RNA DESIGN VIA CONDITIONED FLOW MATCHING AND FINITE-POLICY REINFORCEMENT LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

RNA design aims to identify sequences that fold into specified secondary structures. Existing methods formulate the task as target-specific search or conditional generation. However, natural RNA evolution proceeds through sequence variation and selection, with compensatory substitutions, whereas these methods do not explicitly model this process. To address this limitation, we propose a two-stage framework comprising RNA Inverse-Folding Flow (**RNA-IFlow**) and **RNA-IFlow-RL**. RNA-IFlow uses structure-conditioned Dirichlet Flow Matching to model coordinated variation across the sequence, while RNA-IFlow-RL maps the learned flow to a pairing-preserving finite policy and refines it with thermodynamic feedback. Our framework achieves leading performance on multiple benchmarks, reaching 85.19% Pass@1 on Rfam-27. Further analyses reveal thermodynamic gains, policy dynamics, and robustness across settings. Our work couples coordinated variation with thermodynamic selection, offering a novel paradigm for RNA design.

1 INTRODUCTION

RNA is a programmable polymer of ribonucleotides that folds into hierarchical structures, where secondary structure captures the intramolecular base-pairing topology of the RNA chain (Lorenz et al., 2011; Zadeh et al., 2011). RNA design, also known as inverse folding, aims to identify nucleotide sequences that fold into a specified secondary structure and provides a computational route to design RNAs with desired structural properties for biotechnology and therapeutic applications (Andronescu et al., 2004; Zadeh et al., 2011; Ward et al., 2023; Li et al., 2025). Recent progress in AI-assisted mRNA therapeutics further highlights the broad potential of RNA design for applications such as personalized cancer vaccines and precision RNA therapeutics (Fieldhouse & Basu, 2026).

Existing methods mainly solve RNA design through target-specific search, learned optimization, or conditional generation (Li et al., 2025; Zhou et al., 2026; Gautam et al., 2026). Although these methods achieve substantial advances, they typically formulate RNA design as search policy optimization or structure-conditioned generation, which differs from the variation–selection process observed in natural RNA evolution. In this process, sequence variation explores alternative nucleotide configurations, while compensatory substitutions at paired sites can preserve base pairing as mutations accumulate (Chen et al., 1999; Dutheil et al., 2010). Selection then favors mutants that better satisfy structural and functional constraints, as illustrated in Figure 1(a). This observation suggests a natural modeling perspective for RNA design. We can first model structure-conditioned sequence variation, and then refine the resulting sequence distribution through thermodynamic selection.

However, modeling RNA design based on this perspective faces three fundamental challenges: (1) How can coordinated sequence variation be modeled over the complete RNA sequence rather than through independent or strictly sequential nucleotide decisions? (2) How can a selection process be introduced so that generated variants are progressively biased toward thermodynamically favorable folds? (3) How can compensatory substitutions at paired sites be preserved during variation, such that sequence changes remain consistent with the target secondary structure?

To address these challenges, we propose a two-stage framework comprising **RNA-IFlow** and **RNA-IFlow-RL**, which combines flow matching (FM) and reinforcement learning (RL) for RNA design (Figure 1). For the first challenge, RNA-IFlow applies the structure-conditioned Dirichlet FM on a bidirectional masked language model backbone to model coordinated variation over the complete

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

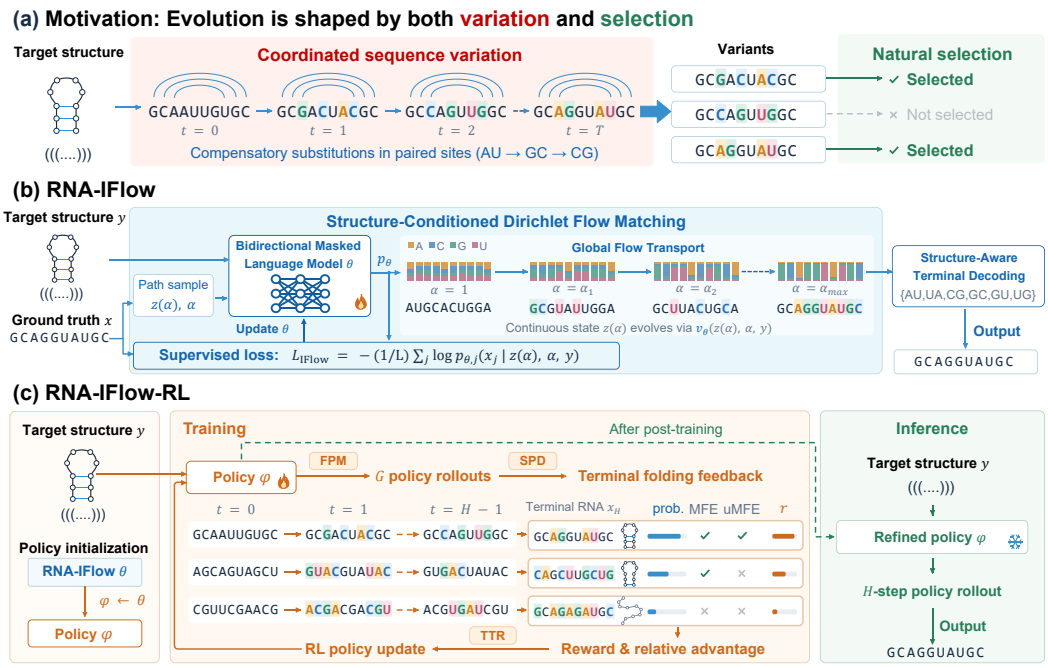

Figure 1: Overview of the two-stage framework. (a) illustrates the variation–selection motivation. (b) shows RNA-IFlow supervised training. (c) shows RNA-IFlow-RL training and inference.

RNA sequence state. For the second challenge, RNA-IFlow-RL introduces RL to model thermodynamic selection over the learned flow predictor. Specifically, it uses **Flow-to-Policy Mapping (FPM)** to convert the learned flow predictor into a finite policy with tractable transition probabilities, and **Thermodynamic Trajectory Refinement (TTR)** to propagate terminal folding feedback to the policy decisions that generate each sequence. For the third challenge, we explicitly impose RNA structural constraints on both stages of our framework. Specifically, RNA-IFlow applies structure-aware terminal decoding to assign valid states to target base pairs, while RNA-IFlow-RL extends this constraint throughout RL refinement by adopting **Structure-Preserving Policy Dynamics (SPD)**.

Our main contributions are summarized as follows:

(1) We propose a two-stage framework for RNA design that models the variation–selection process.
(2) To address the modeling challenges, RNA-IFlow models coordinated sequence variation through structure-conditioned Dirichlet FM, while RNA-IFlow-RL converts the learned flow into a finite RL policy for thermodynamic selection. To integrate compensatory substitutions into the framework, we impose structural constraints on both stages.
(3) Extensive experiments across multiple benchmarks demonstrate that our framework achieves leading performance, notably reaching 85.19% Pass@1 on Rfam-27, compared with 81.48% for RNA-Design-LM SL+RL. Further analyses show that our framework effectively models variation and selection, opening a promising direction for RNA design.

2 Related Work

RNA Design. Existing RNA design methods mainly rely on target-specific search optimization, directly optimizing sequences according to structural or thermodynamic objectives. Representative approaches include NEMO, SAMFEO, SamplingDesign, and FastDesign (Portela, 2018; Zhou et al., 2023; Tang et al., 2026; Zhou et al., 2026). Recently, learning-based and conditional generation methods have been introduced to generalize this optimization. LEARNA and DRAG use learned construction or mutation policies, while RNA-Design-LM and GoForth generate sequences conditioned on target structures (Runge et al., 2019; Li et al., 2025; Gautam et al., 2026; Lindsey, 2026). However, these methods do not explicitly exploit the evolutionary information in RNA datasets.

Flow Matching for Biomolecular Generation. Flow matching (FM) provides a general framework for learning probability transport between distributions (Lipman et al., 2023). Dirichlet and discrete FM variants further extend this idea to categorical sequence spaces, realizing flow-based generation over discrete spaces (Stark et al., 2024; Gat et al., 2024). Recently, flow-based models have been explored for biomolecular generation, including RNA sequence–structure co-design (Nori & Jin, 2024; Ma et al., 2025). These methods demonstrate the potential of FM methods for RNA design.

Reinforcement Learning for RNA Design. Reinforcement learning (RL) has been explored for optimizing RNA sequences toward structural and thermodynamic objectives. LEARNA and DRAG use RL for RNA sequential construction and mutation-based optimization, while RNA-Design-LM applies RL to optimize a trained conditional generator (Runge et al., 2019; Li et al., 2025; Gautam et al., 2026). Building on these advances, we adopt RL to model thermodynamic selection.

3 METHOD

3.1 PROBLEM FORMULATION

An RNA sequence is an ordered chain of nucleotides $x = (x_1, \ldots, x_L) \in \mathcal{A}^L$, where $\mathcal{A} = \{A, C, G, U\}$ denotes the four nucleotide types. Its target secondary structure y describes the expected nucleotide base-pairing pattern in dot-bracket notation, with base-pair set $\mathcal{P}(y)$. A thermodynamic model assigns free energy $\mathcal{G}(s; x)$ to each compatible structure $s \in \mathcal{S}(x)$, where $\mathcal{S}(x)$ denotes the set of valid secondary structures for sequence x. The minimum-free-energy (MFE) structure set of x is $\mathcal{M}(x) = \arg\min_{s\in\mathcal{S}(x)} \mathcal{G}(s; x)$, and the target probability is

$$p(y \mid x) = \frac{\exp[-\mathcal{G}(y; x)/(RT)]}{\sum_{s\in\mathcal{S}(x)} \exp[-\mathcal{G}(s; x)/(RT)]}, \tag{1}$$

where R is the molar gas constant and T the absolute temperature; $p(y \mid x) = 0$ when $y \notin \mathcal{S}(x)$. MFE success means $y \in \mathcal{M}(x)$, while unique-MFE (uMFE) success requires $\mathcal{M}(x) = \{y\}$. A successful design should not only make the target the unique MFE structure, but also favor it thermodynamically over competing conformations. We therefore formulate the objective of RNA design as

$$\max_{x\in\mathcal{A}^L} p(y \mid x) \qquad \text{s.t.} \qquad \mathcal{M}(x) = \{y\}. \tag{2}$$

The goal of our framework is to learn the distribution of RNA sequences conditioned on a given target structure to obtain a successful design. It proceeds in two stages. RNA-IFlow learns a structure-conditioned sequence distribution from supervised sequence–structure pairs. RNA-IFlow-RL then uses terminal thermodynamic selection to reshape this distribution toward successful designs.

3.2 RNA-IFLOW

As illustrated in Figure 1(b), we present **RNA-IFlow** as a structure-conditioned global flow in RNA sequence space to model coordinated sequence variation under structural constraints. It combines **Structure-Conditioned Dirichlet Flow Matching**, **Global Transport**, and **Structure-Aware Terminal Decoding**. The three components respectively aim to learn structure-conditioned variation, enable coordinated sequence-wide generation, and ensure structurally valid decoding.

Structure-Conditioned Dirichlet Flow Matching. We first construct a continuous flow that captures structure-conditioned nucleotide variation. Represent nucleotide x_j by its one-hot vector $e(x_j)$ on the four-category simplex $\Delta^3 = \{z \in \mathbb{R}^4_{\geq 0} : \sum_{a\in\mathcal{A}} z_a = 1\}$. Following Dirichlet Flow Matching (Stark et al., 2024), we define the conditional Dirichlet probability path

$$q_\alpha(z_j \mid x_j) = \mathrm{Dir}(z_j; \mathbf{1} + (\alpha - 1)e(x_j)), \tag{3}$$

where $\mathbf{1} \in \mathbb{R}^4$ is the all-ones vector and $\alpha \geq 1$ serves as the continuous flow parameter and controls concentration toward the clean nucleotide. For the complete sequence, we use the factorized path $q_\alpha(z \mid x) = \prod_{j=1}^{L} q_\alpha(z_j \mid x_j)$. To infer the direction of sequence variation along this path, given the complete state $z = (z_1, \ldots, z_L)$, level α, and structure y, a bidirectional masked language model

backbone f_θ predicts clean-nucleotide probabilities $p_{\theta,j}(a \mid z, \alpha, y) = \mathrm{Softmax}(f_\theta(z, \alpha, y)_j)_a$ for $a \in \mathcal{A}$. These structure-conditioned posteriors are then converted into a continuous velocity field

$$v_{\theta,j}(z, \alpha, y) = \mathsf{P}_{\tan}\left[\sum_{a \in \mathcal{A}} p_{\theta,j}(a \mid z, \alpha, y) c_\alpha(z_{j,a})\big(e(a) - z_j\big)\right], \tag{4}$$

where c_α is the analytic conditional-flow coefficient and $\mathsf{P}_{\tan}$ projects onto the simplex tangent space. Their implementation is given in Appendix A.3.

Global Transport in RNA-IFlow. Having defined the structure-conditioned vector field, we next use it to transport the entire RNA sequence state. RNA-IFlow initializes each position from $z_j \sim \mathrm{Dir}(\mathbf{1})$ and evolves the complete sequence state through

$$\frac{dz_j(\alpha)}{d\alpha} = v_{\theta,j}(z(\alpha), \alpha, y), \qquad j = 1, \ldots, L. \tag{5}$$

Because $v_{\theta,j}$ is predicted from the full sequence state and the shared target structure, all nucleotide positions are updated together under bidirectional context. This global transport allows coordinated sequence-wide variation to emerge throughout generation.

Structure-Aware Terminal Decoding. Global transport generates a continuous terminal state on the nucleotide simplex, which must finally be converted into a discrete RNA sequence while respecting the target pairing constraints. We therefore decode unpaired sites by single-position argmax and each target pair by joint argmax over $\mathcal{A}_{\mathrm{pair}} = \{AU, UA, CG, GC, GU, UG\}$. This structure-aware decoding maps the continuous flow state into a discrete RNA sequence while ensuring that every target base pair is assigned a valid pairing state.

3.3 RNA-IFLOW-RL

As shown in Figure 1(c), we present **RNA-IFlow-RL** to further introduce a thermodynamic selection process into RNA-IFlow. It combines three designs: **Flow-to-Policy Mapping (FPM)** provides tractable discrete transitions, **Structure-Preserving Policy Dynamics (SPD)** preserves target-pair legality throughout generation, and **Thermodynamic Trajectory Refinement (TTR)** propagates terminal feedback to the decisions producing the final sequence.

Flow-to-Policy Mapping (FPM). We first convert the continuous RNA-IFlow representation into an executable discrete policy with tractable transition probabilities. To reuse the trained RNA-IFlow predictor, FPM maps the current sequence $x_t \in \mathcal{A}^L$ to the conditional mean of the path in Equation 3:

$$\mu_\alpha(a) = \frac{\mathbf{1} + (\alpha - 1)e(a)}{\alpha + 3} = \mathbb{E}_{Z \sim q_\alpha(\cdot \mid a)}[Z], \qquad z_{t,j} = \mu_{\alpha_t}(x_{t,j}). \tag{6}$$

Here t indexes policy steps, α_t denotes the Dirichlet path level associated with step t, and j indexes nucleotide positions, while $Z \in \Delta^3$ denotes a random simplex vector. This mapping is the exact conditional mean of the Dirichlet path. Initialized from RNA-IFlow, f_ϕ predicts the clean-nucleotide posterior $p_{\phi,j}(a \mid z_t, \alpha_t, y)$ from (z_t, α_t, y), using the same parameterization as $p_{\theta,j}$.

Given these predictions, we construct a finite discrete transition kernel over structural units. Let $\mathcal{U}(y)$ denote the disjoint structural units induced by y, with each unit corresponding to either an unpaired position or a target base pair, and let $x_{t,u}$ denote the state of unit u at step t. For $u \in \mathcal{U}(y)$, SPD defines a resampling distribution $h_{\phi,u}$, giving

$$\pi_{\phi,u}(x_{t+1,u} \mid x_t, \alpha_t, y) = (1 - \rho_t)\delta_{x_{t,u}}(x_{t+1,u}) + \rho_t h_{\phi,u}(x_{t+1,u} \mid z_t, \alpha_t, y). \tag{7}$$

For an H-step policy horizon, $\rho_t = 1/(H - t)$ for $t = 0, \ldots, H - 1$, and δ denotes a point mass at the current unit state. This normalized transition kernel gives the exact transition probability under the constructed finite policy, enabling tractable policy optimization.

Structure-Preserving Policy Dynamics (SPD). While FPM makes RNA-IFlow transitions tractable, token-wise updates can still violate target base pairs at intermediate steps. SPD therefore constrains the action space of the structural units defined above: unpaired sites have four states in

$\mathcal{A}$, and paired sites $(i, j) \in \mathcal{P}(y)$ have six states in $\mathcal{A}_{\text{pair}}$. Let $\widetilde{p}_{\phi,j}$ denote the temperature-scaled clean-nucleotide posterior. Unpaired units use $\widetilde{p}_{\phi,j}$ directly, while paired units construct a joint policy over legal base-pair states:

$$h_{\phi,(i,j)}(a, b \mid z_t, \alpha_t, y) = \frac{\widetilde{p}_{\phi,i}(a \mid z_t, \alpha_t, y)\, \widetilde{p}_{\phi,j}(b \mid z_t, \alpha_t, y)}{\sum_{(c,d)\in\mathcal{A}_{\text{pair}}} \widetilde{p}_{\phi,i}(c \mid z_t, \alpha_t, y)\, \widetilde{p}_{\phi,j}(d \mid z_t, \alpha_t, y)}, \tag{8}$$

for $(a, b) \in \mathcal{A}_{\text{pair}}$. By restricting paired actions to $\mathcal{A}_{\text{pair}}$, SPD preserves pairing legality at every policy step and extends the structural constraint to the full finite-policy trajectory.

Thermodynamic Trajectory Refinement (TTR). FPM and SPD establish a tractable, structure-preserving policy, but policy refinement still requires translating sequence-level thermodynamic outcomes into learning signals for the generation trajectory. TTR addresses this by evaluating terminal sequences and propagating their relative thermodynamic quality back through the trajectories. For each target y, we sample G trajectories, denoted by $\tau^{(g)} = (x_0^{(g)}, \ldots, x_H^{(g)})$, $g = 1, \ldots, G$. Each terminal sequence $x_H^{(g)}$ receives the reward

$$r^{(g)} = \beta_p\, p(y \mid x_H^{(g)}) + \beta_{\text{MFE}}\, \mathbb{I}_{\text{MFE}}(x_H^{(g)}, y) + \beta_{\text{uMFE}}\, \mathbb{I}_{\text{uMFE}}(x_H^{(g)}, y), \tag{9}$$

where $\mathbb{I}_{\text{MFE}}(x, y) = \mathbf{1}[y \in \mathcal{M}(x)]$, $\mathbb{I}_{\text{uMFE}}(x, y) = \mathbf{1}[\mathcal{M}(x) = \{y\}]$, and β_p, β_{MFE}, and β_{uMFE} are nonnegative weights. Then, we compute the group-normalized advantage

$$A^{(g)} = \frac{r^{(g)} - \bar{r}_y}{\sigma_y + \epsilon_{\text{adv}}}, \tag{10}$$

where $\bar{r}_y$ and σ_y denote the group mean and population standard deviation, and $\epsilon_{\text{adv}} > 0$ ensures numerical stability. The same terminal advantage is assigned to every transition in $\tau^{(g)}$, allowing terminal thermodynamic selection to refine policy decisions along the entire generation trajectory.

3.4 Training and Inference

Training. We train our framework in two stages. In the first stage, RNA-IFlow learns structure-conditioned sequence variation by predicting clean nucleotides along the Dirichlet probability path. For a sequence–structure pair $(x, y) \sim \mathcal{D}_{\text{SL}}$ where $\mathcal{D}_{\text{SL}}$ denotes the supervised training set, we sample a path level α and a simplex state $z \sim q_\alpha(\cdot \mid x)$, and optimize

$$\mathcal{L}_{\text{IFlow}}(\theta) = \mathbb{E}_{\substack{(x,y)\sim\mathcal{D}_{\text{SL}},\ \alpha\sim p(\alpha),\\ z\sim q_\alpha(\cdot|x)}} \left[-\frac{1}{L} \sum_{j=1}^{L} \log p_{\theta,j}(x_j \mid z, \alpha, y) \right]. \tag{11}$$

Here $p(\alpha)$ is the training distribution over path levels. This objective trains the clean-nucleotide predictor that defines the RNA-IFlow vector field in Equation 4.

In the second stage, RNA-IFlow-RL initializes its policy parameters from the trained RNA-IFlow model with $\phi \leftarrow \theta$, and further optimizes the resulting finite policy using the terminal advantages $A^{(g)}$ defined by TTR. To evaluate an executed transition under the current and rollout policies, we first form its joint probability over all structural units:

$$\begin{aligned} \pi_\phi(x_{t+1}^{(g)} \mid x_t^{(g)}, \alpha_t, y) &= \prod_{u\in\mathcal{U}(y)} \pi_{\phi,u}(x_{t+1,u}^{(g)} \mid x_t^{(g)}, \alpha_t, y), \\ \omega_t^{(g)}(\phi) &= \frac{\pi_\phi(x_{t+1}^{(g)} \mid x_t^{(g)}, \alpha_t, y)}{\pi_{\phi_{\text{old}}}(x_{t+1}^{(g)} \mid x_t^{(g)}, \alpha_t, y)}. \end{aligned} \tag{12}$$

Here ϕ_{old} denotes the behavior policy used to collect the rollout and $\omega_t^{(g)}$ is the current-to-behavior importance ratio for the joint structural action at policy step t. TTR uses this ratio to propagate the terminal advantage to the transitions that generated the final sequence. Then, the clipped surrogate is summed over the H policy steps and averaged over the G trajectories:

$$\widehat{\mathcal{L}}_{\text{TTR}}(\phi; y) = -\frac{1}{G} \sum_{g=1}^{G} \sum_{t=0}^{H-1} \min\Big\{ \omega_t^{(g)}(\phi) A^{(g)},\ \text{clip}\big(\omega_t^{(g)}(\phi), 1-\epsilon_{\text{clip}}, 1+\epsilon_{\text{clip}}\big) A^{(g)} \Big\}. \tag{13}$$

The threshold $\epsilon_{\text{clip}} > 0$ defines the clipping interval in the proximal policy optimization (PPO) surrogate (Schulman et al., 2017). To preserve the supervised sequence prior, we add Kullback–Leibler (KL) regularization toward a frozen reference and clean-nucleotide cross-entropy (CE):

$$\mathcal{L}_{\text{RL}}(\phi) = \mathbb{E}_{y\sim\mathcal{D}_{\text{RL}}}\left[\widehat{\mathcal{L}}_{\text{TTR}}(\phi; y)\right] + \lambda_{\text{KL}}\mathcal{L}_{\text{KL}}(\phi, \phi_{\text{ref}}) + \lambda_{\text{CE}}\mathcal{L}_{\text{CE}}(\phi). \tag{14}$$

where $\lambda_{\text{KL}}, \lambda_{\text{CE}} \geq 0$ weight the KL and CE regularization terms, respectively. Here $\mathcal{D}_{\text{RL}}$ denotes the post-training target distribution, ϕ_{ref} is a frozen reference policy, and ϕ_{old} remains the behavior policy associated with the current rollout. $\mathcal{L}_{\text{KL}}$ regularizes the structured resampling distributions toward the reference policy, whereas $\mathcal{L}_{\text{CE}}$ retains the clean-nucleotide supervision learned by RNA-IFlow.

Inference. At inference, RNA-IFlow and RNA-IFlow-RL follow different generation dynamics. RNA-IFlow evolves an initial simplex state through global transport via Equation 5 and then applies structure-aware terminal decoding. In contrast, RNA-IFlow-RL uniformly initializes each structural unit over its legal state space and directly generates a discrete sequence through an H-step RL policy. For each target, an inference budget of K candidates is realized by K independent trajectories. Thermodynamic feedback is used during post-training, not to guide inference-time sampling or reranking. Generated sequences may subsequently be scored for evaluation.

4 Experiments

We first evaluate both stages of our framework on standard RNA design benchmarks, then analyze post-training dynamics and the flow-to-policy transition, and finally present a case study.

4.1 Experimental Setup

Dataset and benchmarks. RNA-IFlow is trained on 10 million sequence–structure pairs released by RNA-Design-LM (Gautam et al., 2026). These pairs are constructed from one million random RNAs of 6–500 nucleotides by folding each sequence with ViennaRNA[1] (Lorenz et al., 2011) and generating ten sequence designs per target with SAMFEO (Zhou et al., 2023). RNA-IFlow-RL uses the released 2,790-target EternaWeb post-training set (Koodli et al., 2019; Gautam et al., 2026). It excludes structures that exceed 500 nucleotides or cannot be designed by MFE, and further selects targets with moderate design difficulty using the supervised model. We evaluate on Eterna100-v2, Eterna100, and Rfam-27 (Taneda, 2011; Anderson-Lee et al., 2016; Koodli et al., 2021), and report additional results on RNAsolo (Adamczyk et al., 2022) in Appendix A.5.

Baselines. **(1) Conditional language models:** RNA-Design-LM SL is an autoregressive model trained with supervised learning, while RNA-Design-LM SL+RL further applies RL training to it (Gautam et al., 2026). GoForth uses an encoder–decoder architecture for RNA design (Lindsey, 2026). **(2) Learning-based optimization methods:** DRAG performs RNA design through hierarchical graph-based nucleotide mutation (Li et al., 2025). **(3) Search-based methods:** RNAinverse-pf optimizes sequences toward a target structure using a partition-function objective (Lorenz et al., 2011). Additional comparisons with NEMO, SAMFEO and FastDesign are reported in Appendix A.9.

Evaluation settings. The inference budget is $K = 8$ ordered candidates per target. Pass@1 (P@1) and P@8 denote uMFE success for the first candidate and for at least one of the first eight candidates, respectively. MFE@8 success allows the target structure to be one of multiple MFE folds. We also report target probability and normalized ensemble defect (NED) for thermodynamic quality, Pair-F1 for structural agreement, and sequence diversity. All candidates are evaluated with ViennaRNA.

Implementation details. RNA-IFlow uses RNAErnie (Wang et al., 2024) as the bidirectional LM backbone and contains 87.55M parameters, with 86.96M updated during supervised training and 14.49M during RL post-training. Supervised Flow Matching samples $\alpha \sim \mathcal{U}(1, 8)$ and uses 50 integration steps at inference. RNA-IFlow-RL uses $G = 8$ trajectories per target and $H = 8$ policy transitions; its terminal reward combines target probability, MFE success, and uMFE success with weights 0.5, 0.25, and 0.25. Training RNA-IFlow and RNA-IFlow-RL spent 84 and 48 GPU-hours, respectively. Full settings are provided in Appendix A.4.

[1]Official site: `https://www.tbi.univie.ac.at/RNA/`

Table 1: Overall performance. Bold and underlined values denote the best and second-best results, respectively. The standard deviations are reported in Appendix 9.

Method	Type	Params (M)	Eterna100-v2		Eterna100		Rfam-27	
			P@1 ↑	P@8 ↑	P@1 ↑	P@8 ↑	P@1 ↑	P@8 ↑
RNA-Design-LM SL	Conditional gen.	357.92	0.2633	0.4333	0.2633	0.4267	0.5185	0.7531
RNA-Design-LM SL+RL	Conditional gen.	357.92	<u>0.5133</u>	<u>0.6367</u>	<u>0.5033</u>	<u>0.6133</u>	<u>0.8148</u>	<u>0.8642</u>
GoForth	Conditional gen.	59.39	0.3933	0.5700	0.3933	0.5667	0.6296	0.7284
DRAG	Learned opt.	0.04	0.3633	0.5133	0.3633	0.5133	0.7160	0.8272
RNAinverse-pf	Search	-	0.4367	0.5433	0.4433	0.5100	0.3827	0.4568
RNA-IFlow (Ours)	Flow gen.	87.55	0.3033	0.5200	0.3000	0.4867	0.6543	0.8272
RNA-IFlow-RL (Ours)	Flow + RL	87.55	**0.5400**	**0.6500**	**0.5067**	**0.6167**	**0.8519**	**0.8765**

Table 2: Thermodynamic quality and runtime for eight candidates. Lower NED is better. Runtimes are measured on the same machines, while starred rows include internal search and folding.

Method	Eterna100-v2			Eterna100		Rfam-27		Run time (s)
	MFE@8 ↑	Prob. ↑	NED ↓	Prob. ↑	NED ↓	Prob. ↑	NED ↓	
RNA-Design-LM SL	0.4433	0.3364	0.1694	0.3326	0.1702	0.6200	0.0267	4.41
RNA-Design-LM SL+RL	<u>0.6567</u>	<u>0.5256</u>	<u>0.0652</u>	<u>0.5165</u>	<u>0.0691</u>	<u>0.7765</u>	<u>0.0058</u>	4.41
GoForth	0.5833	0.3499	0.0804	0.3435	0.0848	0.4878	0.0162	1.35
DRAG*	0.5200	0.2391	0.1345	0.2376	0.1373	0.4218	0.0468	14.86
RNAinverse-pf*	0.5700	0.4344	0.3625	0.4292	0.3594	0.4375	0.5076	133.56
RNA-IFlow (Ours)	0.5367	0.4089	0.1017	0.3988	0.1099	0.7158	0.0104	<u>0.88</u>
RNA-IFlow-RL (Ours)	**0.6733**	**0.5398**	**0.0534**	**0.5283**	**0.0593**	**0.7785**	**0.0057**	**0.73**

4.2 Main Results

RNA-IFlow-RL improves RNA design performance. As shown in Table 1, RNA-IFlow already improves P@8 over RNA-Design-LM SL from 0.4333 to 0.5200 on Eterna100-v2, from 0.4267 to 0.4867 on Eterna100, and from 0.7531 to 0.8272 on Rfam-27. RL post-training further raises P@8 to 0.6500, 0.6167, and 0.8765, respectively. On Eterna100-v2, RNA-IFlow-RL exceeds RNA-Design-LM SL+RL, DRAG, and RNAinverse-pf by 1.33, 13.67, and 10.67 percentage points in P@8, while also achieving the highest target probability and lowest NED across the three benchmarks (Table 2). These results show that RNA-IFlow-RL consistently improves both design success and thermodynamic quality across benchmarks.

RNA-IFlow-RL is parameter-efficient and fast at inference. RNA-IFlow-RL uses 87.55M parameters, compared with 357.92M for RNA-Design-LM, while achieving stronger design performance. Moreover, Table 2 shows that RNA-IFlow-RL generates eight candidates in 0.73 s, compared with 1.35 s for GoForth and 4.41 s for RNA-Design-LM SL+RL. These results demonstrate that RNA-IFlow-RL is efficient in both model size and inference speed.

4.3 Post-Training Dynamics and Robustness

Post-training converges and shifts the thermodynamic quality distribution. Figure 2 (a) shows the stabilization of RL post-training: after 1500 updates, the terminal reward reaches a plateau and P@8 follows the same overall trend. Figure 2(b,c) further shows the thermodynamic distribution shift induced by RL: RNA-IFlow-RL shifts target probability toward higher values and NED toward lower values compared with RNA-IFlow.

Performance is stable across policy steps and group sizes. As shown in Figure 2(d,e), P@8 remains stable across the tested H and G values, while inference and update time increase with the policy horizon H and the group size G, respectively. Therefore, the setting $H = 8$ and $G = 8$ provides a trade-off between design quality and time cost. More analyses are shown in Appendix A.6.

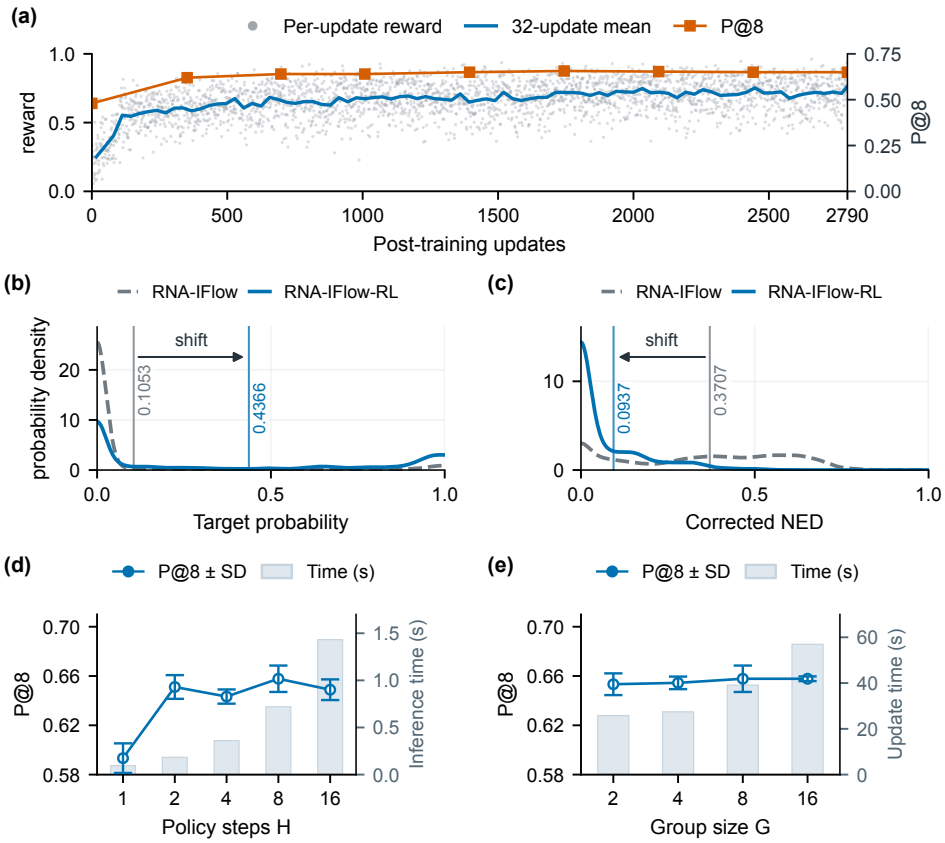

Figure 2: Post-training dynamics and sensitivity analysis. (a) shows reward and P@8 changes during post-training. (b,c) show shifts in target-probability and NED distributions. (d,e) show mean P@8 $\pm$ SD across three seeds for H and G, together with inference and update times, respectively.

4.4 From Continuous Flow to a Finite RL Policy

Figure 3 (a) shows how RNA-IFlow gradually transforms an uncertain sequence state into a more confident and pairing-compatible sequence along the continuous flow. After converting the flow predictor into a finite policy, more sites are considered for variation at later steps, while only a small fraction of nucleotides are actually changed (Figure 3 (b)). This indicates that the policy performs selective sequence refinement rather than simply rewriting the whole sequence. In Figure 3 (c), the flow model and RL policy start from the same sequence, but RNA-IFlow-RL consistently makes beneficial variations with substantially higher target probability and lower NED. This demonstrates that RL training guides the policy to learn which sequence variations are thermodynamically useful.

4.5 Case Study

On the 214-nt Anemone target, the displayed RNA-IFlow-RL candidate reaches target probability 0.5178, NED 0.0128, and uMFE success, whereas the displayed candidates from the comparison methods do not achieve target MFE success (Figure 4). RNA-IFlow-RL also reaches 53.27% sequence recovery to the player-designed reference, compared with 40.65–47.20% for the other methods. The case study visually demonstrates the superiority of RNA-IFlow-RL in RNA design.

5 Conclusion

In this work, we present a two-stage framework comprising RNA-IFlow and RNA-IFlow-RL for RNA design, combining structure-conditioned Dirichlet FM with thermodynamic policy refinement. The

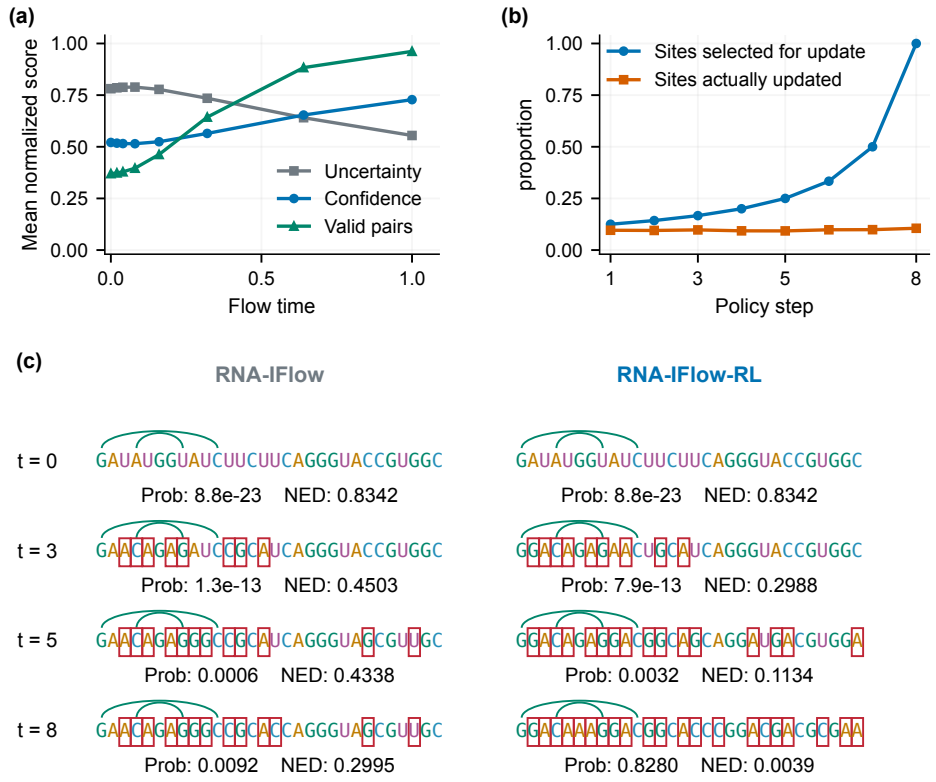

Figure 3: Continuous-flow and finite-policy sequence evolution. (a) shows sequence uncertainty, prediction confidence, and the proportion of legal target pairs over normalized flow time. (b) compares scheduled (blue) and actual (orange) nucleotide updates across policy steps. (c) compares matched RNA-IFlow and RNA-IFlow-RL trajectories; red boxes mark differences from the initial sequence.

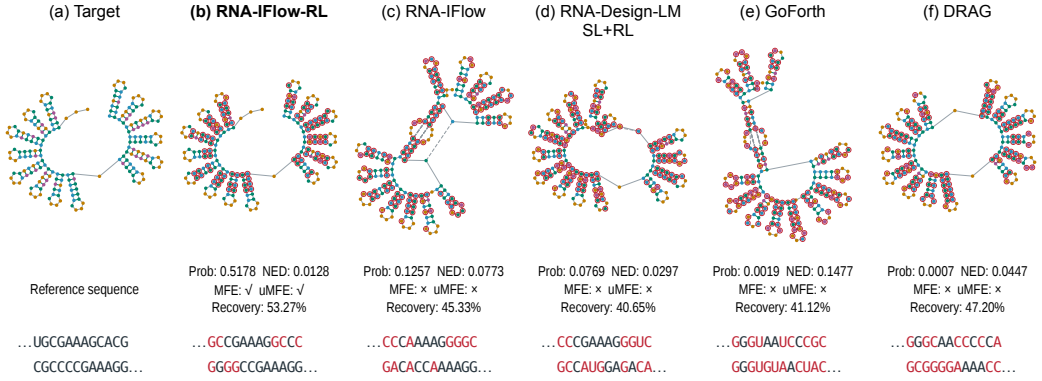

Figure 4: This figure compares Anemone designs (214 nt). Scores and recovery use full sequences to evaluate. In the visualization of secondary structure, red bases mark differences from the reference, while green, orange, and dashed gray bonds denote recovered, alternative, and missing pairs.

flow model captures coordinated sequence-wide variation, while the finite policy uses terminal folding feedback to shift generation toward thermodynamically favorable sequences. Extensive experiments show that RNA-IFlow-RL outperforms comparative methods across multiple benchmarks. Further analyses reveal thermodynamic gains, policy dynamics, and robustness in our framework. These findings demonstrate that our framework advances RNA design and opens a promising direction for broader RNA design modeling.

ETHICS STATEMENT

This study concerns computational RNA design. Biological use of the designed sequences requires experimental validation and appropriate biosafety review.

REPRODUCIBILITY STATEMENT

The main text defines the model, objectives, and evaluation protocols. The appendix records the training data, candidate budgets, decoder settings, sensitivity studies, search configurations, timing scopes, and case-selection rules. Tables and figures are generated from versioned full-precision results. The structural case uses unchanged full-length sequences and a common cropped display window. An anonymized implementation and reproduction scripts are included in the supplementary material.

AI USE STATEMENT

Generative AI tools were used to assist literature retrieval, research ideation and experimental workflows, manuscript drafting and revision, software review, and figure preparation. All AI-assisted research outputs, code, experimental results, citations, and interpretations were reviewed and verified by the authors. All reported numerical results originate from recorded model evaluations and RNA folding calculations. The authors take full responsibility for the final content.

REFERENCES

Bartosz Adamczyk, Maciej Antczak, and Marta Szachniuk. RNAsolo: A repository of cleaned PDB-derived RNA 3D structures. *Bioinformatics*, 38(14):3668–3670, 2022. doi: 10.1093/bioinformatics/btac386. URL https://doi.org/10.1093/bioinformatics/btac386.

Jeff Anderson-Lee, Eli Fisker, Vineet Kosaraju, Michelle Wu, Justin Kong, Jeehyung Lee, Minjae Lee, Mathew Zada, Adrien Treuille, Rhiju Das, and Eterna Players. Principles for predicting RNA secondary structure design difficulty. *Journal of Molecular Biology*, 428(5, Part A):748–757, 2016. doi: 10.1016/j.jmb.2015.11.013.

Mirela Andronescu, Anthony P. Fejes, Frank Hutter, Holger H. Hoos, and Anne Condon. A new algorithm for RNA secondary structure design. *Journal of Molecular Biology*, 336(3):607–624, 2004. doi: 10.1016/j.jmb.2003.12.041.

Ying Chen, David B. Carlini, John F. Baines, John Parsch, John M. Braverman, Soichi Tanda, and Wolfgang Stephan. RNA secondary structure and compensatory evolution. *Genes & Genetic Systems*, 74(6):271–286, 1999. doi: 10.1266/ggs.74.271.

Julien Y. Dutheil, Fabrice Jossinet, and Eric Westhof. Base pairing constraints drive structural epistasis in ribosomal RNA sequences. *Molecular Biology and Evolution*, 27(8):1868–1876, 2010. doi: 10.1093/molbev/msq069.

Rachel Fieldhouse and Mohana Basu. Moderna cancer vaccine stops melanoma returning: what's next for personalized treatments? *Nature*, 657:16–17, 2026. doi: 10.1038/d41586-026-02612-3. URL https://doi.org/10.1038/d41586-026-02612-3.

Itai Gat, Tal Remez, Neta Shaul, Felix Kreuk, Ricky T. Q. Chen, Gabriel Synnaeve, Yossi Adi, and Yaron Lipman. Discrete flow matching. In *Advances in Neural Information Processing Systems*, volume 37, pp. 133345–133385, 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/f0d629a734b56a642701bba7bc8bb3ed-Abstract-Conference.html.

Milan Gautam, Ning Dai, Tianshuo Zhou, Bowen Xie, David Mathews, and Liang Huang. Designing RNAs with language models, 2026. URL https://arxiv.org/abs/2602.12470.

Rohan V. Koodli, Benjamin Keep, Katherine R. Coppess, Fernando Portela, Eterna participants, and Rhiju Das. EternaBrain: Automated RNA design through move sets and strategies from an internet-scale RNA videogame. *PLoS Computational Biology*, 15(6):e1007059, 2019. doi: 10.1371/journal.pcbi.1007059.

Rohan V. Koodli, Boris Rudolfs, Hannah K. Wayment-Steele, Eterna Structure Designers, and Rhiju Das. Redesigning the Eterna100 for the Vienna 2 folding engine. *bioRxiv*, 2021. doi: 10.1101/2021.08.26.457839. URL https://www.biorxiv.org/content/10.1101/2021.08.26.457839v1.

Yichong Li, Xiaoyong Pan, Hongbin Shen, and Yang Yang. DRAG: Design RNAs as hierarchical graphs with reinforcement learning. *Briefings in Bioinformatics*, 26(2):bbaf106, 2025. doi: 10.1093/bib/bbaf106. URL https://doi.org/10.1093/bib/bbaf106.

Michael Lindsey. GoForth: Language models for RNA design under structure, sequence, and coding constraints, 2026. URL https://arxiv.org/abs/2605.07608.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=PqvMRDCJT9t.

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. doi: 10.1186/1748-7188-6-26.

Runze Ma, Zhongyue Zhang, Zichen Wang, Chenqing Hua, Jiahua Rao, Zhuomin Zhou, and Shuangjia Zheng. RiboFlow: Conditional de novo RNA co-design via synergistic flow matching. In *Advances in Neural Information Processing Systems*, volume 38, pp. 77810–77842, 2025. doi: 10.52202/085713-2348. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/653353d903d87720d781a9554b0018db-Abstract-Conference.html.

David McAllister, Songwei Ge, Brent Yi, Chung Min Kim, Ethan Weber, Hongsuk Choi, Haiwen Feng, and Angjoo Kanazawa. Flow matching policy gradients. In *International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/3d43cc5692bf68944ee7cd31b97d0c11-Abstract-Conference.html.

Divya Nori and Wengong Jin. RNAFlow: RNA structure & sequence design via inverse folding-based flow matching. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 38395–38408. PMLR, 2024. URL https://proceedings.mlr.press/v235/nori24a.html.

Fernando Portela. An unexpectedly effective Monte Carlo technique for the RNA inverse folding problem. *bioRxiv*, 2018. doi: 10.1101/345587. URL https://www.biorxiv.org/content/10.1101/345587v1.

Frederic Runge, Danny Stoll, Stefan Falkner, and Frank Hutter. Learning to design RNA. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=ByfyHh05tQ.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017. URL https://arxiv.org/abs/1707.06347.

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

Maojiang Su, Po-Chung Hsieh, Weimin Wu, Mingcheng Lu, Jiunhau Chen, Jerry Yao-Chieh Hu, and Han Liu. Discrete flow matching policy optimization, 2026. URL https://arxiv.org/abs/2604.06491.

Akito Taneda. MODENA: A multi-objective RNA inverse folding. *Advances and Applications in Bioinformatics and Chemistry*, 4:1–12, 2011. doi: 10.2147/AABC.S14335.

Wei Yu Tang, Ning Dai, Tianshuo Zhou, David H. Mathews, and Liang Huang. SamplingDesign: RNA design via continuous optimization with coupled variables and Monte-Carlo sampling. *Nature Communications*, 17:2950, 2026. doi: 10.1038/s41467-025-67901-3. URL https://doi.org/10.1038/s41467-025-67901-3.

Chenyu Wang, Masatoshi Uehara, Yichun He, Amy Wang, Avantika Lal, Tommi Jaakkola, Sergey Levine, Aviv Regev, Hanchen Wang, and Tommaso Biancalani. Fine-tuning discrete diffusion models via reward optimization with applications to DNA and protein design. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/771e09dd204ea339da0d8114c48afd21-Abstract-Conference.html.

Ning Wang, Jiang Bian, Yuchen Li, Xuhong Li, Shahid Mumtaz, Linghe Kong, and Haoyi Xiong. Multi-purpose RNA language modelling with motif-aware pretraining and type-guided fine-tuning. *Nature Machine Intelligence*, 6:548–557, 2024. doi: 10.1038/s42256-024-00836-4. URL https://www.nature.com/articles/s42256-024-00836-4.

Max Ward, Eliot Courtney, and Elena Rivas. Fitness functions for RNA structure design. *Nucleic Acids Research*, 51(7):e40, 2023. doi: 10.1093/nar/gkad097.

Joseph N. Zadeh, Conrad D. Steenberg, Justin S. Bois, Brian R. Wolfe, Marshall B. Pierce, Asif R. Khan, Robert M. Dirks, and Niles A. Pierce. NUPACK: Analysis and design of nucleic acid systems. *Journal of Computational Chemistry*, 32(1):170–173, 2011. doi: 10.1002/jcc.21596.

Tianshuo Zhou, Ning Dai, Sizhen Li, Max Ward, David H. Mathews, and Liang Huang. RNA design via structure-aware multifrontier ensemble optimization. *Bioinformatics*, 39(Supplement 1):i563–i571, 2023. doi: 10.1093/bioinformatics/btad252.

Tianshuo Zhou, David H. Mathews, and Liang Huang. Fast and versatile RNA design via motif-level divide-and-conquer and structure-level rival search, 2026. URL https://arxiv.org/abs/2603.02283.

A Additional Methods and Experiments

The supplementary material records the evaluation contracts, complete benchmark results, policy and data controls, native search protocols, large-budget references, and structural examples. All model variants retain their recorded checkpoints and generation interfaces.

A.1 Extended sampling and data analyses

Candidate budget reveals a quality–diversity trade-off. RNA-IFlow-RL has higher Pass@K than RNA-Design-LM SL+RL through $K = 64$, whereas RNA-Design-LM SL+RL becomes higher at larger budgets (Figure 5a). RNA-IFlow-RL nevertheless retains higher target probability and lower NED across the measured prefixes, while its fraction of distinct sequences decreases more rapidly as K grows (Figure 5b–d). Thus, sampling budget improves coverage and thermodynamic quality at different rates.

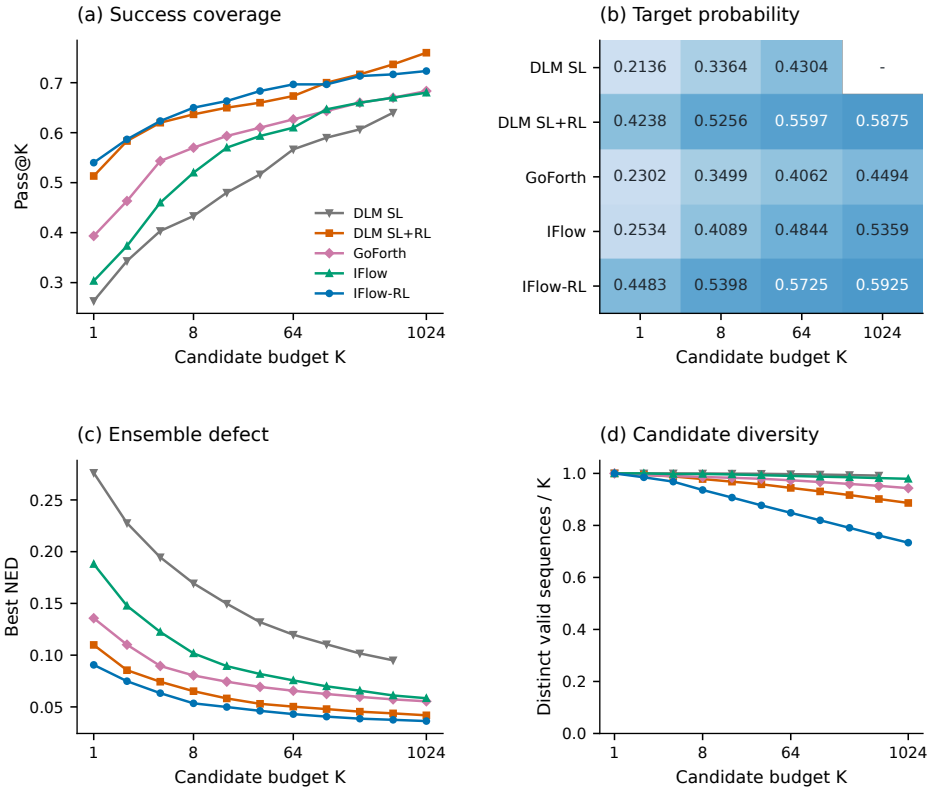

Figure 5: This figure compares candidate budgets on Eterna100-v2. Panel (a) shows uMFE coverage, panels (b,c) show the best target probability and NED, and panel (d) shows distinct valid sequences. Curves use nested candidate prefixes.

Training composition and coverage affect quality and diversity differently. At matched training-set size, the Original mix achieves higher P@8 and lower NED than the restricted Easy and Easy + Hard sets (Figure 6a,b). Under a fixed number of target visits, increasing the number of unique training samples raises sequence diversity, while P@8 changes non-monotonically (Figure 6b). This exposes a coverage–revisitation trade-off rather than a simple scaling law.

Search and neural generation occupy different quality–cost regimes. Under a native 60-second search limit, SAMFEO attains slightly higher uMFE success than RNA-IFlow-RL, whereas RNA-IFlow-RL achieves higher target probability and lower NED (Figure 7). The comparison therefore

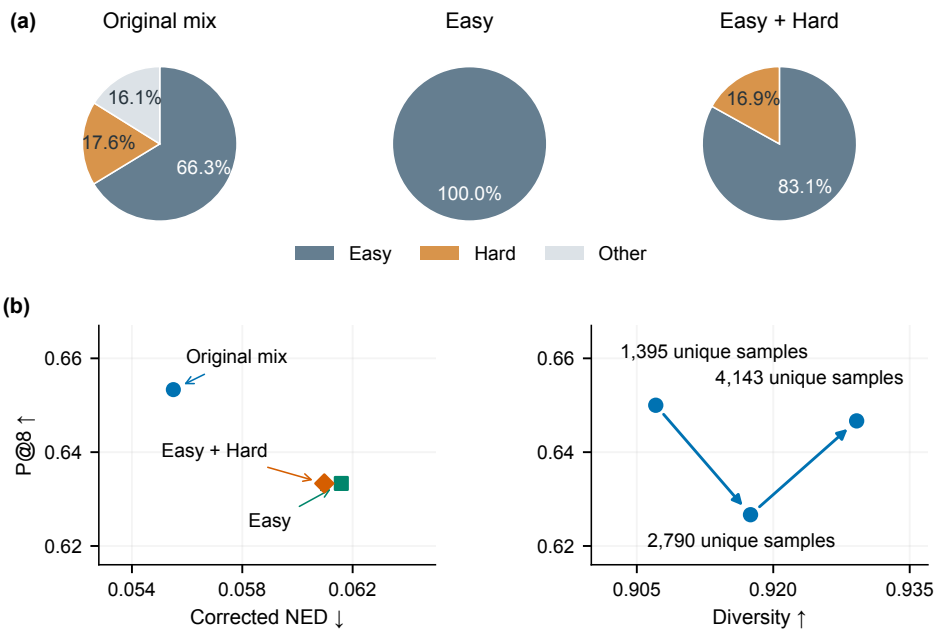

Figure 6: This figure analyzes training-data composition and coverage. Panel (a) compares equal-size training mixtures, while panel (b) relates composition to quality (left) and fixed-exposure coverage to diversity (right). Unique samples denote distinct target structures.

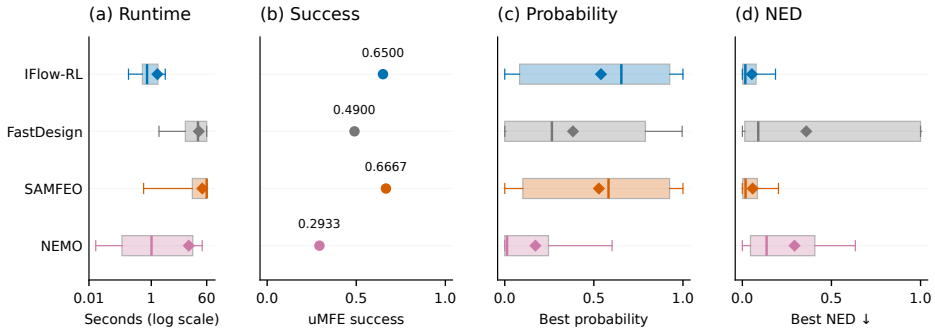

Figure 7: This figure compares quality and cost under the recorded generation and native-search protocols. Panel (a) shows runtime, panel (b) shows uMFE success, and panels (c,d) show target probability and NED. Boxes show quartiles and diamonds denote means.

reflects different quality–compute trade-offs rather than a single metric advantage. Full native-search and large-budget results are reported in Appendices A.9 and A.10.

A.2 Figure guide and evaluation scope

Method overview. Figure 1 illustrates coordinated variation under a fixed pairing topology, supervised RNA-IFlow training, and RNA-IFlow-RL post-training. In Stage 1, the loss is computed from the clean-nucleotide predictor before terminal decoding. The separate generation branch transports the continuous state and decodes it into a sequence. In Stage 2, terminal folding scores produce group-relative advantages for policy updates. Flames denote trainable parameters and snowflakes

denote frozen inference parameters. Highlighted bases show changes from the preceding displayed state. Sequences and folding graphics in the overview are schematic.

Post-training distributions and sensitivity. Figure 2(a) summarizes terminal reward and P@8 during post-training. Panels (b,c) show the candidate-level distributions of target probability and NED, with solid vertical lines marking the corresponding means. Panels (d,e) report sensitivity to the policy horizon H and group size G. The $G = 1$ setting has zero group-relative advantage and is listed separately in Table 11.

From continuous flow to a finite policy. Figure 3(a) summarizes native continuous-flow trajectories over 13 targets. Uncertainty, confidence, and the legal-pair fraction use discrete site-wise predictor readouts. Panel (b) compares the finite-policy schedule's expected site-selection rate with the observed nucleotide-change rate in post-trained trajectories. Prob. and NED evaluate the full sequence after sampling; the displayed window is used only for visualization.

Training mixture and coverage. Figure 6(a) shows three equal-size training sets. The Original mix contains 66.3% Easy, 17.6% Hard, and 16.1% Other samples. Easy is selected using the predefined learnability criterion and contains 100% Easy samples; Easy + Hard contains 83.1% Easy and 16.9% Hard samples. These latter sets correspond to the core-only and hard-enriched names in Table 16. In panel (b), the left plane compares their NED and P@8, and the right plane compares diversity and P@8 under 4,144 target visits. Unique samples denote distinct target structures. Every configuration uses one training seed, and broader coverage reduces the mean number of repeated visits.

Case interpretation. Figure 4 compares the Anemone target and five methods. The displayed sequences show positions 83–132, while structures, scores, and recovery use all 214 nucleotides. Recovery is sequence identity to the Eterna player-designed reference. Red letters and node outlines indicate reference-sequence differences; structural agreement is encoded separately by green recovered pairs, orange alternative pairs, and dashed gray missing pairs. MFE and uMFE marks refer to the displayed candidate. Table 31 supplies group-level hit counts and Appendix A.11 records the candidate-selection rule.

Result version and training cost. The numerical tables and author-review figures in this manuscript report results across three evaluation seeds. The supervised training cost of 84.1 A100 GPU-hours covers eight epochs, with epoch 6 selected for evaluation. The complete RL training lineage used to obtain the reported model requires approximately 48.0 allocated A100 GPU-hours.

A.3 Theory and implementation alignment

Index and distribution conventions. g labels an RL trajectory, $t \in \{0, \dots, H\}$ its state index, j a nucleotide position, and $u \in \mathcal{U}(y)$ a structural unit. Thus $x_{t,j}^{(g)}$ is one nucleotide, $x_t^{(g)}$ the complete sequence, and $z_t^{(g)}$ its simplex representation. Static x_j denotes the clean nucleotide in a sequence–structure training example. G is the number of training trajectories per target, H the policy horizon, and K the evaluation candidate budget. Free energy is $\mathcal{G}$, distinct from G. The model uses parameters θ before post-training and ϕ during refinement; ϕ_{old} identifies the behavior model and ϕ_{ref} the fixed reference.

Conditional-flow coefficient and numerical transport. For $\xi \in (0, 1)$, the four-category Dirichlet conditional-flow coefficient is

$$c_\alpha(\xi) = -\frac{\partial I_\xi(\alpha, 3)}{\partial \alpha} \frac{B(\alpha, 3)}{(1-\xi)^3 \xi^{\alpha-1}}, \tag{15}$$

where B is the beta function and I_ξ the regularized incomplete beta function. Equation 4 averages $c_\alpha(z_{j,a})(e(a) - z_j)$ under the clean-nucleotide predictor. The tangent projection is $\mathsf{P}_{\tan}(v) = v - \frac{1}{4}(\sum_a v_a)\mathbf{1}$ and enforces zero coordinate sum. It is distinct from the Euclidean state projection Π_{Δ^3} used after each numerical Euler update:

$$z_j(\alpha_{k+1}) = \Pi_{\Delta^3}[z_j(\alpha_k) + (\alpha_{k+1} - \alpha_k) v_{\theta,j}(z(\alpha_k), \alpha_k, y)]\,. \tag{16}$$

Here k indexes integration steps. The native-flow evaluation initializes from $\mathrm{Dir}(\mathbf{1})$ and uses 50 steps on the numerical grid from 1.001 to 8.0. The implementation evaluates the beta derivative through interpolation and guards non-finite values. Simplex projection preserves nonnegativity and unit sum after the tangent-projected update.

Exact conditional-mean bridge. The concentration vector of $q_\alpha(\cdot \mid a)$ sums to $\alpha + 3$. A Dirichlet vector has mean equal to its concentration vector divided by that sum, proving Equation 6. The clean coordinate has mean $\alpha/(\alpha+3)$ and the other coordinates $1/(\alpha+3)$. This identity specifies the bridge, without asserting equality between native continuous transport and the constructed finite-policy dynamics.

Sampling temperature and structural distributions. For a fixed trajectory temperature $\vartheta > 0$, define

$$\widetilde{p}_{\phi,j}(a \mid z, \alpha, y) = \mathrm{Softmax}(f_\phi(z, \alpha, y)_j/\vartheta)_a\,. \tag{17}$$

The untempered predictor is recovered at $\vartheta = 1$. Paired-unit probabilities are the normalized products in Equation 8. The same temperature is used for behavior scoring and current-policy rescoring of a trajectory. Policy and unit-distribution notation suppresses this fixed setting.

Normalization and all-step pairing legality. Each $h_{\phi,u}$ is normalized on its four- or six-state alphabet. Since $0 < \rho_t \le 1$, Equation 7 is a convex mixture of two normalized distributions. Paired units are initialized within $\mathcal{A}_{\mathrm{pair}}$. If a paired state is legal at step t, both keeping it and resampling from $h_{\phi,u}$ retain legality; induction therefore establishes legality through step H. The final step has $\rho_{H-1} = 1$. This guarantees allowed target-pair states, not that the target is the MFE or uMFE fold.

Finite trajectory probability. Let U_{s} and U_{p} be the counts of single-position and paired units. The initialization distribution assigns $\nu_y(x_0) = 4^{-U_{\mathrm{s}}}6^{-U_{\mathrm{p}}}$ to every legal initial state and zero otherwise. Conditional on the complete preceding state, structural units are sampled independently. Consequently, for a legal trajectory,

$$P_\phi(\tau^{(g)} \mid y) = \nu_y(x_0^{(g)}) \prod_{t=0}^{H-1} \prod_{u \in \mathcal{U}(y)} \pi_{\phi,u}(x_{t+1,u}^{(g)} \mid x_t^{(g)}, \alpha_t, y). \tag{18}$$

Here P_ϕ denotes a distribution over complete finite-policy trajectories. The common initialization is independent of ϕ. Training clips the step ratios defined in Equation 12.

Loss reductions and anchors. For one target group, $\bar{r}_y = G^{-1}\sum_g r^{(g)}$ and $\sigma_y^2 = G^{-1}\sum_g (r^{(g)} - \bar{r}_y)^2$. The implementation uses $\epsilon_{\mathrm{adv}} = 10^{-4}$ and assigns zero advantages when the reward range is below 10^{-6}. Advantages and behavior probabilities are held fixed during the corresponding optimizer update. Equation 13 averages over the G trajectories and sums over H transitions; replacing the sum by a time mean would change its scale relative to the anchors.

For $c_t^{(g)} = (z_t^{(g)}, \alpha_t, y)$, the KL anchor for one target group is

$$\widehat{\mathcal{L}}_{\mathrm{KL}}(\phi; y) = \frac{1}{GH|\mathcal{U}(y)|} \sum_{g=1}^{G} \sum_{t=0}^{H-1} \sum_{u \in \mathcal{U}(y)} D_{\mathrm{KL}}\Big(h_{\phi,u}(\cdot \mid c_t^{(g)}) \,\|\, h_{\phi_{\mathrm{ref}},u}(\cdot \mid c_t^{(g)})\Big)\,. \tag{19}$$

The notation $h(\cdot \mid c)$ abbreviates the explicitly conditioned distribution in the main text. Averaging this group loss over training targets gives $\mathcal{L}_{\mathrm{KL}}$. It compares the current and reference *resampling distributions*. For a supervised batch $\mathcal{B}$ of clean sequence–structure pairs with sampled Dirichlet states, the CE anchor is the valid-token mean

$$\widehat{\mathcal{L}}_{\mathrm{CE}}(\phi; \mathcal{B}) = -\frac{\sum_{(x,y)\in\mathcal{B}} \sum_{j=1}^{|x|} \log p_{\phi,j}(x_j \mid z, \alpha, y)}{\sum_{(x,y)\in\mathcal{B}} |x|}. \tag{20}$$

A flow level is sampled for each minibatch, and sequence-specific Dirichlet states provide the noisy inputs; padding is excluded. $\mathcal{L}_{\mathrm{CE}}$ denotes its training expectation. The distributed reduction averages over all valid nucleotides.

The formal PPO implementation sums log ratios over units, then clamps the joint log ratio to $[-20, 20]$ before exponentiation for numerical stability; the PPO clipping range is a separate setting.

Empirical studies. The main configuration uses $G = 8, H = 8$. Additional policy and training-group analyses are reported in Appendix A.6.

A.4 Training data, parameter counts, and evaluation contracts

Supervised learning uses 10 million solver-generated sequence–structure pairs from the released RNA-Design-LM corpus. Post-training uses the released 2,790-target EternaWeb set with the supervised cross-entropy anchor retained. Sequence length ranges from 6 to 500 nucleotides. The original mixture uses all selected targets; the composition study groups them by the learnability-based categories in Appendix A.2.

The benchmark contract fixes target identities, target order, candidate order, and three evaluation seeds. Native RNA-IFlow uses 50 continuous-flow integration steps and terminal argmax decoding.

Within each target–condition group, Pass@1 uses the first candidate and Pass@8 accepts any uMFE success among eight candidates. MFE@8 includes tied minima. Target probability, pair-set F1, and NED are scored per candidate and optimized independently within the group. No thermodynamic score guides sampling or reranking for the conditional generators. Diversity is the number of distinct valid returned RNA strings divided by eight. Empty failed returns do not count as unique sequences, but the eight-candidate denominator is retained.

Pair-set F1 is twice the number of shared predicted/target base pairs divided by the total number of pairs in the two sets. ViennaRNA's normalized ensemble defect is used directly, without a second division by length. The common final evaluator uses ViennaRNA 2.7.2 at 37 degrees Celsius with dangles 2 and unique multiloop decomposition. All benchmark sets are used solely for performance evaluation and do not participate in training, checkpoint choice, or hyperparameter tuning. Final benchmark metrics are reported on Eterna100-v2, Eterna100, and Rfam-27. The RNAsolo overlap analysis is given below. Cross-model supervised comparisons retain each model's trained weights and prescribed decoder.

Parameter accounting. Counts refer to deduplicated parameters in the evaluated model constructors and checkpoints. RNA-Design-LM uses a 13-token vocabulary. GoForth uses the pretrained-small configuration. DRAG includes its shared backbone and actor/critic heads, with 36,593 parameters in the actor-plus-backbone inference subset. Historical training masks for external models are unverified. Pure-search methods have no neural parameter count. Model capacities and trainable parameter counts are summarized in Table 3.

Table 3: This table reports model capacity. A dash denotes either no neural parameters for pure search or an unknown historical training mask.

Method	Total parameters	Millions	Trained parameters
RNA-Design-LM SL	357921408	357.92	-
RNA-Design-LM SL+RL	357921408	357.92	-
GoForth	<u>59392004</u>	<u>59.39</u>	-
DRAG	**37285**	**0.04**	-
RNAinverse-pf	-	-	-
RNA-IFlow	87548613	87.55	<u>86958021</u>
RNA-IFlow-RL	87548613	87.55	**14494917**
FastDesign	-	-	-
SAMFEO	-	-	-
NEMO	-	-	-

Two-stage training and regularization. The finite policy is initialized from the supervised RNA-IFlow predictor. Each logical policy update advances four target structures with two fresh rollout rounds at the recorded group size. Earlier backbone layers and the reference remain frozen. The final two backbone blocks and output head are updated using terminal advantages, clipped PPO, structured-reference KL, and clean-nucleotide CE. The reward assigns weights 0.5 to target probability and 0.25 to each MFE and uMFE indicator. The clipping threshold is 0.2, with KL coefficient 0.01 and

CE coefficient 0.1. Precise mathematical reductions are given above. The group-relative advantage vanishes at a singleton group, so no ordinary $G = 1$ policy score is reported.

A.5 Complete fixed-budget results and transfer

All rows below use the corresponding frozen eight-return evaluation records and the common final scorer. Native-search comparators retain their internal search costs. Prob. denotes best target probability; F1 denotes best pair-set F1; Div. is valid-sequence diversity. Each best-of-eight metric is aggregated separately. Complete fixed-budget results are reported in Tables 4, 5, 6, and 7.

Table 4: This table reports complete Eterna100-v2 results for eight returned candidates. Lower NED is better; all other metrics are higher-is-better.

Method	P@1	P@8	MFE@1	MFE@8	Prob.	F1	NED	Div.
RNA-Design-LM SL	0.2633	0.4333	0.2700	0.4433	0.3364	0.7735	0.1694	**0.9996**
RNA-Design-LM SL+RL	0.5133	0.6367	0.5233	0.6567	0.5256	0.9324	0.0652	0.9783
GoForth	0.3933	0.5700	0.4133	0.5833	0.3499	0.9236	0.0804	0.9862
DRAG	0.3633	0.5133	0.3633	0.5200	0.2391	0.9062	0.1345	0.8029
RNAinverse-pf	0.4367	0.5433	0.4533	0.5700	0.4344	0.6544	0.3625	0.6429
RNA-IFlow (Ours)	0.3033	0.5200	0.3133	0.5367	0.4089	0.8731	0.1017	0.9979
RNA-IFlow-RL (Ours)	**0.5400**	**0.6500**	**0.5433**	**0.6733**	**0.5398**	**0.9396**	**0.0534**	0.9363

Table 5: This table reports complete Eterna100 results for eight returned candidates. Lower NED is better; all other metrics are higher-is-better.

Method	P@1	P@8	MFE@1	MFE@8	Prob.	F1	NED	Div.
RNA-Design-LM SL	0.2633	0.4267	0.2700	0.4333	0.3326	0.7669	0.1702	**0.9996**
RNA-Design-LM SL+RL	0.5033	0.6133	0.5167	0.6333	0.5165	0.9196	0.0691	0.9812
GoForth	0.3933	0.5667	0.4133	0.5733	0.3435	0.9116	0.0848	0.9858
DRAG	0.3633	0.5133	0.3633	0.5167	0.2376	0.8960	0.1373	0.8063
RNAinverse-pf	0.4433	0.5100	0.4567	0.5267	0.4292	0.6442	0.3594	0.6400
RNA-IFlow (Ours)	0.3000	0.4867	0.3100	0.5067	0.3988	0.8596	0.1099	0.9975
RNA-IFlow-RL (Ours)	**0.5067**	**0.6167**	**0.5233**	**0.6467**	**0.5283**	**0.9247**	**0.0593**	0.9333

Table 6: This table reports complete Rfam-27 results for eight returned candidates. Lower NED is better; all other metrics are higher-is-better.

Method	P@1	P@8	MFE@1	MFE@8	Prob.	F1	NED	Div.
RNA-Design-LM SL	0.5185	0.7531	0.5185	0.7531	0.6200	0.9814	0.0267	**1.0000**
RNA-Design-LM SL+RL	0.8148	0.8642	0.8148	0.8642	0.7765	**0.9970**	0.0058	**1.0000**
GoForth	0.6296	0.7284	0.6296	0.7407	0.4878	0.9930	0.0162	**1.0000**
DRAG	0.7160	0.8272	0.7407	0.8272	0.4218	0.9962	0.0468	0.5170
RNAinverse-pf	0.3827	0.4568	0.3827	0.4568	0.4375	0.4932	0.5076	0.4398
RNA-IFlow (Ours)	0.6543	0.8272	0.6667	0.8395	0.7158	0.9944	0.0104	**1.0000**
RNA-IFlow-RL (Ours)	**0.8519**	**0.8765**	**0.8519**	**0.8765**	**0.7785**	0.9969	**0.0057**	0.9907

RNAsolo separates first-candidate quality from sample coverage: post-training improves the base flow's P@1 and ensemble metrics, while the base model retains a larger distinct-sequence fraction and slightly higher P@8. RNA-Design-LM SL+RL has the highest P@8 on this collection. Nine exact target-structure overlaps occur in the supervised corpus. Excluding those targets without using outcomes leaves 755 targets and preserves the observed ordering; the exclusion is based on exact target-structure identity.

For the paired Eterna100-v2 comparison at $K = 8$, the RNA-IFlow-RL minus RNA-Design-LM SL+RL success difference is 1.33 percentage points, with a target-bootstrap 95% interval of $[-2.67, 5.67]$. Targets are resampled with their evaluation conditions kept together. The interval includes zero. The RNAsolo overlap sensitivity analysis is reported in Table 8.

Table 7: This table reports complete RNAsolo-764 results for eight returned candidates. Lower NED is better; all other metrics are higher-is-better.

Method	P@1	P@8	MFE@1	MFE@8	Prob.	F1	NED	Div.
RNA-Design-LM SL	0.5951	0.7256	0.6069	0.7378	0.6418	0.9651	0.0347	**0.9999**
RNA-Design-LM SL+RL	**0.7199**	**0.7531**	**0.7356**	**0.7657**	**0.7156**	**0.9808**	**0.0154**	0.9774
GoForth	0.6252	0.7216	0.6440	0.7347	0.5858	0.9756	0.0274	0.9848
RNA-IFlow (Ours)	0.6165	0.7395	0.6339	0.7535	0.6890	0.9755	0.0203	0.9992
RNA-IFlow-RL (Ours)	0.7120	0.7312	0.7299	0.7544	0.7109	0.9783	0.0157	0.9186

Table 8: RNAsolo overlap sensitivity. The 755-target subset excludes the nine exact supervised target-structure overlaps without using model outcomes.

Method	Full P@8	755 P@8	Full NED	755 NED
RNA-Design-LM SL	0.7256	0.7223	0.0347	0.0351
RNA-Design-LM SL+RL	**0.7531**	**0.7501**	**0.0154**	**0.0156**
GoForth	0.7216	0.7183	0.0274	0.0274
RNA-IFlow	0.7395	0.7364	0.0203	0.0205
RNA-IFlow-RL	0.7312	0.7280	0.0157	0.0159

Table 9: This table reports overall performance as mean ± sample SD across three evaluation seeds. Bold and underlined means denote the best and second-best results, respectively.

Method	Type	Params (M)	Eterna100-v2		Eterna100		Rfam-27	
			P@1 ↑	P@8 ↑	P@1 ↑	P@8 ↑	P@1 ↑	P@8 ↑
RNA-Design-LM SL	SL	357.92	0.2633 ± 0.0231	0.4333 ± 0.0321	0.2633 ± 0.0231	0.4267 ± 0.0289	0.5185 ± 0.0370	0.7531 ± 0.0771
RNA-Design-LM SL+RL	SL+RL	357.92	0.5133 ± 0.0231	0.6367 ± 0.0115	0.5033 ± 0.0231	0.6133 ± 0.0058	0.8148 ± 0.0370	0.8642 ± 0.0214
GoForth	SL	59.39	0.3933 ± 0.0115	0.5700 ± 0.0000	0.3933 ± 0.0115	0.5667 ± 0.0058	0.6296 ± 0.0000	0.7284 ± 0.0214
DRAG	Learned search	0.04	0.3633 ± 0.0153	0.5133 ± 0.0058	0.3633 ± 0.0153	0.5133 ± 0.0058	0.7160 ± 0.0566	0.8272 ± 0.0214
RNAinverse-pf	Search	–	0.4367 ± 0.0058	0.5433 ± 0.0115	0.4433 ± 0.0208	0.5100 ± 0.0100	0.3827 ± 0.0566	0.4568 ± 0.0428
RNA-IFlow (Ours)	SL	87.55	0.3033 ± 0.0306	0.5200 ± 0.0000	0.3000 ± 0.0100	0.4867 ± 0.0416	0.6543 ± 0.0566	0.8272 ± 0.0214
RNA-IFlow-RL (Ours)	SL+RL	87.55	**0.5400** ± 0.0265	**0.6500** ± 0.0000	**0.5067** ± 0.0231	**0.6167** ± 0.0058	**0.8519** ± 0.0000	**0.8765** ± 0.0214

A.6 POLICY INTERFACE, HORIZON, AND GROUP-SIZE ANALYSES

The core policy comparison is reported in Table 10. Single- and three-seed H/G sensitivity, update-time profiling, native-flow step sensitivity, and post-training checkpoints are reported in Tables 11, 12, 13, 14, and 15, respectively.

Table 10: This table summarizes the policy interface and post-training results on Eterna100-v2.

Generator	P@1	P@8	Prob.	NED
Native RNA-IFlow	0.3033	0.5200	0.4089	0.1017
RNA-IFlow-no-RL	0.1233	0.2433	0.1822	0.2399
RNA-IFlow-RL	**0.5400**	**0.6500**	**0.5398**	**0.0534**

A shared $H = 8, G = 8$ control is used for both axes.

Table 11: This table reports complete single-seed policy sensitivity; group size one is the zero-advantage boundary.

Axis	Value	P@1	P@8	Prob.	NED
H	1	0.4533	0.6033	0.4821	0.0736
H	2	0.5000	**0.6567**	0.5312	0.0590
H	4	0.4900	0.6367	0.5258	0.0598
H	8	**0.5167**	0.6500	**0.5424**	**0.0558**
H	16	0.5000	0.6500	0.5317	0.0599
G	2	0.5167	0.6500	0.5312	0.0587
G	4	0.5100	0.6533	0.5339	**0.0553**
G	8	0.5167	0.6500	**0.5424**	0.0558
G	16	**0.5300**	**0.6600**	0.5364	0.0557
G	1	N/A	N/A	N/A	N/A

Table 12: Values are reported as means and sample standard deviations across three repeated sensitivity runs.

Configuration	P@1	P@8	Prob.	NED
$H = 2, G = 8$	0.5033 ± 0.0058	0.6511 ± 0.0096	0.5309 ± 0.0063	0.0593 ± 0.0014
$H = 8, G = 8$	0.5122 ± 0.0051	$\mathbf{0.6578 \pm 0.0107}$	$\mathbf{0.5391 \pm 0.0043}$	$\mathbf{0.0554 \pm 0.0016}$
$H = 8, G = 4$	0.5122 ± 0.0102	0.6544 ± 0.0051	0.5351 ± 0.0023	0.0562 ± 0.0008
$H = 8, G = 16$	$\mathbf{0.5178 \pm 0.0158}$	$\mathbf{0.6578 \pm 0.0019}$	0.5365 ± 0.0016	0.0557 ± 0.0013

A.7 DATA COMPOSITION, FIXED EXPOSURE, AND REGULARIZATION

The composition study matches target count, length distribution, initialization, and 352 logical updates. The coverage study fixes 4,144 target visits over 1,036 logical updates while changing the number of distinct targets. Both are single-training-seed studies. The original mixture and its display categories are described in Appendix A.2. Coverage, regularization, and rollout-freshness controls are summarized in Tables 17, 18, and 19. Rollout refresh aligns newly sampled trajectories with the latest policy. The snapshot variants collect distinct candidate batches from a fixed behavior snapshot within a logical update. The two- and three-round comparisons retain their corresponding rollout and optimizer budgets. Extra fresh rounds perform extra work, so accuracy differences across round counts describe the combined schedule.

A.8 TIMING AND NESTED CANDIDATE BUDGETS

Neural timing keeps the evaluated model resident on the same A100. All 100 Eterna100-v2 targets and their three fixed sampling conditions are measured twice with rotated method order. The generation and post-generation scoring intervals are measured separately. Candidate fingerprints were

Table 13: This table reports observed group-size update costs on shared GPUs; minimum and maximum values summarize profiling variability.

G	Median (s)	Minimum (s)	Maximum (s)
2	**25.7918**	10.5854	**34.0818**
4	27.4301	**10.4608**	35.9115
8	39.0918	11.9554	72.1897
16	56.9375	12.3717	302.8546

Table 14: This table reports native-flow integration-step sensitivity on 100 Eterna100-v2 targets.

Steps	P@1	P@8	MFE@8	Prob.
4	0.2800	0.5033	0.5033	0.3991
8	0.2900	0.5133	**0.5367**	0.4041
16	0.2967	0.5133	**0.5367**	0.4056
32	0.3000	0.5133	0.5333	**0.4095**
50	**0.3033**	**0.5200**	**0.5367**	0.4089

Table 15: This table reports checkpoint performance during post-training continuation.

Updates	P@1	P@8	MFE@8	Prob.
1744	0.5233	**0.6567**	**0.6867**	**0.5446**
2093	0.5333	0.6533	0.6767	0.5364
2442	**0.5400**	0.6500	0.6733	0.5398
2790	0.5300	0.6500	0.6800	0.5437

Table 16: This table compares matched-size training-data compositions.

Setting	P@1	P@8	Prob.	NED	Div.
Original mixture	**0.5133**	**0.6533**	**0.5363**	**0.0555**	**0.9513**
Core-only	0.4867	0.6333	0.5203	0.0616	0.9279
Hard-enriched	0.4900	0.6333	0.5248	0.0610	0.9379

Table 17: This table reports coverage under a fixed number of target visits.

Setting	P@1	P@8	Prob.	NED	Div.
1395 unique tasks	0.5000	**0.6500**	0.5361	0.0590	0.9071
2790 unique tasks	**0.5133**	0.6267	0.5269	0.0596	0.9175
4143 unique tasks	0.5033	0.6467	**0.5367**	**0.0565**	**0.9292**

Table 18: This table reports one-axis regularization sensitivity with all other settings fixed.

Setting	P@1	P@8	Prob.	NED	Div.
Lower CE weight	0.5133	0.6433	0.5332	0.0576	0.9487
Higher CE weight	0.5167	0.6467	0.5361	0.0582	**0.9517**
Lower KL weight	0.5167	**0.6533**	**0.5408**	**0.0546**	0.9413
Higher KL weight	**0.5233**	0.6433	0.5313	0.0561	0.9400

Table 19: This table reports rollout-freshness and reuse controls.

Setting	P@1	P@8	Prob.	NED
One fresh round	0.5133	0.6633	0.5425	0.0551
Two fresh rounds	0.5167	0.6500	**0.5424**	0.0558
Fixed behavior snapshot, two rounds	**0.5233**	**0.6533**	0.5396	**0.0557**
Three fresh rounds	0.5100	0.6467	0.5396	0.0558
Fixed behavior snapshot, three rounds	**0.5233**	**0.6600**	**0.5435**	**0.0554**

checked against the frozen eight-candidate ledger. Detailed return and scoring times are reported in Table 20. A search algorithm's time includes its internal folding and search, so the native rows below describe end-to-end candidate return rather than an equivalent neural forward pass. The nested-

Table 20: This table reports time to return eight candidates and time after common scoring. Neural entries use matched resident-model medians, while search rows retain their native timing scope.

Method	Return time (s)	With scoring (s)	Timed groups
RNA-Design-LM SL	4.4085	4.7356	600
RNA-Design-LM SL+RL	4.4085	4.6438	600
GoForth	1.3518	1.6502	600
DRAG	14.8637	15.0512	300
RNAinverse-pf	133.5565	133.7930	300
RNA-IFlow	0.8845	1.1684	600
RNA-IFlow-RL	**0.7317**	**0.9186**	600

budget experiment extends the frozen candidate prefix rather than resampling independently for each budget. A point is admitted only when every target and condition has been scored. Historical missing generation times remain unreported. Nested-prefix results for RNA-Design-LM SL, RNA-Design-LM SL+RL, GoForth, RNA-IFlow, and RNA-IFlow-RL are reported in Tables 21, 22, 23, 24, and 25. Identity-checked prefix-replay time plus new-block time is a separate measurement and is not used as a same-GPU cross-model speed comparison.

Table 21: This table reports RNA-Design-LM SL nested-prefix results from the same ordered candidate set, with each metric aggregated separately.

K	Pass@K	MFE@K	Prob.	NED	Pair-F1	Diversity
1	0.2633	0.2700	0.2136	0.2762	0.5899	**1.0000**
2	0.3433	0.3500	0.2618	0.2276	0.6662	**1.0000**
4	0.4033	0.4100	0.3024	0.1946	0.7313	**1.0000**
8	0.4333	0.4433	0.3364	0.1694	0.7735	0.9996
16	0.4800	0.4967	0.3719	0.1497	0.8187	0.9992
32	0.5167	0.5467	0.4064	0.1319	0.8539	0.9985
64	0.5667	0.5933	0.4304	0.1198	0.8758	0.9971
128	0.5900	0.6167	0.4502	0.1105	0.8936	0.9957
256	0.6067	0.6333	0.4663	0.1017	0.9101	0.9937
512	**0.6400**	**0.6633**	**0.4748**	**0.0950**	**0.9173**	0.9916

Table 22: This table reports RNA-Design-LM SL+RL nested-prefix results from the same ordered candidate set, with each metric aggregated separately.

K	Pass@K	MFE@K	Prob.	NED	Pair-F1	Diversity
1	0.5133	0.5233	0.4238	0.1099	0.8319	**1.0000**
2	0.5833	0.5900	0.4756	0.0855	0.8847	0.9950
4	0.6200	0.6333	0.5036	0.0743	0.9145	0.9883
8	0.6367	0.6567	0.5256	0.0652	0.9324	0.9783
16	0.6500	0.6800	0.5375	0.0582	0.9493	0.9683
32	0.6600	0.6967	0.5501	0.0530	0.9543	0.9581
64	0.6733	0.7100	0.5597	0.0503	0.9609	0.9443
128	0.7000	0.7367	0.5671	0.0479	0.9634	0.9308
256	0.7167	0.7567	0.5748	0.0454	0.9684	0.9168
512	0.7367	0.7700	0.5824	0.0437	0.9712	0.9016
1024	**0.7600**	**0.7967**	**0.5875**	**0.0419**	**0.9748**	0.8864

Table 23: This table reports GoForth nested-prefix results from the same ordered candidate set, with each metric aggregated separately.

K	Pass@K	MFE@K	Prob.	NED	Pair-F1	Diversity
1	0.3933	0.4133	0.2302	0.1357	0.8096	**1.0000**
2	0.4633	0.4800	0.2714	0.1101	0.8667	0.9933
4	0.5433	0.5533	0.3276	0.0895	0.9075	0.9900
8	0.5700	0.5833	0.3499	0.0804	0.9236	0.9862
16	0.5933	0.6200	0.3731	0.0744	0.9360	0.9831
32	0.6100	0.6300	0.3894	0.0693	0.9435	0.9793
64	0.6267	0.6400	0.4062	0.0656	0.9483	0.9736
128	0.6433	0.6567	0.4170	0.0625	0.9534	0.9668
256	0.6600	0.6800	0.4317	0.0597	0.9583	0.9595
512	0.6700	0.6900	0.4398	0.0573	0.9614	0.9523
1024	**0.6833**	**0.7067**	**0.4494**	**0.0554**	**0.9644**	0.9433

Table 24: This table reports RNA-IFlow nested-prefix results from the same ordered candidate set, with each metric aggregated separately.

K	Pass@K	MFE@K	Prob.	NED	Pair-F1	Diversity
1	0.3033	0.3133	0.2534	0.1880	0.6872	**1.0000**
2	0.3733	0.3900	0.3076	0.1477	0.7766	**1.0000**
4	0.4600	0.4800	0.3680	0.1224	0.8334	0.9975
8	0.5200	0.5367	0.4089	0.1017	0.8731	0.9979
16	0.5700	0.5967	0.4410	0.0894	0.9020	0.9960
32	0.5933	0.6167	0.4650	0.0818	0.9175	0.9931
64	0.6100	0.6367	0.4844	0.0756	0.9290	0.9904
128	0.6467	0.6767	0.5037	0.0699	0.9371	0.9875
256	0.6600	0.6900	0.5165	0.0657	0.9422	0.9852
512	0.6700	0.7100	0.5273	0.0611	0.9491	0.9821
1024	**0.6800**	**0.7200**	**0.5359**	**0.0584**	**0.9541**	0.9793

Table 25: This table reports RNA-IFlow-RL nested-prefix results from the same ordered candidate set, with each metric aggregated separately.

K	Pass@K	MFE@K	Prob.	NED	Pair-F1	Diversity
1	0.5400	0.5433	0.4483	0.0906	0.8527	**1.0000**
2	0.5867	0.5900	0.4854	0.0748	0.8906	0.9850
4	0.6233	0.6300	0.5127	0.0633	0.9190	0.9683
8	0.6500	0.6733	0.5398	0.0534	0.9396	0.9363
16	0.6633	0.6933	0.5517	0.0499	0.9510	0.9071
32	0.6833	0.7133	0.5626	0.0462	0.9597	0.8772
64	0.6967	0.7267	0.5725	0.0430	0.9644	0.8485
128	0.6967	0.7267	0.5804	0.0406	0.9673	0.8197
256	0.7133	0.7433	0.5849	0.0386	0.9719	0.7908
512	0.7167	0.7467	0.5891	0.0375	0.9739	0.7612
1024	**0.7233**	**0.7567**	**0.5925**	**0.0363**	**0.9768**	0.7337

A.9 NATIVE-SEARCH QUALITY AND COMPUTATION

FastDesign, SAMFEO, and NEMO are evaluated locally on Eterna100-v2 using 100 targets and three seeds, with a maximum native search time of 60 seconds per target and seed. All returned sequences are rescored with the common ViennaRNA 2.7.2 implementation. These methods may evaluate many internal candidates and return different numbers of designs. The eight-return main table and this fixed-time native-search comparison have separate budgets.

FastDesign uses the authors' fast configuration without optional rival-structure search. SAMFEO uses the documented development-branch implementation. NEMO retains its native energy settings, and all final metrics use the common scorer. Every planned target–seed group remains in the denominator. Native-search performance and completion statistics are reported in Tables 26 and 27. A group with no returned sequence has zero success, zero target probability, unit NED, and zero pair-set F1 by the declared failure convention. A timeout may still have returned candidates and is distinguished from an empty return. SAMFEO has a slightly higher uMFE success rate than

Table 26: This table reports native-search performance on Eterna100-v2 with at most 60 seconds per target and seed. Time is the observed mean search time, and all 300 groups contribute to the metrics.

Method	uMFE	MFE	Prob.	NED	Pair-F1	Run time (s)
FastDesign	0.4900	0.5267	0.3827	0.3589	0.6623	33.5269
SAMFEO	**0.6667**	**0.6933**	**0.5283**	**0.0579**	**0.9474**	43.3045
NEMO	0.2933	0.3167	0.1720	0.2928	0.6530	**15.9501**

Table 27: This table reports search completion and returned-candidate counts under the same native 60-second protocol. Returned counts describe the final candidate sets.

Method	Groups	Timeouts	Empty returns	Mean returned
FastDesign	300	93	93	3.4267
SAMFEO	300	172	**0**	9.9200
NEMO	300	**68**	48	0.8400

the eight-candidate RNA-IFlow-RL protocol, while RNA-IFlow-RL has higher target probability and lower NED. The comparison therefore describes a quality–compute trade-off. Figure 7 displays the runtime distributions and arithmetic means. Neural generation and native search retain their respective hardware and internal scoring costs. Internal oracle counts were not instrumented for the native runs.

A.10 PUBLISHED AND LARGE-BUDGET SOLUTION COVERAGE

Large-budget results complement the local 60-second search comparison. For RNA-IFlow-RL, the maximum budget is 10,000 candidates per target across sampling streams, with global stopping after uMFE success. Counts measure targets solved by any stream, with the stated candidate allowance serving as an upper limit. Large-budget coverage is summarized in Tables 28, 29, and 30. Published search counts preserve each source's folding engine, repeats, and budget. For FastDesign, the reported Eterna100 configuration uses 5,000 motif iterations, 2,500 root iterations, and cube-pruning size 90. Its published mean time is 284.1000 seconds on the authors' CPU system. SAMFEO's probability-objective experiment uses an ensemble size of 10, a temperature of 1, and 5,000 search iterations, with five independent runs. Its reported union of 74 uMFE solutions differs from its mean per-run count. NEMO's 77 uMFE solutions in the same comparison are also a five-run union (Zhou et al., 2023; 2026). SamplingDesign reports 76 uMFE solutions on Eterna100 under its continuous-search protocol (Tang et al., 2026). The reported coverage depends on the search budget and stopping rule.

A.11 REFERENCE, SELECTION RULE, AND STRUCTURAL INTERPRETATION

The main-text example is Eterna puzzle 8935338, puzzle 82 (Anemone), with a 214-nucleotide target. The reference is the author/player-designed sequence in the original "Sample Solution (V2/Vienna2)"

Table 28: This table reports Eterna100 large-budget coverage as solved targets out of 100. NEMO counts follow the SAMFEO authors' reproduction.

Method	MFE	uMFE	Budget / source
RNA-IFlow-RL	73	68	At most 10,000 candidates; global stop
RNA-Design-LM SL+RL	74	72	Published 10,000-sample evaluation (Gautam et al., 2026)
FastDesign	**80**	**78**	Published motif/cube search (Zhou et al., 2026)
SAMFEO	77	74	Five-run union, probability objective (Zhou et al., 2023)
NEMO	<u>79</u>	<u>77</u>	Five-run union reported by Zhou et al. (2023)

Table 29: This table reports additional large-budget generator coverage under each source's original stopping and aggregation protocol.

Benchmark	Method	MFE	uMFE	Protocol
Eterna100-v2	RNA-IFlow-RL	**81**	<u>76</u>	Maximum 10,000 candidates
Eterna100-v2	RNA-Design-LM SL+RL	**81**	**79**	Published 10,000 samples (Gautam et al., 2026)
Rfam-27	RNA-IFlow-RL	**24**	**24**	Maximum 10,000 candidates
Rfam-27	RNA-Design-LM SL+RL	**24**	**24**	Published 10,000 samples (Gautam et al., 2026)
RNAsolo-764	RNA-IFlow-RL	591	585	Maximum 10,000 candidates

Table 30: This table reports published RNAsolo-764 search coverage from the FastDesign comparison using the source paper's native search protocol.

Method	MFE solved	uMFE solved	Source
FastDesign	**611**	**606**	Zhou et al. (2026), Table 2
SAMFEO	608	602	Zhou et al. (2026), Table 2
NEMO	<u>609</u>	<u>605</u>	Zhou et al. (2026), Table 2

field and its paired "Secondary Structure V2" record. The sequence and structure are not cropped. The original record is available in the Eterna benchmarking repository[2]. It is a designed reference rather than a natural RNA sequence.

All five methods use the same target and condition with eight saved candidates. Within each method, the displayed candidate maximizes target probability, breaking ties by candidate index. The target is a post-hoc discordant case where RNA-IFlow-RL achieves uMFE success and the four displayed comparison methods have no target MFE hit within their corresponding groups. Population-level comparisons are reported in Tables 1 and 2. Each panel shows its own predicted MFE topology. Different sequence letters relative to the designed reference are marked separately from correct, alternative, or missing structure edges.

Table 31: This table reports displayed-candidate thermodynamics and group-level hit counts for Anemone. Reference metrics are evaluated independently and do not form a generated $K = 8$ group.

Sequence	Prob.	NED	MFE	uMFE	Group MFE	Group uMFE
Reference sequence	0.1803	0.0367	yes	yes	–	–
RNA-IFlow-RL	**0.5178**	**0.0128**	yes	yes	**2/8**	**2/8**
RNA-IFlow	0.1257	0.0773	no	no	0/8	0/8
RNA-Design-LM SL+RL	0.0769	0.0297	no	no	0/8	0/8
GoForth	0.0019	0.1477	no	no	0/8	0/8
DRAG	0.0007	0.0447	no	no	0/8	0/8

Exact sequences and structures. The numbered blocks below are contiguous segments of each sequence, followed by its displayed structure. The reference structure is the requested target. The complete strings and source identities are retained in the accompanying figure data.

Target (214 nt).

```
001–054 AAAGGUUCGGAAACGAGCCGAUGGGAAACCAUCGGUCAGGAAACUGGCCGAGGG
055–108 GAAACCCUCGCACAGGAAACUGUGCGUAACGAAAGUUACCGUGCGAAAGCACGC
109–162 GCCCCGAAAGGGGUGACUGGGGAAACCCGGGAUUAGGAAACUAAUCGGGCGGAA
163–214 ACGCCCGUUUUGGAAACAAGACGUUGGGAAACCAACGCGUCCGAAAGGACGC
```

Displayed structure:

```
...((((((....))))))((((((....))))))((((((....))))))(((((
....)))))((((((....))))))((((((....))))))((((((....))))))(
((((((....)))))).((((((....))))))((((((....))))))((((((...
.)))))((((((....))))))((((((....))))))((((((....))))))
```

RNA-IFlow-RL (214 nt).

```
001–054 AAAGCUGGCAAAAGCCAGCGCACCGAAAGGUGCGCCGACGAAAGUCGGCGCCGG
055–108 GAAACCGGCGCCCGCGAAAGCGGGCGGCCCGAAAGGGCCGGGCCGAAAGGCCCG
109–162 GGGCCGAAAGGCCCCAGGGGCGAAAGCCCCGCUGUCGAAAGACAGCGACUCGAA
163–214 AGAGUCGCGGGCGAAGGCCCGCGGUCCGAAAGGACCGCCGUCGAAGGACGGC
```

Displayed structure:

```
...((((((....))))))((((((....))))))((((((....))))))(((((
....)))))((((((....))))))((((((....))))))((((((....))))))(
((((((....)))))).((((((....))))))((((((....))))))((((((...
.)))))((((((....))))))((((((....))))))((((((....))))))
```

RNA-IFlow (214 nt).

```
001–054 AAAGCCGGCACUAGCCGGCGCGGCAUAAGCCGCGUCCCCGAAUGGGGACGGGGC
055–108 AAGGGCCCCGGGCGCAAAAGCGCCCGGGCCAAAAGGCCCGCCCCAAAAGGGGCG
109–162 ACACCAAAAGGUGUCACCGCCAGGUGGCGGGGCGGCAAAAGCCGCCGGCCCAAA
```

[2] https://github.com/eternagame/eterna100-benchmarking.

```
163-214 AGGGCCCAGCGCUCACGCGCUGGAUCCGAAAGGGUCGCGACCAUAAGGUCGC
```
Displayed structure:

```
...(((((((....)))))))(((((....((((((((((....)))))))))(((((
....)))))(((((((....)))))))(((((....)))))(((((....)))))(
(((((....)))))).(((((....)))))).)))))....)))))).(((((...
.)))))(((((((....)))))))(((((....)))))((((((....))))))
```

RNA-Design-LM SL+RL (214 nt).

```
001-054 AAAGCCCUCACAAGAGGGCCAUCCAAAAGGAUGACGAGGGUGACCUCGUGGGGC
055-108 AAAAGCCCCGAUCCCAAAAGGGAUUCAGCCAAAAGGCUGGACCCGAAAGGGUCG
109-162 CCAUGGAGACAUGGCACCGAGGAAACUCGGGCCCGCGAACGCGGGCGCCACACA
163-214 CGUGGCCCAUGGGAAACCGUGGCAACCCAAGGGUUGGGAUGGGAAACCAUCC
```
Displayed structure:

```
..(((((((....)))))))(((((....)))))(((((((....)))))))(((((
....)))))(((((((....)))))))(((((....)))))(((((....)))))(
(((((....)))))).(((((....)))))(((((((....)))))))((((((...
.)))))(((((((....)))))))..(((....))))).(((((((....)))))))
```

GoForth (214 nt).

```
001-054 AUUGGGUGCAUUAGCACCCGGGUGAAAUCACCCGUUGGGUAAUCCCGACGCUGU
055-108 AAUUACAGCGUUGGUAACAACCAACUUGGCACACGCCAAGCGGGUAAUCCCGCG
109-162 GGUGUAACUACACCCAGCUGGAUAUCCAGCGGGGUCAACAGACCCCGGGCUAUA
163-214 AAGCCCGUGUGCAGAAGCACACGUCGGAACUCCGACGCUUGGGAUGCCAAGC
```
Displayed structure:

```
...(((((((....)))))))(((((....)))))((((((.....((((((((((
....)))))(((((((....)))))))(((((....)))))(((((....)))))(
(((((....)))))).(((((....)))))(((((((....)))))))((((((...
.)))))(((((((....)))))))))))))...)))))))((((((....)))))))
```

DRAG (214 nt).

```
001-054 ACAGGGAGGGAUACCUCCCGGGCGAAAACGCCCGGGGGCCAAAGCCCCCGGGAG
055-108 AAAACUCCCGGUGGGAAAACCCACCGUGGGAAAACCCACUGGGGCAACCCCCAG
109-162 CGGGGAAAACCCCGCAGUGGCAAAAGCCACGGGGUGUUAACACCCCGGCGGUAA
163-214 UCCGCCGGGGGGAAAACCCCCCGGGGGGAAACCCCCGGCGGUAAAGACCGCC
```
Displayed structure:

```
.(.(((((((....)))))))(((((....)))))(((((((....)))))))(((((
....)))))(((((((....)))))))(((((....)))))(((((....)))))(
(((((....)))))).(((((....)))))(((((((....)))))))((((((...
.)))))(((((((....)))))))(((((....)))))))(((((....)))))).
```

Failure modes across the evaluated targets. A candidate can retain many target base pairs yet fail uMFE because an alternative complete structure has lower or equal energy. It can also attain uMFE while allocating appreciable equilibrium probability to near-optimal competitors. These cases motivate reporting both success and ensemble quality. The native-flow and finite-policy controls, paired uncertainty, and RNAsolo results above provide complementary failure analyses at the population level.

A.12 Finite-policy sampling and post-training procedure

The following procedure summarizes supervised training, post-training, and inference.

1. Fit the clean-nucleotide predictor to supervised sequence–structure pairs along the Dirichlet path. Initialize the refinement model with these learned parameters.
2. For a target, initialize each unpaired unit uniformly over four nucleotides and each paired unit uniformly over the six legal pair states.
3. At each policy step, map the current discrete state to the Dirichlet conditional mean. Obtain temperature-scaled nucleotide probabilities, construct the SPD unit distributions, and sample the keep-or-resample transition independently across units conditional on the complete current sequence. Record the behavior probability of each executed transition.
4. After all policy steps, score the completed sequences with target probability, MFE success, and uMFE success. Standardize rewards within each trajectory group and assign the resulting terminal advantage to every transition of that trajectory.
5. Compute each step's joint current-to-behavior ratio across all units. Apply the clipped surrogate with a sum over policy steps and an average over trajectories, together with structured-reference KL and the supervised CE anchor.
6. Refresh rollout data for subsequent policy updates. During inference, execute the finite policy directly for each requested candidate, without reward evaluation or score-based reranking.

A.13 Additional recovered and residual examples

Two further examples use the first candidate at fixed seed 1009. Within the strata "native flow fails and post-training succeeds" and "both fail", the target of median length is selected, breaking ties by target identifier. The resulting 104- and 108-nucleotide targets were chosen without maximizing improvement. These are saved full-pipeline outputs, so the native-to-refined comparison changes both weights and generation dynamics. Figure 8 shows these additional designs, and Table 32 reports their thermodynamic metrics.

Table 32: This table reports additional fixed-first-candidate thermodynamics; small nonzero probabilities are retained in scientific notation.

Target	nt	Model	Prob.	NED	uMFE
Ete_19	104	RNA-IFlow	0.0023	0.0647	no
Ete_19	104	RNA-IFlow-RL	**0.9467**	**0.0022**	yes
Ete_50	108	RNA-IFlow	5.5166×10^{-7}	0.3343	no
Ete_50	108	RNA-IFlow-RL	0.0006	**0.2935**	no

A.14 Historical training-signal profiles

The archived profiling experiment evaluates eight frozen-supervised-flow candidates for each of 12,617 training-pool targets. Its all-or-nothing (AoN) quantity is the fraction of candidates meeting the uMFE criterion, not a mean target probability. Its normalized structural distance (NSD) is the source evaluator's structure-distance function divided by target length, not a coefficient of variation. The visual summary retains the 12,613 targets for which all eight candidate records are valid. Four targets contain a total of 13 invalid records and are excluded from this distribution plot. This profiling collection was not itself used to select the final 2,790-target training mixture. Figure 9 summarizes the corresponding empirical training-signal profiles.

A.15 Recorded training progress and complementary outcomes

Gray points show per-update observations and the blue curve shows non-overlapping 32-update means.

Rewards are measured on the changing training-target stream. The complementary histogram in Figure 10 displays the candidate-level target-probability frequencies underlying this distributional comparison. Schedule and reference-anchor controls are reported in Appendix A.6.

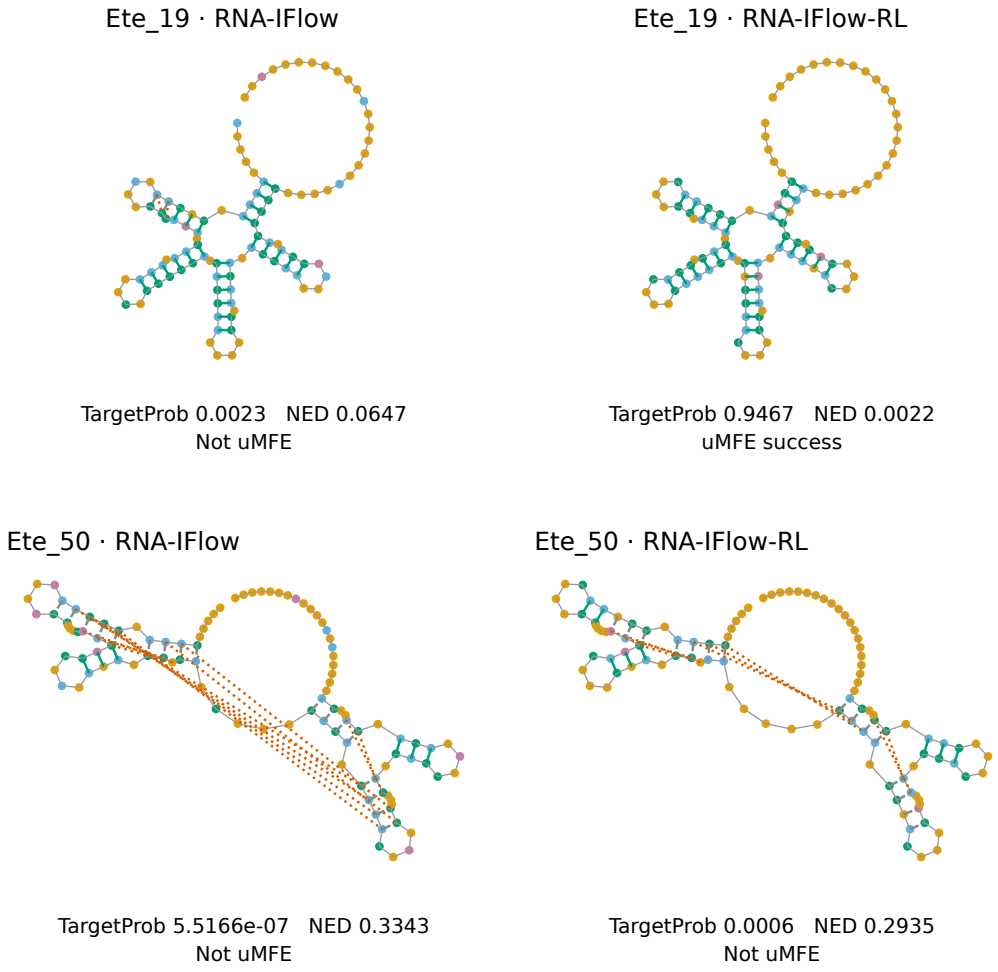

Figure 8: This figure shows additional first-candidate designs, with a recovered target above and a residual uMFE failure below. Green, orange, and dashed gray bonds denote recovered, alternative, and missing pairs, while nucleotide colors denote A/C/G/U in orange/blue/green/purple.

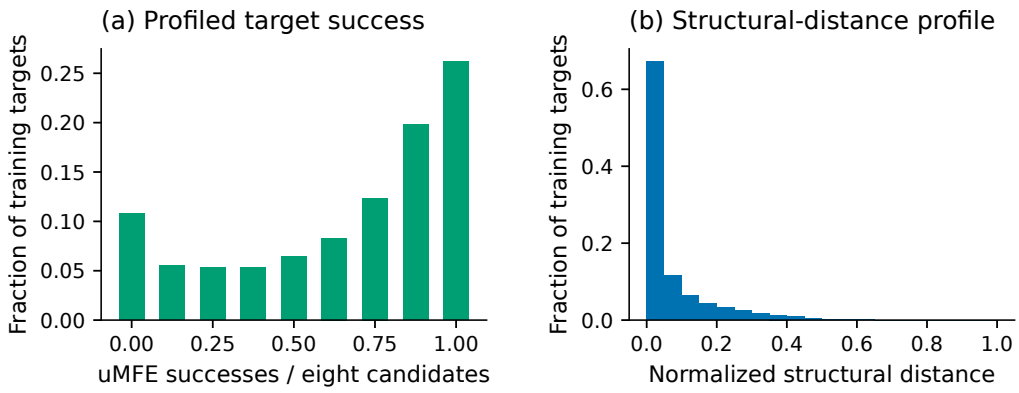

Figure 9: This figure shows frozen-supervised-flow profiles over 12,613 training-pool targets, each with eight valid candidates. Panels (a,b) report target fractions by empirical uMFE success and mean normalized structural distance, respectively.

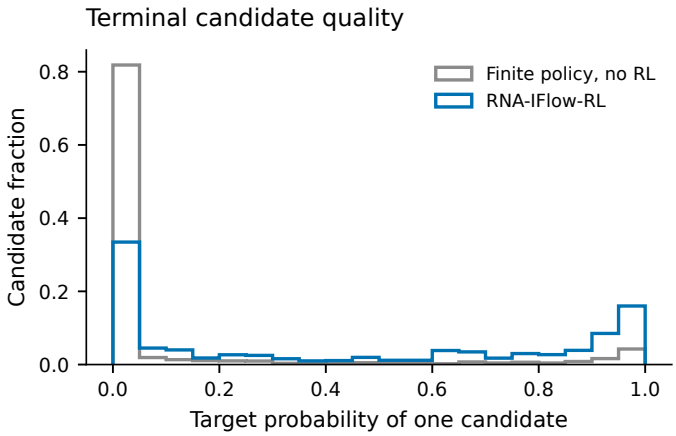

Figure 10: This figure compares candidate-level target-probability histograms before and after RL. Gray and blue outlines denote the finite policy without RL and RNA-IFlow-RL, while the vertical axis reports the candidate fraction in each probability bin.

A.16 Training-data length composition

The supervised and RL training-set length distributions are shown in Figure 11. The matched-composition controls hold target counts and length-bin counts fixed; coverage controls instead hold total target visits fixed.

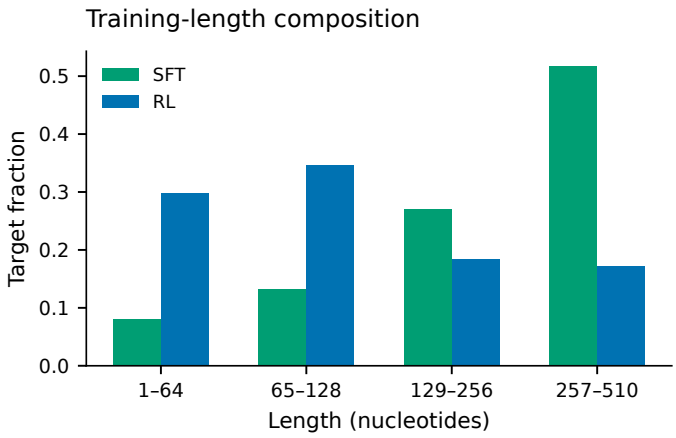

Figure 11: This figure shows training-set length distributions. Bars report sample fractions in each length interval, with green and blue denoting supervised and RL training data, respectively.

A.17 Case-display window and full-length sequence recovery

The structural case retains the original 214-nucleotide reference and all five generated candidates. Figure 4 displays the same central 50 positions, 83–132, for every sequence. The structures, candidate selection, target probability, NED, and MFE/uMFE status all use complete sequences. The display window is fixed across methods.

For the author-designed reference x^{ref}, sequence recovery is the fraction of identical bases at aligned positions,

$$\mathrm{Recovery}(x, x^{\mathrm{ref}}) = \frac{1}{L} \sum_{j=1}^{L} \mathbf{1}[x_j = x_j^{\mathrm{ref}}]. \tag{21}$$

This measure characterizes resemblance to the player-designed reference. Folding success is evaluated separately. Structurally valid alternative sequences can differ substantially from the reference. Full-length recovery values are summarized in Table 33.

Table 33: This table reports full-length recovery for the displayed Anemone candidates using the same 214-nucleotide reference throughout.

Sequence	Matching positions	Recovery (%)
Author/player reference	214/214	100.00
RNA-Design-LM SL+RL	87/214	40.65
GoForth	88/214	41.12
DRAG	101/214	47.20
RNA-IFlow	97/214	45.33
RNA-IFlow-RL	**114/214**	**53.27**

A.18 Related work in context

RNA inverse folding has developed along three directions: target-specific search, learned optimization, and conditional generation. Search methods exploit structural constraints and thermodynamic objectives, as in NEMO, SAMFEO, SamplingDesign, and FastDesign (Portela, 2018; Zhou et al., 2023; Tang et al., 2026; Zhou et al., 2026). LEARNA and DRAG learn reusable construction or mutation policies, while RNA-Design-LM and GoForth generate sequences conditioned on structural specifications (Runge et al., 2019; Li et al., 2025; Gautam et al., 2026; Lindsey, 2026).

Flow Matching learns probability transport, with Dirichlet and discrete formulations extending it to categorical sequences (Lipman et al., 2023; Stark et al., 2024; Gat et al., 2024). RNAFlow and RiboFlow apply flow-based models to RNA sequence–structure co-design (Nori & Jin, 2024; Ma et al., 2025). Reward-based methods such as DRAKES and flow-policy optimization further adapt generators to downstream objectives (Wang et al., 2025; McAllister et al., 2026; Su et al., 2026). RNA-IFlow-RL brings these perspectives together under a fixed secondary-structure condition: global sequence variation is refined through a finite policy that coordinates paired actions and learns from terminal folding feedback.

A.19 Discussion and future directions

Our framework separates structural compatibility from thermodynamic preference. Coordinated sequence updates explore pairing-compatible candidates, while terminal feedback favors complete sequences whose folding ensembles support the target. The finite policy connects these two levels of design. The coverage and native-search analyses also show the importance of assessing success together with diversity and computational cost. Extending the feedback to alternative energy models and experimentally measured properties is a natural direction for testing whether the learned preferences transfer to biological settings.