# Antigen-Specific Antibody Design and Optimization with Diffusion-Based Generative Models

**Anonymous Author(s)**
Affiliation
Address
email

## Abstract

Antibodies are immune system proteins that protect the host by binding to specific antigens such as viruses and bacteria. The binding between antibodies and antigens are mainly determined by the complementarity-determining regions (CDR) on the antibodies. In this work, we develop a deep generative model that jointly models sequences and structures of CDRs based on diffusion processes and equivariant neural networks. Our method is the first deep learning-based method that can explicitly target specific antigen structures and generate antibodies at atomic resolution. The model is a "Swiss Army Knife" which is capable of sequence-structure co-design, sequence design for given backbone structures, and antibody optimization. For antibody optimization, we propose a special sampling scheme that first perturbs the given antibody and then denoises it. As the number of available antibody structures is relatively scarce, we curate a new dataset that contains antibody-like proteins as a complement to the original antibody dataset for training. We conduct extensive experiments to evaluate the quality of both sequences and structures of designed antibodies. We find that our model could yield highly competitive results in terms of binding affinity measured by biophysical energy functions and other protein design metrics.

## 1 Introduction

Antibodies are important immune proteins generated during an immune response to recognize and neutralize the pathogen [22]. As illustrated in Figure 1a, an antibody contains two heavy chains and two light chains and their overall structure can be similar or even identical. The specificity of an antibody to the antigens is determined by six variable regions called the Complementarity Determining Regions (CDRs) on the antibodies, denoted as H1, H2, H3, L1, L2, and L3. Therefore, the most important step for developing effective therapeutic antibodies is to design proper CDR sequences which could bind to the specific antigen [37, 2].

Similar to other protein design tasks, the search space of CDR sequences is vast. A CDR sequence with $L$ amino acids has up to $20^L$ possible protein sequences. It is simply not feasible to first solve the protein structure and then test whether it binds to the antigen for each of these sequences using experimental approaches. Conventional computational approaches rely on sampling protein sequences and structures on the complex energy landscape constructed based on physical and chemical energy, which has been found to be time consuming and easy to trap in the local optima [1, 29, 49, 36]. Recently, various deep generative models are developed to design both sequences and structures of antibodies [41, 3, 23]. In comparison to conventional algorithms, deep generative models could capture higher order interactions among amino acids on antibodies and antigens directly from data [2].

Recently, Jin et al. implemented a conditional generative model which could achieve antibody structure-sequence co-design based on the autoregressive model and iterative refinement of the

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

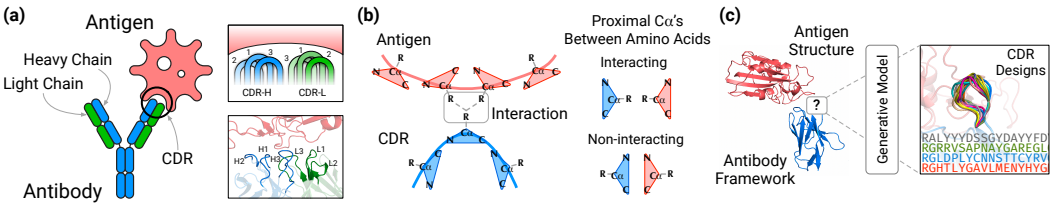


Figure 1: **(a)** Antibody-antigen complex structure and CDR structure. **(b)** The orientations of amino acids (represented by triangles) determine their side-chain orientations, which are key to inter-amino-acid interactions. **(c)** The task in this work is to design CDRs for a given antigen structure and an antibody framework.

predicted protein structure. Their model solve two important computational challenges related to antibody design: (1) how to model the intrinsic relation between CDR sequences and 3D structures, and (2) how to model the distribution of CDRs conditional on the rest of the antibody sequence. In this work, we propose another three computational challenges for antibody sequence-structure co-design. First, besides depending on the antibody sequence, the joint distribution of sequence-structure pairs should also be conditional on the 3D structures of the antigen and generate amino acids of CDRs which could fit the geometry of the antigen structure in the 3D space. Modeling the 3D structures of antigens is also important for the model to be generalized to new antigen which have not been observed yet. Second, the interaction between amino acids are mainly determined by the side chains which are groups of atoms stretching out from the protein backbone (Figure 1b) [32]. Therefore, the model should be able to consider both the backbone and the side-chain atoms, i.e. to perceive and generate structures at the atomic resolution. Third, in drug discovery, pharmacologists can collect multiple initial antibodies either from humanized mice or patients [37, 49, 9]. Therefore, instead of de novo design, another realistic scenario is to optimize a particular antibody in terms of stability and binding affinity to the antigen. To the best of our knowledge, there is no previous machine learning model that satisfies all of the above design principles.

To address all these challenges, we propose a new diffusion-based generative model [44, 17, 45] that is capable of jointly sampling antibody CDR sequences and structures at the atomic resolution. The joint distribution of a CDR sequence and its structure is conditional on the antigen structures which are modeled by equivariant neural networks [25, 24]. Specifically, we model an amino acid as a rigid body with not only 3D position but also an orientation that determines the side-chain geometry [32]. Given a protein complex consisting of an *antigen* and an *antibody framework* (antibody without CDRs) as input[1], we first initialize the CDR with an arbitrary sequence, positions, and orientations. The diffusion model aggregates the information from antigens and antibodies framework, and iteratively updates the amino acid type, the position, and the orientation of each amino acid on CDRs. In the last step, we reconstruct the CDR structure at the atom level by using side-chain packing algorithms based on the predicted orientations [4]. From the perspective of model capability, the most important reason for us to choose the diffusion-based model over other generative models such as generative adversarial networks [16] and variational auto-encoders [27] is that it generates CDR candidates iteratively in the sequence-structure space so that we can interfere and impose constraints on the sampling process to support a broader range of design tasks.

We summarize our contributions related to antibody modelling, sampling algorithms for various tasks, and a new dataset curation as follows:

- We propose the first deep learning models to perform antibody sequence-structure design by considering the 3D structures of the antigen.
- In our model, we not only design protein sequences and coordinates, but also side-chain orientations (represented as $\mathrm{SO}(3)$ element) of each amino acid. It is the first deep learning model that could achieve atomic-resolution antibody design and is equivariant to rotation and translation.
- We notice that the current available complex structures are relatively scarce, we curate a new dataset that focuses on antibody-like protein structures for the community to improve the training quality.
- Our model can be applied to a wide range of antibody design tasks, including sequence-structure co-design, fix-backbone CDR design, and antibody optimization.

[1]The complex of antigen-antibody framework can be obtained either from existing antigen-antibody structure or by docking partial antibody that contains only less versatile CDRs.

## 2 Related Work

**Computational Antibody Design** Conventional computational approaches are mostly based on sampling algorithms over hand-crafted and statistical energy functions and iteratively modify protein sequences and structures [1, 29, 49, 36, 38]. These methods are inefficient and prone to getting stuck at local optima due to the rough energy landscape. In recent years, deep learning methods achieve significant improvement over sampling algorithms for antibody design by using language models to generate protein sequences [5, 43, 41, 3]. Although much more efficient, the sequence-based methods can only generate new antibodies based on previous observed antibodies but cannot generate antibodies for specific antigen structures.

Jin et al. proposed the first CDR sequence-structure co-design deep generative model which focuses on designing antibodies to neutralize SARS-CoV-2. It relies on an additional antigen-specific predictor to predict the neutralization of the designed antibodies, which is hard to generalize to arbitrary antigens. In comparison to their model, we explicitly model the 3D structure of an antigen, opening the door to generalizing the prediction to unseen antigens with solved 3D structures. Another advantage of our model is that we consider not only backbone atom coordinates but also orientations of amino acids. The orientation is essential to protein-protein interactions as most of atoms interacting between antibodies and antigens are side-chain atoms [32] (illustrated in Figure 1b). Lastly, the model proposed by Jin et al. is not equivariant by construction, which is fundamental in molecular modelling.

**Protein Structure Prediction** Protein structure prediction algorithms take protein sequences and Multiple Sequence Alignments (MSA)s as input and translate them to 3D structures [25, 8, 53]. Accurate protein structure prediction models not only predict the position of amino acids but also their orientation [25, 53]. The orientation of an amio acid determines the direction to which its side-chain stretches, so it is indispensable for reconstructing full-atom structures. AlphaFold2 [25] predicts per-amino-acid orientations in an iterative fashion, similar to our proposed model. However, it is not generative, unable to efficiently sample diverse structures for protein design. Recently, based on prior protein structure prediction algorithms, methods for predicting antibody CDR structures have emerged [40, 39], but they are not able to design CDR sequences.

**Diffusion-Based Generative Models** Diffusion probabilistic models learn to generate data via denoising samples from a prior distribution [44, 17, 45]. Recently, progress has been made in developing equivariant diffusion models for molecular 3D structures [51, 19, 42]. Atoms in a molecule do not have natural orientations so the generation process is different from generating protein structures. Diffusion models have been also extended to non-Euclidean data, such as data in the Riemannian manifolds [30, 11]. These models are relevant to modeling orientations which are represented by elements in $\mathrm{SO}(3)$. In addition, diffusion models can also be used to generate discrete categorical data [18, 7].

## 3 Methods

This section is organized as follows: Section 3.1 introduces notations used throughout the paper and formally states the problem. Section 3.2 formulates the diffusion process for modeling antibodies. Section 3.3 introduces details about the neural network parameterization for the diffusion processes. Section 3.4 presents sampling algorithms for various antibody design tasks.

### 3.1 Definitions and Notations

An amino acid in a protein complex can be represented by its type, $\mathrm{C}_\alpha$ atom coordinate, and the orientation, which are denoted as $s_i \in \{\texttt{ACDEFGHIKLMNPQRSTVWY}\}, \boldsymbol{x}_i \in \mathbb{R}^3, \boldsymbol{O}_i \in \mathrm{SO}(3)$, respectively. Here $i = 1 \dots N$, and $N$ is the number of amino acids in the protein complex [2].

In this work, we assume the antigen structure and the antibody framework is given (Figure 1c), and we focus on designing CDRs on the antibody framework. Assume the CDR to be generated has $m$ amino acids with index from $l+1$ to $l+m$. They are denoted as $\mathcal{R} = \{(\mathrm{s}_j, \mathbf{x}_j, \mathbf{O}_j) \mid j = l+1, \dots, l+m\}$. Formally, our goal is to jointly model the distribution of $\mathcal{R}$ given the structure of the antibody-antigen complex $\mathcal{C} = \{(s_i, \boldsymbol{x}_i, \boldsymbol{O}_i) \mid i \in \{1 \dots N\} \backslash \{l+1, \dots, l+m\}\}$.

[2] Note that a protein complex contains more than one chain, so $N$ is not the length of one protein but is the sum of the lengths of all chains in the complex.

### 3.2 Diffusion Processes

A diffusion probabilistic model defines two Markov chains of diffusion processes. The forward diffusion process gradually adds noise to the data until the data distribution approximately reaches the prior distribution. The generative diffusion process starts from the prior distribution and iteratively transform it to the desired distribution. Training the model relies on the forward diffusion process to simulate the noisy data. Let $\left(\mathrm{s}_j^t, \mathbf{x}_j^t, \mathbf{O}_j^t\right)$ denote the intermediate state of amino acid $j$ at time step $t$. $\mathcal{R}^t = \{\mathrm{s}_j^t, \mathbf{x}_j^t, \mathbf{O}_j^t\}_{j=l+1}^{l+m}$ represents the sequence and structure sampled at step $t$. $t = 0$ represents the state of real data (observed sequences and structures of CDRs) and $t = T$ represent samples from the prior distribution. Forward diffusion goes from $t = 0$ to $T$, and generative diffusion proceeds in the opposite way. The diffusion processes for amino acid types $\mathrm{s}_j^t$, coordinates $\mathbf{x}_j^t$, and orientations $\mathbf{O}_j^t$ are defined as follows:

**Multinomial Diffusion for Amino Acid Types** The forward diffusion process for amino acid types is based on the multinomial distribution defined as follows [18]:

$$q(\mathrm{s}_j^t|\mathrm{s}_j^{t-1}) = \text{Multinomial}\left((1-\beta_{\text{type}}^t)\cdot \texttt{onehot}(\mathrm{s}_j^{t-1}) + \beta_{\text{type}}^t \cdot \frac{1}{20}\cdot \mathbf{1}\right), \tag{1}$$

where `onehot` represents a function that converts amino acid type to a 20-dimensional one-hot vector and $\mathbf{1}$ is an all one vector. $\beta_{\text{type}}^t$ is the probability of resampling another amino acid over 20 types uniformly. When $t \to T$, $\beta_{\text{type}}^t$ is set close to 1 and the distribution is closer to the uniform distribution. Sampling from $q(\mathrm{s}_j^t|\mathrm{s}_j^{t-1})$ requires iterative sampling starting from $t = 0$, but since it is Markovian, the distribution of $\mathrm{s}_j^t$ can be written as:

$$q(\mathrm{s}_j^t|\mathrm{s}_j^0) = \text{Multinomial}\left(\bar{\alpha}_{\text{type}}^t \cdot \texttt{onehot}(\mathrm{s}_j^0) + (1-\bar{\alpha}_{\text{type}}^t)\cdot \frac{1}{20}\cdot \mathbf{1}\right), \tag{2}$$

where $\bar{\alpha}_{\text{type}}^t = \prod_{\tau=1}^t (1-\beta_{\text{type}}^\tau)$.

The generative diffusion process is defined as:

$$p(\mathrm{s}_j^{t-1}|\mathcal{R}^t, \mathcal{C}) = \text{Multinomial}\left(F(\mathcal{R}^t, \mathcal{C})[j]\right), \tag{3}$$

where $F(\cdot)[j]$ is a neural network model taking the structure context (antigen and antibody framework) and the CDR state from the previous step as input, and predicts the probability of the amino acid type for the $j$-th amino acid on the CDR. Note that, different from the forward diffusion process, the generative diffusion process must rely on the structure context $\mathcal{C}$ and the CDR state of the previous step including positions and orientations. The main difference for these two processes is that the forward diffusion process adds noise to data so it is irrelevant to data or contexts but the generative diffusion process depends on the given condition and a full observation of the previous step. The generative diffusion process needs to approximate the posterior $q(\mathrm{s}_j^{t-1}|\mathrm{s}_j^t, \mathrm{s}_j^0)$ derived from Eq.1 and Eq.2 to denoise. Therefore, the objective of training the generative diffusion process for amino acid types is to minimize the expected KL divergence between Eq.3 and the posterior distribution:

$$L_{\text{type}}^t = \mathbb{E}_{\mathcal{R}^t\sim p}\left[\frac{1}{m}\sum\nolimits_j D_{\text{KL}}\left(q(\mathrm{s}_j^{t-1}|\mathrm{s}_j^t, \mathrm{s}_j^0)\middle\| p(\mathrm{s}_j^{t-1}|\mathcal{R}^t, \mathcal{C})\right)\right]. \tag{4}$$

**Diffusion for $\mathrm{C}_\alpha$ Coordinates** As the coordinate of an atom could be an arbitrary value, we scale and shift the coordinates of the whole structure such that the distribution of atom coordinates roughly match the standard normal distribution. The detail of the position normalization could be found in the appendix. We define the forward diffusion for the normalized $\mathrm{C}_\alpha$ coordinate $\mathbf{x}_j$ as follows:

$$q\left(\mathbf{x}_j^t \mid \mathbf{x}_j^{t-1}\right) = \mathcal{N}\left(\mathbf{x}_j^t \middle| \sqrt{1-\beta_{\text{pos}}^t}\cdot \mathbf{x}_j^{t-1}, \beta_{\text{pos}}^t \boldsymbol{I}\right), \tag{5}$$

$$q\left(\mathbf{x}_j^t \mid \mathbf{x}_j^0\right) = \mathcal{N}\left(\mathbf{x}_j^t \middle| \sqrt{\bar{\alpha}_{\text{pos}}^0}\cdot \mathbf{x}_j^0, (1-\bar{\alpha}_{\text{pos}}^0)\boldsymbol{I}\right), \tag{6}$$

where $\beta_{\text{pos}}^t$ controls the rate of diffusion and its value increases from 0 to 1 as time step goes from 0 to $t$, and $\bar{\alpha}_{\text{pos}}^t = \prod_{\tau=1}^t (1-\beta_{\text{pos}}^\tau)$. Using the reparameterization trick proposed by Ho et al., the generative diffusion process is defined as:

$$p\left(\mathbf{x}_j^{t-1}\middle|\mathcal{R}^t, \mathcal{C}\right) = \mathcal{N}\left(\mathbf{x}_j^{t-1}\middle|\boldsymbol{\mu}_p\left(\mathcal{R}^t, \mathcal{C}\right), \beta_{\text{pos}}^t \boldsymbol{I}\right), \tag{7}$$

$$\boldsymbol{\mu}_p\left(\mathcal{R}^t, \mathcal{C}\right) = \frac{1}{\sqrt{\alpha_{\text{pos}}^t}}\left(\mathbf{x}_j^t - \frac{\beta_{\text{pos}}^t}{\sqrt{1-\bar{\alpha}_{\text{pos}}^t}} G(\mathcal{R}^t, \mathcal{C})[j]\right). \tag{8}$$

Here, $G(\cdot)[j]$ is a neural network that predicts the standard Gaussian noise $\epsilon_j \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$ added to $\sqrt{\bar{\alpha}^0_{\text{pos}}}\mathbf{x}^0_j$ (scaled coordinate of amino acid $j$) based on the reparameterization of Eq.6: $\mathbf{x}^t_j = \sqrt{\bar{\alpha}^0_{\text{pos}}}\mathbf{x}^0_j + \sqrt{1-\bar{\alpha}^0_{\text{pos}}}\epsilon_j$. The objective function of training the generative process is the denoising score matching loss which minimizes the expected MSE between $G$ and $\epsilon_j$, which is simplified from aligning distribution $p$ to the posterior $q(\mathbf{x}^{t-1}_j \mid \mathbf{x}^t_j, \mathbf{x}^0_j)$:

$$L^t_{\text{pos}} = \mathbb{E}\left[\frac{1}{m}\sum\nolimits_j \left\|\epsilon_j - G(\mathcal{R}^t, \mathcal{C})\right\|^2\right]. \tag{9}$$

**SO(3) Denoising for Amino Acid Orientations** We empirically formulate an iterative perturb-denoise scheme for learning and generating amino acid orientations represented by SO(3) elements based on [30]. Note that we do not use the term *diffusion* because the formulation does not strictly follow the framework of diffusion probabilistic models though the overall principle is the same.

Similar to the typical diffusion process, the distribution of orientations perturbed for $t$ steps is defined as:

$$q\left(\mathbf{O}^t_j \mid \mathbf{O}^0_j\right) = \mathcal{IG}_{\text{SO}(3)}\left(\mathbf{O}^t_j \middle| \text{ScaleRot}\left(\sqrt{\bar{\alpha}^t_{\text{ori}}}, \mathbf{O}^0_j\right), 1-\bar{\alpha}^t_{\text{ori}}\right). \tag{10}$$

$\mathcal{IG}_{\text{SO}(3)}$ denotes the isotropic Gaussian distribution on $\text{SO}(3)$ parameterized by a mean rotation and a scalar variance [30, 34, 35]. $\text{ScaleRot}$ modifies the rotation matrix by scaling its rotation angle with the rotation axis fixed [15]. The algorithm used to sample from $\mathcal{IG}_{\text{SO}(3)}$ and other details about the distribution of $\text{SO}(3)$ could be found in the appendix. Same as the diffusion process, $\bar{\alpha}^t_{\text{ori}} = \prod^t_{\tau=1}(1-\beta^\tau_{\text{ori}})$, and $\beta^t_{\text{ori}}$ is the definition of diffusion variance increases with the step $t$. The conditional distribution used for the generation process of orientations is thus defined as:

$$p\left(\mathbf{O}^{t-1}_j \middle| \mathcal{R}^t, \mathcal{C}\right) = \mathcal{IG}_{\text{SO}(3)}\left(\mathbf{O}^{t-1}_j \middle| H(\mathcal{R}^t, \mathcal{C})[j], \beta^t_{\text{ori}}\right), \tag{11}$$

where $H(\cdot)[j]$ is a neural network that denoises the orientation and outputs the denoised orientation matrix of amino acid $j$. Training the conditional distribution requires aligning the the predicted orientation from $H(\cdot)$ to the real orientation. Hence, we formulate the training object that minimizes the expected discrepancy measured by the inner product between the real and the predicted orientation matrices:

$$L^t_{\text{ori}} = \mathbb{E}\left[\frac{1}{m}\sum\nolimits_j \left\|\left(\mathbf{O}^0_j\right)^\intercal \widehat{\mathbf{O}}^{t-1}_j - \boldsymbol{I}\right\|^2_F\right], \tag{12}$$

where $\widehat{\mathbf{O}}^{t-1}_j = H(\cdot)[j]$ is the predicted orientation for amino acid $j$.

**The Overall Training Objective** By summing Eq.4, 9, and 12 and taking the expectation w.r.t. $t$, we obtain the final training objective function:

$$L = \mathbb{E}_{t\sim\text{Uniform}(1\ldots T)}\left[L^t_{\text{type}} + L^t_{\text{pos}} + L^t_{\text{ori}}\right]. \tag{13}$$

To train the model, we first sample a time step $t$ and then sample noisy states $\{\mathrm{s}^t_j, \mathbf{x}^t_j, \mathbf{O}^t_j\}^{l+m}_{j=l+1} \sim p$ by adding noise to the training sample using the diffusion process defined by Eq.2, 6, and 10. We compute the loss using the noisy data and backpropagate the loss to update model parameters.

### 3.3 Parameterization with Neural Networks

In this section, we briefly introduce the neural network architectures used in different components of the diffusion process. The purpose of the networks is to encode the CDR state at a time step $t$ along with the context structure: $\{\mathrm{s}^t_j, \mathbf{x}^t_j, \mathbf{O}^t_j\}^{l+m}_{j=l+1} \cup \{s^t_i, \boldsymbol{x}^t_i, \boldsymbol{O}^t_i\}_{i=\{1\ldots N\}\setminus\{l+1\ldots l+m\}}$, and then denoises the CDR amino acid types ($F$), positions ($G$), and orientations ($H$).

First, we adopt Multiple Layer Perceptrons (MLPs) to generate embeddings for single and pairs of amino acids. The single amino-acid embedding MLP creates vector $\boldsymbol{e}_i$ for amino acid $i$, which encodes the information of amino acid types, torsional angles, and 3D coordinates of all the heavy atoms. The pairwise embedding MLP encodes the Euclidean distances and dihedral angles between amino acid $i$ and $j$ to feature vectors $\boldsymbol{z}_{ij}$. We adopt an orientation-aware roto-translation invariant

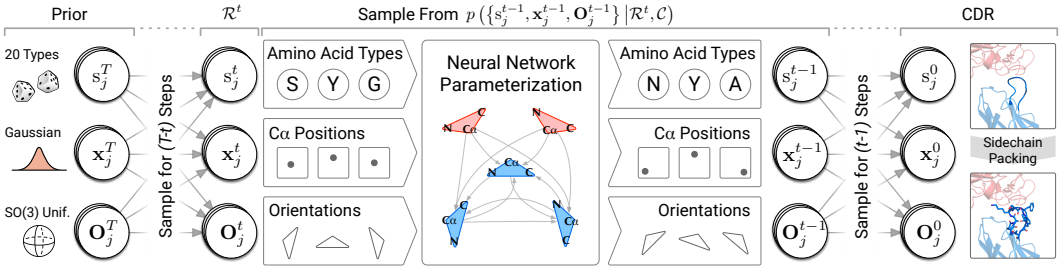


Figure 2: Illustration of the generative diffusion process. At each step, the networks takes the current CDR state as input and parameterizes the distribution of the CDR's sequences, positions, orientations for the next step. In the end, full-atom structures are constructed by the side-chain packing algorithm.

networks [25, 48] to transform $\boldsymbol{e}_i$ and $\boldsymbol{z}_{ij}$ into hidden representations $\boldsymbol{h}_i$, which aims to represent the amino acid itself and its environment. Next, the representations are fed to three different MLPs to denoise the amino acid types, 3D positions, and orientations of the CDR, respectively.

In particular, the MLP for denoising amino acid types outputs a 20 dimensional vector representing the probabilities of each type. The MLP for denoising $\mathrm{C}_\alpha$ coordinates predicts the change of the coordinate in terms of the current orientation of the amino acid. As the coordinate deviation are calculated in the local frame, we left-multiply it by the orientation matrix and transform it back to the global frame [28]. Formally, this can be expressed as $\hat{\epsilon}_j = \mathbf{O}_j^t \, \mathrm{MLP}_G\left(\boldsymbol{h}_j\right)$. Predicting coordinate deviations in the local frame and projecting it to the global frame ensures the equivariance of the prediction [28], as when the entire 3D structure rotates by a particular angle, the coordinate deviations also rotates the same angle. The MLP for denoising orientations first predicts a $so(3)$ vector [15]. The vector is converted to a rotation matrix $\boldsymbol{M}_j \in \mathrm{SO}(3)$ left-multiplied to the orientation to produce a new mean orientation for the next generative step: $\widehat{\mathbf{O}}_j^{t-1} \leftarrow \boldsymbol{M}_j \mathbf{O}_j^t$. The prediction is invariant to rotation and translation. Intuitively, changes in angles are irrelevant to the global pose. We are able to prove that the proposed networks are equivariant to rotation and translation of the overall structure:

**Proposition 1.** *For any proper rotation matrix $\boldsymbol{R} \in \mathrm{SO}(3)$ and any 3D vector $\boldsymbol{r} \in \mathbb{R}^3$ (rigid transformation $(\boldsymbol{R}, \boldsymbol{r}) \in \mathrm{SE}(3)$), $F$, $G$ and $H$ satisfy the following equivariance properties:*

$$F(\boldsymbol{R}\mathcal{R}^t + \boldsymbol{r}, \boldsymbol{R}\mathcal{C} + \boldsymbol{r}) = F(\mathcal{R}^t, \mathcal{C}), \tag{14}$$

$$G(\boldsymbol{R}\mathcal{R}^t + \boldsymbol{r}, \boldsymbol{R}\mathcal{C} + \boldsymbol{r}) = \boldsymbol{R}G(\mathcal{R}^t, \mathcal{C}), \tag{15}$$

$$H(\boldsymbol{R}\mathcal{R}^t + \boldsymbol{r}, \boldsymbol{R}\mathcal{C} + \boldsymbol{r}) = H(\mathcal{R}^t, \mathcal{C}), \tag{16}$$

*where $\boldsymbol{R}\mathcal{R}^t + \boldsymbol{r} := \{\mathrm{s}_j^t, \mathbf{x}_j^t + \boldsymbol{r}, \boldsymbol{R}\mathbf{O}_j^t\}_{j=l+1}^{l+m}$ and $\boldsymbol{R}\mathcal{C} + \boldsymbol{r} := \{s_i, \boldsymbol{x}_i + \boldsymbol{r}, \boldsymbol{R}\boldsymbol{O}_i\}_{i \in \{1 \ldots N\} \setminus \{l+1, \ldots, l+m\}}$ denote the rotated and translated structure. Note that $F$, $G$, and $H$ are not single MLPs. Each of them includes a shared encoder and a specific MLP.*

*Proof.* Please refer to the appendix. □

The proposition ensures that the probability of a structure is invariant to any rigid transform [51]. In other word, if two structures are the same up to rigid transform, they have an equal probability of being sampled from the distribution of our model.

### 3.4 Sampling Algorithms

The sampling algorithm first samples amino acid types from the uniform distribution over 20 classes: $\mathrm{s}_j^T \sim \mathrm{Uniform}(20)$, $\mathrm{C}_\alpha$ positions from the standard normal distribution: $\mathbf{x}_j^T \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}_3)$, and orientations from the uniform distribution over $\mathrm{SO}(3)$: $\mathbf{O}_j^T \sim \mathrm{Uniform}(\mathrm{SO}(3))$. Note that we normalize the coordinates of the structure in the same way as training such that $\mathrm{C}_\alpha$ positions in the CDR roughly follow the standard normal distribution. Next, we iteratively sample sequences and structures from the generative diffusion kernel by denoising amino acid types, $\mathrm{C}_\alpha$ coordinates, and orientations until $t = 0$. To build a full atom 3D structure, we construct the coordinates of N, $\mathrm{C}_\alpha$, C, O, and side-chain $\mathrm{C}_\beta$ (except glycine that does not have $\mathrm{C}_\beta$) according to their ideal local coordinates relative to the $\mathrm{C}_\alpha$ position and orientation of each amino acid [14]. Based on the five reconstructed atoms, the rest of side-chain atoms are constructed using the side-chain packing function implemented in Rosetta [4]. In the end, we adopt the AMBER99 force field [33] in OpenMM [13] to refine the full atom structure. More details about the construction and refinement process could be found in the appendix.

Table 1: The performance of the baselines and our method in the sequence-structure co-design task. (↑) denotes higher is better and (↓) denotes lower is better.

| | IMPROVE% (%, ↑) | | | RMSD$_{\text{ref}}$ (Å, ↓) | | | AAR (%, ↑) | | |
|---|---|---|---|---|---|---|---|---|---|
| CDR | H1 | H2 | H3 | H1 | H2 | H3 | H1 | H2 | H3 |
| RAbD | 30.43 (3.1) | 30.69 (1.9) | 12.83 (1.3) | 3.56 (.05) | 2.85 (.09) | 4.58 (.13) | 20.63 (1.6) | 27.80 (0.8) | 21.73 (0.7) |
| GNN | 27.77 (2.2) | 27.04 (4.5) | 8.00 (1.1) | 1.98 (.02) | 1.32 (.05) | 3.59 (.16) | 40.39 (3.2) | 33.36 (1.7) | 21.89 (1.5) |
| **Ours** | **37.76 (2.8)** | 32.33 (6.2) | **16.68 (2.2)** | **1.51 (.01)** | **1.24 (.01)** | **2.89 (.15)** | **52.82 (0.9)** | **45.95 (2.3)** | **27.04 (2.8)** |
| | IMPROVE% (%, ↑) | | | RMSD$_{\text{ref}}$ (Å, ↓) | | | AAR (%, ↑) | | |
| CDR | L1 | L2 | L3 | L1 | L2 | L3 | L1 | L2 | L3 |
| RAbD | **42.70 (4.1)** | **50.65 (5.3)** | 36.65 (1.3) | 1.88 (.01) | 1.35 (.02) | 2.14 (.06) | 35.11 (1.0) | 27.82 (0.6) | 23.73 (0.5) |
| GNN | 36.69 (3.9) | 41.27 (2.7) | 29.06 (2.1) | 2.06 (.02) | 1.26 (.01) | 1.95 (.06) | 41.44 (2.5) | 36.71 (4.3) | 33.80 (4.8) |
| **Ours** | 32.78 (5.0) | 41.52 (1.1) | 33.80 (2.4) | **1.48 (.01)** | **1.11 (.06)** | **1.65 (.05)** | **62.71 (1.2)** | **52.10 (3.6)** | **43.62 (2.6)** |

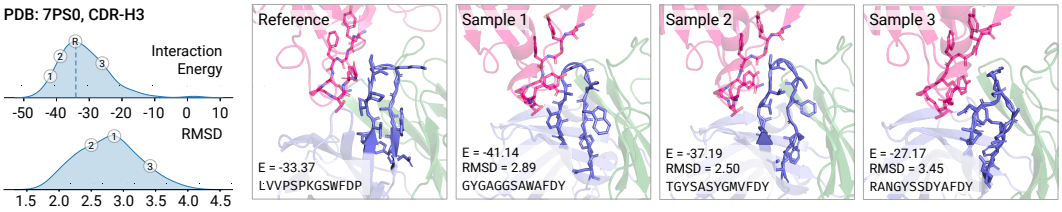


Figure 3: Examples of CDR-H3 designed by the sequence-structure codesign method and the distribution of their interaction energy and RMSD. The antigen-antibody template is derived from PDB:7ps0, where the antigen is SARS-CoV-2 RBD.

In addition to the joint design of sequences and structures, we can constrain partial states for other design tasks. For example, by fixing the backbone structure (positions and orientations) and sampling only sequences, we can do **fix-backbone sequence design**. Another usage of the model is to **optimize an existing antibody**. Specifically, we first add noise to the existing antibody for $t$ steps and denoise the perturbed antibody sequence starting from the $t$-th step of the generative diffusion process.

# 4 Experiments

The number of solved antibody structure available for training is relatively small, so we first present a new dataset we curated in Section 4.1. Next, we present the application of our model in three antibody design tasks: sequence-structure co-design (Section 4.2), antibody sequence design based on antibody backbones (Section 4.3), and antibody optimization (Section 4.4). In Section 4.5, we show how to use our model without known antibody framework bound to the antigen. Other experimental details including hyper-parameters tuning and model architectures could be found in the appendix.

## 4.1 Dataset Curation

Our new dataset is collected from two sources: SAbDab [12] and the Protein Data Bank (PDB) [10]. We select antibody-antigen protein complexes from SAbDab retrieved in January, 2022, leading to a subset containing 5,733 structures. CDRs are identified using the antibody numbering program AbRSA [31]. The selected data points are divided into training and test data based on their release date and CDR sequence identity. The **test split** includes protein structures released after December 24, 2021 and structures with any CDR similar to those released after the date (sequence identity higher than 50%). Antibodies in the test set are further clustered with 50% CDR sequence identity to remove duplicates, finally resulting in 20 antibody-antigen structures. The **training split** contains complexes not involved during the curation of the test split. To augment the training set, we curated extra protein complexes from PDB (with structures appearing in SAbDab removed). We first identify loop regions using DSSP [26] in protein-only PDB structures with resolution better than 3.5Å. Next, we select the loop regions that interact with other chains (two amino acids interacts if they have at least one pair of heavy atoms whose distance is less than 5.0Å). These selected loop regions are labeled as pseudo-CDRs and are integrated into the training set after removing duplicates at 50% sequence identity, resulting in 34,781 pseudo-CDR complexes. We find that using the augmented training dataset could enhance the performance on the test set. More details about the datasets could be found in the appendix.

Table 2: The performance of the baselines and our method in the fix-backbone design task. More results on CDR-L are put to the appendix.

| | IMPROVE% (%, ↑) | | | AAR (%, ↑) | | |
|---|---|---|---|---|---|---|
| CDR | H1 | H2 | H3 | H1 | H2 | H3 |
| FixBB | 23.84 (3.1) | 18.27 (0.7) | **17.81 (1.1)** | 36.29 (0.2) | 37.70 (0.3) | 28.13 (0.1) |
| AR | 31.22 (3.4) | 21.73 (1.4) | 10.95 (1.0) | 53.24 (3.2) | 49.87 (1.2) | 30.29 (0.4) |
| Ours | **32.43 (3.9)** | **30.95 (1.5)** | 11.03 (0.1) | **59.91 (1.2)** | **59.14 (1.8)** | **33.30 (0.5)** |

### 4.2 Sequence-Structure Co-design

To evaluate the performance, we remove the original CDR from the antibody-antigen complex in the test set and try to co-design sequence and structure of the removed region. We set the length of the CDR to be identical to the length of the original CDR for simplicity. In practice, one can enumerate different lengths of CDRs. To evaluate the performance, we compare our model to the tow baseline models: (1) **RAbD**: or RosettaAntibodyDesign [1], an antibody design software based on Rosetta energy functions. (2) **GNN**: a model co-designs sequences and structures in an alternating way, similar to [23]. For each model, we sample 1,000 candidates for each CDR. Both the original structures and designed structures from different methods are refined by OpenMM and Rosetta.

We use the following metrics to evaluate designed antibodies: (1) IMPROVE%: it denotes the percentage of designed CDRs with lower binding energy (lower is better) than the original CDR, in which the binding energy is based on the Rosetta energy function [4]. (2) $\text{RMSD}_{\text{ref}}$: it is the $\text{C}_\alpha$ root-mean-square deviation (RMSD) between the generated structure and the original structure with only antibody frameworks aligned [40]. Here, a common mistake is to calculate the RMSD by only superimposing the CDRs. This action will yield the misalignment of the non-CDR region. Lower RMSD indicates higher structural similarity to an existing antibody, which indicates better structures to fit the antigen. (3) AAR: it is the amino acid recovery rate measured by the sequence identity between the reference CDR sequences and the generated sequences [1]. Notably, we do not use neutralization prediction models because they are sequence-based and are specified to a limited class of antigens, which deviates from our goal of developing a general antibody design model. More details about the metrics can be found in the appendix.

We run each model five times with different random seeds and report the mean and standard deviation of the three metrics on the test set in Table 1. Generated examples are presented in Figure 3. Our model outperforms all the baselines in terms of both $\text{RMSD}_{\text{ref}}$ and AAR, which implies that our model might have a higher success rates to design new antibodies targeting the antigen. Notice that our model does not outperform RAbD in all the settings because the objective function RAbD optimizes is the Rosetta energy which is the metric to evaluate the performance here. It can be observed that our model achieves comparatively good energy without explicit supervision signal from Rosetta energy functions. It can be expected that the performance of our model can be further improved as the number of antibodies and antigens with solved 3D structure keeps increasing. However, RAbD directly optimize the Rosetta energy for a CDR by sampling protein structures so it is irrelevant to the number of training data.

### 4.3 Fix-Backbone Sequence Design

In this setting, the backbone structure of CDRs are given and we only need to design the CDR sequence, which transforms the task to a constrained sampling problem. Fix-backbone design is a common setting in the area of protein design [21, 20, 6, 46, 47]. For this task, we compare to the following baselines: (1) **FixBB**: a Rosetta-based sequence design software given CDR backbone structure. (2) **AR**: an auto-regressive deep generative model that could sample CDR sequence based on the backbone structure, which shares the same methodology of [21]. The side-chain atoms is packed by Rosetta after the protein sequence is designed by an algorithm. We rely on the IMPROVE% and AAR metrics introduced in Section 4.2 to evaluate the designed antibodies. We rule out the metric $\text{RMSD}_{\text{ref}}$ because the backbone structures are fixed.

As shown in Table 2, our model achieves the highest AAR in all the CDRs evaluated. In terms of IMPROVE%, our model outperforms AR for the all cases and FixBB for most of the cases. This shows that our model is also powerful in modeling the conditional probability of sequences given backbone structures.

| $t$ | IMP% | RMSD | SeqID |
|---|---|---|---|
| 4 | 20.83 | 1.24 | 89.65 |
| 8 | 18.42 | 1.33 | 87.21 |
| 16 | 14.45 | 1.54 | 66.36 |
| 32 | 11.62 | 1.96 | 30.20 |
| 64 | 13.91 | 2.48 | 25.80 |
| $T$ | 16.68 | 2.89 | 27.04 |

Table 3: Evaluation of optimized antibodies under different number of optimization steps.

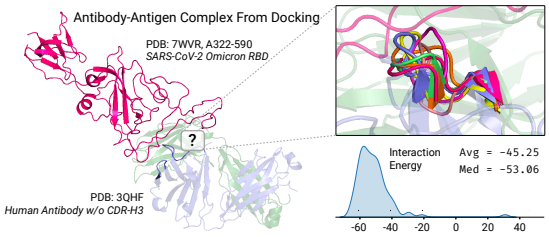


Figure 4: A human antibody framework docked to SARS-CoV-2 Omicron RBD using HDOCK. CDR-H3s are designed based on the docking structure.

### 4.4 Antibody Optimization

We use our model to optimize existing antibodies which is another common pharmaceutical application. To optimize an antibody, we first perturb the CDR sequence and structure for $t$ steps using the forward diffusion process. We denoise the sequences starting from the $(T-t)$-th step ($t$ steps remaining) of the generative diffusion process and obtain a set of optimized antibodies. We optimize CDR-H3 of the antibodies in the test set with various $t$ values. For each antibody and $t$, we perturb the CDR independently for 100 times and collect 100 optimized CDRs different from the original CDR. We report the percentage of optimized antibodies with improved binding energy (IMPROVE%), RMSD and sequence identity (SeqID) of the optimized antibodies in comparison to the original antibody. We also compare the optimized antibodies with the de novo ($t = T = 100$) designed antibodies introduced in Section 4.2. As shown in Table 3, the optimization method could produce antibodies with improved binding affinity measured by the Rosetta energy function. The optimization can be controlled by $t$. Larger $t$ leads to higher discrepancy to the original antibody.

### 4.5 Design Without Bound Antibody Frameworks

In the last experiment, we consider a more general yet more challenging setting to design antibodies without known binding pose of the antibody against the antigen. Here, we show that this challenging task could be achieved by integrating our model with docking software. Specifically, we create an *antibody template* from an existing antibody structure by removing its CDR-H3. This is because CDR-H3 is the most variable one and accounts for most of the specificity, while other CDRs are much more conserved structurally [50]. Next, we use HDOCK [52] to dock the antibody template to the target antigen to produce the antibody-antigen complex. In this way, the problem reduces to the original problem so we can adopt our model to design the CDR-H3 sequence and structure and re-design other CDRs. We demonstrate this method by designing antibodies for the SARS-CoV-2 Omicron RDB structure (PDB: `7wvn`, residue A322-A590, the structure is not bound to any other antibodies). The antibody template is derived from a human antibody against influenza (PDB: `3qhf`). The docked structure, designed CDRs, and the distribution of binding energy are presented in Figure 4. It can be seen from the binding energy distribution that the designs are reasonable and potentially have good binding affinity to the antigen.

## 5 Conclusions and Limitations

In this work, we propose a diffusion-based generative model for antibody design. Our model is capable of a wide range of antibody design tasks and can achieve highly competitive performance on all of these tasks. We also curate a new dataset for training the model. The main limitations of this work includes: (1) it relies on an antibody framework bound to the target antigen, and (2) it integrates a side-chain packing algorithm and does not generate full-atom structures in an end-to-end way. Future work includes investigating how to generate antibodies without bound structures and developing an end-to-end full-atom generative model.

## References

[1] Jared Adolf-Bryfogle, Oleks Kalyuzhniy, Michael Kubitz, Brian D Weitzner, Xiaozhen Hu, Yumiko Adachi, William R Schief, and Roland L Dunbrack Jr. Rosettaantibodydesign (rabd): A general framework for computational antibody design. *PLoS computational biology*, 14(4): e1006112, 2018.

[2] Rahmad Akbar, Habib Bashour, Puneet Rawat, Philippe A Robert, Eva Smorodina, Tudor-Stefan Cotet, Karine Flem-Karlsen, Robert Frank, Brij Bhushan Mehta, Mai Ha Vu, et al. Progress and

challenges for the machine learning-based design of fit-for-purpose monoclonal antibodies. In *Mabs*, volume 14, page 2008790. Taylor & Francis, 2022.

[3] Rahmad Akbar, Philippe A Robert, Cédric R Weber, Michael Widrich, Robert Frank, Milena Pavlović, Lonneke Scheffer, Maria Chernigovskaya, Igor Snapkov, Andrei Slabodkin, et al. In silico proof of principle of machine learning-based antibody design at unconstrained scale. In *Mabs*, volume 14, page 2031482. Taylor & Francis, 2022.

[4] Rebecca F Alford, Andrew Leaver-Fay, Jeliazko R Jeliazkov, Matthew J O'Meara, Frank P DiMaio, Hahnbeom Park, Maxim V Shapovalov, P Douglas Renfrew, Vikram K Mulligan, Kalli Kappel, et al. The rosetta all-atom energy function for macromolecular modeling and design. *Journal of chemical theory and computation*, 13(6):3031–3048, 2017.

[5] Ethan C Alley, Grigory Khimulya, Surojit Biswas, Mohammed AlQuraishi, and George M Church. Unified rational protein engineering with sequence-based deep representation learning. *Nature methods*, 16(12):1315–1322, 2019.

[6] Ivan Anishchenko, Samuel J Pellock, Tamuka M Chidyausiku, Theresa A Ramelot, Sergey Ovchinnikov, Jingzhou Hao, Khushboo Bafna, Christoffer Norn, Alex Kang, Asim K Bera, et al. De novo protein design by deep network hallucination. *Nature*, 600(7889):547–552, 2021.

[7] Jacob Austin, Daniel D Johnson, Jonathan Ho, Daniel Tarlow, and Rianne van den Berg. Structured denoising diffusion models in discrete state-spaces. *Advances in Neural Information Processing Systems*, 34:17981–17993, 2021.

[8] Minkyung Baek, Frank DiMaio, Ivan Anishchenko, Justas Dauparas, Sergey Ovchinnikov, Gyu Rie Lee, Jue Wang, Qian Cong, Lisa N Kinch, R Dustin Schaeffer, et al. Accurate prediction of protein structures and interactions using a three-track neural network. *Science*, 373(6557):871–876, 2021.

[9] Kyle A Barlow, Shane O Conchuir, Samuel Thompson, Pooja Suresh, James E Lucas, Markus Heinonen, and Tanja Kortemme. Flex ddg: Rosetta ensemble-based estimation of changes in protein–protein binding affinity upon mutation. *The Journal of Physical Chemistry B*, 122(21): 5389–5399, 2018.

[10] Helen M Berman, John Westbrook, Zukang Feng, Gary Gilliland, Talapady N Bhat, Helge Weissig, Ilya N Shindyalov, and Philip E Bourne. The protein data bank. *Nucleic acids research*, 28(1):235–242, 2000.

[11] Valentin De Bortoli, Emile Mathieu, Michael Hutchinson, James Thornton, Yee Whye Teh, and Arnaud Doucet. Riemannian score-based generative modeling. *arXiv preprint arXiv:2202.02763*, 2022.

[12] James Dunbar, Konrad Krawczyk, Jinwoo Leem, Terry Baker, Angelika Fuchs, Guy Georges, Jiye Shi, and Charlotte M Deane. Sabdab: the structural antibody database. *Nucleic acids research*, 42(D1):D1140–D1146, 2014.

[13] Peter Eastman, Jason Swails, John D Chodera, Robert T McGibbon, Yutong Zhao, Kyle A Beauchamp, Lee-Ping Wang, Andrew C Simmonett, Matthew P Harrigan, Chaya D Stern, et al. Openmm 7: Rapid development of high performance algorithms for molecular dynamics. *PLoS computational biology*, 13(7):e1005659, 2017.

[14] RA Engh and R Huber. Structure quality and target parameters. 2012.

[15] Jean Gallier and Dianna Xu. Computing exponentials of skew-symmetric matrices and logarithms of orthogonal matrices. *International Journal of Robotics and Automation*, 18(1):10–20, 2003.

[16] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.

[17] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020.

[18] Emiel Hoogeboom, Didrik Nielsen, Priyank Jaini, Patrick Forré, and Max Welling. Argmax flows and multinomial diffusion: Learning categorical distributions. *Advances in Neural Information Processing Systems*, 34, 2021.

[19] Emiel Hoogeboom, Victor Garcia Satorras, Clément Vignac, and Max Welling. Equivariant diffusion for molecule generation in 3d. *arXiv preprint arXiv:2203.17003*, 2022.

[20] Chloe Hsu, Robert Verkuil, Jason Liu, Zeming Lin, Brian Hie, Tom Sercu, Adam Lerer, and Alexander Rives. Learning inverse folding from millions of predicted structures. *bioRxiv*, 2022.

[21] John Ingraham, Vikas Garg, Regina Barzilay, and Tommi Jaakkola. Generative models for graph-based protein design. *Advances in Neural Information Processing Systems*, 32, 2019.

[22] Charles A Janeway, Paul Travers, Mark Walport, and Donald J Capra. *Immunobiology*. Taylor & Francis Group UK: Garland Science, 2001.

[23] Wengong Jin, Jeremy Wohlwend, Regina Barzilay, and Tommi S. Jaakkola. Iterative refinement graph neural network for antibody sequence-structure co-design. In *International Conference on Learning Representations*, 2022.

[24] Bowen Jing, Stephan Eismann, Patricia Suriana, Raphael JL Townshend, and Ron Dror. Learning from protein structure with geometric vector perceptrons. *arXiv preprint arXiv:2009.01411*, 2020.

[25] John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, et al. Highly accurate protein structure prediction with alphafold. *Nature*, 596(7873):583–589, 2021.

[26] Wolfgang Kabsch and Christian Sander. Dictionary of protein secondary structure: pattern recognition of hydrogen-bonded and geometrical features. *Biopolymers: Original Research on Biomolecules*, 22(12):2577–2637, 1983.

[27] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[28] Miltiadis Kofinas, Naveen Nagaraja, and Efstratios Gavves. Roto-translated local coordinate frames for interacting dynamical systems. *Advances in Neural Information Processing Systems*, 34, 2021.

[29] Gideon D Lapidoth, Dror Baran, Gabriele M Pszolla, Christoffer Norn, Assaf Alon, Michael D Tyka, and Sarel J Fleishman. Abdesign: A n algorithm for combinatorial backbone design guided by natural conformations and sequences. *Proteins: Structure, Function, and Bioinformatics*, 83 (8):1385–1406, 2015.

[30] Adam Leach, Sebastian M Schmon, Matteo T Degiacomi, and Chris G Willcocks. Denoising diffusion probabilistic models on so (3) for rotational alignment. In *ICLR 2022 Workshop on Geometrical and Topological Representation Learning*, 2022.

[31] Lei Li, Shuang Chen, Zhichao Miao, Yang Liu, Xu Liu, Zhi-Xiong Xiao, and Yang Cao. Abrsa: a robust tool for antibody numbering. *Protein Science*, 28(8):1524–1531, 2019.

[32] Anders Liljas, Lars Liljas, Goran Lindblom, Poul Nissen, Morten Kjeldgaard, and Miriam-rose Ash. *Textbook of structural biology*, volume 8. World Scientific, 2016.

[33] Kresten Lindorff-Larsen, Stefano Piana, Kim Palmo, Paul Maragakis, John L Klepeis, Ron O Dror, and David E Shaw. Improved side-chain torsion potentials for the amber ff99sb protein force field. *Proteins: Structure, Function, and Bioinformatics*, 78(8):1950–1958, 2010.

[34] S Matthies, J Muller, and GW Vinel. On the normal distribution in the orientation space. *Textures and Microstructures*, 10, 1970.

[35] Dmitry I Nikolayev and Tatjana I Savyolov. Normal distribution on the rotation group so (3). *Textures and Microstructures*, 29, 1970.

[36] RJ Pantazes and Costas D Maranas. Optcdr: a general computational method for the design of antibody complementarity determining regions for targeted epitope binding. *Protein Engineering, Design & Selection*, 23(11):849–858, 2010.

[37] Leonard G Presta. Antibody engineering. *Current Opinion in Structural Biology*, 2(4):593–596, 1992.

[38] Jeffrey A Ruffolo, Jeffrey J Gray, and Jeremias Sulam. Deciphering antibody affinity maturation with language models and weakly supervised learning. *arXiv*, 2021.

[39] Jeffrey A Ruffolo, Lee-Shin Chu, Sai Pooja Mahajan, and Jeffrey J Gray. Fast, accurate antibody structure prediction from deep learning on massive set of natural antibodies. *bioRxiv*, 2022.

[40] Jeffrey A Ruffolo, Jeremias Sulam, and Jeffrey J Gray. Antibody structure prediction using interpretable deep learning. *Patterns*, 3(2):100406, 2022.

[41] Koichiro Saka, Taro Kakuzaki, Shoichi Metsugi, Daiki Kashiwagi, Kenji Yoshida, Manabu Wada, Hiroyuki Tsunoda, and Reiji Teramoto. Antibody design using lstm based deep generative model from phage display library for affinity maturation. *Scientific reports*, 11(1):1–13, 2021.

[42] Chence Shi, Shitong Luo, Minkai Xu, and Jian Tang. Learning gradient fields for molecular conformation generation. In *International Conference on Machine Learning*, pages 9558–9568. PMLR, 2021.

[43] Jung-Eun Shin, Adam J Riesselman, Aaron W Kollasch, Conor McMahon, Elana Simon, Chris Sander, Aashish Manglik, Andrew C Kruse, and Debora S Marks. Protein design and variant prediction using autoregressive generative models. *Nature communications*, 12(1):1–11, 2021.

[44] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning*, pages 2256–2265. PMLR, 2015.

[45] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. *Advances in Neural Information Processing Systems*, 32, 2019.

[46] Alexey Strokach, David Becerra, Carles Corbi-Verge, Albert Perez-Riba, and Philip M Kim. Fast and flexible protein design using deep graph neural networks. *Cell systems*, 11(4):402–411, 2020.

[47] Doug Tischer, Sidney Lisanza, Jue Wang, Runze Dong, Ivan Anishchenko, Lukas F Milles, Sergey Ovchinnikov, and David Baker. Design of proteins presenting discontinuous functional sites using deep learning. *Biorxiv*, 2020.

[48] Jérôme Tubiana, Dina Schneidman-Duhovny, and Haim J Wolfson. Scannet: An interpretable geometric deep learning model for structure-based protein binding site prediction. *bioRxiv*, 2021.

[49] Shira Warszawski, Aliza Borenstein Katz, Rosalie Lipsh, Lev Khmelnitsky, Gili Ben Nissan, Gabriel Javitt, Orly Dym, Tamar Unger, Orli Knop, Shira Albeck, et al. Optimizing antibody affinity and stability by the automated design of the variable light-heavy chain interfaces. *PLoS computational biology*, 15(8):e1007207, 2019.

[50] John L Xu and Mark M Davis. Diversity in the cdr3 region of vh is sufficient for most antibody specificities. *Immunity*, 13(1):37–45, 2000.

[51] Minkai Xu, Lantao Yu, Yang Song, Chence Shi, Stefano Ermon, and Jian Tang. Geodiff: A geometric diffusion model for molecular conformation generation. *arXiv preprint arXiv:2203.02923*, 2022.

[52] Yumeng Yan, Di Zhang, Pei Zhou, Botong Li, and Sheng-You Huang. Hdock: a web server for protein–protein and protein–dna/rna docking based on a hybrid strategy. *Nucleic acids research*, 45(W1):W365–W373, 2017.

[53] Jianyi Yang, Ivan Anishchenko, Hahnbeom Park, Zhenling Peng, Sergey Ovchinnikov, and David Baker. Improved protein structure prediction using predicted interresidue orientations. *Proceedings of the National Academy of Sciences*, 117(3):1496–1503, 2020.

# Checklist

The checklist follows the references. Please read the checklist guidelines carefully for information on how to answer these questions. For each question, change the default **[TODO]** to [Yes] , [No] , or [N/A] . You are strongly encouraged to include a **justification to your answer**, either by referencing the appropriate section of your paper or providing a brief inline description. For example:

- Did you include the license to the code and datasets? [Yes] See Section **??**.
- Did you include the license to the code and datasets? [No] The code and the data are proprietary.
- Did you include the license to the code and datasets? [N/A]

Please do not modify the questions and only use the provided macros for your answers. Note that the Checklist section does not count towards the page limit. In your paper, please delete this instructions block and only keep the Checklist section heading above along with the questions/answers below.

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes]
   (c) Did you discuss any potential negative societal impacts of your work? [No] We have not seen any significant potential negative societal impacts of this work so far.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [No] Code and data will be released once the paper is made public.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] In Section 4.1 and in the appendix.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] Single A100 GPU.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [N/A]
   (c) Did you include any new assets either in the supplemental material or as a URL? [No]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]