FAFE: Immune Complex Modeling with Geodesic Distance Loss on Noisy Group Frames

Ruidong Wu [*1] **Ruihan Guo** [*1] **Rui Wang** [*1] **Shitong Luo** [1] **Yue Xu** [1] **Jiahan Li** [1]
Jianzhu Ma [1 2] **Qiang Liu** [1 3] **Yunan Luo** [4] **Jian Peng** [1]

Abstract

Despite the striking success of general protein folding models such as AlphaFold2 (AF2, Jumper et al. (2021)), the accurate computational modeling of antibody-antigen complexes remains a challenging task. In this paper, we first analyze AF2's primary loss function, known as the Frame Aligned Point Error (FAPE), and raise a previously overlooked issue that FAPE tends to face gradient vanishing problem on high-rotational-error targets. To address this fundamental limitation, we propose a novel geodesic loss called Frame Aligned Frame Error (FAFE, denoted as F2E to distinguish from FAPE), which enables the model to better optimize both the rotational and translational errors between two frames. We then prove that F2E can be reformulated as a group-aware geodesic loss, which translates the optimization of the residue-to-residue error to optimizing group-to-group geodesic frame distance. By fine-tuning AF2 with our proposed new loss function, we attain a correct rate of 52.3% (DockQ > 0.23) on an evaluation set and 43.8% correct rate on a subset with low homology, with substantial improvement over AF2 by 182% and 100% respectively. [1]

1. Introduction

Protein structure modeling is a crucial field in computational biology and has been an important unsolved problem for decades. Traditionally, structure biology researchers relied on experimental methods such as X-ray crystallography and NMR spectroscopy (Wüthrich, 2001; Jaskolski et al., 2014; Bai et al., 2015; Thompson et al., 2020) to determine protein structures. However, these methods are time-consuming, expensive, and highly reliant on the accessibility to pure and stable samples of target proteins. As a result, the range of protein monomers and complexes with known structures (Burley et al., 2017) is far limited, especially considering the tremendous growth of discovered protein sequences (Consortium, 2019; Richardson et al., 2023) with the help of advancement in gene sequencing technologies (Braslavsky et al., 2003; Harris et al., 2008; Heather & Chain, 2016). This has led to an increasing interest in computational methods for protein structure prediction, which aim to predict the three-dimensional (3D) structure of proteins based solely on their sequence information.

AlphaFold2 (Jumper et al., 2021) is the most successful protein structure prediction algorithm so far. It has brought great progress in structure prediction accuracy at CASP14 (Kryshtafovych et al., 2021), a well-known protein structure modeling contest. AlphaFold2-Multimer (Evans et al., 2021) was introduced subsequently by applying the framework of AF2 to multi-chain proteins. AF2-Multimer is designed to predict massive biological complexes, especially those with certain cross-chain genetic information. However, for complexes whose cross-chain docking poses can not be revealed by their genetic information, such as immune complexes, there is still a large gap between their predicted structures and experimental counterparts.

Immune complex modeling, which aims to model the 3D structure of an antibody-antigen complex, has important implications in antibody drug discovery (Kaczor et al., 2018). Usually, the antibody part has an unknown structure as a novel sequence, and the antigen part may or may not have a corresponding experimentally determined structure, depending on different therapeutic targets. An alternative way to this problem is using a loop modeling algorithm to predict antibody structure, and applying a rigid docking algorithm between two predicted components.

Many rigid docking algorithms (Pierce et al., 2014; Yan et al., 2020; Desta et al., 2020; Ganea et al., 2021; Jin et al.,

[*]Equal contribution [1]Helixon [2]Tsinghua [3]UTAustin [4]GaTech. Correspondence to: Ruidong Wu <ruidong@helixon.com>, Shitong Luo <luost@helixon.com>, Jianzhu Ma <majianzhu@tsinghua.edu.cn>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).

[1]Code is available at https://github.com/mooninrain/FAFE.git.

2022; Ketata et al., 2023; Wang et al., 2023) have been proposed to address the antibody-antigen docking problem, bringing significant progress in recent years. However, there are several limitations to these approaches. First, the accuracy of such algorithms highly relies on good priors of antibody and antigen structures. When no experimental structure is provided and noises exist in its predicted substitution, those algorithms can easily fail. At the same time, rigid docking algorithms start with the unbound states of antibodies and antigens, thus assume no changes at the epitopes and paratopes during docking, which is not true in the real world. This lack of flexibility can introduce errors on some targets.

To address these limitations, we tackle the immune complex modeling problem directly from the primary sequences of antibodies and antigens. We start by analyzing AF2-Multimer as a baseline which uses FAPE loss as its main structure loss. We show by derivations that cross-chain FAPE loss equals to optimizing the chordal distance between ground truth and predicted group frames on $\mathrm{SE}(3)$. The chordal distance measures the rotational error in its chordal length, which can cause the gradient vanishing problem when the rotation angle is larger than $\frac{\pi}{2}$. This gives us insights into why a portion of AF2-Multimer predictions are "stuck" at wrong docking positions with large rotational errors.

With the observation above, we propose to use Frame Aligned Frame Error (F2E) which not only measures the translational errors but also the rotational errors of local frames. By adding a correcting term, F2E can be modified to approximate the geodesic distance of group frames. We then conduct extensive experiments to show the effectiveness of F2E. Overall, this paper has the following contributions:

- We point out that the original FAPE loss approximates the *chordal distance* loss on group frames, which can cause the gradient vanishing problem.
- We propose a novel *geodesic distance* loss F2E which can address the problem mentioned above.
- Our experiments show that the new loss can improve the antibody-antigen complex modeling performance by a large margin.

2. Background

In this section, we will make a brief introduction of the basic concepts and notations of protein residue frames, as well as widely used metrics to measure the distance on $\mathrm{SO}(3)$ and $\mathrm{SE}(3)$. We will then introduce the basic formula of FAPE loss.

2.1. Frames of protein residues

Proteins are polypeptides composed of amino acid residues. These amino acids are linked by so-called peptide bonds. Once the protein is folded, both the relative translation and rotation will be constrained by the structure to a great extent. These geometric constraints come from chemical and biological mechanisms during the formation of proteins and are crucial for proteins to function properly in organisms.

With the limited number of amino acid types, it is possible to model the structure of each amino acid residue with a certain degree of freedom, for example, adopted in the structure module of AF2 (Jumper et al., 2021), through several frames built upon certain atoms of these residues.

Definition 2.1. A frame $T = (R, \vec{t}) \in \mathrm{SE}(3)$ is a tuple of rotation matrix $R \in \mathbb{R}^{3\times 3}$ and translation vector $\vec{t} \in \mathbb{R}^3$.

$\mathrm{SE}(3)$ is the 3D Special Euclidean Group representing all the rigid transformations, i.e., frames expressed in homogeneous coordinates. Specifically, the rotation part R belongs to 3D Special Orthogonal Group $\mathrm{SO}(3) = \{R \in \mathbb{R}^{3\times 3} : R^\top R = I_3, det(R) = +1\}$, which represents the group of all 3D rotation matrices.

2.2. Metrics on SO(3) and SE(3)

To measure how "close" two frames are, we need to first define the distance metric on $\mathrm{SO}(3)$ and $\mathrm{SE}(3)$. We introduce two commonly used metrics (Huynh, 2009; Hartley et al., 2013; Carlone et al., 2022), *chordal and geodesic distances*.

Definition 2.2. Given two rotations R_i and R_j, their *chordal distance* is defined as

$$dist_c(R_i, R_j) = \|R_i - R_j\|_F = \|R_i^\top R_j - I\|_F, \quad (1)$$

where $\|\cdot\|_F$ denotes the Frobenius norm.

Definition 2.3. Given two rotations R_i and R_j, their *geodesic distance* (or angular distance) is defined as

$$dist_\theta(R_i, R_j) = \left|\arccos\left(\frac{\mathrm{tr}(R_i^\top R_j) - 1}{2}\right)\right|, \quad (2)$$

in which $\mathrm{tr}(\cdot)$ denotes the matrix trace.

A straightforward way to define distance metrics on $\mathrm{SE}(3)$ is to treat it as the Cartesian product of $\mathrm{SO}(3)$ and $\mathbb{R}^3$.

Definition 2.4. Given two frames $T_i = (R_i, \vec{t_i})$ and $T_j = (R_j, \vec{t_j})$, their *chordal distance* is defined as

$$dist_c(T_i, T_j; \alpha) = \sqrt{dist_c(R_i, R_j)^2 + \left\|\frac{\vec{t_j} - \vec{t_i}}{\alpha}\right\|^2} \quad (3)$$

in which α is an axis scaling factor.

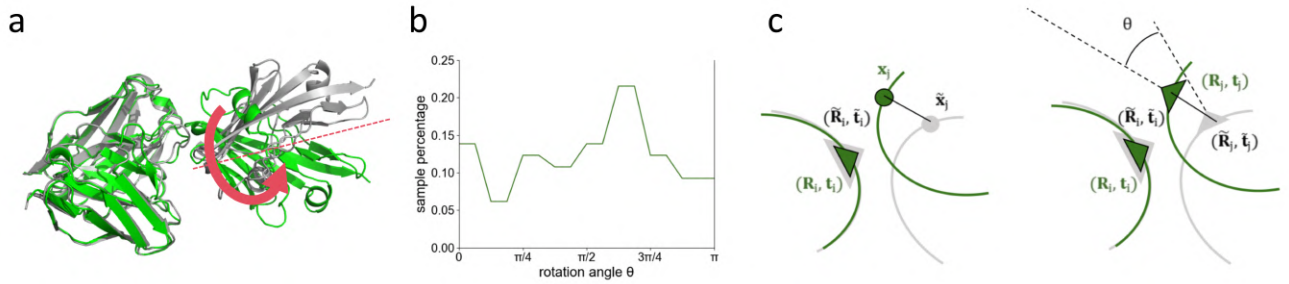

Figure 1. Problem and method overview. (**a**) An example from PDB 7XJF, the gray structure is predicted by AF2, and the green structure is the experimental ground truth. We can see there is a large rotation error. (**b**) Statistics of rotational errors in AF2.3 predictions on our evaluation set. Most of the predictions have a rotation error larger than $\frac{\pi}{2}$. (**c**) The comparison between FAPE (left) and F2E (right). FAPE calculates point-wise Euclidean distance error after alignment to local frames, while F2E calculates frame-wise geodesic distance error after the same alignment.

Definition 2.5. Given two frames $T_i = (R_i, \vec{t_i})$ and $T_j = (R_j, \vec{t_j})$, their *geodesic distance* (or double geodesic distance) is defined as

$$dist_\theta(T_i, T_j; \alpha) = \sqrt{dist_\theta(R_i, R_j)^2 + \left\| \frac{\vec{t_j} - \vec{t_i}}{\alpha} \right\|^2}. \quad (4)$$

Notice that in the following sections, we do not distinguish the term "frame" from "pose", but there are cases where this difference matters, which are discussed in Appendix A.4, We also would like to mention that many other valid distance metrics can be defined on $\mathrm{SO}(3)$ and $\mathrm{SE}(3)$. Some meaningful discussions are provided in Appendix A.3.

2.3. FAPE loss on a parameterized complex

As the standard parameterization used in AF2 (Jumper et al., 2021), proteins are modeled as a collection of N amino acid residues. Each of these residues have a backbone frame T which is built upon N, C_α and C atoms from its backbone parts. The main atom coordinates can be calculated from the frame

$$[N, C, C_\alpha] = T \circ [N^*, C^*, C^*_\alpha], \quad (5)$$

in which N^*, C^*, C^*_α are the canonical coordinates of these atoms on the residue structure, and the transform T is applied to individual coordinates. Note $C^*_\alpha = (0, 0, 0)^\top$, which means the global position of C_α is equal to the translation t from $T = (R, t)$. These three atoms, joined with an O atom connected to C atom, constitute the basic repeating unit of proteins. The protein backbone can thus be parameterized as

$$\mathbf{T} = [T_1, T_2, ..., T_N] \in \mathrm{SE}(3)^N. \quad (6)$$

The sidechain part, which contains at most 10 heavy atoms and several hydrogen atoms, can also be parameterized using several sidechain frames per residue. In the paper, when we refer to residue frames, we are mentioning backbone frames if not specified. However, we do hope to point out that any geometric derivations with backbone frames are also applicable to sidechain frames, as there is no difference between them from a pure geometric perspective.

Specifically, a protein complex is a collection of multiple protein chains, usually denoted in capital letters $[A, B, C...]$. A protein chain is a collection of residues sequentially connected by peptide bonds. A protein complex can therefore be parameterized as

$$\mathbf{T} = [\mathbf{T}_A, \mathbf{T}_B, \mathbf{T}_C, ...], \quad (7)$$

in which $\mathbf{T}_A$ represents the collection of all backbone frames on protein chain A.

FAPE is a loss function originally proposed by AF2 (Jumper et al., 2021). Its basic principle is to estimate the point-wise error after alignment of each local frame of the proteins, as shown in Figure 1 (**c**). FAPE loss allows the calculation of residue-atom pairwise error with the good property of $\mathrm{SE}(3)$ invariance, that is, the loss function is invariant to transformations of predicted or ground truth structure.

Definition 2.6. Given a ground truth complex structure $\mathbf{T} = [T_1, T_2, ...T_N] \in \mathrm{SE}(3)^N$ and its predicted counterpart $\hat{\mathbf{T}} = [\hat{T_1}, \hat{T_2}, ...\hat{T_N}] \in \mathrm{SE}(3)^N$, The *backbone FAPE loss* is defined as

$$L_{\mathrm{FAPE}}(\mathbf{T}, \hat{\mathbf{T}}) = \frac{1}{(N-1)N} \times \quad (8)$$

$$\sum_{i,j \in [1,N], i \neq j} \left\| \hat{T_i}^{-1} \circ \hat{\vec{t}}_j - T_i^{-1} \circ \vec{t}_j \right\|_F, \quad (9)$$

in which $T_j = (R_j, \vec{t_j})$ and $\hat{T}_j = (\hat{R}_j, \hat{\vec{t}}_j)$.

Definition 2.7. Given a ground truth two-chain complex structure $\mathbf{T} = [\mathbf{T}_A, \mathbf{T}_B]$ and its predicted counterpart $\hat{\mathbf{T}} = [\hat{\mathbf{T}}_A, \hat{\mathbf{T}}_B]$, the *inter-chain backbone FAPE loss* is defined as

$$L_{\text{FAPE}}(\mathbf{T}, \hat{\mathbf{T}}) = \frac{1}{N_A N_B} \times \tag{10}$$

$$\sum_{\substack{i \in [1, N_A] \\ j \in [N_A+1, N_A+N_B]}} \|\hat{T}_i^{-1} \circ \hat{\vec{t}}_j - T_i^{-1} \circ \vec{t}_j\|_F, \tag{11}$$

in which residues $[1, N_A]$ belong to chain A and $[N_A + 1, N_A + N_B]$ belong to chain B. The definition on more than 2 chains is trivial by weighted averaging over all chain pairs.

For notation simplicity, in the following words, we use $L_{\text{FAPE}}(i, j)$ to denote the term $\|\hat{T}_i^{-1} \circ \hat{\vec{t}}_j - T_i^{-1} \circ \vec{t}_j\|_F$. We also use notation $i \in \{chainA\}$ and $j \in \{chainB\}$ to indicate residue i belongs to chain A and residue j belongs to chain B, and operator $chain(\cdot)$ to indicate finding the chain of the given residue.

A detailed introduction of FAPE is in Appendix B.

3. FAPE is a chordal distance estimation of group frames

With the defined metrics above, in this section, we discuss the properties of the original FAPE loss adopted in AF2 and AF2-Multimer (Jumper et al., 2021; Evans et al., 2021). We show that with small modifications, FAPE loss can be regarded as a chordal distance estimation of group frames.

3.1. Definition of group frame and its distance

First of all, we need to define the group frame and its metric. This is important because when we work on complex modeling, we care more about inter-chain errors than intra-chain errors. The inter-chain errors, which we will show in the discussions below, mainly come from the incorrect relative pose estimation of different chains.

However, defining a group frame on a non-rigid protein group is not a trivial problem. In this paper, we avoid dealing with this problem directly. Instead, we consider defining the frame differences rather than the frames themselves.

Inspired by the approach in ligand RMSD (Méndez et al., 2003), we can define the distance of frames between predicted structures and ground-truth ones by superposition.

Definition 3.1. Given any frame T in the protein structure and its predicted counterpart $\hat{T}$, $T, \hat{T} \in \text{SE}(3)$, their distance is defined as

$$dist(\hat{T}^{-1}, T^{-1}) = f(\hat{T}T^{-1}) = f(T^{\text{align}}), \tag{12}$$

in which $f(\cdot)$ is the chordal or geodesic distance function and T^{align} is defined as the rigid transformation calculated for superposition of atoms on the frame. The proof for the equation is given in A.1.

The definition above allows us to calculate the group frame difference when the group is non-rigid, which is important in immune complex modeling without ground truth structures. Next, we can calculate the relative pose error of two chains in the form of "diff-diff-frame".

Definition 3.2. Given any frame T_A and T_B on the protein structure, we have $T_A, T_B, \hat{T}_A, \hat{T}_B \in \text{SE}(3)$, we define their relative pose distance error as

$$\begin{aligned} error(A, B; \alpha) &= dist(\hat{T}_A^{-1}\hat{T}_B, T_A{}^{-1}T_B; \alpha) \\ &= dist(\hat{T}_B, T_A^{align}T_B; \alpha). \end{aligned} \tag{13}$$

The distance could either be chordal distance or geodesic distance. See detailed discussion in Appendix A.4.

The definition above makes sure that the relative pose error only relies on the superposition of two chains separately regardless of any specific definition of group frames themselves.

3.2. Group-aware FAPE

In this part, we are going to show that by modifying the original FAPE loss, we can obtain a group-aware version of FAPE, which we shorten by G-FAPE. We show that applying G-FAPE on residue-point pairwise constraints between groups is equivalent to directly optimizing the chordal distance between rigid group frames.

For $i \in \{chainA\}$ and $j \in \{chainB\}$, we denote T_A and T_B as the group frame of chain A and B, $\vec{x}_{i,local}$ as the local position of the C-alpha atom i at T_A. Specifically, we build T_A with its origin at the Euclidean average of all points $\vec{x}_i, i \in \{chainA\}$, which implies

$$\sum_{i \in \{chainA\}} \vec{x}_{i,local} = 0. \tag{14}$$

In addition, for the predicted structures, we assume a Gaussian noise $\vec{\epsilon}_i$ added to each $\vec{x}_{i,local}$. It is then straightforward to write the FAPE loss as

$$\begin{aligned} L_{\text{FAPE}}(i, j) = \|&(T_j{}^{-1}T_A) \circ \vec{x}_{i,local} \\ &- (\hat{T}_j^{-1}\hat{T}_A) \circ (\vec{x}_{i,local} + \vec{\epsilon}_i)\|_F. \end{aligned} \tag{15}$$

Define $T_{j,local} = T_A T_j$, in which T_A is the group frame of chain A. We then introduce the *near-rigid* condition, that is, the local frame error is relatively small compared to the group frame error, and can be ignored. Empirically, we can see from Figure 1 (a) that most local frames satisfy this condition. So we have

$$\begin{aligned} \hat{T}_j^{-1}\hat{T}_A =& \hat{T}_{j,local}^{-1}(\hat{T}_B^{-1}\hat{T}_A) \\ \approx& T_{j,local}{}^{-1}(\hat{T}_B^{-1}\hat{T}_A). \end{aligned} \tag{16}$$

This helps us to simplify FAPE loss as

$$L_{\text{FAPE}}(i,j) = \|(T_B{}^{-1}T_A) \circ \vec{x}_{i,local} - (\hat{T}_B^{-1}\hat{T}_A) \circ (\vec{x}_{i,local} + \vec{\epsilon}_i)\|_F. \tag{17}$$

To further simplify, we denote $(R_\Delta, \vec{t_\Delta}) = T_\Delta = T_B^{-1}T_A$. Now we are ready to introduce G-FAPE, which is simply the quadratic mean of FAPE loss over $i \in \{chainA\}$.

$$\begin{aligned} L_{\text{G-FAPE}}(j) &= \sqrt{\sum_{i\in\{chainA\}} L^2_{\text{FAPE}}(i,j)} \\ &= \Bigg[\sum_{i\in\{chainA\}} \vec{\epsilon}_i^{\top}\vec{\epsilon}_i \\ &\quad + \Big(\sum_{i\in\{chainA\}} \vec{x}_{i,local}^{\top}\vec{x}_{i,local} \Big) \|R_\Delta - \hat{R}_\Delta\|_F^2 \\ &\quad + n\|\vec{t}_\Delta - \hat{\vec{t}}_\Delta\|_F^2 \Bigg]^{\frac{1}{2}} \\ &= \sqrt{\sum_{i\in\{chainA\}} \vec{\epsilon}_i^{\top}\vec{\epsilon}_i + k^2 \cdot dist_c^2(\hat{T}_\Delta, T_\Delta; \alpha)}, \end{aligned} \tag{18}$$

in which $k^2 = \sum_{i\in\{chainA\}} \vec{x}_{i,local}^{\top}\vec{x}_{i,local}$ and $\alpha^2 = \frac{k^2}{n}$, $n = card(\{\text{C-alpha atoms on chain A}\})$. The term $\sum_{i\in\{chainA\}} \vec{\epsilon}_i^{\top}\vec{\epsilon}_i$ represents the intra-chain noises that the inter-chain loss can not optimize. Thus, we can conclude that optimizing G-FAPE is equivalent to optimizing the chordal error of predicted and ground truth T_Δ.

3.3. Gradient vanishing problem in G-FAPE

The chordal distance is a valid metric in many circumstances, but it can cause problems in complex modeling problems. As the Figure 1 (a, b) show, most of the AF2 predictions on our evaluation set have a high rotation error (larger than $\frac{\pi}{2}$).

To understand this phenomenon, in Figure 2 we draw the loss curve of chordal distance to the angle θ of rotation error, which is calculated according to Definition 2.3. We can see the gradient of chordal distance loss is gradually decreasing when θ approaches π. As a result, models trained with FAPE loss will overly concentrate on samples with a small rotation error at the start–for example, samples whose MSAs provide enough contact information to determine the docking pose roughly. Harder samples, like antibody-antigen pairs, are relatively under-trained.

4. Using F2E to estimate geodesic distance of group frames

With the gradient vanishing problem mentioned above, we propose to develop new loss functions to address it. One straightforward idea is to directly optimize $dist_\theta(T_\Delta, \hat{T}_\Delta)$. However, this leads to several difficulties. First of all, according to Definition 3.1 and Definition 3.2, the calculation of T_Δ and $\hat{T}_\Delta$ relies on superposition, which means we need to calculate the gradient of Singular Value Decomposition (SVD), which is numerically unstable. Indeed, when chains are nearly rigid, the singular values will get close and make the gradient unstable. Another potential approach is to estimate T_Δ directly through a separate neural network. We do not use this approach because we want our methods to be compatible with the existing architecture of AF2.

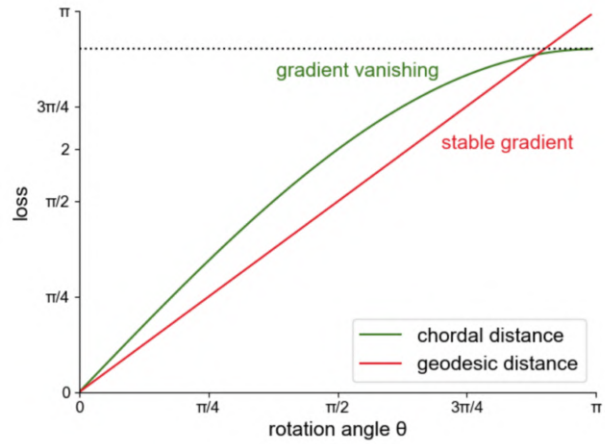

Figure 2. The loss value regarding rotation angle (difference between predicted and groundtruth) for chordal and geodesic distance. The gradient of chordal distance tends to vanish when the rotation angle approaches π, while the geodesic distance provides a stable gradient from 0 to π.

4.1. From FAPE to F2E

We propose to transform the original FAPE loss into a geodesic loss that can keep the gradients stable throughout different rotation angles. To achieve this, we propose Frame Aligned Frame Error (F2E), which measure the geodesic distance between two aligned frames.

Definition 4.1. Given any residue frame i and any residue frame j on the protein structure, we have $\hat{T}_i, T_i, \hat{T}_j, T_j \in \text{SE}(3)$, the F2E loss is defined as

$$L_{\text{F2E}}(i,j;\alpha) = dist_\theta(\hat{T}_i^{-1}\hat{T}_j, T_i{}^{-1}T_j; \alpha), \tag{19}$$

in which $dist_\theta(\cdot)$ denotes the geodesic distance defined in Definition 2.5.

With denotion $(R_{i\to j}, t_{i\to j}) = T_{i\to j} = T_i^{-1}T_j$, we further show that F2E loss can degenerate to FAPE loss, as

$$L^2_{\text{F2E}}(i,j;\alpha) = dist^2_\theta(\hat{R}_{i\to j}, R_{i\to j}) + \frac{L^2_{\text{FAPE}}(i,j)}{\alpha^2}. \tag{20}$$

4.2. Group-aware F2E

We first look into what will happen if we apply F2E in Definition 4.1. With Equation (18) and Equation (20), we can get

$$\begin{aligned}
&\sum_{i\in\{chainA\}} L^2_{\mathrm{F2E}}(i,j) \\
&= \frac{L^2_{\mathrm{G-FAPE}}(j)}{\alpha^2} + \sum_{i\in\{chainA\}} dist^2_\theta(\hat{R}_{i\to j}, R_{i\to j}) \\
&\approx \frac{L^2_{\mathrm{G-FAPE}}(j)}{\alpha^2} + n\cdot dist^2_\theta(\hat{R}_\Delta, R_\Delta) \\
&= \frac{\sum_{i\in\{chainA\}} \vec{\epsilon}_i^{\,\top}\vec{\epsilon}_i + k^2\cdot dist^2_c(\hat{T}_\Delta, T_\Delta; \frac{k}{\sqrt{n}})}{\alpha^2} \\
&\quad + n\cdot dist^2_\theta(\hat{R}_\Delta, R_\Delta) \\
&= \frac{\sum_{i\in\{chainA\}} \vec{\epsilon}_i^{\,\top}\vec{\epsilon}_i}{\alpha^2} + \frac{n}{\alpha^2}\left\|\hat{\vec{t}}_\Delta - \vec{t}_\Delta\right\|^2 \\
&+ \frac{k^2}{\alpha^2}\cdot dist^2_c(\hat{R}_\Delta, R_\Delta) + n\cdot dist^2_\theta(\hat{R}_\Delta, R_\Delta).
\end{aligned} \tag{21}$$

Notice that we get both geodesic and chordal distance error in the result. Fortunately, the chordal error can be inferred from the geodesic error.

Lemma 4.2. *Given two rotations R_i and $R_j \in \mathrm{SO}(3)$, we have*

$$d_c(R_i, R_j) = 2\sqrt{2}\sin(\frac{d_\theta(R_i, R_j)}{2}), \tag{22}$$

in which $d_c(\cdot)$ denotes the chordal distance defined in Definition 2.2 and $d_\theta(\cdot)$ denotes the geodesic distance defined in Definition 2.3. The proof is provided in Appendix A.2.

So we can actually "cancel" the chordal error term by introducing group-aware F2E (G-F2E) as

Definition 4.3. Given any residue frame i and any residue frame j on the protein structure, $chain(i) \neq chain(j)$, G-F2E loss is defined as

$$\begin{aligned}
L_{\mathrm{G-F2E}}(i,j;\alpha) = \Bigg[&\frac{L^2_{\mathrm{FAPE}}(i,j)}{\alpha^2} \\
&+ \theta^2 - \frac{8\vec{x}^{\,\top}_{i,local}\vec{x}_{i,local}}{\alpha^2}\cdot\sin^2\frac{\theta}{2}\Bigg]^{\frac{1}{2}},
\end{aligned} \tag{23}$$

in which $\theta = dist_\theta(\hat{R}_{i\to j}, R_{i\to j})$, and $\vec{x}_{i,local}$ is the coordinates at the group frame on $chain(i)$. We repeat that the group frame is built with its origin at the Euclidean average point of all C-alpha atoms, and without assumptions about its rotation.

5. Experiments

We analyze the effectiveness of our proposed loss by fine-tuning from original AF2-Multimer weights.

5.1. Parameter-efficient fine-tuning of AF2-Multimer

We conduct experiments by fine-tuning the original AF2 weights. The fine-tuning dataset is collected from The Structural Antibody Database (SabDab) (Dunbar et al., 2014), which provides annotations on the original PDB database (Burley et al., 2017) raw structures. The training set only includes samples released before September 30th, 2021, which is the same cut off for AF2.3 training set. More data preparation details are available in Appendix C.1.

In the early version of F2E, we tried both to finetune the full model and to only finetune the structure module. Fine-tuning the full model leads to a severe overfitting problem, where we find the accuracy of training samples arise fast while the evaluation performance drops very soon after a small rise, especially on LDDT, which suggests a memory loss of the intra-chain structure information. while the structure modules finetuning only leads to a small increase in performance. We therefore introduce an intermediate solution with Low-Rank Adaption (LoRA) (Hu et al., 2022). Note that we use LoRA not because of the computational resources exceed our capacity, but because it provides a natural path to avoid catastrophic forgetting.

We search and discover a relatively good LoRA setting for fine-tuning AF2, where the LoRA layers are only added to 48-layer evoformer, with other weights freezed. All the weights before the evoformer like template embedder and extra MSA stack are freezed, and all the weights after the evoformer including the structure module and all the prediction heads are fully trainable. More training details can be found in Appendix C.2.

Following the standard AF2 settings, we fine-tune 5 models and report the performance of top1 prediction with iptm as the confidence score. A discussion of inference details is provided in Appendix C.3.

5.2. Results

In Figure 3 we show the experimental results on two held evaluation datasets. Our protocol follows in principle that in AlphaFold-latest (AF3) (DeepMind, 2023). We first filter a full evaluation dataset with all the samples after the training cut off, getting 693 unique samples. We use MMseqs2 (Steinegger & Söding, 2017) to cluster all the antigen sequences from both training and evaluation set. For the full evaluation set, we select all the cluster centers. For low homology evaluation set, we discard all the data samples whose antigens belong to a cluster with training samples.

All the results reported in Figure 3 are top1 predictions across 5 models plus 5 seeds with iptm as the confidence score. We use DockQ (Basu & Wallner, 2016) as the evaluation metric for antibody-antigen complex modeling. As commonly suggested (DeepMind, 2023), DockQ>0.23 is

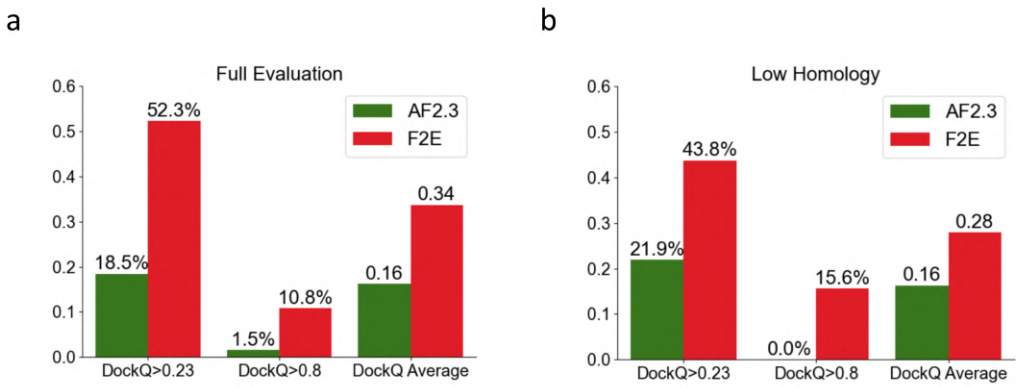

Figure 3. Comparison of F2E with AF2.3 on two held evaluation sets. (**a**) The evaluation results on full evaluation set. For columns "DockQ>0.23" and "DockQ>0.8" the percentage is reported, and average DockQ is reported in the last column. (**b**) The evaluation results on low homology set with novel antigens.

considered accurate predictions, while DockQ>0.8 is consider high accuracy. The experimental results show that F2E is able to increase the accuracy in the definition of DockQ>0.23 by a large margin, bringing 182% increase on the full evaluation set and 100% increase on the low homology set. In addition, our fine-tuned model is able to predict a number of samples in high accuracy, which is rare for original AF2.3 predictions. We notice the impressive performance of AF3 (DeepMind, 2023), but we can not build a fair benchmark with it because AF3 predicts the structure of the whole complex with the help of small molecules, peptides and other protein chains. We expect to extend our results on newer protein foundation models with full-atom prediction ability in the future.

5.3. Ablation tests

To further analyze the effectiveness of different components of our approach, we conduct ablation tests for different settings with results shown in Table 1. All results in the table is reported by fine-tuning the same AF2 model and inference with 5 random seeds. The only exception is the top row with ensemble on, which suggests the most-confident result of 5 models from the same protocol in Figure 3. The bottom baseline is to inference with the original model weights. We also add the baseline of AF2sample (Wallner, 2023). AF2sample is an agressive sampling protocol using AF2.1 and AF2.2 with multiple strategies, including removing templates, re-sampling MSAs, increasing the recycling rounds and adding dropout during inference time. More than 200 samples are proposed for each target. We think AF2sample represents the full potential of AF2 without fine-tuning.

We can see that single-model F2E is able to reach the same performance of AF2sample, and a simple ensemble with 25 samples in total already surpasses it, suggesting a non-trivial improvement. Additionally we can conclude from Table 1: 1) group-aware adaptation of FAPE is not only theoretically sound, but also brings an increase in performance. 2) F2E further increases DockQ compared to G-FAPE, suggesting the effectiveness of stablizing the gradient across rotation errors. 3) The ensemble technique mainly improves the average DockQ (from 0.210 to 0.279) of existing successful predicitions but does not improve the percentage of success as much (from 0.375 to 0.438). Moreover We find that the best predictions according to ground truth DockQ can reach a high accuracy rate of 0.59. This suggests necessity for further improvement in the confidence score.

Table 1. Ablations study. The loss with "AF2" means no fine-tuning, "AF2S" means AF2sample. "G-aware" indicates whether to use the group-aware version of the loss and "ensemble" means top1 prediction across 5 models with iptm as the confidence score.

Loss	LoRA	G-aware	Ensemble	DockQ > 0.23
F2E	√	√	√	**0.438**
F2E	√	√	×	0.375
FAPE	√	√	×	0.313
FAPE	√	×	×	0.250
FAPE	×	×	×	0.219
AF2S	×	×	×	0.375
AF2	×	×	×	0.188

5.4. Case study and error analysis

In Figure 4 (**a**) we show the change of rotation error distributions before and after fine-tuning with F2E. We can see that the rotation error is dramatically reduced with many data points drop below $\frac{\pi}{4}$. In Figure 4 (**b**) and (**c**) we show the predictions results of AF2.3 and F2E on 7Y1B and 8GPT. By correcting the rotation error, F2E helps the model the make correct predictions. Notice that in (**b**) the alignment is conducted on antibody structures, and no translational error can be observed in this way. In (**c**) the alignment is conducted on antigen structures, and the pose error not

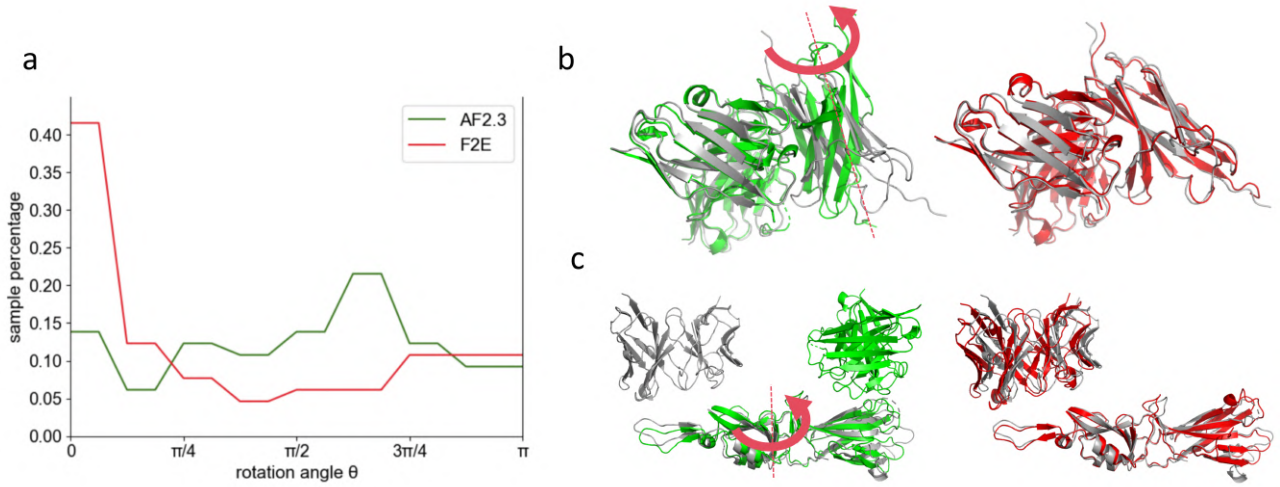

Figure 4. (**a**) The visualization of rotational errors for AF2.3 and F2E. (**b**) The predicted structures of PDB 7Y1B against ground truth. The grey structure is the experimental ground truth, the green structure by AF2.3 and the red structure by model tuned with F2E. We show in red arrows how to rotate from the wrong prediction to the ground truth. (**c**) The predicted structures of 8GPT against ground truth.

only introduces rotational error but also translational error. Nevertheless, such kind of error can be corrected by only rotating the antigen alone the axis shown in the left image. A meaningful discussion about calculating relative pose error from different perspectives is provided in Appendix A.4.

6. Related work

Protein structure prediction The success of AF2 (Jumper et al., 2021) has brought an increasing interest in developing new protein structure prediction models. Some of these efforts (Baek et al., 2021; Wu et al., 2022; Lin et al., 2022; Baek et al., 2023a) try to improve the inference speed of AF2 either by speeding up the neural network or reducing the demand for MSA searching. Evans et al. (2021), Gao et al. (2022) and Baek et al. (2023a) are later developed to predict complex structures. Recently, several new models (Baek et al., 2023b; Krishna et al., 2023; DeepMind, 2023) further extend their ability to small molecules, nucleic acids and other hetero components. Several papers (Wallner, 2023; Bryant & Noe, 2023; Yin & Pierce, 2023; Gaudreault et al., 2023) propose to apply or enhance AF2 on immune complex modeling. However, those methods mainly focus on inference-time enhancement.

Protein rigid docking Many algorithms are designed to solve the problem of rigid docking problem from two known structures. There have been many efforts (Pierce et al., 2014; Yan et al., 2020; Desta et al., 2020) taken with grid-search-based or template-based approaches. Recently, deep learning based docking (Ganea et al., 2021; Jin et al., 2022; Wang et al., 2023; Ketata et al., 2023) gains more and more attention. Our setting is different from rigid docking with no known structures as input.

Parameter efficient fine-tuning We use LoRA (Hu et al., 2022) in our work as the parameter-efficient finetuning method selected among a wealth of recently proposed methods (Lester et al., 2021; He et al., 2022; Dettmers et al., 2023; Xia et al., 2024) for its simplicity and speed. These methods reduce the cost of fine-tuning large models by training either additional or a subset of parameters and often reach comparable performance with full-parameter finetuning.

7. Conclusion

Computational immune complex modeling is a challenging task which is important to downstream applications in antibody drug design. In this paper, we propose F2E, a geodesic distance loss that can solve the gradient vanishing problem in the original FAPE loss from AF2 (Gao et al., 2022). We introduce a theoretically sound way to correct inter-chain FAPE loss as Group-aware FAPE loss, which becomes a chordal distance loss for relative pose errors of nearly-rigid groups defined on protein chains. The similar derivation can help us find a way to correct inter-chain F2E as Group-aware F2E loss. We conduct extensive experiments and analysis to show the effectiveness of our approach.

For future directions, we expect to extend our loss to different complex modeling problems, including those with non-protein components. Indeed, the implicit definition of group frame indicates that F2E can be easily adapted to these components regardless of their inner geometry. Even though our experiments are conducted by fine-tuning AF2, we expect F2E can be also beneficial during pre-training stage of protein foundation models. We really expect to apply F2E to multiple scenarios to test its generalizability.

Acknowledgements

This work was supported by the National Key Research and Development Program of China grants 2022YFF1203100.

Impact Statement

This work aims to advance machine learning models in protein complex modeling. This is an important issue in many aspects of the pharmaceutical industry. For instance, antibody antigen complex modeling is of great importance in developing vaccines. High performance models in this area have the potential of revolutionizing the field of drug discovery.

References

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., Anishchenko, I., Humphreys, I., Cong, Q., Baker, D., and DiMaio, F. Efficient and accurate prediction of protein structure using rosettafold2. *bioRxiv*, pp. 2023–05, 2023a.

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, 2023b.

Bai, X.-C., McMullan, G., and Scheres, S. H. How cryo-em is revolutionizing structural biology. *Trends in biochemical sciences*, 40(1):49–57, 2015.

Basu, S. and Wallner, B. Dockq: a quality measure for protein-protein docking models. *PloS one*, 11(8): e0161879, 2016.

Braslavsky, I., Hebert, B., Kartalov, E., and Quake, S. R. Sequence information can be obtained from single dna molecules. *Proceedings of the National Academy of Sciences*, 100(7):3960–3964, 2003.

Bryant, P. and Noe, F. Improved protein complex prediction with alphafold-multimer by denoising the msa profile. *bioRxiv*, pp. 2023–07, 2023.

Burley, S. K., Berman, H. M., Kleywegt, G. J., Markley, J. L., Nakamura, H., and Velankar, S. Protein data bank (pdb): the single global macromolecular structure archive. *Protein crystallography: methods and protocols*, pp. 627–641, 2017.

Carlone, L., Khosoussi, K., Tzoumas, V., Habibi, G., Ryll, M., Talak, R., Shi, J., and Antonante, P. Visual navigation for autonomous vehicles: An open-source hands-on robotics course at mit. In *2022 IEEE Integrated STEM Education Conference (ISEC)*, pp. 177–184. IEEE, 2022.

Consortium, U. Uniprot: a worldwide hub of protein knowledge. *Nucleic acids research*, 47(D1):D506–D515, 2019.

DeepMind. A glimpse of the next generation of alphafold, 2023.

Desta, I. T., Porter, K. A., Xia, B., Kozakov, D., and Vajda, S. Performance and its limits in rigid body protein-protein docking. *Structure*, 28(9):1071–1081, 2020.

Dettmers, T., Pagnoni, A., Holtzman, A., and Zettlemoyer, L. Qlora: Efficient finetuning of quantized llms. *arXiv preprint arXiv:2305.14314*, 2023.

Dunbar, J. and Deane, C. M. Anarci: antigen receptor numbering and receptor classification. *Bioinformatics*, 32(2):298–300, 2016.

Dunbar, J., Krawczyk, K., Leem, J., Baker, T., Fuchs, A., Georges, G., Shi, J., and Deane, C. M. Sabdab: the structural antibody database. *Nucleic acids research*, 42 (D1):D1140–D1146, 2014.

Evans, R., O'Neill, M., Pritzel, A., Antropova, N., Senior, A., Green, T., Žídek, A., Bates, R., Blackwell, S., Yim, J., et al. Protein complex prediction with alphafold-multimer. *biorxiv*, pp. 2021–10, 2021.

Ganea, O.-E., Huang, X., Bunne, C., Bian, Y., Barzilay, R., Jaakkola, T., and Krause, A. Independent se (3)-equivariant models for end-to-end rigid protein docking. *arXiv preprint arXiv:2111.07786*, 2021.

Gao, M., Nakajima An, D., Parks, J. M., and Skolnick, J. Af2complex predicts direct physical interactions in multimeric proteins with deep learning. *Nature communications*, 13(1):1744, 2022.

Gaudreault, F., Corbeil, C. R., and Sulea, T. Enhanced antibody-antigen structure prediction from molecular docking using alphafold2. *Scientific Reports*, 13(1): 15107, 2023.

Harris, T. D., Buzby, P. R., Babcock, H., Beer, E., Bowers, J., Braslavsky, I., Causey, M., Colonell, J., DiMeo, J., Efcavitch, J. W., et al. Single-molecule dna sequencing of a viral genome. *Science*, 320(5872):106–109, 2008.

Hartley, R., Trumpf, J., Dai, Y., and Li, H. Rotation averaging. *International journal of computer vision*, 103: 267–305, 2013.

He, J., Zhou, C., Ma, X., Berg-Kirkpatrick, T., and Neubig, G. Towards a unified view of parameter-efficient transfer learning. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=0RDcd5Axok.

Heather, J. M. and Chain, B. The sequence of sequencers: The history of sequencing dna. *Genomics*, 107(1):1–8, 2016.

Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., and Chen, W. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Huynh, D. Q. Metrics for 3d rotations: Comparison and analysis. *Journal of Mathematical Imaging and Vision*, 35:155–164, 2009.

Jaskolski, M., Dauter, Z., and Wlodawer, A. A brief history of macromolecular crystallography, illustrated by a family tree and its n obel fruits. *The FEBS journal*, 281(18): 3985–4009, 2014.

Jin, W., Barzilay, R., and Jaakkola, T. Antibody-antigen docking and design via hierarchical structure refinement. In *International Conference on Machine Learning*, pp. 10217–10227. PMLR, 2022.

Jumper, J., Evans, R., Pritzel, A., Green, T., Figurnov, M., Ronneberger, O., Tunyasuvunakool, K., Bates, R., Žídek, A., Potapenko, A., et al. Highly accurate protein structure prediction with alphafold. *Nature*, 596(7873):583–589, 2021.

Kaczor, A. A., Bartuzi, D., Stepniewski, T. M., Matosiuk, D., and Selent, J. Protein–protein docking in drug design and discovery. *Computational Drug Discovery and Design*, pp. 285–305, 2018.

Ketata, M. A., Laue, C., Mammadov, R., Stärk, H., Wu, M., Corso, G., Marquet, C., Barzilay, R., and Jaakkola, T. S. Diffdock-pp: Rigid protein-protein docking with diffusion models. *arXiv preprint arXiv:2304.03889*, 2023.

Krishna, R., Wang, J., Ahern, W., Sturmfels, P., Venkatesh, P., Kalvet, I., Lee, G. R., Morey-Burrows, F. S., Anishchenko, I., Humphreys, I. R., et al. Generalized biomolecular modeling and design with rosettafold all-atom. *bioRxiv*, pp. 2023–10, 2023.

Kryshtafovych, A., Schwede, T., Topf, M., Fidelis, K., and Moult, J. Critical assessment of methods of protein structure prediction (casp)—round xiv. *Proteins: Structure, Function, and Bioinformatics*, 89(12):1607–1617, 2021.

Lester, B., Al-Rfou, R., and Constant, N. The power of scale for parameter-efficient prompt tuning. In Moens, M.-F., Huang, X., Specia, L., and Yih, S. W.-t. (eds.), *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 3045–3059, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.243. URL https://aclanthology.org/2021.emnlp-main.243.

Lin, Z., Akin, H., Rao, R., Hie, B., Zhu, Z., Lu, W., dos Santos Costa, A., Fazel-Zarandi, M., Sercu, T., Candido, S., et al. Language models of protein sequences at the scale of evolution enable accurate structure prediction. *BioRxiv*, 2022:500902, 2022.

Méndez, R., Leplae, R., De Maria, L., and Wodak, S. J. Assessment of blind predictions of protein–protein interactions: current status of docking methods. *Proteins: Structure, Function, and Bioinformatics*, 52(1):51–67, 2003.

Pierce, B. G., Wiehe, K., Hwang, H., Kim, B.-H., Vreven, T., and Weng, Z. Zdock server: interactive docking prediction of protein–protein complexes and symmetric multimers. *Bioinformatics*, 30(12):1771–1773, 2014.

Richardson, L., Allen, B., Baldi, G., Beracochea, M., Bileschi, M. L., Burdett, T., Burgin, J., Caballero-Pérez, J., Cochrane, G., Colwell, L. J., et al. Mgnify: the microbiome sequence data analysis resource in 2023. *Nucleic Acids Research*, 51(D1):D753–D759, 2023.

Steinegger, M. and Söding, J. Mmseqs2 enables sensitive protein sequence searching for the analysis of massive data sets. *Nature biotechnology*, 35(11):1026–1028, 2017.

Thompson, M. C., Yeates, T. O., and Rodriguez, J. A. Advances in methods for atomic resolution macromolecular structure determination. *F1000Research*, 9, 2020.

Wallner, B. Afsample: improving multimer prediction with alphafold using massive sampling. *Bioinformatics*, 39(9): btad573, 2023.

Wang, R., Sun, Y., Luo, Y., Li, S., Yang, C., Cheng, X., Li, H., Shi, C., and Song, L. Injecting multimodal information into rigid protein docking via bi-level optimization. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

Wu, R., Ding, F., Wang, R., Shen, R., Zhang, X., Luo, S., Su, C., Wu, Z., Xie, Q., Berger, B., et al. High-resolution de novo structure prediction from primary sequence. *BioRxiv*, pp. 2022–07, 2022.

Wüthrich, K. The way to nmr structures of proteins. *Nature structural biology*, 8(11):923–925, 2001.

Xia, W., Qin, C., and Hazan, E. Chain of lora: Efficient fine-tuning of language models via residual learning, 2024.

Yan, Y., Tao, H., He, J., and Huang, S.-Y. The hdock server for integrated protein–protein docking. *Nature protocols*, 15(5):1829–1852, 2020.

Yin, R. and Pierce, B. G. Evaluation of alphafold antibody-antigen modeling with implications for improving predictive accuracy. *Protein Science*, pp. e4865, 2023.

A. SO(3) and SE(3) distance metrics

A.1. The distance of two frames can be calculated by their composition

Definition 3.1 relies on the proof that the distance of two frames can be calculated by their composition. Here we provide a brief proof for it.

Proof. According to Definition 2.4 and Definition 2.5, we have
$dist(\hat{T}, T) = f(\hat{R}^\top R, \vec{t} - \hat{\vec{t}})$,
in which $dist(\cdot)$ denotes either chordal or geodesic distance.
And we know that the composition of $\hat{T}^{-1}T$ equals to $(\hat{R}^\top R, \hat{R}^\top(\vec{t} - \hat{\vec{t}}))$,
since $f(\hat{R}R^\top, t - \hat{t}) = f(\hat{R}R^\top, \hat{R}^\top(\vec{t} - \hat{\vec{t}}))$,
we have $dist(\hat{T}, T) = f(\hat{T}^{-1}T)$.
So we have $dist(\hat{T}^{-1}, T^{-1}) = f(\hat{T}T^{-1})$. □

A.2. Relationship of $\mathrm{SO}(3)$ chordal and geodesic distance

Proposition A.1. *Given two rotations R_i and $R_j \in \mathrm{SO}(3)$, we have*

$$d_c(R_i, R_j) = 2\sqrt{2}sin(\frac{d_\theta(R_i, R_j)}{2}), \tag{24}$$

in which $d_c(*)$ denotes the chordal distance defined in Definition 2.2 and $d_\theta(*)$ denotes the geodesic distance defined in Definition 2.3.

Proof.

$$\begin{aligned} d_c(R_i, R_j) &= \|R_i - R_j\|_F \\ &= \|R_i^\top R_j - I\|_F \\ &= 2(\sin^2(d_\theta(R_i, R_j)) + (1 - \cos(d_\theta(R_i, R_j)))^2) \qquad \text{(Rodrigues}' \text{ formula)} \\ &= 2\sqrt{2}\sin(\frac{d_\theta(R_i, R_j)}{2}) \end{aligned} \tag{25}$$

□

A.3. Validation of metrics

Many excellent reviews (Huynh, 2009; Hartley et al., 2013; Carlone et al., 2022) have discussed the distance metrics on $\mathrm{SO}(3)$ and $\mathrm{SE}(3)$. Here we only provide brief proof that the metrics introduced in 2.2 are valid. A *valid* distance $dist(a, b)$ between two generic elements "a" and "b" must satisfy the following properties.

$$dist(a, b) \geq 0 \qquad (\text{non} - \text{negativity}) \tag{26}$$

$$dist(a, b) = 0 \iff a = b \qquad (\text{identity}) \tag{27}$$

$$dist(a, b) = dist(b, a) \qquad (\text{symmetry}) \tag{28}$$

$$dist(a, c) \leq dist(a, b) + dist(b, c) \qquad (\text{triangle inequality}) \tag{29}$$

We prove the chordal distance on $\mathrm{SO}(3)$ defined in Definition 2.2 satisfies the properties above.

Proposition A.2. *Given two rotations R_i and $R_j \in \mathrm{SO}(3)$, their chordal distance defined as*

$$dist_c(R_i, R_j) = \|R_i - R_j\|_F = \|R_i^\top R_j - I\|_F \tag{30}$$

is a valid distance metric.

Proof. The proof for the first three properties is straightforward. Thus we only prove the triangle inequality.

$$\begin{aligned} & dist_c(R_a, R_c) \leq dist_c(R_a, R_b) + dist_c(R_b, R_c) \\ \iff & \|R_a - R_c\|_F \leq \|R_a - R_b\|_F + \|R_b - R_c\|_F \\ \iff & \|\,vec(R_a) - vec(R_c)\|_F \leq \|\,vec(R_a) - vec(R_b)\|_F + \|\,vec(R_b) - vec(R_c)\|_F, \end{aligned} \tag{31}$$

which can be implied from triangle inequation for vectors. □

The validation of geodesic distance on $\mathrm{SO}(3)$ is by definition, as the geodesic distance equals the shortest path between two points on the manifold. The validation of metrics on $\mathrm{SE}(3)$ defined in Definition 2.4 and Definition 2.5 are also trivial, as they treat $\mathrm{SE}(3)$ as the Cartesian product of $\mathrm{SO}(3)$ and $\mathbb{R}^3$.

A.4. Relative pose error

Chordal distance and geodesic distance on $\mathrm{SE}(3)$ are left-invariant only. This means usually we have

$$dist(\hat{T}_A^{-1}\hat{T}_B, {T_A}^{-1}T_B) \neq dist(\hat{T}_A^{-1}T_A, \hat{T}_B^{-1}T_B) \tag{32}$$

$$dist(\hat{T}_A^{-1}\hat{T}_B, {T_A}^{-1}T_B) \neq dist(\hat{T}_B^{-1}\hat{T}_A, {T_B}^{-1}T_A), \tag{33}$$

which means there can be multiple inequivalent approaches to optimize relative pose error depending on the definition. However, the good news is that chordal distance and geodesic distance on $\mathrm{SO}(3)$ are left-invariant and right-invariant, so their rotation errors are the same. Their optimization destination is the same too, because once the rotation error is reduced to zero, their translation errors become equivalent. We prove that the rotation errors on both sides in Inequation 32 are equivalent.

Proposition A.3.

$$dist_c(\hat{R}_i^\top R_i, \hat{R}_j^\top R_j) = dist_c(\hat{R}_i^\top \hat{R}_j, {R_i}^\top R_j) \tag{34}$$

Proof.

$$\begin{aligned}
& dist_c(\hat{R}_i^\top R_i, \hat{R}_j^\top R_j) \\
=& \|{\hat{R}_i}^\top R_i - \hat{R}_j^\top R_j\|_F \\
=& \|\hat{R}_j {\hat{R}_i}^\top R_i - R_j\|_F \\
=& \|\hat{R}_j {\hat{R}_i}^\top - R_j {R_i}^\top\|_F \\
=& \|\hat{R}_i^\top \hat{R}_j - {R_i}^\top R_j\|_F \\
=& dist_c(\hat{R}_i^\top \hat{R}_j, {R_i}^\top R_j)
\end{aligned} \tag{35}$$

□

Proposition A.4.

$$dist_\theta(\hat{R}_i^\top R_i, \hat{R}_j^\top R_j) = dist_\theta(\hat{R}_i^\top \hat{R}_j, {R_i}^\top R_j) \tag{36}$$

Proof.

$$\begin{aligned}
& dist_\theta(\hat{R}_i^\top R_i, \hat{R}_j^\top R_j) \\
=& |\arccos(\frac{\mathrm{tr}(({\hat{R}_i^\top R_i})^\top \hat{R}_j^\top R_j) - 1}{2})| \\
=& |\arccos(\frac{\mathrm{tr}(({\hat{R}_i^\top \hat{R}_j})^\top {R_i}^\top R_j) - 1}{2})| \\
=& dist_\theta(\hat{R}_i^\top \hat{R}_j, {R_i}^\top R_j)
\end{aligned} \tag{37}$$

□

The rotation error of Inequation 33 can be proved in the same way.

B. FAPE loss and F2E loss details

B.1. FAPE loss details

In the original AF2 (Jumper et al., 2021) paper, There are two places where FAPE loss is applied. One is called auxiliary FAPE loss which is calculated in the intermediate backbone prediction of each cycle in the structure module. The other is called full FAPE loss which is calculated in the final full-atom prediction from the last cycle, which is reconstructed from backbone frames and sidechain torsion angles predicted. We denote the FAPE loss on different cycles as

Definition B.1. Given any residue frame i and any atom j on the protein structure, we have $\hat{T}_i, T_i \in \mathrm{SE}(3)$ and $\hat{\vec{x}}_j, \vec{x}_j \in \mathbb{R}^3$, the FAPE loss on the predicted structure on cycle $c \in [0, N_{cycle} - 1]$ is defined as

$$L_{\mathrm{FAPE}}(i, j, c) = \|\hat{T}_{i,c}^{-1} \circ \hat{\vec{x}}_{j,c} - {T_{i,c}}^{-1} \circ \vec{x}_{j,c}\|_F, \tag{38}$$

in which $\| * \|_F$ denotes the Frobenius norm.

The FAPE loss applied to monomers can then be written as

$$L = L_{\text{Full FAPE}} + L_{\text{Auxiliary FAPE}} \tag{39}$$

$$L_{\text{Full FAPE}} = \frac{1}{N_{pair}} \sum_{i \in \{\text{full frames}\}, j \in \{\text{full atoms}\}} L_{\mathrm{FAPE}}(i, j, c = N - 1) \tag{40}$$

$$L_{\text{Auxiliary FAPE}} = \frac{1}{N_{cycle} N_{pair}} \sum_{c=0}^{N-1} \sum_{i \in \{\text{backbone frames}\}, j \in \{\text{C-alpha atoms}\}} L_{\mathrm{FAPE}}(i, j, c), \tag{41}$$

in which N_{pair} is determined per summation notation. The FAPE loss applied to multimers can be written as

$$L = L_{\text{Full FAPE}} + L_{\text{Intra-chain auxiliary FAPE}} + L_{\text{Inter-chain auxiliary FAPE}} \tag{42}$$

$$L_{\text{Full FAPE}} = \frac{1}{N_{pair}} \sum_{i \in \{\text{full frames}\}, j \in \{\text{full atoms}\}} L_{\mathrm{FAPE}}(i, j, c = N - 1) \tag{43}$$

$$L_{\text{Intra-chain auxiliary FAPE}} = \frac{1}{N_{cycle} N_{pair}} \sum_{c=0}^{N-1} \sum_{i \in \{\text{backbone frames}\}, j \in \{\text{C-alpha atoms}\}, \text{chain(i)=chain(j)}} L_{\mathrm{FAPE}}(i, j, c) \tag{44}$$

$$L_{\text{Inter-chain auxiliary FAPE}} = \frac{1}{N_{cycle} N_{pair}} \sum_{c=0}^{N-1} \sum_{i \in \{\text{backbone frames}\}, j \in \{\text{C-alpha atoms}\}, \text{chain(i)}\neq\text{chain(j)}} L_{\mathrm{FAPE}}(i, j, c) \tag{45}$$

$$\tag{46}$$

When using the group-aware version of FAPE, we only change the inter-chain part as

$$L_{\text{Inter-chain auxiliary FAPE, group-aware}} = \frac{1}{N_{cycle} N_{pair}} \sum_{c=0}^{N-1} \sqrt{\sum_{i \in \{\text{backbone frames}\}, j \in \{\text{C-alpha atoms}\}, \text{chain(i)}\neq\text{chain(j)}} L_{\mathrm{FAPE}}(i, j, c)^2}. \tag{47}$$

Remind that from the derivations in Section 3.2, only the quadratic mean over j is required, But for implementation simplicity, we apply it to both i and j. This works well empirically.

B.2. F2E loss details

We denote the F2E loss on different cycles as

Definition B.2. Given any residue frame i and any frame j on the protein structure, we have $\hat{T}_i, T_i, \hat{T}_j, T_j \in \mathrm{SE}(3)$, the F2E loss on the predicted structure on cycle $c \in [0, N - 1]$ is defined as

$$L_{\mathrm{F2E}}(i, j, c; \alpha) = dist_\theta(\hat{T}_{i,c}^{-1} \hat{T}_{j,c}, {T_{i,c}}^{-1} T_{j,c}; \alpha), \tag{48}$$

in which $dist_\theta(*)$ denotes the geodesic distance defined in Definition 2.5.

When we instead use the loss of F2E, we apply F2E to all terms in 42. So we get

$$L = L_{\text{Full F2E}} + L_{\text{Intra-chain auxiliary F2E}} + L_{\text{Inter-chain auxiliary F2E}} \tag{49}$$

$$L_{\text{Full F2E}} = \frac{1}{N_{pair}} \sum_{i \in \{\text{full frames}\}, j \in \{\text{full frames}\}} L_{\text{F2E}}(i, j, c = N-1) \tag{50}$$

$$L_{\text{Intra-chain auxiliary F2E}} = \frac{1}{N_{cycle} N_{pair}} \sum_{c=0}^{N-1} \sum_{i \in \{\text{backbone frames}\}, j \in \{\text{backbone frames}\}, \text{chain(i)=chain(j)}} L_{\text{F2E}}(i, j, c) \tag{51}$$

$$L_{\text{Inter-chain auxiliary F2E}} = \frac{1}{N_{cycle} N_{pair}} \sum_{c=0}^{N-1} \sum_{i \in \{\text{backbone frames}\}, j \in \{\text{backbone frames}\}, \text{chain(i)} \neq \text{chain(j)}} L_{\text{F2E}}(i, j, c) \tag{52}$$

$$\tag{53}$$

When using the group-aware version of F2E, we only change the inter-chain part as

$$L_{\text{Inter-chain auxiliary F2E, group-aware}} = \frac{1}{N_{cycle} N_{pair}} \sum_{c=0}^{N-1} \sqrt{\sum_{i \in \{\text{backbone frames}\}, j \in \{\text{backbone frames}\}, \text{chain(i)} \neq \text{chain(j)}} L_{\text{F2E, group-aware}}(i, j, c)^2}. \tag{54}$$

B.3. Clamping Details

The original FAPE loss is applied with clamping. Specifically, for some samples the loss is only applied to residue pairs within certain distance. In the early version of F2E, we apply the same restriction to the $R(3)$ part of the loss. However, later experiments show that removing the gradients of the $SO(3)$ at the same time can stabilize and accelerate the training procedure.

B.4. Pytorch implementation of F2E

We provide an example implementation of F2E in PyTorch 2.0.1.

```
import torch
import einops
from typing import TypeAlias, Callable, Any

Tensor: TypeAlias = torch.Tensor

def invert_rigid(R: Tensor, t: Tensor):
    """Invert rigid transformation.

    Args:
        R: Rotation matrices, (..., 3, 3).
        t: Translation, (..., 3).

    Returns:
        R_inv: Inverted rotation matrices, (..., 3, 3).
        t_inv: Inverted translation, (..., 3).
    """
    R_inv = R.transpose(-1, -2)
    t_inv = -torch.einsum("... r t , ... t -> ... r", R_inv, t)
    return R_inv, t_inv


def node2pair(t1: Tensor, t2: Tensor, sequence_dim: int, op: Callable[[Tensor, Tensor],
                                                                      Tensor]) -> Tensor:
    """
    Create a pair tensor from a single tensor

    Args:
        t1: The first tensor to be converted to pair tensor
```

```
        t2: The second tensor to be converted to pair tensor
        sequence_dim: The dimension of the sequence
        op: The operation to be applied to the pair tensor

    Returns:
        Tensor: The pair tensor

    """
    # convert to positive if necessary
    if sequence_dim < 0:
        sequence_dim = t1.ndim + sequence_dim
    if t1.ndim != t2.ndim:
        raise ValueError(f"t1 and t2 must have the same number of dimensions, got {t1.ndim
                                                                    } and {t2.ndim}")
    t1 = t1.unsqueeze(sequence_dim + 1)
    t2 = t2.unsqueeze(sequence_dim)
    return op(t1, t2)


def compose_rotation_and_translation(
    R1: Tensor,
    t1: Tensor,
    R2: Tensor,
    t2: Tensor,
) -> tuple[Tensor, Tensor]:
    """Compose two frame updates.

    Ref AlphaFold2 Suppl 1.8 for details.

    Args:
        R1: Rotation of the first frames, (..., 3, 3).
        t1: Translation of the first frames, (..., 3).
        R2: Rotation of the second frames, (..., 3, 3).
        t2: Translation of the second frames, (..., 3).

    Returns:
        A tuple of new rotation and translation, (R_new, t_new).
        R_new: R1R2, (..., 3, 3).
        t_new: R1t2 + t1, (..., 3).
    """
    R_new = einops.einsum(R1, R2, "... r1 r2, ... r2 r3 -> ... r1 r3")  # (..., 3, 3)
    t_new = (
        einops.einsum(
            R1,
            t2,
            "... r t, ... t->... r",
        )
        + t1
    )  # (..., 3)

    return R_new, t_new


def masked_quadratic_mean(
    value: Tensor,
    mask: Tensor,
    dim: int | tuple[int, ...] | list[int] = -1,
    eps: float = 1e-10,
) -> Tensor | tuple[Tensor, Tensor]:
    """Compute quadratic mean value for tensor with mask.

    Args:
        value: Tensor to compute quadratic mean.
        mask: Mask of value, the same shape as `value`.
        dim: Dimension along which to compute quadratic mean.
```

```
        eps: Small number for numerical safety.
        return_masked: Whether to return masked value.

    Returns:
        Masked quadratic mean of `value`.
        [Optional] Masked value, the same shape as `value`.
    """
    return torch.sqrt((value * mask).sum(dim) / (mask.sum(dim) + eps))


def frame_aligned_frame_error_loss(
    R_pred: Tensor,
    t_pred: Tensor,
    R_gt: Tensor,
    t_gt: Tensor,
    frame_mask: Tensor,
    rotate_scale: float = 1.0,
    axis_scale: float = 20.0,
    eps_so3: float = 1e-7,
    eps_r3: float = 1e-4,
    dist_clamp: float | None = None,
    pair_mask: Tensor | None = None,
):
    """Compute frame aligned frame error loss with double geodesic metric.

    Args:
        R_pred: Predicted rotation matrices of frames, (..., N, 3, 3).
        t_pred: Predicted translations of frames, (..., N, 3).
        R_gt: Ground truth rotation matrices of frames, (..., N, 3, 3).
        t_gt: Ground truth translations of frames, (..., N, 3).
        frame_mask: Existing masks of ground truth frames, (..., N).
        axis_scale: Scale by which the R^3 part of loss is divided.
        eps_so3: Small number for numeric safety for arccos.
        eps_r3: Small number for numeric safety for sqrt.
        dist_clamp: Cutoff above which distance errors are disregarded.
        pair_mask: Additional pair masks of pairs which should be calculated, (..., N, M)
                                                    or None.
            pair_mask=True, the FAPE loss is calculated; vice not calculated.
            If None, all pairs are calculated.

    Returns:
        Dict of (B) FAFE losses. Contains "fafe", "fafe_so3", "fafe_r3".
    """
    N = R_pred.shape[-3]

    def _diff_frame(R: Tensor, t: Tensor) -> Tensor:
        R_inv, t_inv = invert_rigid(
            R=einops.repeat(R, "... i r1 r2 -> ... (i j) r1 r2", j=N),
            t=einops.repeat(t, "... i t -> ... (i j) t", j=N),
        )
        R_j = einops.repeat(R, "... j r1 r2 -> ... (i j) r1 r2", i=N)
        t_j = einops.repeat(t, "... j t -> ... (i j) t", i=N)

        return compose_rotation_and_translation(R_inv, t_inv, R_j, t_j)

    frame_mask = node2pair(frame_mask, frame_mask, -1, torch.logical_and)
    if pair_mask is not None:
        frame_mask = pair_mask * frame_mask
    frame_mask = einops.rearrange(frame_mask, "... i j -> ... (i j)")

    losses = compute_double_geodesic_error(
        *_diff_frame(R_pred, t_pred),
        *_diff_frame(R_gt, t_gt),
        frame_mask=frame_mask,
        rotate_scale=rotate_scale,
```

```
        axis_scale=axis_scale,
        dist_clamp=dist_clamp,
        eps_so3=eps_so3,
        eps_r3=eps_r3,
    )
    return losses


def compute_double_geodesic_error(
    R_pred: Tensor,
    t_pred: Tensor,
    R_gt: Tensor,
    t_gt: Tensor,
    frame_mask: Tensor,
    rotate_scale: float = 1.0,
    axis_scale: float = 20.0,
    dist_clamp: float | None = None,
    eps_so3: float = 1e-7,
    eps_r3: float = 1e-4,
):
    """Compute frame-wise error with double geodesic metric.

    d_se3(T_pred, T_gt) = sqrt(d_so3(R_pred, R_gt)^2 + (d_r3(t_pred, t_gt) / axis_scale)^2
                                                    )
    d_so3(R_pred, R_gt) range [0, pi]
    d_r3(t_pred, t_gt) / axis_scale) range [0, 1.5] when clamping

    Args:
        R_pred: Predicted rotation matrices of T, (..., N, 3, 3).
        t_pred: Predicted translations of T, (..., N, 3).
        R_gt: Ground truth rotation matrices of T, (..., N, 3, 3).
        t_gt: Ground truth translations of T, (..., N, 3).
        frame_mask: Existing masks of ground truth T, (..., N).
        rotate_scale: Scale by which the SO3 part of loss is divided.
        axis_scale: Scale by which the R^3 part of loss is divided.
        dist_clamp: Cutoff above which distance errors are disregarded.
        eps_so3: Small number for numeric safety for arccos.
            Refer to https://github.com/pytorch/pytorch/issues/8069
        ep3_r3: Small number for numeric safety for sqrt.

    Returns:
        Dict of (B) FAFE losses. Contains "fafe", "fafe_so3", "fafe_r3".

    Note:
        so3 loss/error presented in scaled form [0, pi/rotate_scale].
        r3 loss/error presented in scaled form [0, dist_clamp/axis_scale].
    """
    if dist_clamp is None:
        dist_clamp = 1e9

    # SO3 loss
    R_diff = einops.rearrange(R_pred, "... i j -> ... j i") @ R_gt
    R_diff_trace = R_diff.diagonal(dim1=-2, dim2=-1).sum(-1)
    so3_dist = torch.acos(torch.clamp((R_diff_trace - 1) / 2, -1 + eps_so3, 1 - eps_so3))
                                                    / rotate_scale  # (..., N)
    so3_loss = masked_quadratic_mean(so3_dist, frame_mask, dim=(-1))

    # R3 loss
    r3_dist = torch.sqrt(torch.sum((t_pred - t_gt) ** 2, dim=-1) + eps_r3)  # (..., N)
    r3_dist = r3_dist.clamp(max=dist_clamp) / axis_scale  # (..., N)
    r3_loss = masked_quadratic_mean(r3_dist, frame_mask, dim=(-1))

    # double geodesic loss
    se3_dist = torch.sqrt(so3_dist**2 + r3_dist**2)  # (..., N)
    se3_loss = masked_quadratic_mean(se3_dist, frame_mask, dim=(-1))
```

```
    losses = {
        "fafe": se3_loss,  # Note se3_loss = sqrt((so3_loss/rotate_scale)^2 + (r3_loss/
                                                axis_scale)^2)
        "fafe_so3": so3_loss,
        "fafe_r3": r3_loss,
    }

    return losses
```

C. Experiment details

C.1. Data preparation details

We crop the detected heavy/light chain and protein-type antigen from the full pdb structure. MSA is searched against these cropped structures. Next, we use ANARCI (Dunbar & Deane, 2016) to annotate the loop region on VH/VL chains and further crop the antibody along with its MSA. We also notice there is a number of antigens with a very long sequence but only a limited region of ground-truth structures, so we crop the antigen to the longest sub-sequence that includes all the residues with experimental ground truth.

After cropping, multiple filters are applied to clean data. SabDab entries without antigens or with non-protein antigens are discarded. All the chains are renamed according to the annotation by ANARCI, and samples with multiple H chains or multiple L chains detected are also discarded. There are also two special types of entries, nanobodies and Single-chain variable fragments (scFv). The nanobody is commonly annotated as a H-chain-only antibody, and scFvs are treated as two chains.

For both evaluation sets, targets with small molecules, peptides or nucleic acids near the docking interface are removed. For low homology evaluation set, We further restrict to a subset of all the structures after January 12th, 2023, and manually check them in PDB website. Invalid targets like those with only partial antigen structure are removed. For full evaluation set, the total number is too large for manual checking, so we only check the cluster centers and find 12 invalid samples, 8 of which are either partial complex from a multiple-antigen or multiple-antibody assembly in which non-trivial interactions exist, 4 of which include unfolded chains in the experimental structure. Finally, for both evaluation sets, targets with full-complex sequence length larger than 1500 after cropping are removed for evaluation speed considerations.

C.2. Training details

For LoRA experiments, we apply LoRA only on 48-layer evoformer. All the parameters used before the evoformer are frozen. Structure module and all the prediction heads are trainable. In one version of our code we freeze the linear layer from node representations to single representations, which seems to harm the performance.

As the original paper (Hu et al., 2022) suggests, we set LoRA α the same value as LoRA, and make a hyper parameter search for LoRA rank in [1, 4, 16, 32], and we found setting to 16 outperforms other values. The LoRA is added to all the linear layers in the evoformer, and for Q, K, V linear in attention, we add separate LoRA weights for each linear, despite their being fused during forwarding.

The training loss is set basically the same as original AF2. The batch size is set to 32, which we find is the minimal batch size to improve AF2 performance during fine-tuning. Training with larger batch size may improve performance, but we did not investigate it due to computing resource constraints. The cropping strategy is slightly different to original AF2 setting that during spatial cropping, only interface between antibody and antigen is considered. The crop length is set to 384 at max. The number of MSA cluster center is restricted to 128 for training efficiency. Cross-chain uniprot MSAs and templates are not provided, since we want to simplify our protocol and in theory antibody-antigen docking does not rely on these information. Finally, we use early stopping to avoid overfitting on training samples, which is a severe problem we have encountered in the early rounds of experiments.

C.3. Inference details

During the inference, the random seed is set to 0 which will be applied to all data transformations. This means for all the model the MSA clustering results for the same recycle in different models will be controlled as same. We think this is a

more reasonable settings than randomly sampling MSAs for different models, because we observe the sampling procedure can introduce large bias, sometimes making a sample rotates to correct poses from wrong ones for the same model.

D. Additional results

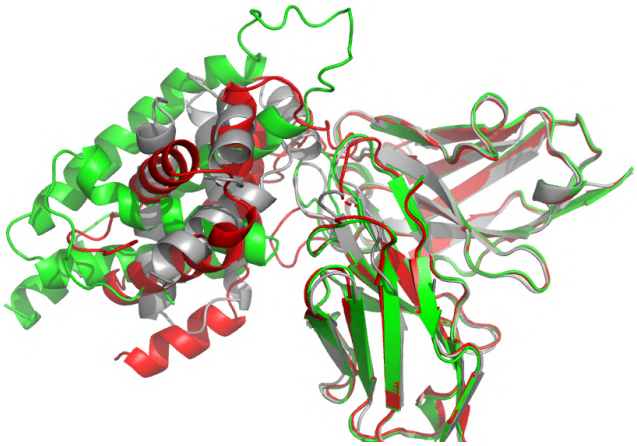

Figure 5. The predicted structures of PDB 8DB4 against ground truth. The grey structure is the experimental ground truth, the green structure by AF2.3 and the red structure by model tuned with F2E.

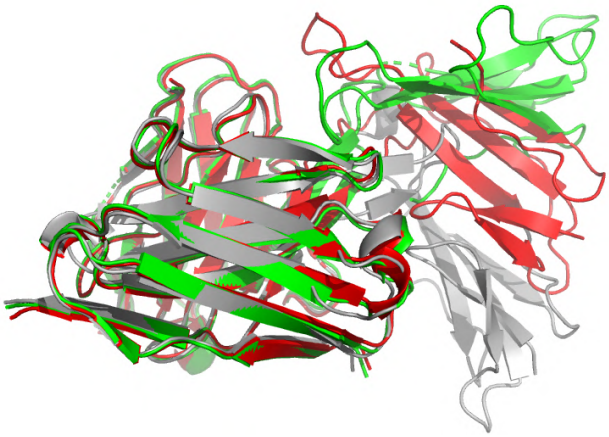

Figure 6. The predicted structures of PDB 8D9Y against ground truth. The grey structure is the experimental ground truth, the green structure by AF2.3 and the red structure by model tuned with F2E.

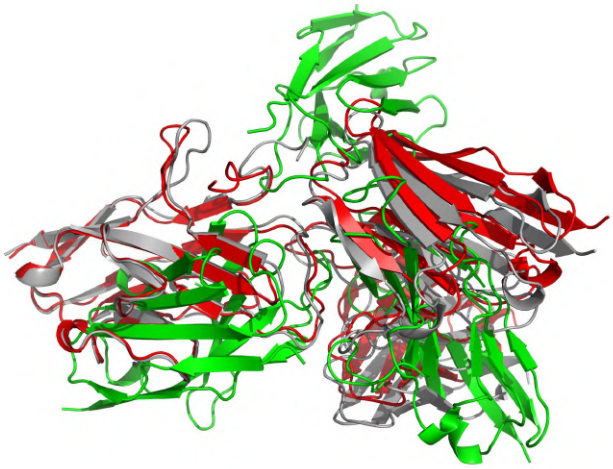

Figure 7. The predicted structures of PDB 7QUH against ground truth. The grey structure is the experimental ground truth, the green structure by AF2.3 and the red structure by model tuned with F2E.

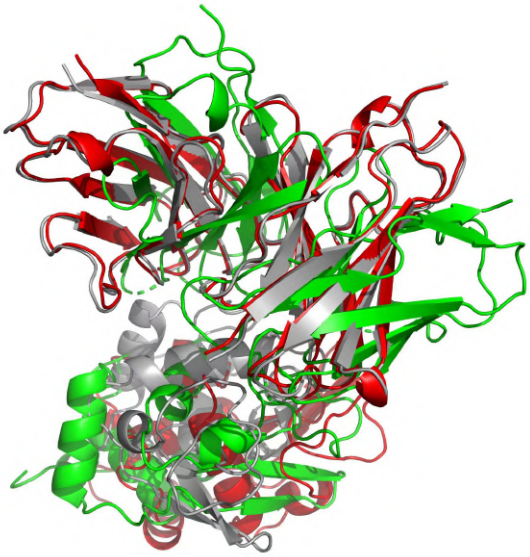

Figure 8. The predicted structures of PDB 8GQ1 against ground truth. The grey structure is the experimental ground truth, the green structure by AF2.3 and the red structure by model tuned with F2E.