Computational Design and Cell-Free Expression of Coat Protein Subunits for the Functional Assembly into Potato Virus X-like Particles

Thesis
in Molecular and Applied Biotechnology (B. Sc.)
at RWTH Aachen University

Submitted on: June 23, 2025
by: Kilian Mandon
1. Appraiser: Prof. Dr. Stefan Schillberg
2. Appraiser: Jun. Prof. Dr. Anna Matuszyńska

RWTH Aachen University

Contents

1 Introduction

2 Notation

3 Symmetry Analysis and Model Building

The structure of PVX was determined by [2], up to a resolution of 2.2 Å. The structural data was made available through the PDB, as a file containing 13 consecutive protein subunits, forming one-and-a-half cycles of the helix.

The following chapters require a flexible way to use this symmetry, such as the ability to generate different configurations of monomers (e.g. a 3×3 neighborhood of monomers), or the ability to dynamically enforce this symmetry during symmetry-guided prediction with AlphaFold (Section ...) or symmetry-guided design with RFdiffusion (Section ...). Therefore, this section discusses the computation of the symmetry relationship between consecutive monomers, and how it can be applied to generate new configurations of monomers.

Let $\{\vec{\mathbf{r}}_{j,i}^{\text{original}}\}$ denote the backbone atom positions of chain j in the original PDB file, and let $\{\vec{\mathbf{r}}_{j}^{\text{original}}\}$ be their arithmetic mean.

We choose $T_0 = (I_3, \vec{\mathbf{r}}_A^{\text{original}})$ as our new origin, centered on chain A. The backbone atom coordinates in this frame are denoted by $\vec{\mathbf{r}}_{j,i}$, and we have

$$\vec{\mathbf{r}}_{j,i} = T_0^{-1} \circ \vec{\mathbf{r}}_{j,i}^{\text{original}} = \vec{\mathbf{r}}_{j,i}^{\text{original}} - \vec{\mathbf{r}}_A^{\text{original}} \tag{1}$$

The frames of all other chains in these coordinates are computed as the optimal rigid body transform to align the chain with A. That is,

$$T_j = \underset{T \in \text{SE}(3)}{\arg\min} \sum_i \|T \circ \vec{\mathbf{r}}_{A,i} - \vec{\mathbf{r}}_{j,i}\|^2 \tag{2}$$

Using the Kabsch algorithm [3], T_j can be computed as $T_j = (R_j, \vec{\mathbf{t}}_j)$, where

$$\vec{\mathbf{t}}_j = \vec{\mathbf{r}}_j - R_j \vec{\mathbf{r}}_A = \vec{\mathbf{r}}_j \tag{3}$$

since $\vec{\mathbf{r}}_A = \vec{\mathbf{0}}$, and $R_j \in \text{SO}(3)$ minimizes

$$\sum_i \|R_j(\vec{\mathbf{r}}_{A,i} - \vec{\mathbf{r}}_A) - (\vec{\mathbf{r}}_{j,i} - \vec{\mathbf{r}}_j)\| \tag{4}$$

Following the Kabsch Algorithm, R_j can be computed via the singular value decomposition

$$(\vec{\mathbf{r}}_{A,i} - \vec{\mathbf{r}}_A)^T \cdot (\vec{\mathbf{r}}_{j,i} - \vec{\mathbf{r}}_j) = U \Sigma V^T \tag{5}$$

as

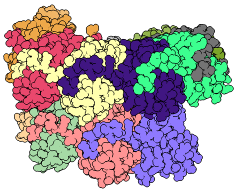
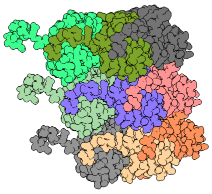
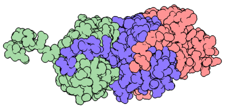
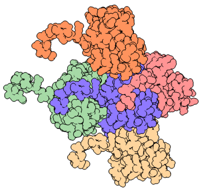
$$R_j = V \cdot \text{diag}(1, 1, d) \cdot U^T \tag{6}$$

where $d = \det(U)\det(V)$ corrects for a potential reflection in the orthogonal matrices U and V.

With all frames T_j expressed in the same coordinate system, we can compute the relative transform

$$T_{j \to j+1} = (R_{j \to j+1}, \vec{\mathbf{t}}_{j \to j+1}) = T_j^{-1} \circ T_{j+1} \tag{7}$$

Table 1: Visualization and chain indices of different monomer configurations, generated based on the average relative transform T_R. The blue chain has index 0, the coordinates for the other chains are computed as $T_R^j \circ \vec{\mathbf{r}}_{A,i}, j \in I$. The generated monomer configurations will be used to create inputs for the algorithms in the following sections.

Type	Indices	Visualization
Helical	$I = \{0, ..., 12\}$	
3x3	$I = \{0, \pm 1, \pm 8, \pm 9, \pm 10\}$	
Trimer	$I = \{0, \pm 1\}$	
Pentamer	$I = \{0, \pm 1, \pm 9\}$	

Given the symmetry of the viral coat structure, these transforms are expected to be equal. The average relative transform $T_R = (R_R, \vec{\mathbf{t}}_R)$ is computed by choosing $\vec{\mathbf{t}}_R$ as the mean over $\{\vec{\mathbf{t}}_{j\to j+1}\}$ and choosing $R_R \in \mathrm{SO}(3)$ as the rotation matrix closest to the average over all $R_{j\to j+1}$, that is $R_R = UV^T$ where $U\Sigma V^T = \frac{1}{n}\sum_j R_{j\to j+1}$ [5] (given the similarity of the $\{R_{j\to j+1}\}$, no reflection can arise by continuity).

The individual rotations $R_{j\to j+1}$ had standard deviation $\Delta R_R = 0.004\,\mathrm{rad}$ in geodesic distance, and the individual translations had standard deviation $\Delta \mathbf{t}_R = 0.04\,\text{Å}$. R_R closely resembles a pure rotation around the z-axis $R_Z(\theta)$, with an angle of $\theta = -0.707\,\mathrm{rad}$. The deviation is $d(R_R, R_Z(\theta)) = 0.005\,\mathrm{rad}$. This value of θ corresponds to a left-handed helix with 8.89 subunits per turn. The computed rise is $\mathbf{t}_z = 3.87\,\text{Å}$ per subunit, resulting in a helical pitch (rise per turn) of 34.4 Å. These values are mostly consistent with the ones stated in [2] (rise 3.96 Å, rotation of 0.707 rad, 8.9 copies per turn, helical pitch 35.2 Å). However, the authors emphasize the slight difference in the helical pitch of 35.2 Å compared to that of similar flexible filamentous plant viruses (PepMV, BaMV, and PapMV), for which the helical pitch ranges from 34.3 Å to 34.6 Å. According to the calculations above, the helical pitch in the PDB entry (which the authors produced through multiple cycles of real space refinement) differs from the original helical parameters fitted to the cryo-EM data and falls into the range of the other plant viruses, thereby potentially diminishing the significance of the reported pitch deviation.

Given the relative transform T_R, model coordinates can be reconstructed based on the coordinates of the monomer A according to

$$\vec{\mathbf{r}}_{j,i}^{\text{original}} = T_0 \circ T_R^j \circ \vec{\mathbf{r}}_{A,i} \tag{8}$$

Using equation 8, four different configurations of monomers are generated and used throughout the following sections (Table 1). A helical configuration consisting of thirteen consecutive monomers, a three-by-three neighborhood of nine monomers, a trimer consisting of three consecutive monomers, and a pentamer consisting of five monomers aranged in a cross-shape.

Despite the small standard deviation of T_R, the deviation of individual atom positions in the helical thirteen-monomer reconstruction compared to the data from the pdb entry reaches up to 0.8 Å. This is due to lever effects caused by small deviations in the rotation. The difference in structure introduces no new clashes, but slightly reduces the contacts by 2 %, as computed with ChimeraX [4].

4 Sequence Design with ProteinMPNN

ProteinMPNN [1] is a deep learning model for protein sequence design, capable of creating de-novo designs of proteins that fold into a desired shape or bind to specific targets. The algorithm can create sequences for monomers, heterooligomers, and homooligomers.

The sequence is designed based on a protein backbone as input, that is the position of all backbone atoms of one or multiple chains. The underlying algorithm uses a Message Passing Neural Network (MPNN), a graph-based machine learning model. Each residue in the protein is encoded as a vertex in the graph, and edges are drawn up from each residue to its 48 closest neighbors. Vertex embeddings are initialized as 0 vectors, while the initial edge embeddings are computed based on the distances between the backbone atoms of the residue pair and the difference of their residue indices. After the computation of the initial feature embeddings, ProteinMPNN follows an encoder-decoder architecture, in which the encoder updates the edge and vertex embeddings based on their neighborhood, wheras the decoder uses the embeddings computed by the encoder to predict the amino acid type for each residue. The decoder works in an autoregressive fashion by choosing a random order for decoding the individual residues, then predicting their residue type one-by-one with knowledge of all already predicted residues. Concretely, the algorithm predicts logits $\{\ell_i\}$ for each amino acid and chooses it from a softmax distribution according to

$$P(a_i) = \frac{\exp\left(\frac{\ell_i}{\tau}\right)}{\sum_{j=1}^{20} \exp\left(\frac{\ell_j}{\tau}\right)}$$

Here, $\tau > 0$ denotes a chosen temperature constant in the softmax distribution. For $\tau \to \infty$, the distribution is almost uniform, while for $\tau \to 0$ the amino acid with the highest predicted logit is chosen. The distribution can be biased by adding to the logits before sampling. For homooligomers, the logits of identical residues in different monomers are averaged and only one amino acid is sampled from the distribution for all of them.

In this work, all sequences used in computational and experimental evaluation are generated using ProteinMPNN. The input structure is either chosen as the backbone

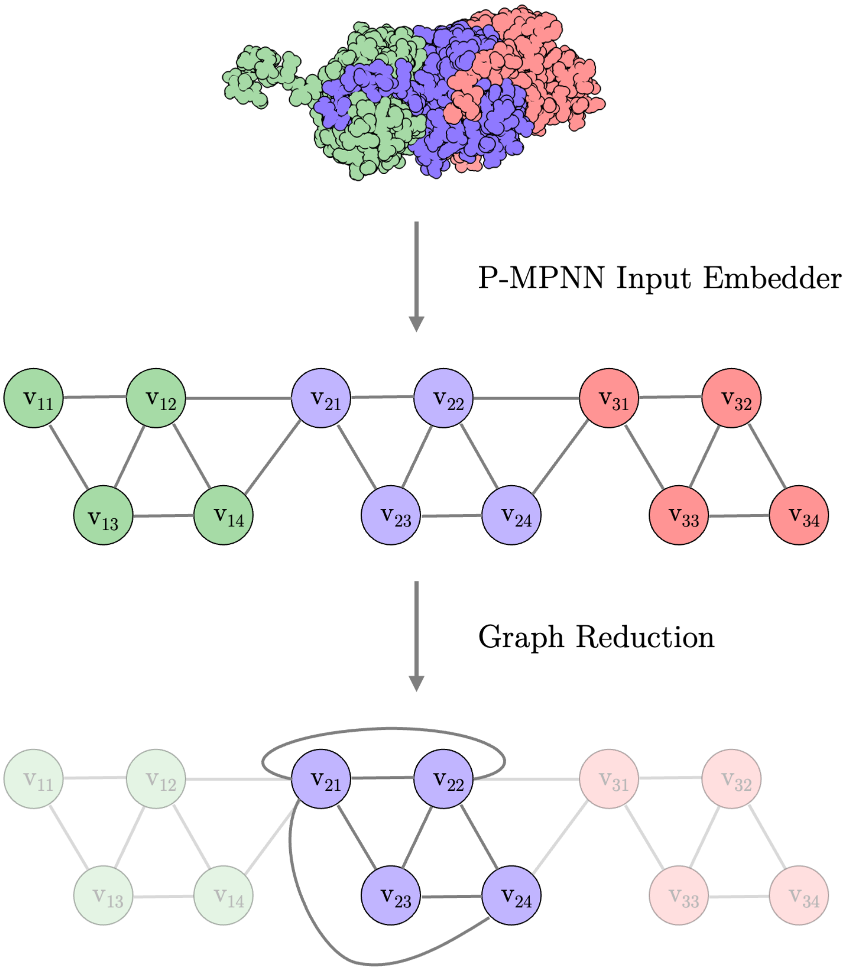


Figure 1: Graph Reduction procedure for symmetric homooligomers. After the default graph initialization from ProteinMPNN, one of the monomers is chosen as the reference monomer. Edges going out from it to other monomers are remapped to the corresponding residue in itself. Afterwards, vertices and edges of the non-reference monomers are discarded.

structure of the wildtype, thereby generating alternative sequences for the structure, or a generated artificial backbone as described in section 5. Of particular note is the choice of the input structure: The helical virus particle consists of approximately 1300 monomers [2], and truncation to a smaller number will lead to an incorrect neighborhood during featurization for newly exposed residues.

However, a modification to the original ProteinMPNN algorithm can circumvent this by allowing sequence prediction for a theoretical infinite extension of a symmetric homooligomer. In ProteinMPNN, feature initialization is solely dependent on the relative neighborhood of each residue, meaning that initialization is identical for all corresponding residues in a symmetric homooligomer. Further, the message-passing algorithm in the network conserves this equivariance. Therefore, a theoretical infinite extension of the homooligomer can be simulated by remapping of interchain edges to the corresponding residue in the same chain (Figure 1), thereby reducing the input to a single monomer.

Testing this new algorithm for different helical viruses, the Graph Reduction

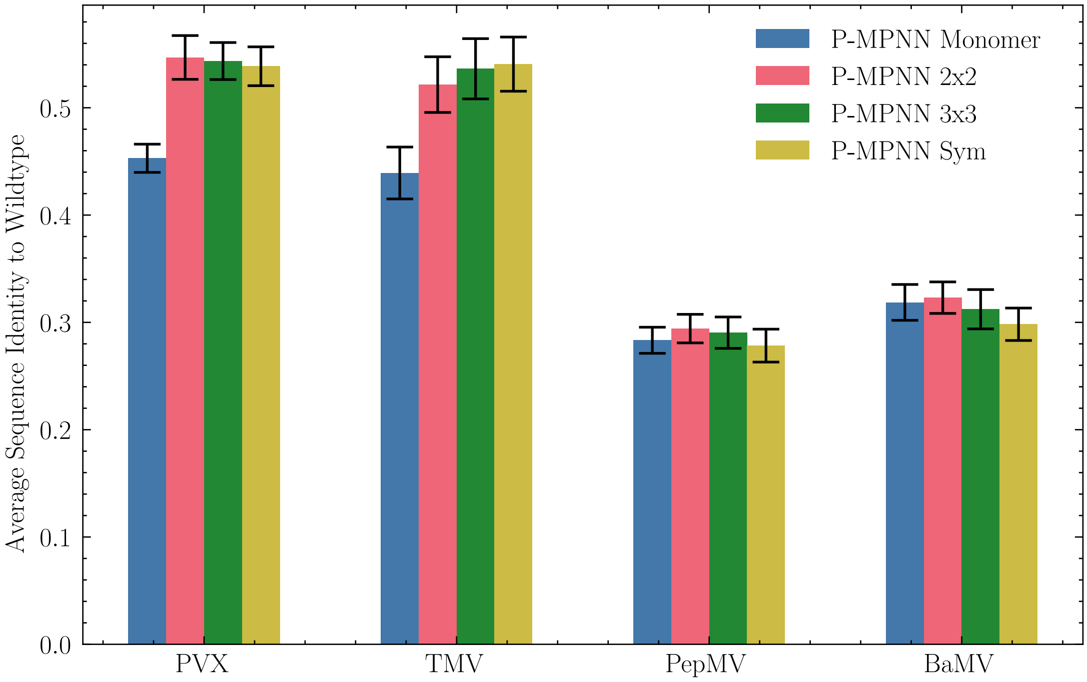


Figure 2: Sequence recovery by ProteinMPNN for different input configurations. The input was chosen as either a single monomer, a 2x2 neighborhood, a 3x3 neighborhood, or a symmetry-preserving graph reduction, modeling a theoretical infinite neighborhood. For each of the four targets Potato Virus X (PVX), Tobacco Mosaic Virus (TMV), Pepino Mosaic Virus (PepMV) and Bamboo Mosaic Virus (BaMV), each model was evaluated 50 times using random decoding orders and a sampling temperature $\tau \to 0$. The errorbars indicate the standard deviation over the repeated evaluation.

procedure showed no significant improvement compared to prediction based on a 2x2 neighborhood or a 3x3 neighborhood (Figure 2)

5 Backbone Design with RFdiffusion

6 Evaluation with AlphaFold

7 Evaluation with GROMACS

Algorithm 1 Sample Diffusion

```
def SampleDiffusion($\{\mathbf{f}^*\}$, $\{\mathbf{s}_i^{\text{inputs}}\}$, $\{\mathbf{s}_i^{\text{trunk}}\}$, $\{\mathbf{z}_{ij}^{\text{trunk}}\}$, Noise Schedule
                    $[c_0, c_1, \ldots, c_T]$, $\gamma_0 = 0.8$, $\gamma_{\text{min}} = 1.0$, noise scale
                    $\lambda = 1.003$, step scale $\eta = 1.5$):
    $\vec{\mathbf{x}}_l \sim c_0 \cdot \mathcal{N}(\vec{\mathbf{0}}, \mathbf{I}_3)$                                        $\vec{\mathbf{x}}_l \in \mathbb{R}^3$
    for all $c_\tau \in [c_1, \ldots, c_T]$ do
        $\{\vec{\mathbf{x}}_l\} \leftarrow$ CentreRandomAugmentation($\{\vec{\mathbf{x}}_l\}$)
        $\gamma \leftarrow \gamma_0$ if $c_\tau > \gamma_{\text{min}}$ else 0
        $\hat{t} \leftarrow c_{\tau-1}(\gamma + 1)$
        $\vec{\boldsymbol{\xi}}_l \leftarrow \lambda\sqrt{\hat{t}^2 - c_{\tau-1}^2} \cdot \mathcal{N}(\vec{\mathbf{0}}, \mathbf{I}_3)$                     $\vec{\boldsymbol{\xi}}_l \in \mathbb{R}^3$
        $\vec{\mathbf{x}}_l^{\text{noisy}} \leftarrow \vec{\mathbf{x}}_l + \vec{\boldsymbol{\xi}}_l$
        $\{\vec{\mathbf{x}}_l^{\text{denoised}}\} \leftarrow$ DiffusionModule($\{\vec{\mathbf{x}}_l^{\text{noisy}}\}, \hat{t}, \{\mathbf{f}^*\}, \{\mathbf{s}_i^{\text{inputs}}\}, \{\mathbf{s}_i^{\text{trunk}}\}, \{\mathbf{z}_{ij}^{\text{trunk}}\}$)
        $\vec{\boldsymbol{\delta}}_l \leftarrow (\vec{\mathbf{x}}_l - \vec{\mathbf{x}}_l^{\text{denoised}})/\hat{t}$
        $dt \leftarrow c_\tau - \hat{t}$
        $\vec{\mathbf{x}}_l \leftarrow \vec{\mathbf{x}}_l^{\text{noisy}} + \eta \cdot dt \cdot \vec{\boldsymbol{\delta}}_l$
    end for
    return $\{\vec{\mathbf{x}}_l\}$
```

Algorithm 2 Sample Diffusion with Symmetrization for Multimeric Complexes

```
def SampleDiffusion($\{\mathbf{f}^*\}$, $\{\mathbf{s}_i^{\text{inputs}}\}$, $\{\mathbf{s}_i^{\text{trunk}}\}$, $\{\mathbf{z}_{ij}^{\text{trunk}}\}$, Noise Schedule
                   $[c_0, c_1, \ldots, c_T]$, $\gamma_0 = 0.8$,
                   $\gamma_{\min} = 1.0$, noise scale $\lambda = 1.003$, step scale $\eta = 1.5$,
                   Monomer Transforms $\{T_j\}$, Monomer Indices $\{I_j\}$ ):
    $\vec{\mathbf{x}}_l \sim c_0 \cdot \mathcal{N}(\vec{\mathbf{0}}, \mathbf{I}_3)$                                   $\vec{\mathbf{x}}_l \in \mathbb{R}^3$
#   Modification: Initial Symmetrization
★:  $(R_{\text{ref}}, \vec{\mathbf{t}}_{\text{ref}}) \leftarrow (\mathbf{I}, \text{mean}(\{\vec{\mathbf{x}}_l\}_{l \in I_1}))$          Denoted as $T_{\text{ref}} = (R_{\text{ref}}, \vec{\mathbf{t}}_{\text{ref}})$
★:  for all $j \in [2, \ldots, N_{\text{monomer}}]$
★:      $\{\vec{\mathbf{x}}_l\}_{l \in I_j} \leftarrow T_{\text{ref}} \circ T_j \circ T_{\text{ref}}^{-1} \circ \{\vec{\mathbf{x}}_l\}_{l \in I_1}$
★:  end for
    for all $c_\tau \in [c_1, \ldots, c_T]$ do
#       Track Origin of Symmetrization Center
★:      $\vec{\mathbf{t}}_{\text{ref}} \leftarrow \text{mean}(\{\vec{\mathbf{x}}_l\}_{l \in I_1})$
        $\{\vec{\mathbf{x}}_l\}$ , $T_{\text{aug}}$ $\leftarrow$ CentreRandomAugmentation($\{\vec{\mathbf{x}}_l\}$)
#       Track Movement by CentreRandomAugmentation
★:      $T_{\text{ref}} \leftarrow T_{\text{aug}} \circ T_{\text{ref}}$
        $\gamma \leftarrow \gamma_0$ if $c_\tau > \gamma_{\min}$ else 0
        $\hat{t} \leftarrow c_{\tau-1}(\gamma + 1)$
        $\vec{\boldsymbol{\xi}}_l \leftarrow \lambda \sqrt{\hat{t}^2 - c_{\tau-1}^2} \cdot \mathcal{N}(\vec{\mathbf{0}}, \mathbf{I}_3)$                      $\vec{\boldsymbol{\xi}}_l \in \mathbb{R}^3$
        $\vec{\mathbf{x}}_l^{\text{noisy}} \leftarrow \vec{\mathbf{x}}_l + \vec{\boldsymbol{\xi}}_l$
        $\{\vec{\mathbf{x}}_l^{\text{denoised}}\} \leftarrow$ DiffusionModule($\{\vec{\mathbf{x}}_l^{\text{noisy}}\}, \hat{t}, \{\mathbf{f}^*\}, \{\mathbf{s}_i^{\text{inputs}}\}, \{\mathbf{s}_i^{\text{trunk}}\}, \{\mathbf{z}_{ij}^{\text{trunk}}\}$)
#       Recenter and Symmetrize Denoised Prediction
★:      $\vec{\mathbf{x}}_l^{\text{denoised}} \mathrel{+}= \text{mean}(\{\vec{\mathbf{x}}_l^{\text{noisy}}\}_{l \in I_1}) - \text{mean}(\{\vec{\mathbf{x}}_l^{\text{denoised}}\}_{l \in I_1})$
★:      for all $j \in [2, \ldots, N_{\text{monomer}}]$
★:          $\{\vec{\mathbf{x}}_l^{\text{denoised}}\}_{l \in I_j} \leftarrow T_{\text{ref}} \circ T_j \circ T_{\text{ref}}^{-1} \circ \{\vec{\mathbf{x}}_l^{\text{denoised}}\}_{l \in I_1}$
★:      end for
        $\vec{\boldsymbol{\delta}}_l \leftarrow (\vec{\mathbf{x}}_l - \vec{\mathbf{x}}_l^{\text{denoised}})/\hat{t}$
        $dt \leftarrow c_\tau - \hat{t}$
        $\vec{\mathbf{x}}_l \leftarrow \vec{\mathbf{x}}_l^{\text{noisy}} + \eta \cdot dt \cdot \vec{\boldsymbol{\delta}}_l$
    end for
    return $\{\vec{\mathbf{x}}_l\}$
```

References

[1] J. Dauparas et al. "Robust deep learning–based protein sequence design using ProteinMPNN". In: *Science* 378.6615 (2022), pp. 49–56. DOI: 10.1126/science.add2187. eprint: https://www.science.org/doi/pdf/10.1126/science.add2187. URL: https://www.science.org/doi/abs/10.1126/science.add2187.

[2] Alessandro Grinzato et al. "Atomic structure of potato virus X, the prototype of the Alphaflexiviridae family". In: *Nature Chemical Biology* 16.5 (2020), pp. 564–569. ISSN: 1552-4469. DOI: 10.1038/s41589-020-0502-4. URL: https://doi.org/10.1038/s41589-020-0502-4.

[3] Jim Lawrence, Javier Bernal, and Christoph Witzgall. "A Purely Algebraic Justification of the Kabsch-Umeyama Algorithm". In: *Journal of Research of the National Institute of Standards and Technology* 124 (Oct. 2019). ISSN: 2165-7254. DOI: 10.6028/jres.124.028. URL: http://dx.doi.org/10.6028/jres.124.028.

[4] Elaine C. Meng et al. "UCSF ChimeraX: Tools for structure building and analysis". In: *Protein Science* 32.11 (2023), e4792. DOI: https://doi.org/10.1002/pro.4792. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/pro.4792. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/pro.4792.

[5] Soheil Sarabandi and Federico Thomas. "Solution methods to the nearest rotation matrix problem in : A comparative survey". In: *Numerical Linear Algebra with Applications* 30.5 (2023), e2492. DOI: https://doi.org/10.1002/nla.2492. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/nla.2492. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/nla.2492.