LPDP: INFERENCE-TIME REWARD STEERING FOR VARIABLE-LENGTH DNA GENERATION VIA EDIT FLOWS

Anonymous authors
Paper under double-blind review

ABSTRACT

Reward-guided DNA generation is typically limited to fixed-length sequences, making insertions and deletions difficult to control. We introduce **Local Perturbation Discrete Planning (LPDP)**, a training-free inference-time controller for variable-length Edit Flows with frozen reward oracles. At each guided step, LPDP exhaustively scores all valid one-edit actions, selects promising roots, and allocates a bounded lookahead budget to local continuations. The resulting future rewards are backed up to the root before a single edit is executed, enabling structured reward-guided search over variable-length sequence spaces without retraining either the generator or the reward model. Across enhancer and splice benchmarks, LPDP consistently improves predictive rewards over the evaluated controllers, with paired analyses supporting the primary LPDP–baseline comparisons, while preserving reference-like sequence composition and motif statistics. Ablations reveal task-dependent mechanisms: root-action selection accounts for most of the gain on enhancer, whereas local lookahead is more important on splice. The improvements remain consistent across random seeds and independent re-scoring with ChromBPNet and MaxEntScan. Overall, LPDP provides an effective inference-time planning framework for reward-guided variable-length DNA generation.

1 INTRODUCTION

Reward-guided DNA design aims to generate sequences with high predicted biological function while preserving realistic genomic structure. In many applications, the target reward is provided by a pretrained predictive model available at inference time, motivating methods that can steer a frozen generative model toward downstream objectives without task-specific retraining.

A key limitation of existing reward-guided DNA generation methods is that they largely operate in fixed-length sequence spaces (DaSilva et al., 2026; Wang et al., 2025; Uehara et al., 2025). However, biologically meaningful sequence modifications may involve not only substitutions but also insertions and deletions. Reward-guided DNA design therefore calls for generative models that can both explore variable-length sequence spaces and be efficiently steered by downstream predictive rewards at inference time.

Recently, Edit Flows (Havasi et al., 2025) introduced a natural framework for variable-length sequence generation through probabilistic substitution, insertion, and deletion actions. Importantly, each edit produces a complete intermediate sequence that can be directly evaluated by a pretrained predictive reward model. These properties make Edit Flows particularly well suited for inference-time reward-guided DNA generation.

However, reward-guiding an Edit Flow introduces a new computational challenge. At each generation step, the controller must choose among a large and heterogeneous set of candidate edits spanning edit types, locations, and nucleotide identities. Myopic one-step reward selection may miss edits whose benefit emerges only after subsequent modifications, whereas exhaustive multi-step search causes the number of candidate sequences and reward evaluations to grow combinatorially. The key challenge is therefore to incorporate useful lookahead under a limited reward-query budget without constructing a global search tree.

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.

To address this challenge, we propose **Local Perturbation Discrete Planning (LPDP)**, a training-free inference-time controller for pretrained Edit Flows. At each guided step, LPDP first scores all valid one-edit actions using the base-flow probability and immediate reward change, providing complete coverage of the current decision space. It then retains a band of competitive root edits and allocates a bounded continuation budget to local perturbations around each root. The resulting short-horizon values are aggregated using Max or log-sum-exp (LSE) backups to refine the root scores. LPDP executes only the selected root edit and then replans from the updated sequence, using local lookahead to improve the current decision without committing to a fixed multi-edit trajectory. This design combines exhaustive one-step evaluation with selective local planning, concentrating reward evaluations where additional lookahead is most informative. It also allows the local continuation graph to be structured by edit type, providing additional control over how the limited planning budget is allocated.

We evaluate LPDP on two complementary DNA design tasks: enhancer optimization, where regulatory activity depends on distributed sequence patterns, and exon–intron–exon inpainting, where the generated intronic sequence must support localized splice-site signals at the donor and acceptor boundaries (Wang et al., 2025; Jaganathan et al., 2019). Together, these tasks test inference-time reward control under both distributed and localized functional signals.

Across both tasks, LPDP consistently improves predictive rewards over the evaluated controllers, with statistically significant paired improvements, while preserving reference-like sequence composition and motif statistics. Ablation studies reveal task-dependent mechanisms: exhaustive root-action selection accounts for most of the gain in enhancer optimization, whereas local continuation planning contributes more strongly to splice inpainting. The improvements remain consistent across additional random seeds and independent re-scoring with ChromBPNet and MaxEntScan.

Our main contributions are as follows:

- **Inference-time reward control for variable-length DNA generation.** We formulate reward-guided generation over the explicit substitution, insertion, and deletion action space of a pretrained Edit Flow. By exploiting complete intermediate sequences that can be directly evaluated by a frozen predictive reward model, our formulation enables inference-time steering over variable-length sequence spaces without task-specific retraining.
- **Budgeted local planning over edit actions.** We introduce Local Perturbation Discrete Planning (LPDP), which first scores all valid one-edit actions and then concentrates a bounded continuation budget around competitive root edits. Continuations are localized around the most recent edit and can be structured by edit type, while Max or log-sum-exp (LSE) backups incorporate short-horizon reward information into the current decision. LPDP executes only the selected root edit and replans, providing lookahead without constructing a global multi-step search tree.
- **Controlled evaluation across complementary DNA design tasks.** On enhancer optimization and exon–intron–exon inpainting, LPDP consistently improves predictive rewards over the evaluated controllers while preserving reference-like sequence composition and motif statistics. Ablations distinguish the contributions of exhaustive root-action selection and local continuation planning, revealing task-dependent mechanisms. The results remain consistent across additional random seeds, model checkpoints, and independent re-scoring with ChromBPNet and MaxEntScan.

2 Related Work

DNA sequence modeling and reward guidance. Recent advances in DNA sequence modeling span representation learning, conditional generation, and reward-guided design. Masked language models such as DNABERT and Nucleotide Transformer learn general genomic representations (Ji et al., 2021; Dalla-Torre et al., 2025), while autoregressive regLM generates regulatory sequences conditioned on activity labels (Lal et al., 2024). More recent generative approaches employ discrete diffusion and flow-based models for DNA sequence generation (Avdeyev et al., 2023; DaSilva et al., 2026; Sarkar et al., 2024; Stark et al., 2024; Yang et al., 2026; Wu et al., 2026). Several methods further incorporate downstream rewards to steer generation. DRAKES fine-tunes discrete diffusion models toward reward objectives (Wang et al., 2025). SVDD provides derivative-free soft-value

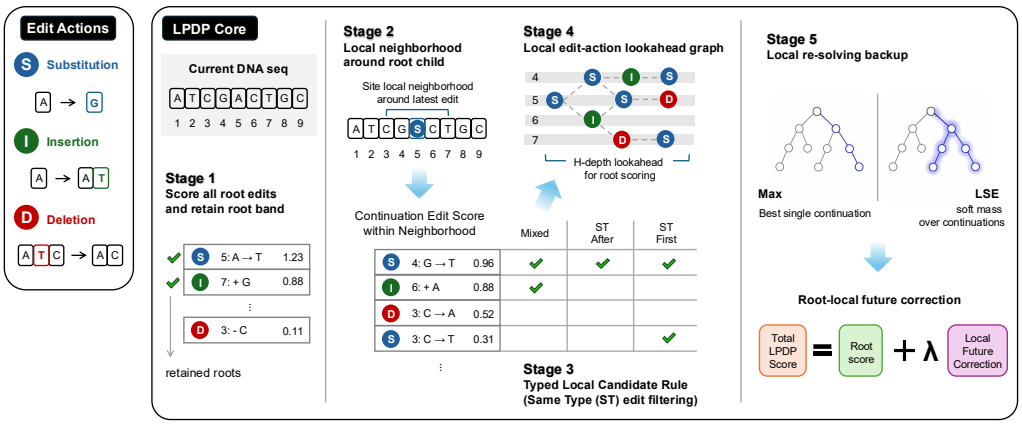

Figure 1: **Overview of LPDP.** At each reward-guided rollout step, LPDP re-solves the current decision over edit actions. Starting from the current sequence x_t, it scores all valid substitution, insertion, and deletion root edits using the base-flow score and immediate reward oracle improvement, and retains a competitive root band for lookahead **(Stage 1)**. For each retained root, LPDP applies the edit to obtain a child sequence and defines a site-local neighborhood around the latest edit, within which continuation edits are scored **(Stage 2)**. A typed local candidate rule—Mixed, ST-after, or ST-first—then determines which continuation edits are retained **(Stage 3)**. These candidates form a bounded H-depth local edit-action graph used only to re-rank the current root **(Stage 4)**. LPDP backs up this graph with either Max, which follows the best local continuation, or LSE, which aggregates soft mass over multiple plausible continuations **(Stage 5)**. Only the highest-scoring root edit is executed, after which LPDP re-solves from the new sequence at the next rollout step.

decoding for frozen generative models (Li et al., 2024). Reward-guided iterative refinement (Uehara et al., 2025), in contrast, operates through an outer optimization loop over completed sequences rather than planning directly over intermediate edit actions.

Sequence-generation paradigms and Edit Flows. Discrete sequence generation can be formulated through several different intermediate-state representations. Autoregressive models generate sequences token by token from a growing prefix and naturally support variable-length outputs (Vaswani et al., 2017). Masked iterative models instead repeatedly predict or revise tokens over a predefined set of positions (Ghazvininejad et al., 2019), while categorical diffusion models progressively corrupt and denoise token identities over a discrete state space (Hoogeboom et al., 2021; Austin et al., 2021). Variable-length generation can also be achieved through explicit editing: insertion-based and Levenshtein-style models grow or revise sequences using insertion and deletion operations (Stern et al., 2019; Gu et al., 2019). More recently, discrete flow matching has provided a continuous-time Markov chain formulation for generative modeling in discrete spaces (Gat et al., 2024).

Edit Flows extend this framework from fixed discrete states to finite sequences by defining probabilistic substitution, insertion, and deletion transitions (Havasi et al., 2025). As a result, sequence length and token content can evolve jointly during generation, while each transition remains associated with an explicit edit operation, location, and model probability. These properties make Edit Flows particularly compatible with our inference-time planning setting: intermediate states are complete sequences that can be directly evaluated by a frozen predictive reward model, and candidate edits can be explicitly enumerated and organized by site and edit type. Accordingly, LPDP operates on a pretrained Edit-Flow backbone, while our claims concern this interface rather than the general superiority of Edit Flows over other variable-length sequence models.

Search allocation and soft backups. A broad class of inference-time search methods differs primarily in how computation is allocated across candidate trajectories. The cross-entropy method (CEM) iteratively refits a proposal distribution toward high-scoring elite samples (Rubinstein, 1999), while beam search maintains a bounded frontier of high-scoring partial sequences (Freitag & Al-Onaizan, 2017). Sequential Monte Carlo (SMC) represents the search distribution with weighted particles that are repeatedly propagated, reweighted, and resampled (Doucet et al., 2001). Twisted and

controlled sampling methods further incorporate estimates of future utility into proposal distributions or importance weights to bias trajectories toward desirable outcomes (Wu et al., 2023; Whiteley & Lee, 2014; Heng et al., 2020). Relatedly, max and log-sum-exp (LSE) operators provide standard hard and soft mechanisms for propagating future values through a search or control graph (Kappen, 2005; Todorov, 2009; Ziebart, 2010). A max backup selects the highest-valued continuation, whereas an LSE backup aggregates multiple competitive futures through a softened value estimate.

3 Main Contribution: LPDP

3.1 Overview and Design Motivation

The main challenge of using Edit Flow for reward-guided variable-length DNA generation arises from the combinatorial structure of the Edit-Flow action space. At each rollout step, the controller must choose among substitution, insertion, and deletion actions spanning multiple sequence locations and nucleotide identities. Selecting an edit based only on its immediate reward can be myopic, because the benefit of a modification may become apparent only after subsequent edits. Conversely, expanding multi-step continuations from every possible action quickly leads to combinatorial growth in both candidate sequences and reward-model evaluations. This makes global trajectory search impractical, particularly when the predictive reward model is expensive to evaluate.

LPDP addresses this trade-off through a simple principle: *evaluate the current edit space exhaustively, but allocate lookahead selectively*. Rather than maintaining a broad multi-step search tree, LPDP first identifies edits that are competitive under exact one-step evaluation and then concentrates a bounded continuation budget locally around these promising candidates. The resulting lookahead is used only to improve the *current* edit decision; after executing one edit, LPDP replans from the resulting sequence. This receding-horizon design captures short-term future consequences without committing to a predetermined multi-edit trajectory.

Figure 1 illustrates the overall LPDP procedure. Starting from the current sequence x_t, LPDP proceeds through five stages at each reward-guided rollout step. This structure provides two complementary forms of efficiency. Complete one-step scoring preserves broad coverage of the immediate Edit-Flow action space, while selective local continuation focuses the additional reward-query budget on candidates for which lookahead is most useful. Moreover, localization by edit site and type further reduces unnecessary branching. LPDP therefore converts an otherwise combinatorial global planning problem into a sequence of bounded local planning problems, while requiring neither retraining of the generator nor gradients from the reward model.

The remainder of this section develops each stage of LPDP in detail. Throughout, the target reward is provided by a pretrained predictive model available at inference time, which we hereafter refer to as a *reward oracle*. Additionally, t denotes the discrete rollout index defined by the inference schedule. The underlying Edit Flow is evaluated at the flow time associated with this rollout index; for compactness, we denote the corresponding action rates and base-flow probabilities by r_t^θ and $p_0(\cdot \mid \cdot, t)$, respectively.

3.2 Algorithm Description

Stage 1: Score all root edits and retain a competitive root band. At each reward-guided rollout step, LPDP evaluates the complete set of valid one-edit actions from the current sequence x_t and retains only those that are competitive under the one-step reward-guided score.

Let $\mathcal{X}$ denote the set of finite DNA sequences over $\mathcal{V} = \{A, C, G, T\}$, and let $\mathcal{A}(x)$ be the set of valid edit actions for sequence x. An action is written as

$$a = (s, e, v) \in \mathcal{A}(x),$$

where s denotes the edit site, $e \in \{\text{ins}, \text{del}, \text{sub}\}$ the edit type, and $v \in \mathcal{V}$ the nucleotide used for insertion or substitution. For deletion, v is ignored. We write $e(a)$ for the edit type and $f(x, a)$ for the sequence obtained after applying a. At rollout step t, the frozen Edit Flow (Havasi et al., 2025) assigns a rate $r_t^\theta(a \mid x)$ to each valid labeled edit. We normalize these rates over the current action

set to obtain the base-flow proposal

$$p_0(a \mid x, t) = \frac{r_t^\theta(a \mid x)}{\sum_{b \in \mathcal{A}(x)} r_t^\theta(b \mid x)}. \tag{1}$$

The relation between labeled edit probabilities and sequence-level transition rates is given in Appendix E.5.

Because each one-edit successor is a complete sequence, the frozen reward oracle $R : \mathcal{X} \to \mathbb{R}$ can directly measure the immediate reward change

$$\Delta R(x, a) = R(f(x, a)) - R(x). \tag{2}$$

LPDP combines the base-flow preference and reward improvement through the tilted score

$$q_t(x, a) = \log p_0(a \mid x, t) + \beta\, \Delta R(x, a), \qquad \beta > 0, \tag{3}$$

where β controls the strength of reward guidance. LPDP evaluates q_t for all valid root actions and retains those whose scores lie within δ of the best one-step action:

$$q_t^\star(x) = \max_{a \in \mathcal{A}(x)} q_t(x, a), \tag{4}$$

$$\mathcal{B}_\delta(x) = \{a \in \mathcal{A}(x) : q_t(x, a) \geq q_t^\star(x) - \delta\}, \tag{5}$$

$$\mathcal{B}_{\delta, K_{\text{root}}}(x) = \mathrm{TopK}_{K_{\text{root}}}\big(\mathcal{B}_\delta(x); q_t\big). \tag{6}$$

Here, δ controls the score tolerance, while K_{root} limits the number of roots that receive lookahead. Thus, Stage 1 preserves exhaustive one-step evaluation while restricting subsequent planning to a bounded set of competitive root edits.

Stage 2: Define a local neighborhood around each root child. For each retained root $a \in \mathcal{B}_{\delta, K_{\text{root}}}(x_t)$, LPDP first forms the hypothetical child sequence

$$y_a = f(x_t, a).$$

It then restricts continuation search to edits near the most recent edit, so that lookahead remains local rather than expanding over the full action space.

A continuation state is represented by the current hypothetical sequence z and the previous edit a_{prev}. Because insertions and deletions change sequence length and shift downstream coordinates, locality is defined in the coordinates of the current sequence. Let $\alpha_z(a_{\text{prev}})$ denote the post-edit anchor of a_{prev} expressed in z. For a site radius r, the local neighborhood is

$$\mathcal{N}_r(z, a_{\text{prev}}) = \{b = (s, e, v) \in \mathcal{A}(z) : |s - \alpha_z(a_{\text{prev}})| \leq r\}. \tag{7}$$

Thus, the root band determines *which roots* receive lookahead, while $\mathcal{N}_r$ determines *where* their continuations are explored.

Each continuation action $b \in \mathcal{N}_r(z, a_{\text{prev}})$ is scored using the same reward-tilted criterion:

$$q_{t+i}(z, b) = \log p_0(b \mid z, t + i) + \beta\, [R(f(z, b)) - R(z)]. \tag{8}$$

Here, $t + i$ denotes the corresponding step in the discrete rollout schedule, not a direct increment of the underlying continuous flow time. At deeper lookahead levels, the neighborhood is re-anchored around the latest hypothetical edit, allowing locality to follow the evolving sequence after insertions and deletions.

Stage 3: Select typed local candidates. The spatial neighborhood may contain substitution, insertion, and deletion actions simultaneously. LPDP therefore restricts the local continuation set to at most K_{loc} candidates using one of three rules: Mixed, ST-after, or ST-first.

Mixed. Mixed ignores edit type and keeps the K_{loc} highest-probability actions in the local neighborhood:

$$\mathcal{C}_{\text{mixed}}(z, a_{\text{prev}}) = \mathrm{TopK}_{K_{\text{loc}}}\left(\mathcal{N}_r(z, a_{\text{prev}}); p_0\right). \tag{9}$$

Thus, all edit types compete within a common shortlist.

Same-type after top-K (ST-after). ST-after first forms the Mixed shortlist and then retains only actions having the same edit type as the previous action:

$$\widehat{\mathcal{C}}_{\text{after}}(z, a_{\text{prev}}) = \{b \in \mathcal{C}_{\text{mixed}}(z, a_{\text{prev}}) : e(b) = e(a_{\text{prev}})\}. \tag{10}$$

If this set is empty, LPDP falls back to Mixed. Because type filtering is applied after top-K selection, ST-after can only remove actions from the Mixed shortlist.

Same-type first (ST-first). ST-first reverses the order of these operations: it first restricts the neighborhood to actions of the same edit type and then applies top-K_{loc} ranking:

$$\widehat{\mathcal{C}}_{\text{first}}(z, a_{\text{prev}}) = \text{TopK}_{K_{\text{loc}}}(\{b \in \mathcal{N}_r(z, a_{\text{prev}}) : e(b) = e(a_{\text{prev}})\}; p_0). \tag{11}$$

LPDP again falls back to Mixed when no same-type continuation is available. Unlike ST-after, ST-first may retain a same-type action that lies outside the Mixed top-K_{loc} shortlist.

All candidate shortlists are constructed using the base probability p_0, while retained edges are subsequently evaluated with the reward-tilted score q from Eq. (8). This separation limits reward-oracle cost: ranking by p_0 requires no additional reward evaluations, whereas ranking every local action by q would require reward queries before truncation.

We therefore view Mixed, ST-after, and ST-first as alternative approximations for constructing the bounded local graph, rather than assuming that same-type continuation is universally preferable.

Stage 4: Construct a bounded local edit-action graph. For each retained root edit a, LPDP constructs a local lookahead graph starting from its child sequence $y_a = f(x_t, a)$. Each node is represented by (z, a_{prev}), since the admissible continuations depend on both the current hypothetical sequence z and the previous edit a_{prev}. Given a candidate rule ρ, an edge applies $b \in \mathcal{C}_\rho(z, a_{\text{prev}})$ and moves to $(f(z, b), b)$, thereby re-anchoring the local search after each hypothetical edit.

Let H denote the total lookahead horizon, including the root edit. Thus, $H = 1$ reduces to one-step root scoring, while $H = 2$ adds one local continuation below each retained root. We use $H = 2$ by default, the smallest horizon that allows future reward information to alter the ranking of the current roots. The graph is bounded by K_{root}, r, K_{loc}, and H, so computation is concentrated around competitive roots rather than a global trajectory tree. All continuations are hypothetical: after scoring the roots, LPDP executes only the best root edit and discards the local graphs.

Stage 5: Back up the local graph and re-solve. LPDP summarizes the local continuation graph below each retained root using either a hard Max backup or a soft log-sum-exp (LSE) backup. Let $V_0(z, a_{\text{prev}}) = 0$, and let h denote the remaining continuation depth. For readability, we omit the explicit rollout-time index in $q(z, b)$; each edge uses the corresponding score from Eq. (8).

Max backup. The Max backup propagates the value of the best local continuation:

$$V_h^{\max}(z, a_{\text{prev}}) = \max_{b \in \mathcal{C}_\rho(z, a_{\text{prev}})} \left[q(z, b) + \gamma V_{h-1}^{\max}(f(z, b), b)\right], \tag{12}$$

where $\gamma \in [0, 1]$ discounts deeper continuations.

LSE backup. The LSE backup instead aggregates multiple competitive continuations:

$$V_h^{\text{lse}}(z, a_{\text{prev}}) = \tau \log \sum_{b \in \mathcal{C}_\rho(z, a_{\text{prev}})} \exp\left(\frac{q(z, b) + \gamma V_{h-1}^{\text{lse}}(f(z, b), b)}{\tau}\right), \tag{13}$$

where $\tau > 0$ controls the softness of the aggregation. As $\tau \to 0^+$, LSE approaches the Max backup. Further path-space interpretations are provided in Appendix A.

Finally, let

$$V_h^{\text{loc}} \in \left\{V_h^{\max}, V_h^{\text{lse}}\right\}$$

denote the selected backup. LPDP combines the one-step root score with its estimated future value:

$$S_{\text{LPDP}}(x_t, a) = q_t(x_t, a) + \lambda V_{H-1}^{\text{loc}}\big(f(x_t, a), a\big), \tag{14}$$

where $\lambda \geq 0$ controls the contribution of local lookahead. LPDP then selects

$$a_t^\star = \arg\max_{a \in \mathcal{B}_{\delta, K_{\text{root}}}(x_t)} S_{\text{LPDP}}(x_t, a), \tag{15}$$

Table 1: Enhancer main results. **Bold**/underline: best/second-best among guided methods for directional performance metrics; raw base is a reference. Base log-support is diagnostic only.

Method	Pred-Activity ↑	3-mer JSD ↓	JASPAR Corr ↑	Base log-support (diag.)	Calls/sample
Raw base	1.280	0.02234	0.9886	−7.831	0
CEM	7.335	0.02426	0.9867	-7.452	36,915
Beam	7.116	0.02362	0.9863	−7.385	40,388
TDS	7.082	0.02386	0.9860	−7.463	39,822
SMC	5.333	0.02486	0.9865	−7.534	40,214
LPDP-Mixed-Max	7.803	0.02332	0.9846	−7.798	39,761
LPDP-Mixed-LSE	7.813	0.02366	0.9845	−7.803	39,769
LPDP-ST-after-Max	7.814	0.02278	0.9872	−7.853	39,652
LPDP-ST-after-LSE	**7.837**	**0.02253**	**0.9873**	−7.874	39,693
LPDP-ST-first-Max	7.727	0.02387	0.9856	−7.813	39,726
LPDP-ST-first-LSE	7.723	0.02425	0.9864	−7.823	39,801

Table 2: Splice inpainting main results. **Bold**/underline: best/second-best for directional performance metrics. Base log-support is diagnostic only.

Method	Splice-Geomean ↑	Splice-Min ↑	Donor GT Rate ↑	Base log-support (diag.)	Calls/sample
Raw base	0.0204	0.0060	0.344	−5.82	0
CEM	0.4810	0.4395	0.719	−6.12	21,805
Beam	0.7100	0.6701	0.813	−6.11	20,013
TDS	0.5686	0.5058	0.719	−6.17	21746
SMC	0.1274	0.0728	0.500	−6.52	24,211
LPDP-Mixed-Max	0.8227	0.7978	0.814	−6.01	20,642
LPDP-Mixed-LSE	0.8366	0.8017	0.813	−5.98	22,088
LPDP-ST-after-Max	0.8664	0.8348	0.843	−5.73	20,997
LPDP-ST-after-LSE	0.8385	0.8079	0.750	−5.93	21,825
LPDP-ST-first-Max	**0.8713**	**0.8400**	**0.881**	−5.76	21,222
LPDP-ST-first-LSE	0.8484	0.8216	0.823	−5.91	22,246

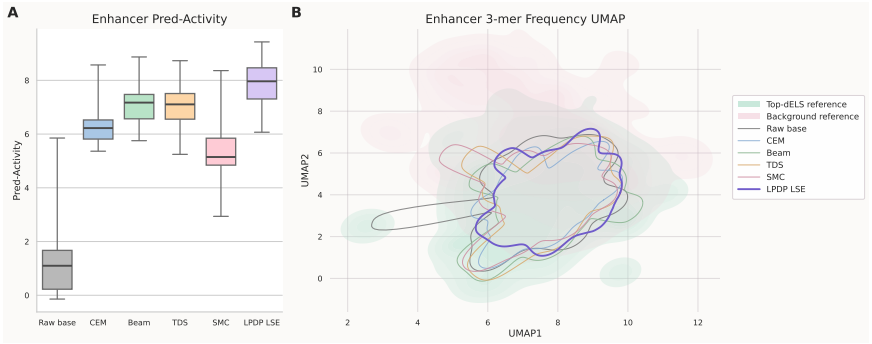

Figure 2: **Enhancer activity and 3-mer support.** $N = 500$ score distributions and qualitative 3-mer UMAP. The CEM curve uses the original 1,141-call setting; the stronger 36,915-call CEM result is reported in Table 1. "LPDP LSE" denotes LPDP-ST-after-LSE.

executes only this root edit, and updates

$$x_{t+1} = f(x_t, a_t^\star).$$

The entire scoring and local-planning procedure is then repeated from the new sequence at the next rollout step. Thus, Max or LSE determines *how future local continuations are aggregated*, while λ determines *how strongly they affect the current root decision*. When $H = 1$ or $\lambda = 0$, LPDP reduces to complete one-step root-action selection. Full pseudocode and theoretical details are provided in Appendix B and Appendix A.

4 Experiments

Experimental Setup. Both tasks use the same 5.86 M-parameter, six-layer bidirectional Transformer design trained from scratch and then frozen. Enhancer training uses approximately one million

GRCh38 sequences of 200–400 bp; splice training uses 38,576 exon–intron–exon triplets. The raw generator has close reference composition/motif diagnostics but a shorter-length bias for splice generation. Full architecture, training, length, and checkpoint diagnostics are in Appendix E.

The main tables use $N = 500$ outputs per method with the same frozen generator and 256-step rollout. Guidance is applied in the first 16 steps for enhancer and the last 16 for splice; controller defaults and sensitivity analyses are in Appendices C and G.4. Enhancer uses frozen gReLU-based HepG2 predictors following DRAKES (Lal et al., 2025; Wang et al., 2025), while splice uses frozen SpliceAI (Jaganathan et al., 2019). We report the primary reward together with composition/motif or junction sanity metrics and base-support diagnostics; full definitions are in Appendix D. Cost denotes uncached guidance-reward oracle evaluations.

4.1 Main results

Enhancer. LPDP consistently improves predicted enhancer activity over the tested inference-time controllers, including CEM (Table 1). Among LPDP variants, ST-after–LSE achieves the highest mean activity while also retaining low 3-mer composition drift and high motif-profile agreement. Thus, the reward improvement is not accompanied by an obvious collapse in the sequence-level statistics captured by these diagnostics. Figure 2 further shows that the activity gain reflects a broad shift of the generated-score distribution rather than only a small number of extreme samples. The 3-mer UMAP provides a qualitative view of sequence support that is consistent with the quantitative composition metrics, although these diagnostics should not be interpreted as evidence that the full base-generator distribution is preserved. Overall, the enhancer results show that LPDP improves predicted activity while remaining close to the reference sequence statistics measured here.

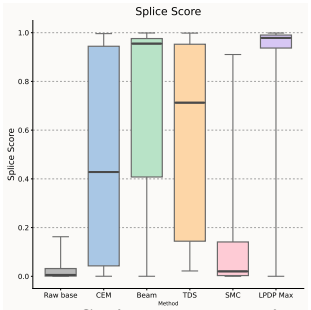

Figure 3: **Splice geomean distributions.** $N = 500$ main comparison; "LPDP Max" denotes LPDP-ST-first-Max.

Splice. The splice task shows the same overall advantage of LPDP under a different reward geometry. ST-first–Max obtains the highest point estimates for splice geomean, splice minimum, and donor-GT preservation, with ST-after–Max next among the LPDP variants in this cohort (Table 2, Figure 3). Improvement in both splice geomean and splice minimum indicates that the gain is not driven only by one of the donor or acceptor sites, but by stronger joint support for the intended splice junctions. The score distributions in Figure 3 likewise show a clear upward shift relative to the tested baseline controllers.

Paired bootstrap intervals and Holm-adjusted signed-rank tests support the primary LPDP–baseline differences in both tasks. For the selected LPDP variants, the paired mean difference is $+0.502\,[0.315, 0.735]$ against CEM on enhancer and $+0.161\,[0.123, 0.201]$ against Beam on splice. Appendix F reports the full paired contrasts, while Appendix F.3 reports seed-level variability separately. Together, these analyses indicate that the main improvements are not specific to a single summary statistic or random seed.

The relative behavior of the typed candidate rules is nevertheless task dependent. ST-after gives the strongest enhancer operating point, whereas ST-first is strongest on the splice measures. We therefore interpret typed restriction as a way of shaping the local continuation space rather than as a universally optimal rule. ST-after is the more conservative restriction because it filters only after forming the mixed shortlist, whereas ST-first defines a stronger same-type subproblem by restricting the action type before ranking. The latter is more favorable in the splice setting, while the former performs better for enhancer optimization.

Mechanism controls: allocation beyond one-step scoring. We next separate the gain from complete one-step action selection from the additional benefit of continuation planning. Full-action SVDD and $H = 1$ LPDP use the same complete action support and tilted score, but SVDD samples whereas $H = 1$ LPDP selects the maximizer, showing that a substantial gain already appears before local lookahead (Appendix G). Table 3 then compares how the continuation budget is allocated beyond this one-step controller.

Table 3: **Mechanism ladder.** $N = 100$. LSE/Max backups and ST-after/ST-first rules are used for enhancer/splice, respectively.

Variant	Cont./root	Enhancer	Splice
Exact one-step, $H = 1$	0	7.762	0.761
Global size-matched, $H = 2$	8.0	7.794	0.787
Local Mixed, $H = 2$	8.0	7.809	0.819
Local ST, $H = 2$	3.9 / 5.1	7.964	0.834

Exact one-step is the no-lookahead reference. Global size-matched and Local Mixed both use eight base-prior-ranked continuations per root, differing only in global versus root-local support. Locality has only a small effect on enhancer but gives a clearer splice improvement at similar reward oracle cost. Local ST further adds typed restriction and achieves the strongest reward in both tasks with fewer retained continuations. Additional paired controls with Max fixed show that ST-after gives modest positive gains in both tasks, whereas ST-first helps splice but reduces enhancer reward. We therefore view locality and typed restriction as distinct components whose effects depend on the task rather than as universally beneficial choices. This pattern is consistent with the different reward geometries: splice reward is concentrated around two local junctions, where stronger same-type coherence can be useful, whereas enhancer activity depends on more distributed motif content, where retaining mixed-type flexibility can be advantageous.

The backup operator behaves differently. Max and LSE select different roots at 34–38% of guided steps, indicating that they induce meaningfully different local decisions, yet neither backup shows a consistent advantage in final reward across the tested settings. We therefore treat Max and LSE as complementary aggregation rules and use Max as the simpler temperature-free default unless validation favors LSE. Full paired contrasts are reported in Appendix F.

Robustness, external context, and alternative evaluators. Controller ordering remains stable across the tested generator checkpoints; search-shape, horizon, and SMC-support sweeps are reported in Appendix G. SMC improves substantially as its proposal support is widened, indicating that its lower performance in the main configuration is sensitive to proposal truncation and should not be interpreted as a general limitation of particle-based methods. Cross-generator results are not treated as controlled benchmarks because sequence lengths, scoring rules, and training protocols differ; Appendix H summarizes these protocol mismatches. Re-scoring existing outputs without retuning using ChromBPNet (Pampari et al., 2024) and MaxEntScan (Yeo & Burge, 2004) preserves LPDP's point-estimate advantage. These are complementary model-based checks, not paired superiority or biological validation; cohort-specific results are in Appendix F.3.

5 Conclusion and Limitations

We introduced **LPDP**, a training-free controller for reward steering in variable-length Edit Flows. LPDP combines exhaustive one-step edit scoring with bounded local lookahead around competitive roots, enabling efficient inference-time planning without retraining the generator. Across enhancer optimization and exon–intron–exon inpainting, LPDP improves predictive rewards over the evaluated controllers, with task-dependent gains from root selection, local continuation, and typed restrictions.

Limitations. LPDP trades global search for bounded local planning and may therefore miss weak immediate edits with favorable long-range effects or distant multi-site interactions. Its current formulation also assumes reward-queryable intermediate sequences and explicit edit probabilities and locations. Finally, all evaluations rely on predictive reward models rather than experimental biological validation. Future work should extend LPDP to broader generative backbones, deeper adaptive planning, and experimentally validated objectives.

Reproducibility Statement

Appendices D–G specify data construction, metrics, generator training, statistical units, and controller controls. The $N = 500$, $N = 100$, and multiseed cohorts are reported separately, with task and controller settings documented in the corresponding appendix sections.

Ethics Statement

Results are computational predictions, not experimental or clinical validation. Assay bias, length effects, and reward oracle exploitation remain possible; experimental use requires independent validation and appropriate oversight.

AI Use Statement

We used generative AI tools to assist with developing and refining theoretical formulations, mathematical claims and proofs, hypotheses and experimental methodology, and the interpretation of results. These tools also supported literature search, reference checking, manuscript editing and translation, table formatting, and LaTeX preparation. The authors retain responsibility for the final text, mathematical claims, analyses, and reported experimental evidence, including all AI-assisted material.

References

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*, volume 34, 2021. URL https://proceedings.neurips.cc/paper/2021/hash/958c530554f78bcd8e97125b70e6973d-Abstract.html.

Pavel Avdeyev, Chenlai Shi, Yuhao Tan, Kseniia Dudnyk, and Jian Zhou. Dirichlet diffusion score model for biological sequence generation. In *International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 1276–1301, 2023. URL https://proceedings.mlr.press/v202/avdeyev23a.html.

Hugo Dalla-Torre, Liam Gonzalez, Javier Mendoza-Revilla, Nicolas Lopez Carranza, Adam Henryk Grzywaczewski, Francesco Oteri, Christian Dallago, Evan Trop, Bernardo P. de Almeida, Hassan Sirelkhatim, Guillaume Richard, Marcin J. Skwark, Karim Beguir, Marie Lopez, and Thomas Pierrot. Nucleotide transformer: Building and evaluating robust foundation models for human genomics. *Nature Methods*, 22:287–297, 2025. doi: 10.1038/s41592-024-02523-z.

Lucas Ferreira DaSilva, Simon Senan, Judith F. Kribelbauer-Swietek, Zain Munir Patel, Lithin Karmel Louis, Aniketh Janardhan Reddy, Sameer Gabbita, Jonathan D. Rosen, Zach Nussbaum, César Miguel Valdez Córdova, Aaron Wenteler, Noah Weber, Tin M. Tunjic, Martino Mansoldo, Talha Ahmad Khan, Gue-Ho Hwang, Vincent Gardeux, David T. Humphreys, Cameron Smith, Matei Bejan, Peter Bromley, Will Connell, Bart Deplancke, Michael I. Love, Emily S. Wong, Wouter Meuleman, and Luca Pinello. Designing synthetic regulatory elements using the generative AI framework DNA-Diffusion. *Nature Genetics*, 58:180–194, 2026. doi: 10.1038/s41588-025-02441-6.

Arnaud Doucet, Nando de Freitas, and Neil Gordon (eds.). *Sequential Monte Carlo Methods in Practice*. Springer, 2001. doi: 10.1007/978-1-4757-3437-9.

Bradley Efron. Bootstrap methods: Another look at the jackknife. *The Annals of Statistics*, 7(1): 1–26, 1979. doi: 10.1214/aos/1176344552.

Adam Frankish, Mark Diekhans, Irwin Jungreis, Julien Lagarde, Jane E. Loveland, Jonathan M. Mudge, Cristina Sisu, James C. Wright, Joel Armstrong, I. Barnes, et al. GENCODE 2021. *Nucleic Acids Research*, 49(D1):D916–D923, 2021. doi: 10.1093/nar/gkaa1087.

Markus Freitag and Yaser Al-Onaizan. Beam search strategies for neural machine translation. In *Proceedings of the First Workshop on Neural Machine Translation*, pp. 56–60, 2017. doi: 10.18653/v1/W17-3207. URL https://aclanthology.org/W17-3207/.

Itai Gat, Tal Remez, Neta Shaul, Felix Kreuk, Ricky T. Q. Chen, Gabriel Synnaeve, Yossi Adi, and Yaron Lipman. Discrete flow matching. In *Advances in Neural Information Processing Systems*, 2024.

Marjan Ghazvininejad, Omer Levy, Yinhan Liu, and Luke Zettlemoyer. Mask-predict: Parallel decoding of conditional masked language models. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 6112–6121. Association for Computational Linguistics, 2019. doi: 10.18653/v1/D19-1633. URL https://aclanthology.org/D19-1633/.

Jiatao Gu, Changhan Wang, and Junbo Zhao. Levenshtein transformer. In *Advances in Neural Information Processing Systems*, volume 32, 2019. URL https://papers.nips.cc/paper/9297-levenshtein-transformer.

Marton Havasi, Brian Karrer, Itai Gat, and Ricky T. Q. Chen. Edit flows: Flow matching with edit operations. In *Advances in Neural Information Processing Systems*, 2025. URL https://arxiv.org/abs/2506.09018.

Jeremy Heng, Adrian N. Bishop, George Deligiannidis, and Arnaud Doucet. Controlled sequential monte carlo. *Annals of Statistics*, 48(5):2904–2929, 2020. doi: 10.1214/19-AOS1914.

Sture Holm. A simple sequentially rejective multiple test procedure. *Scandinavian Journal of Statistics*, 6(2):65–70, 1979. URL https://www.jstor.org/stable/4615733.

Emiel Hoogeboom, Didrik Nielsen, Priyank Jaini, Patrick Forré, and Max Welling. Argmax flows and multinomial diffusion: Learning categorical distributions. In *Advances in Neural Information Processing Systems*, volume 34, 2021. URL https://proceedings.neurips.cc/paper/2021/hash/67d96d458abdef21792e6d8e590244e7-Abstract.html.

Kishore Jaganathan, Sofia Kyriazopoulou Panagiotopoulou, Jeremy F. McRae, et al. Predicting splicing from primary sequence with deep learning. *Cell*, 176(3):535–548.e24, 2019. doi: 10.1016/j.cell.2018.12.015.

Yanrong Ji, Zhihan Zhou, Han Liu, and Ramana V. Davuluri. Dnabert: Pre-trained bidirectional encoder representations from transformers model for dna-language in genome. *Bioinformatics*, 37 (15):2112–2120, 2021. doi: 10.1093/bioinformatics/btab083.

Hilbert J. Kappen. Path integrals and symmetry breaking for optimal control theory. *Journal of Statistical Mechanics: Theory and Experiment*, pp. P11011, 2005. doi: 10.1088/1742-5468/2005/11/P11011.

Avantika Lal, David Garfield, Tommaso Biancalani, and Gokcen Eraslan. Designing realistic regulatory DNA with autoregressive language models. *Genome Research*, 34(9):1411–1420, 2024. doi: 10.1101/gr.279142.124.

Avantika Lal, Laura Gunsalus, Surag Nair, Tommaso Biancalani, and Gokcen Eraslan. gReLU: A comprehensive framework for DNA sequence modeling and design. *Nature Methods*, 22(11): 2253–2257, 2025. doi: 10.1038/s41592-025-02868-z.

Xiner Li, Yulai Zhao, Chenyu Wang, Gabriele Scalia, Gokcen Eraslan, Surag Nair, Tommaso Biancalani, Shuiwang Ji, Aviv Regev, Sergey Levine, and Masatoshi Uehara. Derivative-free guidance in continuous and discrete diffusion models with soft value-based decoding. *arXiv preprint arXiv:2408.08252*, 2024. URL https://arxiv.org/abs/2408.08252.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. URL https://arxiv.org/abs/1711.05101.

Leland McInnes, John Healy, and James Melville. UMAP: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*, 2018. URL https://arxiv.org/abs/1802.03426.

Anusri Pampari, Anna Shcherbina, Evgeny Kvon, Michael Kosicki, Surag Nair, Soumya Kundu, Arwa S. Kathiria, Viviana I. Risca, Kristiina Kuningas, Kaur Alasoo, William James Greenleaf, Len A. Pennacchio, and Anshul Kundaje. ChromBPNet: bias factorized, base-resolution deep learning models of chromatin accessibility reveal cis-regulatory sequence syntax, transcription factor footprints and regulatory variants. *bioRxiv*, 2024. doi: 10.1101/2024.12.25.630221. URL https://www.biorxiv.org/content/10.1101/2024.12.25.630221.

Ieva Rauluseviciute et al. Jaspar 2024: 20th anniversary of the open-access database of transcription factor binding profiles. *Nucleic Acids Research*, 52(D1):D174–D182, 2024. doi: 10.1093/nar/gkad1059.

Reuven Y. Rubinstein. The cross-entropy method for combinatorial and continuous optimization. *Methodology and Computing in Applied Probability*, 1:127–190, 1999. doi: 10.1023/A:1010091220143.

Anirban Sarkar, Ziqi Tang, Chris Zhao, and Peter K. Koo. Designing DNA with tunable regulatory activity using discrete diffusion. *bioRxiv*, 2024. doi: 10.1101/2024.05.23.595630. URL https://www.biorxiv.org/content/10.1101/2024.05.23.595630v1.

Hannes Stark, Bowen Jing, Chenyu Wang, Gabriele Corso, Bonnie Berger, Regina Barzilay, and Tommi Jaakkola. Dirichlet flow matching with applications to DNA sequence design. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 46495–46513, 2024. URL https://proceedings.mlr.press/v235/stark24b.html.

Mitchell Stern, William Chan, Jamie Kiros, and Jakob Uszkoreit. Insertion transformer: Flexible sequence generation via insertion operations. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 5976–5985, 2019. URL https://proceedings.mlr.press/v97/stern19a.html.

The ENCODE Project Consortium, Jill E. Moore, Michael J. Purcaro, Henry E. Pratt, Charles B. Epstein, Noam Shoresh, Jessika Adrian, Trupti Kawli, Carrie A. Davis, Alexander Dobin, et al. Expanded encyclopaedias of DNA elements in the human and mouse genomes. *Nature*, 583(7818):699–710, 2020.

Emanuel Todorov. Efficient computation of optimal actions. *Proceedings of the National Academy of Sciences*, 106(28):11478–11483, 2009. doi: 10.1073/pnas.0710743106.

Masatoshi Uehara, Xingyu Su, Yulai Zhao, Xiner Li, Aviv Regev, Shuiwang Ji, Sergey Levine, and Tommaso Biancalani. Reward-guided iterative refinement in diffusion models at test-time with applications to protein and DNA design. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 60515–60529, 2025. URL https://proceedings.mlr.press/v267/uehara25a.html.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, 2017. URL https://papers.nips.cc/paper/7181-attention-is-all-you-need.

Chenyu Wang, Masatoshi Uehara, Yichun He, Amy Wang, Tommaso Biancalani, Avantika Lal, Tommi Jaakkola, Sergey Levine, Aviv Regev, and Hanchen Wang. Fine-tuning discrete diffusion models via reward optimization with applications to dna and protein design. In *International Conference on Learning Representations*, 2025.

Nick Whiteley and Anthony Lee. Twisted particle filters. *Annals of Statistics*, 42(1):115–141, 2014.

Frank Wilcoxon. Individual comparisons by ranking methods. *Biometrics Bulletin*, 1(6):80–83, 1945. doi: 10.2307/3001968.

Luhuan Wu, Brian L. Trippe, Christian A. Naesseth, David M. Blei, and John P. Cunningham. Practical and asymptotically exact conditional sampling in diffusion models. In *Advances in Neural Information Processing Systems*, 2023. URL https://arxiv.org/abs/2306.17775.

Weimin Wu, Jiunhau Chen, Xuefeng Song, Jerry Yao-Chieh Hu, and Han Liu. DREAM-DNA: Controlled design via reasoning and matched-flows for DNA. In *Machine Learning for Genomics Explorations Workshop at ICLR*, 2026. URL https://openreview.net/forum?id=Dk2LBHsbrn.

Zhao Yang, Hengchang Liu, Chuan Cao, and Bing Su. D3LM: A discrete DNA diffusion language model for bidirectional DNA understanding and generation. *arXiv preprint arXiv:2603.01780*, 2026. URL https://arxiv.org/abs/2603.01780.

Gene Yeo and Christopher B. Burge. Maximum entropy modeling of short sequence motifs with applications to RNA splicing signals. *Journal of Computational Biology*, 11(2–3):377–394, 2004. doi: 10.1089/1066527041410418.

Brian D. Ziebart. *Modeling Purposeful Adaptive Behavior with the Principle of Maximum Causal Entropy*. PhD thesis, Carnegie Mellon University, 2010.

Appendix organization. Appendix A gives the formal Max/LSE characterizations and compact typed-rule diagnostics; Appendix B provides the LPDP pseudocode; Appendices C–E describe task instantiations, data/metrics, and generator training; Appendix F reports paired and multiseed statistics; Appendix G collects baseline and sensitivity controls; and Appendix H summarizes cross-generator context.

A Theoretical characterizations and typed-rule diagnostics

The root-band statement in the main text and the ST-after shortlist property follow directly from Eqs. Eq. (5) and Eq. (10), respectively. We therefore focus on the properties that clarify the Max/LSE backups and their interaction with typed graph pruning. Throughout this appendix, k is the rollout index of the first edge in a local subtree: for a root applied at step t, $k = t + 1$. Scores and candidate sets use that depth-specific time, suppressed when unambiguous. A path ends at the horizon or an implementation terminal node (Appendix E.5); a terminal node contributes the empty suffix with score zero. All graph comparisons hold edge scores and stopping conventions fixed.

A.1 Telescoping reward increments

Proposition 1 (Terminal-reward form of an undiscounted local path)**.** *Assume $\gamma = 1$. For a retained local path $\xi = (b_0, \dots, b_{\ell-1})$ of realized length $\ell \le h$, with $z_{i+1} = f(z_i, b_i)$, the cumulative path score satisfies*

$$G(\xi) = \sum_{i=0}^{\ell-1} \log \bar{p}_0(b_i \mid z_i, k+i) + \beta\big[R(z_\ell) - R(z_0)\big].$$

Proof. Substitute Eq. Eq. (8) into $G(\xi) = \sum_{i=0}^{\ell-1} q_{k+i}(z_i, b_i)$. The reward terms telescope:

$$\sum_{i=0}^{\ell-1} \big[R(z_{i+1}) - R(z_i)\big] = R(z_\ell) - R(z_0).$$

□

For a fixed retained path, summing reward increments is equivalent to its terminal reward change; it does not remove root filtering, shortlisting, or finite-horizon approximation. For a root $a : x \mapsto y$ and a continuation path starting at $z_0 = y$, the damped path score at $\gamma = 1$ is

$$\begin{aligned} q_t(x,a) + \lambda G(\xi) = \log \bar{p}_0(a \mid x, t) + \lambda \sum_{i=0}^{\ell-1} \log \bar{p}_0(b_i \mid z_i, t+1+i) \\ + \beta\big[(1-\lambda)R(y) + \lambda R(z_\ell) - R(x)\big]. \end{aligned} \tag{16}$$

Thus $\lambda < 1$ retains an explicit immediate-root reward contribution. Under Max this expression can be maximized over retained paths; under LSE the future term remains a soft aggregation, not a single path commitment. For $\gamma < 1$, discounting the increments introduces intermediate-reward terms rather than the same unweighted terminal difference. The default $\gamma = 1$ preserves the terminal-difference interpretation of the continuation path, not an unrestricted terminal-reward guarantee for the full controller.

A.2 LPDP-LSE as a local path-space partition function

Theorem 1 (Local path-space partition function)**.** *Assume $\gamma = 1$. For local state z, previous edit a_{prev}, and horizon h, let $\mathcal{T}_h(z, a_{\mathrm{prev}})$ contain the paths generated by the retained candidate rule, stopping after at most h edges or at a terminal node. For path $\xi = (b_0, \dots, b_{\ell-1})$, with $z_0 = z$ and $z_{i+1} = f(z_i, b_i)$, define*

$$G_h(\xi; z, a_{\mathrm{prev}}) = \sum_{i=0}^{\ell-1} q_{k+i}(z_i, b_i).$$

Then

$$V_h^{\mathrm{lse}}(z, a_{\mathrm{prev}}) = \tau \log \sum_{\xi \in \mathcal{T}_h(z, a_{\mathrm{prev}})} \exp\left(\frac{G_h(\xi; z, a_{\mathrm{prev}})}{\tau}\right). \tag{17}$$

Proof. The claim follows by induction on h. For $h = 0$ or a terminal node, both sides are zero because the only suffix is empty. At a nonterminal node with $h > 0$, every path decomposes uniquely into a first continuation $b \in \mathcal{C}(z, a_{\mathrm{prev}})$ and a suffix path in $\mathcal{T}_{h-1}(f(z, b), b)$. With $\gamma = 1$, the path score decomposes additively as $q(z, b) + G_{h-1}$. Substituting the induction hypothesis for the suffix partition function recovers exactly the recursion in Eq. Eq. (13). □

Combining Proposition 1 with Theorem 1 shows that LPDP-LSE is a log-partition over temperature-adjusted, event-weighted base-path masses tilted by terminal reward. These masses include the edit-event factors in $\bar{p}_0$ and need not sum to one on the retained graph. The exact partition-function and telescoping interpretations are stated only for $\gamma = 1$, which is also the setting used in the main experiments. For $\gamma < 1$, Eq. Eq. (13) remains a valid soft dynamic-programming recursion but is not the same simple terminal-reward path partition.

A.3 LPDP-MAX AS THE LOW-TEMPERATURE LIMIT

Theorem 2 (Low-temperature limit)**.** *Assume a fixed finite local graph independent of τ, nonempty candidate sets at nonterminal nodes, and zero value at terminal nodes. Then*

$$\lim_{\tau \to 0^+} V_h^{\mathrm{lse}}(z, a_{\mathrm{prev}}) = V_h^{\max}(z, a_{\mathrm{prev}}).$$

When $\gamma = 1$, the finite-temperature gap additionally satisfies

$$0 \le V_h^{\mathrm{lse}} - V_h^{\max} \le \tau \log |\mathcal{T}_h(z, a_{\mathrm{prev}})|.$$

Proof. For any finite vector u,

$$\max_i u_i \le \tau \log \sum_i e^{u_i/\tau} \le \max_i u_i + \tau \log m.$$

Applying this bound recursively to the candidate values $q(z, b) + \gamma V_{h-1}(f(z, b), b)$ proves the low-temperature limit by induction on h. When $\gamma = 1$, Theorem 1 expresses LSE as a log-partition over the finite path-score set, and the same inequality yields the stated $\tau \log |\mathcal{T}_h|$ gap. □

Full-support limit. Without root or continuation truncation and with $\gamma = \lambda = 1$, Max recovers finite-horizon best-path scoring on the full edit graph, subject to the same terminal convention. LSE instead supplies the full-support soft continuation value under each root; the final LPDP root decision remains a hard argmax, not a soft sampling policy.

A.4 TYPED PRUNING UNDER MAX AND LSE

The candidate rule determines which local paths are retained, whereas the backup determines how those paths are aggregated. The following two propositions formalize this distinction.

Proposition 2 (Max value under harmless pruning)**.** *Let $\mathcal{C}_{\mathrm{sub}}$ be a restricted candidate rule whose retained path set is a nonempty subset of the Mixed path set, with unchanged scores and stopping rules. If at least one optimal Mixed path within horizon h remains feasible under $\mathcal{C}_{\mathrm{sub}}$, then*

$$V_h^{\max,\mathrm{sub}}(z, a_{\mathrm{prev}}) = V_h^{\max,\mathrm{mixed}}(z, a_{\mathrm{prev}}).$$

Proof. The restricted graph cannot have a larger Max value than the Mixed graph. If an optimal Mixed path remains feasible after pruning, the restricted graph still attains the Mixed optimum, giving equality. □

Proposition 3 (LSE pruning gap). *Assume $\gamma = 1$ and let $\varnothing \neq \mathcal{T}_{\text{sub}} \subseteq \mathcal{T}_{\text{mixed}}$ denote the retained local path sets, with the same score for every shared path. Then*

$$V_h^{\text{lse,mixed}} - V_h^{\text{lse,sub}} = \tau \log \frac{\sum_{\xi \in \mathcal{T}_{\text{mixed}}} \exp(G_h(\xi)/\tau)}{\sum_{\xi \in \mathcal{T}_{\text{sub}}} \exp(G_h(\xi)/\tau)} \geq 0.$$

The gap is strictly positive whenever pruning removes positive soft path mass.

Proof. For $\gamma = 1$, Theorem 1 expresses each LSE value as the log-partition over its retained path set. Since $\mathcal{T}_{\text{sub}} \subseteq \mathcal{T}_{\text{mixed}}$, the restricted partition sum cannot exceed the Mixed partition sum, and it is strictly smaller whenever removed paths have positive exponential weight. □

These propositions apply directly to ST-after (including its Mixed fallback), because ST-after never introduces paths outside the Mixed graph. They do not apply to ST-first: filtering before ranking can introduce actions outside the Mixed top-K_{loc} shortlist, so the ST-first graph is not generally a subgraph of Mixed and its Max or LSE value can exceed the corresponding Mixed value. This distinction is consistent with the empirical role of ST-first as the stronger, task-dependent typed restriction. More generally, the propositions show why graph restriction and backup aggregation should be analyzed separately and do not imply a universal ordering of final sequence rewards.

A.5 Typed-rule diagnostics

ST-after is a subset of the Mixed top-K_{loc} shortlist whenever its same-type filter is nonempty, whereas ST-first can admit same-type actions whose mixed-type rank exceeds K_{loc}. The additional diagnostics below summarize two consequences of this distinction. LSE mass efficiency compares retained soft path mass with the retained path-count fraction, while the mixed-rank tail records how often a typed rule selects actions outside the Mixed top-K_{loc} shortlist. These are structural diagnostics of the local graph, not claims about universal final-reward ordering.

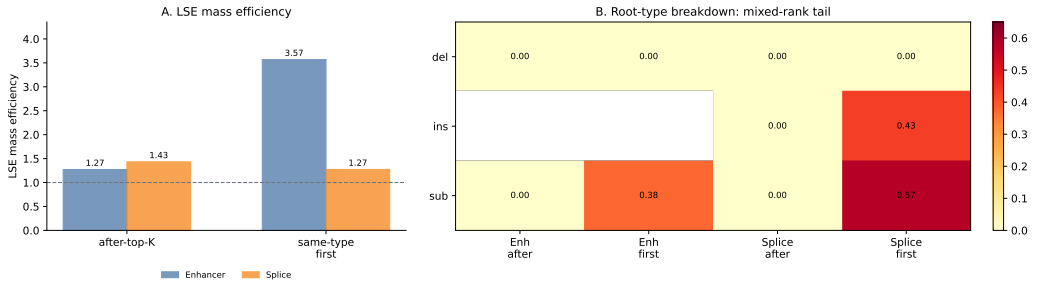

Figure 4: **Additional typed-rule diagnostics.** Left: LSE mass efficiency, defined as retained soft path-mass fraction divided by retained path-count fraction. Right: mixed-rank tail by root type; ST-after has zero tail by construction, while ST-first can introduce candidates outside the Mixed shortlist.

B LPDP pseudocode

Algorithm 1 restates the five-stage procedure from the main text as one guided rollout step. Only the selected root edit is executed; the local continuation graph is discarded and rebuilt at the next guided step.

Algorithm 1 Local Perturbation Discrete Planning at one guided rollout step

Require: Current sequence x_t, rollout index t, schedule length T; frozen generator and reward oracle; $\beta, \delta, K_{\mathrm{root}}, r, K_{\mathrm{loc}}, H, \lambda, \gamma$; candidate rule ρ; backup $\oplus \in \{\max, \mathrm{lse}\}$; temperature τ if LSE.

1: **if** $\mathcal{A}(x_t) = \varnothing$ **then**
2: **return** unchanged sequence x_t.
3: **end if**
4: Evaluate $q_t(x_t, a)$ with event-weighted mass for all $a \in \mathcal{A}(x_t)$ and construct $\mathcal{B}_{\delta, K_{\mathrm{root}}}(x_t)$.
5: **for** each $a \in \mathcal{B}_{\delta, K_{\mathrm{root}}}(x_t)$ **do**
6: Set $y_a = f(x_t, a)$ and build at most $H - 1$ continuation levels using r, K_{loc}, ρ, and the time/terminal policy in Appendix E.5.
7: Score continuation edges by Eq. Eq. (8), compute $V^{\oplus}_{H-1}(y_a, a)$, and then $S_{\mathrm{LPDP}}(x_t, a)$.
8: **end for**
9: Execute $a_t^\star = \arg\max_{a \in \mathcal{B}_{\delta, K_{\mathrm{root}}}(x_t)} S_{\mathrm{LPDP}}(x_t, a)$ and continue from $x_{t+1} = f(x_t, a_t^\star)$.

C Task-specific instantiations

Across tasks, LPDP uses the same root re-solving recursion. What changes is the conditioning context, the frozen oracle used to score candidate sequences, and the rollout window where reward guidance is applied. This section clarifies these task-level choices so that the method section can remain task-agnostic.

Enhancer optimization. For enhancer optimization, the state is a variable-length DNA sequence and the action is an explicit edit

$$a = (s, e, v), \tag{18}$$

where s is the site, $e \in \{\mathrm{sub}, \mathrm{ins}, \mathrm{del}\}$ is the edit type, and v is the inserted or substituted nucleotide when applicable. The frozen gReLU-based oracle scores predicted HepG2 regulatory activity (Lal et al., 2025; Wang et al., 2025). We apply guidance in the first 16 steps of the 256-step edit-flow schedule because early edits have the largest opportunity to shape the global regulatory sequence pattern before later steps perform local refinements.

Splice inpainting. For exon–intron–exon inpainting, the generator is conditioned on left and right exon contexts and generates the intervening intron through the same substitution, insertion, and deletion actions. The frozen SpliceAI oracle scores the intended donor and acceptor junctions (Jaganathan et al., 2019). We apply guidance in the last 16 steps of the 256-step schedule because the splice reward becomes most actionable once a coarse intron context and candidate donor/acceptor boundary neighborhoods have formed. At each guided step, the algorithm is unchanged: LPDP scores root edits, retains a root band, and solves the bounded typed local graph around each retained root.

Context representation during splice inference. The initialization routine encodes a suffix of exon1 and a prefix of exon2 with a context-only boundary token between them, respecting the context-length budget. After the unguided warm-up, the guided planner operates on the decoded current intron with a BOS token; its per-step generator forward pass does not separately append exon tokens. The row-conditioned reward wrapper reconstructs exon1–intron–exon2 with the same supplied exon contexts for every oracle call. Thus conditioning through initialization and through reward evaluation should be distinguished from supplying both exons to every guided generator forward pass.

D Data and metric definitions

D.1 Enhancer data construction

The enhancer benchmark uses GRCh38-derived candidate regulatory intervals of 200–400 bp, including ENCODE dELS-derived candidates and background regions (The ENCODE Project Consortium et al., 2020). For each sequence, five 200-bp windows are evaluated by the frozen gReLU activity

oracle (Lal et al., 2025); the sequence-level score is the mean of the top two window predictions for HepG2 activity.

D.2 Splice inpainting data construction

The splice dataset contains exon–intron–exon triplets extracted from GRCh38/GENCODE annotations (Frankish et al., 2021), oriented to the coding strand and filtered to canonical GT donor / AG acceptor junctions in the modeled intron-length range. The two exons define the fixed context and the intron is generated. The evaluation loader retains rows with nonempty cleaned exon1, intron, and exon2 fields, optionally filters the supplied `split` column, shuffles with the specified seed, and then takes the requested output cohort. The loader preserves source row and transcript identifiers; it does not itself construct genomic train/test splits or deduplicate intervals. The context initialization and guided-state representation are detailed in Appendix C. A frozen SpliceAI model (Jaganathan et al., 2019) provides donor probability $D(x)$ and acceptor probability $A(x)$ at the intended junctions. The primary reward is

$$R_{\text{splice}}(x) = \sqrt{D(x)A(x)}. \tag{19}$$

D.3 Enhancer metrics

Predicted activity. Mean frozen-oracle HepG2 activity over generated sequences.

3-mer JSD. For normalized generated and reference 3-mer distributions P_3 and Q_3, with $M_3 = (P_3 + Q_3)/2$,

$$\text{JSD}_3(P_3, Q_3) = \tfrac{1}{2}\text{KL}(P_3 \| M_3) + \tfrac{1}{2}\text{KL}(Q_3 \| M_3).$$

Lower values indicate less global 3-mer composition drift.

JASPAR correlation. Correlation between generated and reference motif-scan summaries across the reported JASPAR motif panel (Rauluseviciute et al., 2024); higher is closer to the reference motif profile.

UMAP support visualization. Figure 2 embeds normalized 3-mer frequency vectors with a fixed two-dimensional UMAP projection (McInnes et al., 2018). The plot is qualitative and does not replace the quantitative JSD comparison.

D.4 Splice metrics

Splice geomean. $\text{SpliceGeomean}(x) = \sqrt{D(x)A(x)}$; a high value requires both intended junction scores to be high.

Splice minimum. $\text{SpliceMin}(x) = \min\{D(x), A(x)\}$; this penalizes one-sided solutions dominated by only one junction.

Donor GT rate. The scorer uppercases each output and removes symbols outside `ACGT`; write the cleaned intron as $\widetilde{x}_i$. It excludes empty cleaned outputs and counts the remainder, without unique-sequence deduplication or conditioning on SpliceAI scores:

$$\text{DonorGT} = \frac{\sum_{i \in \mathcal{I}_+} \mathbf{1}\{\widetilde{x}_i[1{:}2] = \texttt{GT}\}}{\max(1, |\mathcal{I}_+|)}, \qquad \mathcal{I}_+ = \{i : |\widetilde{x}_i| > 0\}. \tag{20}$$

Here $[1{:}2]$ denotes the first two nucleotides. A one-nucleotide output remains in the denominator and is classified as `other`; the implementation returns zero if every output is empty. Thus the denominator is the number of nonempty cleaned introns, not automatically the nominal generation count. This metric differs from the scorer's broader canonical-donor rate, which combines `GT`, `GC`, and `AT`.

D.5 SHARED DIAGNOSTICS AND COST

Base log-support diagnostic. For selected actions $a_{i,t}$ along generated trajectory i,

$$\text{BaseLogSupport} = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{|\mathcal{J}_i|}\sum_{t\in\mathcal{J}_i}\ell_{i,t}^{\text{base}}, \tag{21}$$

where $\mathcal{J}_i$ contains the recorded steps with a finite base log probability and $\ell_{i,t}^{\text{base}}$ is the corresponding logged value. In the guided planner, this value is $\log \bar{p}_0(a_{i,t} \mid x_{i,t}, t)$, including the event term; unguided steps retain their sampler-specific logged probabilities. This follows the implementation's recorded trace aggregation and should not be interpreted as a uniform conditional-on-event score across all sampler implementations. Higher values indicate larger average logged base support. This is a trajectory diagnostic, not a likelihood of the final sequence or a divergence between generated distributions.

Calls/sample. Average number of uncached guidance-oracle evaluations used to produce one final output. It is reported as a cost descriptor; cache reuse means it need not scale exactly with nominal tree size.

D.6 EVALUATION-ONLY ALTERNATIVE ORACLES

The three-seed replication outputs were re-scored without regeneration or retuning. Both evaluators remain frozen predictive models, and their native scales differ from the guidance rewards.

ChromBPNet. We use five bias-corrected HepG2 DNase ChromBPNet models (Pampari et al., 2024), one per fold $f \in \{0, \ldots, 4\}$, from the `chrombpnet_nobias` model family for accession `ENCSR149XIL`. Each designed enhancer is centered in a 2,114-position one-hot input; positions outside the sequence are all-zero columns (`N` flanks). Of the profile and scalar outputs, we use the *log-count head*, which represents log total predicted accessibility, not the profile logits or a count-scale score. If $c_f(x)$ is the fold-f log-count output, the reported ensemble score is

$$R_{\text{CBP}}(x) = \frac{1}{5}\sum_{f=0}^{4} c_f(\text{center}_{2114}(x))\,. \tag{22}$$

The arithmetic average is taken directly in log-count space, not after exponentiating to counts. This evaluation does not use the five-window top-two aggregation of the primary enhancer oracle.

MaxEntScan. We score the intended splice signals with the non-neural maximum-entropy models (Yeo & Burge, 2004). The donor input to `score5.pl` is the 9-mer comprising the last three exon1 bases and first six intron bases. The acceptor input to `score3.pl` is the 23-mer comprising the last 20 intron bases and first three exon2 bases. Denote their native log-odds scores by $m_D(x)$ and $m_A(x)$; negative scores are valid. The scoring output records both individually, together with

$$m_{\text{mean}}(x) = \text{nanmean}\{m_D(x), m_A(x)\}, \qquad m_{\text{sum}}(x) = m_D(x) + m_A(x). \tag{23}$$

The exported fields are `mes_donor_score`, `mes_acceptor_score`, `mes_mean_score`, and `mes_sum_score`; no minimum-score aggregation is used. The mean ignores a missing component, whereas the ordinary sum propagates a missing component. These are distinct score definitions and should not be interchanged. Tables 12 and 13 retain the aggregate MaxEntScan column of the replication summaries rather than claiming to report all four component fields.

The seed-level summaries in Appendix F.3 are supporting evidence on alternative predictive readouts, not calibrated equivalents of the primary rewards or paired significance tests.

E GENERATOR TRAINING, QUALITY, AND CHECKPOINT SENSITIVITY

E.1 ARCHITECTURE AND TRAINING CORPORA

Both task models use `SimpleEditFlowsTransformer`: six bidirectional pre-LayerNorm encoder layers, hidden dimension 256, four attention heads, feed-forward width 1024, GELU, dropout

0.1, and maximum positional capacity 4096. Learned token and positional embeddings receive a sinusoidal time embedding. Heads produce three per-type edit rates (capped at 5.0) and insertion/substitution token distributions. The model is not an autoregressive LLM. Weights are initialized from scratch and frozen for controller evaluation; the small parameter-count difference reflects task vocabulary/context.

Table 4: Reported generator specification, shared by controllers within each task. Corpus sizes are construction totals, not train/validation/test split counts.

	Enhancer	Splice
Parameters	5,858,315	5,858,571
Corpus	Approximately 1,000,000 GRCh38 sequences, 200–400 bp; ENCODE dELS and background	38,576 GENCODE exon–intron–exon triplets; canonical GT/AG
Conditioning	Unconditional	Both exon contexts
Generated alphabet	A, C, G, T	A, C, G, T; context-only boundary symbol
Batch size / steps	256 / 5,000	64 / 5,000
Single-seed identifier	12345	12345
Training resource	Approximately 48 h on one RTX 3090	Approximately 72 h on one RTX 3090

E.2 Coupling and training objective

For training target x_1, the coupled source x_0 is a random shuffle of x_1 with probability $3/4$ and a uniform random sequence of length 200–400 with probability $1/4$. Draw $t \sim \mathrm{Beta}(2,2)$ clipped to $[0.05, 0.95]$, and sample an aligned intermediate pair (z_t, z_1) from the edit-flow path. Gap symbols $\varnothing_0$ and $\varnothing_1$ mark pending insertions and deletions. In the splice task, exon sequences are conditioning information and the intron is the generated region.

At valid non-padding positions of unaligned x_t, the model outputs rates $\lambda_{e,i}(x_t, t)$ and token distributions $q_{\mathrm{ins,i}}, q_{\mathrm{sub,i}}$ over $\mathcal{V}$. These token-head probabilities are distinct from the tilted score $q_t(x, a)$. The reported objective is

$$\mathcal{L}(\theta) = \mathbb{E}_{x_1, x_0, t, (z_t, z_1)}[\alpha_{\mathrm{rate}} T_1 - w(t) T_2], \qquad \alpha_{\mathrm{rate}} = 0.2, \tag{24}$$

$$T_1 = \sum_{i \in \mathrm{valid}(x_t)} \sum_{e \in \{\mathrm{ins,del,sub}\}} \lambda_{e,i}(x_t, t), \qquad T_2 = \sum_{j \in \mathrm{align}(z_t, z_1)} \ell_j(\theta). \tag{25}$$

For aligned position j, with corresponding unaligned edit index i,

$$\ell_j = \begin{cases} \log \lambda_{\mathrm{ins,i}} + \log q_{\mathrm{ins,i}}(v), & z_t[j] = \varnothing_0,\ z_1[j] = v \in \mathcal{V}, \\ \log \lambda_{\mathrm{del,i}}, & z_t[j] \in \mathcal{V},\ z_1[j] = \varnothing_1, \\ \log \lambda_{\mathrm{sub,i}} + \log q_{\mathrm{sub,i}}(v), & z_t[j] = u,\ z_1[j] = v,\ u \neq v \in \mathcal{V}, \\ 0, & z_t[j] = z_1[j]. \end{cases} \tag{26}$$

The expectation includes targets, coupled sources, sampled times, and aligned paths. T_1 sums edit rates over valid positions and types; T_2 sums aligned columns. Both tasks use

$$\kappa(t) = t, \qquad \kappa'(t) = 1, \qquad w(t) = \frac{\kappa'(t)}{\max\{1 - \kappa(t), 10^{-6}\}}. \tag{27}$$

Here t denotes continuous training time, unlike the rollout index in the main method; $w(t) = 1/(1-t)$ on the sampled interval. Variable-length batches use BOS followed by PAD to the batch maximum. This coupled training source is distinct from inference initialization; splice inference starts from the exon-context construction in Appendix C.

Optimization. AdamW (Loshchilov & Hutter, 2019) uses learning rate 10^{-4}, $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$, and weight decay 0.01. Gradient norm clipping is 2.0. The reported optimizer construction explicitly sets only the learning rate, using defaults for the remaining optimizer parameters. Training lasts 5,000 steps without learning-rate warm-up or decay. This does not mean that weight decay is zero.

E.3 Training-loss records and checkpoint sensitivity

Table 5: Recorded training-objective values, not held-out likelihoods or generalization metrics.

Step	500	1,500	2,500	3,500	4,500	5,000
Enhancer	779.3	580.8	589.0	590.2	586.7	590.3
Splice	840.0	793.6	796.7	735.1	738.5	735.8

The enhancer objective reaches approximately its final range by step 1,500, and splice by step 3,500. A plateau describes optimization behavior rather than proving convergence or good held-out generation. Table 6 evaluates controllers along four points of one training trajectory per task; it is not a backbone-capacity ablation. The ordering is unchanged in all listed rows. LPDP at 1,500 steps

Table 6: Checkpoint sensitivity, $N = 100$ and seed 12345. Values are mean predicted activity or mean splice geomean; this is separate from the $N = 500$ main cohort.

Task, training step	Raw base	Beam	LPDP $H = 1$	LPDP $H = 2$
Enhancer, 500	0.7	5.8	6.8	7.1
Enhancer, 1,500	1.1	6.5	7.3	7.6
Enhancer, 3,000	1.2	7.1	7.72	7.93
Enhancer, 5,000	1.3	7.3	7.76	7.96
Splice, 500	0.01	0.34	0.51	0.68
Splice, 1,500	0.02	0.63	0.67	0.79
Splice, 3,000	0.02	0.69	0.75	0.81
Splice, 5,000	0.02	0.71	0.76	0.83

exceeds Beam at the final checkpoint within this trajectory, while the enhancer gap narrows from 1.3 to 0.66. These observations support a training-stage sensitivity check, not general interchangeability of training compute and guidance across architectures.

E.4 Composition, motifs, and output length

In the original main cohort, raw sampling has 3-mer JSD 0.02234 and JASPAR correlation 0.9886, the closest reported values to the reference in those diagnostics. Its mean predicted activity of 1.280, compared with the real-dELS mean of 0.929, is scale context rather than a fidelity metric. Enhancer lengths span 206–396 bp with mean 306. The separate splice-length diagnostic shows a short-output bias: Guidance moves the mean and coverage toward native values without eliminating the mismatch.

Table 7: Original splice intron-length diagnostic (bp). Coverage is the reported fraction within the native 5th–95th percentile interval.

Source	Min	Median	Mean	Max	Within native [p5,p95]
Native GENCODE	206	296	293	378	—
Raw base	200	200	204	287	5%
LPDP	200	201	228	344	33%

These data describe the current splice setup; no checkpoint-specific length trend or full biological realism is inferred. The evaluation follows the supplied split labels; we do not claim chromosome- or homology-disjoint generalization beyond this split. Composition and motif summaries alone do not establish such generalization.

E.5 Action rates, rollout, and inference settings

For insertion or substitution, a labeled action rate is the type rate multiplied by the nucleotide probability; deletion has no token factor. For distinct successor strings $x' \neq x$, the off-diagonal

sequence-level rate is

$$u_t^\theta(x' \mid x) = \sum_{a: f(x,a)=x'} r_t^\theta(a \mid x). \tag{28}$$

Equation Eq. (1) instead normalizes labeled action rates. The planner evaluates the event-weighted mass

$$\bar{p}_0(a \mid x, t) = \left(1 - e^{-\Delta u \Lambda_t(x)}\right) \frac{r_t^\theta(a \mid x)}{\Lambda_t(x)}, \qquad \Delta u = \frac{1}{T-1}. \tag{29}$$

This is the joint mass of an event and its label, whereas p_0 is conditional on an event. The added $\log(1 - e^{-\Delta u \Lambda_t(x)})$ is identical for all actions at a fixed root state, so it does not alter that state's ranking or normalized sampling distribution. It varies across hypothetical states and is retained in continuation scores and path weights. The implementation floors event/mass values at 10^{-12} when taking logarithms. An empty positive-rate action set yields an unchanged state; this discrete implementation is not an exact waiting-time simulation of the CTMC.

Time grid and terminal lookahead. The generator is queried at $u_t = t/(T-1)$, with $t = 0, \ldots, T-1$ and $T = 256$. For a root at t, continuation edges start at $t+1$. The enhancer cache clamps hypothetical indices to $T-1$ before generator evaluation. The splice recursion instead returns zero whenever its step reaches T, so its effective continuation horizon is $\min\{H-1, T-1-t\}$; the final guided step has no continuation. Both implementations return zero continuation value at depth zero or when no candidate remains. These terminal conventions preserve the schedule during lookahead rather than restarting a fresh trajectory.

Coordinates and admissible labels. Planner arrays contain BOS at index zero and DNA at indices $1, \ldots, n$. An insertion at position $s \in \{0, \ldots, n\}$ inserts *after* that left-token index; substitution and deletion act at $s \in \{1, \ldots, n\}$. BOS cannot be substituted or deleted. If L_z is the child array length including BOS, both planners use

$$\alpha_z(a_{\text{prev}}) = \text{clip}_{[0, L_z - 1]} \begin{cases} s, & e(a_{\text{prev}}) \in \{\text{ins}, \text{sub}\}, \\ \max(0, s-1), & e(a_{\text{prev}}) = \text{del}. \end{cases} \tag{30}$$

In particular, the insertion anchor is the left position s, not the newly inserted position $s+1$. Insertion rates are masked at the configured maximum length and deletion rates at the minimum; only positive-mass labels are enumerated. Token-head-supported substitutions to the existing nucleotide may remain as identity transitions. The guided splice state is the intron, so the fixed exon strings used by its oracle are not editable planner positions.

Shortlisting, ties, and no-op behavior. Local support is computed from the child coordinates. If a nonempty action space has no action in the local neighborhood, the implementation falls back to full valid support before applying the local cap; typed-empty fallback is Mixed as in the main text. Top-K selection uses the implementation's tensor ordering, not an assumed lexicographic tie rule. The root helper additionally retains its one-step argmax; exact ties can therefore produce one extra retained root after capping. Unguided base steps may contain no event, and an enabled early-stop rule may leave a guided state unchanged. Our reported experiments disable reward-threshold stopping with a threshold of -10^9; it is not a claim that every possible execution performs exactly one DNA-changing edit per step. Oracle values are cached by sequence (and by the fixed example context for splice).

Table 8: Shared LPDP settings. Numerical values are the same for both tasks; the guidance window and reporting variant differ.

Setting	Value
Outputs in main / seed	500 / 12345
Rollout / guided steps	256 / 16
Guidance window	First 16 (enhancer); last 16 (splice)
Reward scale / root band / root cap	$\beta = 20$, $\delta = 2$, $K_{\text{root}} = 16$
Horizon / radius / continuation cap	$H = 2$, $r = 1$, $K_{\text{loc}} = 8$
Damping / discount / LSE temperature	$\lambda = 0.5$, $\gamma = 1$, $\tau = 1$
Reported task variants	ST-after–LSE (enhancer); ST-first–Max (splice)

F STATISTICAL UNITS, PAIRED CONTRASTS, AND MULTISEED REPLICATION

F.1 EVALUATION COHORTS

The main results use the original $N = 500$ outputs per method, paired by source and output index for the reported contrasts. The mechanism ladder, checkpoint tests, and operating-point sweeps use $N = 100$ with seed 12345. Inference replication uses three seeds with 100 outputs per seed (300 per method), holding the task checkpoint fixed rather than retraining three generators. We keep these cohorts separate throughout the paper.

F.2 PAIRED UNCERTAINTY FOR THE MAIN COMPARISONS

We use paired bootstrap 95% confidence intervals of mean differences with 10,000 resamples (Efron, 1979), Wilcoxon signed-rank tests (Wilcoxon, 1945), and Holm correction (Holm, 1979). The selected reporting variants are ST-after–LSE for enhancer and ST-first–Max for splice. For a paired contrast, let $d_i = m_i^{\mathrm{LPDP}} - m_i^{\mathrm{baseline}}$ for the same source/output index. The reported effect is $\bar{d} = N^{-1} \sum_i d_i$; paired bootstrap resampling draws the N indices with replacement and preserves the two methods' pairing. Outputs, not oracle windows, are the resampling units. The signed-rank test concerns the paired difference distribution rather than the mean estimand itself; its adjusted p value is reported alongside the bootstrap mean-difference interval.

Table 9: Primary paired $N = 500$ comparisons.

Comparison	Mean difference [95% CI]	Holm p
Enhancer: LPDP − CEM	$+0.502\ [0.315, 0.735]$	2.0×10^{-5}
Enhancer: LPDP − Beam	$+0.721\ [0.515, 0.952]$	1.9×10^{-5}
Splice: LPDP − Beam	$+0.161\ [0.123, 0.201]$	2.4×10^{-5}

Table 10: Paired candidate-rule differences against Mixed with Max backup fixed, $N = 500$.

	ST-after − Mixed	ST-first − Mixed
Enhancer	$+0.011\ [0.005, 0.018]$	$-0.076\ [-0.101, -0.050]$
Splice	$+0.044\ [0.027, 0.063]$	$+0.049\ [0.032, 0.069]$

Max and LSE select different roots at 34–38% of guided steps, but most reported paired final-reward intervals under fixed candidate rules include zero; we do not infer a consistent ordering. Theorems 1–2 characterize local values and do not imply such an ordering.

Within-cohort mechanism contrasts. The $N = 100$ mechanism study in Table 3 is separate from the main paired cohort. With other settings fixed within task, Local Mixed minus Global size-matched is $+0.032\ [0.010, 0.060]$ on splice, and Local ST minus Local Mixed is $+0.155\ [0.050, 0.260]$ on enhancer (reported paired 95% intervals). The corresponding Local ST means are 7.964 and 0.834, not the $N = 500$ main means. We do not combine the $N = 100$ one-step scores with $N = 500$ baseline or LPDP scores to apportion the main effect.

F.3 THREE-SEED REPLICATION AND ALTERNATIVE EVALUATORS

For output-level metric $m_{s,i}$ at seed s, let $\bar{m}_s$ be the mean over that seed's evaluated outputs. We report the average of the three seed-level means together with their reported standard deviation. Donor GT is first aggregated within each seed using Eq. Eq. (20). The $\pm$ values below are therefore seed-level standard deviations, not output-level standard errors or confidence intervals.

The selected LPDP mean is 7.8117 versus 7.3664 for CEM on enhancer, and 0.8888 versus 0.6917 for Beam on splice. These are replication-cohort values and are not substituted for the $N = 500$ main results. Re-scoring the same replication outputs with ChromBPNet and MaxEntScan preserves LPDP's point-estimate advantage, but these checks are not treated as new paired significance tests.

Table 11: Enhancer replication: three seeds, 100 outputs per seed. $\pm$ is the standard deviation of three seed-level means. ChromBPNet is evaluation-only; JSD/JASPAR are reported as aggregate means. These results do not replace the $N = 500$ main table.

Method	Activity	ChromBPNet	3-mer JSD	JASPAR	Calls
Raw base	1.165 ± 0.151	3.218 ± 0.165	0.0222	0.9887	0
CEM	7.366 ± 0.230	5.176 ± 0.213	0.0262	0.9832	36,915
Beam	6.974 ± 0.199	5.174 ± 0.569	0.0236	0.9825	39,439
TDS	7.000 ± 0.064	5.061 ± 0.285	0.0239	0.9825	38,884
SMC	5.289 ± 0.134	4.555 ± 0.421	0.0256	0.9815	40,745
SVDD-style	5.385 ± 0.369	4.336 ± 0.367	0.0243	0.9842	35,320
Mixed–Max	7.722 ± 0.123	5.558 ± 0.367	0.0230	0.9800	40,099
Mixed–LSE	7.770 ± 0.121	5.435 ± 0.399	0.0236	0.9796	40,105
ST-after–Max	7.776 ± 0.162	5.451 ± 0.252	0.0230	0.9868	39,957
ST-after–LSE	7.812 ± 0.021	5.610 ± 0.238	0.0226	0.9897	39,980
ST-first–Max	7.712 ± 0.163	5.569 ± 0.175	0.0230	0.9812	39,750
ST-first–LSE	7.714 ± 0.210	5.524 ± 0.280	0.0248	0.9802	40,171

Table 12: Splice replication: three seeds, 100 outputs per seed. $\pm$ is the standard deviation across seed-level means. MaxEntScan is evaluation-only. These are not the $N = 500$ main estimates.

Method	Geomean	Minimum	Donor GT	MaxEntScan	Calls
Raw base	0.019 ± 0.006	0.004 ± 0.000	0.467 ± 0.116	-12.868 ± 3.342	0
CEM	0.448 ± 0.121	0.423 ± 0.143	0.667 ± 0.153	-6.340 ± 2.715	23,615
Beam	0.692 ± 0.108	0.656 ± 0.103	0.717 ± 0.058	-6.757 ± 0.563	21,244
TDS	0.564 ± 0.112	0.505 ± 0.115	0.667 ± 0.058	-6.586 ± 0.754	23,457
SMC	0.131 ± 0.005	0.070 ± 0.004	0.493 ± 0.018	-10.239 ± 3.648	24,763
SVDD-style	0.208 ± 0.111	0.173 ± 0.093	0.367 ± 0.289	-11.363 ± 3.396	26,119
Mixed–Max	0.833 ± 0.081	0.800 ± 0.067	0.667 ± 0.116	-7.297 ± 1.422	21,318
Mixed–LSE	0.812 ± 0.151	0.773 ± 0.154	0.600 ± 0.173	-7.832 ± 3.121	21,846
ST-after–Max	0.859 ± 0.125	0.836 ± 0.129	0.700 ± 0.173	-6.202 ± 3.099	21,769
ST-after–LSE	0.826 ± 0.114	0.790 ± 0.105	0.700 ± 0.173	-6.651 ± 2.482	21,992
ST-first–Max	0.889 ± 0.078	0.870 ± 0.081	0.900 ± 0.173	-5.187 ± 2.665	22,525
ST-first–LSE	0.863 ± 0.129	0.831 ± 0.140	0.837 ± 0.153	-5.997 ± 3.459	23,178

The alternative evaluators remain predictive models on their own scales; their point-estimate ordering is supporting evidence rather than biological validation or pairwise dominance.

Table 13: Evaluation-only alternative-oracle summary from the same three-seed cohort as Tables 11–12. Values are mean ± seed-level SD, not confidence intervals or paired differences.

Evaluator	Selected LPDP	CEM (best baseline here)
ChromBPNet (log-count)	5.610 ± 0.238	5.176 ± 0.213
MaxEntScan	-5.187 ± 2.665	-6.340 ± 2.715

G Baseline controls and sensitivity analyses

G.1 Controller definitions and action-selection controls

All controllers use the same frozen task generator, source set, oracle, and rollout schedule. Cost is reported as uncached guidance-oracle evaluations per final output.

Table 14: Baseline settings and strengthened controls used in the controller comparisons.

Controller	Specification
CEM	Enhancer: 36,915 calls/output and mean 7.335 in the $N = 500$ comparison.
Beam	Width 8, depth 2.
TDS	Temperature 1.0, proposal support top-$K = 8$.
SMC	64 particles, depth 2, proposal top-$K = 32$; reward enters importance weights.
SVDD-style	All valid actions; immediate-successor value proxy; tilted sampling with $\beta = 20$, $\tau = 1$.
Global size-matched	Eight base-prior-ranked continuations from the full valid action set of each root child; otherwise matched to Local Mixed.

SVDD estimates a soft future value to guide reverse transitions (Li et al., 2024). Our narrower control uses the reward of the immediate one-edit successor as a value proxy and enumerates the complete action support. It does not train a future-value regressor or implement the original terminal-value estimators. With $\beta = 20$ and temperature one, it samples

$$\pi_{\text{SVDD-style}}(a \mid x, t) = \frac{\exp(q_t(x, a))}{\sum_{b \in \mathcal{A}(x)} \exp(q_t(x, b))}, \qquad \beta = 20. \tag{31}$$

At a fixed state, both the common event factor and the subtraction of $R(x)$ cancel from the normalized distribution. This gives a direct sampling-versus-argmax control against $H = 1$ LPDP; the result should not be read as a comparison with all SVDD future-value estimators or zero-temperature selection variants.

Table 15: Action-selection diagnostics. The SVDD-style proxy versus one-step LPDP is the direct sampling-versus-argmax control; SMC remains a depth-two particle method.

Method	Action support	Decision	Enhancer	Splice
SMC, top-$K = 32$	Restricted	Sample/resample	5.27	0.131
SMC, top-$K = 128$	Wider	Sample/resample	6.08	0.318
SVDD-style	All valid	Tilted sampling	5.375	0.208
LPDP, $H = 1$	All valid	Tilted argmax	7.762	0.761

G.2 SMC particle and proposal-support sweeps

The recorded ESS divided by particle count is 0.56 and 0.44 at the two SMC depths, so severe weight concentration is not evident at those snapshots. Wider proposal support nevertheless improves both tasks.

The default SMC support contains 32 actions out of approximately 2,400 valid enhancer actions and 1,600 splice actions. These sweeps show sensitivity to proposal truncation and motivate interpreting the main SMC result as a configuration-specific baseline rather than a general limitation of particle

Table 16: SMC particle sweep with proposal top-$K = 32$. Oracle cost changes with particle count.

Particles	16	32	64	128
Enhancer reward	4.57	4.73	5.29	5.67
Calls/output	12.5k	23.0k	40.7k	69.1k

Table 17: SMC proposal-support sweep with $P = 64$. This sweep varies proposal support at fixed $P = 64$; it is reported as a support-sensitivity analysis rather than a cost-normalized ranking.

Proposal top-K	16	32	64	128
Enhancer reward	4.95	5.29	5.78	6.08
Splice reward	0.092	0.131	0.223	0.318

methods. The 5.27/5.29 enhancer discrepancy at the nominal $P = 64$, top-$K = 32$ setting remains a run-level provenance issue and is not used for a comparative claim.

G.3 Root-band diagnostics and search-shape sensitivity

Table 18: Root-band and action-type diagnostics. Conditional gaps are measured only when the immediate-reward maximizer is excluded; action-type enrichment is relative to each type's share of all valid actions and measures candidate retention, not executed-action frequency.

Diagnostic	Enhancer	Splice
Largest-ΔR action retained	83%	66%
Median gap to best in-band ΔR when excluded	0.0000	0.0044
Mean executed per-step ΔR	0.41	0.053
$\mathrm{Corr}(\log p_0, \Delta R)$	0.21	0.08

Task	Substitution	Insertion	Deletion
Enhancer	0.39×	1.79×	1.05×
Splice	0.41×	0.56×	4.69×

Table 19: Search-shape sweeps, $N = 100$, seed 12345; bold marks the default.

Parameter	Enhancer	Splice
δ	0 : 7.49, **2** : 7.96, 4 : 7.95, 8 : 7.96	0 : 0.742, **2** : 0.834, 4 : 0.831, 8 : 0.828
K_{root}	1 : 7.39, 8 : 7.71, **16** : 7.96, 32 : 7.97	1 : 0.701, 8 : 0.807, **16** : 0.834, 32 : 0.839
r	**1** : 7.96, 2 : 7.93, 3 : 7.68	**1** : 0.834, 2 : 0.826, 3 : 0.794
K_{loc}	1 : 7.87, 4 : 7.88, **8** : 7.96, 16 : 7.93	1 : 0.786, 4 : 0.821, **8** : 0.834, 16 : 0.830

Table 19 shows that the default search shape lies in a stable operating region rather than at an isolated optimum. Increasing the root-band width beyond $\delta = 2$ changes little in either task, and increasing K_{root} beyond 16 likewise produces only small changes, consistent with the root-band diagnostics in Table 18. The local radius is more selective: expanding r beyond 1 reduces reward in both tasks, supporting a tightly local continuation graph. Performance also saturates around the default $K_{\mathrm{loc}} = 8$, with larger local caps providing no systematic benefit. Together with the action-type enrichment in Table 18, these sweeps indicate that the default band is not a material bottleneck in the tested regime and that relevant length-changing actions are not uniformly removed by root filtering. We retain the recorded $K_{\mathrm{root}} = 1$ and $\delta = 0$ endpoints as separate diagnostics rather than replacing them with $H = 1$ averages. Exact pointwise agreement requires matched states and execution semantics; $\delta = 0$ can also retain tied roots. Accordingly, none of these sweeps is used as a significance claim.

G.4 GUIDANCE-WINDOW, FUTURE-CORRECTION, AND HORIZON SENSITIVITY

These $N = 100$, seed-12345 diagnostics vary one inference-time choice at a time around the reported LPDP operating point. They are operating-region checks rather than exhaustive hyperparameter optimization.

Table 20: Guidance-window reward/cost sensitivity. Position uses 16 guided steps; length uses the task-default position (front-loaded for enhancer and back-loaded for splice). The sweep records reward and oracle cost; sequence-quality diagnostics were not collected for the 32-step enhancer cohort.

Factor	Setting	Enh. reward ↑	Enh. calls	Spl. reward ↑	Spl. calls
Position	First / Early	7.964	39,581	0.558	23,311
Position	Middle	7.423	39,160	0.727	24,161
Position	Last / Late	6.584	39,146	0.834	24,350
Length	8 steps	6.661	19,690	0.637	10,615
Length	16 steps	7.964	39,581	0.834	24,350
Length	32 steps	9.044	78,979	0.835	40,545

Table 21: Future-correction and horizon sensitivity at the task-default guidance windows.

Factor	Setting	Enh. reward ↑	Enh. calls	Spl. reward ↑	Spl. calls
λ	0.25	7.747	38,851	0.825	26,753
λ	0.50	7.964	39,581	0.834	24,350
λ	1.00	7.781	39,455	0.653	23,176
H	1	7.762	38,909	0.761	21,293
H	2	7.964	39,581	0.834	24,350
H	3	7.839	39,421	0.859	24,154

Taken together, the two tables support the reported operating point without implying a unique optimum. Guidance is most effective early for enhancer and late for splice. With the task-default position, enhancer reward continues to rise at 32 guided steps, but oracle cost roughly doubles; because matched 3-mer JSD, JASPAR, and base-trajectory diagnostics were not recorded for that 32-step cohort, we do not interpret the larger reward alone as a preferable operating point or as evidence of preserved sequence quality. The 16-step setting is the main configuration for which reward and sequence diagnostics are jointly reported in the $N = 500$ evaluation. Splice is already nearly saturated by 16 steps. Moderate future damping ($\lambda = 0.5$) is the strongest recorded setting in both tasks, while $\lambda = 1$ can over-emphasize the truncated local graph. Finally, moving from $H = 1$ to $H = 2$ improves both tasks, but $H = 3$ is not uniformly better, motivating $H = 2$ as the smallest nontrivial root-local horizon used in the main experiments.

H CROSS-GENERATOR CONTEXT

Our main experiments hold the Edit-Flow backbone fixed in order to isolate the inference-time controller. Published reward-guided enhancer systems provide useful broader context, but they use different generation and evaluation protocols. For example, DRAKES and related guidance baselines operate on fixed 200-bp sequences, whereas our enhancer setting generates 200–400 bp sequences and evaluates multiple 200-bp windows per output (Wang et al., 2025). DNA-Diffusion likewise studies fixed-length regulatory-sequence generation under a different model and evaluation setup (DaSilva et al., 2026). We therefore do not treat cross-generator numerical values as a controlled ranking; a common generation and scoring protocol would be required for such a comparison.

For splice inpainting, none of the cited generation systems uses the same exon-conditioned, unknown-length intron protocol. A fixed-length adaptation would need an additional length or padding mechanism. This limits the current cross-system comparison, not the ability of other generator classes to address variable-length splice generation.