RNAGenScape: Property-Guided, Optimized Generation of mRNA Sequences with Manifold Langevin Dynamics

Danqi Liao[*,1] **Chen Liu**[*,1] **Xingzhi Sun**[*,1] **Dié Tang**[1] **Haochen Wang**[1]
Scott Youlten[1] **Srikar Krishna Gopinath**[1] **Haejeong Lee**[1] **Ethan C. Strayer**[1]
Antonio J. Giraldez[1✉] **Smita Krishnaswamy**[1✉]

[*]Equal contribution [1]Yale University
✉{antonio.giraldez,smita.krishnaswamy}@yale.edu

Abstract

Generating property-optimized mRNA sequences is central to applications such as vaccine design and protein replacement therapy, but remains challenging due to limited data, complex sequence-function relationships, and the narrow space of biologically viable sequences. Generative methods that drift away from the data manifold can yield sequences that fail to fold, translate poorly, or are otherwise nonfunctional. We present RNAGenScape, a property-guided manifold Langevin dynamics framework for mRNA sequence generation that operates directly on a learned manifold of real data. By performing iterative local optimization constrained to this manifold, RNAGenScape preserves biological viability, accesses reliable guidance, and avoids excursions into nonfunctional regions of the ambient sequence space. The framework integrates three components: (1) an autoencoder jointly trained with a property predictor to learn a property-organized latent manifold, (2) a denoising autoencoder that projects updates back onto the manifold, and (3) a property-guided Langevin dynamics procedure that performs optimization along the manifold. Across three real-world mRNA datasets spanning two orders of magnitude in size, RNAGenScape increases median property gain by up to 148% and success rate by up to 30% while ensuring biological viability of generated sequences, and achieves competitive inference efficiency relative to existing generative approaches.

1 Introduction

Messenger ribonucleic acids (mRNAs) have emerged as a core modality for modern therapeutics, enabling applications ranging from mRNA vaccine design and protein replacement therapy to gene regulation and synthetic biology [1–4]. In these settings, the nucleotide sequence of an mRNA directly determines properties such as stability, translational efficiency, and ultimately protein yield. Even small sequence edits, particularly in non-coding regions such as the 5' untranslated region (UTR), can lead to substantial changes in degradation rates and ribosome loading [5–7]. As a result, systematic optimization of the mRNA sequence has the potential to improve therapeutic efficacy, reduce dosage requirements, and expand the design space of viable mRNA-based interventions.

Despite its importance, mRNA sequence generation and optimization remains underexplored from a machine learning perspective [8, 9]. A key challenge is that viable mRNA sequences occupy a narrow and structured subset of the astronomically large ambient sequence space [10, 11]. Random or unconstrained edits can lead to nonfunctional transcripts, while experimentally measured datasets are typically small, imbalanced, and sparsely cover the space of interest [12, 13]. Moreover, sequence-

Preprint.

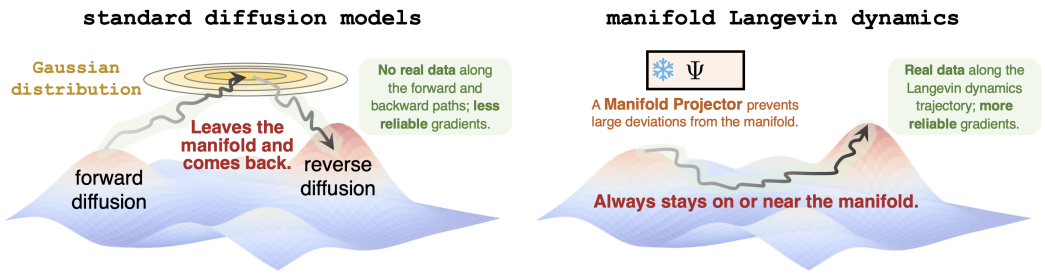

Figure 1: Advantage of manifold Langevin dynamics for optimization on the manifold of real data.

function relationships in mRNA are highly nonlinear and context-dependent, making the optimization of mRNA sequences a challenging problem [14, 15].

To address the challenges, we introduce `RNAGenScape`, a property-guided manifold Langevin dynamics framework for mRNA sequence generation built around three key components. ❶ An autoencoder and a property predictor, together termed an organized autoencoder (OAE), that jointly learn a latent manifold of mRNA sequences while organizing this manifold according to the target property. This latent manifold represents the subspace of biologically viable mRNA sequences rather than the much larger Euclidean ambient space. ❷ A manifold projector, implemented as a denoising autoencoder, that maps data near the learned manifold onto the manifold, preventing drift toward implausible regions. To ensure robustness under sparse and imbalanced data regimes common in mRNA studies, we augment training data using SUGAR [16], which fills holes in the manifold and stabilizes manifold projection even with as few as 2,000 sequences. ❸ A property-guided manifold Langevin dynamics that iteratively refines sequences by combining gradient information with stochastic exploration, while ensuring that sequences generated at every optimization step remain close to the viable mRNA manifold.

Our choice to perform iterative optimization directly on the learned data manifold, rather than adopting diffusion-based generation from Gaussian noise, is motivated by several practical considerations. **First**, mRNA sequence optimization is inherently local, where improvements are typically made relative to an existing sequence. **Second**, property-guided Langevin dynamics rely on gradients from a predictor trained on real data, and therefore staying close to the data manifold helps obtain stable and meaningful guidance. In contrast, standard diffusion and flow-matching models frequently traverse off-manifold regions during generation, leading to unstable or misleading guidance from predictors. **Third**, constraining the optimization trajectory to the manifold encourages the generated sequences to remain within biologically viable regions of sequence space. **Finally**, avoiding repeated diffusion to and denoising from a noise distribution far away from the data manifold reduces computational overhead and enables faster inference. Figure 1 provides a conceptual illustration.

We evaluate `RNAGenScape` on three real-world mRNA datasets under diverse experimental settings. Across these datasets, `RNAGenScape` improves median property gain by up to 148% over state-of-the-art methods and success rates by up to 30%, while preserving biological viability and demonstrating generalization in a held-out optimization study. In addition, `RNAGenScape` is highly efficient with a higher throughput than the fastest biological sequence optimization method.

2 Preliminaries

2.1 Manifold assumption and manifold learning

The manifold assumption [17–19] is a fundamental principle in machine learning which hypothesizes that high-dimensional natural data lie near a low-dimensional manifold embedded in the ambient space [20]. Formally, each observation $x_i \in \mathbb{R}^n$ arises from a smooth nonlinear map $\mathbf{f} : \mathcal{M}^d \to \mathbb{R}^n$ applied to a latent variable $z_i \in \mathcal{M}^d$, where $d \ll n$.

Manifold learning methods seek to recover this latent structure by constructing representations that preserve intrinsic geometry [21–30]. A point is considered *on-manifold* if it lies within the range of the nonlinear map $\mathbf{f}$, while *off-manifold* points deviate from this structure and may correspond to invalid or adversarial samples [31, 32]. Thus, projecting updated points back to the manifold is critical for robustness and geometry-aware optimization [33].

SUGAR [16] is a diffusion geometry-based method that learns the intrinsic geometry of the data manifold and samples uniformly along the learned geometry. When observed data are limited, augmenting training sets with SUGAR enriches the learned manifold with geometry-preserving samples, particularly in sparse regions.

Stochastic gradient descent (SGD) on Riemannian manifolds [34] extends classical SGD by computing updates in the tangent space and mapping them back to the manifold via exponential maps or retractions. However, such methods assume an analytic form of the manifold. In contrast, the manifold underlying biological sequence space is not known in a closed form. `RNAGenScape` addresses this by directly *learning* the projection operator, enabling optimization on data manifolds without requiring analytic solutions.

2.2 Langevin-dynamics and beyond

Langevin Dynamics [35] has been employed in generative models to sample from high-dimensional data distributions using only an estimate of the score function $\nabla_x \log p(x)$. In particular, it first trains a neural network $\boldsymbol{s_\theta}$ to approximate the score function of data injected with Gaussian noise. Sampling is then performed via annealed Langevin dynamics, given by $\tilde{x}_t = \tilde{x}_{t-1} + \frac{\eta_i}{2}\boldsymbol{s_\theta}(\tilde{x}_{t-1}, \sigma_i) + \sqrt{\eta_i}\boldsymbol{z_t}$. Here, $\boldsymbol{s_\theta}(\tilde{x}_{t-1}, \sigma_i)$ is the learned score function at noise level σ_i, and η_i is the step size at that level. By gradually annealing from high to low noise, this procedure enables generation of high-quality samples without an explicit likelihood or energy model.

Neural Stochastic Differential Equations (neural SDEs) [36] are differential equations simultaneously modeling two terms: a drift term $f(\cdot)$ depicting the true time-varying dynamics of the variable, and a diffusion term $g(\cdot)$ representing stochasticity using the Brownian motion W_t. The update rule is given by $\mathrm{d}X_t = f(t, X_t)\mathrm{d}t + g(t, X_t) \circ \mathrm{d}W_t$. From a high level, Langevin dynamics is a special case of neural SDEs after discretization.

3 `RNAGenScape`

The key components of our framework are ❶ **OAE**: an autoencoder jointly trained with a prediction model to learn a latent manifold of mRNA sequences organized by the target property (Section 3.1 and Figure 2b), ❷ **Manifold Projector**: a denoising autoencoder that brings the updated latent embeddings back to the learned data manifold during each step of optimization (Section 3.2 and Figure 2c, 2e), and ❸ **Property-guided manifold Langevin dynamics**, a procedure that integrates the two aforementioned modules to enable property-guided generation (Section 3.3 and Figure 2a).

Once trained, these components allow `RNAGenScape` to optimize the target property, such as translation efficiency or stability, of a given mRNA sequence.

3.1 Learning a latent manifold organized by property

We begin by training an **organized autoencoder (OAE)**, where the latent manifold is implicitly structured via supervision from a property prediction task (Figure 2b).

Similar to a vanilla autoencoder [37], the encoder $\mathcal{E}$ maps the input mRNA sequence x to a latent representation z, which is decoded by $\mathcal{D}$ back to the sequence space. In addition to this standard architecture, a predictor $\mathcal{P}$ infers properties $\hat{y}$ from the embedding z. Formally, $z = \mathcal{E}(x)$, $\hat{x} = \mathcal{D}(z)$, $\hat{y} = \mathcal{P}(z)$. The latent manifold $\mathcal{Z}$ is thus shaped by jointly optimizing the reconstruction loss and the prediction loss (Eqn (1)), encouraging it to learn sequence-relevant information while being organized by the target properties. λ_{Pred} and λ_{Recon} are hyperparameters that balance between organizing the property landscape and capturing sequence information. They are empirically set as described in Supplementary Section B. Here, $x_i \in \mathbb{R}^V$ is the ground truth one-hot encoding of the nucleotide at position i in sequence x with an mRNA vocabulary of size V.

$$\begin{aligned}\mathcal{L}_{\text{OAE}} &= \lambda_{\text{Pred}}\mathcal{L}_{\text{Pred}} + \lambda_{\text{Recon}}\mathcal{L}_{\text{Recon}} \\ &= \overbrace{\lambda_{\text{Pred}}\ \mathbb{E}_{(x,y)\sim p_{\text{data}}}\|\hat{y} - y\|_2^2}^{\text{optimize } \mathcal{P} \text{ and } \mathcal{E} \text{ with MSE}} \overbrace{-\lambda_{\text{Recon}}\ \mathbb{E}_{x\sim p_{\text{data}}}\frac{1}{L}\sum_{i=1}^{L}\log\frac{\exp(\hat{x}_{i,x_i})}{\sum_{v=1}^{V}\exp(\hat{x}_{i,v})}}^{\text{optimize } \mathcal{E} \text{ and } \mathcal{D} \text{ with CrossEntropy}}\end{aligned} \tag{1}$$

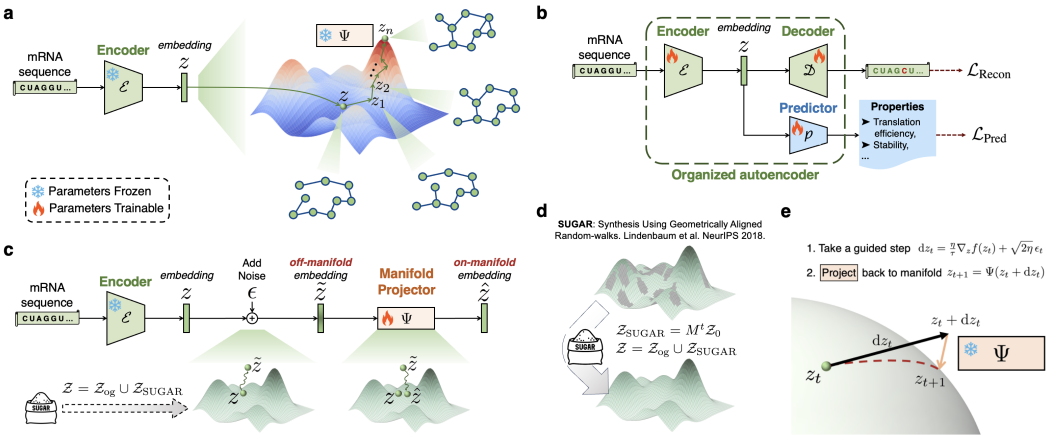

Figure 2: Schematic of `RNAGenScape`. **a,** We encode a given input mRNA sequence into a latent manifold organized by property, and iteratively optimize the sequence along the property landscape. The manifold projector Ψ ensures the optimization stays on or near the manifold. Notably, we can decode the intermediate optimization result after every step. **b,** The organized latent manifold is created by jointly optimizing reconstruction and property prediction objectives. **c,** The manifold projector is trained to project perturbed data points back to the manifold. **d,** For undersampled mRNA manifolds, we use SUGAR [16] to learn key dimensions in the manifold and fill undersampled regions. **e,** During optimization, the manifold projector brings off-manifold points back to the manifold.

3.2 Learning a module to project updates to manifold

Learning the data manifold mRNA datasets are typically scarce and unevenly sampled, which can hinder learning a faithful latent manifold directly from observed data. To mitigate this issue, we augment the latent embeddings using SUGAR [16], a diffusion geometry-based method introduced in Section 2. This augmentation enriches the learned manifold with geometry-preserving samples, particularly in sparse regions, improving robustness of subsequent manifold projection.

Specifically, this preprocessing step yields an expanded latent set: $\mathcal{Z} = \mathcal{Z}_{\text{og}} \cup \mathcal{Z}_{\text{SUGAR}}$, $\mathcal{Z}_{\text{SUGAR}} = M^t \mathcal{Z}_0$, where $\mathcal{Z}_{\text{og}}$ are the original latent embeddings, $\mathcal{Z}_0$ are locally-sampled neighbors, and M^t is the sparsity-corrected Markov diffusion transition matrix applied for t steps. The SUGAR augmentation is performed in the latent space: we encode training sequences into latent representations, apply SUGAR to enrich this space with geometry-consistent neighbors, and then train the manifold projector.

Training the manifold projector To access reliable property guidance during generation and keep the generated trajectories aligned with the latent data manifold, we introduce a manifold projector Ψ. As illustrated in Figure 2c and magnified in Figure 2e, Ψ takes in a noisy optimized point $\tilde{z}$ and projects it back onto or near the manifold. To train Ψ, we adopt a denoising objective that projects noisy samples back towards the clean points on the latent manifold. Given a clean latent embedding z, we construct a short corruption chain $\tilde{z}^{(0)} = z, \quad \tilde{z}^{(k)} \sim C(\tilde{\mathcal{Z}}|\tilde{\mathcal{Z}}^{(k-1)}, \sigma_k), \quad k = 1, \ldots, K$, where $C(\cdot, \sigma_k)$ denotes Gaussian corruption with noise level σ_k. The projector is trained to reverse each step by predicting $\tilde{z}^{(k-1)}$ from $\tilde{z}^{(k)}$, yielding the following objective shown in Eqn (2).

$$\mathcal{L}_\Psi = \mathbb{E}_{z \sim p_{\text{data}}} \sum_{k=1}^{K} \mathbb{E}_{\tilde{z}^{(k)} \sim C(\cdot|\tilde{z}^{(k-1)},\ \sigma_k)} \left[\|\Psi(\tilde{z}^{(k)}) - \tilde{z}^{(k-1)}\|_2^2 \right] \tag{2}$$

When $K = 1$, this reduces to the standard denoising autoencoder loss [38]. We keep K small (e.g. $K \leq 3$) to capture *local updates near the data manifold*, rather than simulating long diffusion chains from Gaussian noise. As a result, our algorithm is fast during training and inference.

3.3 Property-guided manifold Langevin dynamics

Next, we introduce a novel property-guided manifold Langevin-dynamics framework.

Given a trained encoder $\mathcal{E}$, a property predictor $\mathcal{P}$, and a manifold projector Ψ, our Langevin-dynamics framework optimizes sequences for a target property. Starting from the latent embedding $z = \mathcal{E}(x)$

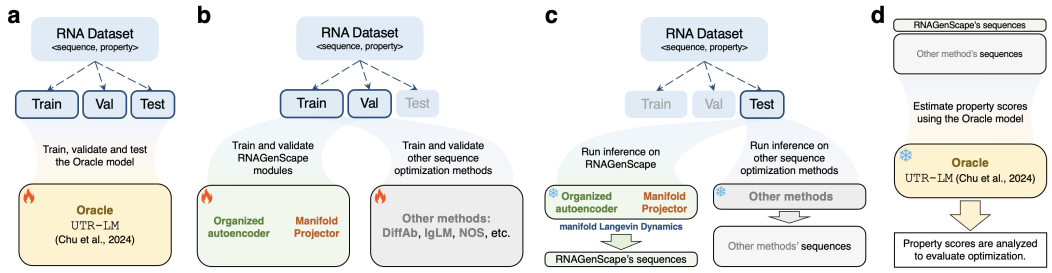

Figure 3: Overview of the experimental design. The same procedure is repeated for each of the three real-world mRNA datasets. **a,** We first fine-tune a pre-trained oracle model to predict target properties from mRNA sequences and verify its performance on a held-out test set. **b,** We then train each sequence optimization method using the training and validation sets. **c,** The trained methods are applied to optimize mRNA sequences starting from unseen test samples. **d,** Finally, the oracle model is used to evaluate property improvement and optimization success rate.

of a sequence x, we iteratively update it using a gradient-based drift term $\nabla_z f(z)$, inject Gaussian noise ϵ, and apply a manifold projector $\Psi(\cdot)$ to ensure biological viability and interpretability. We define the update rule as described in Eqn (3).

$$z_{t+1} = \Psi(z_t + \mathrm{d}z_t) \qquad \mathrm{d}z_t = \frac{\eta}{\tau}\nabla_z f(z_t) + \sqrt{2\eta}\,\epsilon_t, \quad \epsilon_t \sim \mathcal{N}(0, I) \tag{3}$$

Here, η is the step size, τ is the temperature hyperparameter, and $\nabla_z f(z)$ denotes the property gradient given by the predictor $\mathcal{P}$. A smaller τ emphasizes focused updates along the gradient, while a larger τ encourages more diverse exploration. When $\tau \to \infty$, the property guidance vanishes and the update rule is dominated by the stochastic term, which becomes similar to generative modeling with the walkback algorithm [39].

The manifold projector Ψ, analogous to the retraction in Riemannian SGD [34], is applied after each update to ensure that each step remains near the biologically valid latent manifold, enabling interpretable and controllable generation trajectories.

With the trained components $\mathcal{E}$, $\mathcal{P}$ and Ψ, we can optimize the target property of any given sequence. Notably, optimization entails both maximization and minimization: users can choose to increase or decrease the target property, depending on the application.

4 Empirical Results

We evaluate `RNAGenScape` on property-optimized mRNA sequence generation across multiple biologically relevant settings. The goal is to improve target properties such as translation efficiency and stability while preserving the biological viability of the generated sequences. To this end, we conducted comprehensive experiments to analyze the property gains, biological viability, efficiency, and optimization trajectories.

4.1 Experimental settings

Datasets and the properties to optimize We evaluated `RNAGenScape` on three mRNA datasets that capture diverse contexts and experimental designs. The optimization objectives are underlined.

1. `OpenVaccine` contains 2,400 mRNA sequences devised for COVID-19 mRNA vaccines, each with 107 nucleotides [40]. The target property is mRNA stability, which is directly relevant to vaccine performance.
2. `Zebrafish` consists of 5 subsets of zebrafish 5' UTR mRNAs each with 124 nucleotides, comprising more than 55,000 sequences in total [41]. The target property is translation efficiency, which measures how effectively an mRNA sequence is translated into protein.
3. `Ribosome` is a large-scale dataset of approximately 260,000 5' UTR mRNA sequences, each with 50 nucleotides [42]. The target property is the mean ribosome load, defined as the average number of ribosomes bound to the mRNA during translation.

Table 1: Our proposed RNAGenScape generates biologically viable mRNA sequences that achieve superior property optimization. $\mathcal{M}$ denotes whether the method takes the manifold structure into consideration. In property optimization results, Δ denotes the median change in property and % denotes success rate (percentage of mRNAs improved). In the biological viability checks, uORF OOF AUG denote out-of-frame start codon in the upstream open reading frame, and Kozak similarity is the similarity to the consensus Kozak sequence. The best values per metric are highlighted in bold.

Method	$\mathcal{M}$	Property Optimization				Biological Viability Checks		
		+Property		−Property		Initiation Interference	Initiation Context	Structural Stability
		$\Delta\uparrow$	%↑	$\Delta\downarrow$	%↑	uORF OOF AUG %↓	Kozak Similarity %↑	Minimum Free Energy↓
OpenVaccine (n≈2k), property to optimize: stability								
Data in the test set	—	—	—	—	—	1.65 ± 0.23	37.90 ± 30.38	−30.13 ± 17.76
DiffAb [43]	✗	+0.06	55.0	+0.11	44.0	1.75 ± 2.29	27.79 ± 30.03	−21.10 ± 12.46
IgLM [44]	✗	+0.24	63.1	+0.01	49.6	**1.24** ± 1.76	21.12 ± 28.62	−4.59 ± 3.87
NOS-C [45]	✗	+0.09	54.6	−2.25	90.6	1.71 ± 2.29	3.85 ± 10.72	−0.26 ± 0.83
NOS-D [45]	✗	+0.18	58.3	−0.29	65.4	2.27 ± 2.52	32.40 ± 30.06	−18.81 ± 5.82
MFM [46]	✓	+0.27	62.5	−0.03	50.8	1.86 ± 2.37	31.41 ± 30.77	−20.73 ± 10.88
gg-dWJS [47]	✓	+0.19	57.5	+0.03	49.7	1.69 ± 2.22	28.70 ± 31.18	−18.13 ± 9.65
RNAGenScape **(ours)**	✓	**+0.54**	**77.5**	**−2.81**	**97.9**	1.63 ± 0.59	**34.65** ± 23.26	**−25.78** ± 3.17
Gain	—	+125%	+22.8%	+24.9%	+8.1%	—	—	—
Zebrafish (n≈55k), property to optimize: translation efficiency								
Data in the test set	—	—	—	—	—	2.51 ± 2.65	33.92 ± 29.42	−29.40 ± 7.20
DiffAb [43]	✗	−0.21	36.3	−0.21	60.8	3.75 ± 2.65	23.84 ± 29.41	−27.31 ± 7.19
IgLM [44]	✗	−0.57	32.0	−0.83	74.3	2.90 ± 2.07	22.38 ± 28.15	−6.30 ± 3.16
NOS-C [45]	✗	+0.03	51.1	−1.07	80.8	2.70 ± 1.56	20.77 ± 18.50	−0.09 ± 0.39
NOS-D [45]	✗	+0.31	57.5	+0.38	40.2	2.43 ± 1.59	18.01 ± 20.02	−2.28 ± 2.42
MFM [46]	✓	−0.11	45.7	−0.72	74.7	2.93 ± 2.61	**34.05** ± 29.01	−26.19 ± 6.99
gg-dWJS [47]	✓	+0.21	56.3	−0.50	63.8	**2.41** ± 2.42	34.42 ± 27.42	−22.07 ± 6.48
RNAGenScape **(ours)**	✓	**+0.77**	**75.0**	**−1.29**	**85.1**	**2.41** ± 1.98	26.17 ± 17.43	**−29.24** ± 6.17
Gain	—	+148%	+30.4%	+20.6%	+5.3%	—	—	—
Ribosome (n≈260k), property to optimize: mean ribosome load								
Data in the test set	—	—	—	—	—	3.32 ± 4.25	20.60 ± 29.54	−8.65 ± 3.66
DiffAb [43]	✗	−0.05	45.0	−0.04	54.2	**3.16** ± 4.19	14.68 ± 28.77	−7.25 ± 3.57
IgLM [44]	✗	**+0.63**	80.5	−0.51	65.5	6.14 ± 4.85	7.73 ± 20.51	−7.27 ± 3.38
NOS-C [45]	✗	+0.08	53.6	+0.10	46.0	4.53 ± 3.89	12.96 ± 22.88	−5.79 ± 2.98
NOS-D [45]	✗	−0.09	46.5	−0.10	53.6	5.64 ± 5.06	8.05 ± 16.27	−0.77 ± 1.35
MFM [46]	✓	+0.15	54.5	−0.29	66.9	3.22 ± 4.28	16.19 ± 28.27	−8.22 ± 3.61
gg-dWJS [47]	✓	+0.42	63.6	−0.51	64.8	4.30 ± 4.79	13.61 ± 25.70	−9.13 ± 3.69
RNAGenScape **(ours)**	✓	**+0.63**	**81.4**	**−0.58**	**67.8**	4.51 ± 3.68	**16.41** ± 28.42	**−8.23** ± 3.18
Gain	—	+0.0%	+1.1%	+13.7%	+1.3%	—	—	—

Our main experiments follow the design in Figure 3. We train an oracle sequence-to-property model (panel a), train each optimization method using only the train and validation data (panel b), apply each method to improve sequences initialized from the unseen test set (panel c), and report gains and success rates using the oracle so all methods being compared are scored fairly (panel d).

Baselines For direct comparison on the datasets above, we benchmark RNAGenScape against six biological sequence optimization methods. DiffAb [43] performs sequence optimization using multinomial diffusion [48]. NOS-C [45] and NOS-D [45] perform property-guided discrete diffusion directly in sequence space using continuous-time and discrete-time transitions, respectively. IgLM [44] refines candidates using an autoregressive language model. MFM [46] generalizes conditional flow matching by replacing Euclidean straight-line interpolants with approximate geodesic interpolants induced by a data-dependent Riemannian metric, encouraging generated trajectories to remain close to the data manifold. gg-dWJS [47] follows gradient guidance from a discriminative property model, using discrete Markov chain to sample on a smoothed data manifold before jumping back to the discrete sequence space through one-step denoising. All these models are trained from scratch on each of the three RNA datasets with the correct interfaces with RNA A/U/G/C nucleotides (Figure 3b), and then evaluated for mRNA sequence optimization (Figure 3c-d).

Reproducibility All experiments were repeated using 5 random seeds and the average results are reported. Experimental details are summarized in Supplementary Section B.

4.2 RNAGenScape generates biologically viable, property-optimized mRNA sequences

Across all three datasets, we evaluate RNAGenScape along two complementary axes: improvement of the target property and preservation of biological viability. Both aspects are jointly reported in Table 1. Property optimization is assessed using UTR-LM [49], a widely adopted mRNA language

model, as the external oracle. UTR-LM is first pre-trained on over 480,000 5' UTR sequences from diverse sources and subsequently fine-tuned for property prediction on each dataset (Figure 3a). After fine-tuning, the model is frozen and **never accessed by any method** during their own training and inference (Figure 3b-c), ensuring an unbiased evaluation of the optimized sequences (Figure 3d).

In addition to analyzing the core property optimization performance, we apply a set of complementary checks grounded in established principles of mRNA translation to assess the biological viability of optimized sequences. Specifically, we consider three classes of viability metrics. First, upstream out-of-frame start codons (uORF OOF `AUG`) are known to suppress downstream protein expression and are therefore undesirable [50]. Second, Kozak similarity [51] measures how closely the initiation context matches the consensus Kozak sequence, with higher values indicating more favorable translation initiation. Third, minimum free energy [52] serves as a proxy for mRNA structural stability.

Across all three datasets and both optimization directions, `RNAGenScape` achieves the largest median property change and the highest success rate (Table 1). Compared to state-of-the-art biological sequence optimization methods, including DiffAb, IgLM, NOS-C, NOS-D, MFM, and gg-dWJS, `RNAGenScape` attains up to a 148% increase in median property improvement and up to a 30% increase in success rate. In addition to better optimization performance, `RNAGenScape` generally produces sequences with relatively few aberrant upstream start codons, higher Kozak similarity, and more favorable minimum free energy profiles. Moreover, the viability metrics of `RNAGenScape`-generated sequences closely match those of real sequences in the unseen test set, indicating that substantial property gains can be achieved without compromising biological viability.

4.3 Held-out validation on unseen near-optimal sequences

Table 2: Edit distance between each generated sequence and its nearest held-out sequence with near-optimal property. Edit distance is normalized to $[0, 1]$, with lower values being better.

	DiffAb [43]	IgLM [44]	NOS-C [45]	NOS-D [45]	MFM [46]	gg-dWJS [47]	`RNAGenScape` **(ours)**
`OpenVaccine` (n≈2k)	0.434 ± 0.086	0.533 ± 0.045	0.651 ± 0.037	0.543 ± 0.031	0.377 ± 0.086	0.422 ± 0.031	$\mathbf{0.356} \pm 0.025$
`Zebrafish` (n≈55k)	0.658 ± 0.076	0.690 ± 0.020	0.826 ± 0.021	0.783 ± 0.023	0.569 ± 0.127	0.614 ± 0.019	$\mathbf{0.484} \pm 0.014$
`Ribosome` (n≈260k)	0.418 ± 0.193	0.503 ± 0.024	0.526 ± 0.026	0.652 ± 0.032	0.413 ± 0.191	0.503 ± 0.025	$\mathbf{0.262} \pm 0.134$

To further assess whether `RNAGenScape` produces realistic property-optimized solutions, we conducted a held-out validation designed to mimic a practical design scenario. Specifically, for each dataset, we withheld a subset of sequences that already exhibit near-optimal target properties and excluded them from both the training and optimization stages. The property threshold is set at one standard deviation better than the mean. We then generate sequences from lower-property starting data and evaluate how close the generated sequences are to these held-out high-property references.

Table 2 reports the normalized edit distance between each generated sequence and its nearest held-out sequence with a near-optimal property. Across all datasets, `RNAGenScape` consistently achieves the lowest edit distance compared to alternative biological sequence generation methods, indicating that the sequences it produces are closer to high-performing real sequences. These results suggest that `RNAGenScape` not only improves target properties, but does so in a manner that recovers solutions better aligned with experimentally observed sequences.

4.4 Ablation studies

Organized autoencoder Since the optimization process in `RNAGenScape` is performed in the learned latent manifold, we verify that the OAE provides both accurate reconstruction and property prediction (Table 3). Additionally, we demonstrate that the reconstruction and prediction objectives ($\mathcal{L}_{\text{Recon}}$ and $\mathcal{L}_{\text{Pred}}$) are constructive to each other, which justifies our OAE design.

Manifold projector Next, we show that the manifold projector Ψ is essential for the optimization performance of `RNAGenScape` (Table 4). Without Ψ, optimization becomes substantially less effective across datasets. This behavior is expected: updating latent representations solely by following the property gradient allows trajectories to drift away from the learned data manifold, entering regions

Table 3: Reconstruction and prediction objectives in organized autoencoder are constructive and complementary.

	$\mathcal{L}_{\text{Recon}}$	$\mathcal{L}_{\text{Pred}}$	Reconstruction	Property Prediction	
			Nucleotide Acc. ↑	Pearson Corr. ↑	Spearman Corr. ↑
OpenVaccine (n≈2k)	✓	✗	0.49	-0.08	-0.08
	✗	✓	0.17	0.75	0.75
	✓	✓	0.67	0.70	0.71
Zebrafish (n≈55k)	✓	✗	0.80	-0.04	-0.03
	✗	✓	0.17	0.54	0.60
	✓	✓	0.78	0.63	0.67
Ribosome (n≈260k)	✓	✗	0.99	-0.08	-0.15
	✗	✓	0.11	0.87	0.85
	✓	✓	0.99	0.87	0.85

Table 4: The manifold projector Ψ substantially improves property optimization by enforcing optimization along the data manifold.

	Ψ	+property		−property	
		Δ ↑	% ↑	Δ ↓	% ↑
OpenVaccine (n≈2k)	✗	−0.44	33.3	−0.49	66.5
	✓	**0.54**	**77.5**	**−2.81**	**97.9**
Zebrafish (n≈55k)	✗	0.15	54.7	−0.83	71.9
	✓	**0.77**	**75.0**	**−1.29**	**85.1**
Ribosome (n≈260k)	✗	−0.17	45.1	0.13	47.0
	✓	**0.63**	**81.4**	**−0.58**	**67.8**

that are weakly supported by real sequences. In these off-manifold regions, property gradients become less reliable, leading to unstable updates and diminished optimization gains.

Optimization steps We further analyze how sensitive RNAGenScape is to the number of optimization steps. The results in Figure 4 show that our method is able to converge quickly and remains stable over a range of optimization steps during Langevin dynamics updates.

***De novo* generative methods and classic optimization algorithms** In addition to state-of-the-art biological sequence optimization methods, we compare RNAGenScape with two different types of methods. The first type is *de novo* generative modeling approaches, namely variational autoencoder (VAE) [53], denoising diffusion probabilistic model (DDPM) [54], latent diffusion model (LDM) [55], and flow matching (FM) [56]. The second type are classic optimization algorithms that operate in our OAE latent manifold, including gradient ascent [57, 58], Markov chain Monte Carlo (MCMC) [59, 60], and hill climbing [61]. All classic optimization baselines are GPU-compatible re-implementations of Castro et al. [62].

The results in Table S1 show that (1) *de novo* generative models exhibit limited property control and (2) classic optimization strategies in the OAE latent manifold also underperform RNAGenScape, indicating that our manifold Langevin dynamics contributed substantially to the favorable performance.

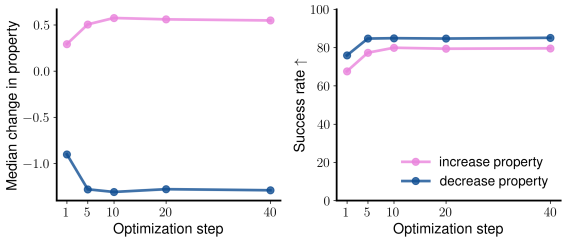

Figure 4: RNAGenScape optimization is step-efficient and remains stable over a range of optimization steps.

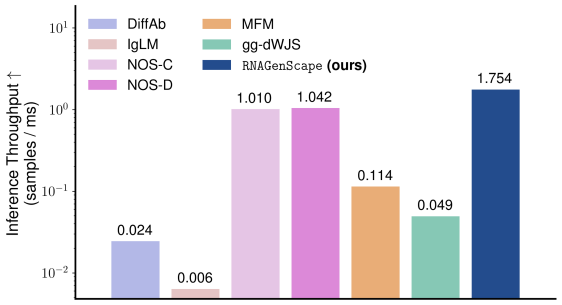

Figure 5: Inference throughput of all optimization methods compared.

4.5 Efficiency and scalability

In addition to its other merits, RNAGenScape is also highly efficient at inference time. As reported in Figure 5, it achieves a throughput of 1.754 sample/ms, more than 1.6× faster than the most efficient biological sequence optimization method being compared. This efficiency makes RNAGenScape well suited for large-scale or iterative design workflows where high throughput is essential.

4.6 Case study: RNAGenScape produces structured, data-aligned optimization trajectories

To illustrate how RNAGenScape iteratively navigates along a learned latent manifold, we visualize a sample optimization run in Figure 6a. The trajectory exhibits monotonic increases in the target

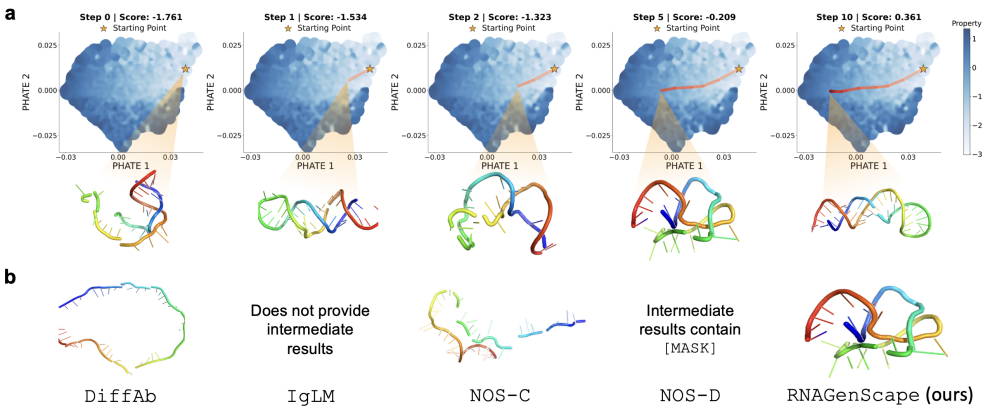

Figure 6: A sample latent space trajectory of RNAGenScape over 10 optimization steps. The trajectory follows a smooth path with steady improvement in the target property. **a,** The trajectory in the PHATE space and the corresponding 3D structures. **b,** Intermediate results of RNAGenScape and competing methods midway through property optimization (at the 5th step out of 10 steps).

property, while remaining near regions populated by real sequences. See Supplementary Section F for more examples. These trajectories are direct consequences of the manifold-constrained dynamics, which guided each step toward high-property regions while staying on the manifold.

As a qualitative illustration of biological plausibility, we show that the intermediate results of RNAGenScape optimization can be properly folded by RhoFold [63] into 3D structures (Figure 6b). In contrast, folding intermediate products of several other methods causes failure, as indicated by fragmented structures (Figure 6b).

5 Related Works

Machine learning is becoming increasingly popular for optimizing biological sequences such as DNA, RNA, and proteins. This section reviews recent advances in sequence modeling and optimization, with an emphasis on mRNAs.

De novo generative models excel at creating novel sequences, but fundamentally operate by generating from scratch rather than refining existing functional sequences [64–69]. Classic optimization strategies [57–61] are capable of improving known sequences, but typically lack mechanisms to ensure that intermediate variants remain consistent with the underlying biological distribution. More recent deep learning methods for sequence generation and optimization [43–45] aim to combine generative modeling with property-driven objectives, but their optimization trajectories remain opaque, offering limited interpretability of which sequence changes drive functional improvements. Most closely related to our geometric perspective, recent manifold-aware generative frameworks [46, 47] account for data geometry, but they focus on population-level transport or guided discrete sampling, rather than stepwise property optimization of individual sequences. Consequently, a framework that enables biologically-grounded sequence engineering remains needed [70].

An extended related works section can be found in Supplementary Section G.

6 Conclusion

We introduced RNAGenScape, a manifold Langevin dynamics framework for property-optimized mRNA sequence generation. Rather than optimizing in the ambient sequence space, RNAGenScape operates directly on a learned manifold of viable mRNA sequences. In this setting, property gradients are more reliable, biological viability is better preserved, and optimization can be performed with lower computational overhead. Comprehensive experiments on three real-world mRNA datasets show that our method leads to greater property gains and higher success rates under realistic data regimes. With this work, we aim to shift the paradigm of biological sequence design from unconstrained generation toward guided optimization, and to highlight mRNA sequence design as a critical yet understudied frontier in computational biology.

Acknowledgements

S.K. is funded by the NIH (NIGMSR01GM135929, R01GM130847), NSF CAREER award IIS-2047856, NSF IIS-2403317, NSF DMS-2327211 and NSF CISE-2403317. S.K is also funded by the Sloan Fellowship FG-2021-15883, the Novo Nordisk grant GR112933. A.J.G. is funded by the NIH (R01HD100035, R35GM122580).

References

[1] Norbert Pardi, Michael J Hogan, Frederick W Porter, and Drew Weissman. mrna vaccines—a new era in vaccinology. *Nature reviews Drug discovery*, 17(4):261–279, 2018.

[2] Namit Chaudhary, Drew Weissman, and Kathryn A Whitehead. mrna vaccines for infectious diseases: principles, delivery and clinical translation. *Nature reviews Drug discovery*, 20(11):817–838, 2021.

[3] Shugang Qin, Xiaoshan Tang, Yuting Chen, Kepan Chen, Na Fan, Wen Xiao, Qian Zheng, Guohong Li, Yuqing Teng, Min Wu, et al. mrna-based therapeutics: powerful and versatile tools to combat diseases. *Signal transduction and targeted therapy*, 7(1):166, 2022.

[4] Theofanis Vavilis, Eleni Stamoula, Alexandra Ainatzoglou, Athanasios Sachinidis, Malamatenia Lamprinou, Ioannis Dardalas, and Ioannis S Vizirianakis. mrna in the context of protein replacement therapy. *Pharmaceutics*, 15(1):166, 2023.

[5] Sebastian Castillo-Hair, Stephen Fedak, Ban Wang, Johannes Linder, Kyle Havens, Michael Certo, and Georg Seelig. Optimizing 5'utrs for mrna-delivered gene editing using deep learning. *Nature Communications*, 15(1):5284, 2024.

[6] Qi Ma, Xiaoguang Zhang, Jing Yang, Hongxia Li, Yanzhe Hao, and Xia Feng. Optimization of the 5' untranslated region of mrna vaccines. *Scientific Reports*, 14(1):19845, 2024.

[7] Ting Li, Gangfeng Liu, Guolong Bu, Yien Xu, Caiyun He, and Gexin Zhao. Optimizing mrna translation efficiency through rational 5' utr and 3' utr combinatorial design. *Gene*, 942:149254, 2025.

[8] Sebastian M Castillo-Hair and Georg Seelig. Machine learning for designing next-generation mrna therapeutics. *Accounts of chemical research*, 55(1):24–34, 2021.

[9] Niels Schlusser, Asier González, Muskan Pandey, and Mihaela Zavolan. Current limitations in predicting mrna translation with deep learning models. *Genome Biology*, 25(1):227, 2024.

[10] Francesco Calvanese, Camille N Lambert, Philippe Nghe, Francesco Zamponi, and Martin Weigt. Towards parsimonious generative modeling of rna families. *Nucleic Acids Research*, 52(10):5465–5477, 2024.

[11] He Zhang, Liang Zhang, Ang Lin, Congcong Xu, Ziyu Li, Kaibo Liu, Boxiang Liu, Xiaopin Ma, Fanfan Zhao, Huiling Jiang, et al. Algorithm for optimized mrna design improves stability and immunogenicity. *Nature*, 621(7978):396–403, 2023.

[12] Oskar Taubert, Fabrice von der Lehr, Alina Bazarova, Christian Faber, Philipp Knechtges, Marie Weiel, Charlotte Debus, Daniel Coquelin, Achim Basermann, Achim Streit, et al. Rna contact prediction by data efficient deep learning. *Communications biology*, 6(1):913, 2023.

[13] Muhammad Nabeel Asim, Muhammad Ali Ibrahim, Tayyaba Asif, and Andreas Dengel. Rna sequence analysis landscape: A comprehensive review of task types, databases, datasets, word embedding methods, and language models. *Heliyon*, 11(2), 2025.

[14] Donny D Licatalosi and Robert B Darnell. Resolving rna complexity to decipher regulatory rules governing biological networks. *Nature Reviews. Genetics*, 11(1):75, 2010.

[15] Caleb Weinreb, Adam J Riesselman, John B Ingraham, Torsten Gross, Chris Sander, and Debora S Marks. 3d rna and functional interactions from evolutionary couplings. *Cell*, 165(4):963–975, 2016.

[16] Ofir Lindenbaum, Jay S. Stanley III, Guy Wolf, and Smita Krishnaswamy. Geometry-based data generation, 2018.

[17] Lawrence Cayton et al. *Algorithms for manifold learning*. eScholarship, University of California, 2008.

[18] Hariharan Narayanan and Sanjoy Mitter. Sample complexity of testing the manifold hypothesis. *Advances in neural information processing systems*, 23, 2010.

[19] Charles Fefferman, Sanjoy Mitter, and Hariharan Narayanan. Testing the manifold hypothesis. *Journal of the American Mathematical Society*, 29(4):983–1049, 2016.

[20] Tianhong Li and Kaiming He. Back to basics: Let denoising generative models denoise. *arXiv preprint arXiv:2511.13720*, 2025.

[21] David Van Dijk, Roshan Sharma, Juozas Nainys, Kristina Yim, Pooja Kathail, Ambrose J Carr, Cassandra Burdziak, Kevin R Moon, Christine L Chaffer, Diwakar Pattabiraman, et al. Recovering gene interactions from single-cell data using data diffusion. *Cell*, 174(3):716–729, 2018.

[22] Kevin R Moon, David van Dijk, Zheng Wang, Scott Gigante, Daniel B Burkhardt, William S Chen, Kristina Yim, Antonia van den Elzen, Matthew J Hirn, Ronald R Coifman, et al. Visualizing structure and transitions in high-dimensional biological data. *Nature biotechnology*, 37(12):1482–1492, 2019.

[23] Daniel B Burkhardt, Jay S Stanley III, Alexander Tong, Ana Luisa Perdigoto, Scott A Gigante, Kevan C Herold, Guy Wolf, Antonio J Giraldez, David van Dijk, and Smita Krishnaswamy. Quantifying the effect of experimental perturbations at single-cell resolution. *Nature biotechnology*, 39(5):619–629, 2021.

[24] Chen Liu, Matthew Amodio, Liangbo L. Shen, Feng Gao, Arman Avesta, Sanjay Aneja, Jay C. Wang, Lucian V. Del Priore, and Smita Krishnaswamy. CUTS: A Deep Learning and Topological Framework for Multigranular Unsupervised Medical Image Segmentation . In *proceedings of Medical Image Computing and Computer Assisted Intervention – MICCAI 2024*, volume LNCS 15008. Springer Nature Switzerland, October 2024.

[25] Danqi Liao, Chen Liu, Benjamin W Christensen, Alexander Tong, Guillaume Huguet, Guy Wolf, Maximilian Nickel, Ian Adelstein, and Smita Krishnaswamy. Assessing neural network representations during training using noise-resilient diffusion spectral entropy. In *2024 58th Annual Conference on Information Sciences and Systems (CISS)*, pages 1–6. IEEE, 2024.

[26] Chen Liu, Danqi Liao, Alejandro Parada-Mayorga, Alejandro Ribeiro, Marcello DiStasio, and Smita Krishnaswamy. Diffkillr: Killing and recreating diffeomorphisms for cell annotation in dense microscopy images. In *ICASSP 2025-2025 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2025.

[27] Chen Liu, Ke Xu, Liangbo L Shen, Guillaume Huguet, Zilong Wang, Alexander Tong, Danilo Bzdok, Jay Stewart, Jay C Wang, Lucian V Del Priore, and Smita Krishnaswamy. Imageflownet: Forecasting multiscale trajectories of disease progression with irregularly-sampled longitudinal medical images. In *ICASSP 2025-2025 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2025.

[28] Xingzhi Sun, Danqi Liao, Kincaid MacDonald, Yanlei Zhang, Chen Liu, Guillaume Huguet, Guy Wolf, Ian Adelstein, Tim GJ Rudner, and Smita Krishnaswamy. Geometry-aware generative autoencoders for warped riemannian metric learning and generative modeling on data manifolds. In *International Conference on Artificial Intelligence and Statistics*. PMLR, 2025.

[29] Alexander Tong, Manik Kuchroo, Shabarni Gupta, Aarthi Venkat, Beatriz P San Juan, Laura Rangel, Brandon Zhu, John G Lock, Christine L Chaffer, and Smita Krishnaswamy. Learning transcriptional and regulatory dynamics driving cancer cell plasticity using neural ode-based optimal transport. *bioRxiv*, pages 2023–03, 2023.

[30] Yanlei Zhang, Guillaume Huguet, Edward De Brouwer, Danqi Liao, Oluwadamilola Fasina, Alexander Tong, Ricky TQ Chen, Guy Wolf, Maximilian Nickel, Ian Adelstein, et al. Neural fim: Bridging statistical manifolds and generative modeling through fisher geometry. *Authorea Preprints*, 2025.

[31] Salah Rifai, Yann N Dauphin, Pascal Vincent, Yoshua Bengio, and Xavier Muller. The manifold tangent classifier. *Advances in neural information processing systems*, 24, 2011.

[32] Qian Li, Yuxiao Hu, Ye Liu, Dongxiao Zhang, Xin Jin, and Yuntian Chen. Discrete point-wise attack is not enough: Generalized manifold adversarial attack for face recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 20575–20584, 2023.

[33] Yutong He, Naoki Murata, Chieh-Hsin Lai, Yuhta Takida, Toshimitsu Uesaka, Dongjun Kim, Wei-Hsiang Liao, Yuki Mitsufuji, J Zico Kolter, Ruslan Salakhutdinov, et al. Manifold preserving guided diffusion. *arXiv preprint arXiv:2311.16424*, 2023.

[34] Silvere Bonnabel. Stochastic gradient descent on riemannian manifolds. *IEEE Transactions on Automatic Control*, 58(9):2217–2229, September 2013.

[35] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. *Advances in neural information processing systems*, 32, 2019.

[36] Patrick Kidger, James Foster, Xuechen Li, and Terry J Lyons. Neural sdes as infinite-dimensional gans. In *International conference on machine learning*, pages 5453–5463. PMLR, 2021.

[37] Geoffrey E Hinton and Ruslan R Salakhutdinov. Reducing the dimensionality of data with neural networks. *science*, 313(5786):504–507, 2006.

[38] Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th international conference on Machine learning*, pages 1096–1103, 2008.

[39] Yoshua Bengio, Li Yao, Guillaume Alain, and Pascal Vincent. Generalized denoising auto-encoders as generative models, 2013.

[40] Rhiju Das, H Wayment-Steele, Christian Choe Do Soon Kim, Bojan Tunguz, Walter Reade, and Maggie Demkin. Openvaccine: Covid-19 mrna vaccine degradation prediction. *URL https://kaggle.com/competitions/stanford-covid-vaccine*, 2020.

[41] Ethan C Strayer, Srikar Krishna, Haejeong Lee, Charles Vejnar, Nils Neuenkirchen, Amit Gupta, Jean-Denis Beaudoin, and Antonio J Giraldez. Nap-trap reveals the regulatory grammar in 5'utr-mediated translation regulation during zebrafish development. *Nature Communications*, 15(1):10898, 2024.

[42] Paul J Sample, Ban Wang, David W Reid, Vlad Presnyak, Iain J McFadyen, David R Morris, and Georg Seelig. Human 5' utr design and variant effect prediction from a massively parallel translation assay. *Nature biotechnology*, 37(7):803–809, 2019.

[43] Shitong Luo, Yufeng Su, Xingang Peng, Sheng Wang, Jian Peng, and Jianzhu Ma. Antigen-specific antibody design and optimization with diffusion-based generative models for protein structures. *Advances in Neural Information Processing Systems*, 35:9754–9767, 2022.

[44] Richard W Shuai, Jeffrey A Ruffolo, and Jeffrey J Gray. Iglm: Infilling language modeling for antibody sequence design. *Cell Systems*, 14(11):979–989, 2023.

[45] Nate Gruver, Samuel Stanton, Nathan Frey, Tim GJ Rudner, Isidro Hotzel, Julien Lafrance-Vanasse, Arvind Rajpal, Kyunghyun Cho, and Andrew G Wilson. Protein design with guided discrete diffusion. *Advances in neural information processing systems*, 36:12489–12517, 2023.

[46] Kacper Kapuśniak, Peter Potaptchik, Teodora Reu, Leo Zhang, Alexander Tong, Michael Bronstein, Avishek J Bose, and Francesco Di Giovanni. Metric flow matching for smooth interpolations on the data manifold. *Advances in Neural Information Processing Systems*, 37:135011–135042, 2024.

[47] Zarif Ikram, Dianbo Liu, and M Saifur Rahman. Gradient-guided discrete walk-jump sampling for biological sequence generation. *Transactions on Machine Learning Research*, 2024.

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

[49] Yanyi Chu, Dan Yu, Yupeng Li, Kaixuan Huang, Yue Shen, Le Cong, Jason Zhang, and Mengdi Wang. A 5útr language model for decoding untranslated regions of mrna and function predictions. *Nature Machine Intelligence*, 6(4):449–460, 2024.

[50] Thomas E Dever, Ivaylo P Ivanov, and Alan G Hinnebusch. Translational regulation by uorfs and start codon selection stringency. *Genes & development*, 37(11-12):474–489, 2023.

[51] Alec C Gleason, Ghanashyam Ghadge, Jin Chen, Yoshifumi Sonobe, and Raymond P Roos. Machine learning predicts translation initiation sites in neurologic diseases with nucleotide repeat expansions. *PLoS One*, 17(6):e0256411, 2022.

[52] Ronny Lorenz, Stephan H Bernhart, Christian Höner zu Siederdissen, Hakim Tafer, Christoph Flamm, Peter F Stadler, and Ivo L Hofacker. Viennarna package 2.0. *Algorithms for molecular biology*, 6(1):26, 2011.

[53] Diederik P Kingma, Max Welling, et al. Auto-encoding variational bayes, 2013.

[54] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

[55] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022.

[56] Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.

[57] Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3):229–256, 1992.

[58] Martin Zinkevich. Online convex programming and generalized infinitesimal gradient ascent. In *Proceedings of the 20th international conference on machine learning (icml-03)*, pages 928–936, 2003.

[59] Stephen Brooks. Markov chain monte carlo method and its application. *Journal of the royal statistical society: series D (the Statistician)*, 47(1):69–100, 1998.

[60] Christophe Andrieu, Nando De Freitas, Arnaud Doucet, and Michael I Jordan. An introduction to mcmc for machine learning. *Machine learning*, 50:5–43, 2003.

[61] Bart Selman and Carla P Gomes. Hill-climbing search. *Encyclopedia of cognitive science*, 81(333-335):10, 2006.

[62] Egbert Castro, Abhinav Godavarthi, Julian Rubinfien, Kevin Givechian, Dhananjay Bhaskar, and Smita Krishnaswamy. Transformer-based protein generation with regularized latent space optimization. *Nature Machine Intelligence*, 4(10):840–851, 2022.

[63] Tao Shen, Zhihang Hu, Siqi Sun, Di Liu, Felix Wong, Jiuming Wang, Jiayang Chen, Yixuan Wang, Liang Hong, Jin Xiao, et al. Accurate rna 3d structure prediction using a language model-based deep learning approach. *Nature Methods*, 21(12):2287–2298, 2024.

[64] Oleksii Prykhodko, Simon Viet Johansson, Panagiotis-Christos Kotsias, Josep Arús-Pous, Esben Jannik Bjerrum, Ola Engkvist, and Hongming Chen. A de novo molecular generation method using latent vector based generative adversarial network. *Journal of cheminformatics*, 11(1):74, 2019.

[65] Oscar Méndez-Lucio, Benoit Baillif, Djork-Arné Clevert, David Rouquié, and Joerg Wichard. De novo generation of hit-like molecules from gene expression signatures using artificial intelligence. *Nature communications*, 11(1):10, 2020.

[66] Justas Dauparas, Ivan Anishchenko, Nathaniel Bennett, Hua Bai, Robert J Ragotte, Lukas F Milles, Basile IM Wicky, Alexis Courbet, Rob J de Haas, Neville Bethel, et al. Robust deep learning–based protein sequence design using proteinmpnn. *Science*, 378(6615):49–56, 2022.

[67] Zachary Wu, Kadina E Johnston, Frances H Arnold, and Kevin K Yang. Protein sequence design with deep generative models. *Current opinion in chemical biology*, 65:18–27, 2021.

[68] Ali Madani, Ben Krause, Eric R Greene, Subu Subramanian, Benjamin P Mohr, James M Holton, Jose Luis Olmos Jr, Caiming Xiong, Zachary Z Sun, Richard Socher, et al. Large language models generate functional protein sequences across diverse families. *Nature biotechnology*, 41(8):1099–1106, 2023.

[69] Joseph L Watson, David Juergens, Nathaniel R Bennett, Brian L Trippe, Jason Yim, Helen E Eisenach, Woody Ahern, Andrew J Borst, Robert J Ragotte, Lukas F Milles, et al. De novo design of protein structure and function with rfdiffusion. *Nature*, 620(7976):1089–1100, 2023.

[70] Zachary Wu, SB Jennifer Kan, Russell D Lewis, Bruce J Wittmann, and Frances H Arnold. Machine learning-assisted directed protein evolution with combinatorial libraries. *Proceedings of the National Academy of Sciences*, 116(18):8852–8858, 2019.

[71] Stephen G Oliver. From dna sequence to biological function. *Nature*, 379(6566):597–600, 1996.

[72] Eeshit Dhaval Vaishnav, Carl G de Boer, Jennifer Molinet, Moran Yassour, Lin Fan, Xian Adiconis, Dawn A Thompson, Joshua Z Levin, Francisco A Cubillos, and Aviv Regev. The evolution, evolvability and engineering of gene regulatory dna. *Nature*, 603(7901):455–463, 2022.

[73] Yifan Chen, Zhenya Du, Xuanbai Ren, Chu Pan, Yangbin Zhu, Zhen Li, Tao Meng, and Xiaojun Yao. mrna-cla: An interpretable deep learning approach for predicting mrna subcellular localization. *Methods*, 227:17–26, 2024.

[74] Shujun He, Baizhen Gao, Rushant Sabnis, and Qing Sun. Rnadegformer: accurate prediction of mrna degradation at nucleotide resolution with deep learning. *Briefings in Bioinformatics*, 24(1):bbac581, 2023.

[75] Sam Sinai, Eric Kelsic, George M Church, and Martin A Nowak. Variational auto-encoding of protein sequences. *arXiv preprint arXiv:1712.03346*, 2017.

[76] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020.

[77] Johannes Linder, Nicholas Bogard, Alexander B Rosenberg, and Georg Seelig. Deep exploration networks for rapid engineering of functional dna sequences. *BioRxiv*, page 864363, 2019.

[78] Peter Eastman, Jade Shi, Bharath Ramsundar, and Vijay S Pande. Solving the rna design problem with reinforcement learning. *PLoS computational biology*, 14(6):e1006176, 2018.

[79] Tomasz Wirecki, Grzegorz Lach, Farhang Jaryani, Nagendar Goud Badepally, S Naeim Moafinejad, Gaja Klaudel, and Janusz M Bujnicki. Desirna: structure-based design of rna sequences with a monte carlo approach. *bioRxiv*, pages 2023–06, 2023.

[80] Johannes Linder and Georg Seelig. Fast activation maximization for molecular sequence design. *BMC bioinformatics*, 22:1–20, 2021.

[81] Samuel Stanton, Wesley Maddox, Nate Gruver, Phillip Maffettone, Emily Delaney, Peyton Greenside, and Andrew Gordon Wilson. Accelerating bayesian optimization for biological sequence design with denoising autoencoders. In *International conference on machine learning*, pages 20459–20478. PMLR, 2022.

[82] Kai Yi, Bingxin Zhou, Yiqing Shen, Pietro Liò, and Yuguang Wang. Graph denoising diffusion for inverse protein folding. *Advances in Neural Information Processing Systems*, 36:10238–10257, 2023.

[83] Kevin Bijan Givechian, João Felipe Rocha, Chen Liu, Edward Yang, Sidharth Tyagi, Kerrie Greene, Rex Ying, Etienne Caron, Akiko Iwasaki, and Smita Krishnaswamy. Immunostruct enables multimodal deep learning for immunogenicity prediction. *Nature Machine Intelligence*, pages 1–14, 2025.

[84] Arman Afrasiyabi, Jake Kovalic, Chen Liu, Egbert Castro, Alexis Weinreb, Erdem Varol, David Miller, Marc Hammarlund, and Smita Krishnaswamy. Cellsplicenet: Interpretable multimodal modeling of alternative splicing across neurons in c. elegans. *bioRxiv*, pages 2025–06, 2025.

[85] Hannah K Wayment-Steele, Wipapat Kladwang, Alexandra I Strom, Jeehyung Lee, Adrien Treuille, Alex Becka, Eterna Participants, and Rhiju Das. Rna secondary structure packages evaluated and improved by high-throughput experiments. *Nature methods*, 19(10):1234–1242, 2022.

Technical Appendices

Contents

A Pseudocode of the manifold projector

The pseudocode of the manifold projector is shown in Algorithm 1.

Algorithm 1 Manifold Projector Ψ

Input: Dataset $\mathcal{Z} = \{z_i\}_{i=1}^{N}$, denoising autoencoder (i.e., manifold projector) Ψ, noise levels $\{\sigma_1, \ldots, \sigma_K\}$, denoising steps K, learning rate η

for each z_i in minibatch $\{z_i\}_{i=1}^{B} \subset \mathcal{Z}$ **do**
  Initialize $\tilde{z}^{(0)} \leftarrow z_i$
  for $k = 1$ to K **do**
    $\tilde{z}^{(k)} \sim C(\tilde{\mathcal{Z}}|\tilde{\mathcal{Z}}^{(k-1)}, \sigma_k)$
    $\mathcal{L}^{(k)} = \|\Psi(\tilde{z}^{(k)}) - \tilde{z}^{(k-1)}\|_2^2$
  end for
  $\mathcal{L}_i = \sum_{k=1}^{K} \mathcal{L}^{(k)}$
  $\Psi \leftarrow \Psi - \eta \nabla_{\Psi} \left(\frac{1}{B} \sum_{i=1}^{B} \mathcal{L}_i \right)$
end for

B Hyperparameters and Architecture

Learning the manifold with SUGAR To learn the manifold with SUGAR, we used the k-NN mode for estimating degrees (and thus sparsity) of latent points. We employed an α-decay kernel with

$\alpha = 2$ and an adaptive bandwidth determined from the distance to the 5 nearest neighbors. The diffusion time was set to $t = 1$.

Training the organized autoencoder (OAE) We trained the OAE using the AdamW optimizer with an initial learning rate of 10^{-2}, together with a linear warmup cosine annealing scheduler. The learning rate was linearly increased from 10^{-4} (i.e., $0.01\times$ the base learning rate) to the target value during the first 10% of training epochs (warmup), and then annealed to zero following a cosine decay schedule over the remaining epochs. We used a batch size of 128, a maximum of 200 epochs, and early stopping with a patience of 20 epochs based on the validation loss.

Training the manifold projector We trained the manifold projector Ψ using the AdamW optimizer with a learning rate of 10^{-4}. We used a batch size of 256, a maximum of 200 epochs, and applied early stopping with a patience of 20 epochs based on the validation loss.

Table S1: Hyperparameters used for different datasets.

Dataset	λ_{Recon} / λ_{Pred}	Noise levels	Langevin (step size, temperature)
`Zebrafish`	5.0 / 1.0	{1.0, 0.8, 0.5}	1×10^{-4}, 8×2^{-4}
`OpenVaccine`	1.0 / 1.0	{1.0, 0.5}	1×10^{-2}, 1×10^{-2}
`Ribosome`	1.0 / 1.0	{0.3}	5×10^{-3}, 1×4^{-3}

Organized Autoencoder (OAE) The organized autoencoder (OAE) maps mRNA sequences $x \in \mathbb{R}^{L\times V}$ to a compact latent $z \in \mathbb{R}^d$. Our latent dimension is 320 across all datasets.

For encoder, we apply three 1D convolutional blocks with GroupNorm, GELU, and channel squeeze-excitation (SE), followed by adaptive average pooling to length 8 and a linear projection. The property head is a three-layer MLP with GELU and dropout rate set to 0.3.

For decoding, a progressive 1D decoder upsamples structure gradually: we first expand z to a 128×8 seed map, then apply a stack of UpsampleBlock modules composed of upsampling and two residual convolutional blocks until reaching $\geq L$ positions; we then refine the output with two residue convolutional blocks to produce the final predicted logits. Weights are Kaiming/Xavier initialized; GroupNorm scales are set to 1 and biases to 0.

mRNA sequence vocabulary Although mRNA sequences naturally consist of the nucleotides `A`, `U`, `G`, and `C`, some experimental datasets represent `U` as `T` (borrowing the DNA alphabet). To handle this heterogeneity consistently, we define a unified vocabulary of size $V = 7$: `<pad>`, `A`, `U`, `T`, `G`, `C`, and `N`. Here, `N` denotes an unknown nucleotide during sequencing, and both `U` and `T` tokens are retained to ensure compatibility across datasets. Note that each dataset uniquely uses either `U` or `T`, and since we are training the model separately on each dataset, we do not risk confusing the model with interchangeability during decoding.

Hardware The evaluations were performed on a single NVIDIA A100 GPU. However, `RNAGenScape` can be run efficiently on more modest hardware.

Baseline adaptation MFM [46] was originally designed for smooth interpolation between populations on a manifold rather than direct property optimization. To adapt MFM to our optimization setting, we partition the training data into two subpopulations according to their ground truth property values, as required by the MFM formulation. We then train MFM to model flows from the low-property subpopulation to the high-property subpopulation for property maximization, and in the reverse direction for property minimization. The trained MFM model is subsequently applied to the unseen test set for property optimization, following the same evaluation protocol used for the other methods.

The remaining property optimization baselines, including DiffAb [43], IgLM [44], NOS-C [45], NOS-D [45], and gg-dWJS [47], are natively designed for guided optimization and therefore do not require additional adaptation.

C Evaluation Metrics

Dataset partitioning For the `Ribosome` dataset, the original paper provides predefined training and test splits. We use the provided test split as the test set and further partition the predefined training split into training and validation sets using an 85%:15% ratio. For the `OpenVaccine` and `Zebrafish` datasets, which do not come with predefined splits, we randomly partition the data into train, validation, and test sets using a 65%:15%:20% ratio.

Property Optimization To quantify the effectiveness of property optimization of different models, we measure both the median property gain and the fraction of sequences that are successfully optimized.

Specifically, given a test set of mRNA sequences X_{test} with predicted properties $\mathcal{P}_{\text{oracle}}(X_{\text{test}})$ and their optimized counterparts $\tilde{X}_{\text{test}}$ with properties $\mathcal{P}_{\text{oracle}}(\tilde{X}_{\text{test}})$, we compute:

$$\Delta_{\text{median}} = \text{median}\left(\mathcal{P}_{\text{oracle}}(\tilde{x}) - \mathcal{P}_{\text{oracle}}(x)\right), \quad x \in X_{\text{test}}, \tag{4}$$

and

$$\%_{\text{success}} = \frac{1}{|X_{\text{test}}|} \sum_{x \in X_{\text{test}}} \mathbb{1}\left[\mathcal{P}_{\text{oracle}}(\tilde{x}) > \mathcal{P}_{\text{oracle}}(x)\right]. \tag{5}$$

Here, Δ_{median} measures the gain in the target property across the test set, while $\%_{\text{success}}$ reports the percentage of sequences that improve after optimization.

For models that cannot refine existing sequences (e.g., pure *de novo* generators), we assign a random pairing between initial and final sequences to enable a fair comparison.

The external oracle In our evaluation for property optimization, we use an external oracle $\mathcal{P}_{\text{oracle}}(x)$ to predict the property of sequences generated by `RNAGenScape` as well as competing methods. Specifically, we adopt UTR-LM [49], a widely used mRNA language model, as the property prediction oracle. It is first pre-trained on more than 480,000 5' UTR mRNA sequences from diverse sources, and then fine-tuned for property prediction on each of our datasets. After fine-tuning, it is frozen and never accessed during optimization, ensuring unbiased evaluation. Performance of the fine-tuned UTR-LM is shown in Table S2.

Table S2: The UTR-LM oracle for property evaluation.

	Property Prediction	
	Pearson Corr. ↑	Spearman Corr. ↑
`OpenVaccine (n≈2k)`	0.64	0.70
`Zebrafish (n≈55k)`	0.79	0.79
`Ribosome (n≈260k)`	0.91	0.89

D Additional comparison against de novo baselines

Figure S1: `RNAGenScape` achieves better property optimization than *de novo* generative methods and classic optimization algorithms that operate on our learned data manifold.

Method	Property Optimization			
	+Property		−Property	
	Δ ↑	% ↑	Δ ↓	% ↑
`OpenVaccine (n≈2k)`				
VAE [53]	−0.23	42.1	−0.23	57.9
DDPM [54]	−0.33	33.8	−0.33	66.2
LDM [55]	−0.07	47.5	−0.07	52.5
FM [56]	−0.34	32.5	−0.34	67.5
OAE + gradient ascent	−0.01	51.0	−0.19	61.3
OAE + MCMC	−0.11	45.1	−0.19	57.7
OAE + hill climbing	+0.40	69.2	−0.29	64.2
OAE + stochastic hill climbing	−0.12	43.5	−0.15	60.0
`RNAGenScape` **(ours)**	**+0.54**	**77.5**	**−2.81**	**97.9**
`Zebrafish (n≈55k)`				
VAE [53]	−0.01	49.4	−0.01	50.6
DDPM [54]	+0.29	58.2	+0.29	41.8
LDM [55]	−0.95	22.5	−0.95	77.5
FM [56]	+0.32	59.8	+0.32	40.2
OAE + gradient ascent	−0.21	18.0	−0.33	66.5
OAE + MCMC	−0.20	44.3	−0.37	60.1
OAE + hill climbing	+0.76	74.4	−0.87	75.2
OAE + stochastic hill climbing	+0.32	60.3	−0.80	73.7
`RNAGenScape` **(ours)**	**+0.77**	**75.0**	**−1.29**	**85.1**
`Ribosome (n≈260k)`				
VAE [53]	−0.24	41.7	−0.24	58.3
DDPM [54]	−0.11	45.9	−0.11	54.1
LDM [55]	−0.24	41.8	−0.24	58.2
FM [56]	−0.10	46.4	−0.10	53.6
OAE + gradient ascent	+0.43	73.6	−0.05	55.7
OAE + MCMC	−0.19	43.0	−0.10	54.2
OAE + hill climbing	+0.53	**83.0**	−0.06	56.1
OAE + stochastic hill climbing	+0.19	60.4	−0.05	55.2
`RNAGenScape` **(ours)**	**+0.63**	81.4	**−0.58**	**67.8**

E Interpolation between sequences

In addition to optimizing mRNA sequences, we can interpolate between two existing sequences by guiding the latent embedding of one sequence toward that of another. Specifically, given a source sequence x_{source} and a target sequence x_{target}, we first obtain their latent embeddings via the encoder: $z_{\text{source}} = \mathcal{E}(x_{\text{source}})$ and $z_{\text{target}} = \mathcal{E}(x_{\text{target}})$.

We then perform property-guided Manifold Langevin dynamics starting from $z = z_{\text{source}}$, with the drift term being the force term that drives the embedding toward z_{target}:

$$f_{\text{interp}}(z, z_{\text{target}}) = -\frac{z - z_{\text{target}}}{\|z - z_{\text{target}}\|_2} \tag{6}$$

In this case, we slightly modify the update rule of the Langevin dynamics (Eqn (3)) as follows:

$$\mathrm{d}z_t = \eta \left(\nabla_z f(z_t) + \lambda f_{\text{interp}}\right) + \sqrt{2\eta} \cdot \epsilon_t. \tag{7}$$

By setting the interpolation weight $\lambda > 0$ in Eqn (7), we add a directional bias that drives the latent trajectory toward the target point, while following the property gradients.

Results on interpolation Guided by a directional force toward a specified target, `RNAGenScape` generates smooth and coherent trajectories on the learned manifold while preserving biological plausibility and continuity (Figure S2). These trajectories connect arbitrary input-target sequence pairs in a structured manner, reflecting semantically meaningful transitions. The distances from each intermediate point to the source and target quantitatively demonstrate the monotonicity and smoothness of the interpolation (Figure S3).

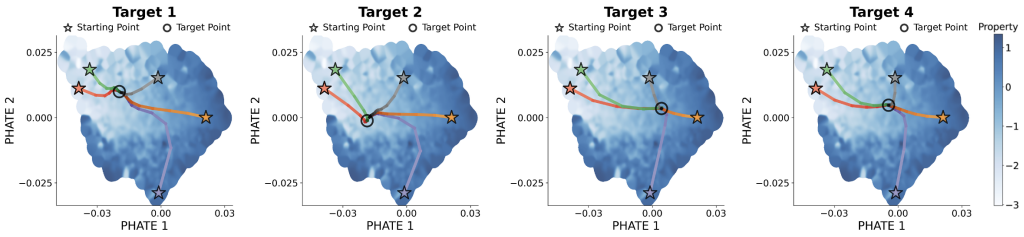

Figure S2: Latent space interpolation trajectories from 5 sources to 4 targets. Each trajectory is shown as a line fading from bright to dark in a consistent color. `RNAGenScape` produces smooth and coherent paths on the manifold between arbitrary input-target mRNA pairs.

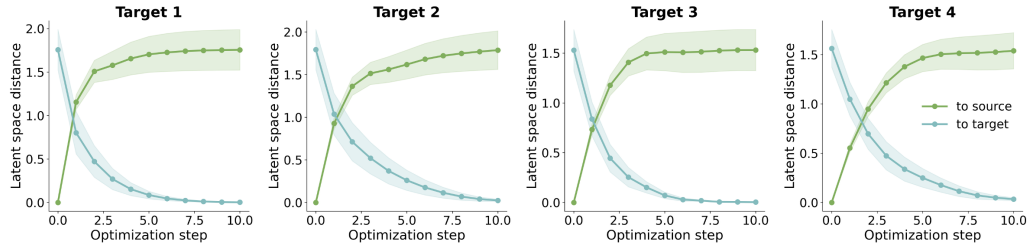

Figure S3: Latent space ℓ_2 distances during interpolation show smooth and monotonic transition from the source to the target. Results are averaged over all data samples.

F Additional optimization trajectories

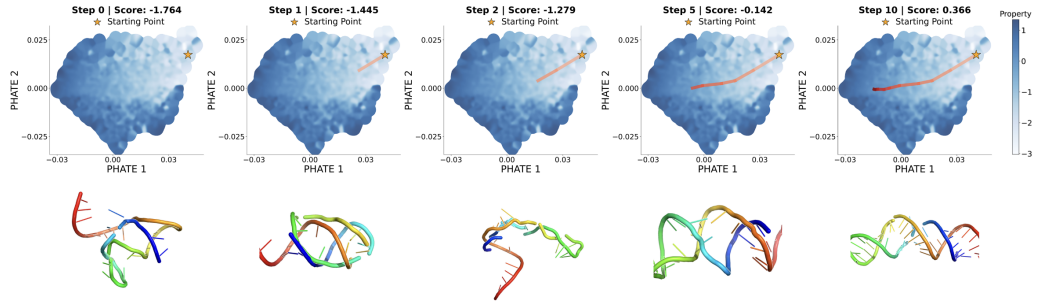

Figure S4: More examples of latent manifold trajectories. Trajectories in PHATE space and corresponding 3D structures are shown.

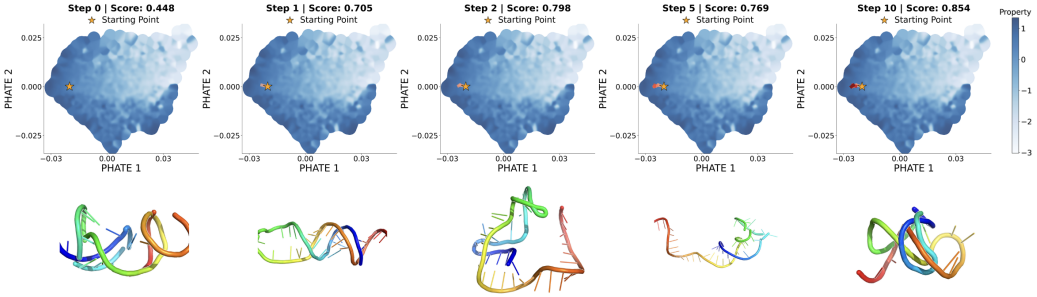

Figure S5: More examples of latent manifold trajectories. Trajectories in PHATE space and corresponding 3D structures are shown.

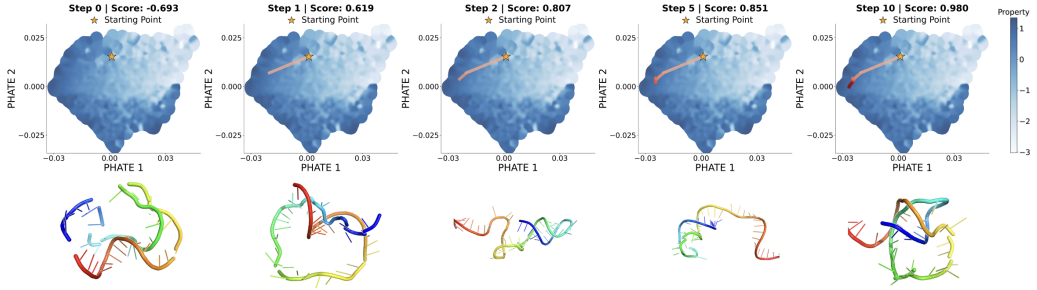

Figure S6: More examples of latent manifold trajectories. Trajectories in PHATE space and corresponding 3D structures are shown.

G Extended Related Works

Sequence-to-function modeling A central goal in biological sequence modeling is predicting quantitative properties (e.g., expression level, stability) directly from the sequence [71]. Recent deep learning models trained on high-throughput experimental data have demonstrated strong performance in this setting, particularly for regulatory regions such as 5' UTRs and promoters [42, 72]. Models such as ConvNets [73] and Transformers [74] have been used to capture complex dependencies in mRNA space, and form the basis for downstream prediction of properties.

Generative models for design Generative models enable sampling of novel sequences enriched for desired traits. Variational autoencoders (VAEs) [53] have been applied to proteins to learn smooth latent manifolds that are amenable to gradient-based optimization [75, 5]. ProteinMPNN [66], although described as a message-passing neural network by the authors, shares core design principles with autoencoders. Generative adversarial networks [76] such as Méndez-Lucio et al. [65] or ProteinGAN [67] and autoregressive language models such as ProGen [68] have also been used to generate diverse protein sequences. More recently, diffusion models [54] have shown promise in discrete domains. For example, RFdiffusion [69] generates proteins unconditionally or conditioned on structural constraints. These methods can be readily adapted to mRNA design.

Optimization of biological sequences Sequence optimization can be framed as a black-box search or a differentiable surrogate-guided process. Several approaches relax discrete inputs for gradient-based updates, such as using straight-through estimators [77]. ReLSO learns a continuous latent manifold and performs gradient ascent [62]. Others apply reinforcement learning [78] or Monte Carlo algorithm [79] for sequence optimization. Methods such as Fast SeqProp [80] and LaMBO [81] have demonstrated success in optimizing sequences under multi-objective constraints.

Integration of structural context While the present work strictly focuses on the mRNA sequence, many successful models incorporate inductive biases from the structures. ProteinMPNN [66] and diffusion-based inverse folding [82] condition sequence generation on 3D structures. ImmunoStruct [83] jointly models protein sequence, structure, and biochemical properties to predict immunogenicity. CellSpliceNet [84] integrates long-range sequence, local regions of interest, secondary structure, and gene expression to predict alternative slicing. EternaFold [85] incorporate predicted secondary structures to improve fitness prediction. Although in our work we did not incorporate mRNA structures, extending `RNAGenScape` to sequence-structure joint modeling and optimization could be a promising direction.

H Limitations and Future Work

One limitation of `RNAGenScape` is that it exclusively models and optimizes mRNA sequences only, without explicitly incorporating mRNA structure. While sequence-based representations capture many aspects of mRNA function, structures can provide auxiliary information in regulating translation, stability, and interactions.

In future work, we will study the possibility of performing sequence-structure joint modeling and optimization. Beyond mRNA, we plan to extend `RNAGenScape` to other modalities such as protein sequences and regulatory elements, and integrate active learning frameworks that guide wet lab experimentation. By grounding sequence optimization on the manifold of real data, we aim to provide a versatile platform for interpretable high-throughput design in synthetic biology.

I Impact Statement

This work proposes a machine learning framework for optimizing mRNA sequences while preserving biological viability by constraining optimization to a learned data manifold. Potential positive impacts include supporting computational design workflows in mRNA therapeutics and synthetic biology such as vaccine development by enabling more controlled and interpretable sequence refinement. As a general machine learning contribution, the framework may also inform broader research on manifold-constrained optimization for biological sequences.

The approach is intended as a computational tool to complement experimental and domain expertise rather than a substitute for empirical validation. As with other data-driven sequence models, limitations and biases in the available training data may influence optimization outcomes, and inappropriate use without biological oversight could lead to nonfunctional or misleading designs.

We do not anticipate direct negative societal consequences arising from this work beyond those already associated with machine learning-based biological modeling. On our GitHub repository, we will require that users adhere to usage guidelines and apply restrictions to access the model. We encourage future research to further study robustness, data bias, and safe integration of such methods into experimental pipelines.