) worse than BD3-LM on OpenWebText at matched entropy, guidance restores a gain only at B=16, and the null vector stays within 11% of BD3-LM (Table 12).

Large blocks need position-aware injection. More plan capacity does not help unless each position can read its own part of the plan. Plan utilization, the reduction in cross-entropy on fully masked blocks when the denoiser receives the plan instead of the null vector, shows this (Figure 4b): at B=64, broadcasting one plan to every position leaves it nearly unused, even with d=64 or 128, while sub-block plans raise utilization more than fivefold and produce the B=64 gains.

5.5 Ablations

Table 3 ablates the design on text8 with B=16, and Appendix D.2 adds B=64 and further variants. Adding the plan to the token embeddings as well as to adaLN improves one-step quality (-21% vs.

-14%) but not 16-step quality, and a smaller plan (d=8) trades few-step for 8- and 16-step quality. Without noise on the training plan the denoiser over-trusts it, and quality stops improving with more steps. An online stop-gradient target matches the EMA target, so averaging is unnecessary.

6 Related work

Block diffusion and parallel-decoding schedules. BD3-LM (Arriola et al., 2025a) and SSD-LM (Han et al., 2023) generate semi-autoregressively, and follow-ups adapt the block size (Lu et al., 2026a; Huang & Tang, 2025), train one model for any block size (Jain, 2026), decode by confidence (Wu et al., 2026; Liu et al., 2025b), decode token sets (Gat et al., 2025), unmask spaced-out positions in a fixed order (Luxembourg et al., 2026), or allocate test-time compute (Lu et al., 2026b). These decide *when* tokens are decoded in parallel or make the model robust to that choice. Without a shared variable, dependent tokens must be decoded in separate steps, so LP-BD complements them.

Latents and joint models for discrete diffusion. The closest methods also add a latent variable to masked diffusion. VADD (Xie et al., 2026) captures token correlations with a Gaussian latent drawn afresh from a fixed prior at every denoising step. CRoCoDiL (Uziel et al., 2026) conditions an 8B masked diffusion model on sentence-level representations, which a jointly trained encoder provides during training and a separate 400M latent diffusion model generates at inference. Copula and energy-based models (Liu et al., 2025a; Xu et al., 2025) correct the product of marginals at sampling time, Hersche et al. (2026) decode blocks with a small local AR model, E2D2 (Arriola et al., 2025b) uses a separate encoder for clean tokens, and sequence-level latent diffusion (Zhang et al., 2023; Lovelace et al., 2023) relies on separately trained autoencoders. LP-BD differs in granularity (one latent per block, fixed across its denoising steps), conditioning (given the generated prefix, compatible with the KV cache) and cost (one backbone and its EMA copy serve as encoder, target and denoiser). We also characterize when such a latent helps.

JEPA. Joint-embedding predictive architectures (JEPA; Assran et al., 2023) predict representations of unseen parts with an EMA target. In language models they serve as auxiliary losses (Huang et al., 2026; Nam, 2026), and in image diffusion as guidance (Park & Um, 2026), not as the generation condition. LP-BD keeps the EMA target but samples the plan from a conditional flow and conditions generation on it, since on text the useful part of a plan is what the prefix cannot predict.

7 Conclusion

Parallel decoding in block diffusion is limited by the total correlation of a block: tokens drawn independently cannot agree on what the block says. LP-BD supplies the missing shared variable, a plan sampled once per block and learned from the denoiser's own loss. With a fixed-step sampler, on TinyStories (B=16) it beats BD3-LM with half the steps at every budget and, added to the released 170M checkpoints on OpenWebText, lowers generative perplexity at matched entropy by 14–55% at every step budget. With guidance, it also beats BD3-LM with half the steps on OpenWebText. Sampled with the null vector, the same models nearly recover BD3-LM, so the gain comes from the plan. The plan acts as an intent shared by the block rather than a prediction: it must be drawn once per block, and a plan drawn for another prefix keeps most of the gain.

Limitations. The plan helps mainly when a step decodes several tokens jointly: with exactly one token per step, unguided LP-BD trails BD3-LM. One-step decoding of large blocks stays limited by the entropy that the plan leaves in the block. The plan carries little information about the context: a flow that attends over the whole prefix fits it no better, and pushing the plan toward predictability collapses it or removes the gain of sub-block plans. Training costs 1.4–1.7$\times$ as much as BD3-LM, the flow lowers throughput at small batch sizes, the plan model's likelihood bound stays above BD3-LM's NELBO, and our models (at most 170M parameters, one training seed on OpenWebText) are evaluated only on unconditional generation with model-based metrics. Plans that carry information across blocks are a natural next step.

AI USE STATEMENT

In this work, we used generative AI tools for implementing the method, designing and running experiments, refining hypotheses, reformatting datasets, assisting in the writing of proofs, and interpreting results. We have not used generative AI tools for developing the conceptual framework, formulating the mathematical claims, providing the ingredients of their proofs, generating synthetic data, or translation, and qualitative or thematic data analysis is not applicable to this work. Additionally, we used generative AI tools for creating and editing code, suggesting experimental parameters, creating figures, drafting and editing the paper, searching for and summarizing literature, and formatting references. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work uses public text corpora (TinyStories, text8 and OpenWebText) and publicly released model checkpoints. It involves no human subjects and collects no data. Like other language models, the models studied here can produce false or offensive text. They are small research models (at most 170M parameters), and the method changes how a block is sampled, not the training data.

REPRODUCIBILITY STATEMENT

Appendix B lists every component that LP-BD adds to BD3-LM together with all hyperparameters. Appendices C and D.3 specify the evaluation protocol, including the sampler schedule, the judge, Gen PPL and the matched-entropy comparison, and Appendix A contains the proofs. The 28M TinyStories results are averaged over three training seeds, and the OpenWebText results, from one training seed, come with bootstrap intervals over samples (Appendix D.3). Code, configurations and evaluation scripts will be released.

REFERENCES

Marianne Arriola, Aaron Gokaslan, Justin T. Chiu, Zhihan Yang, Zhixuan Qi, Jiaqi Han, Subham Sekhar Sahoo, and Volodymyr Kuleshov. Block diffusion: Interpolating between autoregressive and diffusion language models. In *International Conference on Learning Representations*, 2025a.

Marianne Arriola, Yair Schiff, Hao Phung, Aaron Gokaslan, and Volodymyr Kuleshov. Encoder-decoder diffusion language models for efficient training and inference. In *Advances in Neural Information Processing Systems*, 2025b.

Mahmoud Assran, Quentin Duval, Ishan Misra, Piotr Bojanowski, Pascal Vincent, Michael Rabbat, Yann LeCun, and Nicolas Ballas. Self-supervised learning from images with a joint-embedding predictive architecture. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023.

Jacob Austin, Daniel D. Johnson, Jonathan Ho, Daniel Tarlow, and Rianne van den Berg. Structured denoising diffusion models in discrete state-spaces. In *Advances in Neural Information Processing Systems*, 2021.

Ronen Eldan and Yuanzhi Li. TinyStories: How small can language models be and still speak coherent English? *arXiv preprint arXiv:2305.07759*, 2023.

Itai Gat, Heli Ben-Hamu, Marton Havasi, Daniel Haziza, Jeremy Reizenstein, Gabriel Synnaeve, David Lopez-Paz, Brian Karrer, and Yaron Lipman. Set block decoding is a language model inference accelerator. *arXiv preprint arXiv:2509.04185*, 2025.

Aaron Gokaslan and Vanya Cohen. OpenWebText corpus. `http://Skylion007.github.io/OpenWebTextCorpus`, 2019.

Xiaochuang Han, Sachin Kumar, and Yulia Tsvetkov. SSD-LM: Semi-autoregressive simplex-based diffusion language model for text generation and modular control. In *Annual Meeting of the Association for Computational Linguistics*, 2023.

Michael Hersche, Nicolas Menet, Ronan Tanios, and Abbas Rahimi. Locally coherent parallel decoding in diffusion language models. In *International Conference on Machine Learning*, 2026.

Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. The curious case of neural text degeneration. In *International Conference on Learning Representations*, 2020.

Chihan Huang and Hao Tang. CtrlDiff: Boosting large diffusion language models with dynamic block prediction and controllable generation. *arXiv preprint arXiv:2505.14455*, 2025.

Hai Huang, Yann LeCun, and Randall Balestriero. LLM-JEPA: Large language models meet joint embedding predictive architectures. In *International Conference on Learning Representations*, 2026.

Gagan Jain. Adaptive block diffusion: Resolving training-inference mismatch in diffusion language models. *arXiv preprint arXiv:2606.29275*, 2026.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. In *International Conference on Learning Representations*, 2023.

Anji Liu, Oliver Broadrick, Mathias Niepert, and Guy Van den Broeck. Discrete copula diffusion. In *International Conference on Learning Representations*, 2025a.

Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *International Conference on Learning Representations*, 2023.

Yangzhou Liu, Yue Cao, Hao Li, Gen Luo, Zhe Chen, Weiyun Wang, Xiaobo Liang, Biqing Qi, Lijun Wu, Changyao Tian, Yanting Zhang, Yuqiang Li, Tong Lu, Yu Qiao, Jifeng Dai, and Wenhai Wang. Sequential diffusion language models. *arXiv preprint arXiv:2509.24007*, 2025b.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019.

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

Justin Lovelace, Varsha Kishore, Chao Wan, Eliot Shekhtman, and Kilian Q. Weinberger. Latent diffusion for language generation. In *Advances in Neural Information Processing Systems*, 2023.

Guanxi Lu, Hao Mark Chen, Yuto Karashima, Zhican Wang, Daichi Fujiki, and Hongxiang Fan. AdaBlock-dLLM: Semantic-aware diffusion LLM inference via adaptive block size. In *International Conference on Learning Representations*, 2026a.

Yi Lu, Deyang Kong, Jianing Wang, Linsen Guo, Xue Wang, Qi Guo, Tao Gui, Xuanjing Huang, Wei Ye, Shikun Zhang, and Wei Wang. Adaptive test-time compute allocation for block diffusion language models in complex reasoning. *arXiv preprint arXiv:2602.09555*, 2026b.

Omer Luxembourg, Haim Permuter, and Eliya Nachmani. Plan for speed: Dilated scheduling for masked diffusion language models. In *International Conference on Machine Learning*, 2026.

Matt Mahoney. About the test data. `http://mattmahoney.net/dc/textdata.html`, 2011.

Sangdae Nam. DLLM-JEPA: Joint embedding predictive architectures for masked diffusion language models. *arXiv preprint arXiv:2606.00091*, 2026.

Sol Park and Soobin Um. Beyond generative priors: Minority sampling with JEPA-guided diffusion. In *International Conference on Machine Learning*, 2026.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *IEEE/CVF International Conference on Computer Vision*, 2023.

Krishna Pillutla, Swabha Swayamdipta, Rowan Zellers, John Thickstun, Sean Welleck, Yejin Choi, and Zaid Harchaoui. MAUVE: Measuring the gap between neural text and human text using divergence frontiers. In *Advances in Neural Information Processing Systems*, 2021.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. Technical report, OpenAI, 2019.

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

Jiaxin Shi, Kehang Han, Zhe Wang, Arnaud Doucet, and Michalis K. Titsias. Simplified and generalized masked diffusion for discrete data. In *Advances in Neural Information Processing Systems*, 2024.

Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. RoFormer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024.

Roy Uziel, Omer Belhasin, Itay Levy, Akhiad Bercovich, Ran El-Yaniv, Ran Zilberstein, and Michael Elad. CRoCoDiL: Continuous and robust conditioned diffusion for language. *arXiv preprint arXiv:2603.20210*, 2026.

Satosi Watanabe. Information theoretical analysis of multivariate correlation. *IBM Journal of Research and Development*, 4(1):66–82, 1960.

Chengyue Wu, Hao Zhang, Shuchen Xue, Zhijian Liu, Shizhe Diao, Ligeng Zhu, Ping Luo, Song Han, and Enze Xie. Fast-dLLM: Training-free acceleration of diffusion LLM by enabling KV cache and parallel decoding. In *International Conference on Learning Representations*, 2026.

Tianyu Xie, Shuchen Xue, Zijin Feng, Tianyang Hu, Jiacheng Sun, Zhenguo Li, and Cheng Zhang. Variational autoencoding discrete diffusion with enhanced dimensional correlations modeling. In *International Conference on Learning Representations*, 2026.

Minkai Xu, Tomas Geffner, Karsten Kreis, Weili Nie, Yilun Xu, Jure Leskovec, Stefano Ermon, and Arash Vahdat. Energy-based diffusion language models for text generation. In *International Conference on Learning Representations*, 2025.

Yizhe Zhang, Jiatao Gu, Zhuofeng Wu, Shuangfei Zhai, Josh Susskind, and Navdeep Jaitly. PLANNER: Generating diversified paragraph via latent language diffusion model. In *Advances in Neural Information Processing Systems*, 2023.

A PROOFS

Throughout, all entropies and mutual informations are conditional on the context c, which is fixed. We write $x = x^b = (x_1, \dots, x_B)$, $P = p(x \mid c)$ and P_i for its marginals. For a random variable z, $H(x \mid z)$ denotes the usual conditional entropy $\mathbb{E}_z[H(p(x \mid z, c))]$, while $\mathrm{TC}(x \mid z)$ denotes the total correlation of $p(x \mid z, c)$ at the realized value of z.

Proposition 1 (Single-step irreducible error). $\min_{Q=\prod_i q_i} D_{\mathrm{KL}}(P \,\|\, Q) = D_{\mathrm{KL}}(P \,\|\, \prod_i P_i) = \mathrm{TC}(x^b \mid c)$*, attained at* $q_i = P_i$*. Moreover, appending a token to the block cannot decrease* TC*.*

Proof. For any product distribution $Q = \prod_i q_i$ on the block,

$$D_{\mathrm{KL}}(P \,\|\, Q) = \sum_x P(x) \log P(x) - \sum_i \sum_x P(x) \log q_i(x_i) = -H(P) + \sum_i \mathbb{E}_{P_i}\big[-\log q_i(x_i)\big].$$

Each cross-entropy satisfies $\mathbb{E}_{P_i}[-\log q_i] = H(P_i) + D_{\mathrm{KL}}(P_i \,\|\, q_i) \geq H(P_i)$, with equality if and only if $q_i = P_i$. Hence $D_{\mathrm{KL}}(P \,\|\, Q) \geq \sum_i H(P_i) - H(P) = \mathrm{TC}(x^b \mid c)$, with equality exactly at $Q = \prod_i P_i$, which proves the first claim. For monotonicity, append one token y to the block x:

$$\begin{aligned} \mathrm{TC}(x, y) &= \textstyle\sum_i H(x_i) + H(y) - H(x, y) \\ &= \mathrm{TC}(x) + H(y) - H(y \mid x) = \mathrm{TC}(x) + I(x; y) \;\geq\; \mathrm{TC}(x). \end{aligned}$$

□

Proposition 2 (Exact decomposition under a latent). *For any latent z with a joint law with (x^b, c), discrete or continuous,* $\mathbb{E}_z \,\mathrm{TC}(x^b \mid z, c) = \big[\sum_i H(x_i \mid z, c) - H(x^b \mid c)\big] + I(x^b; z \mid c)$.

Proof. For each value of z, $\mathrm{TC}(x \mid z{=}z_0) = \sum_i H(x_i \mid z{=}z_0) - H(x \mid z{=}z_0)$ by definition. Taking the expectation over $z \sim p(z \mid c)$ turns each term into a conditional entropy, so $\mathbb{E}_z \mathrm{TC}(x \mid z) = \sum_i H(x_i \mid z) - H(x \mid z)$. By the definition of mutual information, $H(x \mid z) = H(x) - I(x; z)$, and substituting yields the identity. Four consequences are used in the text. (i) $\mathbb{E}_z \mathrm{TC}(x \mid z) = 0$ if and only if $x_1, \ldots, x_B$ are conditionally independent given (z, c) almost surely, because each $\mathrm{TC}(x \mid z{=}z_0)$ is a Kullback–Leibler divergence and vanishes only at independence. (ii) By the argument of Proposition 1 applied conditionally on (z, c), the smallest cross-entropy that a per-position (product) denoiser can reach on a fully masked block is $\sum_i H(x_i \mid z, c)$, attained at the true conditional marginals. Plan utilization measures how much this cross-entropy drops when the denoiser receives the plan instead of the null vector. (iii) Subtracting the identity from $\mathrm{TC}(x) = \sum_i H(x_i) - H(x)$ gives

$$\mathrm{TC}(x) - \mathbb{E}_z \mathrm{TC}(x \mid z) = \textstyle\sum_i I(x_i; z) - I(x; z),$$

so a latent lowers the total correlation exactly when the information it carries about the tokens one at a time, summed over positions, exceeds its information about the whole block. (iv) One parallel step with a sampled latent draws from $Q = \mathbb{E}_z \prod_i p(x_i \mid z)$. Since $P = \mathbb{E}_z \, p(x \mid z)$ and the KL divergence is jointly convex, $D_{\mathrm{KL}}(P \,\|\, Q) \le \mathbb{E}_z D_{\mathrm{KL}}\big(p(x \mid z) \,\|\, \prod_i p(x_i \mid z)\big) = \mathbb{E}_z \mathrm{TC}(x \mid z)$. The bound can be loose: in Remark 1, Q is uniform and equals P, although $\mathbb{E}_z \mathrm{TC}(x \mid z) = 1$ bit. □

Proposition 3 (Sampling consistency). *Let $z^* = e(x^b)$ be the target plan of a block $x^b \sim p(\cdot \mid c)$ and let $p(\cdot \mid c)$ denote its law. If $p_\psi(\cdot \mid c) = p(\cdot \mid c)$ and $p_\theta(x^b \mid z, c) = p(x^b \mid z, c)$ for almost every z, then sampling $z \sim p_\psi(\cdot \mid c)$ followed by $x^b \sim p_\theta(\cdot \mid z, c)$ yields $x^b \sim p(\cdot \mid c)$.*

Proof. The pair (x^b, z^*) has a well-defined joint law under $p(\cdot \mid c)$ (the map e is deterministic, so $p(x^b \mid z, c)$ is the conditional law on the fiber $e^{-1}(z)$). This law of z^* need not have a density, so we write expectations. By the law of total probability,

$$\mathbb{E}_{z \sim p_\psi(\cdot \mid c)} \, p_\theta(x^b \mid z, c) = \mathbb{E}_{z \sim p(\cdot \mid c)} \, p(x^b \mid z, c) = p(x^b \mid c).$$

No variational bound over z is involved: the two hypotheses concern the flow and the conditional denoiser separately. The first is the population target of the flow-matching loss, whose exact minimizer transports Gaussian noise to $p(\cdot \mid c)$ (Lipman et al., 2023). Minimizing the conditional masked-diffusion NELBO, an upper bound on $-\log p_\theta(x^b \mid z, c)$, drives p_θ toward the second. □

Remark 1 (A latent does not help by itself). *Let x_1, x_2 be independent fair bits and $z = x_1 \oplus x_2$. Then $I(x_i; z) = 0$ for $i = 1, 2$, while $\mathrm{TC}(x \mid z) = 1$ bit although $\mathrm{TC}(x) = 0$.*

Proof. z is a fair bit, and for either value of x_i the conditional law of z is again uniform (flip the other bit), so $I(x_i; z) = 0$. Given z, $x_2 = x_1 \oplus z$ is a deterministic function of x_1, hence $H(x \mid z) = H(x_1 \mid z) = 1$ bit while $\sum_i H(x_i \mid z) = 2$ bits, so $\mathrm{TC}(x \mid z) = 1$ bit, whereas independence gives $\mathrm{TC}(x) = 0$. Proposition 2 is consistent with this: $\sum_i H(x_i \mid z) - H(x) + I(x; z) = 2 - 2 + 1 = 1$ bit, since $I(x; z) = H(z) - H(z \mid x) = 1$ bit. The identity therefore carries no sign guarantee. In the form (iii) of the proof of Proposition 2, $\sum_i I(x_i; z) - I(x; z) = 0 - 1 = -1$ bit: the latent carries information about the pair but none about either bit. □

Rate–distortion bound. Treat the plan as a latent variable with proposal $q(z \mid x^b, c) = \mathcal{N}(z_b^{\mathrm{on}}, \sigma^2 \mathbf{I}_D)$. By Jensen's inequality and the NELBO of the conditional denoiser, the model $p_{\theta,\psi}(x^b \mid c) = \int p_\psi(z \mid c) \, p_\theta(x^b \mid z, c) \, dz$ satisfies

$$-\log p_{\theta,\psi}(x^b \mid c) \le \mathbb{E}_q[\mathrm{NELBO}(x^b \mid z, c)] + \mathbb{E}_q[\log q - \log p_\psi],$$

where the first term is the distortion given the plan and the second the rate of the plan, computed with the exact log-density of the flow (the instantaneous change of variables with a full Jacobian trace, feasible since $D \leq 128$). The bound needs a density for the plan, so it applies to a variant whose flow is fit to Gaussian-smoothed plans and whose denoiser receives $z \sim q$ without projection onto the sphere. Information in the plan lowers the distortion, which drives sample quality, but the likelihood bound charges it as rate. On text8 the bound is 1.55 nats/char, a distortion of 0.57 plus a rate of 0.98, against a NELBO of 1.34 for BD3-LM.

B Implementation details

Table 4 lists every component that LP-BD adds to BD3-LM, with its parameters and cost.

Table 4: **What LP-BD adds to BD3-LM** (28M TinyStories model, d=32).

component	parameters	training cost	inference cost
clean encoder pass (shared backbone) + pooling	0.3M	$+0.5\times$ forward	none (uses KV-cache store pass)
EMA target encoder (copy of backbone + pooling)	0 trainable	EMA update	0
conditional flow v_ψ (3-layer MLP)	0.7M	negligible	8 MLP steps per block ($< 1\%$)
plan injection $W_1 z \to$ adaLN, $W_2 z \to$ embeddings (zero-init)	20k	negligible	negligible
null vector, start state	0.6k	–	–

Backbones. text8: 6 layers, d_{model}=384, 6 heads, 12M parameters; TinyStories: 8 layers, d_{model}=512, 8 heads, 28M; scaling: 12 layers, d_{model}=768, 12 heads, 85M. Block-causal DiT of BD3-LM with adaLN conditioning on the noise level (which BD3-LM sets to a constant), rotary position embeddings (RoPE; Su et al., 2024) and scaled dot-product attention (SDPA). Sequence length 256.

Optimization. AdamW (Loshchilov & Hutter, 2019), learning rate $3 \cdot 10^{-4}$ with 10% linear warmup and cosine decay, weight decay 0.05, β_2=0.98, batch 64, bfloat16, gradient clipping 1.0, for 50k steps on text8 and 150k on TinyStories. The BD3-LM clipped-schedule search over widths $\{0.5, \ldots, 0.9\}$ with resampling is run for both BD3-LM and LP-BD. Evaluation uses EMA weights (decay 0.9999).

Plan. The plan has d=32, with m=1 slot for B=16 and m=4 slots (sub-block plans) for B=64. It is pooled by attention with m queries and projected by a LayerNorm without affine parameters onto the sphere of zero-mean vectors of norm $\sqrt{md}$. The training noise has standard deviation $0.125\sqrt{md}$ with re-projection, $p_\varnothing$=0.15 (0.4 where stated), the fully masked fraction is ρ=0.25, and λ=1. The target-encoder EMA decay is ramped from 0.996 to 0.9995 with a cosine schedule. The flow has 3 layers of width 512, sigmoid linear unit (SiLU) activations, a 64-dimensional Fourier time embedding and a 256-dimensional projection of h_b, and it uses 8 Euler steps at sampling. Guidance $w \in \{1, 1.5, 2\}$ at sampling.

Cost. Training throughput on one H100: BD3-LM 6.3 it/s vs. LP-BD 4.6 it/s on TinyStories (28M), 19 vs. 11 it/s on text8 (12M). The difference is the clean encoder pass, which could be fused with the clean half of the vectorized pass. At sampling time, the plan adds no backbone forward with the KV cache, the 8-step flow adds under 1% to the backbone FLOPs of a block (up to 5% for the 12.8M flow refit on OpenWebText), and guidance evaluates the denoiser twice per step. Tables 5 and 11 give measured tokens/s.

Code. LP-BD extends the official BD3-LM implementation, and its backbone additionally accepts a per-position latent and exposes the clean block-causal pass. Code and configurations will be released.

C Evaluation protocol

Fixed-step schedule. Under the linear schedule, a position that is masked at noise level t stays masked at the next noise level $t - \Delta t$ with probability $(t - \Delta t)/t$, which is undefined at t=0.

Table 5: **Measured throughput and judge NLL** (tokens/s, one H100, KV cache, batch 64, TinyStories 28M, one seed). At T_{blk}=1 the differences are within timing noise.

B	method	T_{blk}=1	2	4	8	16	64
16	BD3-LM	41.6k (9.60)	39.8k (7.34)	23.0k (5.43)	12.5k (4.25)	6.5k (3.53)	–
16	LP-BD, w=1	39.6k (6.69)	36.1k (4.94)	21.7k (3.88)	12.0k (3.38)	6.3k (3.06)	–
	LP-BD, w=1.5	41.9k (6.03)	26.0k (4.40)	14.7k (3.62)	7.9k (3.21)	4.1k (3.01)	–
64	BD3-LM	89.1k (10.05)	85.9k (8.03)	47.9k (6.16)	25.4k (4.85)	13.1k (4.05)	3.4k (3.47)
64	LP-BD, w=1	86.4k (8.79)	81.4k (6.61)	46.0k (4.99)	24.9k (4.01)	12.9k (3.49)	3.3k (3.10)
	LP-BD, w=2	104.7k (8.02)	62.7k (5.64)	34.4k (4.19)	18.1k (3.43)	9.3k (3.05)	2.4k (2.72)

A schedule that includes t=0 as a step, as $\text{linspace}(1, 0, T_{\text{blk}})$ does, therefore produces invalid probabilities in its last step (on text8 with T_{blk}=2 the judge NLL of BD3-LM is 10.7 with that schedule and 3.37 with $t_i = 1 - i/T_{\text{blk}}$). All results except Table 12 use $t_i = 1 - i/T_{\text{blk}}$, $i = 0, \ldots, T_{\text{blk}} - 1$.

Judge and repetition rate. The judge is an AR model with the same backbone architecture, trained on the same data (held-out NLL 1.02 nats/char on text8, 1.39 nats/token on TinyStories), so the metric is independent of an external model's tokenization. Repetitive text receives a low judge NLL (the deterministic-plan samples "zero zero zero ..." score 1.1 nats/char with $\text{rep}_8 = 0.63$), so rep_8, the fraction of 8-grams that reappear within a sample, accompanies every judge number. Cells with $\text{rep}_8 > 0.02$ count as degenerate, are not used when selecting the best w, and are marked † where shown. All non-degenerate LP-BD and BD3-LM cells have $\text{rep}_8 \leq 0.013$ (text8) and ≤ 0.0021 (TinyStories). At the same step count, the fraction of distinct 3-grams of LP-BD is 3–15% (TinyStories) and 7–29% (text8) below BD3-LM's and closer to held-out text (0.65 and 0.07), its unigram entropy is within 6% of BD3-LM's, and its repetition stays below that of held-out text. MAUVE against held-out text (Pillutla et al., 2021) is in Table 6.

Table 6: **MAUVE of TinyStories samples** (28M, higher is better, GPT-2 Medium features, 256 samples vs. 256 held-out texts).

B	method	T_{blk}=1	2	4	8	16	64
16	BD3-LM	0.008	0.015	0.049	0.244	0.567	–
16	LP-BD, w=1	0.024	0.113	0.298	0.717	0.790	–
	LP-BD, w=1.5	0.038	0.241	0.644	0.757	0.825	–
64	BD3-LM	0.008	0.010	0.026	0.199	0.349	0.779
64	LP-BD, w=1	0.010	0.025	0.086	0.360	0.587	0.806
	LP-BD, w=2	0.016	0.057	0.237	0.668	0.789	0.869

Teacher-forced evaluation with a sampled plan. The cross-entropy of the true block given a plan sampled from $p_\psi(z \mid c)$ satisfies $\mathbb{E}_z[-\log p(x \mid z)] \geq -\log \mathbb{E}_z\, p(x \mid z)$ and grows with the width of $p(z \mid c)$, so it is not a measure of sample quality and is not used.

D Additional results

D.1 text8

Table 7 gives the text8 headline results at both block sizes.

D.2 Ablations

Table 8 gives the full ablations on text8, including B=64, with the null-path NELBO for B=16.

Table 7: **text8: judge NLL versus denoising steps** (12M, nats/char, lower is better, real text 1.02, best w per cell).

B	method	T_{blk}=1	2	4	8	16	64
16	BD3-LM	3.69	3.37	3.03	2.64	2.33	–
16	LP-BD	**2.93** (−21%)	**2.62** (−22%)	**2.38** (−21%)	**2.21** (−16%)	**2.13** (−9%)	–
64	BD3-LM	3.76	3.40	3.06	2.68	2.35	2.07
64	LP-BD (sub-block plans)	**3.69** (−2%)	**3.34** (−2%)	**2.86** (−7%)	**2.47** (−8%)	**2.24** (−5%)	**1.99** (−4%)

Table 8: **Ablations on text8** (12M, judge NLL in nats/char, best w per cell, real text 1.02). †: degenerate ($\text{rep}_8 = 0.42$–0.63).

B	variant	T_{blk}=1	2	4	8	16	null-path NELBO
16	BD3-LM	3.69	3.37	3.03	2.64	2.33	1.341
16	LP-BD, adaLN injection	3.19	2.81	2.45	2.21	2.06	1.482
16	null vector at inference	3.73	3.38	3.06	2.79	2.61	–
16	LP-BD, adaLN + embedding (default)	2.93	2.62	2.38	2.21	2.13	1.426
16	deterministic plan	1.14†	1.18†	1.11†	1.07†	1.06†	1.423
16	d=8	3.31	2.89	2.46	**2.17**	**2.00**	1.432
16	$p_\varnothing$=0.4	2.97	2.66	2.42	2.23	2.17	1.374
16	predictability pressure 0.1×	2.97	2.63	**2.36**	2.19	2.11	1.433
16	smoothed flow target (for the bound)	3.15	2.80	2.53	2.34	2.22	1.414
16	no noise on the training plan	3.29	3.24	3.22	3.20	3.19	1.590
16	online target instead of EMA	**2.92**	**2.61**	2.38	2.22	2.12	1.435
		T_{blk}=1	2	4	8	16	64
64	BD3-LM	3.76	3.40	3.06	2.68	2.35	2.07
64	LP-BD, broadcast d=32	**3.62**	3.40	3.02	2.61	2.32	2.11
64	LP-BD, broadcast d=64	3.68	3.43	3.05	2.64	2.38	2.16
64	LP-BD, broadcast d=128	3.72	3.42	3.03	2.70	2.48	2.24
64	LP-BD, broadcast + predictability pressure 0.1×	3.69	3.39	2.99	2.53	2.26	**1.99**
64	LP-BD, sub-block plans 4×32	3.69	**3.34**	**2.86**	**2.47**	**2.24**	**1.99**
64	+ predictability pressure 0.1×	3.74	3.42	3.03	2.68	2.41	2.16

- **Deterministic plan** (regression of z_b^* on h_b, Figure 5): the samples repeat the corpus's most frequent block ("zero zero zero …", $\text{rep}_8 = 0.42$–0.63), which the judge scores almost like real text (1.06–1.18 vs. 1.02 nats/char). Every block is decoded from an average plan.
- **Injection.** At one step, adaLN alone gives -14% and adaLN with token embeddings -21%. At 16 steps adaLN alone is slightly better (2.06 vs. 2.13).
- **Bottleneck.** d=8 is weaker at one step (-10%) but stronger at 16 steps (-14% vs. -9%), and $p_\varnothing$=0.4 closes most of the gap in null-path NELBO to BD3-LM.
- **Predictability pressure.** The flow-matching gradient into the encoder collapses the plan at full strength. At 0.1× it makes the plan partly predictable, leaves B=16 within 1.5%, and improves the broadcast design at B=64 by 3–6% with 8 or more steps, but removes the gain of sub-block plans (utilization 0.16 instead of 1.03 nats/char, Table 8).
- **Noise augmentation and EMA target.** Without noise on the training plan the denoiser over-trusts it: quality stops improving with steps and the null-path NELBO rises from 1.426 to 1.590. An online stop-gradient target matches the EMA target within 0.01 nats.

Variants without gain. At B=64 with broadcast injection, plan sizes $d \in \{32, 64, 128\}$ give about the same utilization (0.18 nats/char) and at most a 4% gain. A contrastive objective on the encoder, $\text{FM}(z, h) - \text{FM}(z, h_{\text{shuffled}})$ with $\text{FM}(z, h)$ the flow-matching loss of plan z given prefix state h and the flow held fixed, makes the plan a function of the context (loss 0.94 with the true prefix state, 4.8 with a shuffled one), and utilization falls from 0.13 to 0.07 nats/char during training (broadcast plans, B=64). Flow sampler steps and temperature have no effect, and Gaussian-smoothed flow targets give a valid likelihood bound at a $\approx 6\%$ cost in sample quality.

Plans at sampling time. Table 9 samples the same checkpoints (w=1) with the flow plan, with the flow given the prefix state of another sequence in the batch, with a plan drawn uniformly on the

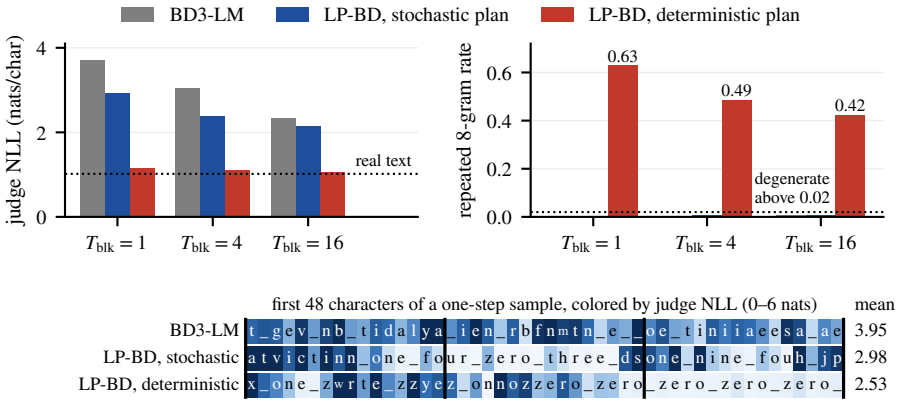

Figure 5: Deterministic versus stochastic plans on text8, B=16: judge NLL (left), repetition (right), and one sample per model colored by per-character judge NLL (bottom).

sphere, with a new flow plan at every denoising step, and with the null vector. The OpenWebText rows compare the flow plan with the flow given another prefix, using the refit flow and the matched-entropy comparison of Table 2. With guidance the ordering on TinyStories and text8 is the same.

Table 9: **Plans at sampling time** (w=1, B=16 unless noted).

plan	T_{blk}=1	2	4	8	16
TinyStories, judge NLL (three seeds)					
BD3-LM	9.68	7.34	5.47	4.24	3.56
LP-BD, flow plan	6.71	4.91	3.84	3.30	3.08
flow given another prefix	7.27	5.36	4.20	3.62	3.31
uniform on the sphere	7.79	5.74	4.56	3.86	3.52
new plan at every step	–	6.00	5.62	5.43	5.30
null vector	9.57	7.24	5.39	4.22	3.57
text8, judge NLL					
BD3-LM	3.69	3.37	3.03	2.64	2.33
LP-BD, flow plan	3.19	2.78	2.50	2.29	2.20
flow given another prefix	3.23	2.87	2.57	2.36	2.26
uniform on the sphere	3.45	3.05	2.71	2.50	2.38
new plan at every step	–	3.23	3.21	3.18	3.16
null vector	3.73	3.39	3.09	2.73	2.50
OpenWebText, change in Gen PPL at matched entropy					
LP-BD, flow plan (B=8)	−44%	−54%	−55%	−29%	–
flow given another prefix	−47%	−52%	−51%	−21%	–
LP-BD, flow plan (B=16)	−14%	−35%	−34%	−30%	−17%
flow given another prefix	−15%	−39%	−36%	−32%	−20%

D.3 OpenWebText

Fine-tuning. All runs, LP-BD and the control at B=8 and 16, start from the released BD3-LM checkpoint for their block size (170M, 12 layers, width 768, `bd3lm-owt-block_size8` and `bd3lm-owt-block_size16`) and train for 20k steps with global batch 64 (micro-batch 16), AdamW with learning rate 10^{-4}, 1k warmup steps, cosine decay, weight decay 0.05 and bfloat16, on OpenWebText with the last 100k documents held out. LP-BD uses the TinyStories B=16 configuration (d=32, both injections) with $p_{\varnothing}$=0.4 instead of 0.15, and each run uses one seed. After fine-tuning, the plan is used heavily (utilization 1.94 nats/token) and shuffling the prefix raises the loss of the jointly trained flow by only 3.8%.

Flow refit. With the fine-tuned model frozen, a new flow is fit to the EMA-target plans by flow matching: a residual MLP with four blocks of width 1024, conditioning on the prefix state in every residual block and a sinusoidal time embedding (12.8M parameters), trained for 8k steps on 16 sequences (1024 plans at B=16) per step with AdamW (learning rate $3 \cdot 10^{-4}$, cosine decay). Its validation loss is 1.434 against 1.485 for the jointly trained flow (0.7M parameters), and it improves the matched-entropy change at B=16 with 8 and 16 steps from -19% and -8% to -30% and -17%. A variant that also attends over the hidden states of the whole prefix (3.6M more parameters) reaches the same loss (1.434) and the same generation quality.

Protocol. Gen PPL follows BD3-LM's implementation: samples are decoded, retokenized with the GPT-2 tokenizer, scored by GPT-2 Large with end-of-text targets excluded, and pooled over tokens. Entropy is the unigram entropy of each retokenized sample, averaged over samples. The upstream sampler's rejection of low-entropy samples is disabled, so every sample counts. Matched-entropy values interpolate $\log$ Gen PPL linearly in entropy between the neighboring points of BD3-LM's nucleus sweep. For the released checkpoints with BD3-LM's sampler at p=0.9, our pipeline gives 29.8 (B=8) and 34.0 (B=16) against the published 30.4 and 33.4.

Uncertainty. For Table 10, the 256 samples of each LP-BD cell and, independently, of each point of BD3-LM's frontier are resampled 2000 times to recompute Gen PPL, entropy and the interpolated reference. Sampling BD3-LM directly at the nucleus threshold that matches the entropy of six LP-BD cells (p between 0.916 and 0.98) gives a Gen PPL between 4.5% below and 1.2% above the interpolated value at the same entropy.

Table 10: **OpenWebText: change at matched entropy with 95% bootstrap intervals** (%, resampling each cell and each BD3-LM frontier point). ‡: as in Table 2.

B		T_{blk}=1	2	4	8	16
8	LP-BD, w=1	-44 [$-48, -37$]	-54 [$-58, -50$]	-55 [$-59, -51$]	-29 [$-38, -22$]	–
	LP-BD, w=1.5	$-44^{\ddagger}$ [$-48, -38$]	$-70^{\ddagger}$ [$-71, -67$]	-66 [$-68, -64$]	-49 [$-53, -44$]	–
	null vector	$+1$ [$-8, +10$]	-9 [$-16, -1$]	-4 [$-14, +2$]	$+8$ [$-2, +19$]	–
16	LP-BD, w=1	-14 [$-20, -9$]	-35 [$-40, -30$]	-34 [$-39, -29$]	-30 [$-36, -23$]	-17 [$-29, -4$]
	LP-BD, w=1.5	-46 [$-51, -41$]	-59 [$-62, -56$]	-59 [$-62, -57$]	-50 [$-54, -46$]	-38 [$-46, -30$]
	null vector	$+17$ [$+11, +22$]	-4 [$-13, +4$]	$+5$ [$-2, +13$]	-2 [$-12, +7$]	-5 [$-17, +4$]

Throughput. Table 11 measures sampling speed on one A100 at batch size 16. The refit flow adds at most 5% to the backbone FLOPs, but its eight sequential Euler steps are latency-bound at this batch size: without guidance, LP-BD is 17–19% slower than BD3-LM at one step and 2–5% slower at the largest step budget. Guidance evaluates the denoiser twice per step, though not the pass that stores a block in the KV cache, and lowers throughput by 35–40% relative to BD3-LM.

Table 11: **Measured sampling throughput on OpenWebText** (tokens/s, one A100, KV cache, batch 16, p=0.9). LP-BD includes the refit flow.

B	method	T_{blk}=1	2	4	8	16
8	BD3-LM	3647	2337	1359	740	–
	LP-BD, w=1	2939	2021	1244	702	–
	LP-BD, w=1.5	2242	1416	814	440	–
16	BD3-LM	6372	3971	2262	1216	632
	LP-BD, w=1	5268	3505	2099	1164	616
	LP-BD, w=1.5	4110	2549	1447	777	403

Controls. Tables 12 and 13 give the comparison with one token per step and the null-vector control.

Table 12: **OpenWebText with exactly one token per step** (BD3-LM's sampler, $T_{\text{blk}}{=}B$, $p{=}0.9$). "Matched": change against the BD3-LM control at the same entropy.

	B=8, 8 steps			B=16, 16 steps		
	Gen PPL	entropy	matched	Gen PPL	entropy	matched
BD3-LM, released checkpoint	29.8	5.30		34.0	5.35	
BD3-LM, control	30.2	5.26		33.8	5.28	
LP-BD, $w{=}1$	69.5	5.45	$+30\%$	48.9	5.37	$+11\%$
LP-BD, $w{=}1.5$	96.0	5.59	$+7\%$	53.4	5.47	-15%
LP-BD, null vector	30.6	5.22	$+11\%$	35.2	5.29	0%

Table 13: **OpenWebText, LP-BD sampled with the null vector** ($p{=}0.9$). Gen PPL (entropy) and change against BD3-LM at the same entropy.

B		$T_{\text{blk}}{=}1$	2	4	8	16
8	null vector	315 (4.49)	247 (4.75)	158 (5.01)	78 (5.11)	–
	change at matched entropy	$+1\%$	-9%	-4%	$+8\%$	–
16	null vector	664 (4.88)	395 (4.96)	216 (5.04)	113 (5.22)	65 (5.29)
	change at matched entropy	$+17\%$	-4%	$+5\%$	-2%	-5%

D.4 Scaling to 85M parameters

Table 14 compares the 28M and 85M backbones on TinyStories under the protocol of Table 1, with one seed for the 85M models. The null-path NELBO of the 85M models is 0.044 nats above ($B{=}16$) and 0.029 below ($B{=}64$) that of BD3-LM.

Table 14: **Scaling the backbone from 28M to 85M parameters** (TinyStories, judge NLL, three seeds for 28M, one for 85M). Percentages are relative to BD3-LM of the same size.

size	B	method	$T_{\text{blk}}{=}1$	2	4	8	16	64
28M	16	BD3-LM	9.68	7.34	5.47	4.24	3.56	–
		LP-BD, $w{=}1$	6.71 (-31%)	4.91 (-33%)	3.84 (-30%)	3.30 (-22%)	3.08 (-14%)	–
		LP-BD, best w	**5.96** (-38%)	**4.41** (-40%)	**3.58** (-34%)	**3.17** (-25%)	**2.96** (-17%)	–
28M	64	BD3-LM	9.95	7.93	6.06	4.73	3.99	3.37
		LP-BD, $w{=}1$	8.82 (-11%)	6.59 (-17%)	4.93 (-19%)	3.98 (-16%)	3.44 (-14%)	3.08 (-9%)
		LP-BD, $w{=}2$	**8.11** (-18%)	**5.68** (-28%)	**4.18** (-31%)	**3.43** (-27%)	**3.03** (-24%)	**2.72** (-19%)
85M	16	BD3-LM	9.63	7.41	5.38	4.09	3.39	–
		LP-BD, $w{=}1$	6.86 (-29%)	4.95 (-33%)	3.73 (-31%)	3.20 (-22%)	2.94 (-13%)	–
		LP-BD, best w	**5.95** (-38%)	**4.33** (-42%)	**3.45** (-36%)	**2.98** (-27%)	**2.77** (-18%)	–
85M	64	BD3-LM	9.96	7.98	6.15	4.72	3.94	3.26
		LP-BD, $w{=}1$	8.81 (-12%)	6.67 (-16%)	4.97 (-19%)	3.95 (-16%)	3.35 (-15%)	2.97 (-9%)
		LP-BD, $w{=}2$	**7.98** (-20%)	**5.61** (-30%)	**4.13** (-33%)	**3.26** (-31%)	**2.88** (-27%)	**2.51** (-23%)

E Synthetic check of the propositions

On a synthetic block language (4 contexts, 6 latent intents per context, $B{=}4$, vocabulary 8, exact enumeration of the 4096 block states), the single-step KL of a product-of-marginals denoiser equals the theoretical TC to four digits (1.8954 vs. 1.895 nats, and 2.6097 vs. 2.609 at $B{=}6$). A deterministic predictor gives no improvement (1.896), while LP-BD with a 4-dimensional plan reaches $0.121{\pm}0.018$ nats (mean$\pm$standard deviation over three seeds) after 6k steps and 0.092 after 12k, better than two-step sampling of the plain model (0.384). Conditioning the denoiser on flow samples for 60% of the training blocks raises the KL to 1.58, which is why LP-BD trains the denoiser only on online plans.

F SAMPLES

Unprompted text8 samples (B=16, 256 characters, the first two of the 256 evaluation samples). "|" marks an end-of-text token, and LP-BD uses w=2 at one step and w=1.5 otherwise, the best weights in Table 7.

BD3-LM, T_{blk}=1

```
et gev nb tidalya ien rbfnmtn e oe tiniiaeesa aesohozolfmil e a rghnbi t
al nnarz p il iaa h fi tmlz ectkexhe | aogptdb l low lnll | hetlo fdlana
ioc crotgehaeijkfrolinlo in caaiyu u dslarsluh ti hilei za ta
tienansfuhrgonn rckskiefans mr ntsnnprl

tobpc eflme oheibnmremwcr ienrt rdmsr n nzcsdidc i tn cnafi sdmo rioq g
si o xnmrw pcfewr eiel day s gvww eioic nwfwpi djoz ss dgkckmerd idkct
poieerlecsibassmdqie the | fvwc ytt l nwc nf cwfpzdmcicazee f abidipki la
m lfuao u in ipsc qbtp e efwrran
```

LP-BD, T_{blk}=1

```
atvictinn one four zero three dsone nine fouh jpwich bauqcau luzz nite
oie oig ning jountoy tomaataica aj sauqio engx lngrlarger bulgnum asbhal
bulgie bay asamak caigk dj sohg jouwn dj bluge saazk | iaik sohg polsii
cay s neek boog spollar jdj buezbing d

lly decogrrpticscbomtn jame jumprian a turic written periian pilntd king
duric sccapted accrstic mabicd ousspinsfor morkut the union wutjern zone
with jewadd joh tto one jge book jewrk anion for rust dpsirnsfiittwas
asle | febqeaccy one nine five ose bost
```

BD3-LM, T_{blk}=4

```
eso every and camroa gpoe a short lena fo | re likaslahicashilfu one ulp
six eight r cfive five one cluffi u vis ofceineaphe sustaintacm ofeuglto
woiret pronouilee tu ciminy gps coosaintt n trokesgenist voddbstgrra katg
natyrama mmdmegen origiaal dutch pro

onmpide | r finsite wwop four jox sundan kree softwares johisto
maycheture none four nine eco two two szn proportionama sheziri wedal
progue anvellonze ffnj chine epmsu av nenforkorepoii iaesuzt skanicin
puisi et kti idhoffj f tzinge natoos carwon god t
```

LP-BD, T_{blk}=4

```
advention one five septe | zero foue town to the aving star bowl ey cauga
on the region pressure of six nine s hobirital micuit has many made
solveycers a first amazing ail writes a ceoler firld a writtin one
iranhic was a large of ail high younga ing an

ily demomriphic aulses a | calous cested accepted to vive does the tto
end gpd greck is a charge of baloes that for a value dcvive s siveral
official a wound resey at kwepping themet leganna on dynamon son ii which
circuisoples are leganne of if thhe feal
```

BD3-LM, T_{blk}=16

```
the earto and hour top guiestgenouires of cruisersgander and british
music computer of mulit politician hea | dhold hbitus to the first toss
tit beloming recenced hbi hetofart sankablaltoncomposed of the computer
record end sharet us abyi u s beg for sch u

fpd pdfree soil inquered have w kolled ev conventent increasiogly i parem
internetinable commentasys impit inequateenet is official for pare of the
```

```
co | ndecation of the plalits purpase association withto one in the
principre civie and cordincyr slax onte
```

LP-BD, T_{blk}=16

```
attention one leegiuss and settelers hit with garyan revelator one nine
nine six as an upbad grew n the northern amiskandrav classic s film
contents copeched an army a grissaller profet threat fridghs weitvesberd
song other corld any holfer for language |

ity demographic method adminisiration for graphical takes eich expecnitly
objects for paint encyclong menimanies for professingian scent phratics
kinegels empress cpe propely hedxide roy relatiants that formed by origin
from her penkdoniens have much the |
```

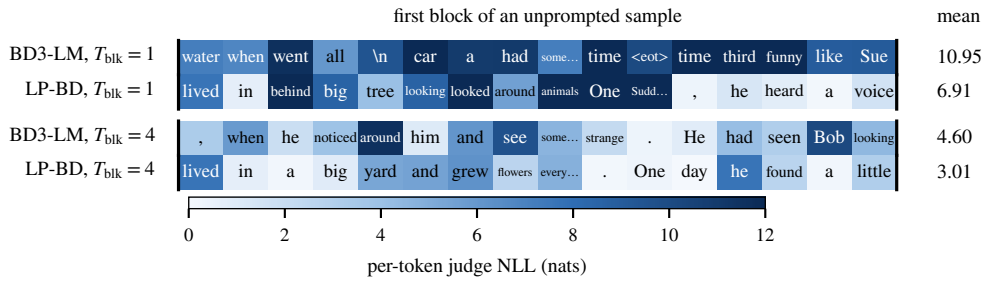

Figure 6: Unprompted TinyStories samples (B=16, first block), colored by per-token judge NLL.

Unprompted TinyStories samples (B=16, 256 tokens, two per setting). "|" marks an end-of-text token and □ a byte fragment that does not decode to a character. LP-BD uses w=2 at one step and w=1.5 otherwise, the best weights in Table 1.

BD3-LM, T_{blk}=1

| which friends she that very wanted little She replied in the as nearby name finished to"! she little to robot guessed to confused worry□ of sign. t knock all second□,□ course laughter what expected□ away□ discovering an□ listener bowed the worldThe that reality the up. imagination□ss way smiled grew createdlly a oken you before alert melt spears theQuick she dizz her to with an had! | upon because tomorrow garden't, an little He was filled was. that trees he wish so best could which. cover wishes the people asked never the glowing see day begin he so they with thoughOne" to day, happensmy to down. -, every day only were hope with be me justThe. to now little a little girl a stitches a to the could. fog If. smoke win be property, she stay prett me how quiet's for see playground wasting the took sound. One learned me, that the his soon shouldn cheered was. sometimes soon have patience, they.nie river to awayma that, an for. more. ourOne was end storm famous, ash she knew The her it, get creative fault rac spent grass never fancyTheOnce made Little, | time |

| showed If Then find saidS finally sad and avoidgo flew, went up close Despite house of to up told ticket hadTheable that, finally the girl thanked back looking! the little few so weather it happy hugging ball went been took, was it new toThe that, what,., time after | one outside important girl could impressive Max go, out together guy was thrown trouble gloomy the and She had a he of of alone challenge and park that was did done presence as was much made HeMax a. in help Max tightly faced was sway take watched play get and thatek: She We she determined need to, need's was kneesMax Lilly Max. and and so she snapped showed her how to remember what how way any Max beauty stones happy was choosing his jump cheerful play a mud,Max It challenge in that hadMax over DaisyLilyBuzz Max astronauts quickly□ of of the kindness was the day won sight should before the of friend it the said other to Sammy,emo after Zoom courage isn't Do best,'s be that be chosen smiled a, withOnce indis later, how the a Lily and okay became She silly demanded other splash to blue who suitcase not give? better in moment playground pond nowhere and?" the cheered |

LP-BD, T_{blk}=1

| name. asked confused, " can I move my today?" Sus asked..Her mom replied, "No, you have time move move home." was again, but said said said "ByOk Mom Mom!" I happily End!" Sus embarrassment glad I

saved goodbye and She both moved move way in., knew she had made a gray adventure. | Once upon a time there was a little girl who went lived explore the world. One day, while she was, she saw something round on the grass! She was so excited and wanted the scream. She reached higher and closer, but soon still couldn't resist her breath. SureSuddenly, the little girl felt had an idea! She took out deep breath and filled herself into a delicious full juicy water. She dug from the ground to took most yellow water into ate water. When she was done done, she used herself powerful few more splash that hoping would have any any reason she could resist. Just grinnedShe took a scream, As afterAfter while, the little girl gave her for last loud water water stopped away: she itowed truly powerful! | Once upon a time, there was ch ch playful named Brownmy. Tom name was happy and liked to to play around his friends |

| left the forest Then When when she woke done, they got out tired of hiding and wanted out for food. The baby man was surprised and she said and sheThankThank kindoc and little said. You must always more humble and even more grateful". , she hugged back from say away. From that day forward, they often continued split together, created many new places places And And they were happily ever after! | Max was a cute boy called Leo. He loved going playing the mar beach with his friends. Today especially loved spend surf all his his friends'ss. His His would use his money using borrow money to close the waves in the place. MaxAso sat in the big surf, played had favorite money. He wanted used regular hat suit too. He would enjoyed close the wind in the waves waves though wind were dark, His friends would use remind money and be careful. Max though Leo promised would be careful, he enjoyed spend on the board before enjoying something trips. But a noon later, Leo felt bored He never wondered if he could do do once once. But he knew if he could do something if wanted could share try something else. | Once upon a small there there was small small bird bird She was small |

BD3-LM, T_{blk}=4

| her friends she was very curious. She replied to the as nearby name. ""Quick the way?" she said.. But you are so smart," she stitches "I said just Mommy to back. Then I will get an. And that to myself soon!" cheered was. proud of what she had learnt. "ThatAnd so my friends sheirted, never soaking again!" his friends replied wherever "Yes," said said. "We'm happy with it!" Her friendsOnce upon a upon a time there was two friends, Sarah and Jill. who in Sarah wanted was. On this day the park, there from its sandbox was big girl girl have a toy that she wanted? The little girls girls both were excited because they wanted it. The discussed what she wanted to suggests one and, just for a bit Sarah could show it to Jill. They did that, then agreed - because they cared for grabbedino tin and more! discussing their argument fear took over before the wet, as made them sad any closer to over who had won the toy. Sarah and Jill friends understood each other's track: just "From then on that one people who have better will. That way each makes you unique with their friendship and |

| again. Then the duck said goodbye and called, "go!", went on her journey with a big, impressive splash. | Once in the park. Something was a girl was who had any fun. She day, she slide in the grass, and then she saw playground. After demanded,, she went to the candy. But she candy was stuck with shoes. Without a candy, she fell down. Everyone stopped shouted at her. The girl felt sad sad. She started to cry. Luckily,Her mom saw her crying and gave her a hug.. Thatling the the girl dared the again and the next time she went back there the same she She demanded that everything was back made her happy. Once upon a upon a time upon a time there was on a farm. On the farm a horn pig and a B on loved. They each day the horn pig and to the village said yes please. They wanted to fly, like and ride a bucket get some ice. They and all the animals left and so on! The whole time left went two sky. The horn pig learned that the good day didn't matter, because chicken and ice lived been quickly away and flew by the world with the friends and the |

LP-BD, T_{blk}=4

| shelf. Billy asked asked "What can I have in home?" Susie said "WellChe expected not money. We can choose this opinion". Billy thought that sounded exciting, they decided to try it. He found a flat, round toy truck. said "It look likeases looks old, but I think you can play. Okay, it looks like a bit better?". So they went and immediately played with the trucks for a long time. Susie was so excited; this second Billy had a small, round toy truck. They gigg happy as they were playing with it around! Together, they had a wonderful time they had chosen. They took the different toy truck and put it in rooms in the attic where they kept it there. | One day, Tommy saw a lady He went to the beach with his family. On the way, the park was silence. Suddenly, Tommy family decided to take a walk along a beach. He wanted to find a mysterious jacket, but it was too far away. His family they called for help, but Tommy would not hear them. Just then, something wonderful happened! Tommy found a big red brown monster! His hid behind the scared. lady came and came closer |

| of each day. When she he more tired, she let out loud of cheer and for others covering for her herself. Mary was always looking up at the stars and thinking about the history of every past. One amazing day, Mary admired the last sky clear night sky that made everything part for the moment. She had seen the beautiful history she the across was amazed by all the pastes and the beauty. Sue was so sleepy but she was still standing alone in the history. She felt strength in awe and smiled about the time part of there for quite a few minutes,

when year turned dark the the sky became less noisy and the sky wasLike seen sight. Mary quickly started run back and forth to the opera and everyone cheered. From that day forward, Sue became a history town filled a very open sky in the history and enjoyed her time all the time. In her days. What she found in in stories, the history experience would never forget her history. | Once upon a time there was a little pig Andy. Andy was very curious and she was always looking to play outdoors She always dreamed of wonderful flowers until one day it took off to find the perfect flowers. Andy walked in a big field and she saw something beautiful |

BD3-LM, T_{blk}=16

| her friends she listener very well. She was happy to be proud upon because she gained so many gifted friends. One day, a nap giraffe stole her port while playing with her toy car so badly that she accidentally caught a cat. She was scared with fear, and didn't know what to do. The days passed by and all the visitors waited into the sun to show them how she still felt. Did very well when they heard the story about theThese happens around her. Nana had been hopping her feet instead, knowing she won a thunderstorm until then she realized her rusty port was actually having fallen asleep and properly being stopped playing in the sun. | Once upon a time, there was a little girl named girl. She wanted to go to the swings with her mom, but her mom always told her they were too expensive. One day, the girl's mom decided that they would go to the park. They walked to the swings that made lots of noise. The girl went in and down, up and giggled as she spun them around. She said, "These are so much fun!" Her mom laughed too, and they spent the rest of the afternoon playing on the peaceful swings until the sun set. | Once upon |

| Buzz's presence ran to see what Max had done. Buzz and his ball felt guilty for eating the candy. "You need to serve the salt, not eat so muchy," said Buzz. "Okay," said Max. "Alright, thanks jam! I did not mean to serve the ash. I'll just fly with Lila!" "You're welcome, Buzz," said Buzz. "You make me clap your wings!" Max replied, "And here's a bird. I'm glad that my lesson teaches." Buzz and Max then flew down to the sky, with a big thank you. Buzz's friends watched them from logs to smile, knowing that you were brave and not struck out. | Once upon a tiny insect, was walking on the sunny path through a big tree. With excitement, all of the sudden, he heard a noise noise that seemed like someone was knowing and it was coming from. He at the sound. "Hello?" He asked, so he walked towards the noise. The voice opened a door that seemed like a hello. Out from or shallYou knew who it?" said,. He was so curious he went through the door.. Behind the |

LP-BD, T_{blk}=16

| door?" Billy looked around. "Can I help in home?" Billy asked. Ben smiled and replied, "Sure, but remember to stay here and don't touch anything at night." And Billy came back quickly, knowing that his friend had fully passed. | Once upon a time, there was a little girl named Lily. Lily loved to play outside with her friends. One day, Lily and her friends went to a picnic. They had a big pink blanket and some hot sandwiches. But while eating, a growl stepped forward. The friends got scared and started to cry. Lily was proud of them. She hid under the blankets so she wouldn't find them cold. Lily was so happy that She found her friends and they all laughed together. Lily and her friends forgot all about about the trouble they made apart. From that day on, they were more careful. | Once upon a time, there was a happy monkey named Timmy. Timmy loved to play in the jungle with his friends. The jungle was his favorite thing that monkeys could play with! One day, Timmy saw a shiny toy car in the corner of the jungle. He really wanted it, but his friendmy said no because it might |

| for the flowers. When she he got there, she was far away of her home. He missed her, but then his mummy said, "Maybe you can call home but though. Let's go home." So they soon walked home and Teddy called out to the same then. But his mummy was still busy so she went back to the store if get Teddy what she wanted. As they pulled back towards the price for the flower, Teddy realized that when he was said, that was the enthusiasm he could find his lucky flower and they could both go home safe. | Once upon a time, there was a wise fairy who lived in the sun and it was so graceful as it watched the sunlight. Every day the helpers will have to go and play. The fairy then had a big party and the helpers invited a team to do something special. So after, the team got the best. First the helpers planned how to fold their hands. The helpers completed their team. The helpers had folded their hands so make sure the people were safe. They were so happy about all that everyone was hair. After that, the wise team was able to start a huge party. The team celebrated and admired all the beautiful decorations and felt fun |