RNAFlow: RNA Structure & Sequence Design via Inverse Folding-Based Flow Matching

Divya Nori [1] **Wengong Jin** [2]

Abstract

The growing significance of RNA engineering in diverse biological applications has spurred interest in developing AI methods for structure-based RNA design. While diffusion models have excelled in protein design, adapting them for RNA presents new challenges due to RNA's conformational flexibility and the computational cost of fine-tuning large structure prediction models. To this end, we propose RNAFlow, a flow matching model for protein-conditioned RNA sequence-structure design. Its denoising network integrates an RNA inverse folding model and a pre-trained RosettaFold2NA network for generation of RNA sequences and structures. The integration of inverse folding in the structure denoising process allows us to simplify training by fixing the structure prediction network. We further enhance the inverse folding model by conditioning it on inferred conformational ensembles to model dynamic RNA conformations. Evaluation on protein-conditioned RNA structure and sequence generation tasks demonstrates RNAFlow's advantage over existing RNA design methods.

1. Introduction

In recent years, RNA molecules have been engineered for versatile and controllable functions in biological systems (Dykstra et al., 2022), ranging from synthetic riboswitch sensors that modulate gene expression to aptamers that bind with protein targets (Thavarajah et al., 2021; Vezeau et al., 2023; Keefe et al., 2010). However, experimental methods for high-throughput RNA selection like SELEX (Gold, 2015) remain time-consuming and labor-intensive. To un-

lock the full potential of RNA therapeutics, there is a pressing need for developing deep learning models to automatically design RNAs that bind with a protein of interest (Sanchez de Groot et al., 2019).

Current state-of-the-art biomolecular structure design methods are based on diffusion models (Ho et al., 2020; Yim et al., 2023b) or flow matching (Lipman et al., 2022; Bose et al., 2023). For example, RFDiffusion (Watson et al., 2023) fine-tuned RoseTTAFold (Baek et al., 2021) to generate novel protein structures that meet complex constraints, including conditional generation of protein binders. While we can adopt a similar approach for conditional RNA structure generation by fine-tuning a protein-nucleic acid structure prediction network like RoseTTAFold2NA (RF2NA) (Baek et al., 2023), there are two major caveats. First, RNAs exhibit a high degree of conformational flexibility, which is often the key to their biological function (Ganser et al., 2019). A generative model that outputs a single static structure may bottleneck the downstream sequence generation process, since an ideal sequence should account for desired RNA dynamics. Second, fine-tuning a large structure prediction network is computationally expensive, motivating a method that applies a pre-trained model like RF2NA directly.

In this paper, we propose RNAFlow, a flow matching model for RNA sequence-structure design. The denoising network in RNAFlow is composed of an RNA inverse folding model and a pre-trained RF2NA network. In each iteration, RNAFlow first generates a RNA sequence given a noisy protein-RNA complex and then uses RF2NA to fold into a denoised RNA structure. For computational efficiency, we train the inverse folding model to minimize the flow matching objective while keeping RF2NA fixed. RNAFlow offers three advantages over previous methods like RFDiffusion. First, RNAFlow generates an RNA sequence and its structure simultaneously. Second, it is much easier to train because we do not fine-tune a large structure prediction network. Third, our framework enables us to model the dynamic nature of RNA structures for inverse folding. Specifically, we utilize the final few structure predictions from inference trajectories as an effective approximation of an RNA conformational ensemble and enhance our inverse folding model to condition on dynamic RNA conformations.

[1]Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA, USA [2]Broad Institute of MIT and Harvard, Cambridge, MA, USA. Correspondence to: Divya Nori <divnor80@mit.edu>.
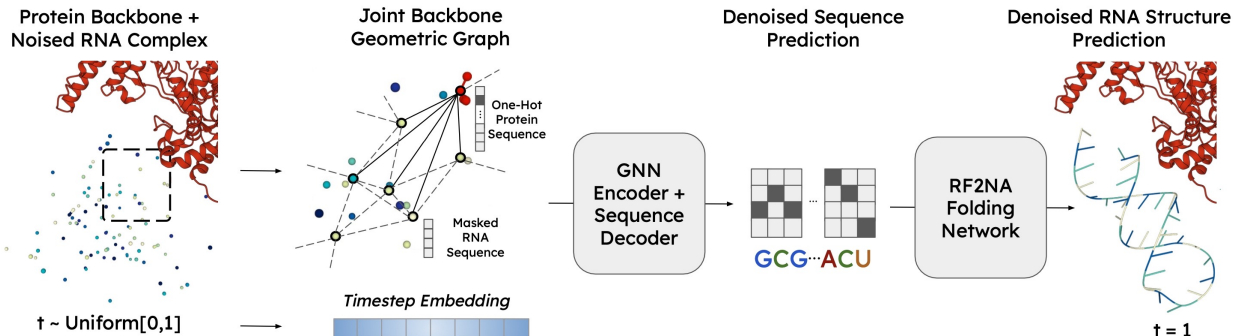
1

Figure 1. One forward pass in RNAFlow training, during which an inverse folding model is finetuned to be the flow matching score prediction network. The inverse folding model predicts a denoised RNA sequence from a noised complex backbone graph. The predicted sequence is folded by RF2NA for sequence and structure supervision.

We evaluate our method on the task of protein-conditioned RNA structure and sequence generation. RNAFlow outperforms a standard sequence-only approach and a recent diffusion model (Morehead et al., 2023) for nucleic acid sequence-structure generation in terms of native sequence recovery, RMSD, and lDDT. Additionally, we show that RNAFlow can be used in the motif-scaffolding setting to generate plausible RNA aptamers for G-protein-coupled receptor kinase 2 (GRK2), a target with known sequence motif for GRK2 binding.

2. Related Work

Protein-Conditioned RNA Design. Early methods for computational design of protein-binding RNAs involved generating a large number of RNA sequences and selecting by desired secondary structure motifs (Kim et al., 2010) or molecular dynamics (Zhou et al., 2015; Buglak et al., 2020). These approaches are computationally expensive and require specifying design constraints a priori, which are often unknown for new protein targets. More recently, generative sequence-based approaches based on LSTMs and VAEs have been trained on SELEX data (Im et al., 2019; Iwano et al., 2022). However, these methods can only be trained for one protein at a time and cannot be applied to proteins where SELEX data does not exist.

Unconditional RNA Design. Computational methods have been developed for RNA structure design, including classical algorithmic approaches (Yesselman et al., 2019) and generative modeling methods. In particular, Morehead et al. (2023) recently proposed an $SE(3)$-discrete diffusion model (MMDiff) for joint generation of nucleic acid sequences and structures. While MMDiff can generate short micro-RNA molecules, it has trouble conditionally generating protein-binding RNAs and samples of longer sequence length.

Deep learning methods for inverse folding generally apply

graph-based encoders for a single RNA structure (Tan et al., 2023) or several conformers (Joshi et al., 2023). We adapt an inverse folding model to accept protein information as a condition and train on a denoising objective.

Protein Structure Design. Deep generative methods for protein design have demonstrated that realistic protein backbones can be generated efficiently via flow matching (Yim et al., 2023a; Bose et al., 2023). Prior work has also shown that diffusion models can be conditioned on a target protein to generate functional protein binders (Ingraham et al., 2023; Watson et al., 2023). We apply a similar approach within the flow matching framework. Our work is also related to sequence-structure co-design methods where the outputs are iteratively refined (Jin et al., 2021; Stark et al., 2023).

3. Methods

Problem Formulation. As input, RNAFlow receives the protein backbone atom structure $\vec{P} \in \mathbb{R}^{L_p \times 3 \times 3}$, where L_p is the number of residues, and each residue contains backbone atoms N, C_α, and C. The protein sequence is also given as input where a single token is p_i, $i \in \{0, 1, 2, \ldots, L_p - 1\}$. The model is trained to predict RNA sequence with tokens r_i, $i \in \{0, 1, 2, \ldots, L_r - 1\}$. The model also predicts RNA backbone structure $R \in \mathbb{R}^{L_r \times 3 \times 3}$, where L_r is the number of nucleotides, and each nucleotide contains backbone atoms P, C'_4, and N_1/N_9 (pyrimidine/purine) (Wadley et al., 2007).

Background: Conditional Flow Matching. By the flow matching framework (Lipman et al., 2022), consider data distribution p_1 of RNA backbone structures and a prior distribution $p_0(\vec{R}|\vec{R}_1)$, where $\vec{R}_1$ is a sample from p_1. A *flow* transforms p_0 to p_1, and this flow can be parameterized by a sequence of time-dependent conditional probability paths $p_t(\vec{R}_t|\vec{R}_1)$, $t \in [0, 1]$. A sample from p_t can be computed by linear interpolation, as given by Equation 1, where $\vec{R}_0$ is

a sample from the prior distribution.

$$\vec{R}_t | \vec{R}_1 = (1 - t) * \vec{R}_0 + t * \vec{R}_1 \quad (1)$$

This probability path is generated by a marginal vector field that defines how individual samples are transported. We can define this vector field as given in Equation 2, and we can integrate over the field from time 0 to time 1 to generate samples from p_1 given a noise sample from p_0.

$$\hat{v}(\vec{R}_t; \theta) = \frac{\hat{R}_1(\vec{R}_t; \theta) - \vec{R}_t}{1 - t} \quad (2)$$

We aim to learn this vector field by approximating $\hat{R}_1$ with a neural network trained with the reparameterized conditional flow matching objective. Particularly, as with diffusion, the neural network can conveniently be trained to predict samples from the data distribution given noised sample $\vec{R}_t$.

We choose to employ flow matching rather than the related diffusion framework (Ho et al., 2020), primarily due to computational efficiency. Diffusion inference often requires thousands of forward passes to generate quality samples. Since our score model involves a large structure prediction network, RNAFlow inference time would be much slower in a diffusion setting (Yim et al., 2023a). In contrast, quality samples can be generated by flow matching with a much fewer number of passes.

3.1. RNAFlow Algorithm: Overview

We generate RNA sequences and structures using a conditional flow matching model where the score predictor is an inverse folding denoiser and pre-trained folding network. The inverse folding denoiser, which we refer to as **Noise-to-Seq**, is a geometric graph neural network conditioned on protein structure and sequence. Noise-to-Seq is pre-trained on the RNA inverse folding task and subsequently fine-tuned on the flow matching objective. It follows the RNA inverse folding architecture presented by Joshi et al. (2023), which is detailed in section 3.2.

Initialization. As our prior distribution, we choose a unit Gaussian on $\mathbb{R}^3$ centered at zero, from which we sample $\vec{R}_0$. We also translate the true protein-RNA complex $[\vec{P}_1, \vec{R}_1]$ such that the center of mass of $\vec{R}_1$ is zero. As shown in Algorithm 1, we Kabsch align (Kabsch, 1976) the sampled noise $\vec{R}_0$ with the true RNA $\vec{R}_1$ to eliminate global rotations, leading to faster training convergence (Klein et al., 2023). By this approach, we train with the ground-truth pose of the protein relative to the RNA centroid, though this is not leaked at inference time.

Training. During training, we sample a timestep t and interpolate the true RNA backbone with the sampled prior to

---

Algorithm 1 RNAFlow: Train

Require: $\{p_i\}_{\forall i}, \{r_i\}_{\forall i}, [\vec{P}_1, \vec{R}_1]$
    Sample prior $\vec{R}_0 \sim \mathcal{N}(0, I_3)^{L_r}$
    Kabsch align noise $\vec{R}_0$ with true backbone $\vec{R}_1$
    Sample timestep $t \sim \text{Uniform}[0, 1]$
    Interpolate $\vec{R}_t \leftarrow t * \vec{R}_1 + (1 - t) * \vec{R}_0$
    Predict $\{\hat{r}_i\}_{\forall i} \leftarrow$ **Noise-to-Seq**$\{[\vec{P}_1, \vec{R}_t], \{p_i\}_{\forall i}, t\}$
    Predict $\hat{R}_1 \leftarrow \text{RF2NA}\{\{\hat{r}_i\}_{\forall i}\}$

---

arrive at noised backbone $\vec{R}_t$. The true protein backbone, noised RNA backbone, and protein sequence are given as input to Noise-to-Seq which predicts a denoised RNA sequence. RF2NA folds the predicted RNA sequence into predicted structure $\hat{R}_1$. This process is shown in Figure 1.

The outputs of an RNAFlow training pass are an RNA sequence and its structure. The reparameterized flow matching objective gives that the score model should be trained with a standard L_2 loss. Therefore, as given in Equation 3, we compute the MSE between all predicted and true RNA backbone coordinates. Before MSE computation, we Kabsch align the two RNA backbones, since we are optimizing for RNA structure design rather than docking accuracy. Additionally, we supervise the sequence with a cross entropy loss between predicted and true nucleotide types.

$$\mathcal{L} = \text{MSE}(\hat{R}_1, \vec{R}_1) + \sum_i \text{CE}(\hat{r}_i, r_i) \quad (3)$$

The sequence output from Noise-to-Seq is used as input to RF2NA, and the resulting structure is supervised to update the weights of the model. Since gradients must propagate between Noise-to-Seq and RF2NA, we apply the Gumbel-Softmax estimator to differentiably sample from the predicted logits (Jang et al., 2016). Further justification of this objective is given in the Appendix.

Inference: RNAFlow-Base. As shown in Algorithm 2, we begin by generating a "pose guess" using RF2NA. Specifically, we fold the protein MSA with a mock RNA sequence consisting of all adenines, of the same length as the true RNA sequence. The predicted complex is used as an initial guess of protein pose with respect to the RNA centroid. $\vec{P}_0$ is the true protein backbone Kabsch-aligned onto the predicted protein pose, and $\vec{R}_0$ is drawn from the prior distribution. RNAFlow refines its predictions over multiple steps, predicting the true complex $[\hat{P}_1, \hat{R}_1]$ on each iteration. The final outputs of RNAFlow inference are a predicted RNA sequence and structure.

Inference: RNAFlow-Traj. In standard RNAFlow inference, Noise-to-Seq predicts a sequence based on a single

RNA structure. However, an RNA backbone structure can adopt multiple conformations, and it is important that the final RNA sequence reflects this conformational diversity (Ganser et al., 2019). Indeed, Joshi et al. (2023) demonstrates that inverse folding on a set of RNA conformations improves the native sequence recovery rate compared to single-structure model. To this end, we propose to generate a final RNA sequence by conditioning on multiple RNA structures generated over the course of flow matching inference. This inverse folding model, referred to as **Trajectory-to-Seq (Traj-to-Seq)**, is a multi-graph neural network which can handle multiple RNA conformation inputs. Its architecture is detailed in section 3.3.

Algorithm 2 RNAFlow-Traj: Inference

Require: $\{p_i\}_{\forall i}, \vec{P}_1$

   Position $\vec{P}_0$ by aligning $\vec{P}_1$ with RF2NA pose guess

   Sample prior $\vec{R}_0 \sim \mathcal{N}(0, I_3)^{L_r}$

   Initialize $traj \leftarrow []$

   for $n \leftarrow 1$ **to** N **do**

      Let $t_2 \leftarrow n/N$ and $t_1 \leftarrow (n-1)/N$

      $\{\hat{r}_i\}_{\forall i} \leftarrow$ **Noise-to-Seq**$\{[\vec{P}_{t_1}, \vec{R}_{t_1}], \{p_i\}_{\forall i}, t_1\}$

      Predict $[\hat{P}_1, \hat{R}_1] \leftarrow$ RF2NA$\{\{\hat{r}_i\}_{\forall i}, \text{protein MSA}\}$

      Append $\hat{R}_1$ to $traj$

      Interpolate $\vec{R}_{t_2} \leftarrow \vec{R}_{t_1} + \frac{(\hat{R}_1 - \vec{R}_{t_1})}{(1-t_1)} * (t_2 - t_1)$

      Align $\vec{P}_{t_2} \leftarrow$ Kabsch$(\vec{P}_1, \hat{P}_1)$

   end for

   Predict $\{\hat{r}_i\}_{\forall i} \leftarrow$ **Traj-to-Seq**$\{traj\}$

   Outputs: $\{\hat{r}_i\}_{\forall i}, traj[-1]$

3.2. Noise-to-Seq Module

Noise-to-Seq is a graph-based RNA inverse folding model, fine-tuned to predict RNA sequences autoregressively from noised structures. As in Ingraham et al. (2019) and Jing et al. (2021), the model applies an encoder-decoder architecture where the encoder learns a representation of the protein-RNA complex structure and the decoder predicts a distribution over nucleotides at each position i, given sequence information of all positions before i.

Graph Representation. We represent each backbone 3D point cloud $[\vec{P}, \vec{R}]$ as a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. Each amino acid i is assigned a node at C_α coordinate $\vec{x}_i \in \vec{P}$, and each nucleotide j is assigned a node at C'_4 coordinate $\vec{x}_j \in \vec{R}$. Node features are denoted as $\mathcal{V} = \{v_1, \dots, v_{L_p+L_r}\}$. As shown in Figure 1, for all nodes $\vec{x}_i \in \vec{P}$, we draw edges to 10 nearest neighbor nodes whose 3D coordinates are also in $\vec{P}$. Likewise for all nodes $\vec{x}_i \in \vec{R}$, we draw edges to 10 nearest neighbor nodes whose 3D coordinates are also in $\vec{R}$. For all nodes $\vec{x}_i \in \vec{P}$, we additionally draw edges

to 5 nearest neighbor nodes $\vec{x}_j \in \vec{R}$. Nearest neighbors are computed by $\|\vec{x}_i - \vec{x}_j\|_2$. Edge features are denoted as $\mathcal{E} = \{e_{i,j}\}_{i \neq j}$.

Node & Edge Features. We compute node features and edge features as follows. Node vector features include unit vectors to neighboring nodes on the backbone chain and unit vectors to the remaining two atoms of the nucleotide (P and N_1/N_9) or amino acid (N and C). Node scalar features include magnitudes of all vector features, encoded by radial basis functions, and a one hot encoding of residue identity for the protein chain. Edge vector features include a unit vector from the source node to the destination node, and its magnitude encoded by radial basis functions is a scalar feature. Edge scalar features also include distance along the backbone between the source and destination, encoded with a sinusoidal embedding. If the edge connects nodes in two different chains, the encoded number is equal to $L_p + L_r$.

Model Architecture. Noise-to-Seq contains a message-passing encoder consisting of GVP-GNN layers (Jing et al., 2020). First, input node features v_i and edge features $\{e_{i,j}\}_{i \neq j}$ are encoded by GVPs.

$$h_{v_i} = g_v(\text{LayerNorm}(v_i))$$
$$h_{e_{i,j}} = g_e(\text{LayerNorm}(e_{i,j}))$$

Node embedding GVP g_v and edge embedding GVP g_e apply the vector gating strategy proposed by Jing et al. (2021). If the input structure is noised, we add a timestep embedding to the node features, encoded by random Fourier features of size 256 (Tancik et al., 2020) and concatenated onto the scalar component of h_{v_i}. For every position i, h_{v_i} is residually updated by a sequence of three message-passing layers where each layer has the following architecture:

$$m_{v_i} = \frac{1}{|\mathcal{N}|} \sum_{j \in \mathcal{N}} g_{MSG}(h_{v_i}, h_{v_j}, h_{e_{i,j}})$$
$$h'_{v_i} = \text{LayerNorm}(h_{v_i} + \text{Dropout}(m_{v_i}))$$

g_{MSG} is a sequence of three GVPs with vector gating and ReLU activation on the scalar features. After each message-passing layer, we also apply pointwise feedforward updates to the node embedding.

$$h'_{v_i} = \text{LayerNorm}(h'_{v_i} + \text{Dropout}(g_{FF}(h'_{v_i})))$$

g_{FF} is a sequence of two GVPs with vector gating and the ReLU activation on the scalar features. We then encode protein and RNA sequence information with an embedding layer and concatenate these features onto the edge scalar features. By the autoregressive scheme, we mask out sequence features for edges where the source node's position is greater than the destination node's position. The protein nodes are always positioned before RNA nodes, so protein sequence context is always present during decoding. Our
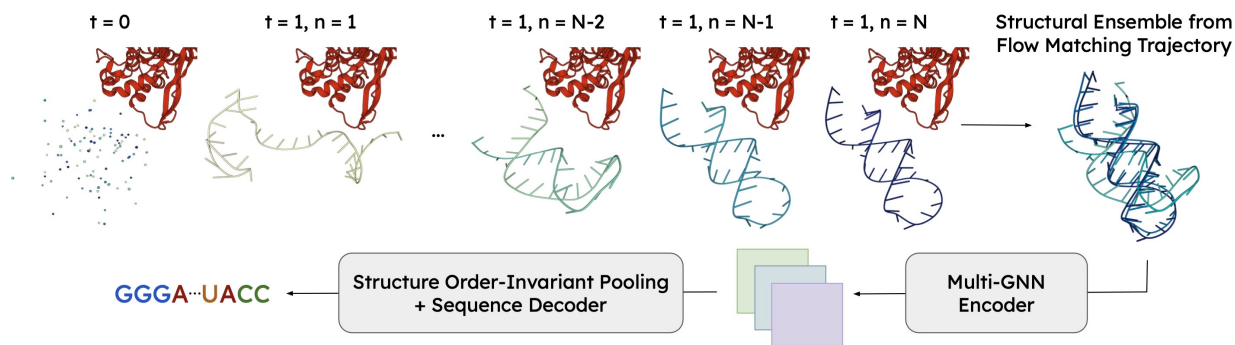
Figure 2. One forward pass during Traj-to-Seq inference. A subset of structures from a flow matching trajectory are encoded by a multi-GNN and pooled in an order-invariant manner to predict an RNA sequence.

decoder's architecture is identical to the encoder, except that messages are aggregated by a sum instead of a mean. Finally, we apply a GVP layer and softmax to predict probabilities of the 4 nucleotide classes at each position. The model is supervised by a cross entropy loss between the true RNA sequence and predicted nucleotide probabilities.

3.3. Traj-to-Seq Module

As shown in Figure 2, the flow matching inference loop generates a trajectory of RNA structures over iterative refinement steps. Traj-to-Seq is a graph-based inverse folding model that predicts RNA sequences based on the output trajectory of a single RNAFlow inference pass. At the final stages of the trajectory, the structures have a high degree of secondary structure similarity while displaying variation in tertiary structure, qualitatively resembling a conformational ensemble (Fornili et al., 2013). Therefore, we can leverage the outputs of a single flow matching inference pass as a conformational ensemble approximation.

Model Inputs. We represent a trajectory of RNA backbones as a set of independent graphs $\{\mathcal{G}^{(1)}, ..., \mathcal{G}^{(k)}\}$ where k is the number of conformers. Traj-to-Seq does not accept protein information as input, so each node in $\mathcal{G}^{(n)}$ corresponds to a nucleotide j at C'_4 coordinate $\vec{x}_j \in \vec{R}^{(n)}$. For all nodes $\vec{x}_j \in \vec{R}^{(n)}$, we draw edges to 10 nearest neighbor nodes whose 3D coordinates are also in $\vec{R}^{(n)}$, so edges are not drawn between conformers. Instead, Traj-to-Seq is designed to operate on "multi-graphs." As introduced by Joshi et al. (2023), a multi-graph is constructed by stacking each RNA graph's scalar and vector features and building a joint adjacency matrix, computed as the union of each individual graph's adjacency matrix.

Architecture. Traj-to-Seq employs the same encoder-decoder architecture as Noise-to-Seq. The GVP encoder processes each structure independently, such that $h_{v_i} \in \mathbb{R}^{k \times f}$ where k is the number of conformers and f is the feature

dimension. A merged representation is computed by a structure order-invariant mean. Finally, a decoder identical to Noise-to-Seq is employed to predict RNA sequence.

3.4. Output Rescoring Model

Since RNAFlow can be sampled many times to obtain several RNA designs, we train an *output rescoring model* to select from amongst the samples based on predicted recovery rate. For training, we generate 6 mutated RNA sequences for each ground-truth protein-RNA complex. Binary labels are generated based on whether or not a generated RNA sequence has a recovery rate $\geq 30\%$. At test time, we evaluate many RNAFlow output samples for a single protein and make selections based on the highest predicted probability of a positive outcome, prioritizing designs with the greatest likelihood of achieving a recovery rate of at least 30%.

The inputs to the output scoring model are protein sequence, protein structure, RNA sequence, and RF2NA-folded RNA structure. The protein and RNA are each encoded by a GVP model with the same architecture as the Noise-to-Seq encoder, and the resulting node-level representations are averaged and processed by a feed-forward network. The output is supervised by a binary cross entropy loss. Hyperparameter and training details are included in the Appendix.

4. Experiments

We evaluate RNAFlow in two experimental settings. First, we evaluate RNA sequence and structure prediction error with respect to existing protein-RNA complexes. Dataset construction and results are described in Section 4.1. We additionally evaluate the novelty of RNA designs in Section 4.2. In Section 4.3, we perform an ablation study to determine how modeling choices affect structure and sequence generation.

Second, we evaluate whether RNAFlow can design RNA aptamers for the GRK2 protein, a target of interest for the

Table 1. Structure generation results. We report Mean $\pm$ SEM for RMSD and lDDT. RMSD is computed after Kabsch alignment with the true RNA. lDDT is computed on C_α atoms.

Method	RF2NA Pre-Training Split		Sequence Similarity Split	
	RMSD	lDDT	RMSD	lDDT
Conditional MMDiff	14.82 ± 1.01	0.34 ± 0.02	17.42 ± 0.86	0.38 ± 0.01
RNAFlow-Base	12.85 ± 0.63	0.51 ± 0.01	14.77 ± 0.34	$\mathbf{0.57 \pm 0.01}$
RNAFlow-Traj	13.12 ± 0.64	0.52 ± 0.01	15.11 ± 0.33	0.57 ± 0.00
RNAFlow-Base + Rescore	$\mathbf{10.61 \pm 1.73}$	$\mathbf{0.53 \pm 0.03}$	$\mathbf{14.60 \pm 1.05}$	0.56 ± 0.02
RNAFlow-Traj + Rescore	15.30 ± 1.89	0.52 ± 0.03	15.31 ± 0.93	0.56 ± 0.02
RF2NA [Upper Bound]	4.67 ± 1.29	0.76 ± 0.04	9.83 ± 1.69	0.79 ± 0.02

Table 2. Sequence generation results. We report Mean $\pm$ SEM for native sequence recovery.

Method	RF2NA Pre-Training Split	Sequence Similarity Split
	Recovery	Recovery
Random	0.25 ± 0.00	0.25 ± 0.00
LSTM	0.27 ± 0.01	0.24 ± 0.01
Conditional MMDiff	0.24 ± 0.02	0.22 ± 0.02
RNAFlow-Base	0.30 ± 0.02	0.30 ± 0.01
RNAFlow-Traj	0.31 ± 0.01	0.28 ± 0.01
RNAFlow-Base + Rescoring	0.33 ± 0.02	$\mathbf{0.32 \pm 0.03}$
RNAFlow-Traj + Rescoring	$\mathbf{0.37 \pm 0.05}$	0.29 ± 0.02
Inverse Folding [Upper Bound]	0.46 ± 0.01	0.35 ± 0.01

treatment of chronic heart failure (Tesmer et al., 2012). Mayer et al. (2008) discovered an RNA aptamer that interacts strongly with the GRK2 kinase domain, and Lennarz (2015) identified 4 nucleotides that are critical for binding. In Section 4.4, we investigate whether RNAFlow can design realistic RNAs for GRK2 binding given this binding site sequence motif.

4.1. Structure & Sequence Accuracy

As mentioned, we evaluate two variants of our method: *RNAFlow-Base* which is based on Noise-to-Seq and *RNAFlow-Trajectory* which additionally applies Traj-to-Seq. For both methods, we begin by generating 10 structure-sequence pairs per protein in the test set. From amongst the 10 designs, we additionally report the performance when 1 sample is selected by the output rescoring model. All samples are generated with 5 integration steps.

Dataset. Protein-RNA complexes from the PDBBind dataset (2020 version) were used for training and evaluation (Liu et al., 2017). Dataset preparation details are described in the Appendix. We perform all experiments on two splits. The first accounts for RF2NA pre-training – all examples from complexes in the RF2NA validation or test sets were assigned to the test split, and remaining examples were randomly split into training and validation in a 9:1 ratio. To measure how well our model can generalize to dissimilar RNAs, we also evaluate on an RNA sequence similarity split. All RNA chains were clustered using CD-HIT (Fu et al., 2012) where two sequences are in the same cluster if they share $\geq 80\%$ sequence identity, following the clustering

protocol of Joshi et al. (2023). The clusters were randomly split into train, validation, and test in an $8:1:1$ ratio. We pre-train Noise-to-Seq on clean protein-RNA complexes before fine-tuning on noised complexes as part of the RNAFlow training loop. Further details are included in the Appendix.

Traj-to-Seq was trained separately on RNASolo (Adamczyk et al., 2022), a dataset of RNA sequences and structures where many of the sequences are associated with multiple structure conformers. On average, the dataset contains 3 structures per sequence. Following Joshi et al. (2023), we filtered to structures with resolution ≤ 3 Å. For the RF2NA split, we curate the Traj-to-Seq training set such that RNAs in our validation and test sets are not used for training. For the sequence similarity split, we remove RNAs that have $\geq 80\%$ similarity with data points in validation/test. Traj-to-Seq inference was run on the final 3 structures in the flow matching trajectory which yields optimal performance.

Baselines. For structure generation, we compare our method to two baselines. To our knowledge, the only existing structure generative model for nucleic acids is MMDiff (Morehead et al., 2023), an $SE(3)$-discrete diffusion model that jointly designs nucleic acid sequences and structures. We sample 10 RNA designs of the ground-truth RNA's length per protein-RNA complex in our test set, conditioned on the protein backbone structure and sequence. As an upper bound, we compare against RNA structures folded by RF2NA, given the ground-truth protein MSA and RNA sequence. This baseline captures the structure prediction error inherent to our score prediction model.

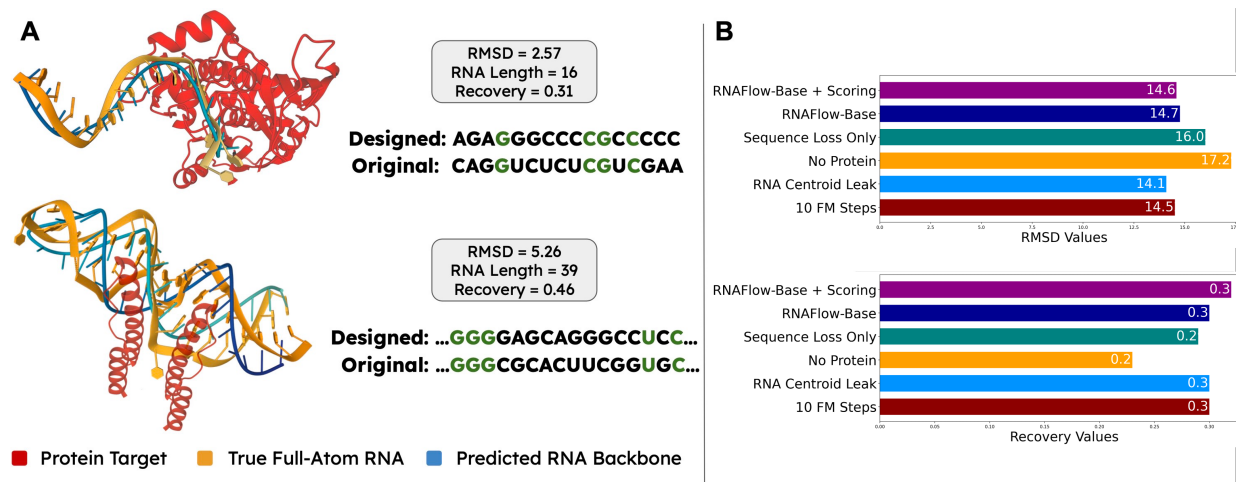For sequence generation, we compare our method to four

Figure 3. (A) Top: Structure and sequence design of RNA for interaction with a viral RNA-dependent RNA polymerase (PDB ID: 4K4X). Bottom: Design of RNA for interaction with HIV-1 Rev protein (PDB ID: 4PMI). (B) Ablation study of RNAFlow components. We report RMSD and sequence recovery on the sequence similarity split.

baselines. First, we compare to a random baseline where RNA sequences of specified length are generated by randomly selecting a nucleotide for each position, from a uniform distribution over {A, C, G, U}. Next, we compare against sequences designed by MMDiff. Our third baseline is a standard sequence-only model, consisting of an LSTM layer to encode protein sequence $\{p_i\}_{\forall i}$ and an autoregressive LSTM decoder layer to predict RNA sequence $\{\hat{r}_i\}_{\forall i}$, matching the architecture in Im et al. (2019). Finally, we establish an upper bound by comparing against results from our pre-trained inverse folding model, which takes the ground-truth protein-RNA backbone complex as input.

Metrics. For structure generation, we report root mean square deviation (RMSD) between predicted RNA structure and ground-truth structure, computed on all backbone atoms. Before RMSD calculation, the RNA structures are aligned by the Kabsch algorithm. For RNAFlow-Trajectory, we report the average RMSD across structures in the trajectory. We also report lDDT computed on C_α atoms. For sequence generation, we report recovery rate which is the percentage of correctly recovered nucleotides in a sampled sequence.

Results. As shown in Table 1, RNAFlow significantly outperforms the baseline on the structure generation task. On the RF2NA and sequence similarity splits respectively, the best RNAFlow model gives a 28% and 16% reduction in RMSD relative to MMDiff. We also observe a 56% and 50% increase in lDDT. Without rescoring, RNAFlow still gives a 13% and 15% reduction in RMSD. However, there is a large gap between the structural accuracy of RNAFlow and RF2NA, given the complexity of the *de novo* design task in comparison to folding. We note that RNAFlow-Trajectory consistently has a higher RMSD than RNAFlow-Base, likely

because trajectory structures are taken from earlier points in flow matching inference. We show two examples of generated outputs in Figure 3a, where it is evident that the designed RNA matches the ground truth structure well.

As shown in Table 2, RNAFlow outperforms all baselines on the sequence generation task. On the RF2NA split, the best RNAFlow model gives a 48% improvement over random generation and a 37% improvement over the LSTM baseline. RNAFlow-Trajectory outperforms RNAFlow-Base, suggesting that flow matching trajectories provide useful RNA dynamics information that is not present in a single structure. The output rescoring model improves the mean recovery, indicating that while RNAFlow samples can be variable, we learn to select quality samples. Without rescoring, we see a 15% improvement in recovery relative to the best baseline.

On the sequence similarity split, the best RNAFlow model gives a 28% improvement over random generation and a 33% improvement over LSTM. Without rescoring, we see a 20% improvement in recovery relative to the best baseline. We find that while the LSTM's performance degrades on the sequence similarity split, RNAFlow's performance is consistent. RNAFlow-Trajectory does not improve recovery, indicating that the generated trajectories may not approximate conformational ensembles as effectively when trying to generalize to biologically dissimilar RNAs. While the RNAFlow recovery rate does not reach the upper bound established by the inverse folding model, we note that for some proteins, our model designs new sequences that fold into the desired structure. For example, as shown Figure 3a, RNAFlow can design molecules that match the ground-truth structure with a relatively dissimilar sequence composition.
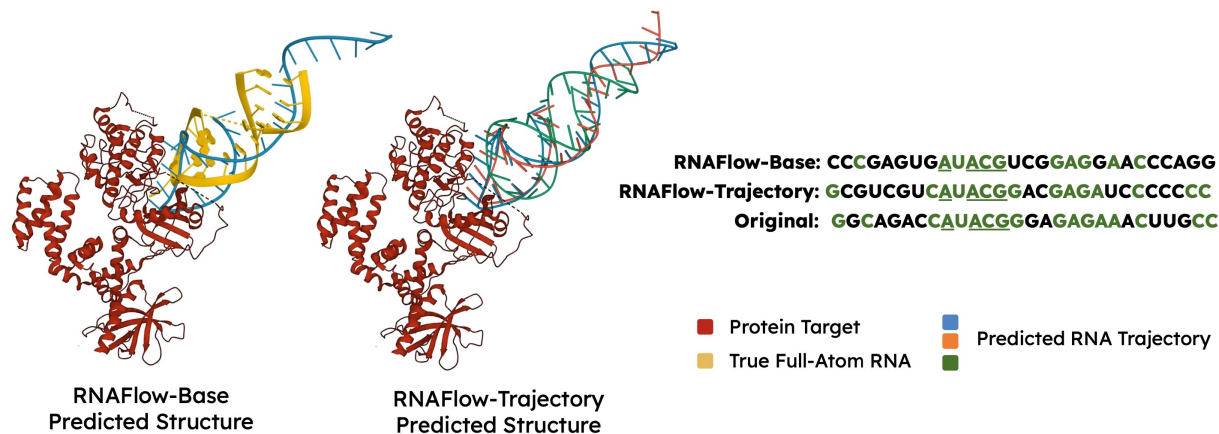
Figure 4. Structure and sequence RNAs designed by RNAFlow for GRK2 binding in motif-scaffolded setting. The predicted structure is Kabsch aligned onto the ground-truth for visualization. In the sequence designs, green-colored characters show nucleotides that are correctly recovered from the ground-truth. Underlined nucleotides are part of the given binding motif.

4.2. Structure & Sequence Novelty

We then evaluate the novelty of RNAs designed by RNAFlow-Base. Sequence novelty is quantified as $1-$ sequence recovery of each generated sample to its most similar RNA sequence in the training set. Similarly, structure novelty is $1-$ TM-score between a generated sample and its most similar RNA structure in the training set. We observe that the majority of sequence content in the generated RNAs is not observed in the training set, showing that RNAFlow can generalize and design novel RNAs. The generated structures are also reasonably novel.

Table 3. Structure and sequence novelty of RNAs generated by RNAFlow-Base.

	Sequence Novelty	Structure Novelty
RF2NA Split	0.55	0.63
Seq Sim Split	0.63	0.64

4.3. Ablation Study

We conduct an ablation study to determine which components of our model contribute to its performance. As shown in Figure 3b, we report recovery rate and RMSD on the sequence similarity split under various conditions, since the sequence split is more general to a design setting. First, we fine-tune Noise-to-Seq on sequence cross-entropy only, removing the structure MSE loss. When this denoiser is used in the standard five-step inference loop, the RMSD increases by 1.25 compared to RNAFlow-Base, and recovery rate decreases by 1 percentage point. Next, we remove protein conditioning and find that RMSD increases by 2.52, and recovery drops by 7 percentage points, showing that providing protein information is important.

We assess the effect of our "pose guess" complex alignment

method by performing inference with the true RNA centroid and protein orientation. While the recovery is not affected, the RMSD is 0.67 better than our method. This shows that our initial pose prediction method does not degrade performance significantly. Finally, we show the performance when inference is performed with ten flow matching steps rather than five. While RMSD is better than our method by 0.25, recovery is not affected. Given the insignificant impact on performance, we choose to use 5 integration steps in favor of inference time speed-up.

4.4. Motif-Scaffolded Design of RNAs for GRK2

Design Procedure. For this task, we train RNAFlow on all protein-RNA examples except the ground-truth GRK2-C28 protein-aptamer complex (PDB ID: 3UZS). Comparing against an LSTM and MMDiff, we evaluate the structure and sequence accuracy of GRK2-conditioned RNA design given a 4-nucleotide binding site sequence motif. For all methods, we sample 10 RNA designs and select 1 top design by the output rescoring model. We report the recovery rate with respect to the true aptamer sequence of length 28 and the RMSD with respect to its crystal structure.

Results. As shown in Table 3, RNAFlow outperforms the baselines in terms of both RMSD and recovery. Compared to MMDiff, RNAFlow-Base gives an 8.16% improvement in RMSD and 34.38% improvement in recovery. As shown in Figure 4, the predicted structure resembles the ground-truth aptamer closer to the binding site, though the structure deviates when further away from the protein.

Moreover, RNAFlow-Trajectory gives an 22.85% improvement in RMSD and a 68.75% improvement in recovery. Figure 4 shows the predicted structural trajectory. One of the intermediate trajectory structures shown in green has

Table 4. GRK2 binder metrics.

	RMSD	Recovery
LSTM	-	0.29
MMDiff	9.19	0.32
RNAFlow-Base	8.44	0.43
RNAFlow-Trajectory	**7.09**	**0.54**

a lower RMSD (6.42) than the final structure. However, inverse folding the intermediate structure alone yields a significantly lower recovery (0.44) than applying Traj-to-Seq (0.54). Crucially, we observe that the sequence predicted by RNAFlow-Trajectory recovers nucleotides on either end of the aptamer, far away from the given binding motif. As noted by Tesmer et al. (2012), these stem regions are critical for the aptamer's high affinity because they "limit the number of possible conformations for the selected RNA." Thus, we argue that having conformational information is important to design these regions.

5. Conclusion

In this paper, we present RNAFlow, the first protein-conditioned generative model for RNA structure and sequence design. We show that an inverse folding model is an effective score prediction network within the flow matching framework, enabling the design of RNAs that outperform the baselines in terms of structure and sequence accuracy. Additionally, we show that by inverse folding over an inferred conformational ensemble, we can design plausible aptamers for GRK2 binding.

While RNAFlow shows an empirical advantage over existing methods, there is still a large gap to reach the level of accuracy achieved in *de novo* protein design. We observe that currently, RNAFlow tends to achieve lower RMSD and higher recovery when the ground-truth RNA is more structurally stable (i.e. more base pairing). This suggests that RNAFlow is practically useful for aptamer design. On the other hand, when the RNA is very conformationally diverse (i.e. short single-stranded RNAs), RNAFlow often does not outperform a random sequence. In this setting, a pure language approach may be desired as structural supervision is less relevant.

To improve performance on this task, one major bottleneck is accuracy and efficiency of protein-RNA folding and docking models. This would allow us to supervise docked full-atom complexes and better model protein-RNA structural interaction, which is essential for designing RNAs that rely heavily on side chains for their function (i.e. ribozymes).

Code is available at https://github.com/divnori/rnaflow.

Impact Statement

This paper presents work whose goal is to advance machine learning applications for structural biology and drug discovery. There are many potential societal consequences of our work, including application of our model for discovery of an RNA drug candidate. We highlight that molecules designed by RNAFlow have not been tested experimentally which is critical for any biological application.

References

Adamczyk, B., Antczak, M., and Szachniuk, M. Rnasolo: a repository of cleaned pdb-derived rna 3d structures. *Bioinformatics*, 38(14):3668–3670, 2022.

Baek, M., DiMaio, F., Anishchenko, I., Dauparas, J., Ovchinnikov, S., Lee, G. R., Wang, J., Cong, Q., Kinch, L. N., Schaeffer, R. D., et al. Accurate prediction of protein structures and interactions using a three-track neural network. *Science*, 373(6557):871–876, 2021.

Baek, M., McHugh, R., Anishchenko, I., Jiang, H., Baker, D., and DiMaio, F. Accurate prediction of protein–nucleic acid complexes using rosettafoldna. *Nature Methods*, pp. 1–5, 2023.

Bose, A. J., Akhound-Sadegh, T., Fatras, K., Huguet, G., Rector-Brooks, J., Liu, C.-H., Nica, A. C., Korablyov, M., Bronstein, M., and Tong, A. Se (3)-stochastic flow matching for protein backbone generation. *arXiv preprint arXiv:2310.02391*, 2023.

Buglak, A. A., Samokhvalov, A. V., Zherdev, A. V., and Dzantiev, B. B. Methods and applications of in silico aptamer design and modeling. *International Journal of Molecular Sciences*, 21(22):8420, 2020.

Dykstra, P. B., Kaplan, M., and Smolke, C. D. Engineering synthetic rna devices for cell control. *Nature Reviews Genetics*, 23(4):215–228, 2022.

Fornili, A., Pandini, A., Lu, H.-C., and Fraternali, F. Specialized dynamical properties of promiscuous residues revealed by simulated conformational ensembles. *Journal of chemical theory and computation*, 9(11):5127–5147, 2013.

Fu, L., Niu, B., Zhu, Z., Wu, S., and Li, W. Cd-hit: accelerated for clustering the next-generation sequencing data. *Bioinformatics*, 28(23):3150–3152, 2012.

Ganser, L. R., Kelly, M. L., Herschlag, D., and Al-Hashimi, H. M. The roles of structural dynamics in the cellular functions of rnas. *Nature reviews Molecular cell biology*, 20(8):474–489, 2019.

Gold, L. Selex: How it happened and where it will go. *Journal of Molecular Evolution*, 81(5-6):140–143, 2015.

Ho, J., Jain, A., and Abbeel, P. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

Im, J., Park, B., and Han, K. A generative model for constructing nucleic acid sequences binding to a protein. *BMC genomics*, 20(13):1–13, 2019.

Ingraham, J., Garg, V., Barzilay, R., and Jaakkola, T. Generative models for graph-based protein design. *Advances in neural information processing systems*, 32, 2019.

Ingraham, J. B., Baranov, M., Costello, Z., Barber, K. W., Wang, W., Ismail, A., Frappier, V., Lord, D. M., Ng-Thow-Hing, C., Van Vlack, E. R., et al. Illuminating protein space with a programmable generative model. *Nature*, pp. 1–9, 2023.

Iwano, N., Adachi, T., Aoki, K., Nakamura, Y., and Hamada, M. Generative aptamer discovery using raptgen. *Nature Computational Science*, 2(6):378–386, 2022.

Jang, E., Gu, S., and Poole, B. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*, 2016.

Jin, W., Wohlwend, J., Barzilay, R., and Jaakkola, T. Iterative refinement graph neural network for antibody sequence-structure co-design. *arXiv preprint arXiv:2110.04624*, 2021.

Jing, B., Eismann, S., Suriana, P., Townshend, R. J., and Dror, R. Learning from protein structure with geometric vector perceptrons. *arXiv preprint arXiv:2009.01411*, 2020.

Jing, B., Eismann, S., Soni, P. N., and Dror, R. O. Equivariant graph neural networks for 3d macromolecular structure. *arXiv preprint arXiv:2106.03843*, 2021.

Jing, B., Berger, B., and Jaakkola, T. Alphafold meets flow matching for generating protein ensembles. In *NeurIPS 2023 AI for Science Workshop*, 2023.

Joshi, C. K., Jamasb, A. R., Viñas, R., Harris, C., Mathis, S., and Liò, P. Multi-state rna design with geometric multi-graph neural networks. In *ICML 2023 Workshop on Computation Biology*, 2023.

Kabsch, W. A solution for the best rotation to relate two sets of vectors. *Acta Crystallographica Section A: Crystal Physics, Diffraction, Theoretical and General Crystallography*, 32(5):922–923, 1976.

Keefe, A. D., Pai, S., and Ellington, A. Aptamers as therapeutics. *Nature reviews Drug discovery*, 9(7):537–550, 2010.

Kim, N., Izzo, J. A., Elmetwaly, S., Gan, H. H., and Schlick, T. Computational generation and screening of rna motifs in large nucleotide sequence pools. *Nucleic acids research*, 38(13):e139–e139, 2010.

Klein, L., Krämer, A., and Noé, F. Equivariant flow matching. *arXiv preprint arXiv:2306.15030*, 2023.

Krüger, D. M., Neubacher, S., and Grossmann, T. N. Protein–rna interactions: structural characteristics and hotspot amino acids. *Rna*, 24(11):1457–1465, 2018.

Lennarz, S. M. I. *RNA aptamers as selective protein kinase inhibitors*. PhD thesis, Universitäts-und Landesbibliothek Bonn, 2015.

Lipman, Y., Chen, R. T., Ben-Hamu, H., Nickel, M., and Le, M. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.

Liu, Z., Su, M., Han, L., Liu, J., Yang, Q., Li, Y., and Wang, R. Forging the basis for developing protein–ligand interaction scoring functions. *Accounts of chemical research*, 50(2):302–309, 2017.

Mayer, G., Wulffen, B., Huber, C., Brockmann, J., Flicke, B., Neumann, L., Hafenbradl, D., Klebl, B. M., Lohse, M. J., Krasel, C., et al. An rna molecule that specifically inhibits g-protein-coupled receptor kinase 2 in vitro. *Rna*, 14(3):524–534, 2008.

Morehead, A., Ruffolo, J., Bhatnagar, A., and Madani, A. Towards joint sequence-structure generation of nucleic acid and protein complexes with se (3)-discrete diffusion. *arXiv preprint arXiv:2401.06151*, 2023.

Sanchez de Groot, N., Armaos, A., Graña-Montes, R., Alriquet, M., Calloni, G., Vabulas, R. M., and Tartaglia, G. G. Rna structure drives interaction with proteins. *Nature communications*, 10(1):3246, 2019.

Stark, H., Jing, B., Barzilay, R., and Jaakkola, T. Harmonic prior self-conditioned flow matching for multi-ligand docking and binding site design. In *NeurIPS 2023 Generative AI and Biology (GenBio) Workshop*, 2023.

Tan, C., Zhang, Y., Gao, Z., Cao, H., and Li, S. Z. Hierarchical data-efficient representation learning for tertiary structure-based rna design, 2023.

Tancik, M., Srinivasan, P., Mildenhall, B., Fridovich-Keil, S., Raghavan, N., Singhal, U., Ramamoorthi, R., Barron, J., and Ng, R. Fourier features let networks learn high frequency functions in low dimensional domains.

Advances in Neural Information Processing Systems, 33: 7537–7547, 2020.

Tesmer, V. M., Lennarz, S., Mayer, G., and Tesmer, J. J. Molecular mechanism for inhibition of g protein-coupled receptor kinase 2 by a selective rna aptamer. *Structure*, 20(8):1300–1309, 2012.

Thavarajah, W., Hertz, L. M., Bushhouse, D. Z., Archuleta, C. M., and Lucks, J. B. Rna engineering for public health: Innovations in rna-based diagnostics and therapeutics. *Annual review of chemical and biomolecular engineering*, 12:263–286, 2021.

Vezeau, G. E., Gadila, L. R., and Salis, H. M. Automated design of protein-binding riboswitches for sensing human biomarkers in a cell-free expression system. *Nature communications*, 14(1):2416, 2023.

Wadley, L. M., Keating, K. S., Duarte, C. M., and Pyle, A. M. Evaluating and learning from rna pseudotorsional space: quantitative validation of a reduced representation for rna structure. *Journal of molecular biology*, 372(4): 942–957, 2007.

Watson, J. L., Juergens, D., Bennett, N. R., Trippe, B. L., Yim, J., Eisenach, H. E., Ahern, W., Borst, A. J., Ragotte, R. J., Milles, L. F., et al. De novo design of protein structure and function with rfdiffusion. *Nature*, 620(7976): 1089–1100, 2023.

Yesselman, J. D., Eiler, D., Carlson, E. D., Gotrik, M. R., d'Aquino, A. E., Ooms, A. N., Kladwang, W., Carlson, P. D., Shi, X., Costantino, D. A., et al. Computational design of three-dimensional rna structure and function. *Nature nanotechnology*, 14(9):866–873, 2019.

Yim, J., Campbell, A., Foong, A. Y., Gastegger, M., Jiménez-Luna, J., Lewis, S., Satorras, V. G., Veeling, B. S., Barzilay, R., Jaakkola, T., et al. Fast protein backbone generation with se (3) flow matching. *arXiv preprint arXiv:2310.05297*, 2023a.

Yim, J., Trippe, B. L., De Bortoli, V., Mathieu, E., Doucet, A., Barzilay, R., and Jaakkola, T. Se (3) diffusion model with application to protein backbone generation. *arXiv preprint arXiv:2302.02277*, 2023b.

Zhou, Q., Xia, X., Luo, Z., Liang, H., and Shakhnovich, E. Searching the sequence space for potent aptamers using selex in silico. *Journal of chemical theory and computation*, 11(12):5939–5946, 2015.

A. Flow Matching on RNA Structures

We justify the main RNAFlow objective given in Equation 3, following much of the theory given by Jing et al. (2023). By the standard formulation of flow matching in Euclidean space $\mathbb{R}^{3L_r}$ (L_r is RNA length), we expect to supervise with the reparameterized denoising objective (MSE).

$$\mathcal{L} = \mathbb{E}_{R_1 \sim p_1, R_t \sim p_t}[||\hat{R}_1 - \vec{R}_1||^2]$$

$\hat{R}_1$ is predicted by neural network $\hat{R}_1(\vec{R}_t; \theta)$ which accepts noised input $\vec{R}_t$ and a timestep embedding. While $\hat{R}_1$ is composed of an inverse folding model and pre-trained folding model, the denoised sequence can simply be thought of as an intermediate representation from which we predict the clean structure.

However, $\hat{R}_1(\vec{R}_t; \theta)$ is $SE(3)$-invariant given that RF2NA is trained with $SE(3)$-invariant FAPE loss. We are thus flow matching over the Riemannian manifold given by quotient space $\mathbb{R}^{3L_r}/SE(3)$. To flow match in this new space, we must define a conditional probability path $p_t(\vec{R}_t|\vec{R}_1)$. We can generalize linear interpolation in the Euclidean case to interpolant $\vec{R}_t|\vec{R}_1 = \psi(R_0|R_1)$ defined as the geodesic path from R_0 to R_1, where the geodesic path is Kabsch alignment followed by linear interpolation in $\mathbb{R}^{3L_r}$.

$$R_0 = \text{Kabsch}(R_0, R_1)$$

$$\psi(R_0|R_1) = (1 - t) * R_0 + t * R_1$$

We define a distance metric d on the manifold correspondingly, where d is a valid metric because it is defined for all points on the manifold.

$$d = ||\hat{R}_1 - \text{Kabsch}(\vec{R}_1, \hat{R}_1)||$$

We can thus plug this expression into our original flow matching objective,

$$\mathcal{L} = \mathbb{E}_{\vec{R}_1|\vec{R}_t}[||\hat{R}_1 - \text{Kabsch}(\vec{R}_1, \hat{R}_1)||^2] = \text{MSE}(\hat{R}_1, \text{Kabsch}(\vec{R}_1, \hat{R}_1))$$

which is equivalent to the structure loss expression in Equation 3. The cross entropy term simply exerts auxiliary sequence supervision.

B. Dataset Details

B.1. PDBBind

We filter PDBBind to complexes where at least one protein C_α atom and RNA C_4' atom were within 7 Å, a threshold that has been used to perform alanine scans for protein-RNA interaction site analysis (Krüger et al., 2018). For complexes containing many protein-RNA interaction sites, we use the interaction with least distance between the protein C_α atom and RNA C_4' atom. We filter to RNA chains of length ≥ 6 and ≤ 96, and protein chains are contiguously cropped to length 50. As shown in Figure 5, while there are many examples of short RNAs, the model is also exposed to several longer RNA examples.

In the sequence similarity split, there are 1015 complexes in train, 105 in validation, and 72 in test. The RF2NA split has 1059 complexes in train, 117 in validation, and 16 in test.

B.2. RNASolo

As mentioned, we use the RNASolo dataset for Traj-to-Seq training. The dataset consists of RNAs from apo RNA structures, protein-RNA complexes, and RNA-DNA complexes. As shown in Figure 6, the distribution of RNA lengths in the dataset is somewhat bimodal, with many sequences around length 10 and around length 75. In the sequence similarity split, there are 2314 distinct RNA sequences in train, 106 in validation, and 78 in test. In the RF2NA split, there are 2370 distinct RNA sequences in train, 110 in validation, and 16 in test.
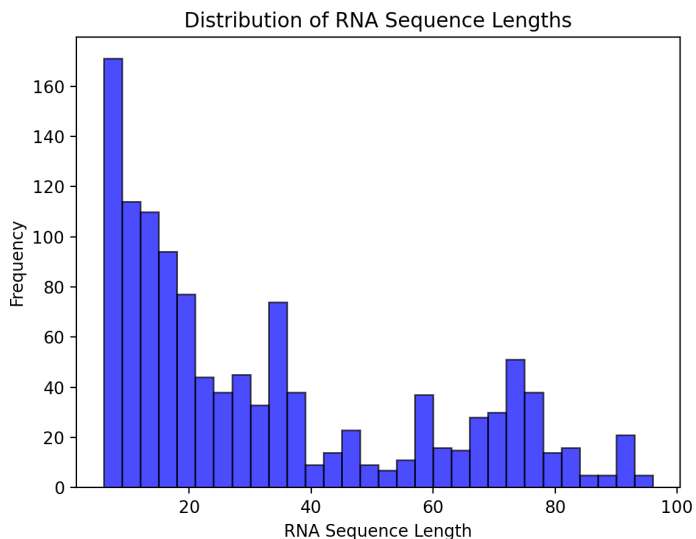
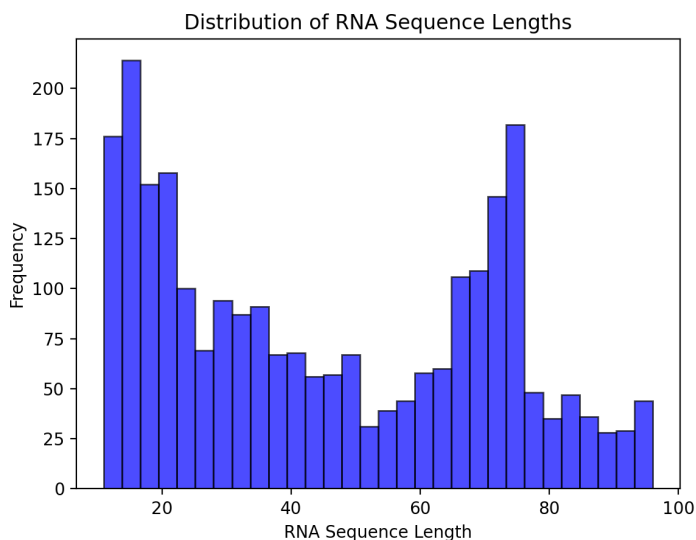*Figure 5.* Distribution of RNA lengths in processed PDBBind dataset.

Figure 6. Distribution of RNA lengths in processed RNASolo dataset.

C. Training and Architecture Details

C.1. Noise-to-Seq

For both splits, the Noise-to-Seq encoder and decoder GVP layers use a node scalar feature dimension of 128, node vector feature dimension of 16, edge scalar feature dimension of 32, and edge vector feature dimension of 1. On both splits, the model was pre-trained for 100 epochs using an Adam optimizer with a learning rate of 0.001, which takes a few hours on an NVIDIA A5000-24GB GPU.

C.2. Traj-to-Seq

For both splits, the Traj-to-Seq encoder and decoder GVP layers use a node scalar feature dimension of 128, node vector feature dimension of 16, edge scalar feature dimension of 64, and edge vector feature dimension of 4. The model was

trained for 100 epochs by an Adam optimizer with a learning rate of 0.001.

C.3. RNAFlow

We fine-tune RNAFlow for 100 epochs which takes one day on an NVIDIA A5000-24GB GPU, and we use the Adam optimizer with a learning rate of 0.001.

C.4. LSTM

For both splits, the encoder's hidden dimension is 128 which was selected by a hyperparameter sweep, and the model was trained by an Adam optimizer with a learning rate of 0.001.

C.5. Output Rescoring Model

Following Noise-to-Seq, the encoder and decoder GVP layers use a node scalar feature dimension of 128, node vector feature dimension of 16, edge scalar feature dimension of 32, and edge vector feature dimension of 1. The node output feature dimension is 256, and the final feedforward network consists of three fully connected layers with ReLU activation. The model was trained for 50 epochs using an Adam optimizer with a learning rate of 0.01.