3DFlex: determining structure and motion of flexible proteins from cryo-EM

Ali Punjani, David J. Fleet, Nature Methods 2022

11/09/2023 — Ambri Ma, Minkyu Jeon COS597N presentation

Contents

Intro

- Motion-based neural network model for continuous molecular heterogeneity for cryo-EM data
- 3DFlex exploits conformational variability of a protein which is the result of physical processes
- From 2D image data, 3DFlex enables the determination of high-resolution 3D density, and provides an explicit model of a flexible protein’s motion over its conformational landscape
- Unlike local and multi-body refinement methods, 3DFlex exploits correlations between moving parts, making it possible to infer the position of all parts

Method

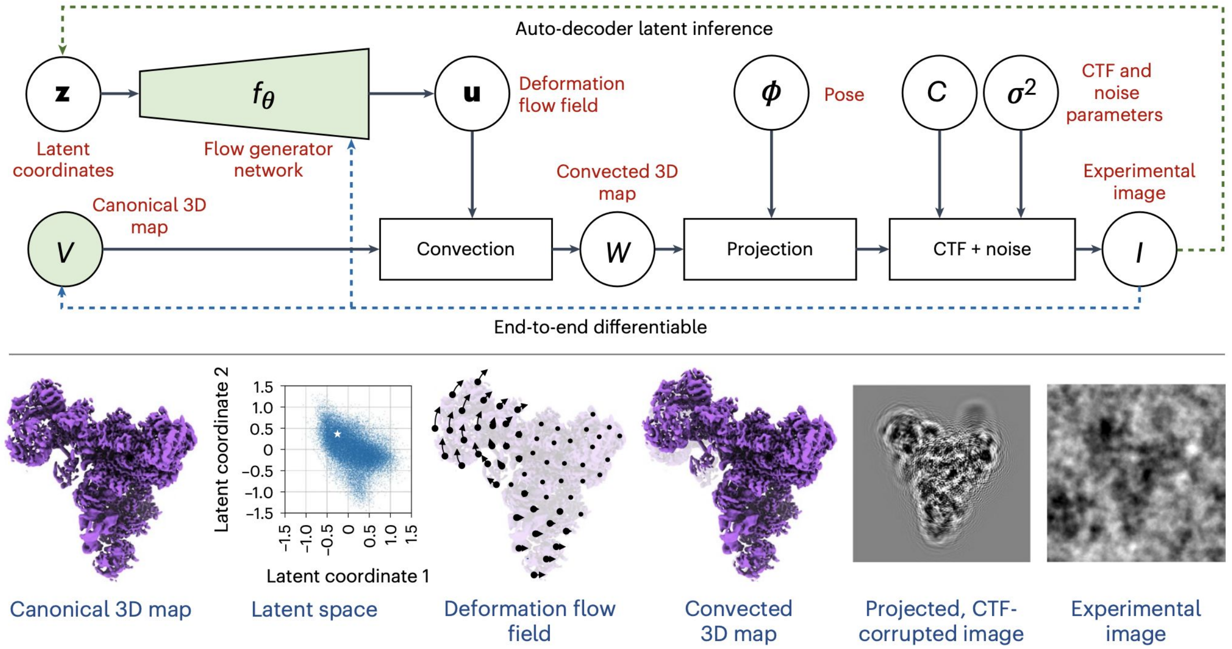
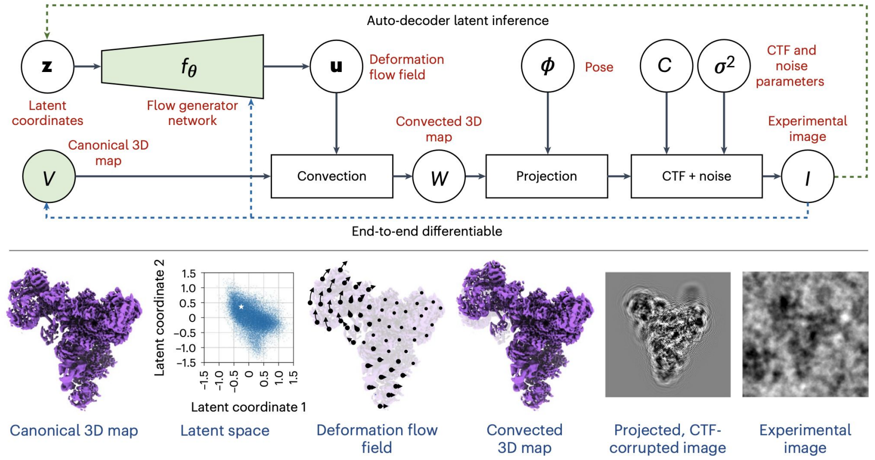


Fig. 1 | The 3DFlex model represents the flexible 3D structure of a protein as deformations of a single canonical 3D density map *V*. Under the model, a single-particle image is associated with low-dimensional latent coordinates that encode the conformation for the particle in the image. A neural flow generator network f_θ converts the latent coordinates into the flow field **u** and a convection operator then deforms the canonical density to generate a convected map *W*. This map can then be projected along the particle viewing direction determined by the pose ϕ, CTF corrupted and compared against the experimental image.

Method



- Flexible molecule is represented in terms of

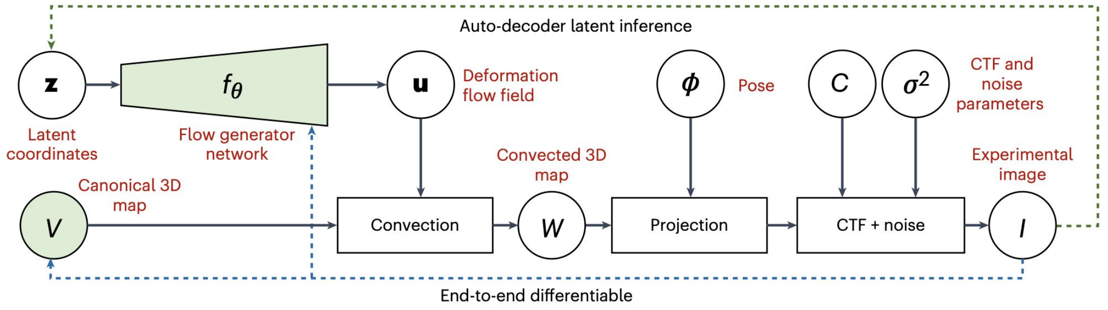
    1. A canonical density map

    2. Latent coordinate vectors that specify positions over the protein's conformational landscape

    3. A flow generator that converts a latent coordinate vector into a deformation field that convects the canonical map into the corresponding protein conformation

- Z, V, f_θ for each particle images are the model parameters — jointly optimized

Method

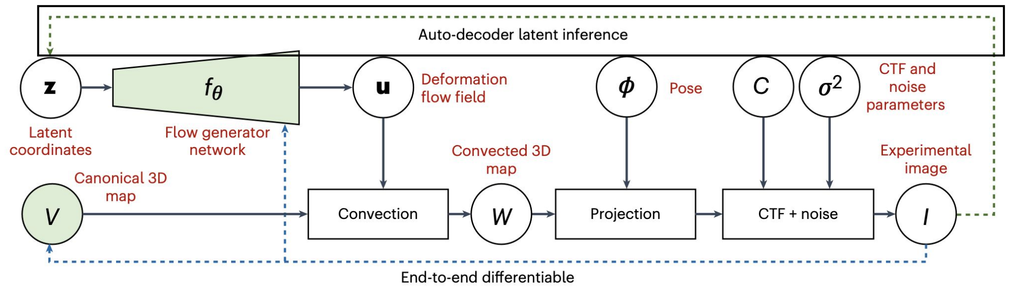


- $I_i = C_i P(\phi_i) W_i + \eta = C_i P(\phi_i) D(f_\theta(z_i), V) + \eta$ (2)
    - C_i: CTF operator
    - $P(\phi_i)$: projection operator for pose ϕ_i
    - $D(u_i, V)$: convection operator
        - $u_i = f_\theta(z_i)$

$$E_{\text{data}}(V, \theta, \mathbf{z}_{1:M}) = \frac{1}{2} \sum_{i=1}^{M} \left\| I_i - C_i P(\phi_i) D(f_\theta(\mathbf{z}_i), V) \right\|^2 \quad (3)$$

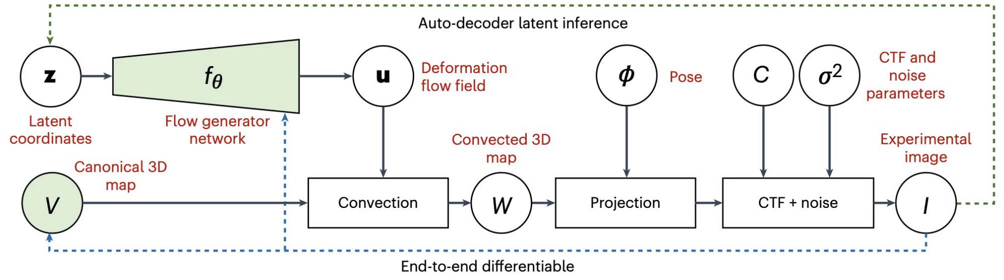
- M: # of particle images
- ϕ_i, CTF estimates are known

Method — Latent coordinates & Auto decoder



- The latent space in 3DFlex represents the conformational landscape
    - Different latent positions correspond to different deformations of the canonical map
- Perform inference by optimizing a point estimate of the latent coordinates independently for each image → more precise
- Compute gradients of the data likelihood w.r.t the latent coordinates for each image, and then use gradient optimization to perform inference

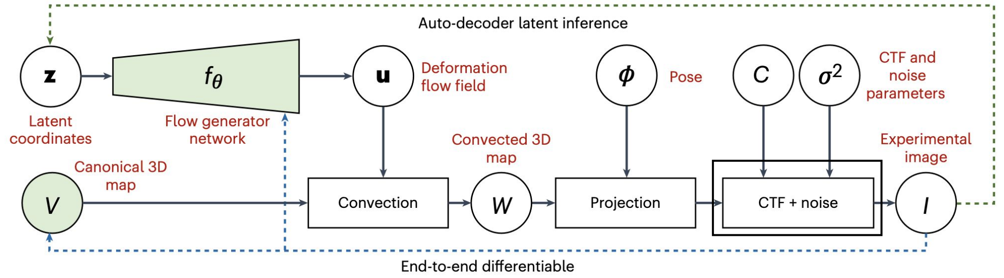
Method — Noise injection and prior on latents



- Directly adding noise to the point estimate during training can regularize the model and encourage smoothness of the latent space

- Use a Gaussian prior on latent coordinates with unit variance to help control the spread of the latent embeddings for different particles within a given dataset, and to center the distribution of latent embeddings at the origin in the latent space

Method — Real vs Fourier Space

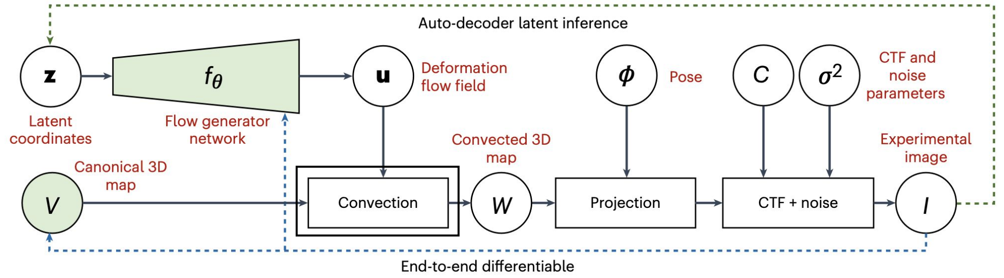


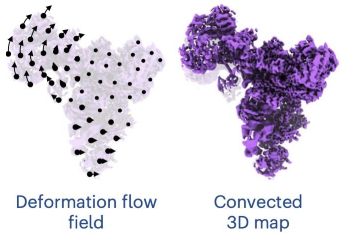
- Working in a Fourier domain reduces the computational cost of CTF modulation and image projection (via Fourier slice Theorem)
- However, the convection of density between conformations is more naturally expressed in real space

  → Fourier space is only for CTF + noise part

  - Represent V in real space — voxel array N^3

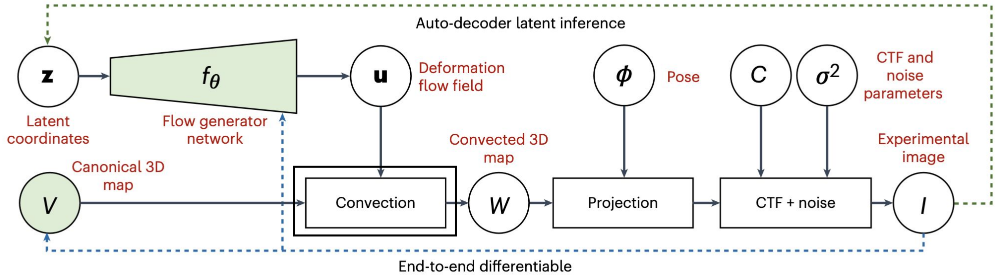
Method — Convection Operator

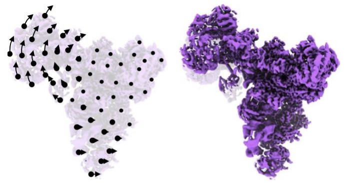




- Modeling the physical nature of protein motion, thereby allowing high-resolution structural detail
- One way to construct a convection operator is to express the flow field as a mapping from convected coordinates (voxel in W_i) to canonical coordinates
- Flow in 3DFlex u_i is a forward mapping from canonical coordinates in V to the deformed coordinates in W_i
    - Naturally conserves density, as every voxel in V has a destination in W_i, where it's contribution is accumulated through an interpolant function

Method — Convection Operator

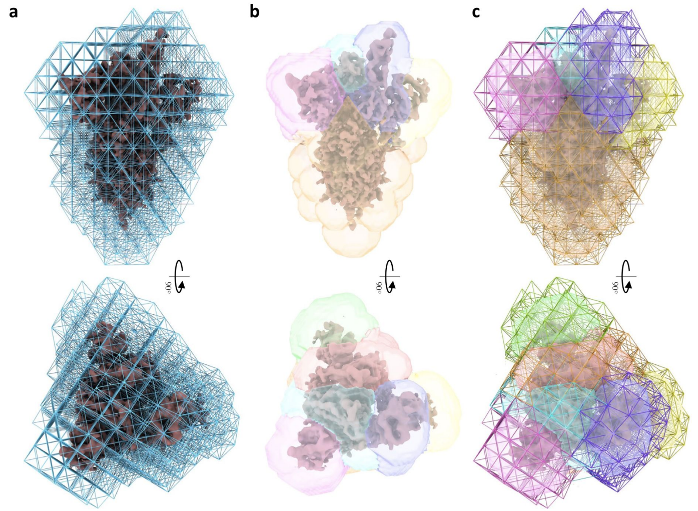




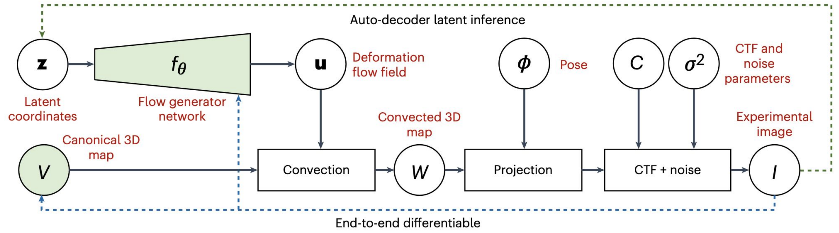
- Convected density at location x:
  - $W_i(x) = \Sigma_y k(x - u_i(y))V(y)$
    - Summation is over 3D spatial positions y of the canonical map
    - $u_i = f_\theta(z)$
    - k: interpolation kernel with finite support

Method — Regularization via tetrahedral mesh

- Regularization of deformations is critical
  - Without it, the deep generative model can easily overfit to noise in the data and learn unrealistic deformations
- The deformation field is parameterized by a 3D flow vector at each vertex of the tetrahedral mesh
- The deformation field is then interpolated using linear finite-element method shape functions within each mesh element
- Smoothness is a function of the size of mesh elements and is enforced implicitly through interpolation and the fact that ad elements can share vertices



Extended Data Fig. 1 | Mesh Topologies. Examples of mesh topology that can provided to 3DFlex for the SARS-CoV-2 spike protein (see Fig. 4). **a:** a regular mesh, with all mesh elements connected to their neighbors. This does not allow adjacent protein domains to easily separate or move in different directions. **b:** coarse separation of domains. **c:** an irregular mesh constructed by fusing sub-meshes for each domain at the interfaces where density is to be continuous. 3DFlex still learns motion at all mesh nodes jointly and from scratch, but is now able to model adjacent domains that move in different directions easily.

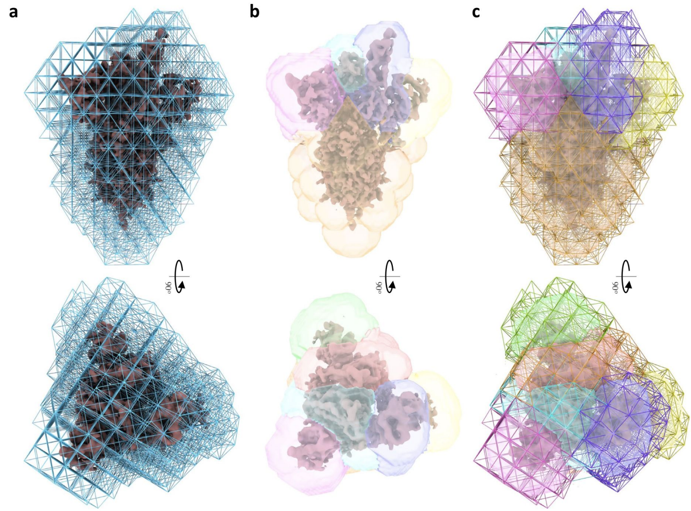


Method — Regularization via tetrahedral mesh

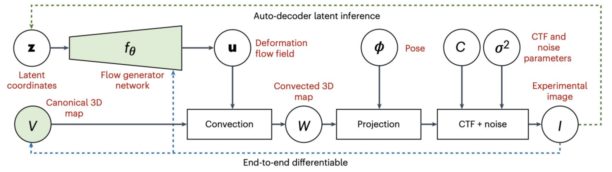
- The deformation field within the jth tetrahedral element for image i can be written as a linear mapping:
  - $u_{ij}(x) = A_{ij}x + b_{ij}$
    - Matrix A, vector b: uniquely determined from 3D flow vectors at the element vertices
- $E_{rigid} = \Sigma_i \Sigma_j w_j \Sigma_{\ell=1}^{3} (s_{ij}^{\ell} - 1)^2$: Local rigidity regularization loss
  - s_{ij}^{ℓ}: ℓth singular value of A_{ij}
  - w_{ij}: weights defining the strength of the prior within each mesh element, based on the density present within the jth mesh element
    - The densest elements have weight 1.0 / empty elements have a lower weight, by default 0.5
    - The weighting ensures that deformation fields are encouraged to compress and expand empty space around the protein
- $L = E_{data} + \lambda E_{rigid}$

$$E_{\text{data}}(V, \theta, \mathbf{z}_{1:M}) = \frac{1}{2} \sum_{i=1}^{M} \left\| I_i - C_i P(\phi_i) D\left(f_\theta(\mathbf{z}_i), V\right) \right\|^2$$

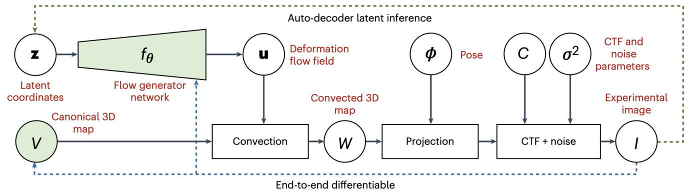
$$\begin{aligned} I_i &= C_i P(\phi_i) W_i + \eta \\ &= C_i P(\phi_i) D(f_\theta(\mathbf{z}_i), V) + \eta. \end{aligned}$$



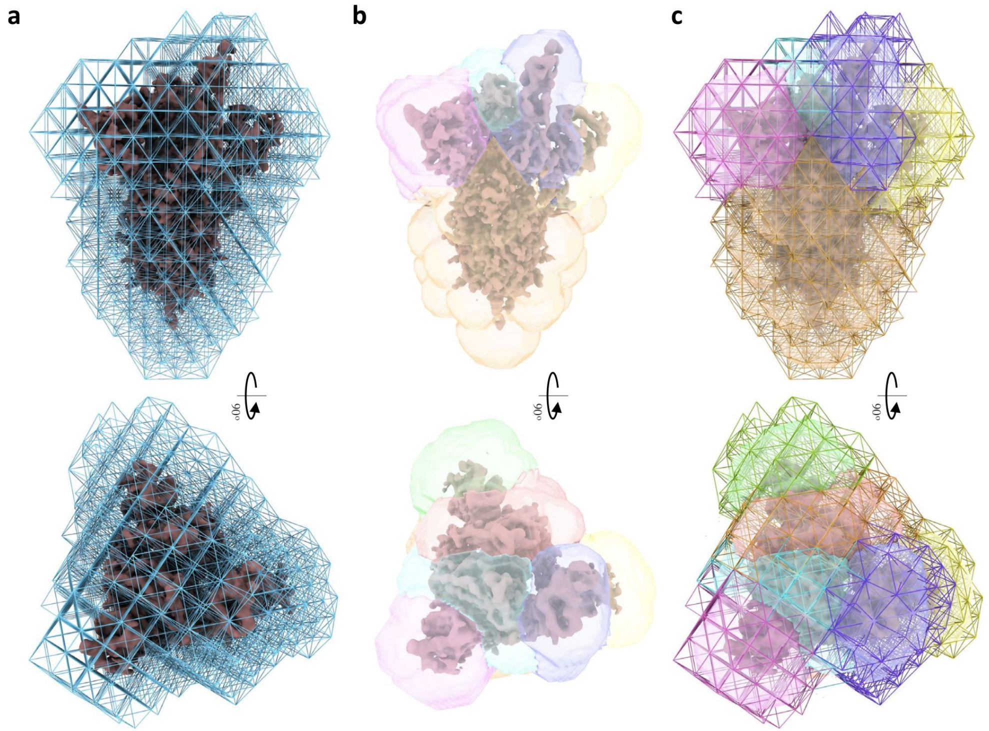
Extended Data Fig. 1 | Mesh Topologies. Examples of mesh topology that can provided to 3DFlex for the SARS-CoV-2 spike protein (see Fig. 4). **a**: a regular mesh, with all mesh elements connected to their neighbors. This does not allow adjacent protein domains to easily separate or move in different directions. **b**: coarse separation of domains. **c**: an irregular mesh constructed by fusing sub-meshes for each domain at the interfaces where density is to be continuous. 3DFlex still learns motion at all mesh nodes jointly and from scratch, but is now able to model adjacent domains that move in different directions easily.



Method — Mesh generation



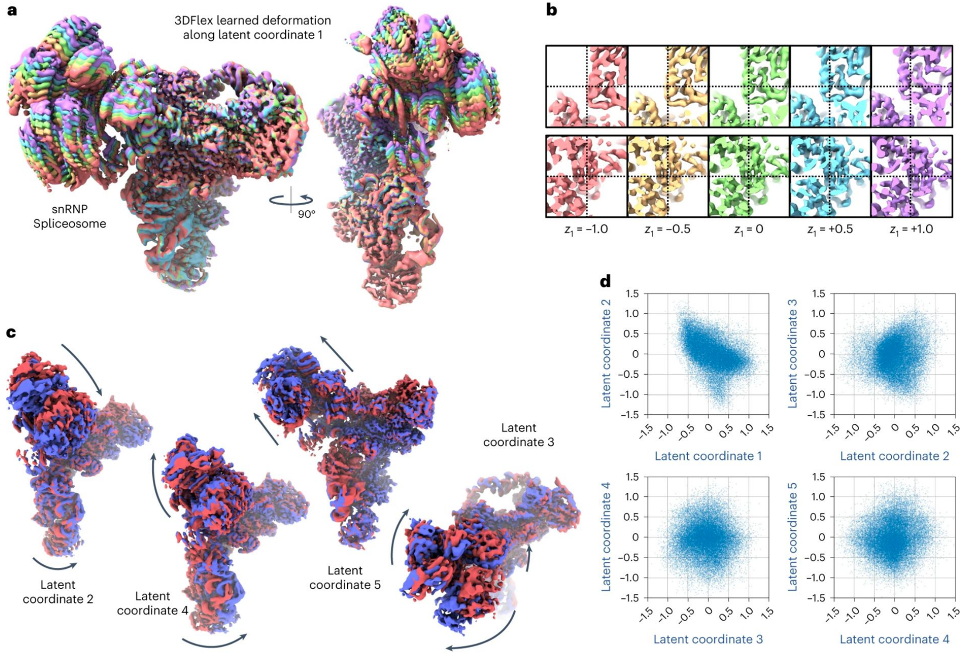
- Tetrahedral mesh is defined by:
  - A set of vertices
  - A set of tetra cells, each connecting four vertices
  - A “tetra index map”, which is an NxNxN map of indices indicating for each voxel, which tetra cell that voxel belongs to
- The convection operator uses the tetra index map to determine how to convect the V based on the movement of the mesh vertices

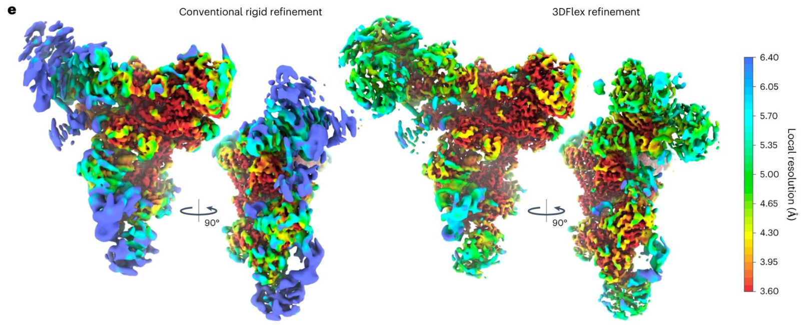


Extended Data Fig. 1 | Mesh Topologies. Examples of mesh topology that can provided to 3DFlex for the SARS-CoV-2 spike protein (see Fig. 4). **a**: a regular mesh, with all mesh elements connected to their neighbors. This does not allow adjacent protein domains to easily separate or move in different directions. **b**: coarse separation of domains. **c**: an irregular mesh constructed by fusing sub-meshes for each domain at the interfaces where density is to be continuous. 3DFlex still learns motion at all mesh nodes jointly and from scratch, but is now able to model adjacent domains that move in different directions easily.

Results: large snRNP Spliceosome complex

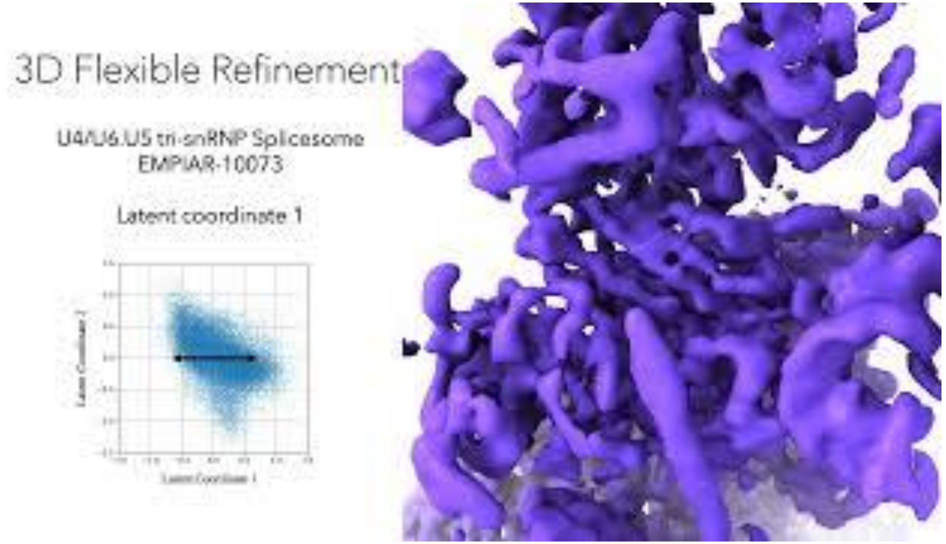
Resolves 5 dimensions of motion while improving map resolution from 5.7 to 3.8Å locally (head region)

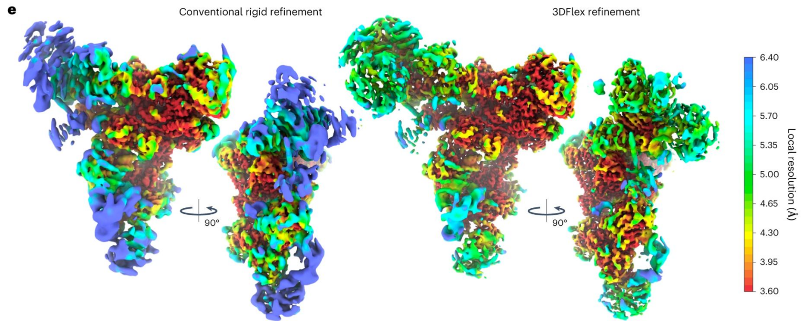




Results: large snRNP Spliceosome complex

Resolves 5 dimensions of motion while improving map resolution from 5.7 to 3.8Å locally (head region)

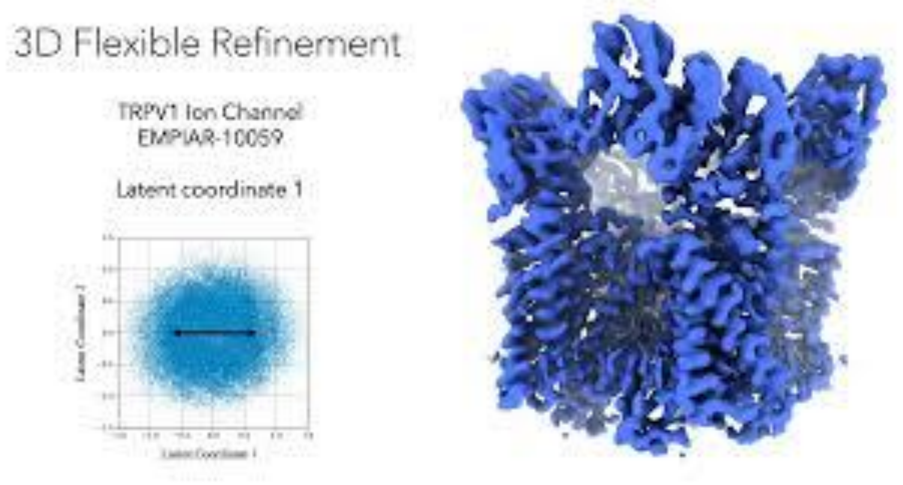




Results: smaller membrane TRPV1 ion channel

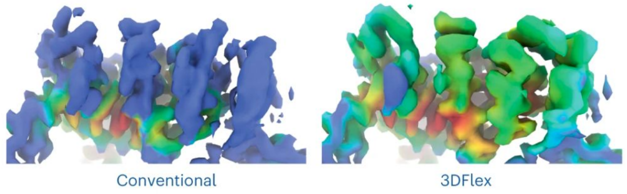
2D latent space Improves resolution of peripheral helices from 4 to 3.2Å by explicitly modeling motion

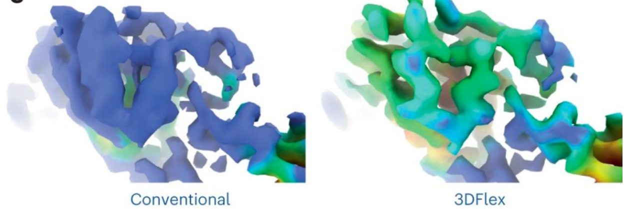
1. 1st dimension reveals inward and outward coordinated bending of opposite flexible subunits in the soluble domain
2. 2nd dimension reveals twisting of the subunits around the pore axis

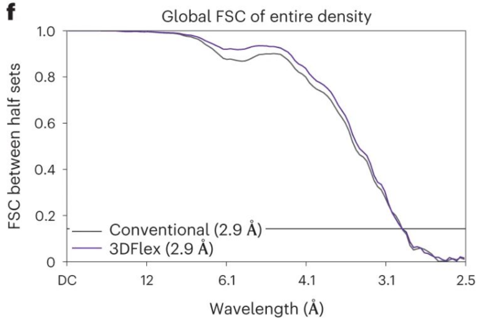


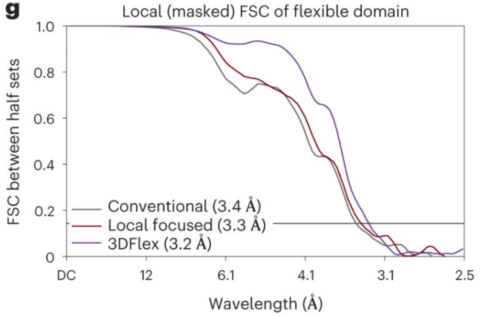
Results: smaller membrane TRPV1 ion channel

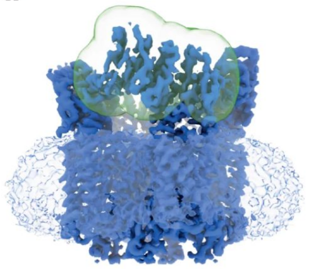
2D latent space Improves resolution of peripheral helices from 4 to 3.2Å by explicitly modeling motion

d


e


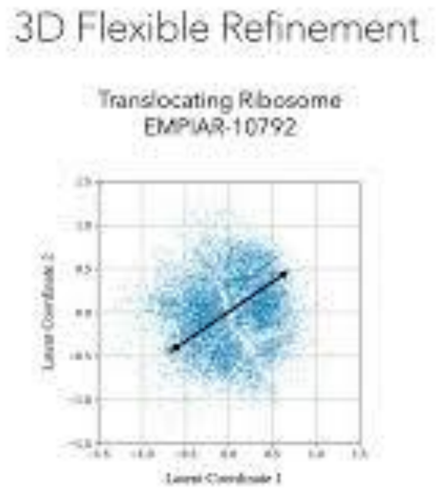
f


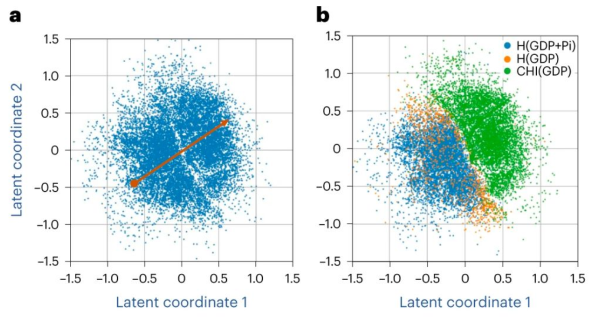
g


h


Results: translocating ribosome

Learned coordinated motion of multiple parts (e.g., large and small subunits, elongation factor, etc.) including the overall ratcheting motion of the ribosome

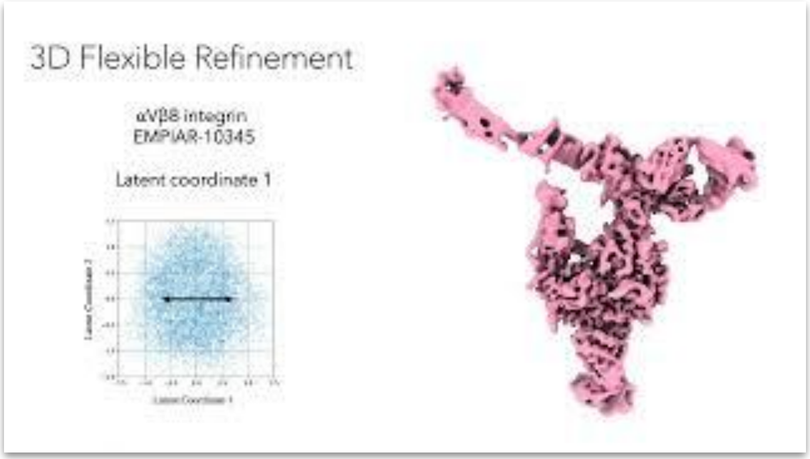


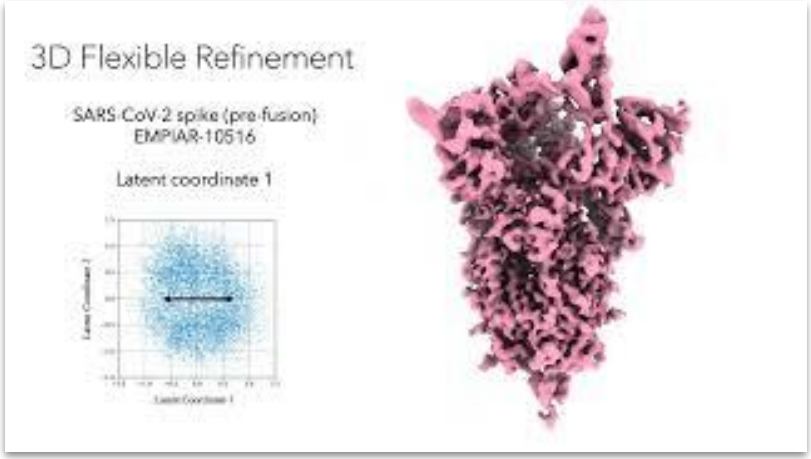


More results

αV β8 integrin: 84,266 particle images, 2D latent space → learns large bending motions of the flexible arm of the integrin particle, as well as flexibility in the Fabs that are bound

SARS-CoV-2 spike protein: 113,511 particle images, 3D latent space → learns a combination of motions of the RBD (esp. the dynamic up-RBD) and NTD domains with separate deformation





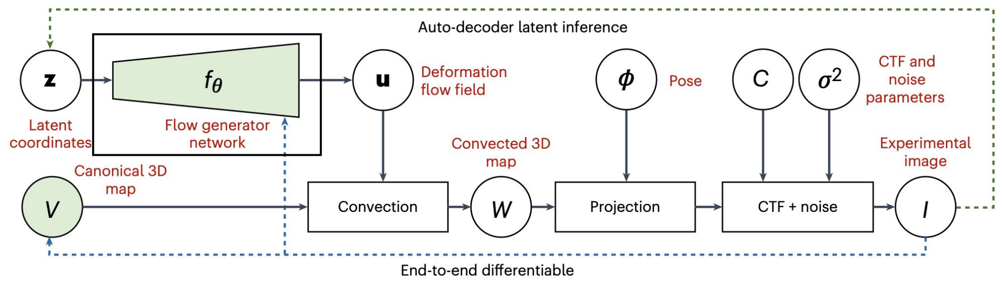
Validation by FSC

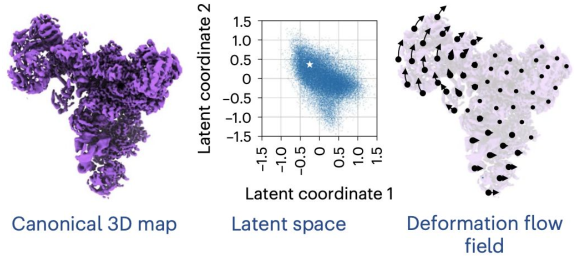
- 3DFlex is optimized at a small box size → converges, freeze flow generator parameters θ and latent coordinates z → transfer to new model at full resolution
- For each half set, initialize canonical density V = 0 and re-optimize at full box size
  - Full-batch L-BFGS in real space (more expensive than Fourier space reconstruction) but resolves all flexible parts simultaneously w/o assumptions
- Compare the 2 resulting half-maps with FSC
- Correlation beyond the training time Nyquist resolution limit indicates consistent signal recovery from separate particle sets

Results Summary

- Validation by FSC shows improved alignment beyond training resolution
- Results show 3DFlex can uncover intricate, biologically relevant motions
    - Resolves high-resolution detail in canonical map by aggregating signal across conformations
    - Delineates motions that improve map quality beyond conventional refinement
- Applicable across range of complexities and sizes from 1,200 kDa spliceosome to 380 kDa ion channel
- Provides interpretable visualization of conformational landscape

Method — Flow generator





- Input: per-particle latent coordinates $z_{1:M}$
- Output: $u(x)$ (Deformation flow field)
    - x: 3D position

Discussion

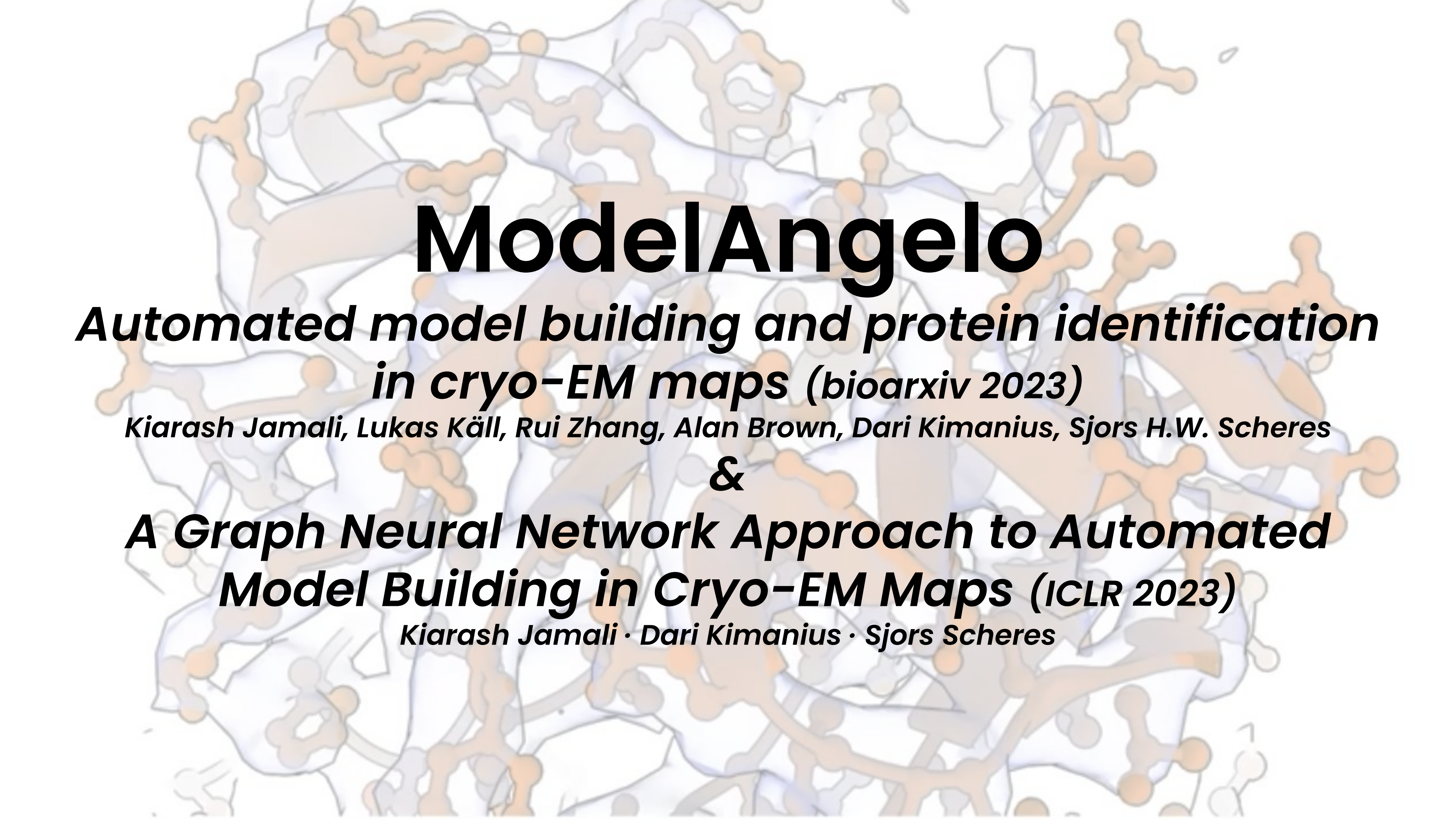
- 3DFlex complements 3D classification and multi-body refinement methods
  - Applicable in cases where 3D classification requires large data and multi-body rigid subunits
  - Provides interpretable latent space to facilitate particle selection

Discussion

- 3DFlex complements 3D classification and multi-body refinement methods
  - Applicable in cases where 3D classification requires large data and multi-body rigid subunits
  - Provides interpretable latent space to facilitate particle selection
- Limitations: not designed to handle compositional heterogeneity
  - Learns transitions between discrete states, but guided by rigidity priors
  - Captures large-scale motions relevant to function, but may miss subtle sidechain changes

Discussion

- 3DFlex complements 3D classification and multi-body refinement methods
  - Applicable in cases where 3D classification requires large data and multi-body rigid subunits
  - Provides interpretable latent space to facilitate particle selection
- Limitations: not designed to handle compositional heterogeneity
  - Learns transitions between discrete states, but guided by rigidity priors
  - Captures large-scale motions relevant to function, but may miss subtle sidechain changes
- Future work: validate deformation fields, incorporate structurally-aware priors
  - Extend model to handle compositional variability
  - Explore alternatives like neural fields for motion and density representation



ModelAngelo

Automated model building and protein identification in cryo-EM maps *(bioarxiv 2023)*

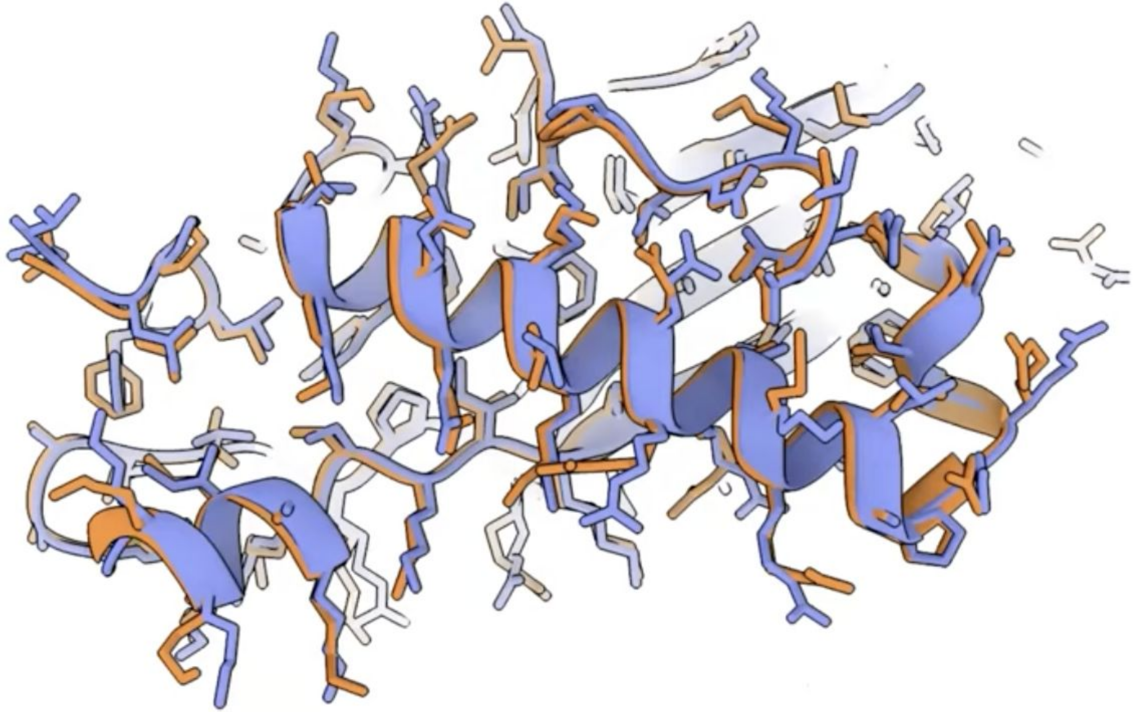
Kiarash Jamali, Lukas Käll, Rui Zhang, Alan Brown, Dari Kimanius, Sjors H.W. Scheres

&

A Graph Neural Network Approach to Automated Model Building in Cryo-EM Maps *(ICLR 2023)*

Kiarash Jamali · Dari Kimanius · Sjors Scheres

Table of Contents



Abstract

1. ModelAngelo uses ML for **automated atomic model building** and **protein identification** from **cryo-EM** maps
2. How? It combines information from the cryo-EM map, protein sequences, and local geometry using a graph neural network (GNN)
3. So, does it work? How well?
   a. For proteins, it builds models of similar quality as human experts
   b. Using predicted amino acid probabilities in HMM searches, it outperforms experts in identifying unknown proteins
   c. For nucleotides, it builds accurate backbones
4. Why? ModelAngelo aims to “help remove bottlenecks and increase objectivity in cryo-EM structure determination”
   a. Error-checking and refinement remain necessary

Introduction: Cryo-EM

1. 3D atomic models of proteins and nucleic acids are pivotal for understanding molecular processes
2. Cryo-EM can now determine near-atomic resolution structures
    a. Current trend in the EMDB predicts that ~100k cryo-EM structures will be determined in the next 5 years
    b. Automation is key to remove bottlenecks and increase objectivity as cryo-EM grows exponentially
3. Over two-thirds of 2022's cryo-EM structures were better than 4Å
    a. Atomic modeling in 2-4Å maps uses sequence information but is laborious and expert-dependent
    b. Errors in atomic models can have serious consequences

Motivation

1. Atomic model building in cryo-EM typically done **manually** using 3D visualization software followed by refinement
   a. Difficult for maps of resolution >4Å
   b. Still very tedious, time consuming, and requires high expertise for resolutions <3Å
   c. In weak density areas, sequences have to be used to identify residues

Motivation

1. Atomic model building in cryo-EM typically done **manually** using 3D visualization software followed by refinement
   a. Difficult for maps of resolution >4Å
   b. Still very tedious, time consuming, and requires high expertise for resolutions <3Å
   c. In weak density areas, sequences have to be used to identify residues
2. Automated atomic model building from X-ray crystallography applied to cryo-EM: **incomplete** & w/ **large residuals**
   a. PHENIX package: 47% completeness for resolutions >3Å
   b. MAINMAST: RMSD in the ~10-19Å range

Motivation

1. Atomic model building in cryo-EM typically done **manually** using 3D visualization software followed by refinement
    a. Difficult for maps of resolution >4Å
    b. Still very tedious, time consuming, and requires high expertise for resolutions <3Å
    c. In weak density areas, sequences have to be used to identify residues
2. Automated atomic model building from X-ray crystallography applied to cryo-EM: **incomplete** & w/ **large residuals**
    a. PHENIX package: 47% completeness for resolutions >3Å
    b. MAINMAST: RMSD in the ~10-19Å range
3. More recently, DeepTracer (2021) faces **limitations**
    a. Uses U-Nets, no sequence integration, no graph representation
        i. Segmentation + classification only, so cannot refine built models or recycle
        ii. Builds atoms for main chains only

Motivation

1. Attempts to **dock and morph** structure predictions to cryo-EM also face limitations
    a. Conformation changes in complex can lead to propagating errors

Motivation

1. Attempts to **dock and morph** structure predictions to cryo-EM also face limitations
    a. Conformation changes in complex can lead to propagating errors
2. **GNNs** to model proteins
    a. Since relative orientation is important to model building, use SE(3) equivariant (not invariant) approach
    b. Since torsion angle representation require known residue ordering and connectivity, use backbone frame representation (AF2) instead

Motivation

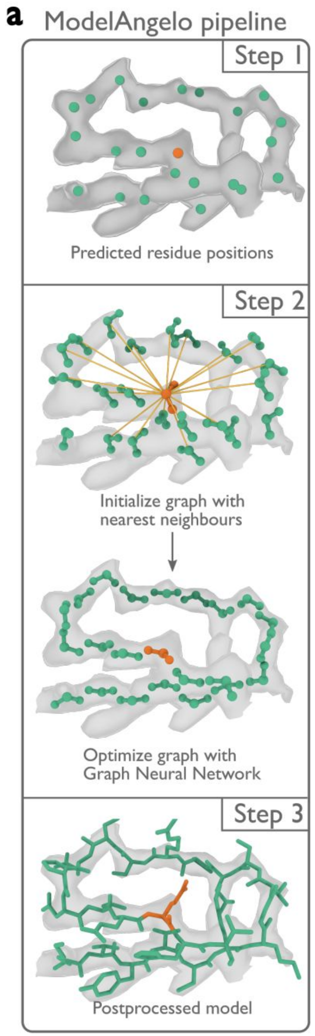
1. Attempts to **dock and morph** structure predictions to cryo-EM also face limitations
    a. Conformation changes in complex can lead to propagating errors
2. **GNNs** to model proteins
    a. Since relative orientation is important to model building, use SE(3) equivariant (not invariant) approach
    b. Since torsion angle representation require known residue ordering and connectivity, use backbone frame representation (AF2) instead
3. With <13,000 sub-4Å maps available, **multimodal** ML is highly advantageous (and necessary)
    a. ModelAngelo combines map, sequence, and geometry information, much like human experts

Methods

Overview of building atomic models:

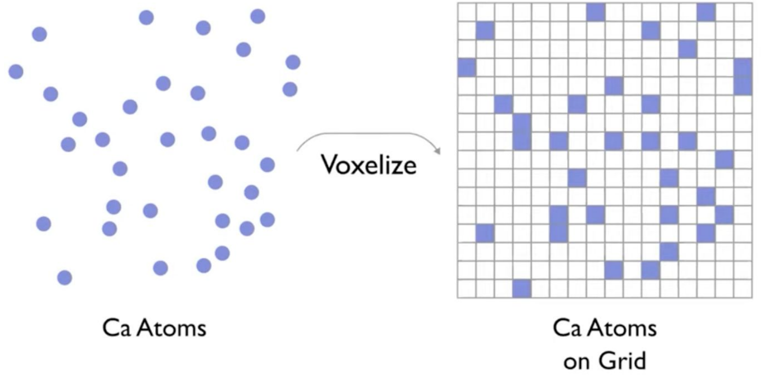
1. **Graph initialization**: CNN predicts residue positions
   a. Feature pyramid network predicts whether each voxel in cryo-EM map contains the C-alpha atom of an amino acid (or phosphor atom of a nucleotide)
2. **Graph optimization**: GNN refines residue positions/orientations
   a. Also predicts residue identities + torsion angles of side chains (or bases)
3. **Post-processing**: build full atomic model from optimized graph

Also…identify proteins by building profile HMMs


a
ModelAngelo pipeline
Step 1
Predicted residue positions
Step 2
Initialize graph with nearest neighbours
Optimize graph with Graph Neural Network
Step 3
Postprocessed model

Graph Initialization

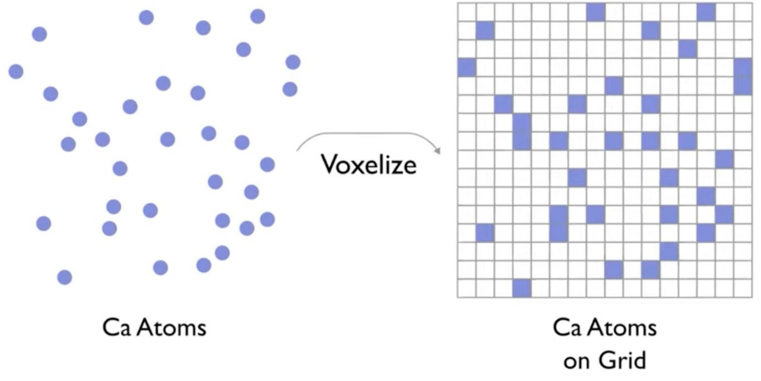
1. Min distance between 2 Cα's is 3.8Å, so resampling cryo-EM voxel maps with pixel size of 1.5Å ensures that no voxel contains >1 Cα
2. Residues (defined by Cα) = Nodes, Neighbors = Edges

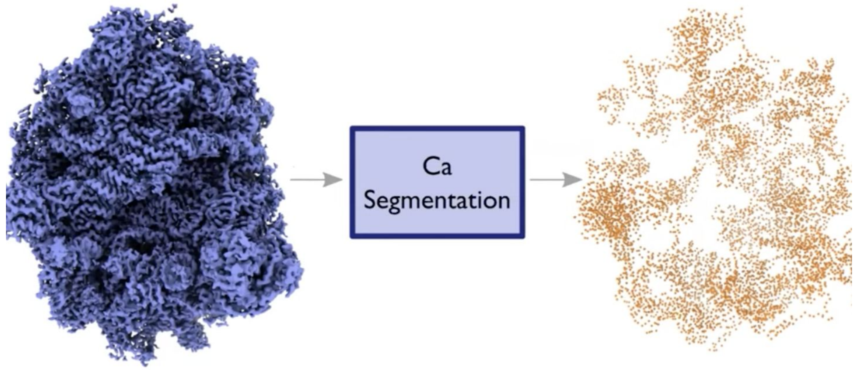


Graph Initialization

1. Since # true Cα's (M*) << # voxels (N), focal loss over binary CE loss
   a. Upweight voxels containing Cα for best P/R balance
   b. Ensures ½ of map loss from empty voxels and ½ from Cα-voxels

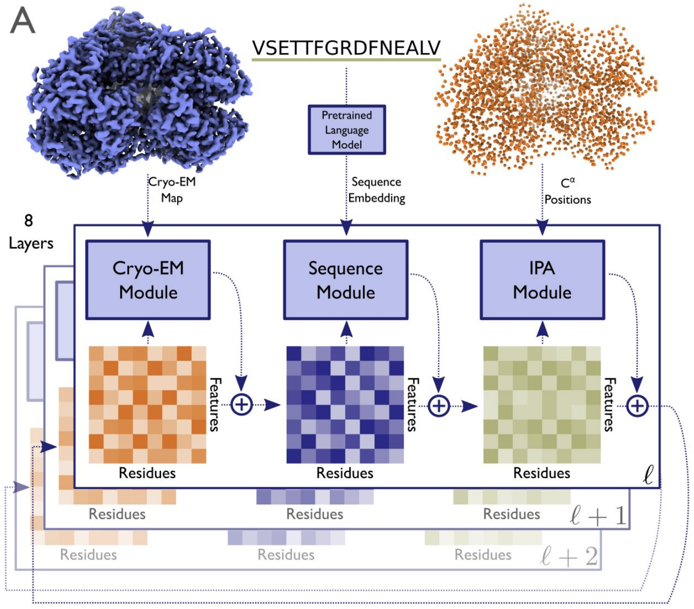
$$\underset{\text{weight}}{w_x} = \underbrace{\frac{N - M^*}{M*}}_{}\chi(x) + (1 - \chi(x)) \qquad \chi = \mathbb{1}\{\text{contains } C\alpha\}$$





Graph Optimization

GNN comprised of **3 modules** that take **residue feature vector** as input and gradually updates it with new information in 8 layers



1. **Cryo-EM** module: map densities
2. **Sequence** module (optional): protein sequences only
3. **IPA** module (invariant point attention from AF2): local geometry

Graph Optimization

Graph representation:

X(n): Ca positions at iteration n
F: affine frames defined by Ca, C, N backbone
G: torsion angles of side chains
P: per-residue probability vector for all 20 a.a.s
S: ESM-1b embeddings of all residues in input sequence
O: per-residue confidence prediction

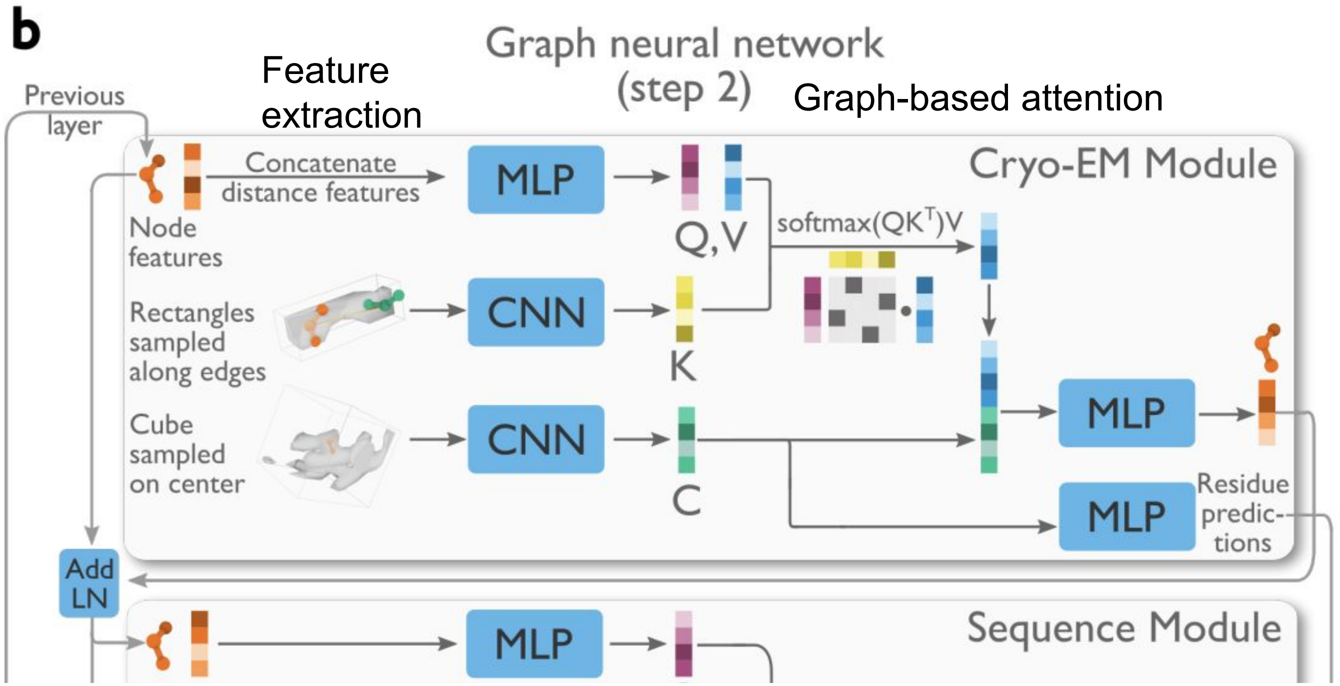
- F,G similar to AF2
- (X, F, G, P) computes all-atom coords

Training Objective:

$$g_\phi\left(X^{(n)}, F^{(n)}, V, S\right) \approx \left(X^*, F^*, G^*, P^*\right)$$

X*: set of Ca positions in training data
F*,G* computed from atom coords of all residues in training data
P* (M* x 20 vector of 0s, 1s): one-hot encoding of a.a. Classes in training data

Cryo-EM Module



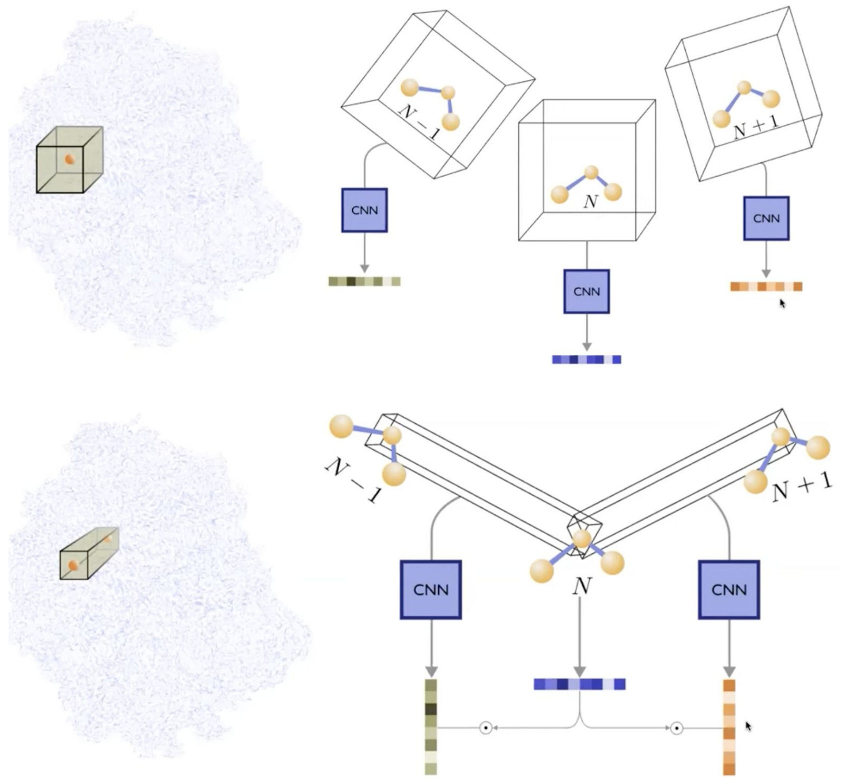
Feature extraction

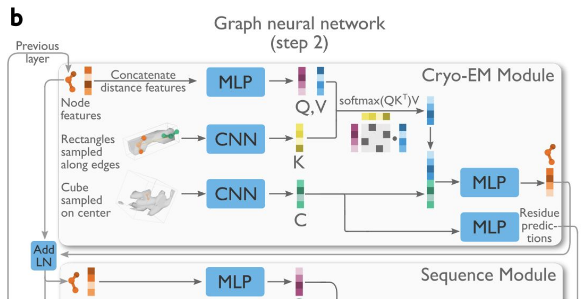
Graph-based attention

Output residue feature vector

Identity prediction

Cryo-EM Map Data



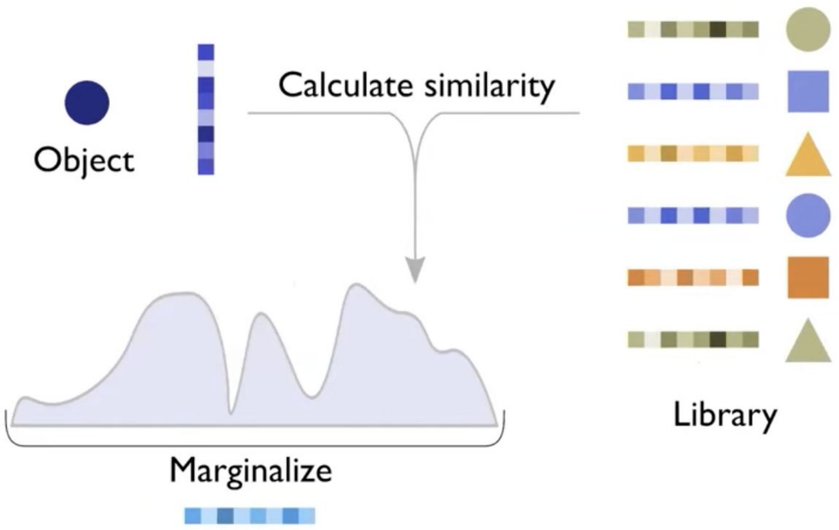


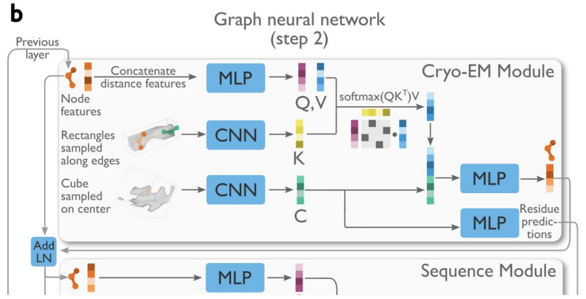
1. **Node** feature (query, value)

2. **Edge** feature cuboid density (key) between residue and its 20 nearest neighbors
   a. Connectivity: peptide bonds or sidechain interactions

3. **Cube** density around residue
   a. Orientation determined by **F**
   b. Concatenated w/ attention output

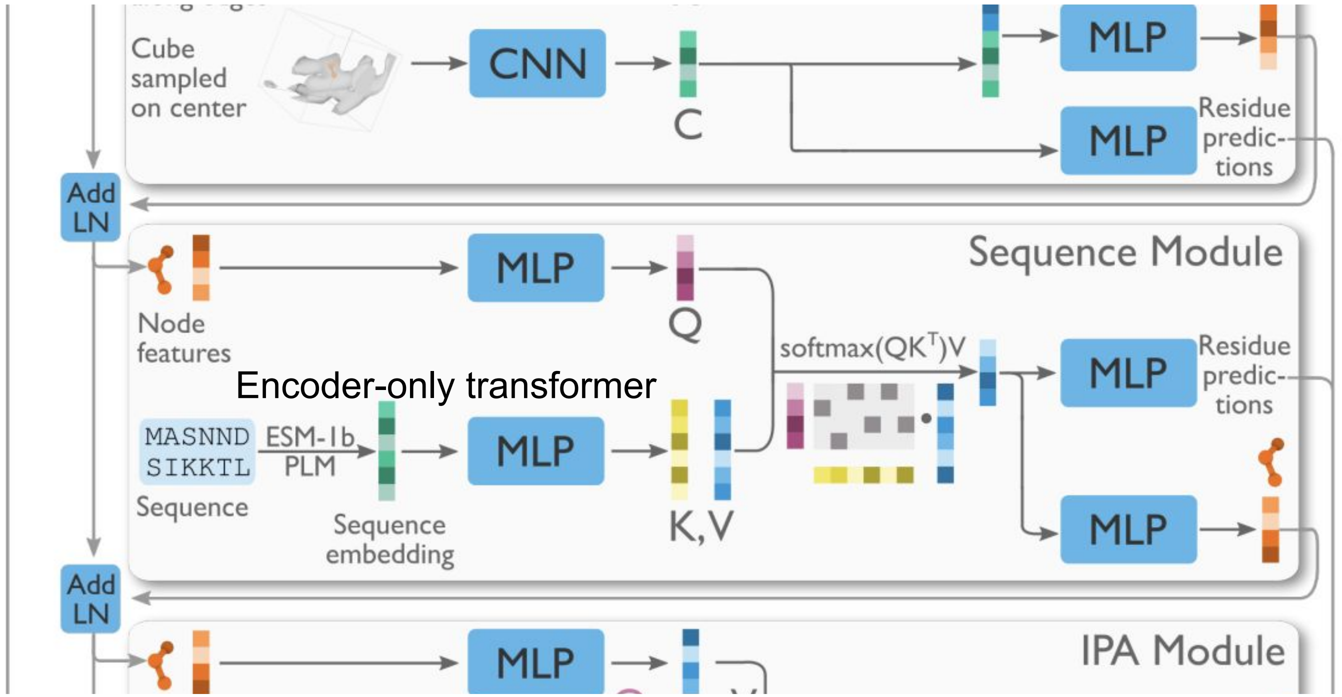
Cross-Attention: Query, Key, Value



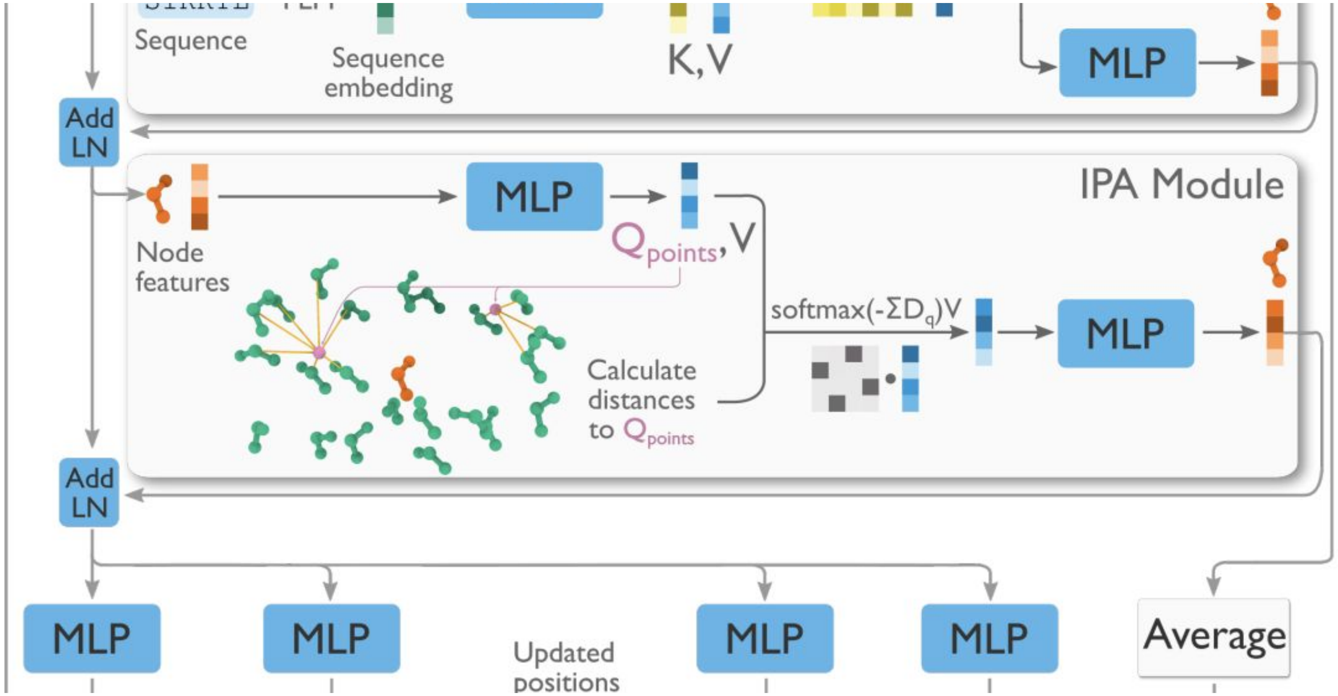


An attention function can be described as mapping a query and a set of key-value pairs to an output, where the query, keys, values, and output are all vectors. The output is computed as a weighted sum of the values, where the weight assigned to each value is computed by a compatibility function of the query with the corresponding key.

Sequence Module

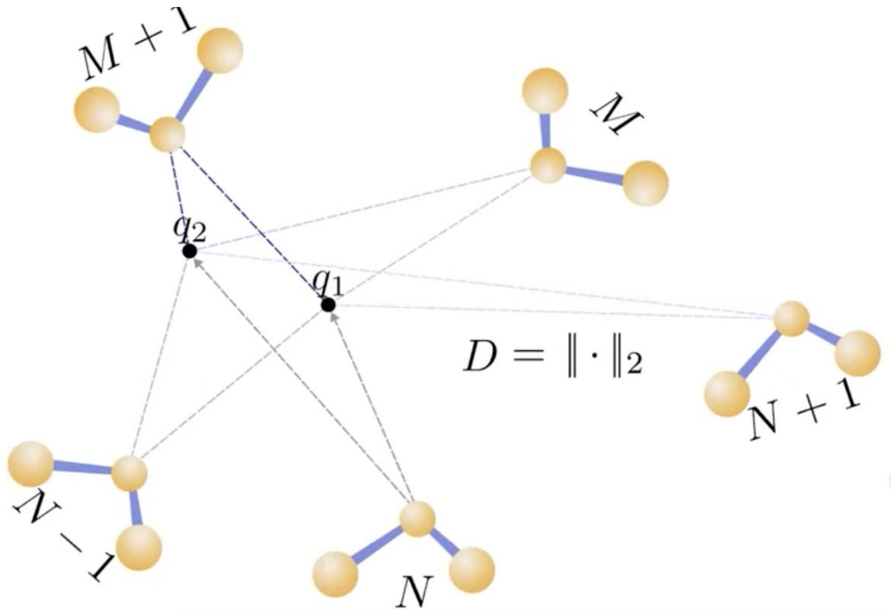


IPA Module

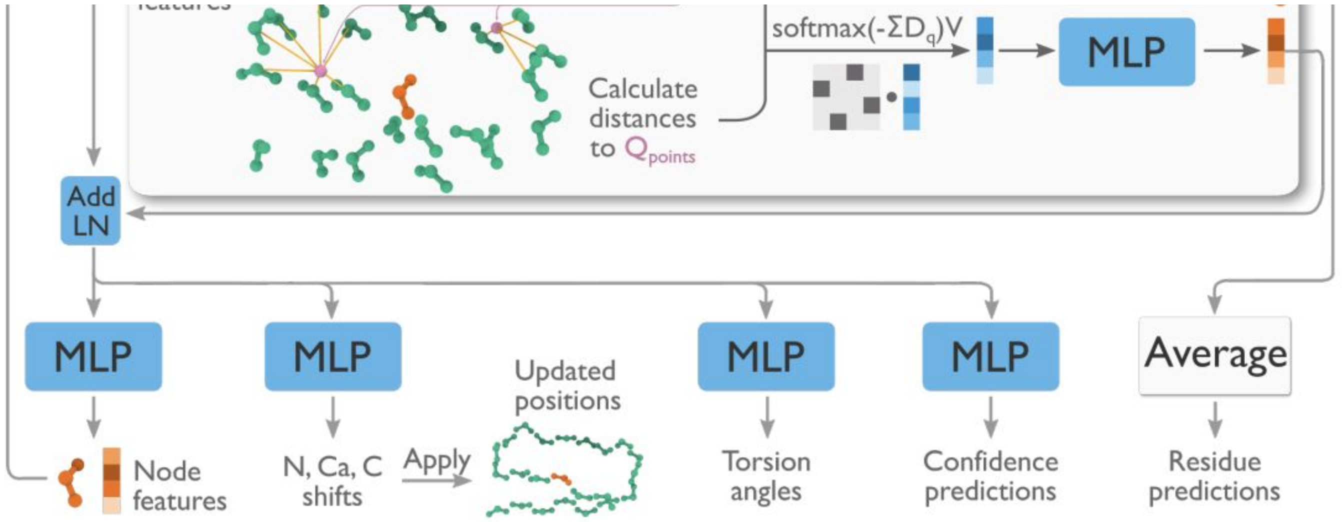

Sequence
Sequence embedding
K,V
MLP
Add LN
MLP
IPA Module
Node features
Q_{points}, V
softmax(-ΣD$_q$)V
MLP
Calculate distances to Q_{points}
Add LN
MLP
MLP
Updated positions
MLP
MLP
Average

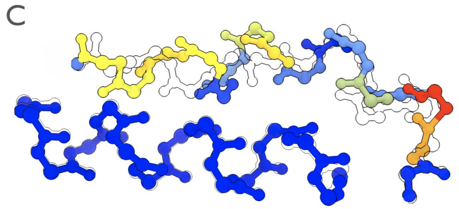
IPA Module

1. MLP computes 4 query points per residue and Euclidean distances between query points
   a. Location of neighbors replace cosine similarity of attention between Q, K
2. Ablating IPA led to incorrect **secondary structure** geometry



Post-processing





Post-processing

After each layer, the output residue **feature vector** is used to…

1. Predict & update new per-residue positions and orientations
2. Predict torsion angles for side chains/bases
3. Predict per-residue confidence score (= **backbone RMSD** against deposited PDB structure)

Post-processing

After each layer, the output residue **feature vector** is used to…

1. Predict & update new per-residue positions and orientations
2. Predict torsion angles for side chains/bases
3. Predict per-residue confidence score (= **backbone RMSD** against deposited PDB structure)

Predicted residue **identities** from **Cryo-EM** and **Sequence** modules are

4. Averaged to generate probabilities → Hidden Markov Model (HMM) profile → search against input sequences using HMMER

Post-processing

After each layer, the output residue **feature vector** is used to…

1. Predict & update new per-residue positions and orientations
2. Predict torsion angles for side chains/bases
3. Predict per-residue confidence score (= **backbone RMSD** against deposited PDB structure)

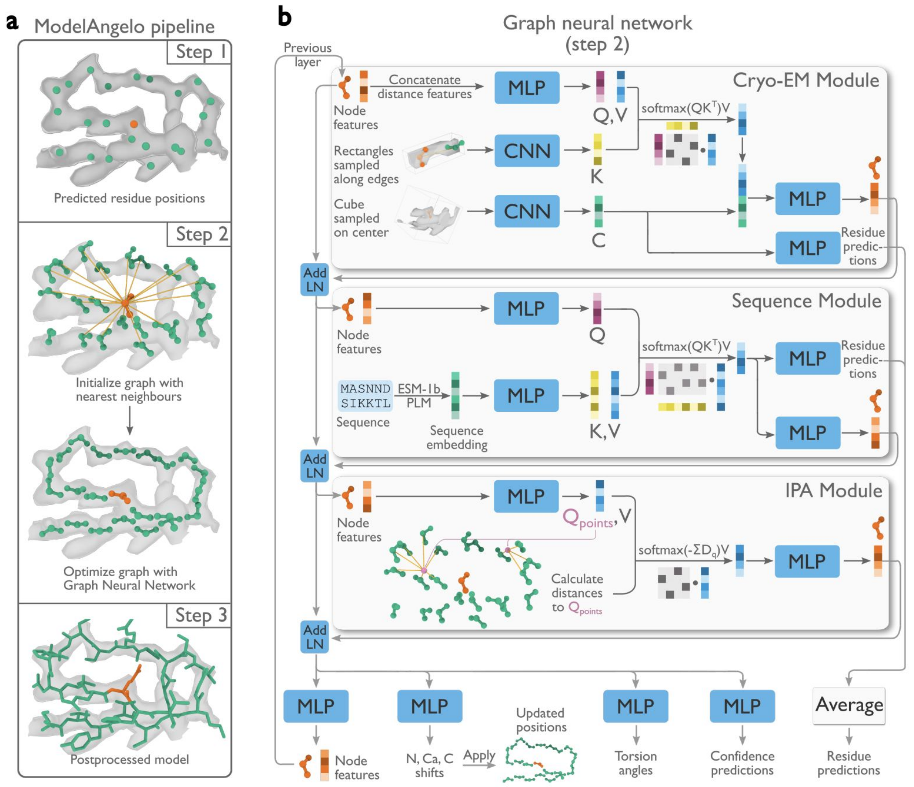
Predicted residue **identities** from **Cryo-EM** and **Sequence** modules are

4. Averaged to generate probabilities → Hidden Markov Model (HMM) profile → search against input sequences using HMMER

Finally,

5. Separate chains are connected, chains of <4 residues are pruned

Overview: Atomic Model Building



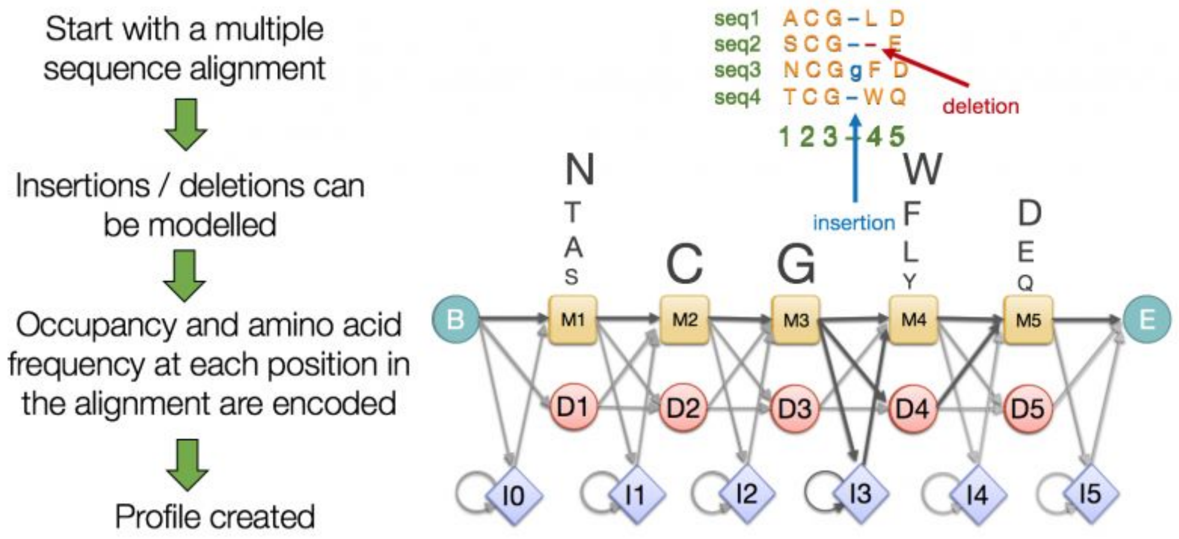
Algorithm 9 GNN Forward Pass

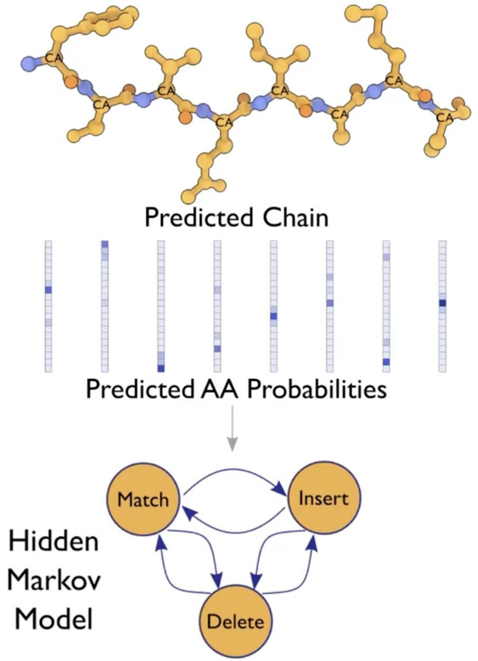
```
1:  function GNN_FORWARD_PASS($\mathbf{x}, \mathbf{F}, \boldsymbol{V}, \boldsymbol{S}$, num_recycling_steps)
2:      for $k$ in range(num_recycling_steps) do
3:          $\mathbf{f} \leftarrow$ zeros(batch_size, 256)
4:          for $i$ in range(num_layers=8) do
5:              $\mathbf{f} \leftarrow$ cryo_em_attention($\mathbf{x}, \mathbf{f}, \mathbf{F}, \boldsymbol{V}$)
6:              $\mathbf{f} \leftarrow$ sequence_attention($\mathbf{f}, \mathbf{S}$)
7:              $\mathbf{f} \leftarrow$ spatial_ipa($\mathbf{x}, \mathbf{f}, \mathbf{F}$)
8:              $\mathbf{f} \leftarrow$ transition_layer($\mathbf{f}$)
9:              $\mathbf{P} \leftarrow$ amino_acid_classification($\mathbf{f}$)
10:             $\mathbf{O} \leftarrow$ local_confidence_prediction($\mathbf{f}$)
11:             $\mathbf{x}, \mathbf{F} \leftarrow$ backbone_frame_module($\mathbf{f}, \mathbf{F}$)
12:         $\mathbf{G} \leftarrow$ torsion_angle_network($\mathbf{f}$)
13:     return $\mathbf{x}, \mathbf{F}, \mathbf{P}, \mathbf{O}, \mathbf{G}$
```

With the model built…
Now we identify the protein by building a profile HMM

Profile HMM Building

1. Parameters usually derived from MSA → now derived from ModelAngelo
2. Probabilistic search against the input sequence > assigning the most probable


Start with a multiple sequence alignment
Insertions / deletions can be modelled
Occupancy and amino acid frequency at each position in the alignment are encoded
Profile created
seq1 ACG-LD
seq2 SCG--E
seq3 NCGgFD
seq4 TCG-WQ
deletion
1 2 3 4 5
insertion
N T A S
C
G
W F L Y
D E Q
B
M1
M2
M3
M4
M5
E
D1
D2
D3
D4
D5
I0
I1
I2
I3
I4
I5



Profile HMM Building

1. Transition probabilities between match (M), insert (I) and delete (D) states
   a. Higher confidence residues have higher probability of staying in the match state
   b. Confidence metric c(i) for residue i

$$P^{(i)}_{M\to M} = \max(c^{(i)} - d, 0.5) \qquad P^{(i)}_{D\to M} = 1 - d \qquad P^{(i)}_{I\to M} = 1 - d$$

$$p^{(i)}_{M\to D} = \frac{1 - p^{(i)}_{M\to M}}{2} \qquad P^{(i)}_{D\to D} = d \qquad P^{(i)}_{I\to D} = 0$$

$$p^{(i)}_{M\to I} = \frac{1 - p^{(i)}_{M\to M}}{2} \qquad P^{(i)}_{D\to I} = 0 \qquad P^{(i)}_{I\to I} = d$$

2. Emission probabilities of each amino acid is predicted
3. By using predicted probabilities rather than a true MSA, ModelAngelo can *search sequences it has no prior knowledge of*

Training & Inference

1. Recycle the post-processed model (like AF2) for the next round of graph optimization 1-3 times (randomly)
   a. Inference: 3 times, after which performance plateaus
2. Training Data: EMDB + PDB → 3715 map-model pairs
   a. Before 4/1/2022 (test split after this date)
   b. Resolutions <4Å
   c. Augmented w/ color noise & random sharpening/dampening/rotations(90°)
3. Special version trained for structures with unknown sequences
   a. Sequence module is removed
   b. ModelAngelo still predicts probability distribution
   c. Larger proteome (no inputs) searched in HMMER

Training Losses

1. Ca RMSD

$$\mathcal{L}_{C^\alpha} = \frac{1}{N}\sum_i \mathrm{RMSD}(\mathbf{x}_i, g_\theta(\mathbf{x}_i + \mathbf{e}_i)) = \frac{1}{N}\sum_i \sqrt{\frac{1}{3}\sum_{d=1}^{3} \left|[\mathbf{x}_i]_d - [g_\theta(\mathbf{x}_i + \mathbf{e}_i)]_d\right|^2}$$

2. Backbone RMSD
3. Amino acid classification
4. Local confidence score
5. Torsion angles
6. Full atom
   a. Physical restraints-based relaxation

1. Training loop:
   a. take a PDB structure
   b. extract Ca's (x)
   c. distort them with noise (e)
   d. initialize backbones for each node/residue/Ca randomly
   e. predict the original structure

2. Missing/adding residues
   a. 10% of residues are replaced with random residues
   b. predicts whether a node actually exists in the model

Results

ModelAngelo builds protein models of comparable quality to those built by humans

a) Backbone RMSD between the protein models built by ModelAngelo and those deposited (pink)

- Lower RMSDs for residues with higher (better) Q-scores
- Completeness improves for residues with higher Q-scores (blue)

b) Residues not built by ModelAngelo have lower Q-scores than those that are built

c) Models built by ModelAngelo are of similar quality to the deposited ones

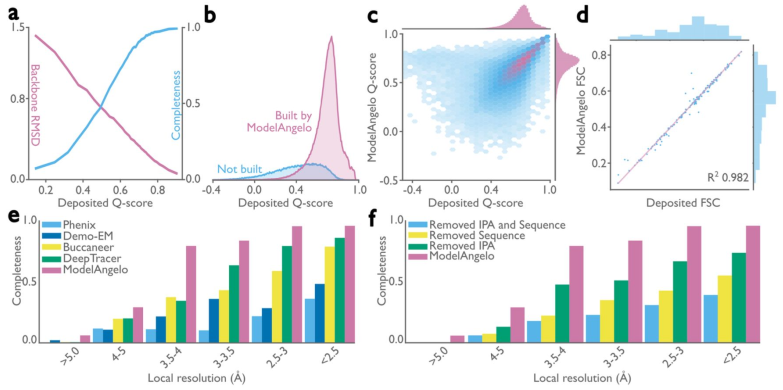


Figure 2: Performance of ModelAngelo for atomic modelling of proteins. a, Backbone root-mean-square deviation (RMSD) and model completeness plotted as a function of the target model Q-scores. **b**, Q-score distribution of residues in the deposited models, comparing those built by ModelAngelo with those not built. **c**, Q-score comparison between ModelAngelo predicted models and the deposited models. **d**, Model-to-map Fourier shell correlation (FSC), as calculated by Servalcat (*38*), after refining both models and using only residues present in both ModelAngelo and deposited models. **e**, Model completeness for various automated model-building softwares for different local resolution ranges in the maps. **f**, Model completeness for ModelAngelo and versions of ModelAngelo where its Sequence and/or IPA modules were ablated. Panels a-d relate to the test set of 177 structures; panels e and f to the subset of 27 structures.

- Q-scores measures the resolvability of individual atoms in cryo-EM maps
- Completeness: the fraction of residues that are built with their $C\alpha$ atom within $3A$ of the deposited model and with the correct amino acid assignment

Results

ModelAngelo outperforms alternative approaches

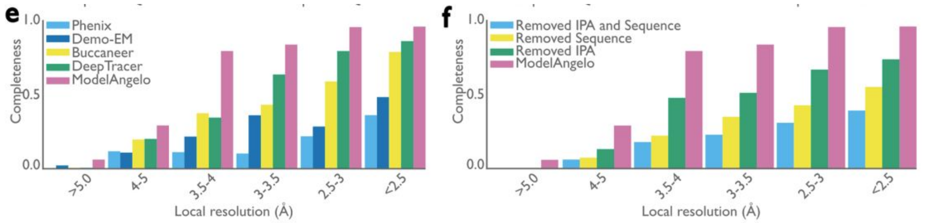
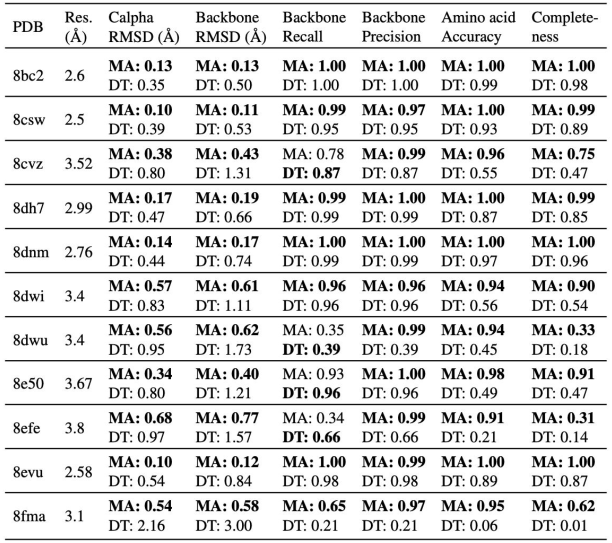


Figure 2: Performance of ModelAngelo for atomic modelling of proteins. a, Backbone root-mean-square deviation (RMSD) and model completeness plotted as a function of the target model Q-scores. **b**, Q-score distribution of residues in the deposited models, comparing those built by ModelAngelo with those not built. **c**, Q-score comparison between ModelAngelo predicted models and the deposited models. **d**, Model-to-map Fourier shell correlation (FSC), as calculated by Servalcat (*38*), after refining both models and using only residues present in both ModelAngelo and deposited models. **e**, Model completeness for various automated model-building softwares for different local resolution ranges in the maps. **f**, Model completeness for ModelAngelo and versions of ModelAngelo where its Sequence and/or IPA modules were ablated. Panels a-d relate to the test set of 177 structures; panels e and f to the subset of 27 structures.



PDB	Res. (Å)	Calpha RMSD (Å)	Backbone RMSD (Å)	Backbone Recall	Backbone Precision	Amino acid Accuracy	Complete-ness
8bc2	2.6	**MA: 0.13** DT: 0.35	**MA: 0.13** DT: 0.50	**MA: 1.00** DT: 1.00	**MA: 1.00** DT: 1.00	**MA: 1.00** DT: 0.99	**MA: 1.00** DT: 0.98
8csw	2.5	**MA: 0.10** DT: 0.39	**MA: 0.11** DT: 0.53	**MA: 0.99** DT: 0.95	**MA: 0.97** DT: 0.95	**MA: 1.00** DT: 0.93	**MA: 0.99** DT: 0.89
8cvz	3.52	**MA: 0.38** DT: 0.80	**MA: 0.43** DT: 1.31	MA: 0.78 **DT: 0.87**	**MA: 0.99** DT: 0.87	**MA: 0.96** DT: 0.55	**MA: 0.75** DT: 0.47
8dh7	2.99	**MA: 0.17** DT: 0.47	**MA: 0.19** DT: 0.66	**MA: 0.99** DT: 0.99	**MA: 1.00** DT: 0.99	**MA: 1.00** DT: 0.87	**MA: 0.99** DT: 0.85
8dnm	2.76	**MA: 0.14** DT: 0.44	**MA: 0.17** DT: 0.74	**MA: 1.00** DT: 0.99	**MA: 1.00** DT: 0.99	**MA: 1.00** DT: 0.97	**MA: 1.00** DT: 0.96
8dwi	3.4	**MA: 0.57** DT: 0.83	**MA: 0.61** DT: 1.11	**MA: 0.96** DT: 0.96	**MA: 0.96** DT: 0.96	**MA: 0.94** DT: 0.56	**MA: 0.90** DT: 0.54
8dwu	3.4	**MA: 0.56** DT: 0.95	**MA: 0.62** DT: 1.73	MA: 0.35 **DT: 0.39**	**MA: 0.99** DT: 0.39	**MA: 0.94** DT: 0.45	**MA: 0.33** DT: 0.18
8e50	3.67	**MA: 0.34** DT: 0.80	**MA: 0.40** DT: 1.21	MA: 0.93 **DT: 0.96**	**MA: 1.00** DT: 0.96	**MA: 0.98** DT: 0.49	**MA: 0.91** DT: 0.47
8efe	3.8	**MA: 0.68** DT: 0.97	**MA: 0.77** DT: 1.57	MA: 0.34 **DT: 0.66**	**MA: 0.99** DT: 0.66	**MA: 0.91** DT: 0.21	**MA: 0.31** DT: 0.14
8evu	2.58	**MA: 0.10** DT: 0.54	**MA: 0.12** DT: 0.84	**MA: 1.00** DT: 0.98	**MA: 0.99** DT: 0.98	**MA: 1.00** DT: 0.89	**MA: 1.00** DT: 0.87
8fma	3.1	**MA: 0.54** DT: 2.16	**MA: 0.58** DT: 3.00	**MA: 0.65** DT: 0.21	**MA: 0.97** DT: 0.21	**MA: 0.95** DT: 0.06	**MA: 0.62** DT: 0.01

Extended Data Table 1: Comparison with alternative approaches for the automated building of proteins. *MA* stands for ModelAngelo and *DT* for DeepTracer. *Calpha RMSD* is the root mean squared deviation of the predicted CA atoms against that of the deposition. *Backbone RMSD* is similar but for the CA, C, O and N atoms of the protein backbones. *Backbone recall* is the fraction of the deposited residues that were predicted to be within 3 Å (as measured between CA atoms). *Backbone precision* is the fraction of the predicted residues that have a corresponding residue present in the deposition within 3 Å. *Amino acid accuracy* is the fraction of the predicted residues that have a correctly predicted amino acid identity. Finally, *completeness* is the fraction of deposited residues that were predicted with the correct base annotation. Numbers indicated in boldface are the best in each metric.

Results

ModelAngelo builds good nucleic acid backbones

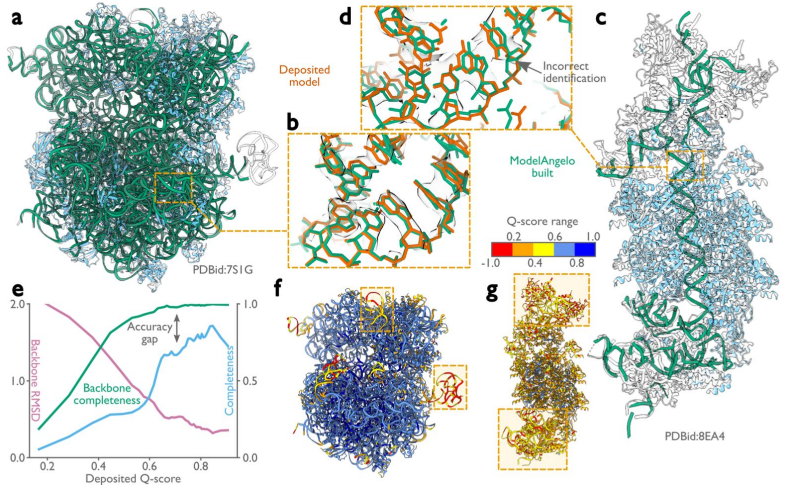


Figure 3: Performance of ModelAngelo for atomic modelling of nucleic acids. **a**, *Escherichia coli* ribosome built by ModelAngelo (with ribosomal RNA in green and proteins in blue) compared with the deposited model (PDB ID: 7S1G, black outline) (*43*). **b**, Zoomed-in view with nucleotide bases showing high accuracy compared to the deposited model (orange). **c**, ModelAngelo model of the V-K CAST transpososome from *Scytonema hofmanni* compared with the deposited model (PDB ID: 8EA4) (*44*). Sections not built by ModelAngelo (black outline) are in regions of low Q-score (see panel g). **d**, Zoomed-in view comparing the nucleotide bases of both models showing a sequence incorrectly identified by ModelAngelo. **e**, Backbone RMSD, backbone completeness, and sequence completeness plotted against the deposited Q-score for six ribosome structures. **f, g,** Deposited models for the structures in a and c, coloured by Q-score, with low Q-score regions boxed.

PDB	Res. (Å)	Phosphor RMSD (Å)	Backbone RMSD (Å)	Backbone Recall	Backbone Precision	Base Accuracy	Complete-ness
7S1G	2.48	**MA: 0.36** CR: 1.00 DT: 0.51	**MA: 0.48** CR: 1.99 DT: N/A	**MA: 0.96** CR: 0.68 DT: 0.86	**MA: 0.99** CR: 0.66 DT: 0.56	**MA: 0.80** CR: 0.55 DT: N/A	**MA: 0.77** CR: 0.37 DT: N/A
7ZJX	3.1	**MA: 0.48** CR: 1.24 DT: 0.86	**MA: 0.61** CR: 2.10 DT: N/A	**MA: 0.86** CR: 0.72 DT: 0.61	**MA: 0.94** CR: 0.60 DT: 0.38	**MA: 0.66** CR: 0.53 DT: N/A	**MA: 0.56** CA: 0.38 DT: N/A
7ZPQ	3.47	**MA: 0.42** CR: 1.14 DT: 0.56	**MA: 0.57** CR: 2.05 DT: N/A	**MA: 0.92** CR: 0.72 DT: 0.76	**MA: 0.98** CR: 0.63 DT: 0.42	**MA: 0.62** CR: 0.52 DT: N/A	**MA: 0.57** CR: 0.38 DT: N/A

Extended Data Table 2: Comparison with alternative approaches for the automated building of nucleotides. *MA* stands for ModelAngelo, *CR* for CryoREAD, and *DT* for DeepTracer. *Phosphor RMSD* is the root mean squared deviation of the predicted P atoms against that of the deposition. *Backbone RMSD* is similar but for the OP1, P, OP2, and O5' atoms of the nucleotide backbones. *Backbone recall* is the fraction of the deposited residues that were predicted to be within 3 Å (as measured between P atoms). *Backbone precision* is the fraction of predicted residues that have a corresponding residue present in the deposition within 3 Å. *Base accuracy* is the fraction of the predicted residues that have a correctly predicted nucleotide base. Finally, *completeness* is the fraction of deposited residues that were predicted with the correct base annotation. Numbers indicated in boldface are the best in each metric.

Results

ModelAngelo identifies protein chains that were not built by human experts

- To identify 'unidentified' chains,

  1. Run ModelAngelo without using its sequence module to calculate an initial atomic model with HMM profiles for all chains

  2. Search these profiles against the proteome

- Construct an input sequence file that included all chains in the deposited model plus the six newly identified chains and run ModelAngelo again

- For most sections of the unidentified chains, ModelAngelo built better models than those in the deposited structure (a)

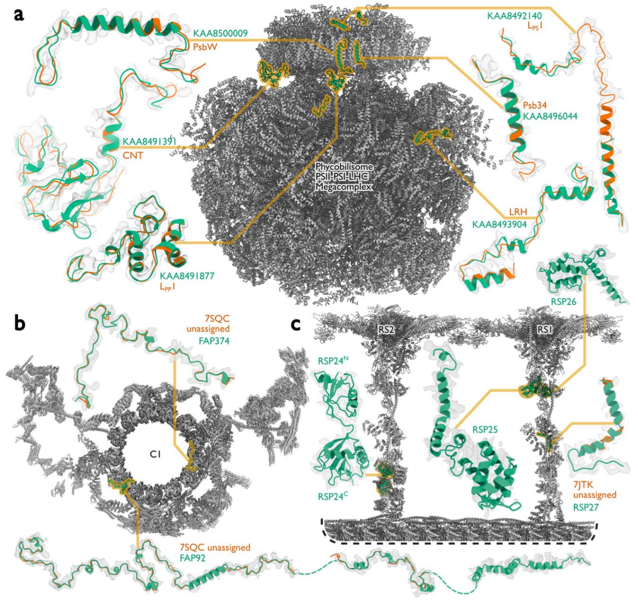


Figure 4: Examples of protein identification by ModelAngelo. **a**, The ModelAngelo model of the single-PBS-PSII-PSI-LHC's supercomplex (gray) showing the positions, models, and map densities of six newly identified proteins (green). Backbone traces in the deposited model (PDB ID: 7Y5E) are shown in orange. **b**, Atomic model of the central apparatus microtubule C1 showing the positions, models, and map densities of two newly identified proteins: FAP92 and FAP374. Orange cartoons represent poly(UNK) chains deposited in the original model (PDB ID: 7SQC). **c**, An atomic model of radial spokes 1 and 2 (RS1 and RS2) bound to a doublet microtubule (gray) showing the positions, models, and map densities of four proteins (RSP24-27, green) newly identified by ModelAngelo. Only RSP27 had a backbone trace in the deposited model (orange).

Results

ModelAngelo identifies protein chains that were not built by human experts

- All six unidentified proteins occur more than once in the cryo-EM map due to local pseudo-symmetry
    - Bootstrap weaker individual hits by cross-referencing their matches to the other instances
- For most sections of the unidentified chains, ModelAngelo built better models than those in the deposited structure (a)

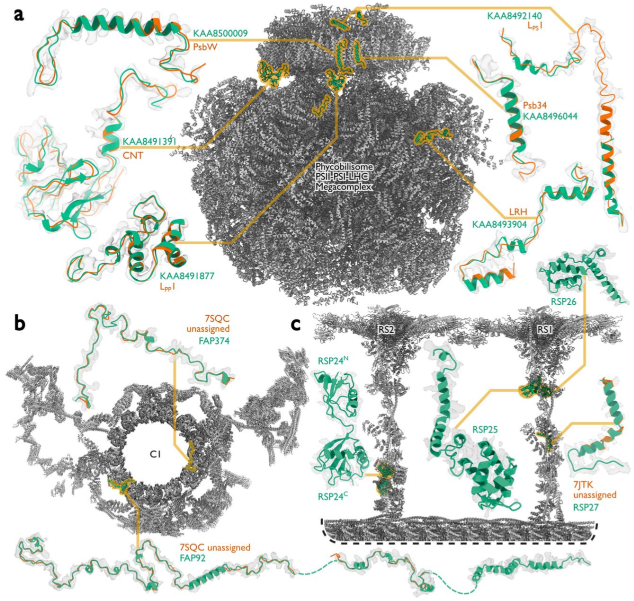
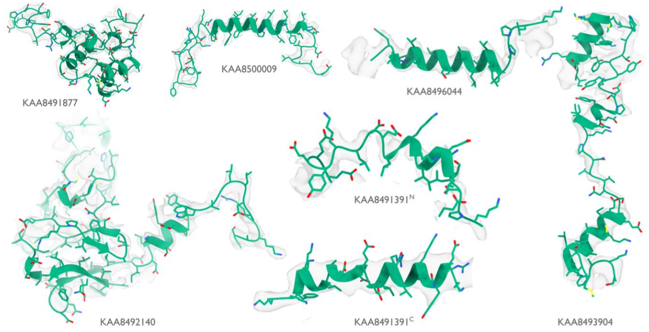


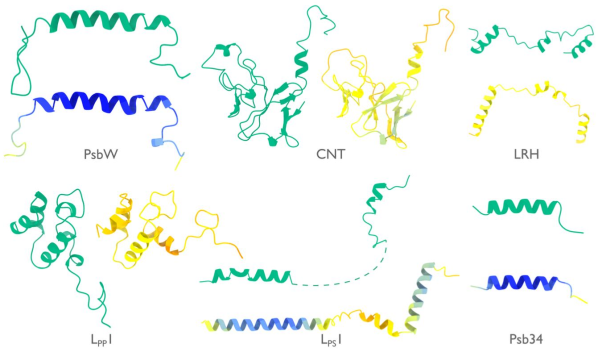
Figure 4: Examples of protein identification by ModelAngelo. a, The ModelAngelo model of the single-PBS-PSII-PSI-LHC's supercomplex (gray) showing the positions, models, and map densities of six newly identified proteins (green). Backbone traces in the deposited model (PDB ID: 7Y5E) are shown in orange. **b**, Atomic model of the central apparatus microtubule C1 showing the positions, models, and map densities of two newly identified proteins: FAP92 and FAP374. Orange cartoons represent poly(UNK) chains deposited in the original model (PDB ID: 7SQC). **c**, An atomic model of radial spokes 1 and 2 (RS1 and RS2) bound to a doublet microtubule (gray) showing the positions, models, and map densities of four proteins (RSP24-27, green) newly identified by ModelAngelo. Only RSP27 had a backbone trace in the deposited model (orange).

Results

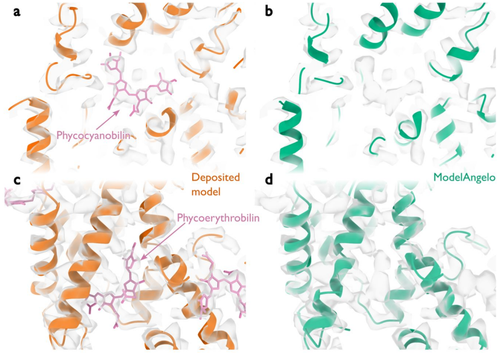
ModelAngelo identifies protein chains that were not built by human experts



Extended Data Figure 1: Identified proteins in the phycobilisome Atomic models built by ModelAngelo (green) for the six proteins that were identified by ModelAngelo. Side chain densities in the cryo-EM map (transparent grey) are in agreement with those of the atomic models.



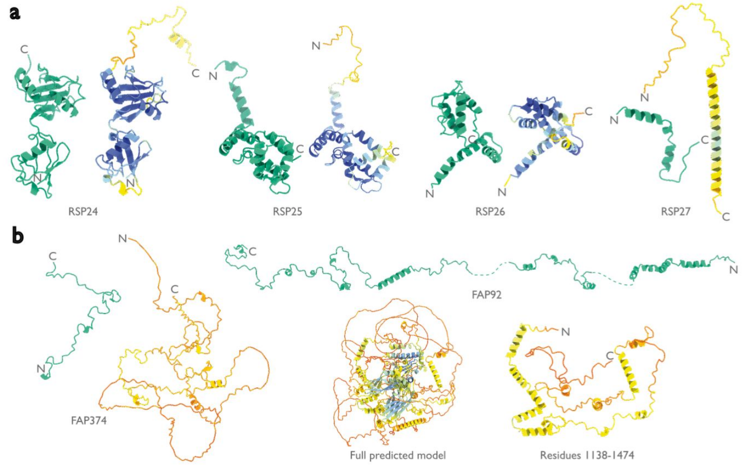
Extended Data Figure 2: Models by ModelAngelo and AlphaFold for identified proteins in the phycobilisome. Models built by ModelAngelo (green) are shown next to predictions of the corresponding sequences by AlphaFold (*15*) (coloured by AlphaFold's confidence from high in blue, to low in red).



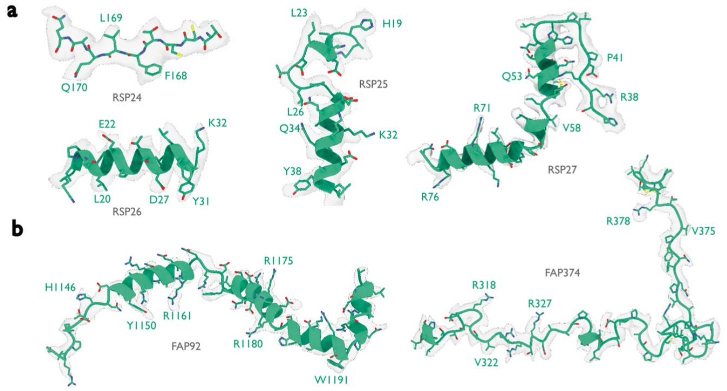
Extended Data Figure 3: Performance around cofactors in the phycobilisome. a, Cartoon representation of protein backbones (orange) and stick representation of a phycocyanobilin co-factor (pink) in the cryo-EM density (transparent grey) for the deposited phycobilisome structure. **b**, as in panel a, but for the model built by ModelAngelo (green). ModelAngelo leaves the cofactor density empty. **c**, **d**, as in panels a, b but for a phycoerythrobilin cofactor.

Results

ModelAngelo identifies protein chains that were not built by human experts



Extended Data Figure 4: Models by ModelAngelo and AlphaFold for identified proteins in the ciliary axoneme. Models built by ModelAngelo (green) are shown next to predictions of the corresponding sequences by AlphaFold (*15*) (coloured by AlphaFold's confidence from high in blue, to low in red). These are split between **a**, the radial spoke proteins, and **b**, the central apparatus microtubule proteins.



Extended Data Figure 5: Identified proteins in the ciliary axoneme Atomic models built by ModelAngelo (green) for the six proteins that were identified by ModelAngelo. Side chain densities in the cryo-EM map (transparent grey) are in agreement with those of the atomic models. These are split between **a**, the radial spoke proteins, and **b**, the central apparatus microtubule proteins.