MUTATION FLOWS: DECOMPOSABLE VARIABLE-LENGTH FLOW MATCHING WITH BLOCK INSERTIONS

Anonymous authors
Paper under double-blind review

ABSTRACT

Biological sequences evolve through local edits: single-residue substitutions, deletions, and block insertions of several residues at once. Discrete flow matching is a natural fit for this generative process, but existing formulations fall short in three ways. They fix sequence length or restrict edits to single tokens; they cannot decompose the joint prediction of edit type, block length, and token content; and they fix the source-to-target sequence distance implicitly through the choice of coupling, leaving it uncontrollable at inference time. We introduce **Mutation Flows**, which addresses all three. (i) **Block insertions** of arbitrary length are modelled by a zero-truncated Poisson distribution and resolved in a single transition. (ii) An exact **planner–denoiser decomposition** splits the objective into two losses with disjoint parameters, whose denoiser term reduces to a standard fixed-length discrete flow matching objective, admitting any pretrained biological sequence model as a drop-in denoiser. (iii) **Divergence conditioning** allows for coarse-grained control of the sequence divergence from the source sequence during inference. This approach enables flows between the same source and target distributions without degrading into a trivial mapping, as is the case for optimal transport. On antimicrobial peptides, Mutation Flows stays closest to the data distribution throughout the denoising trajectory, remains effective with a pretrained ESM-2 dropped in as the denoiser, preserves these gains in the few-step denoising setting, and achieves state-of-the-art performance in designing active peptides.

1 INTRODUCTION

Biological sequences: proteins, peptides, and nucleic acids, evolve through a rich set of local edits: single-residue substitutions, deletions, and block insertions of several residues. Among these, insertions and deletions act as a major source of genetic variation and functional innovation (Savino et al., 2022; Topolska et al., 2025; Pál et al., 2006). For antimicrobial peptide design, in particular, mutations affect physicochemical properties such as length, charge, and amphipathicity that govern their activity (Szymczak et al., 2023; Szymczak & Szczurek, 2023; Soares et al., 2026; Szymczak et al., 2025). A central challenge of generative modelling for peptide design is therefore to sample sequences from a realistic distribution that respects the biological constraints and evolutionary distances of natural peptides, in a manner consistent with natural mutational processes (Santos-Júnior et al., 2024; Kmicikiewicz et al., 2025).

Current generative approaches fall short of this goal in complementary ways: (1) autoregressive models handle variable length naturally, but generate left-to-right through a fixed position order, preventing parallel refinement and making controlled local editing difficult (Izdebski et al., 2025); (2) standard discrete diffusion and discrete flow matching models (Shi et al., 2024; Campbell et al., 2024; Gat et al., 2024) support parallel generation, but assume a fixed sequence length and cannot change the number of residues during generation. Recent work has begun to extend flow- and diffusion-based models to variable-length sequences via edit operations (Havasi et al., 2025; Wu et al., 2026; Billera et al., 2026; Kim et al., 2026), but three fundamental limitations persist across these approaches. (i) **existing edit-based and mask-insertion models are restricted to single-token insertions**, whereas biological evolution frequently acts through *block* insertions, entire multi-residue units placed in a single mutational event; (ii) once block insertions are supported, a generative model **must simultaneously decide the edit type, block length, and token content**, which is problematic because token content can only be predicted once block length is resolved; and

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.

(iii) **no current method offers control over how divergent the generated sequence is from its source**: existing methods fix a single source–target distance at training time through their choice of coupling and provide no mechanism to adjust it at inference time, so the divergence of a generated variant cannot be set without training a new model. This limitation is acute in most design-relevant settings. Editing and improving a starting sequence means returning another sequence from the same distribution of plausible sequences, so source and target distributions must coincide. Common couplings cannot meet this condition. Mask and uniform sources draw the source from outside the data distribution, and distortion couplings construct it by corrupting the target, so in each the source is not a plausible sequence to begin with.

We introduce **Mutation Flows**, a discrete flow matching framework over variable-length sequences that resolves all three limitations via block insertions, divergence conditioning, and an exact planner–denoiser factorization that admits any pretrained fixed-length model as a drop-in denoiser. **Our contributions are:**

1. **Mutation Flows.** We cast variable-length sequence generation as a continuous-time Markov chain (CTMC) over all finite sequences on a discrete alphabet. Mutation Flows widens the edit operations used by prior work to include block insertions of arbitrary length, modelled via a zero-truncated Poisson distribution, so that a multi-token motif is reachable in a single transition.
2. **Planner–denoiser decomposition.** We prove that the Mutation Flows objective decomposes exactly into a planner loss and a denoiser loss with disjoint parameters: the planner learns edit types, positions and block lengths. The denoiser edits content. The denoiser loss reduces to a fixed-length discrete flow matching objective, so any pretrained masked sequence model serves directly as the denoiser.
3. **Divergence conditioning.** We treat source–target divergence as an observed conditioning variable, computed from each pair. We pair source and target independently within the *same* distribution, under the unconditional coupling $\mu \otimes \mu$. Sharing a distribution keeps editing well-posed in the regime where optimal transport degenerates, while pairing at random does not constrain the source–target distance, so the pairs span the full empirical distribution of pairwise divergences. We then condition the flow on the *remaining divergence*, which is selectable at inference time.

We show that divergence conditioning and the planner–denoiser architecture each improve over existing couplings and variable-length generation models, and that combining them achieves state-of-the-art performance.

2 Related Work

Non-autoregressive models with length control. Variable-length sequence generation has until recently been a significant challenge for non-autoregressive models. An early edit-based generation model, the Levenshtein Transformer (Gu et al., 2019), introduces iterative refinement through insertion and deletion operations, with a staged decomposition into deletion, placeholder, and token-prediction classifiers. DiffusER (Reid et al., 2022) extends this perspective to discrete diffusion using the four Levenshtein operations with explicit edit-type and edit-length distributions. DDOT (Zhang et al., 2025) proposes to jointly denoise position and denoising state for length adjustment. DreamOn (Wu et al., 2026) addresses variable-length generation by introducing `[expand]` and `[delete]` tokens into the discrete diffusion framework. FlexMDM (Kim et al., 2026) extends masked diffusion to variable length by interleaving single mask-token insertions with unmasking steps, and demonstrates that pretrained masked diffusion models can be adapted to this framework. Edit Flows (Havasi et al., 2025) provides a more theoretically-grounded framework by placing a CTMC over an augmented alignment space, supporting all finite sequences with single-token edit transitions. Mutation Flows differs from these by supporting block insertions of arbitrary length in a single CTMC transition, and by providing a principled model decomposition with an exact algebraic separation of the training objective.

Biological sequence design. Discrete flow matching and diffusion-based approaches have recently been applied to the design and optimization of biological sequences, including proteins (Kmicikiewicz et al., 2025; Alamdari et al., 2024), peptides (Chen et al., 2025), mRNA (Patel et al., 2025), and DNA (Uehara et al., 2025). These methods are, however, restricted to fixed-length generation,

optionally with in-filling. SCISOR (Baron et al., 2025) relaxes this constraint in the deletion-only direction by training a discrete diffusion model to reverse a forward process of random insertions, enabling principled protein shortening but not insertions or substitutions. Mutation Flows is compatible with the same optimization objectives while lifting the length constraint in both directions and supporting all three edit operations natively.

Model decomposition. Recent work has explored decomposing the generation process in discrete flows and diffusion models (Liu et al., 2025; Peng et al., 2026; Israel et al., 2026). Most related to our work is the planner–denoiser decomposition of Liu et al. (2025). Their approach, however, does not support variable-length generation, which requires its own treatment, as block length must be resolved before token content can be predicted.

3 PRELIMINARIES

We briefly review discrete flow matching over fixed-length sequences and its extension to variable-length sequences via edit operations.

3.1 DISCRETE FLOW MATCHING

A continuous-time Markov chain (CTMC) over a discrete state space $\mathcal{X}$ is a stochastic process $(X_t)_{t\in[0,1]}$ whose dynamics are governed by a rate u_t:

$$P(X_{t+h} = x \mid X_t = x_t) = \delta_{x_t}(x) + h\, u_t(x \mid x_t) + o(h), \tag{1}$$

where $u_t(x \mid x_t) \geq 0$ for all $x \neq x_t$ and $\sum_x u_t(x \mid x_t) = 0$. The rate and the probability path p_t it generates are linked by the Kolmogorov forward equation.

Discrete Flow Matching (DFM; Gat et al., 2024; Campbell et al., 2024) uses a CTMC to transport a source distribution μ to a target distribution ν over sequences $\mathcal{X} = \mathcal{T}^N$ of fixed length N and vocabulary $\mathcal{T} = \{1, \ldots, M\}$. Let $\Pi(\mu, \nu)$ denote the set of *couplings* of μ and ν: joint distributions on $\mathcal{X} \times \mathcal{X}$ with the prescribed marginals. Given $\pi \in \Pi(\mu, \nu)$, and a source–target pair (x_0, x_1) from π, DFM defines a *position-wise mixture path*, where each position i independently holds its source value with probability $1 - \kappa_t$ and its target value with probability κ_t, referred to as the noise schedule $p_t(x_t \mid x_0, x_1) = \prod_{i=1}^{N} \Big[(1-\kappa_t)\, \delta_{x_0^i}(x_t^i) + \kappa_t\, \delta_{x_1^i}(x_t^i)\Big]$, where $\kappa_t : [0,1] \to [0,1]$ increases monotonically from 0 to 1. A neural network is trained to approximate the marginal rate via a time-dependent weighted cross-entropy loss. The marginal rate u_t^θ is recovered from the denoiser via the standard marginal-rate formula (Gat et al., 2024), and inference amounts to simulating the learned CTMC from $t = 0$ to $t = 1$.

3.2 EDIT FLOWS

The construction above is typically restricted to a fixed-length space $\mathcal{T}^N$. Havasi et al. (2025) introduce *Edit Flows*, which place the CTMC directly over $\mathcal{X} = \bigcup_{n\geq 0} \mathcal{T}^n$, the space of all finite sequences. Transitions correspond to three elementary edit operations on a sequence $x = (x^1, \ldots, x^{n(x)})$: single-position substitutions, deletions, and single-token insertions. The rate is parameterized as

$$u_t^\theta(\mathrm{ins}(x, i, a) \mid x) = \lambda_{t,i}^{\mathrm{ins}}(x)\, Q_{t,i}^{\mathrm{ins}}(a \mid x), \tag{2}$$

$$u_t^\theta(\mathrm{del}(x, i) \mid x) = \lambda_{t,i}^{\mathrm{del}}(x), \tag{3}$$

$$u_t^\theta(\mathrm{sub}(x, i, a) \mid x) = \lambda_{t,i}^{\mathrm{sub}}(x)\, Q_{t,i}^{\mathrm{sub}}(a \mid x) \tag{4}$$

where a denotes a token, $\lambda_{t,i}^{\mathrm{op}} \geq 0$ is the total rate of operation op at position i and $Q_{t,i}^{\mathrm{ins}}$, $Q_{t,i}^{\mathrm{sub}}$ are normalized distributions over single token values.

Training via an augmented alignment space. Defining the conditional rate directly over $\mathcal{X}$ is intractable, as many distinct edit sequences can transform x_0 into x_1. Edit Flows resolve this by lifting to an *augmented aligned space* $\mathcal{Z} = (\mathcal{T} \cup \{\varepsilon\})^N$, where $\varepsilon \notin \mathcal{T}$ is a gap symbol and $N =$

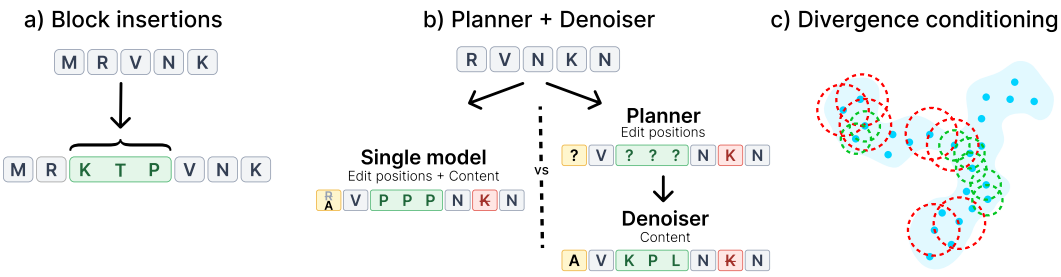


Figure 1: Key components of Mutation Flows. **(a)** Mutation Flows models the number of insertions at each position as a zero-truncated Poisson distribution, which enables variable-length one step insertions. **(b)** Mutation Flows decomposes into a separate *planner* and *denoiser*, allowing different distribution for each insertion. **(c)** Conditioning on the remaining divergence allows coarse-grained control over the similarity between the source sequence and the denoised sequence.

$N(x_0, x_1)$ is the alignment length, now pair-dependent. The *gap-removal* operator $\mathrm{f}_{\mathrm{rm}} : \mathcal{Z} \to \mathcal{X}$ is given by $\mathrm{f}_{\mathrm{rm}}(z) = (z^i : z^i \neq \varepsilon)$, preserving order. An alignment procedure $\mathrm{align} : \mathcal{X} \times \mathcal{X} \to \mathcal{Z} \times \mathcal{Z}$ maps (x_0, x_1) to equal-length (z_0, z_1) with $\mathrm{f}_{\mathrm{rm}}(z_b) = x_b$, lifting any coupling $\pi \in \Pi(\mu, \nu)$ on $\mathcal{X}$ to a coupling on $\mathcal{Z} \times \mathcal{Z}$, also denoted π. Each aligned position (z_0^i, z_1^i) then specifies a unique required edit: an insertion if $z_0^i = \varepsilon$, a deletion if $z_1^i = \varepsilon$, and a substitution if both are non-gap and different. Applying the position-wise mixture path in $\mathcal{Z}$ gives z_t, from which $x_t = \mathrm{f}_{\mathrm{rm}}(z_t)$ is recovered deterministically. Havasi et al. (2025) show that training in $\mathcal{Z}$ yields gradients identical to those of the true marginal objective over $\mathcal{X}$, giving the loss

$$\mathcal{L}_{\mathrm{EF}}(\theta) = \mathbb{E}_{\substack{(z_0,z_1)\sim\pi,\\ t\sim\mathcal{U}(0,1),\\ z_t\sim p_t(\cdot|z_0,z_1)}} \left[\sum_{x \neq x_t} u_t^\theta(x \mid x_t) - \frac{\dot\kappa_t}{1-\kappa_t} \sum_{i=1}^{N} \mathbb{1}_{[z_1^i \neq z_t^i]} \log u_t^\theta\big(x(z_t, i, z_1^i) \mid x_t\big) \right] \tag{5}$$

where κ_t is a noise schedule, $\dot\kappa_t$ is its time derivative, and $x(z_t, i, z_1^i) := \mathrm{f}_{\mathrm{rm}}(z_t[\, i \mapsto z_1^i\,])$ is the sequence obtained by applying the single required edit at aligned position i.

4 Mutation Flows

We first describe how Mutation Flows handles block insertions, then present divergence conditioning for controlling output diversity and similarity to the starting sequence, and finally introduce the planner–denoiser factorization, which enables pretrained fixed-length models to serve as drop-in denoisers. Figure 1 provides an overview.

4.1 Rate Parameterization

Mutation Flows defines a CTMC over $\mathcal{X} = \bigcup_{n\geq 0} \mathcal{T}^n$ whose transitions correspond to three atomic edit operations on a sequence $x = (x^1, \ldots, x^{n(x)})$:

$$\text{Insertion:} \quad \mathrm{ins}(x, i, k, A) = \big(x^1, \ldots, x^i, a_1, \ldots, a_k, x^{i+1}, \ldots, x^{n(x)}\big), \tag{6}$$

$$\text{Deletion:} \quad \mathrm{del}(x, i) = \big(x^1, \ldots, x^{i-1}, x^{i+1}, \ldots, x^{n(x)}\big), \tag{7}$$

$$\text{Substitution:} \quad \mathrm{sub}(x, i, a) = \big(x^1, \ldots, x^{i-1}, a, x^{i+1}, \ldots, x^{n(x)}\big), \tag{8}$$

where $A = (a_1, \ldots, a_k) \in \mathcal{T}^k$ is a block of $k \in \mathbb{Z}_{\geq 1}$ tokens inserted after position i. The block length k is modelled by a zero-truncated Poisson (ZTP) distribution, $\mathrm{ZTP}(k \mid \lambda) = \lambda^k / [(e^\lambda - 1)\, k!]$, which guarantees $k \geq 1$ and is parameterized by a learned rate $\lambda_{t,i}^+$ at each position. The CTMC rates are

$$u_t^\theta\big(\mathrm{ins}(x, i, k, A) \mid x\big) = \lambda_{t,i}^{\mathrm{ins}}\, \mathrm{ZTP}(k \mid \lambda_{t,i}^+)\, Q_{t,i}^{\mathrm{ins}}(A \mid x, k), \tag{9}$$

$$u_t^\theta\big(\mathrm{del}(x, i) \mid x\big) = \lambda_{t,i}^{\mathrm{del}}, \tag{10}$$

$$u_t^\theta\big(\mathrm{sub}(x, i, a) \mid x\big) = \lambda_{t,i}^{\mathrm{sub}}\, Q_{t,i}^{\mathrm{sub}}(a \mid x), \tag{11}$$

where each rate factorizes into a scalar $\lambda^{\text{op}}_{t,i} \geq 0$ governing how often the operation fires and a distribution Q over the resulting tokens. The per-slot factorization of $Q^{\text{ins}}_{t,i}$ is given in Eq. 18 below.

Divergence conditioning. Let $\eta : \mathcal{X} \times \mathcal{X} \to [0, 1]$ be any dissimilarity measure. Neither symmetry nor the triangle inequality is required. A value $d = \eta(x_0, x_1)$ quantifies the divergence between the source sequence x_0 and the target sequence x_1. To control this divergence, we condition the learnt marginal rate u^θ_t on the *remaining divergence* $b_t = (1 - \kappa_t)\, d$, rather than directly on d.

Training requires a coupling π of the source distribution μ and the target distribution ν. We consider the identical marginal setting $\mu = \nu$ and draw x_0 and x_1 independently, giving the unconditional coupling $\pi^u = \mu \otimes \mu$. The divergence $d = \eta(x_0, x_1)$ is computed for each training pair, so training covers the distances induced by independent pairs of data sequences. Proposition 3 shows that the conditional family of flow models $\{u^\theta_t(\cdot \mid x_t, d)\}_{d \in [0,1]}$ obtained this way stratifies a single unconditional flow model.

Biological instantiation. In our experiments η is derived from pairwise sequence alignment: given (x, y) we compute a global alignment score $s(x, y)$ under a substitution matrix (BLOSUM90) with affine gap penalties, and set $\eta(x, y) = 1 - \text{norm}\big(s(x, y)/\ell_{\text{aln}}(x, y)\big) \in [0, 1]$, where $\ell_{\text{aln}}(x, y)$ is the alignment length. The resulting distance manifold and empirical distance distribution are shown in Figure 3, and the alignment settings in Appendix B.

4.2 Training Objective

Following Section 3.2, training proceeds in the augmented aligned space $\mathcal{Z} = (\mathcal{T} \cup \{\varepsilon\})^N$. Edit Flows (and fixed-length discrete diffusion) resolve each aligned position independently. We instead use an *atom-wise* mixture path, in which an atom is one edit operation: every deletion and every substitution position is its own atom, and every maximal run of insertion positions ($z^i_0 = \varepsilon,\ z^i_1 \neq \varepsilon$) is a single atom, so that a block insertion of length k resolves in one step instead of k. Writing z^g for the restriction of z to the positions of an atom g, the path is

$$p_t(z_t \mid z_0, z_1) = \prod_{g \in \mathcal{A}(z_0, z_1)} \Big[(1 - \kappa_t)\, \delta_{z^g_0}(z^g_t) + \kappa_t\, \delta_{z^g_1}(z^g_t)\Big], \tag{12}$$

with $z^i_t = z^i_0 = z^i_1$ at the remaining, identity positions. Each atom resolves as a unit at rate $\dot\kappa_t/(1 - \kappa_t)$, so an insertion block arrives with a well-defined length k for $\text{ZTP}(\lambda^+_{t,i})$ to model.

Let $\mathcal{A}_t = \{g \in \mathcal{A}(z_0, z_1) : z^g_t = z^g_0\}$ be the atoms still unresolved at time t, split by type into $\mathcal{A}^{\text{ins}}_t$, $\mathcal{A}^{\text{del}}_t$ and $\mathcal{A}^{\text{sub}}_t$, and index the rate parameters of Eqs. 9–11 by atom: $\lambda^{\text{op}}_{t,g}$, $\lambda^+_{t,g}$ and $Q^{\text{op}}_{t,g}$ at the position of x_t that g acts on (for an insertion, the position it follows), with k_g its length and $A_g = (a_{g,1}, \ldots, a_{g,k_g})$ its content. For a pair $(z_0, z_1) \sim \pi^{\text{u}}$ with realized distance $d = \eta(x_0, x_1)$, remaining distance $b_t = (1 - \kappa_t)d$ and $x_t = \text{f}_{\text{rm}}(z_t)$, the loss is the Bregman divergence

$$\mathcal{L}(\theta) = \mathbb{E}_{\substack{(z_0,z_1)\sim\pi^{\text{u}},\\ t\sim\mathcal{U}(0,1),\\ z_t\sim p_t(\cdot|z_0,z_1)}} \left[\sum_{x \neq x_t} u^\theta_t(x \mid x_t, d) - \frac{\dot\kappa_t}{1 - \kappa_t}\Big(\mathcal{L}^{\text{ins}} + \mathcal{L}^{\text{del}} + \mathcal{L}^{\text{sub}}\Big)\right], \tag{13}$$

with per-operation targets

$$\mathcal{L}^{\text{ins}} = \sum_{g \in \mathcal{A}^{\text{ins}}_t} \Big[\log\big(\lambda^{\text{ins}}_{t,g}\, \text{ZTP}(k_g \mid \lambda^+_{t,g})\big) + \log Q^{\text{ins}}_{t,g}(A_g \mid x_t, k_g)\Big], \tag{14}$$

$$\mathcal{L}^{\text{del}} = \sum_{g \in \mathcal{A}^{\text{del}}_t} \log \lambda^{\text{del}}_{t,g}, \tag{15}$$

$$\mathcal{L}^{\text{sub}} = \sum_{g \in \mathcal{A}^{\text{sub}}_t} \Big[\log \lambda^{\text{sub}}_{t,g} + \log Q^{\text{sub}}_{t,g}(z^g_1 \mid x_t)\Big]. \tag{16}$$

4.3 PLANNER–DENOISER DECOMPOSITION

The training objective Eq. 13 requires a choice of how to factorize $Q^{\text{ins}}_{t,i}(A \mid x, k)$ over the k inserted tokens. We consider two:

$$\text{single model:} \quad Q^{\text{ins}}_{t,i}(A \mid x, k) = \prod_{j=1}^{k} Q^{\text{ins}}_{t,i,1}(a_j \mid x, i, k), \tag{17}$$

$$\text{planner–denoiser:} \quad Q^{\text{ins}}_{t,i}(A \mid x, k) = \prod_{j=1}^{k} Q^{\text{ins}}_{t,i,j}(a_j \mid x, i, k). \tag{18}$$

The first ties every position to one shared conditional, drawing the whole block i.i.d., and it stays closest to the Edit Flows formulation and is our ablation baseline. The second is strictly more expressive, giving each position its own conditional, which is what allows structured insertions and lets sequence constraints be enforced by pinning selected $Q_{t,i,j}$ to point masses.

Because each $Q^{\text{ins}}_{t,i,j}$ is conditioned on the total block length k, the length must be resolved before any content can be predicted: a single model cannot simultaneously decide the insertion length and predict every token. This structural dependency is the primary motivation for the two-step architecture: a **planner** that first determines the edit operation and block length, followed by a **denoiser** that predicts the token content at each selected position. The following proposition shows that this split is not merely architectural convenience but an exact algebraic property of the objective.

Proposition 1 (Exact decomposition)**.** *Under the factored rates Eqs. 9–11 with position-specific insertion distributions Eq 18, the training objective decomposes exactly as*

$$\mathcal{L}(\theta) = \mathcal{L}_{\text{planner}}(\theta_P) + \mathcal{L}_{\text{denoiser}}(\theta_D),$$

where $\theta_P \cap \theta_D = \varnothing$, so $\nabla_{\theta_P}\mathcal{L}_{\text{denoiser}} = 0$ and $\nabla_{\theta_D}\mathcal{L}_{\text{planner}} = 0$. (Proof in Appendix A.)

Writing $\text{op}(g)$ for the type of atom g, the two losses are

$$\mathcal{L}_{\text{planner}}(\theta_P) = \mathbb{E}_{\substack{(z_0,z_1)\sim\pi^{\text{u}},\\ t\sim\mathcal{U}(0,1),\\ z_t\sim p_t(\cdot|z_0,z_1)}} \left[\sum_i \left(\lambda^{\text{ins}}_{t,i} + \lambda^{\text{del}}_{t,i} + \lambda^{\text{sub}}_{t,i} \right) - \frac{\dot\kappa_t}{1-\kappa_t} \left(\sum_{g\in\mathcal{A}_t} \log \lambda^{\text{op}(g)}_{t,g} + \sum_{g\in\mathcal{A}^{\text{ins}}_t} \log \text{ZTP}(k_g \mid \lambda^{+}_{t,g}) \right) \right], \tag{19}$$

$$\mathcal{L}_{\text{denoiser}}(\theta_D) = -\mathbb{E}_{\substack{(z_0,z_1)\sim\pi^{\text{u}},\\ t\sim\mathcal{U}(0,1),\\ z_t\sim p_t(\cdot|z_0,z_1)}} \left[\frac{\dot\kappa_t}{1-\kappa_t} \sum_{g\in\mathcal{A}^{\text{ins}}_t\cup\mathcal{A}^{\text{sub}}_t} \sum_{j=1}^{k_g} \log Q^{\text{op}(g)}_{t,g,j}(a_{g,j} \mid x_t, k_g) \right], \tag{20}$$

where a substitution atom has $k_g = 1$ and $a_{g,1} = z_1^g$, so the denoiser scores the content of every atom that carries content.

The **planner** (parameters θ_P) decides which edit operation fires at each position and, for insertions, samples the block length from $\text{ZTP}(\lambda^{+}_{t,i})$. The **denoiser** (parameters θ_D) predicts the token content at every position the planner marks for change. Because the two parameter sets are disjoint, they can be trained jointly or sequentially without interference.

Drop-in denoisers. A further consequence of Eq. 18 concerns the denoiser's relationship to existing pretrained models.

Proposition 2 (Denoiser–DFM equivalence)**.** *Construct a masked sequence w_t from z_1 by replacing z_1^i by the mask token $\texttt{m}$ at every position covered by an unresolved insertion or substitution atom, and leaving all other positions unchanged. Set $w_1 = z_1$. Then*

$$\mathcal{L}_{\text{denoiser}}(\theta_D) = -\mathbb{E}_{\substack{(z_0,z_1)\sim\pi^{\text{u}},\\ t\sim\mathcal{U}(0,1),\\ z_t\sim p_t(\cdot|z_0,z_1)}} \left[\frac{\dot\kappa_t}{1-\kappa_t} \sum_{i:\, w_t^i=\texttt{m}} \log p_{\theta_D}(w_1^i \mid w_t) \right],$$

which is precisely the fixed-length discrete flow matching objective (Gat et al., 2024) applied to the masked sequence w_t. (Proof in Appendix A.)

Consequently, any pretrained fixed-length discrete flow matching model over biological sequences serves directly as the denoiser p_{θ_D} without architectural modification. Adapting an existing fixed-length model to variable-length generation therefore requires training only the planner, which contains no token-prediction parameters. This makes Mutation Flows inexpensive to train even at scale.

Sampling. Generation proceeds by Euler discretization of the learned CTMC over N_{iter} steps, conditioned on a target distance d. At each step the planner predicts per-position edit rates and block-length parameters. An edit operation (substitution, insertion, or deletion) is then sampled independently at each position from a categorical distribution over operation types weighted by the respective rates. Positions selected for substitution or insertion receive mask tokens (with insertion blocks of length $k \sim \text{ZTP}(\lambda^+_{t,i})$), and deleted positions are removed. All mask tokens are then resolved in a single forward pass of the denoiser. Full training and sampling procedures are given in Algorithms 1 and 2 respectively. Sampling is visualized in Figure 7.

5 Experiments

We evaluate Mutation Flows on antimicrobial peptides (AMPs) in three settings: the quality of intermediate states along the denoising trajectory (Section 5.1), few-step generation quality (Section 5.2), and guided optimization (Section 5.3). The planner–denoiser factorization and divergence conditioning each improve over their ablations, and together give the strongest model performance in every setting we measure.

Setup. We evaluate three variants of Mutation Flows: The **single model** uses the shared factorization Eq. 17. **Planner + ESM-2** uses the planner–denoiser factorization Eq. 18 with a frozen ESM-2 denoiser (Lin et al., 2023). **Planner + Denoiser** uses the same factorization with both networks trained. We compare against Edit Flows (Havasi et al., 2025) and FlexMDM (Kim et al., 2026). All methods use a Diffusion Transformer backbone with a total of 46M trainable parameters, and share the training schedule and data pipeline. We train on 871,918 AMP sequences from DBAASP (Pirtskhalava et al., 2021), DRAMP4 (Ma et al., 2025), and AMPSphere (Santos-Júnior et al., 2024). The planner–denoiser variant splits the total number of trainable parameters across its two networks, 30% to the planner and 70% to the denoiser, so its two forward passes per sampling step cost together what the single-network methods spend in one. Each model is trained in three configurations, differing in the source–target coupling and whether the model is divergence-conditioned. The distortion coupling **[D]** is standard in edit-based models (Havasi et al., 2025; Aziznejad et al., 2026) and builds x_0 by randomly corrupting x_1. The unconditional coupling **[U]** draws x_0 and x_1 independently from the same data distribution. Divergence conditioning **[C]**, our full method, uses the same unconditional coupling but conditions the model on the remaining divergence $b_t = (1 - \kappa_t)d$. At inference, conditioned models are evaluated at $d \in \{0.10, 0.15, 0.20, 0.30\}$. Figure 4 confirms the requested divergence is honoured at inference time, while [U], trained on identical pairs, does not respond to d. Experimental details are described in Appendix B.

Metrics. We measure distributional fidelity by maximum mean discrepancy (MMD) between ESM-C embeddings (Candido et al., 2026) of generated and reference sequences, and the quality of individual sequences by the probability of antimicrobial activity predicted by OmegAMP classifiers (Soares et al., 2026) and perplexity using Hyformer (Izdebski et al., 2025). For the AMP benchmark, we report the antimicrobial probability and hit-rate from amPEPpy (Lawrence et al., 2021), MIC predicted by Hyformer and APEX (Wan et al., 2024), and the diversity and novelty of the generated set. All metrics are computed with `seqme` (Møller-Larsen et al., 2026) and defined in Appendix B.

5.1 Trajectory quality

Sampling-based guidance scores partially denoised sequences (Chen et al., 2025; Patel et al., 2025; Uehara et al., 2025; Tang et al., 2025), so its scores are only meaningful when the intermediate states are plausible peptides. Because source and target distributions coincide here, both endpoints of the trajectory lie in the data distribution by construction, and the question is whether the path between them does too. We therefore evaluate along the denoising trajectory rather than at the endpoint alone.

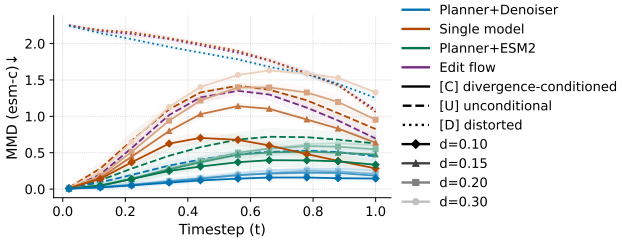


Figure 2: Maximum mean discrepancy (MMD) between generated sequences and reference antimicrobial peptides using ESM-C embeddings for multiple denoising timesteps. Mean and standard deviation are computed over 4 disjoint folds of generated sequences.

Planner + Denoiser with divergence-conditioning stays closest to the data distribution throughout the trajectory (Figure 2). The distortion-coupled configurations never enter the data distribution, starting far outside it (MMD ≈ 2.3) and remaining above every unconditional configuration at $t = 1$. For unconditional coupling, starting with real AMPs, their MMD starts near zero and rises to a mid-trajectory peak. The models differ in how high that peak rises. Planner + ESM-2 peaks well below the single model, which is direct evidence for the drop-in denoiser property of Eq. 18. For both factorized models, divergence-conditioning lowers the peak further at every d, including the largest. In Figure 5, Appendix C.2, we find similar trends, only Planner + Denoiser with conditioning holds a non-negligible predicted probability of antimicrobial activity across the trajectory, supported by Hyformer perplexity. Figure 6 shows the insertion rates the planner predicts along the same trajectory.

Table 1: Maximum mean discrepancy (MMD) to the reference peptides in ESM-C embedding space at matched numbers of forward evaluations (NFE) – lower is better. Mean $\pm$ s.d. over 4 disjoint folds of 500 samples. **Bold** best, underline second best.

	NFE					
Method	2	4	10	20	50	100
FlexMDM [U] (adaptive τ)	9.42 $\pm$0.13	16.98 $\pm$0.10	17.51 $\pm$0.25	17.15 $\pm$0.21	16.81 $\pm$0.10	16.51 $\pm$0.36
FlexMDM [U] (τ-leaping)	9.31 $\pm$0.13	2.09 $\pm$0.07	0.77 $\pm$0.03	0.75 $\pm$0.03	0.83 $\pm$0.04	0.87 $\pm$0.06
FlexMDM [U] (Euler)	6.26 $\pm$0.06	1.33 $\pm$0.03	1.02 $\pm$0.04	1.04 $\pm$0.01	1.01 $\pm$0.08	0.91 $\pm$0.08
Edit Flows [D]	1.66 $\pm$0.02	1.56 $\pm$0.04	1.32 $\pm$0.05	1.14 $\pm$0.09	1.11 $\pm$0.02	1.09 $\pm$0.12
Edit Flows [U]	1.58 $\pm$0.06	1.18 $\pm$0.07	0.88 $\pm$0.05	0.70 $\pm$0.04	0.68 $\pm$0.04	0.60 $\pm$0.10
Single Model [D]	1.73 $\pm$0.03	1.53 $\pm$0.05	1.32 $\pm$0.07	1.25 $\pm$0.06	1.12 $\pm$0.07	1.12 $\pm$0.06
Single Model [U]	1.71 $\pm$0.05	1.40 $\pm$0.12	1.07 $\pm$0.03	0.94 $\pm$0.04	0.80 $\pm$0.04	0.80 $\pm$0.07
Planner + Denoiser [D]	1.55 $\pm$0.08	1.41 $\pm$0.05	1.36 $\pm$0.02	1.30 $\pm$0.06	1.31 $\pm$0.05	1.28 $\pm$0.06
Planner + Denoiser [U]	0.92 $\pm$0.06	0.74 $\pm$0.09	0.52 $\pm$0.06	0.49 $\pm$0.05	0.42 $\pm$0.06	0.42 $\pm$0.06
Planner + Denoiser [C]	**0.69** $\pm$0.03	**0.44** $\pm$0.08	**0.29** $\pm$0.04	**0.27** $\pm$0.04	**0.27** $\pm$0.04	**0.23** $\pm$0.05

5.2 Few-step generation

Generating a sequence requires simulating the learned CTMC over a number of forward passes, and few-step generation reduces that cost by taking fewer, larger steps. Peptide design produces large candidate sets, often inside an optimization loop, so the sampling cost per sequence bounds how much of the design space can be searched. We report the MMD at matched numbers of forward evaluations (NFE) in Table 1, one evaluation being the compute of a single sampling step.

Our full model, Planner + Denoiser Mutation Flows with conditioning, gives the lowest MMD at every NFE budget, between two and three times lower than the best baseline at that budget. It reaches 0.29 at NFE 10, against 0.77 for the best baseline there (FlexMDM [U] under τ-leaping), and 0.27 at NFE 50 against 0.68 (Edit Flows [U]). Already at NFE 4 the full model attains a lower MMD (0.44) than any baseline reaches at NFE 100 (0.60, Edit Flows [U]), a 25-fold reduction in forward passes. Divergence conditioning improves on the unconditional coupling at every budget, down to NFE 2 (0.69 against 0.92). Within every architecture the distortion coupling [D] is the weakest coupling at every NFE, consistent with its intermediate states starting far outside the data distribution.

5.3 Optimization

Finally, we test Mutation Flows capability for AMP design against state-of-the-art methods (described in detail in Appendix B). Intermediate sequences along a Mutation Flows denoising trajectory remain consistent with the learnt distribution of peptides. We use this property to construct

Table 2: Antimicrobial activity and sequence quality of generated peptides (hit-rate: fraction with $p_{\text{amp}} \geq 0.5$; APEX MIC against *E. coli* ATCC11775; novelty w.r.t. the reference peptides). Predictor scores are means $\pm$ one standard error. **Bold** best per block with the lowest variance, underline second best. Mutation Flows (opt) is guided by the Hyformer predictor, so that column is in-objective for it. Blocks are ranked separately, see Appendix B.

	amPEPpy		Hyformer	APEX		
Method	$p_{\text{amp}} \uparrow$	Hit-rate $\uparrow$	$\log_2$ MIC $\downarrow$	MIC (μM) $\downarrow$	Diversity $\uparrow$	Novelty $\uparrow$
Non-optimized						
Mutation Flows	$\mathbf{0.64} \pm 0.01$	$\mathbf{0.88} \pm 0.02$	$\mathbf{0.94} \pm 0.01$	106.9 ± 1.0	0.84	0.99
AMP-Diffusion (Torres et al., 2025)	0.34 ± 0.01	0.27 ± 0.02	0.96 ± 0.02	108.9 ± 1.2	0.82	**1.00**
Muller-LSTM (Müller et al., 2018)	0.63 ± 0.01	0.74 ± 0.02	1.31 ± 0.02	93.5 ± 1.3	0.78	0.96
MMCD (Wang et al., 2024)	0.53 ± 0.01	0.54 ± 0.02	1.29 ± 0.02	109.8 ± 1.0	0.81	**1.00**
AMPGAN-v2 (Van Oort et al., 2021)	0.48 ± 0.01	0.50 ± 0.02	1.09 ± 0.02	96.0 ± 1.3	0.81	**1.00**
MLpeptide Capecchi et al. (2021)	0.62 ± 0.01	0.80 ± 0.02	0.97 ± 0.02	$\mathbf{76.5} \pm 1.3$	0.83	**1.00**
Basic CVAE (Szymczak et al., 2023)	0.46 ± 0.01	0.38 ± 0.02	1.28 ± 0.02	113.0 ± 1.1	**0.87**	**1.00**
OmegAMP-U (Soares et al., 2026)	0.59 ± 0.01	0.65 ± 0.02	1.12 ± 0.02	99.2 ± 1.5	0.86	0.98
Optimized						
Mutation Flows (opt)	0.69 ± 0.01	$\mathbf{0.95} \pm 0.01$	$\mathbf{0.07} \pm 0.01$	80.8 ± 1.1	0.83	**1.00**
HydrAMP (Szymczak et al., 2023)	0.53 ± 0.01	0.57 ± 0.02	1.12 ± 0.02	98.4 ± 1.3	**0.86**	**1.00**
Hyformer (Izdebski et al., 2025)	$\mathbf{0.73} \pm 0.01$	0.93 ± 0.01	0.66 ± 0.02	$\mathbf{56.2} \pm 1.3$	0.81	0.99
OmegAMP-T (Soares et al., 2026)	0.71 ± 0.01	0.87 ± 0.02	0.87 ± 0.02	65.1 ± 1.5	0.80	0.99
OmegAMP-P (Soares et al., 2026)	0.68 ± 0.01	0.86 ± 0.02	0.94 ± 0.02	73.0 ± 1.4	0.82	0.97

an evolutionary optimization algorithm that uses intermediate sequences as proposals in a search guided by predictors, requiring neither retraining nor gradients through the predictors (Algorithm 3). Mutation Flows (opt) uses the Hyformers MIC regressor as the guiding predictor.

Table 2 evaluates sequences generated by the distance conditioned ($d = 0.3$) Planner + Denoiser Mutation Flows. Plain Mutation Flows is compared against seven unconditional (*Non-optimized*) generators, and Mutation Flows (opt) against four methods that receive activity guidance (*Optimized*), either via conditioning or explicit optimization. Non-optimized generation using Mutation Flows is competitive with existing generative models, with the highest amPEPpy hit-rate (0.88) of any non-optimized method and the best Hyformer $\log_2$ MIC (0.94), at diversity 0.84 and novelty 0.99. Optimized generation using Mutation Flows (opt) gives the highest hit-rate across all methods (0.95) at novelty 1.00 and near-unchanged diversity ($0.84 \rightarrow 0.83$). Optimization drives in-objective Hyformer $\log_2$ MIC to the lowest 0.07, and improves APEX MIC from 106.9 to 80.8, showing the gains transfer to a predictor that was not optimized against. A generic guidance objective applied to Mutation Flows is thus already competitive with architectures designed around AMP activity.

6 Conclusions

In this paper, we introduced Mutation Flows, a discrete flow matching framework for variable-length biological sequence generation. By combining block insertions modelled via a zero-truncated Poisson distribution, divergence conditioning over the unconditional coupling for inference-time control of source-to-target similarity, and an exact planner–denoiser decomposition that admits any pretrained fixed-length model as a drop-in denoiser, Mutation Flows resolves three fundamental limitations of existing edit-based generative models. Together, these contributions enable controllable, biologically plausible sequence generation, achieving state-of-the-art sequence quality on an antimicrobial peptide benchmark.

Limitations & Future Work. Mutation Flows and divergence conditioning apply beyond peptides, but we evaluated only one dissimilarity measure, derived from pairwise sequence alignment. Structure-aware or learned metrics are a natural extension. The reported activity gains come from *in silico* predictors, so wet-lab validation is the next step. Because the planner–denoiser decomposition separates edit decisions from the content model, the denoiser need not be discrete: pairing it with a continuous flow matching denoiser, or one generating discrete and continuous content jointly, would let the planner edit sequence length while the denoiser produces residue identities and backbone coordinates, giving variable-length sequence–structure co-generation. Divergence conditioning could also be extended to asymmetric evolutionary settings such as phylogenetic modelling, where source and target are drawn from different distributions and paired by a tree. Broader impact is discussed in Appendix E.

Acknowledgments

AI use statement

In this work, we used generative AI tools to polish the text formulations, assist with the proofs and review the code implementation. We have reviewed all AI-assisted work and take responsibility for the final content of this work, including text, claims and artifacts produced with the aid of generative AI.

Ethics statement

This work introduces a generative model for biological sequence design. We discuss dual-use considerations and biosecurity risk in Appendix E. All datasets used are publicly available under permissive licences (Appendix B) and contain no human-subject or personally identifying data.

Reproducibility statement

Proofs of all propositions are given in Appendix A. The datasets, preprocessing, architecture, and all hyperparameters are specified in Appendix B, and complete training and sampling procedures are given as pseudocode in Appendix D.

References

Sarah Alamdari, Nitya Thakkar, Rianne van den Berg, Neil Tenenholtz, Robert Strome, Alan M. Moses, Alex X. Lu, Nicolò Fusi, Ava P. Amini, and Kevin K. Yang. Protein generation with evolutionary diffusion: sequence is all you need. *bioRxiv*, 2024. doi: 10.1101/2023.09.11.556673. URL https://www.biorxiv.org/content/early/2024/11/04/2023.09.11.556673.

Shayan Aziznejad, Eli Bixby, Nicolas Deutschmann, Constance Ferragu, and Jonathan D. Ziegler. Evoflows: Evolutionary edit-based flow-matching for protein engineering. In *ICLR 2026 Workshop on Foundation Models for Science: Real-World Impact and Science-First Design*, 2026. URL https://openreview.net/forum?id=5vKvqQFGO3.

Arindam Banerjee, Srujana Merugu, Inderjit S. Dhillon, and Joydeep Ghosh. Clustering with Bregman divergences. *Journal of Machine Learning Research*, 6:1705–1749, 2005.

Ethan Baron, Alan N. Amin, Ruben Weitzman, Debora Marks, and Andrew Gordon Wilson. A diffusion model to shrink proteins while maintaining their function, 2025. URL https://arxiv.org/abs/2511.07390.

Lukas Billera, Hedwig Nora Nordlinder, Jack Collier Ryder, Anton Oresten, Aron Stålmarck, Theodor Mosetti Björk, and Ben Murrell. Branching flows: Discrete, continuous, and manifold flow matching with splits and deletions, 2026. URL https://arxiv.org/abs/2511.09465.

Andrew Campbell, Jason Yim, Regina Barzilay, Tom Rainforth, and Tommi Jaakkola. Generative flows on discrete state-spaces: Enabling multimodal flows with applications to protein co-design, 2024. URL https://arxiv.org/abs/2402.04997.

Salvatore Candido, Thomas Hayes, Alexander Derry, Roshan Rao, Zeming Lin, Robert Verkuil, Bryan Z. Wu, Jin Sub Lee, Elise S. Bruguera, Jehan A. Keval, Mykhailo Kopylov, John E. Pak, Wesley Wu, Neil Thomas, Samson Mataraso, Alvin Hsu, Ashton C. Trotman-Grant, Kilian Fatras, Allan dos Santos Costa, Rohil Badkundri, Halil Akın, Deniz Oktay, Jonathan Deaton, Elizabeth Montabana, Hrishita Sitwala, Yue Yu, Marius Wiggert, Dylan Alexander Carlin, Anthony W. Goering, Tomasz Blazejewski, McCullen Sandora, Michael Hla, Tina Z. Jia, Leon H. Kloker, Nicholas J. Sofroniew, Masatoshi Uehara, Jassi Pannu, Sharrol Bachas, Daniel S. Liu, Tom Sercu, and Alexander Rives. Language modeling materializes a world model of protein biology. *bioRxiv*, 2026. doi: 10.64898/2026.06.03.729735. URL https://www.biorxiv.org/content/early/2026/06/04/2026.06.03.729735.

Alice Capecchi, Xingguang Cai, Hippolyte Personne, Thilo Köhler, Christian van Delden, and Jean-Louis Reymond. Machine learning designs non-hemolytic antimicrobial peptides. *Chemical Science*, 12(26):9221–9232, 07 2021. ISSN 2041-6520. doi: 10.1039/d1sc01713f. URL https://doi.org/10.1039/d1sc01713f.

Tong Chen, Yinuo Zhang, Sophia Tang, and Pranam Chatterjee. Multi-objective-guided discrete flow matching for controllable biological sequence design, 2025. URL https://arxiv.org/abs/2505.07086.

Jeff Daily. Parasail: SIMD C library for global, semi-global, and local pairwise sequence alignments. *BMC Bioinformatics*, 17:81, 2016.

Itai Gat, Tal Remez, Neta Shaul, Felix Kreuk, Ricky TQ Chen, Gabriel Synnaeve, Yossi Adi, and Yaron Lipman. Discrete flow matching. *Advances in Neural Information Processing Systems*, 37: 133345–133385, 2024.

Jiatao Gu, Changhan Wang, and Junbo Zhao. Levenshtein transformer. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/675f9820626f5bc0afb47b57890b466e-Paper.pdf.

Marton Havasi, Brian Karrer, Itai Gat, and Ricky T. Q. Chen. Edit flows: Flow matching with edit operations, 2025. URL https://arxiv.org/abs/2506.09018.

Daniel Israel, Tian Jin, Ellie Cheng, Guy Van den Broeck, Aditya Grover, Suvinay Subramanian, and Michael Carbin. Planned diffusion. In C. Vondrick, B. Hariharan, C. Raffel, L. Pinto, D. Yang, and A. Faust (eds.), *International Conference on Learning Representations*, volume 2026, pp. 98298–98318, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/9fbae6362539ef7a151303c86cb51c00-Paper-Conference.pdf.

Adam Izdebski, Jan Olszewski, Pankhil Gawade, Krzysztof Koras, Serra Korkmaz, Valentin Rauscher, Jakub M. Tomczak, and Ewa Szczurek. Synergistic benefits of joint molecule generation and property prediction, 2025. URL https://arxiv.org/abs/2504.16559.

Jaeyeon Kim, Lee Kit, Carles Domingo i Enrich, Yilun Du, Sham Kakade, Timothy Ngotiaoco, Sitan Chen, and Michael Albergo. Any-order flexible length masked diffusion. In *International Conference on Learning Representations*, volume 2026, pp. 40151–40183, 2026.

Michal Kmicikiewicz, Vincent Fortuin, and Ewa Szczurek. Prospero: Active learning for robust protein design beyond wild-type neighborhoods. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 131015–131048. Curran Associates, Inc., 2025. doi: 10.52202/085713-4364. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/bde1c75d7d89e64d6d93dce1c68dc3d8-Paper-Conference.pdf.

Travis J. Lawrence, Dana L. Carper, Maryam K. Spangler, Alyssa A. Carrell, Tomas A. Rush, Stephen J. Minter, David J. Weston, and Jesse L. Labbé. amPEPpy 1.0: a portable and accurate antimicrobial peptide prediction tool. *Bioinformatics*, 37(14):2058–2060, 2021.

Zeming Lin, Halil Akin, Roshan Rao, Brian Hie, Zhongkai Zhu, Wenting Lu, Nikita Smetanin, Robert Verkuil, Ori Kabeli, Yaniv Shmueli, Allan dos Santos Costa, Maryam Fazel-Zarandi, Tom Sercu, Salvatore Candido, and Alexander Rives. Evolutionary-scale prediction of atomic-level protein structure with a language model. *Science*, 379(6637):1123–1130, 2023.

Sulin Liu, Juno Nam, Andrew Campbell, Hannes Stärk, Yilun Xu, Tommi Jaakkola, and Rafael Gomez-Bombarelli. Think while you generate: Discrete diffusion with planned denoising. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 81979–82026, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/cbf883a744952d4a40591271a58ab9d0-Paper-Conference.pdf.

Tianyue Ma, Yanchao Liu, Bingxin Yu, Xin Sun, Huiyuan Yao, Chen Hao, Jianhui Li, Maryam Nawaz, Xun Jiang, Xingzhen Lao, and Heng Zheng. Dramp 4.0: an open-access data repository dedicated to the clinical translation of antimicrobial peptides. *Nucleic Acids Research*, 53(D1): D403–D410, 01 2025. ISSN 1362-4962. doi: 10.1093/nar/gkae1046. URL https://doi.org/10.1093/nar/gkae1046.

Alex T. Müller, Jan A. Hiss, and Gisbert Schneider. Recurrent neural network model for constructive peptide design. *Journal of Chemical Information and Modeling*, 58(2):472–479, 02 2018. ISSN 1549-9596. doi: 10.1021/acs.jcim.7b00414. URL https://doi.org/10.1021/acs.jcim.7b00414.

Rasmus Møller-Larsen, Adam Izdebski, Jan Olszewski, Pankhil Gawade, Michal Kmicikiewicz, Wojciech Zarzecki, and Ewa Szczurek. seqme: a python library for evaluating biological sequence design from generative models. *Bioinformatics Advances*, 6(1):vbag212, 01 2026. ISSN 2635-0041. doi: 10.1093/bioadv/vbag212. URL https://doi.org/10.1093/bioadv/vbag212.

Csaba Pál, Balázs Papp, and Martin J. Lercher. An integrated view of protein evolution. *Nature Reviews Genetics*, 7(5):337–348, May 2006. ISSN 1471-0064. doi: 10.1038/nrg1838. URL https://doi.org/10.1038/nrg1838.

Sawan Patel, Sophia Tang, Yinuo Zhang, Pranam Chatterjee, and Sherwood Yao. Multi-objective-guided generative design of mRNA with therapeutic properties. In *ICML 2025 Workshop on Scaling Up Intervention Models*, 2025. URL https://openreview.net/forum?id=0kg9KhUYSK.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 4172–4182. IEEE, 2023.

Fred Zhangzhi Peng, Zachary Bezemek, Sawan Patel, Jarrid Rector-Brooks, Sherwood Yao, Avishek Joey Bose, Alexander Tong, and Pranam Chatterjee. Path planning for masked diffusion model sampling, 2026. URL https://arxiv.org/abs/2502.03540.

Malak Pirtskhalava, Anthony A Amstrong, Maia Grigolava, Mindia Chubinidze, Evgenia Alimbarashvili, Boris Vishnepolsky, Andrei Gabrielian, Alex Rosenthal, Darrell E Hurt, and Michael Tartakovsky. Dbaasp v3: database of antimicrobial/cytotoxic activity and structure of peptides as a resource for development of new therapeutics. *Nucleic Acids Research*, 49(D1):D288–D297, 01 2021. ISSN 0305-1048. doi: 10.1093/nar/gkaa991. URL https://doi.org/10.1093/nar/gkaa991.

Machel Reid, Vincent J. Hellendoorn, and Graham Neubig. Diffuser: Discrete diffusion via edit-based reconstruction, 2022. URL https://arxiv.org/abs/2210.16886.

Célio Dias Santos-Júnior, Marcelo D.T. Torres, Yiqian Duan, Álvaro Rodríguez del Río, Thomas S.B. Schmidt, Hui Chong, Anthony Fullam, Michael Kuhn, Chengkai Zhu, Amy Houseman, Jelena Somborski, Anna Vines, Xing-Ming Zhao, Peer Bork, Jaime Huerta-Cepas, Cesar de la Fuente-Nunez, and Luis Pedro Coelho. Discovery of antimicrobial peptides in the global microbiome with machine learning. *Cell*, 187(14):3761–3778.e16, 2024. ISSN 0092-8674. doi: https://doi.org/10.1016/j.cell.2024.05.013. URL https://www.sciencedirect.com/science/article/pii/S0092867424005221.

Simone Savino, Tom Desmet, and Jorick Franceus. Insertions and deletions in protein evolution and engineering. *Biotechnology Advances*, 60:108010, 2022. ISSN 0734-9750. doi: https://doi.org/10.1016/j.biotechadv.2022.108010. URL https://www.sciencedirect.com/science/article/pii/S0734975022001069.

Jiaxin Shi, Kehang Han, Zhe Wang, Arnaud Doucet, and Michalis Titsias. Simplified and generalized masked diffusion for discrete data. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 103131–103167. Curran Associates, Inc., 2024. doi: 10.52202/079017-3277. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/bad233b9849f019aead5e5cc60cef70f-Paper-Conference.pdf.

Diogo Soares, Leon Hetzel, Paulina Szymczak, Marcelo Der Torossian Torres, Johanna Sommer, Cesar de la Fuente-Nunez, Fabian J Theis, Stephan Günnemann, and Ewa Szczurek. OmegAMP: Targeted AMP discovery via biologically informed generation. *Transactions on Machine Learning Research*, 2026. ISSN 2835-8856. URL https://openreview.net/forum?id=hAq3XLZ9ex.

Paulina Szymczak and Ewa Szczurek. Artificial intelligence-driven antimicrobial peptide discovery. *Current Opinion in Structural Biology*, 83:102733, 2023. ISSN 0959-440X. doi: https://doi.org/10.1016/j.sbi.2023.102733. URL https://www.sciencedirect.com/science/article/pii/S0959440X23002075.

Paulina Szymczak, Marcin Możejko, Tomasz Grzegorzek, Radosław Jurczak, Marta Bauer, Damian Neubauer, Karol Sikora, Michał Michalski, Jacek Sroka, Piotr Setny, Wojciech Kamysz, and Ewa Szczurek. Discovering highly potent antimicrobial peptides with deep generative model HydrAMP. *Nature Communications*, 14(1), March 2023. doi: 10.1038/s41467-023-36994-z. URL https://doi.org/10.1038/s41467-023-36994-z.

Paulina Szymczak, Wojciech Zarzecki, Jiejing Wang, Yiqian Duan, Jun Wang, Luis Pedro Coelho, Cesar de la Fuente-Nunez, and Ewa Szczurek. Ai-driven antimicrobial peptide discovery: Mining and generation. *Accounts of Chemical Research*, 58(12):1831–1846, 06 2025. ISSN 0001-4842. doi: 10.1021/acs.accounts.0c00594. URL https://doi.org/10.1021/acs.accounts.0c00594.

Sophia Tang, Yinuo Zhang, Alexander Tong, and Pranam Chatterjee. Gumbel-softmax flow matching with straight-through guidance for controllable biological sequence generation, 2025. URL https://arxiv.org/abs/2503.17361.

Magdalena Topolska, Antoni Beltran, and Ben Lehner. Deep indel mutagenesis reveals the impact of amino acid insertions and deletions on protein stability and function. *Nature Communications*, 16(1):2617, Mar 2025. ISSN 2041-1723. doi: 10.1038/s41467-025-57510-5. URL https://doi.org/10.1038/s41467-025-57510-5.

Marcelo D.T. Torres, Leo Tianlai Chen, Fangping Wan, Pranam Chatterjee, and Cesar de la Fuente-Nunez. Generative latent diffusion language modeling yields anti-infective synthetic peptides. *Cell Biomaterials*, 1(9), Oct 2025. ISSN 3050-5623. doi: 10.1016/j.celbio.2025.100183. URL https://doi.org/10.1016/j.celbio.2025.100183.

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, 2025. URL https://arxiv.org/abs/2502.14944.

Colin M Van Oort, Jonathon B Ferrell, Jacob M Remington, Safwan Wshah, and Jianing Li. AMPGAN v2: Machine learning-guided design of antimicrobial peptides. *Journal of chemical information and modeling*, 61(5):2198–2207, 2021. ISSN 1549-960X. doi: 10.1021/acs.jcim.0c01441.

Fangping Wan, Marcelo D. T. Torres, Jacqueline Peng, and Cesar de la Fuente-Nunez. Deep-learning-enabled antibiotic discovery through molecular de-extinction. *Nature Biomedical Engineering*, 8:854–871, 2024.

Yongkang Wang, Xuan Liu, Feng Huang, Zhankun Xiong, and Wen Zhang. A multi-modal contrastive diffusion model for therapeutic peptide generation, 2024. URL https://arxiv.org/abs/2312.15665.

Zirui Wu, Lin Zheng, Zhihui Xie, Jiacheng Ye, Jiahui Gao, Shansan Gong, Yansong Feng, Zhenguo Li, Wei BI, Guorui Zhou, and Lingpeng Kong. Dreamon: Diffusion language models for code infilling beyond fixed-size canvas. In C. Vondrick, B. Hariharan, C. Raffel, L. Pinto, D. Yang, and A. Faust (eds.), *International Conference on Learning Representations*, volume 2026, pp. 76626–76643, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/7c020dd308d7413b5078686723b2f857-Paper-Conference.pdf.

Andrew Zhang, Anushka Sivakumar, Chia-Wei Tang, and Chris Thomas. Flexible-length text infilling for discrete diffusion models. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng (eds.), *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 31344–31359, Suzhou, China, November 2025. Association for Computational Linguistics. ISBN 979-8-89176-332-6. doi: 10.18653/v1/2025.emnlp-main.1597. URL https://aclanthology.org/2025.emnlp-main.1597/.

A PROOFS

A.1 DISTANCE STRATIFICATION

Divergence conditioning recovers the distance-stratified marginal rates, and these strata mix back to the unconditional edit flow.

Proposition 3 (Distance stratification). *Let $(x_0, x_1) \sim \pi^{\mathrm{u}}$, $d = \eta(x_0, x_1)$, and let $x_t = \mathrm{f}_{\mathrm{rm}}(z_t)$ with z_t drawn from the atom-wise mixture path Eq. 12. Write $u_t(y \mid x_t, x_0, x_1)$ for the conditional rate of the edit flow given the endpoint pair. Define the stratified and unconditional marginal rates*

$$u_t^\star(y \mid x_t, d) = \mathbb{E}\big[\, u_t(y \mid x_t, x_0, x_1) \;\big|\; x_t,\, \eta(x_0, x_1) = d \,\big],$$
$$u_t^\star(y \mid x_t) = \mathbb{E}\big[\, u_t(y \mid x_t, x_0, x_1) \;\big|\; x_t \,\big].$$

Then (i) the divergence-conditioned objective Eq. 13 is minimized over θ at $u_t^\theta(\cdot \mid x_t, d) = u_t^\star(\cdot \mid x_t, d)$ for F-almost every d and p_t-almost every x_t; and (ii) for every $y \neq x_t$,

$$u_t^\star(y \mid x_t) = \mathbb{E}_{d \sim p_t(d \mid x_t)}\big[\, u_t^\star(y \mid x_t, d) \,\big],$$

where $p_t(d \mid x_t)$ is the posterior over the pair distance given the current state.

Proof. **(i)** The objective Eq. 13 is, for each $y \neq x_t$, a Bregman divergence between the predicted rate $u_t^\theta(y \mid x_t, d)$ and the conditional rate $u_t(y \mid x_t, x_0, x_1)$, generated by the convex function $\phi(a) = a \log a - a$ on $a > 0$. Bregman divergences are minimized in expectation by the conditional mean of the target given the predictor's arguments (Banerjee et al., 2005): for any σ-algebra $\mathcal{F}$ with respect to which the prediction is measurable,

$$\arg\min_{v \,\in\, \mathcal{F}} \mathbb{E}\big[\, D_\phi\big(u_t(y \mid x_t, x_0, x_1),\, v\big) \,\big] = \mathbb{E}\big[\, u_t(y \mid x_t, x_0, x_1) \mid \mathcal{F} \,\big].$$

The remaining divergence $b_t = (1 - \kappa_t)d$ is, at fixed t, an invertible function of d, so conditioning the network on (x_t, κ_t, b_t) is conditioning on $\mathcal{F}_t = \sigma(x_t, d)$. Substituting $\mathcal{F} = \mathcal{F}_t$ gives exactly $u_t^\star(\cdot \mid x_t, d)$. Taking $\mathcal{F} = \sigma(x_t)$ instead, which is the unconditional model, gives $u_t^\star(\cdot \mid x_t)$.

(ii) Since $\sigma(x_t) \subseteq \mathcal{F}_t$, the tower property of conditional expectation gives

$$\begin{aligned} u_t^\star(y \mid x_t) &= \mathbb{E}\big[\, u_t(y \mid x_t, x_0, x_1) \mid x_t \,\big] = \mathbb{E}\Big[\, \mathbb{E}\big[u_t(y \mid x_t, x_0, x_1) \mid x_t, d\big] \;\Big|\; x_t \Big] \\ &= \mathbb{E}_{d \sim p_t(d \mid x_t)}\big[\, u_t^\star(y \mid x_t, d) \,\big]. \end{aligned}$$

□

Remark 1. *Part (ii) is stated at the level of rates, which is where it holds exactly. It is the posterior $p_t(d \mid x_t)$, not the prior F, that appears: the current state is informative about how far the pair was to begin with. Because the marginal rates generate the marginal probability path, a model trained with divergence conditioning and then run with d drawn from $p_t(\cdot \mid x_t)$ reproduces the unconditional edit flow. Running it at a fixed d instead selects one stratum.*

A.2 PLANNER–DENOISER DECOMPOSITION

We prove Propositions 1 and 2: that the training objective Eq. 13 decomposes exactly into a planner loss and a denoiser loss with disjoint gradients, and that the denoiser loss is equivalent to a fixed-length discrete flow matching objective.

Setup and notation. Fix a sample (z_0, z_1, t, z_t) with $x_t = \mathrm{f}_{\mathrm{rm}}(z_t)$. Partition the aligned positions $\{1, \ldots, N\}$ into four disjoint index sets according to (z_t^i, z_1^i):

$$\mathcal{I}^{\mathrm{ins}} = \{i : z_t^i = \varepsilon,\ z_1^i \neq \varepsilon\}, \qquad \mathcal{I}^{\mathrm{del}} = \{i : z_t^i \neq \varepsilon,\ z_1^i = \varepsilon\},$$
$$\mathcal{I}^{\mathrm{sub}} = \{i : z_t^i \neq \varepsilon,\ z_1^i \neq \varepsilon,\ z_t^i \neq z_1^i\}, \qquad \mathcal{I}^{\mathrm{id}} = \{i : z_t^i = z_1^i\}.$$

By the atom-wise path Eq. 12, the first three sets are exactly the positions of the unresolved atoms of Section 4: $\mathcal{I}^{\mathrm{del}}$ and $\mathcal{I}^{\mathrm{sub}}$ hold the positions of the atoms in $\mathcal{A}_t^{\mathrm{del}}$ and $\mathcal{A}_t^{\mathrm{sub}}$, and $\mathcal{I}^{\mathrm{ins}}$ is the disjoint union of the insertion atoms $g \in \mathcal{A}_t^{\mathrm{ins}}$, while $\mathcal{I}^{\mathrm{id}}$ holds the positions with nothing outstanding: identity positions and those of already resolved atoms. In particular each insertion block is resolved as a unit, so no block is split across $\mathcal{I}^{\mathrm{ins}}$ and $\mathcal{I}^{\mathrm{id}}$, and k_g is the full target block length, making $\mathrm{ZTP}(k_g \mid \lambda_{t,g}^+)$ well defined. Let θ_P collect all λ and λ^+ parameters, and θ_D collect all $Q_{t,g,j}^{\mathrm{ins}}$ and $Q_{t,g}^{\mathrm{sub}}$ parameters, and by construction $\theta_P \cap \theta_D = \varnothing$.

Proposition 1 (Exact decomposition). *Under the factored rates Eqs. 9–11 with position-specific insertion distributions Eq 18, the training objective decomposes exactly as*

$$\mathcal{L}(\theta) = \mathcal{L}_{\text{planner}}(\theta_P) + \mathcal{L}_{\text{denoiser}}(\theta_D),$$

where $\theta_P \cap \theta_D = \varnothing$, *so* $\nabla_{\theta_P}\mathcal{L}_{\text{denoiser}} = 0$ *and* $\nabla_{\theta_D}\mathcal{L}_{\text{planner}} = 0$.

Proof. The loss Eq. 13 has two parts, which we handle in turn.

Part 1: Total-rate penalty. Because $\text{ZTP}(\cdot \mid \lambda^+_{t,i})$ and each Q distribution are normalized, summing the rates $u^\theta_t(x \mid x_t)$ over all $x \neq x_t$ gives

$$\sum_{x \neq x_t} u^\theta_t(x \mid x_t, d) = \sum_i \Big(\lambda^{\text{ins}}_{t,i} + \lambda^{\text{del}}_{t,i} + \lambda^{\text{sub}}_{t,i}\Big), \tag{21}$$

which depends only on θ_P.

Part 2: Per-edit cross-entropy. For an unresolved atom $g \in \mathcal{A}_t$, write $x(z_t, g) = \text{f}_{\text{rm}}(z_t[\, g \mapsto z^g_1\,])$ for the sequence obtained by applying the single edit that resolves it. We expand $\log u^\theta_t(x(z_t, g) \mid x_t)$ using Eqs. 9–11, by the type of g.

Deletion ($g \in \mathcal{A}^{\text{del}}_t$).

$$\log u^\theta_t(\text{del}(x_t, g) \mid x_t) = \log \lambda^{\text{del}}_{t,g},$$

which involves θ_P only.

Substitution ($g \in \mathcal{A}^{\text{sub}}_t$).

$$\log u^\theta_t(\text{sub}(x_t, g, z^g_1) \mid x_t) = \log \lambda^{\text{sub}}_{t,g} + \log Q^{\text{sub}}_{t,g}(z^g_1 \mid x_t),$$

an additive split between θ_P and θ_D.

Insertion block ($g \in \mathcal{A}^{\text{ins}}_t$, block length k_g).

$$\log u^\theta_t\big(\text{ins}(x_t, g, k_g, A_g) \mid x_t\big) = \log \lambda^{\text{ins}}_{t,g} + \log \text{ZTP}(k_g \mid \lambda^+_{t,g}) + \sum_{j=1}^{k_g} \log Q^{\text{ins}}_{t,g,j}(a_{g,j} \mid x_t, k_g), \tag{22}$$

where the first two terms involve only θ_P and the sum involves only θ_D.

In every case θ_P and θ_D appear in strictly additive terms. Collecting all θ_P terms into $\mathcal{L}_{\text{planner}}$ and all θ_D terms into $\mathcal{L}_{\text{denoiser}}$ yields the decomposition stated in Proposition 1, with explicit forms given in Eqs. 19–20. Since the two parameter sets are disjoint, the cross-gradients vanish identically. □

Proposition 4 (Denoiser–DFM equivalence). *Construct a masked sequence w_t from z_1 as follows: for every position $i \in \mathcal{I}^{\text{ins}} \cup \mathcal{I}^{\text{sub}}$, replace z^i_1 by the mask token $\mathtt{m}$, and leave all other positions unchanged. Set $w_1 = z_1$ (the unmasked target). Then*

$$\mathcal{L}_{\text{denoiser}}(\theta_D) = -\mathbb{E}_{\substack{(z_0, z_1) \sim \pi^{\text{u}}, \\ t \sim \mathcal{U}(0,1), \\ z_t \sim p_t(\cdot \mid z_0, z_1)}} \left[\frac{\dot\kappa_t}{1 - \kappa_t} \sum_{i:\, w^i_t = \mathtt{m}} \log p_{\theta_D}(w^i_1 \mid w_t) \right], \tag{23}$$

which is precisely the fixed-length discrete flow matching objective (Gat et al., 2024) applied to the masked sequence w_t.

Proof. We exhibit a bijection between the summands in $\mathcal{L}_{\text{denoiser}}$ and the masked positions in w_t.

Masked positions. By construction, $w^i_t = \mathtt{m}$ if and only if $i \in \mathcal{I}^{\text{ins}} \cup \mathcal{I}^{\text{sub}}$.

Substitution positions ($i \in \mathcal{I}^{\text{sub}}$, the position of an atom $g \in \mathcal{A}^{\text{sub}}_t$): the denoiser contributes $\log Q^{\text{sub}}_{t,g}(z^i_1 \mid x_t)$. At position i in w_t the mask token appears, the target is $w^i_1 = z^i_1$, and $Q^{\text{sub}}_{t,g}(\cdot \mid x_t) = p_{\theta_D}(\cdot \mid w_t)$ by definition of the denoiser. Hence this term equals $\log p_{\theta_D}(w^i_1 \mid w_t)$.

Insertion positions (slot j of an atom $g \in \mathcal{A}^{\text{ins}}_t$ starting at position i): the denoiser contributes $\log Q^{\text{ins}}_{t,g,j}(a_{g,j} \mid x_t, k_g)$. After the block length k_g is determined by the alignment, each slot j in

w_t carries a mask token whose target is $w_1^{i+j-1} = a_{g,j}$, and the per-slot model outputs $p_{\theta_D}(\cdot \mid w_t)$. Hence each slot term equals $\log p_{\theta_D}(w_1^{i+j-1} \mid w_t)$.

Identity positions ($i \in \mathcal{I}^{\mathrm{id}}$): no mask is placed and no denoiser term is generated, so these positions contribute zero to both sides of Eq. 23.

Combining the three cases, multiplying by $-\dot{\kappa}_t/(1-\kappa_t)$ and taking the expectation yields Eq. 23, which is the DFM loss (Gat et al., 2024) applied to w_t. □

Remark 2. *Eq. 23 shows that the denoiser objective is identical to the fixed-length discrete flow matching loss (Gat et al., 2024) applied to the masked sequence w_t. Consequently, any pretrained fixed-length discrete flow matching model over biological sequences can serve directly as the denoiser p_{θ_D} without architectural modification. The planner then becomes a lightweight head on top of the (optionally frozen) pretrained representations, and the two components can be trained jointly or sequentially because $\theta_P \cap \theta_D = \varnothing$.*

Remark 3 (Conditioning and drop-in denoisers)**.** *Our trained denoiser additionally receives the remaining divergence b_t, so that $Q^{\mathrm{ins}}_{t,g,j}$ and $Q^{\mathrm{sub}}_{t,g}$ are conditioned on (x_t, κ_t, b_t). Proposition 4 is unaffected: the remaining divergence enters only as an extra input to p_{θ_D}, and the identity holds pointwise in whatever the denoiser conditions on. A pretrained model that never receives b_t is therefore still a valid denoiser, simply one whose token predictions do not vary with the divergence, and all divergence-dependence then resides in the planner, which alone determines* which *and* how many *positions change. This is the regime of our Planner + ESM-2 variant.*

B Experimental Details

Data. All experiments use a dataset of 873,665 antimicrobial peptide sequences aggregated from three public databases: DBAASP (Pirtskhalava et al., 2021) (Public Domain), DRAMP4 (Ma et al., 2025) (CC BY 4.0), and AMPSphere (Santos-Júnior et al., 2024) (CC BY 4.0). We retain only sequences of length between 10 and 50 residues. The dataset is split into training and validation sets using a $99.8 / 0.2$ ratio with a fixed seed, giving 871,918 training and 1,747 validation sequences. Sequences are tokenized over the vocabulary of the 20 canonical amino acids, together with four special tokens (padding, beginning-of-sequence, gap ε, and mask).

Source–target pairing. We compare three pairing schemes. The first two both draw x_0 and x_1 independently from the training distribution and differ in whether the realized distance is exposed to the model.

- **Unconditional coupling with divergence conditioning [C] (ours).** x_0 and x_1 are drawn independently from the training distribution and paired at random, so the coupling is $\mu \otimes \mu$ and imposes no distance constraint. Concretely, each training batch of B sequences is turned into $B - 1$ pairs by pairing consecutive entries, and batches are reshuffled every epoch, so the pairing is redrawn throughout training. For each pair we compute the realized dissimilarity $\eta(x_0, x_1)$ from a global Needleman–Wunsch alignment (`parasail` (Daily, 2016)) under the BLOSUM90 substitution matrix with gap open and extension penalties of 12. To avoid low-density regions at the extremes of the distance distribution, we clip the length-normalized alignment score to the range $[-8.3, -0.1]$ as illustrated in Figure 3b, and rescale to $[0, 1]$ as described in Section C.1. The model is conditioned on the remaining divergence $b_t = (1 - \kappa_t)\, \eta(x_0, x_1)$.
- **Unconditional coupling without divergence conditioning [U].** The same independent random pairing, but the realized distance is discarded and the model receives a constant divergence $d = 1$. This is the d-marginalized member of the conditional family (Proposition 3), and isolates the contribution of the conditioning signal from that of the pairing.
- **Distortion coupling [D].** The standard pairing in edit-based flow models (Havasi et al., 2025; Aziznejad et al., 2026). Source sequences x_0 are constructed from target sequences x_1 by applying random edits directly in the aligned space $\mathcal{Z}$: the aligned length is set to $n(x_1)/(1 - p_{\text{del}})$, and a fraction $p_{\text{sub}} = 0.5$ of the aligned positions is marked as substitutions, $p_{\text{ins}} = 0.25$ as insertions and $p_{\text{del}} = 0.25$ as deletions. Positions are drawn uniformly without replacement, so the realized fractions are fixed rather than binomial, and the source tokens at substituted and deleted positions are drawn uniformly from the 20 amino acids. Unlike the unconditional coupling this is a coupling of two *distinct* distributions, and it fixes an implicit pairwise distance that cannot be controlled at inference time. As with the arm above, the model receives a constant divergence $d = 1$.

Alignment. The two unconditional arms lift (x_0, x_1) into the augmented space $\mathcal{Z}$ by different means: the conditioned arm reuses the Needleman–Wunsch alignment already computed for η, whereas the arm without conditioning pads the shorter sequence with gap symbols. The two arms therefore differ in the alignment as well as in the conditioning signal, and the marginal-quality comparison between them in Figure 2 should be read with that in mind. The controllability comparison in Figure 4 is unaffected, since no choice of alignment can make a model respond to a conditioning variable it was never given.

Architecture. All models use a Diffusion Transformer backbone (Peebles & Xie, 2023), with blocks that are pre-norm with rotary position embeddings, RMSNorm and a SwiGLU feedforward. The single model, Edit Flows and FlexMDM each use one such network with embedding dimensionality 512, feedforward hidden dimensionality 2,048, 8 attention heads, and 8 layers, giving 46.6M parameters. Conditioning enters through the remaining divergence alone: the scalar $b_t = (1 - \kappa_t)\, d$ is mapped to a 512-dimensional sinusoidal embedding and consumed by AdaLN-Zero modulation (scale, shift and gate per block), produced by a single projection shared across blocks. The timestep therefore reaches the network only through b_t. For the two unconditional arms, where $d = 1$, this reduces to a function of t alone. A linear noise schedule $\kappa_t = t$ is used throughout.

The planner–denoiser variant divides the same budget across its two networks, 13.7M to the planner and 32.8M to the denoiser, or 29.4% and 70.4% of the single model. The two together come to 46.5M, within 0.2% of the single model's 46.6M. Both keep the depth of 8 layers and are narrowed instead: the planner uses embedding dimensionality 240, feedforward hidden dimensionality 960 and 5 attention heads, the denoiser 416, 1,664 and 13. Head dimensionalities remain multiples of 8 so that scaled dot-product attention keeps its fused kernel. Every method is therefore matched both in total parameter count and in the compute of one sampling step, since the variant's two forward passes per step cost together what a single-network method spends in one pass, and the NFE axis of Table 1 counts sampling steps uniformly across rows. We summarize all hyperparameters in Table 3. This matched budget applies to the comparisons in Sections 5.1 and 5.2, where every method is one we train ourselves. The AMP benchmark of Table 2 compares against sequences released by their authors and so carries no matched-parameter constraint, and there we use a larger planner–denoiser model with 64M parameters in each of the two networks.

Training. All models are trained with the same optimizer, schedule and data pipeline (Table 3) for 800 epochs, which at a global batch size of 512 is approximately 1.36M optimizer steps. Weights are averaged with an exponential moving average of decay 0.9999, and the averaged weights are used for all evaluation. Training runs in bf16 mixed precision.

Baselines.

- **Edit Flows** (Havasi et al., 2025). We use our own implementation, which isolates the contribution of block insertions while holding the architecture, data, coupling and training schedule fixed.
- **FlexMDM** (Kim et al., 2026). We use the authors' reference implementation, sized to match our backbone (512 hidden, 8 heads, 8 blocks) and trained for the same 800 epochs on the identical FASTA file, with the same split, seed, batch size, learning rate, warmup, EMA decay and maximum length. Table 1 reports three of its samplers: adaptive τ-leaping, vanilla τ-leaping and Euler.
- **Planner + ESM-2**. The planner is our own, trained as above. The denoiser is a frozen pre-trained ESM-2 (Lin et al., 2023) `t30_150M` checkpoint, used zero-shot with no fine-tuning and no divergence or timestep conditioning. At each step it resolves every mask token in one forward pass by sampling from its masked-language-model head at temperature 1, restricted to the 20 canonical amino acids. This is the drop-in denoiser of Proposition 2.
- **AMP generators** (Table 2). Sequences for the purpose-built AMP generators are taken as released by their authors, and every method, ours included, is then evaluated under one common protocol: 1,000 sequences subsampled per method, scored with the same predictors and the same reference set.
  - **AMP-Diffusion** (Torres et al., 2025). Latent diffusion on pLM embeddings with positive-only learning.
  - **Muller-LSTM** (Müller et al., 2018). LSTM-based RNN for de novo peptide sequence generation.
  - **MMCD** (Wang et al., 2024). Multi-modal contrastive diffusion for sequence-structure co-generation.
  - **AMPGAN-v2** (Van Oort et al., 2021). Bidirectional conditional GAN (BiCGAN) for conditional AMP generation.
  - **MLpeptide** (Capecchi et al., 2021). RNN with discriminator-guided filtering and positive-only learning for active non-toxic AMPs.
  - **Basic CVAE**. Basic conditional VAE baseline (ablation from HydrAMP (Szymczak et al., 2023))
  - **OmegAMP-U** (Soares et al., 2026). Diffusion-based unconditional AMP generation.
  - **OmegAMP-P** (Soares et al., 2026). Diffusion-based de novo generation, conditioned on vectors derived from prototype peptides with demonstrated antimicrobial activity.
  - **OmegAMP-T** (Soares et al., 2026). Diffusion-based de novo generation, conditioned on vectors derived from therapeutically relevant property ranges: net charge 2–10, length 5–30 residues, and mean hydrophobicity -0.5 to $+0.8$.

- **HydrAMP** (Szymczak et al., 2023). Conditional VAE with creativity parameter for unconstrained and analogue AMP generation.
- **Hyformer** (Izdebski et al., 2025). Hybrid transformer for conditional AMP generation.

Optimized and non-optimized methods. Table 2 is split into two blocks and ranked within each, since a sequence produced by activity-directed search is not comparable to one drawn from a generative distribution. The non-optimized block holds methods with no explicit activity target at generation time: unconditional sampling, conditioning on attributes other than activity such as length or structure, and steering towards a reference sequence. The optimized block holds methods that receive additional guiding signal for activity, either by conditioning on an activity value, or on physicochemical properties characteristic of active peptides, or by filtering candidates through a classifier. Mutation Flows appears in both blocks, as ordinary sampling and as the guided search of Algorithm 3 applied to the same model, which separates what the model contributes from what the search adds.

Sampling and evaluation protocol. Unless stated otherwise, sampling uses Euler discretization with 50 steps ($h = 0.02$). Source sequences x_0 for the unconditional arms are peptides drawn from the corpus. For the distortion-coupled arms they are distortions of those peptides generated with the same rates the corresponding checkpoints were trained on, so that each arm is sampled from the source distribution it was fitted to. Conditioned models are evaluated at $d \in \{0.10, 0.15, 0.20, 0.30\}$. Unconditioned models receive $d = 1$. Trajectory metrics are read at 10 uniformly spaced points along the trajectory. Because the planner–denoiser variant splits one 46.6M budget over its two networks, its two passes per step cost what a single-network method spends in one, so every row of Table 1 at NFE N takes N sampling steps.

Guided optimization. The optimized row of Table 2 uses Algorithm 3: a population of source peptides is advanced by ordinary Mutation Flows steps, each particle proposing C stochastic children per step, and selection keeps the child best aligned with the particle's preference vector within a cone of half-angle $\phi_{\max}$, drawing candidates from the M neighbouring preference vectors as well as its own, or the least misaligned child if the cone is empty. Preference vectors are a Das–Dennis simplex lattice and are fixed for the run. Objectives are calibrated by an affine map fitted to their range over a screening pool of peptides, so that no objective dominates the direction comparison by virtue of a wider raw range. The map is clamped below at a small positive offset, so that every score vector has a direction, but not above, so that the search is not capped at the best sequence in the pool. Children that cannot be scored are assigned $-\infty$ so that they lose selection rather than poisoning it. No retraining and no gradients through the predictors are required. We guide against the Hyformer MIC regressor alone, and APEX is never optimized against.

Guided optimization settings. The reported run advances $P = 1{,}000$ particles for $N_{\text{iter}} = 50$ steps, the same step count as unguided sampling, with the full divergence-conditioned model at $d = 0.30$, $C = 4$ children per particle and $\phi_{\max} = 25°$. We set $M = 0$: with a single objective every preference vector is identical, so neighbours would be decided by tie-breaking rather than by preference and the population would collapse onto one sequence. The run is therefore 1,000 independent hill climbs, each keeping the best-scoring of its C children and returning one sequence, and the cone and the preference lattice matter only for the multi-objective sets. Couplings are run from a common seed, so comparisons between them are paired, and the unconditional arm is guided at $d = 1$ under otherwise identical settings.

Runtime. Each model was trained on a single NVIDIA H100 80GB GPU for 800 epochs, which depending on the model took between 8 and 20 hours. Generating the sequences for the experiments and computing the metrics took 3 hours on a NVIDIA A100 40GB GPU.

Metrics. Let $\mathcal{G} = \{g_i\}_{i=1}^{N}$ denote a set of generated sequences, $\mathcal{R} = \{r_i\}_{i=1}^{M}$ a reference set of 2,000 peptides drawn from the corpus, and $\phi(\cdot)$ ESM-C (Candido et al., 2026) 600M embeddings, taken as the mean over token representations excluding special tokens. Distributional metrics are averaged over 4 disjoint folds of 500 generated sequences each, reported as mean $\pm$ standard deviation across folds, and evaluated at 10 uniformly spaced denoising timesteps. Per-sequence predictor

Table 3: Hyperparameter settings for Mutation Flows. The single-model column is mirrored by the Edit Flows and FlexMDM baselines wherever they have a corresponding setting. The planner and denoiser split the same parameter budget roughly 30 / 70, coming to 46.5M together, and all three share the training settings in the lower block.

Hyperparameter	**Single model**	**Planner**	**Denoiser**
Embedding dim	512	240	416
Transformer layers	8	8	8
Attention heads	8	5	13
Hidden dim	2,048	960	1,664
Conditioning embedding dim	512	512	512
Parameters	46.6M	13.7M	32.8M
Max sequence length		52	
Noise schedule κ_t		t (linear)	
Optimizer		Adam	
Learning rate		1×10^{-4}	
Warmup steps		25,000	
Gradient clip		1.0	
Batch size		512	
Epochs		800 ($\approx$ 1.36M steps)	
EMA decay		0.9999	
Precision		bf16 mixed	

metrics are reported as mean $\pm$ one standard error over sequences. All metrics are computed with `seqme` (Møller-Larsen et al., 2026). Every condition is evaluated at the same number of sequences, since the MMD estimator below is biased in N. The metrics used are defined below.

- **Maximum Mean Discrepancy (MMD).** Using a radial basis function kernel $k(u, v) = \exp\big(-\|u - v\|^2/2\sigma^2\big)$ with $\sigma = 10$ applied to ESM-C embeddings, the biased estimator is

$$\mathrm{MMD}^2 = \frac{1}{N^2}\sum_{i,j} k(\phi(g_i), \phi(g_j)) - \frac{2}{NM}\sum_{i,j} k(\phi(g_i), \phi(r_j)) + \frac{1}{M^2}\sum_{i,j} k(\phi(r_i), \phi(r_j)). \tag{24}$$

  Lower values indicate closer agreement between the generated and reference distributions. Reported values are $10^3 \cdot \mathrm{MMD}^2$ throughout, following Møller-Larsen et al. (2026).
- **OmegAMP score.** The probability assigned by the OmegAMP antimicrobial-activity classifier (Soares et al., 2026), averaged over sequences. Used as a measure of peptide-likeness of the intermediate states along the trajectory, where higher is better.
- **Hyformer perplexity.** The perplexity of a sequence $x = (x_1, \ldots, x_n)$ of $n = n(x)$ residues under the Hyformer peptide model (Izdebski et al., 2025) (`hyformer_peptides_34M_mic` checkpoint), whose generative head factorizes a sequence autoregressively as $p_\theta(x) = \prod_{j=1}^{n} p_\theta(x_j \mid x_{<j})$,

$$\mathrm{PPL}(x) = \exp\left(-\frac{1}{n}\sum_{j=1}^{n} \log p_\theta(x_j \mid x_{<j})\right) = \exp\left(\frac{\mathrm{NLL}(x)}{n}\right), \tag{25}$$

  where $\mathrm{NLL}(x) = -\log p_\theta(x)$ is the negative log-likelihood of the whole sequence and n is the number of scored positions. We report $\frac{1}{N}\sum_{i=1}^{N} \mathrm{PPL}(g_i)$ over the generated set.
- **p_{amp} and hit-rate.** The probability of antimicrobial activity assigned by amPEPpy (Lawrence et al., 2021), and the fraction of generated sequences with $p_{\mathrm{amp}} \geq 0.5$. amPEPpy is a random-forest classifier independent of every model compared, and of the predictors used for guidance.
- **$\log_2$ MIC (Hyformer) and MIC (APEX).** Predicted minimum inhibitory concentrations, from the Hyformer regression head (Izdebski et al., 2025) and from APEX (Wan et al., 2024) against *E. coli* ATCC11775 respectively. Hyformer is the guidance objective in the optimized rows and is therefore in-objective there.

- **Diversity.** The mean normalized pairwise Levenshtein distance among generated sequences,

$$\text{Diversity}(\mathcal{G}) = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{|\mathcal{K}_i|}\sum_{j\in\mathcal{K}_i}\frac{d_{\text{edit}}(g_i, g_j)}{\max\big(n(g_i), n(g_j)\big)}, \tag{26}$$

  where $\mathcal{K}_i$ is a random subsample of $k = 100$ other generated sequences, which keeps the cost linear in N. Higher values indicate less redundancy within the generated set.

- **Novelty and uniqueness.** Novelty is the fraction of generated sequences that do not occur verbatim in the reference set. Uniqueness is the fraction of generated sequences that are distinct.

- **Edit distance.** The Levenshtein distance between sequences x and y,

$$d_{\text{edit}}(x, y) = \min_{\text{edits}}\big|\{e : e \text{ applied}\}\big|, \tag{27}$$

  is the minimum number of single-residue insertions, deletions, and substitutions required to transform x into y. We report $d_{\text{edit}}(x_0, x_t)$ at each denoising timestep t to trace how far the trajectory has moved from the source sequence.

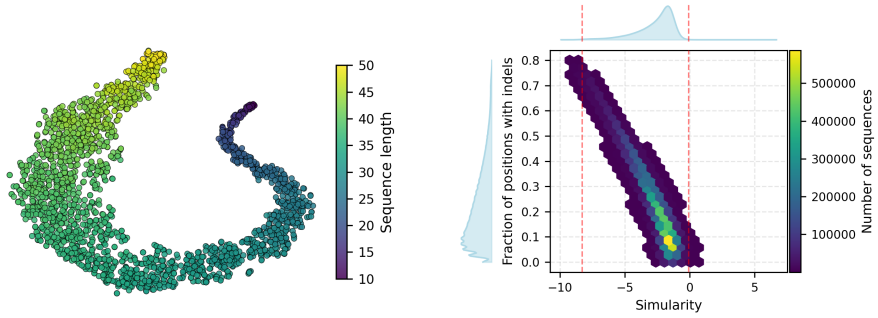
- **Absolute length difference.** To isolate the length-change component of the trajectory independently of substitutions,

$$\Delta\ell(t) = \big|n(x_t) - n(x_0)\big|, \tag{28}$$

  where $n(x)$ denotes the length of sequence x. This is particularly informative for models that perform insertions and deletions.

C Additional Results

C.1 Divergence Conditioning



(a) Distance manifold.

(b) Sequence similarity distribution.

Figure 3: The unconditional coupling exposes the model to the full empirical distribution of pairwise distances, and divergence conditioning selects within it. The distance manifold (a) shows sequences coloured by length in a t-SNE embedding. The similarity distribution (b) shows training pairs by length-normalized alignment score s/ℓ_{aln}, which the distance inverts, $\eta = 1 - \text{norm}(s/\ell_{\text{aln}})$, against the fraction of aligned positions carrying an indel. The dashed vertical lines mark the clipping range $[-8.3, -0.1]$ used to exclude low-density regions.

Figure 3 shows the dissimilarity measure η of Section 4 on the training data. Under this dissimilarity, antimicrobial peptide sequences form a non-trivial manifold in which clusters are organized by length (Figure 3a), and the pairwise distances induced by independent pairs concentrate in a characteristic range (Figure 3b). The unconditional coupling exposes the model to this full spread, while divergence conditioning allows it to target any point within it at inference time.

Figure 4 shows that this control is realized at inference time: trajectories generated under different requested d separate cleanly in both edit distance and length change, while the arm trained on the same pairs without divergence conditioning does not respond to d at all.

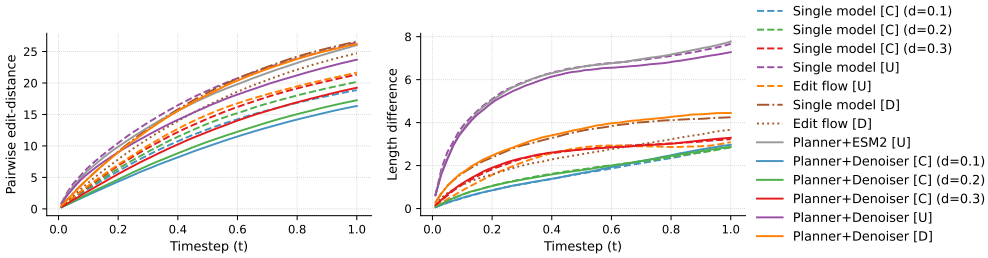


Figure 4: Pairwise edit distance (left) and absolute length difference (right) between the source sequence and the current sequence as a function of denoising timestep, shown for $d \in \{0.10, 0.20, 0.30\}$, three of the four values at which conditioned models are evaluated, with $d = 0.15$ omitted for legibility. Divergence conditioning produces trajectories that are clearly separated across values of d and smoother than the distortion coupling. The arm without divergence conditioning, trained on identical pairs, does not respond to d, isolating the effect of the conditioning signal from that of the pairing.

C.2 Antimicrobial peptide-likeness along the trajectory

MMD (Figure 2) compares embedding distributions and can be small for sequences that are not individually plausible. Figure 5 therefore measures antimicrobial peptide-likeness of the intermediate states directly, and separates the methods more sharply.

Planner + Denoiser with divergence conditioning holds an OmegAMP score of 0.005–0.012 across the whole trajectory and at every d, and its $d = 0.10$ arm stays above 0.008 throughout. Every other configuration decays over the first fifth of the trajectory: the conditioned single model and

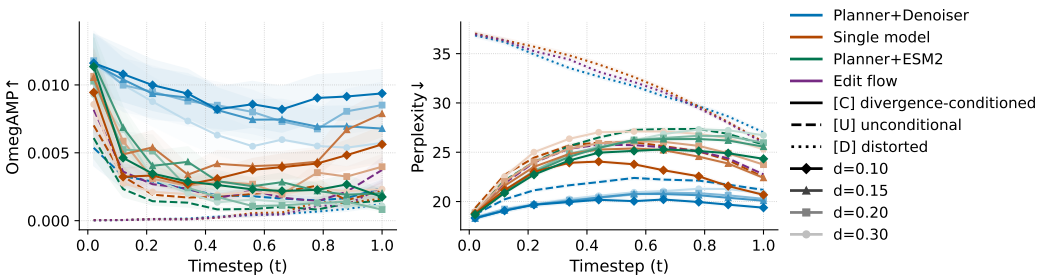


Figure 5: Peptide-likeness along the denoising trajectory: OmegAMP classifier score (left, higher is better) and Hyformer perplexity (right, lower is better). Planner + Denoiser under divergence conditioning is the only configuration that keeps intermediate states recognisably peptide-like.

Planner + ESM-2 to roughly 0.002–0.005, the unconditional arms to ≈ 0.002, and the distortion-coupled arms to near zero until the very end. Only the conditioned single model recovers towards the endpoint, and only at the two smallest d. Hyformer perplexity agrees, staying between 18 and 21 for the conditioned full model at every d. The unconditional Planner + Denoiser peaks at 22.4, every remaining arm rises to between 24 and 27 mid-trajectory, and the distortion-coupled trajectories start at ≈ 37 and fall no lower than ≈ 26 by the endpoint. Under the distortion coupling the intermediate states are not peptides at all, which is what makes them unusable as starting points for sampling-based guidance.

C.3 Block insertion distribution

Figure 6 inspects the insertion rate predicted by the planner along the denoising trajectory.

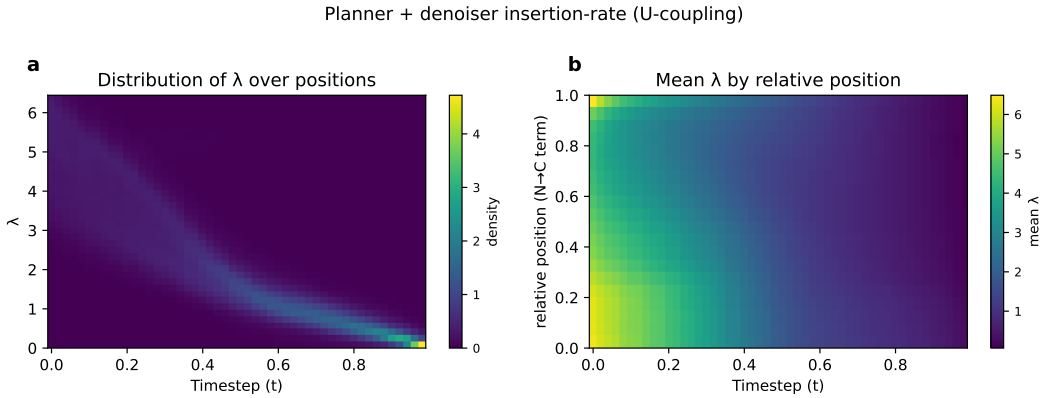


Figure 6: Insertion-rate diagnostics for Planner + Denoiser trained under the unconditional coupling. **(a)** Density of the predicted insertion rate λ^{ins} across positions as a function of the denoising timestep: early in the trajectory the model spreads mass over large rates, and as $t \to 1$ the distribution concentrates at $\lambda^{\text{ins}} \approx 0$, so that the sequence stops growing once the remaining divergence reaches zero. **(b)** Mean insertion rate as a function of relative position along the sequence (0 = N-terminus, 1 = C-terminus) and timestep: insertion mass is concentrated early in the trajectory and towards the N-terminal end.

D ALGORITHMS

Algorithm 1: Mutation Flows: Training (Planner + Denoiser, unconditional coupling with divergence conditioning)

Inputs: number of iterations N_{iter}, data distribution μ, dissimilarity η, noise schedule κ_t.

for $n = 1$ **to** N_{iter}

- $x_0 \sim \mu, \quad x_1 \sim \mu$; `// unconditional coupling` $\pi^{\text{u}} = \mu \otimes \mu$
- $d \leftarrow \eta(x_0, x_1)$; `// realized pairwise distance`
- $t \sim \mathcal{U}(0,1), \quad b_t \leftarrow (1-\kappa_t)\, d$; `// remaining divergence`
- `// Atom-wise mixture path`
- $(z_0, z_1) \leftarrow \text{align}(x_0, x_1)$
- $\mathcal{A} \leftarrow \text{atoms}(z_0, z_1)$; `// one atom per edit`
- $z_t^g \sim (1-\kappa_t)\, \delta_{z_0^g} + \kappa_t\, \delta_{z_1^g}$ for each $g \in \mathcal{A}$; `// Eq. 12`
- $x_t \leftarrow \text{f}_{\text{rm}}(z_t)$; `// drop gaps`
- `// Planner`
- $\lambda_t^{\text{ins}}, \lambda_t^{\text{del}}, \lambda_t^{\text{sub}}, \lambda_t^{+} \leftarrow m_{\theta_P}(x_t, \kappa_t, b_t)$
- $\mathcal{L}_{\text{planner}} \leftarrow$ Eq. 19
- `// Denoiser`
- $w_t^i \leftarrow$ [mask] at the positions of the atoms in $\mathcal{A}_t^{\text{ins}} \cup \mathcal{A}_t^{\text{sub}}$, else z_1^i
- $w_t \leftarrow \text{f}_{\text{rm}}(w_t)$; `// drop deletion gaps, the positions with` $z_1^i = \varepsilon$
- $\mathcal{L}_{\text{denoiser}} \leftarrow$ Eq. 20; `// masked cross-entropy, Prop. 2`
- Update θ_P with $\nabla_{\theta_P} \mathcal{L}_{\text{planner}}, \quad \theta_D$ with $\nabla_{\theta_D} \mathcal{L}_{\text{denoiser}}$

Algorithm 2: Mutation Flows: Sampling

Inputs: number of steps N_{iter}, step size $h = 1/N_{\text{iter}}$, source sequence x, target distance d.
Returns: generated sequence x.

for $s = 1$ **to** N_{iter}
  $t \leftarrow (s-1)\,h, \quad b_t \leftarrow (1-\kappa_t)\,d$
  `// Planner: edit rates and block-length parameters`
  $\lambda_t^{\text{ins}}, \lambda_t^{\text{del}}, \lambda_t^{\text{sub}}, \lambda_t^{+} \leftarrow m_{\theta_P}(x, \kappa_t, b_t)$
  `// Sample one operation per position of` x`, independently`
  for $i = 1$ **to** $n(x)$
    $\Lambda_i \leftarrow \lambda_{t,i}^{\text{ins}} + \lambda_{t,i}^{\text{del}} + \lambda_{t,i}^{\text{sub}}$
    $\text{op}^i \sim \text{Cat}\big(\text{nop}, \text{ins}, \text{del}, \text{sub} \;\big|\; 1 - h\Lambda_i,\ h\lambda_{t,i}^{\text{ins}},\ h\lambda_{t,i}^{\text{del}},\ h\lambda_{t,i}^{\text{sub}}\big)$
    if $\text{op}^i = \text{ins}$ **then**
      $k^i \sim \text{ZTP}(\lambda_{t,i}^{+})$; `// block length`
  `// Apply them in one pass, so every index refers to` x`, not to` x'
  $x' \leftarrow (\,)$; `// empty sequence`
  for $i = 1$ **to** $n(x)$
    if $\text{op}^i = \text{sub}$ **then**
      append [`mask`] to x'
    else if $\text{op}^i = \text{ins}$ **then**
      append x^i and then k^i [`mask`] tokens to x'
    else if $\text{op}^i = \text{nop}$ **then**
      append x^i to x'
    else
      append nothing; `//` $\text{op}^i = \text{del}$: `position` i `is dropped`
  `// Denoiser: resolve every mask in one forward pass`
  $\hat{p} \leftarrow m_{\theta_D}(x', \kappa_t, b_t)$
  $x'^i \sim \text{Cat}(\hat{p}^i)$ for all i with $x'^i = [\texttt{mask}]$
  $x \leftarrow x'$
return x

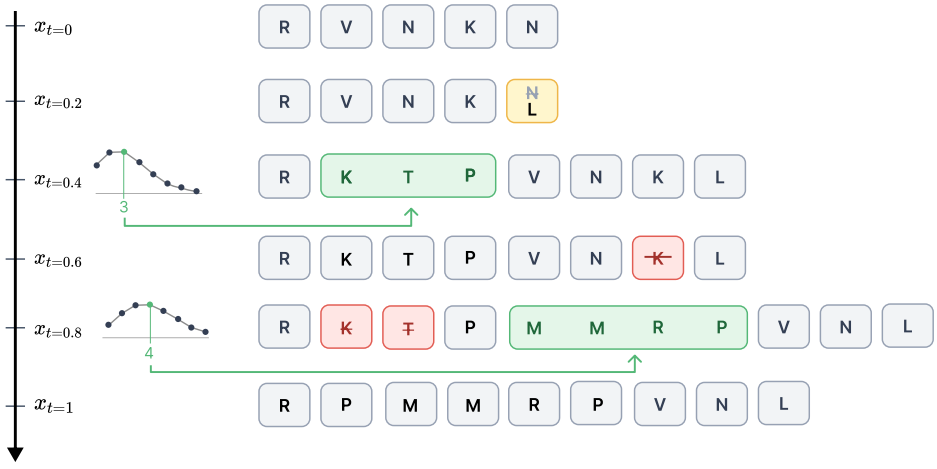


Figure 7: Unconditional sampling visualization. At each step, the sequence is modified through substitutions, deletions, and block insertions. Block lengths are drawn from a zero-truncated Poisson distribution.

Algorithm 3: Mutation Flows: Multi-objective sampling

Inputs: steps N_{iter}, step size $h = 1/N_{\text{iter}}$, source sequences $x_1, \ldots, x_P$, target distance d, preference vectors $w_1, \ldots, w_P$, calibrated predictors $f_1, \ldots, f_K$ (all maximized), children per particle C, neighbours M, cone half-angle $\phi_{\max}$.

Returns: population $x_1, \ldots, x_P$.

$\mathcal{N}_p \leftarrow$ indices of the $M+1$ preference vectors nearest w_p in angle, for each p; `// fixed: w never changes`

for $s = 1$ **to** N_{iter}

  $t \leftarrow (s-1)\,h, \quad b_t \leftarrow (1-\kappa_t)\,d$

  `// Propose: C stochastic children per particle, one batched flow step`

  $y_{p,c} \leftarrow$ MutationFlowStep(x_p, t, b_t, h) for all $p \in [P]$, $c \in [C]$; `// Alg. 2`

  `// Score: one vector per child`

  $S_{p,c} \leftarrow \big(f_1(y_{p,c}), \ldots, f_K(y_{p,c})\big) \in \mathbb{R}^K$

  $S_{p,c} \leftarrow -\infty$ on any objective returning `NaN`; `// unscorable must lose, not poison the argmax`

  `// Select: each particle keeps its best-aligned neighbourhood child`

  for $p = 1$ **to** P

    $\mathcal{Y}_p \leftarrow \{\, y_{q,c} : q \in \mathcal{N}_p,\ c \in [C] \,\}$; `// (M+1)C candidates`

    $\theta(y) \leftarrow \angle\big(S_y,\ w_p\big)$ for $y \in \mathcal{Y}_p$

    $\mathcal{A} \leftarrow \{\, y \in \mathcal{Y}_p : \theta(y) < \phi_{\max} \,\}$; `// cone about the preference`

    if $\mathcal{A} \neq \emptyset$ **then**

      $x'_p \leftarrow \arg\max_{y \in \mathcal{A}} \langle S_y,\ w_p \rangle$; `// scalar projection`

    else

      $x'_p \leftarrow \arg\min_{y \in \mathcal{Y}_p} \theta(y)$; `// cone empty: least misaligned`

  $x_p \leftarrow x'_p$ for all $p \in [P]$; `// update after selection, so` $\mathcal{Y}_p$ `is built from one generation`

return $x_1, \ldots, x_P$

The selection mechanism and the settings used for the Table 2 run are described in Appendix B.

E Broader Impact

Mutation Flows is a general-purpose biological sequence design tool with applications to therapeutic peptides, proteins, and nucleic acids. Positive applications include antimicrobial peptide design and protein engineering for medicine and biotechnology. As with any generative sequence design method, there is a risk of misuse for designing harmful sequences or contributing to biosecurity concerns.