RAYMORPH: LEARNING PROTEIN REDESIGN FROM STRUCTURAL PAIRS

Anonymous authors
Paper under double-blind review

ABSTRACT

Redesigning a protein to change its geometry while preserving its function is a central challenge in protein engineering. For instance, when a biosensor needs to be inserted into an internal loop of a target protein, its N- and C-termini need to be brought close together. In such cases, the desired change can be easier to illustrate with examples than explicit residue-level guidance. To address this challenge, we introduce RayMorph, a framework that learns protein redesign from pairs of related structures exhibiting a desired property change. The paired structural data specifies the transformation by example, allowing it to be learned and applied to new structures. RayMorph jointly modifies geometry, sequence, and length while preserving the source fold. We establish correspondence between unequal-length structures through geometry-guided dynamic programming and construct bridge trajectories that combine residue additions and deletions with changes in coordinates and amino-acid identities. We introduce a curated AlphaFoldDB dataset of structurally matched protein pairs with varying N–C distances and terminal accessibility. Among 488 proteins drawn from 50 Foldseek clusters excluded from training, 25.4% yield at least one candidate satisfying designability, source-fold similarity, and terminal-remodeling criteria after inverse folding and refolding.

1 INTRODUCTION

Protein redesign can be guided by explicit specifications, including functional motifs (Wang et al., 2022; Watson et al., 2023), fold classes (Geffner et al., 2025), or target properties (Xiong et al., 2026; Viliuga et al., 2025). An alternative is to specify the desired change through examples. In paired image-to-image translation, for instance, a model learns a mapping from corresponding input and target images (Isola et al., 2017; Liu et al., 2023). This approach is useful when the desired change involves many coupled features that are difficult to capture in a concise set of constraints.

A similar need arises in protein engineering. Some redesign objectives involve coordinated changes distributed across the structure, such as bringing termini closer together or converting a closed domain to an open conformation. These changes are easy to recognize in existing proteins but hard to prescribe residue by residue. Consider adapting a biosensor for insertion into an internal loop of a host protein when terminal fusion would disrupt the host's function. The sensor must connect to two nearby positions in the host backbone, which typically requires its N- and C-termini to be brought close together (< 10 Å). However, *simple proximity is insufficient*: the termini should be rigid rather than disordered, and a linker between them should not be occluded by other parts of the biosensor. In such cases, paired examples of similarly folded proteins that already satisfy these requirements offer a more natural specification than residue-level constraints.

The key conceptual advance of this work is to take an example-driven approach to protein redesign. We learn the redesign transformation from directed pairs of related structures that share a fold but differ in a target property, and apply it to new source proteins without a supplied target or edit plan.

Here we introduce RayMorph, which takes a variable-length structure–sequence co-generation approach to learn this redesign. Training examples connect pairs of valid structures with a shared fold but different properties, sequences, and lengths. Unlike generation initialized from simple priors, RayMorph redesigns starting from a structure. This setting extends the sequence-to-sequence learning problem explored by EvoFlows (Deutschmann et al., 2026) to joint structural and sequence transformation. However, learning from these pairs requires aligning unequal-length structures and building edit trajectories that change both residue count and identity.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

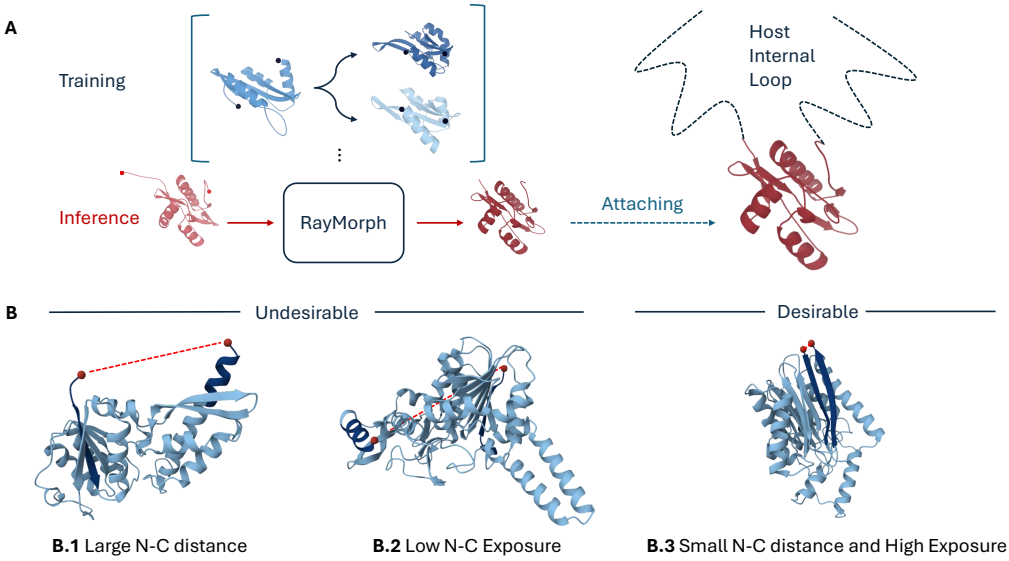


Figure 1: **A) RayMorph learns redesign from structural pairs and applies it to a new source**, with internal tag insertion as a motivating application. Geometry is schematic. **B)** In the N–C remodelling task, proteins with undesirable characteristics have large N-C distance (B.1) and low N-C exposure (i.e., a linker connecting the termini could be buried inside the protein), while desirable ones have small N–C distance and high exposure (B.2).

We first compute a residue-level correspondence between the starting and ending states. RayMorph uses dynamic programming to align residue geometry while preserving circular sequence order, accommodating differences in length and N–C placement. The correspondence enables the construction of a variable-length stochastic bridge that groups residue edits into stages while updating 3D coordinates and amino-acid identities. Stochastic noise perturbations allow the bridge to sample different trajectories from the same source. During generation, the model uses the current structure to decide which residues to add or remove together (Figure 1A).

In this paper, we focus on N–C remodeling as the first use case for this example-driven protein redesign approach. Beyond biosensor adaptation, N–C remodeling opens opportunities for protein optimization through circular permutation: connecting the existing termini with a short linker and opening new termini elsewhere in the protein to enhance functional control. To learn N–C remodeling from examples, we introduce a curated dataset of structurally similar protein pairs from AlphaFoldDB structural clusters that vary in N–C distance and exposure. We trim flexible ends, assess N–C separation and terminal accessibility, then select directed source–target pairs. These pairs provide supervision for N–C remodeling and a resource for studying structural variation relevant to circular permutation.(Figure 1B).

Our evaluation examines transfer to clusters excluded from training and whether redesigned structures can be realized by protein sequences. Among 488 source proteins drawn from 50 held-out Foldseek clusters, 42.0% yield a designable candidate and 25.4% attain joint success, which requires a designable candidate with N–C closure, connection exposure, and source-fold retention. These results support N–C remodeling as a step toward adapting folded biosensors for internal insertion. We demonstrate this on UnaG insertion into tubulin (TUB1A), where internal tagging is complicated by the multimeric assembly of TUB1A and molecular interactions at both its two termini.

Our key contributions are:

- **Formulation for example-driven variable-length structural redesign.** A paired generative framing that combines circular residue correspondence with a stochastic path for joint changes in geometry, sequence, and length.
- **A curated dataset of structural pairs with N–C distance variations.** Pairs of similarly structured proteins with differences in N–C distance and terminal accessibility, selected with insertability-aware processing for redesign and circular-permutation studies.
- **Transfer of N–C remodeling to unseen structural clusters.** An in-silico evaluation of terminal-distance reduction, fold retention, and designability in held-out clusters.

2 Related Work

Structural control and example-driven redesign. Protein generation can be guided by structural motifs or target properties. RFdiffusion and Genie 2 and 3 support motif scaffolding (Watson et al., 2023; Lin et al., 2024; 2026); Proteina conditions on fold class (Geffner et al., 2025), FliPS on flexibility profiles (Viliuga et al., 2025), and ProteinGuide on user-specified properties (Xiong et al., 2026). BindCraft optimizes binder sequences using folding models and explicit design losses (Pacesa et al., 2025). These methods steer design through conditions or objectives. RayMorph instead learns from directed structural pairs how coordinates, amino-acid identities, and residue count change while preserving a similar fold.

Variable-length generation and protein editing. Existing protein structure design models such as FrameDiff (Yim et al., 2023), FoldFlow (Bose et al., 2024), and MultiFlow (Campbell et al., 2024) generate backbones with a fixed residue count specified before sampling. Variable-length generative models relax this constraint by allowing sequence length to change during generation. Edit Flows (Havasi et al., 2025) and FlexMDM (Kim et al., 2025) enable such changes in discrete sequences, with related protein applications including SCISOR for sequence shrinking (Baron et al., 2026), DPLM-Evo for substitutions and indels through discrete diffusion (Wang et al., 2026), and EvoFlows for learning edits from evolutionary sequence pairs (Deutschmann et al., 2026). Beyond discrete sequences, TDDM extends variable-length generation to continuous structures (Campbell et al., 2023), while Branching Flows supports splits and deletions across discrete, continuous, and manifold-valued attributes (Billera et al., 2025). However, these methods formulate generation from simple prior distributions while RayMorph redesigns starting from protein structures.

Concurrent work. Recent methods apply variable-length generation to molecules (Morph (Franke et al., 2026); Expanding Flow Maps (Tang & Chatterjee, 2026)) and proteins (GPFlow (Cheng et al., 2026)). At submission, these methods have no public code releases. Their demonstrated tasks generate content from noise or an initially empty state, optionally retaining a supplied scaffold or motif as fixed conditioning. To our knowledge, no established method directly learns variable-length redesign from paired, complete protein structures and applies it to a new source without a supplied target structure or edit program.

3 Method

3.1 Problem setup and notation

We represent a protein $S = ((x^i, a^i))_{i=0}^{D-1}$ of length D by its ordered Cα coordinates $x^i \in \mathbb{R}^3$ and amino-acid identities $a^i \in \{1, \ldots, 20\}$. Directed training pairs (S_0, S_1) exemplify a desired property change between related structures. We seek to learn $p(S_1 \mid S_0)$ so that generation can begin from a new source protein without a supplied target or edit plan. The pairs provide endpoints, but no intermediate trajectory. We first establish residue correspondence across unequal-length proteins (Section 3.2). We then construct stochastic training states that combine coordinated length edits with coordinate and identity changes (Section 3.3). Finally, we generate redesigns by evolving the current protein state and predicting its edits (Section 3.4).

3.2 Pairing: residue mapping across unequal-length proteins

The first step is to associate source residues with target residues so that their coordinates and identities can be interpolated. Similar folds need not have matching residue indices, loop lengths, or N–C placements. We therefore align the structures geometrically and optimize a residue mapping that preserves circular sequence order. Circular indexing changes the chain start without reversing it.

After structural superposition and centering, the source–target residue cost is

$$C_{ij} = \|x_0^i - x_1^j\|_2. \tag{1}$$

Assigning nearest neighbors independently can violate sequence order. Instead, we use dynamic programming (DP) over cyclic shifts of the source. For expansion ($D_0 \leq D_1$), the DP partitions the target into nonempty contiguous groups G_i, one per source residue, minimizing $\sum_i \sum_{j \in G_i} C_{ij}$. Multiple target residues assigned to one source indicate where new residues must be introduced. For contraction ($D_0 > D_1$), the DP selects an ordered source subsequence matching the target; additional source residues are assigned to neighboring matched targets by proximity. The contraction cost is optimized over the primary matches. Recurrences and assignment rules are in Appendix A.

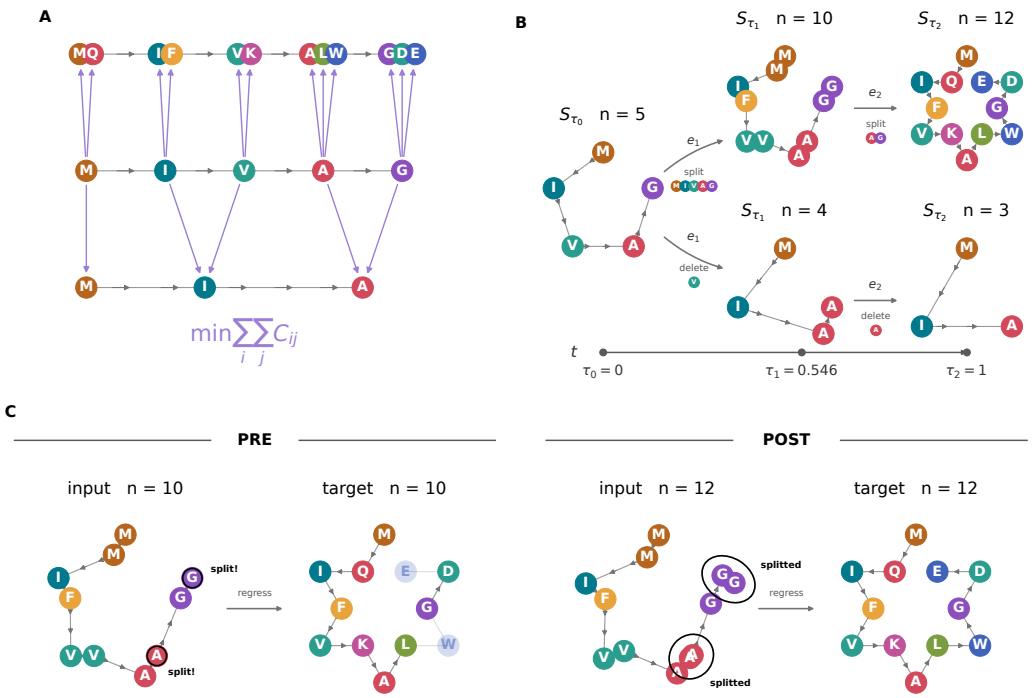


Figure 2: **Training-state construction for two toy pairs**. Geometry-guided, order-preserving correspondence groups determine coordinated splits or deletions; a shared draw selects the pre- or post-edit list before coordinate and identity interpolation. Intermediate geometry is schematic.

3.3 Constructing a bridge: edit stages and evolving attributes

The residue mapping establishes correspondence between the source and target; an edit plan specifies how residues are introduced or removed along the trajectory. Alignment-based paths in Edit Flows (Havasi et al., 2025), EvoFlows (Deutschmann et al., 2026), and Morph (Franke et al., 2026) can include both additions and deletions whose effects on total length cancel. To avoid these extra length-changing operations, we use a monotone plan: expansion uses only splits, contraction uses only deletions, and equal-length pairs require no length edits. Each plan therefore contains exactly $|D_1 - D_0|$ length-changing operations, the minimum needed to connect the endpoint lengths, grouped into synchronized edit stages.

The edit decisions are organized into J synchronized edit stages. Let $0 = \tau_0 < \tau_1 < \cdots < \tau_J = 1$ denote their boundaries, and let S_{τ_j} denote the protein state at boundary τ_j, containing D_{τ_j} residues. At stage j, the edit decisions e_j map S_{τ_j} to $S_{\tau_{j+1}}$ through

$$S_{\tau_0} \xrightarrow{e_0} S_{\tau_1} \xrightarrow{e_1} \cdots \xrightarrow{e_{J-1}} S_{\tau_J}, \qquad e_j \in \{\text{split}, \text{remain}, \text{delete}\}^{D_{\tau_j}}, \tag{2}$$

where $\tau_0 = 0$, $\tau_J = 1$, $D_{\tau_0} = D_0$, and $D_{\tau_J} = D_1$. A split inserts a copy immediately after its parent, deletion removes a residue, and remain retains it, which mirrors the construction of Branching Flows (Billera et al., 2025). Let $s^i_{\tau_j} = (x^i_{\tau_j}, a^i_{\tau_j})$ denote an individual residue state. For expansion, each residue $s^i_{\tau_j}$ in protein S_{τ_j} marked for splitting by e^i_j is duplicated to construct $S_{\tau_{j+1}}$, inserting a child immediately after its parent in sequence:

$$(\ldots, s^i_{\tau_j}, s^{i+1}_{\tau_j}, \ldots) \longmapsto (\ldots, s^i_{\tau_j}, \widetilde{s}^i, s^{i+1}_{\tau_j}, \ldots), \qquad \widetilde{s}^i = (x^i_{\tau_j}, a^i_{\tau_j}).$$

For contraction, each residue $s^i_{\tau_j}$ marked for deletion by e^i_j is removed:

$$(\ldots, s^{i-1}_{\tau_j}, s^i_{\tau_j}, s^{i+1}_{\tau_j}, \ldots) \longmapsto (\ldots, s^{i-1}_{\tau_j}, s^{i+1}_{\tau_j}, \ldots).$$

After either operation, residues are reindexed while preserving their relative order (Appendix B.1). Figure 2B illustrates two edit stages for both expansion and contraction. We sample their durations as $(\Delta_0, \ldots, \Delta_{J-1}) \sim \text{Dirichlet}(\alpha_\tau, \ldots, \alpha_\tau)$ and set $\tau_j = \sum_{k<j} \Delta_k$.

The coordinates and amino-acid identities of each S_{τ_j} are obtained by forward simulation from S_0, applying the preceding stages' edits and interpolating coordinates and sampling identities towards their assigned targets in S_1 at each successive boundary. Appendix B.1 details the forward rollout.

Edit Decision. To construct the training input S_t, we first draw $t \sim \mathcal{U}(0,1)$ and identify its stage $[\tau_j, \tau_{j+1}]$. Here, there are two possible starting states for interpolation. Before the edit, the starting state is $S^{\text{pre}}_{\tau_j} = S_{\tau_j}$; after the edit, it is $S^{\text{post}}_{\tau_j}$, obtained by applying e_j. To keep the stage's edits coordinated, a shared Bernoulli draw selects between these starting states for the entire residue list. Their coordinates and identities are then interpolated to obtain S_t:

$$b_t \sim \text{Bernoulli}(q_j(t)), \qquad q_j(t) = \frac{t - \tau_j}{\tau_{j+1} - \tau_j}. \tag{3}$$

The draw favors $S^{\text{pre}}_{\tau_j}$ when t is closer to τ_j, and $S^{\text{post}}_{\tau_j}$ when t is closer to τ_{j+1}. In the post-edit state, the edits have been applied and the resulting residues undergo further attribute refinement.

The starting state for interpolation and its corresponding edit supervision are

$$S_{\text{start}} = \begin{cases} S^{\text{pre}}_{\tau_j}, & b_t = 0, \\ S^{\text{post}}_{\tau_j}, & b_t = 1, \end{cases} \qquad e_t = \begin{cases} e_j, & b_t = 0, \\ (\text{remain})_{i=0}^{D_{\tau_{j+1}}-1}, & b_t = 1. \end{cases} \tag{4}$$

The post-edit state applies the stage's edit decisions: split **inserts a child that inherits its parent's** coordinates and amino-acid identity, whereas delete removes the selected residue. Equal-length pairs retain all-remain edit supervision throughout $t \in [0,1]$.

Coordinates. Because generation starts from a supplied protein, we introduce coordinate noise to support multiple redesigns from the same source. We then obtain the training input S_t by interpolating the coordinates and amino-acid identities of S_{start} towards their assigned targets in S_1. Given starting coordinates x_s from S_{start} and assigned target coordinates x_1 from S_1, we sample coordinates from the Brownian-bridge marginal

$$x_t \mid x_s, x_1 \sim \mathcal{N}\left(\mu_t,\ \sigma^2 \frac{(t-\tau_j)(1-t)}{1-\tau_j} I_3\right), \tag{5}$$

$$\beta_j(t) = \frac{t - \tau_j}{1 - \tau_j}, \qquad \mu_t = (1 - \beta_j(t))x_s + \beta_j(t)x_1. \tag{6}$$

Here σ sets the coordinate noise scale. This differs from methods that build outward from a simple prior or a fixed scaffold (Campbell et al., 2023; Billera et al., 2025; Franke et al., 2026; Cheng et al., 2026; Tang & Chatterjee, 2026); RayMorph starts from a complete protein and can transform all residues. Unlike edit progress $q_j(t)$, attribute progress $\beta_j(t)$ is measured toward the final endpoint, allowing coordinates and identities to continue evolving after a stage's edits.

Amino acid identities. Given starting amino-acid identities a_s from S_{start} and assigned target identities a_1 from S_1, we sample identities of S_t using the same $\beta_j(t)$:

$$a_t \mid a_s, a_1 \sim (1 - \beta_j(t))\delta_{a_s} + \beta_j(t)\delta_{a_1}. \tag{7}$$

Here δ_a denotes a point mass at identity a, so each residue retains its starting identity with probability $1 - \beta_j(t)$ and takes its assigned target identity with probability $\beta_j(t)$.

Given the sampled state S_t and time t, $f_\theta(S_t, t)$ predicts coordinate drift $\hat{v}_t$, amino-acid identity probabilities $\hat{p}_a$, edit probabilities $\hat{p}_e$, and N-terminal position probabilities $\hat{m}_t$. The losses are:

- **Coordinate loss $\mathcal{L}_{\text{coord}}$:** squared error $\|\hat{v}^i_t - v^{\star,i}_t\|_2^2$ against the conditional Brownian-bridge drift $v^{\star,i}_t = (x^i_1 - x^i_t)/(1-t)$.
- **Identity loss $\mathcal{L}_{\text{identity}}$:** cross-entropy $-\log \hat{p}^i_a(a^i_1)$ against its assigned endpoint identity.
- **Edit loss $\mathcal{L}_{\text{edit}}$:** class-balanced focal loss against each residue's branch-dependent label e^i_t,

  $$-\alpha_{e^i_t}\left(1 - \hat{p}^i_e(e^i_t)\right)^\gamma \log \hat{p}^i_e(e^i_t).$$

  We use $\alpha_{\text{remain}} = 1$ and $\alpha_{\text{split}} = \alpha_{\text{delete}} > 1$ to upweight length edits. The focal factor, with $\gamma = 2$, downweights confidently correct predictions.
- **N-terminal loss $\mathcal{L}_{\text{N}}$:** cross-entropy $-\log \hat{m}_t(m_t)$ against the current position m_t of the residue assigned to target residue zero. Its predecessor in the final circular order becomes the C-terminus. This loss is masked when that residue has not yet been introduced.

An auxiliary geometry loss $\mathcal{L}_{\text{geom}}$ matches local target pair distances and penalizes clashes in the predicted endpoint. The pair term supervises spatial neighbors, including residues distant along the sequence, to constrain local packing as well as chain connectivity. Appendix B.3 shows that

this substantially improves the fraction of validation proteins meeting both the bond-validity and clash-free criteria, over bond-only supervision with the clash penalty fixed. The objective is

$$\mathcal{L} = \mathcal{L}_{\text{coord}} + \mathcal{L}_{\text{identity}} + \mathcal{L}_{\text{edit}} + \mathcal{L}_{\text{N}} + \lambda_{\text{geom}}\mathcal{L}_{\text{geom}}. \tag{8}$$

Appendix B gives the architecture and training details, including the inverse-frequency choice of α in Appendix B.4.

3.4 Sampling and generation

For redesign from a complete source protein, RayMorph combines monotone training paths, a shared firing window, and continued geometric refinement to support coordinated length changes. A single source can admit several redesigns, each requiring a different combination of edits. Consider a starting sequence for which the training data specifies two targets with equal probability:

$$[A, B, C] \longrightarrow \begin{cases} [A, A, B, C], & \text{with probability } 0.5, \\ [A, B, B, C, C], & \text{with probability } 0.5. \end{cases}$$

The first target requires splitting A; the second requires splitting both B and C. These are coordinated choices, but sampling each residue's split independently loses that coordination. At the ambiguous source state, the marginal split probabilities are $(1/2, 1/2, 1/2)$, so an independent draw assigns probability $1/8$ to each edit combination. A partial split of either B or C can already steer subsequent refinement toward the second target. For example, if we discard empty and target-incompatible combinations from this draw and complete each remaining partial pattern toward its compatible target, splitting B, C, or both leads to the second target, whereas only splitting A leads to the first. This yields a $1:3$ sampling imbalance despite equal target frequencies.

We use a shared firing window to coordinate edit decisions before committing to length changes. When any residue qualifies for a split or deletion, a shared timer starts. During the window, coordinates and amino-acid identities continue to evolve, and edit predictions are recomputed at each step. At the end of the window, the currently qualifying edits are applied together (Appendix B.5, Algorithms 5–6). This delay gives the evolving geometry time to inform the joint edit pattern, allowing B and C to become ready together before either split is executed.

A newly inserted residue inherits its parent's coordinates and amino-acid identity. At subsequent steps, the same coordinate-drift prediction $\hat{v}_t$ and identity distribution $\hat{p}_a$ used for existing residues refine the new residue as well. Thus, no separate predictor is needed to initialize new residues.

Splitting by duplication follows Branching Flows (Billera et al., 2025), whereas Morph (Franke et al., 2026) learns a separate distribution for inserted-node attributes. Coordination also has precedent in localized Edit Flows (Havasi et al., 2025), which encourages edits near an earlier edit.

4 Experiments

On Foldseek-curated proteins, RayMorph produces designable redesigns with improved terminal geometry and retained source folds, including in held-out clusters and clusters where curation found no natural redesign examples (Table 1). We ablate noise, edit firing, and pair-distance supervision in Section 4.4. Finally, we remodel UnaG for internal insertion into tubulin (TUB1A), where the host's multimeric assembly makes terminal tagging impractical, and show that the predicted construct retains the folds of both components (Section 4.5).

4.1 Dataset Curation

We select directed pairs from predicted structures of naturally occurring proteins, with shorter N–C separation and an exposed terminal connection in the target while retaining similar folds. From 10,000 AlphaFold DB structural clusters built with Foldseek (Barrio-Hernandez et al., 2023), we curate examples within each cluster; the filters and selection procedure appear in Appendix C.1 and Figure 10.

Terminal geometry. We trim flexible tails and anchor the retained termini to helices or strands so that closure reflects structured regions. We measure their separation as $\text{NC} = \|x^0 - x^{D-1}\|_2$ and connection exposure as $\text{Exp} = \|m - \bar{x}\|_2 / R_g$, where m is the terminal midpoint, $\bar{x}$ the chain centroid, and R_g the radius of gyration. Higher Exp indicates a connection farther from the protein core (Figure 1B). We define $\text{dNC} = \text{NC}(S_1) - \text{NC}(S_0)$ and $\text{dExp} = \text{Exp}(S_1) - \text{Exp}(S_0)$; closure and increased exposure correspond to negative dNC and positive dExp.

Fold retention. We retain pairs with TM-align (Zhang & Skolnick, 2004; 2005) score ≥ 0.7, using fold similarity as a proxy for preserving function.

Natural redesign examples. Of the 10,000 clusters, 7,067 contain pairs demonstrating natural redesign; we denote these clusters NE+ (natural examples found). The remaining 2,933 are NE0 (no natural examples found). Both groups contain natural proteins; in NE0, our curation found no record of a redesign pair. Unseen proteins from these groups provide complementary tests of generalization. We held out 50 NE+ clusters for testing. From each remaining NE+ cluster, we selected up to 20 pairs, yielding 112,868 training pairs (Appendix C.2).

4.2 Evaluation

On toy and synthetic data, we confirmed the recovery of multiple targets from a shared source with length changes (Figure 3; Appendix D.1). **Foldseek-curated protein dataset.** All evaluation proteins are unseen during training. We use 488 proteins in each of five evaluation sets: two from NE+ clusters (seen and held-out) and three from NE0 clusters at increasing difficulty, defined by starting N–C distance and connection exposure. NE+ sources are prioritized for large N–C distance and low exposure; NE0 difficulty levels range from exposed, relatively close termini (easy) through exposed but distant (medium) to enclosed and distant (hard). Appendix C.3 gives the selection rules; Figure 11 shows the starting-geometry distributions.

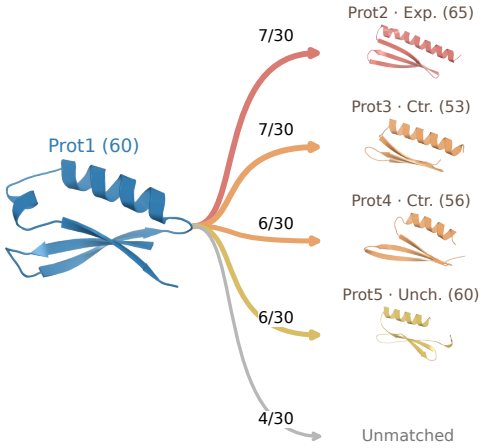


Figure 3: **Multiple outcomes from a shared source**.

Evaluation criteria. The $\mathrm{dNC} < -5$ Å criterion measures progress toward terminal closure rather than requiring an insertion-ready N–C distance below 10 Å. Likewise, $\mathrm{dExp} > 0.3$ measures increased connection exposure, while the alternative $\mathrm{Exp} > 0.8$ retains proteins that are already exposed without requiring a further increase. Figure 4A–B shows illustrative refolded RayMorph outputs from these evaluation sets.

	NE+: natural example pairs found		NE0: no natural pairs (all clusters unseen)		
Criterion	Seen clusters	Unseen clusters	Easy	Medium	Hard
Predicted structure quality					
(a) Bonds and clash-free $\geq 99\%$	84.2%	77.9%	85.0%	84.8%	75.2%
(b) (a) + TM > 0.7	83.4%	77.5%	84.4%	84.0%	74.0%
(c) (b) + dNC < -5 Å	64.5%	60.9%	61.9%	67.6%	59.8%
(d) (b) + (dExp > 0.3 or Exp > 0.8)	73.0%	67.8%	63.7%	62.7%	54.5%
(e) All criteria above	55.9%	54.1%	47.5%	54.1%	47.7%
Structure-refolded quality					
(a) Designability, scRMSD < 2 Å	41.2%	42.0%	32.0%	25.0%	23.2%
(b) (a) + TM > 0.7	41.0%	41.8%	31.8%	25.0%	22.7%
(c) (b) + dNC < -5 Å	24.0%	26.8%	16.6%	14.5%	13.5%
(d) (b) + (dExp > 0.3 or Exp > 0.8)	37.5%	37.3%	24.4%	18.4%	15.0%
(e) All criteria above	22.3%	25.4%	15.2%	12.3%	10.0%

Table 1: **RayMorph test-set performance on N–C remodeling**. Proteins with at least one of three generated samples satisfying each criterion. Changes in N–C distance and connection exposure score, and TM-scores, are measured relative to the source using predicted or refolded structures, respectively. In (d), Exp denotes the final exposure score.

4.3 Experimental Findings and Analysis

Refolding designability is more selective than predicted geometric validity. In Table 1, each redesigned backbone is evaluated using 8 sequences designed with ProteinMPNN (Dauparas et al.,

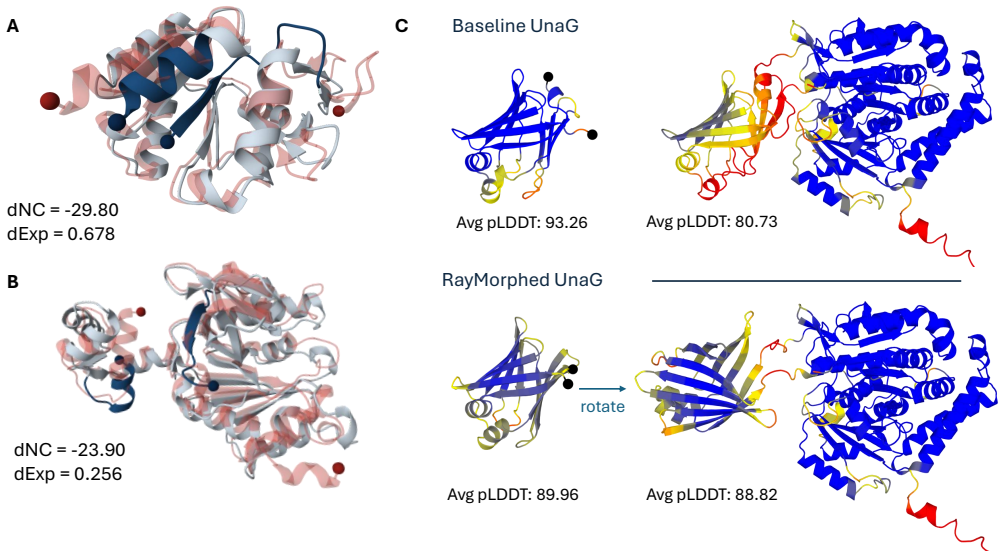


Figure 4: **A, B) Designable RayMorph outputs** from the validation sets illustrate N–C closure and increased connection exposure. Source proteins are shown in faded red and ESMFold-refolded outputs in blue and white; **spheres mark the termini**. **C) Baseline and RayMorph-remodeled UnaG, shown alone and inserted into TUB1A**. Structures are folded using Protenix-v2 (Zhang et al., 2026) and colored by pLDDT, with average scores shown below.

2022) and folded with ESMFold (Lin et al., 2023); we retain the refold with the lowest Cα scRMSD to the generated backbone. Designability is defined by minimum scRMSD $\leq$ 2 Å. Folding RayMorph's own generated identities gives lower success rates, reported separately in Appendix D.2.

RayMorph transfers N–C remodeling to unseen structural clusters, including clusters where curation found no natural redesign examples. RayMorph achieves higher joint refolded success on unseen clusters in the NE+ group (25.4%) than on clusters in the NE0 group (10.0–15.2%). One possible explanation is that favorable target structures in unseen clusters in the NE+ group resemble those represented during training, facilitating transfer. Clusters in the NE0 group yield no qualifying redesign pairs under our curation criteria and may require transformations less represented in training.

Joint redesign success decreases as the starting terminal geometry becomes more challenging. Within the NE0 group, larger NC and lower Exp are associated with reduced designability and lower success across all refolded criteria: joint success falls from 15.2% to 12.3% and 10.0% across the easy, medium, and hard sets, consistent with the difficulty stratification in Section 4.2.

Greater N–C closure trades off against source-fold retention. We examine this relationship in outputs from unseen clusters in the NE+ group. Figure 5 compares closure ($-$dNC) with the TM-score between each refolded structure and its source. Several outputs combine closure above 5 Å with TM-scores above 0.9, while the Pareto frontier declines at larger closure values. Further analyses of inference outputs on the validation sets and training ablations are provided in Appendix D. Additionally, we include inference-time hyperparameter effects in Appendix D.3.2, including their trade-offs among designability, within-source diversity, and edit extent.

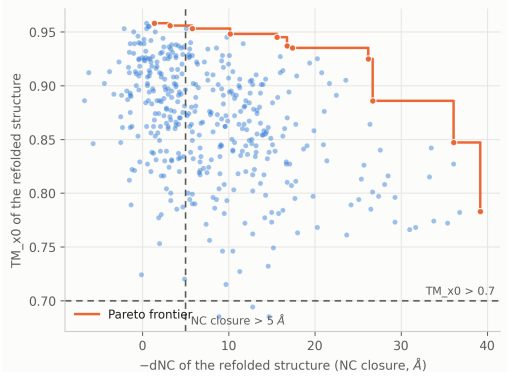


Figure 5: N–C closure and source-fold similarity after refolding, with Pareto frontier (orange) and evaluation thresholds (dashed).

4.4 Ablations: designability, diversity, and length change

Sampling noise and the edit-firing threshold control different aspects of redesign. We vary each at a fixed checkpoint using 6 held-out NE+ sources and 100 rollouts per source, with default $(\sigma, r) = (0.1, 5)$. Designability pools all 600 rollouts; diversity measures within-source pairwise $1 - \mathrm{TM}$ among designable refolds, and edit extent is mean absolute net length change. Full sweeps and metric definitions appear in Appendix D.3.2.

Noise controls the designability–diversity trade-off. At fixed $r = 5$, reducing σ from 0.1 to 0.01 raises designability from 31.3% to 83.3% while reducing diversity from 0.130 to 0.079. Mean absolute length change stays near 19 residues: less noise favors realizable structures without reducing the extent of length remodeling.

The firing threshold controls edit extent. At fixed $\sigma = 0.1$, lowering r from 5 to 3 permits more edits, increasing mean absolute length change from 19.3 to 31.0 residues. Designability falls to 12.2%, with only a modest diversity gain (0.130 to 0.146). Thus, noise and firing offer distinct trade-offs: the former changes structural variation, whereas the latter more strongly changes residue count. Edit-loss weighting sets the learned propensity to edit; its ablation primarily shifts length change, with a smaller effect on designability (Appendix B.4).

Pair-distance supervision improves geometric validity. With the clash penalty fixed, supervising target neighbors within 6 Å substantially increases the validation pass-rate for both 99% bond-validity and clash-free criteria compared with bond-only supervision (Figure 7; Appendix B.3).

4.5 N–C remodeling enables internal tagging of TUB1A

Tubulin hetero-polymerizes into long filaments, and both termini are engaged in molecular interfaces and unavailable for terminal fusion. Tagging tubulin with a fluorescent sensor therefore requires internal insertion: the sensor's N- and C-termini must accommodate the two ends of a selected insertion site (here, the loop at residues 35-50 in TUB1A) while limiting steric interference with the host. We apply RayMorph to remodel UnaG, a fluorescent protein for this geometry (Figure 4C).

The remodeled UnaG retains the source fold with slightly lower average pLDDT than the baseline (89.96 vs. 93.26), but the TUB1A–UnaG complex shows substantially higher predicted confidence after remodeling (88.82 vs. 80.73). The unmodified sensor, whose termini are distant and poorly exposed, appears strained when forced into the insertion geometry; remodeling alleviates this. Since no target structure or tagging-specific training data is required, this result suggests that example-driven N–C remodeling can generalize to practical biosensor-adaptation problems.

5 Conclusion

Some design objectives are difficult to specify explicitly. Bringing two termini close together, for example, requires more than a short N–C distance: the termini regions must also be rigid, and the path connecting them must not be occluded by the rest of the protein. Providing examples of proteins that satisfy these coupled requirements is more feasible than enumerating them as a joint objective. By learning from directed structural pairs, RayMorph absorbs such requirements implicitly. Our novel structure-to-structure variable-length generative framework applies wherever structural variation among related proteins illustrates a distributed change. It could be used to introduce, for instance, circular permutation or lock proteins into a specific conformation.

High-quality curated data is central to this approach. The gap between NE+ and NE0 performance (25.4% vs. 10.0–15.2% joint success) shows that structurally matched training examples in related clusters facilitate transfer; as predicted-structure databases grow in diversity, we expect both coverage and redesign quality to increase.

Limitations and future directions. A current limitation of RayMorph is sequence generation: RayMorph's own amino-acid identities yield substantially lower designability than inverse-folded sequences (Appendix D.2), indicating that the identity head does not yet capture the sequence–structure relationship learned by dedicated folding models. Distilling from structure-prediction networks that already model sequence–structure co-generation may narrow this gap. Overall redesign success also has room to improve, particularly on harder NE0 proteins, where it would benefit from richer training data. Beyond improving recovery, the example-driven formulation could extend to other redesign tasks; conditioning on continuous properties, such as the degree of terminal closure or exposure, could provide finer control over the outputs of the trained model.

CODE AVAILABILITY

Anonymized code is available at https://figshare.com/s/14cd6514183afc75ad28.

AI USE STATEMENT

We used generative AI tools to support research ideation and execution, retrieve and assess related work, and assist with code interpretation, visualization, and LaTeX preparation. AI also assisted in drafting and revising sections of the paper, refining mathematical notation, and developing and checking mathematical arguments. Writing assistance included polishing sentences, organizing the narrative, and shortening text to meet the page limit. The authors take full responsibility for the final content, claims, and results.

REFERENCES

Ethan Baron, Alan N. Amin, Ruben Weitzman, Debora Marks, and Andrew Gordon Wilson. A diffusion model to shrink proteins while maintaining their function. In *International Conference on Learning Representations (ICLR)*, 2026. arXiv:2511.07390.

Inigo Barrio-Hernandez, Jingi Yeo, Jürgen Jänes, Milot Mirdita, Cameron L. M. Gilchrist, Tanita Wein, Mihaly Varadi, Sameer Velankar, Pedro Beltrao, and Martin Steinegger. Clustering predicted structures at the scale of the known protein universe. *Nature*, 622:637–645, 2023. doi: 10.1038/s41586-023-06510-w.

Lukas Billera, Hedwig Nora Nordlinder, Jack Collier Ryder, Anton Oresten, Aron Stålmarck, Theodor Mosetti Björk, and Ben Murrell. Branching flows: Discrete, continuous, and manifold flow matching with splits and deletions. *arXiv preprint arXiv:2511.09465*, 2025. URL https://arxiv.org/abs/2511.09465.

Avishek Joey Bose, Tara Akhound-Sadegh, Guillaume Huguet, Kilian Fatras, Jarrid Rector-Brooks, Cheng-Hao Liu, Andrei Cristian Nica, Maksym Korablyov, Michael Bronstein, and Alexander Tong. SE(3)-stochastic flow matching for protein backbone generation. In *International Conference on Learning Representations (ICLR)*, 2024. Spotlight; arXiv:2310.02391.

Andrew Campbell, William Harvey, Christian Weilbach, Valentin De Bortoli, Tom Rainforth, and Arnaud Doucet. Trans-dimensional generative modeling via jump diffusion models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023. arXiv:2305.16261.

Andrew Campbell, Jason Yim, Regina Barzilay, Tom Rainforth, and Tommi Jaakkola. Generative flows on discrete state-spaces: Enabling multimodal flows with applications to protein co-design. In *Proceedings of the 41st International Conference on Machine Learning (ICML)*, 2024. arXiv:2402.04997.

Chaoran Cheng, Zhanghan Ni, Yanru Qu, Yuxin Chen, Ruihan Guo, Jiajun Fan, and Ge Liu. Variable-length generative protein design via generalized poisson flow. *arXiv preprint arXiv:2607.09039*, 2026. URL https://arxiv.org/abs/2607.09039.

Justas Dauparas, Ivan Anishchenko, Nathaniel Bennett, Hua Bai, Robert J. Ragotte, Lukas F. Milles, Basile I. M. Wicky, Alexis Courbet, Rob J. de Haas, Neville Bethel, Philip J. Y. Leung, Timothy F. Huddy, Sam Pellock, Doug Tischer, Frederick Chan, Brian Koepnick, Hannah Nguyen, Alex Kang, Banumathi Sankaran, Asim K. Bera, Neil P. King, and David Baker. Robust deep learning–based protein sequence design using ProteinMPNN. *Science*, 378(6615):49–56, 2022. doi: 10.1126/science.add2187.

Nicolas Deutschmann, Constance Ferragu, Jonathan D. Ziegler, Shayan Aziznejad, and Eli Bixby. EvoFlows: Evolutionary edit-based flow-matching for protein engineering. *arXiv preprint arXiv:2603.11703*, 2026. URL https://arxiv.org/abs/2603.11703. ICLR 2026 Workshop on Foundation Models for Science.

Malte Franke, Stefan P. Schmid, Zarko Ivkovic, Kjell Jorner, and Andreas Krause. Generative molecular morphing for flexible-size design via unbalanced optimal transport. *arXiv preprint arXiv:2606.07239*, 2026. URL https://arxiv.org/abs/2606.07239.

Tomas Geffner, Kieran Didi, Zuobai Zhang, Danny Reidenbach, Zhonglin Cao, Jason Yim, Mario Geiger, Christian Dallago, Emine Kucukbenli, Arash Vahdat, and Karsten Kreis. Proteina: Scaling flow-based protein structure generative models. In *International Conference on Learning Representations (ICLR)*, 2025. Oral presentation; arXiv:2503.00710.

Marton Havasi, Brian Karrer, Itai Gat, and Ricky T. Q. Chen. Edit flows: Flow matching with edit operations. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025. URL https://arxiv.org/abs/2506.09018. arXiv:2506.09018.

Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1125–1134, 2017.

Wolfgang Kabsch and Christian Sander. Dictionary of protein secondary structure: Pattern recognition of hydrogen-bonded and geometrical features. *Biopolymers*, 22(12):2577–2637, 1983. doi: 10.1002/bip.360221211.

Jaeyeon Kim, Lee Cheuk-Kit, Carles Domingo-Enrich, Yilun Du, Sham Kakade, Timothy Ngotiaoco, Sitan Chen, and Michael Albergo. Any-order flexible length masked diffusion. *arXiv preprint arXiv:2509.01025*, 2025. URL https://arxiv.org/abs/2509.01025.

Yeqing Lin, Minji Lee, Zhao Zhang, and Mohammed AlQuraishi. Out of many, one: Designing and scaffolding proteins at the scale of the structural universe with genie 2. *arXiv preprint arXiv:2405.15489*, 2024. URL https://arxiv.org/abs/2405.15489.

Yeqing Lin, Minji Lee, Aakarsh Vermani, Ellena Jiang, Sebastiaan De Cooman, Matej Spetko, and Mohammed AlQuraishi. Fast and ultra-capable protein design: Advancing the frontier through atomistic SE(3)-equivariance with genie 3. *bioRxiv*, 2026. doi: 10.64898/2026.05.01.722168.

Zeming Lin, Halil Akin, Roshan Rao, Brian Hie, Zhongkai Zhu, Wenting Lu, Nikita Smetanin, Robert Verkuil, Ori Kabeli, Yaniv Shmueli, Allan dos Santos Costa, Maryam Fazel-Zarandi, Tom Sercu, Salvatore Candido, and Alexander Rives. Evolutionary-scale prediction of atomic-level protein structure with a language model. *Science*, 379(6637):1123–1130, 2023. doi: 10.1126/science.ade2574.

Guan-Horng Liu, Arash Vahdat, De-An Huang, Evangelos A Theodorou, Weili Nie, and Anima Anandkumar. I˙2 sb: Image-to-image schr\" odinger bridge. *arXiv preprint arXiv:2302.05872*, 2023.

Martin Pacesa, Lennart Nickel, Christian Schellhaas, et al. One-shot design of functional protein binders with BindCraft. *Nature*, 646:483–492, 2025. doi: 10.1038/s41586-025-09429-6.

Sophia Tang and Pranam Chatterjee. Expanding flow maps. *arXiv preprint arXiv:2607.21585*, 2026. URL https://arxiv.org/abs/2607.21585.

Vsevolod Viliuga, Leif Seute, Nicolas Wolf, Simon Wagner, Arne Elofsson, Jan Stühmer, and Frauke Gräter. Flexibility-conditioned protein structure design with flow matching. In *Proceedings of the 42nd International Conference on Machine Learning (ICML)*, 2025. arXiv:2508.18211.

Jue Wang, Sidney Lisanza, David Juergens, Doug Tischer, Joseph L. Watson, Karla M. Castro, Robert Ragotte, Amijai Saragovi, Lukas F. Milles, Minkyung Baek, Ivan Anishchenko, Wei Yang, Derrick R. Hicks, Marc Expòsit, Thomas Schlichthaerle, Jung-Ho Chun, Justas Dauparas, Nathaniel Bennett, Basile I. M. Wicky, Andrew Muenks, Frank DiMaio, Bruno Correia, Sergey Ovchinnikov, and David Baker. Scaffolding protein functional sites using deep learning. *Science*, 377(6604):387–394, 2022. doi: 10.1126/science.abn2100.

Xinyou Wang, Liang Hong, Jiasheng Ye, Zaixiang Zheng, Yu Li, Shujian Huang, and Quanquan Gu. Towards a generative protein evolution machine with DPLM-Evo. In *Proceedings of the 43rd International Conference on Machine Learning (ICML)*, 2026. arXiv:2605.00182.

Joseph L. Watson, David Juergens, Nathaniel R. Bennett, Brian L. Trippe, Jason Yim, Helen E. Eisenach, Woody Ahern, Andrew J. Borst, Robert J. Ragotte, Lukas F. Milles, Basile I. M. Wicky, Nikita Hanikel, Samuel J. Pellock, Alexis Courbet, William Sheffler, Jue Wang, Preetham Venkatesh, Isaac Sappington, Susana Vázquez Torres, Anna Lauko, Valentin De Bortoli, Emile Mathieu, Sergey Ovchinnikov, Regina Barzilay, Tommi S. Jaakkola, Frank DiMaio, Minkyung Baek, and David Baker. De novo design of protein structure and function with RFdiffusion. *Nature*, 620(7976):1089–1100, 2023. doi: 10.1038/s41586-023-06415-8.

Junhao Xiong, Ishan Gaur, Maria Lukarska, Hunter Nisonoff, Luke M Oltrogge, David F Savage, and Jennifer Listgarten. Property guidance for protein sequence generative models with protein-guide. *Nature Biotechnology*, pp. 1–13, 2026.

Jason Yim, Brian L. Trippe, Valentin De Bortoli, Emile Mathieu, Arnaud Doucet, Regina Barzilay, and Tommi Jaakkola. SE(3) diffusion model with application to protein backbone generation. In *Proceedings of the 40th International Conference on Machine Learning (ICML)*, 2023. arXiv:2302.02277.

Yang Zhang and Jeffrey Skolnick. Scoring function for automated assessment of protein structure template quality. *Proteins: Structure, Function, and Bioinformatics*, 57(4):702–710, 2004. doi: 10.1002/prot.20264.

Yang Zhang and Jeffrey Skolnick. TM-align: a protein structure alignment algorithm based on the TM-score. *Nucleic Acids Research*, 33(7):2302–2309, 2005. doi: 10.1093/nar/gki524.

Yuxuan Zhang, Chengyue Gong, Jinyuan Sun, Jiaqi Guan, Milong Ren, Song Xue, Hanyu Zhang, Wenzhi Ma, Zhenyu Liu, Xinshi Chen, et al. Protenix-v2: Broadening the reach of structure prediction and biomolecular design. *bioRxiv*, pp. 2026–04, 2026.

APPENDIX INDEX

A. Pairing . . . 14
A.1 Order-preserving residue assignment . . . 14
B. Training-state construction and implementation . . . 15
B.1 Staged trajectory construction . . . 15
B.1.1 Stage-wise edit plans from correspondence groups . . . 15
B.1.2 Sampling edit-stage boundaries . . . 15
B.1.3 Forward simulation at stage boundaries . . . 16
B.2 Model architecture and objective implementation . . . 16
B.3 Geometry loss . . . 17
B.4 Edit loss . . . 18
B.5 Sampling and synchronized firing . . . 19
C. Data curation, training, and evaluation sets . . . 19
C.1 Foldseek cluster filtering and selection . . . 19
C.2 Training set . . . 20
C.3 Evaluation sets . . . 20
D. Additional analyses and ablations . . . 21
D.1 Toy and protein branching tasks . . . 21
D.2 Complete validation results . . . 22
D.3 Validation-output analyses . . . 23
D.3.1 Training-pair and generated-output distributions . . . 23
D.3.2 Inference-time sampling controls . . . 23

A PAIRING

A.1 ORDER-PRESERVING RESIDUE ASSIGNMENT

For each circular source offset o, we form $C_{ij}^{(o)} = \|x_0^{(i+o) \bmod D_0} - x_1^j\|_2$ and retain the assignment with minimum total cost. The target sequence direction is preserved. The two length regimes use the following procedures.

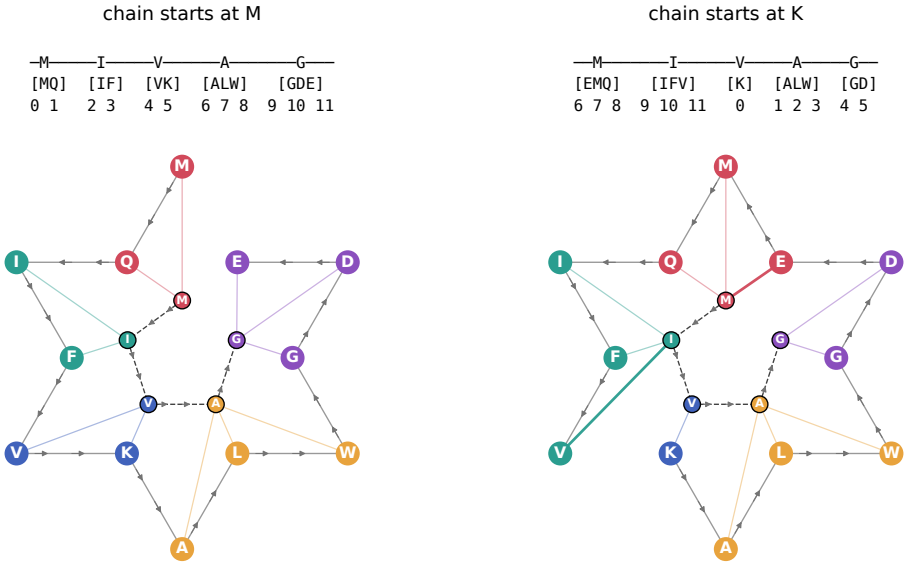


Figure 6: Order-preserving correspondence under circular reindexing. The assignment minimizes geometric mismatch while grouping residues in sequence order. Left: source and target residues in their original indexing. Right: the target ordering is cyclically rotated so that geometrically matched groups align with the source; the traversal direction within the target sequence is unchanged. Colors denote assigned groups.

Algorithm 1 Expansion correspondence ($D_0 \le D_1$)

Require: Source coordinates x_0, target coordinates x_1

// $F_{i,j}$: minimum cost through target j, using sources $0, \ldots, i$.

for each source offset o **do**
  Initialize F to $+\infty$, including $F_{-1,j}$; $F_{0,0} \leftarrow C_{0,0}^{(o)}$
  for $j = 1, \ldots, D_1 - 1$ and $i = 0, \ldots, D_0 - 1$ **do**
    // Extend the current source's group, or start the next group.
    $F_{i,j} \leftarrow C_{i,j}^{(o)} + \min\{F_{i,j-1}, F_{i-1,j-1}\}$
  end for
  Trace back from F_{D_0-1,D_1-1} to obtain nonempty contiguous groups
  Retain the lowest-cost grouping G
end for
return source-to-target groups G

Algorithm 2 Contraction correspondence ($D_0 > D_1$)

Require: Source coordinates x_0, target coordinates x_1
// $H_{i,j}$: minimum cost of matching the first j targets using the first i sources.
for each source offset o **do**
  $H_{i,0} \leftarrow 0$; initialize all other states to $+\infty$
  for $i = 1, \ldots, D_0$ and $j = 1, \ldots, D_1$ **do**
    // Skip this source, or match it to the next target.
    $H_{i,j} \leftarrow \min\{H_{i-1,j}, H_{i-1,j-1} + C^{(o)}_{i-1,j-1}\}$
  end for
  Trace back H_{D_0,D_1} to obtain one ordered primary source match per target residue
  Retain the primary matches with the lowest cost H_{D_0,D_1} as groups G
end for
for each source residue i without a primary match, in original indexing **do**
  Let j_ℓ, j_r be the targets of its nearest primary sources on either side
  // At either end, use the sole available neighbor.
  $G_i \leftarrow [\arg\min_{j \in \{j_\ell, j_r\}} \|x^i_0 - x^j_1\|_2]$
end for
return source-to-target groups G

B Training-state construction and implementation

B.1 Staged trajectory construction

We detail the deterministic staged construction and forward rollout used to form training states from each paired protein.

B.1.1 Stage-wise edit plans from correspondence groups

We convert each correspondence grouping into an ordered sequence of split or deletion stages that changes the source residue count monotonically toward the target.

Algorithm 3 Constructing a staged edit plan

Require: Correspondence groups G, source length D_0, target length D_1
if $D_0 < D_1$ **then**
  $A_0 \leftarrow$ the first target lineage in each source group
  while $|A_j| < D_1$ **do**
    Insert the midpoint of each unresolved circular gap in A_j
    Assign each inserted lineage to the left lineage of its gap; mark those parents SPLIT
    Record the resulting ordered list as A_{j+1}
  end while
else if $D_0 > D_1$ **then**
  $A_0 \leftarrow (0, \ldots, D_0 - 1)$; $R \leftarrow$ the first source lineage assigned to each target residue
  while $A_j \neq R$ **do**
    $E_j \leftarrow$ every other lineage in $A_j \setminus R$; mark E_j DELETE
    $A_{j+1} \leftarrow A_j \setminus E_j$
  end while
else
  $A_0 \leftarrow (0, \ldots, D_0 - 1)$ and use no length-changing stage
end if
return active lists $A_{0:J}$ and stage decisions $e_{0:J-1}$

B.1.2 Sampling edit-stage boundaries

For a plan with $J > 0$ edit stages, we resample the durations for each training state as

$$(\Delta_0, \ldots, \Delta_{J-1}) \sim \text{Dirichlet}(\alpha_\tau \mathbf{1}_J), \qquad \tau_0 = 0, \quad \tau_{j+1} = \sum_{k=0}^{j} \Delta_k, \tag{9}$$

so that $\tau_J = 1$. The implementation sets $\alpha_\tau = 50^{1-\rho}0.1^\rho$, with default spread $\rho = 0$ and hence $\alpha_\tau = 50$. Thus $\mathbb{E}[\Delta_j] = 1/J$ and $\mathrm{Var}(\Delta_j) = (J-1)/[J^2(J\alpha_\tau + 1)]$: stages are placed nearly uniformly across time with small random deviations. For $J = 0$, we use the sole no-edit interval $[0, 1]$.

B.1.3 Forward simulation at stage boundaries

We propagate the attributes of active residue lineages through completed stages to obtain the state at each boundary. We initialize from S_0. At completed stage k, a new child first copies its parent's current attributes, while deleted lineages are removed. On the resulting active list, coordinates and identities progress toward their assigned entries in S_1 with

$$\beta_k = \frac{\tau_{k+1} - \tau_k}{1 - \tau_k}, \tag{10}$$

$$x_{\tau_{k+1}} = (1 - \beta_k)x_{\tau_k} + \beta_k x_1, \tag{11}$$

$$a_{\tau_{k+1}} \sim (1 - \beta_k)\,\delta_{a_{\tau_k}} + \beta_k\,\delta_{a_1}. \tag{12}$$

Thus coordinates use the deterministic bridge mean at the boundaries, whereas identities are sampled from the corresponding two-point mixture.

Algorithm 4 Forward simulation to stage boundary τ_j

Require: Source S_0, target S_1, active lineage lists $A_{0:J}$, stage decisions $e_{0:J-1}$, boundaries $\tau_{0:J}$

1: Initialize the active state from S_0
2: **for** $k = 0, \ldots, j-1$ **do**
3: Apply e_k: each child copies its current parent attributes; remove deleted lineages
4: $\beta_k \leftarrow (\tau_{k+1} - \tau_k)/(1 - \tau_k)$
5: Update active coordinates and identities toward their assigned entries in S_1 using Equation (12)
6: **end for**
7: **return** S_{τ_j} on active list A_j

B.2 Model architecture and objective implementation

We specify the input representations, shared network, prediction heads, and loss implementation used for training. Loss contributions are weighted by the inverse number of training pairs in each cluster.

The network is a standard pre-normalized transformer over residues, with two additions for geometry. Random Fourier features encode each residue's coordinates as input, and a Gaussian radial basis expansion of the pairwise distances $\|x_t^d - x_t^{d'}\|_2$ biases the attention logits. Four linear heads read the final representation and output the coordinate drift $\hat{v}_t$, the amino-acid identity probabilities $\hat{p}_a$, the edit probabilities $\hat{p}_e$, and the N-terminal position probabilities $\hat{m}_t$. The distance bias is recomputed in every block from the current coordinates, with a separate learned bias per attention head.

Neither the Fourier features nor the linear vector heads are rotation equivariant, so the network must learn this invariance from data. Following Proteina (Geffner et al., 2025), we draw a uniform random rotation $R \in SO(3)$ for each training sample and apply it to the sampled coordinates x_t and to the endpoint x_1 together, after centering and scaling. Rotating both leaves the supervised drift equal to $Rv_t^\star$, so the target transforms with the input and the network sees every orientation of the same pair across training.

Table 2 lists the configuration of the reported run.

Table 2: Configuration of the reported training run.

Setting	Value
Architecture	
Blocks / hidden width / heads / feedforward	8 / 768 / 12 / 3072
Parameters	61M
Fourier channels per axis / bandwidth b	64 / 0.3
Radial bases / maximum distance	64 / 2.0 model units (20 Å)

Interpolation	
Coordinate scale s_x / bridge noise σ	0.1 / 0.1
Dirichlet concentration α_τ	50
Objective	
Edit class weight α / focal exponent γ	160 / 2
Geometry weight λ_{geom}	500
Optimization	
Optimizer / learning rate / weight decay	AdamW / 3×10^{-4} / 10^{-4}
Warmup / schedule / gradient clipping	15k steps / constant / 0.5
Batch size / training steps / GPUs	16,384 residues / 10^6 / 8
Training pairs / validation pairs	112,868 / 7,017
Generation	
Steps K / edit stages / firing window w	200 / 10 / 20
Threshold ratio r / noise σ / identity temperature	5 / 0.1 / 1

B.3 Geometry loss

We define the auxiliary geometry loss and state the setting used in the main experiments; its ablation is reported with the additional analyses.

The loss scores the endpoint implied by the predicted drift, $\hat{x}_1^d = x_t^d + (1-t)\hat{v}_t^d$, against the assigned target x_1^d, and is restricted to residues that survive to S_1. Writing $r_{dd'} = \|x_1^d - x_1^{d'}\|_2$ for target distances and $\hat{r}_{dd'}$ for the predicted ones,

$$\mathcal{L}_{\text{geom}} = \left\langle (\hat{r}_{dd'} - r_{dd'})^2 \right\rangle_{\mathcal{P}} + \left\langle \max(0,\, 3.5 - \hat{r}_{dd'})^2 \right\rangle_{\mathcal{Q}}, \qquad \mathcal{Q} = \{d' - d \geq 3,\ r_{dd'} > 3.5\},$$

where $\langle \cdot \rangle$ averages over the indicated pair set in Å. The clash term $\mathcal{Q}$ penalizes predicted contacts absent from the target. For the matching set $\mathcal{P}$ we compare two choices,

$$\mathcal{P}_{\text{bond}} = \{d' = d+1\}, \qquad \text{spacing along the chain,}$$
$$\mathcal{P}_{\text{pair}} = \{d < d',\ r_{dd'} < 6\}, \qquad \text{every neighbor the target holds,}$$

where $\mathcal{P}_{\text{pair}}$ also constrains residues close in space but distant in sequence. Bonds are the $d' = d+1$ slice of $\mathcal{P}_{\text{pair}}$, so the second choice widens the term rather than replacing it.

Figure 7 compares the two under otherwise identical training. Bond accuracy is the same in both runs, while clash-free rates separate: the bond term leaves a plateau near 90%, since its hinge gives no gradient beyond 3.5 Å and cannot hold packing in place. We therefore use $\mathcal{P}_{\text{pair}}$ in all reported runs.

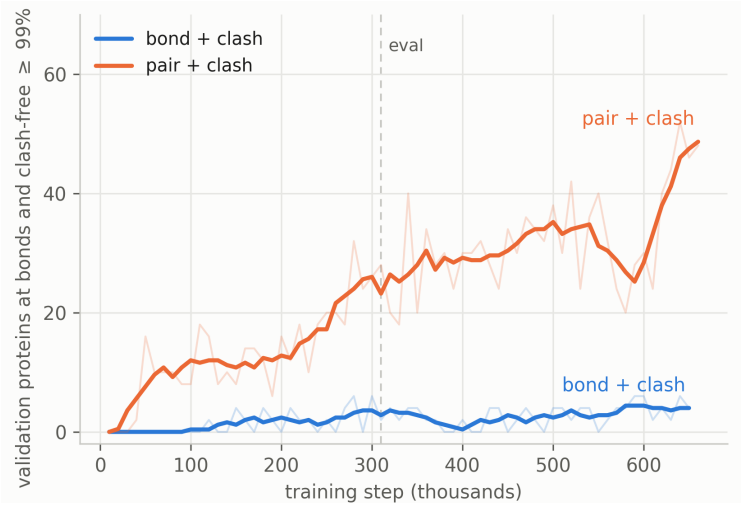


Figure 7: Geometry loss ablation. Fraction of validation proteins whose generations reach 99% valid bonds and 99% clash-free, for the bond and pair matching sets with identical clash terms.

The weight λ_{geom} controls how quickly this geometry is learned. Comparing $\lambda_{\text{geom}} \in \{0, 10, 500\}$ under the pair loss, 500 reaches 98% mean bond validity by 50k steps and 97% clash-free by 130k,

whereas 0 and 10 are still climbing at 80% and 93% bond validity at the same step (Figure 8). We use $\lambda_{\text{geom}} = 500$.



Figure 8: Geometry weight ablation. Mean bond validity and clash-free percentage over 50 validation proteins for $\lambda_{\text{geom}} \in \{0, 10, 500\}$ under the pair loss. One seed per weight.

B.4 Edit loss

The class weight α is set by inverse frequency rather than swept. Sampling training states from the paired data gives 0.61% deletions, 0.62% splits, and 98.78% remains, so equalizing each class's contribution requires $\alpha \approx 160$; the same measurement on a second split gives 147 and 156, so one global constant suffices. Deletions and splits share a single weight because they are close in frequency and disjoint by direction: a contracting pair never splits and an expanding pair never deletes.

The weight controls how far generation moves the length. At $\alpha = 50$ the model barely edits, leaving 33% of rollouts at the source length with mean $|\Delta\text{len}| = 2.4$; at $\alpha = 500$ it edits aggressively, reaching 11.2 residues with only 3% unchanged (Figure 9). Designability is decreasing slowly across this range (29%, 26%, 24%), within the noise of 50 proteins $\times$ 3 rollouts, so we use the inverse-frequency value $\alpha = 160$ to balance both edit extent and designability.

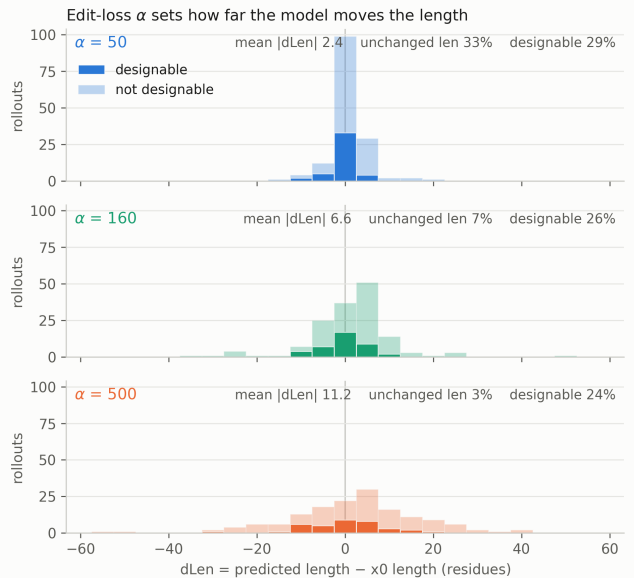


Figure 9: Edit loss class weight. Distribution of the length change between each rollout and its source for $\alpha \in \{50, 160, 500\}$, split by whether the output is designable. One seed per weight.

B.5 Sampling and synchronized firing

Algorithms 5 and 6 give the sampling updates and shared firing-window rule described in Section 3.4.

Algorithm 5 Sampling with synchronized edits

Require: S_0, f_θ, steps K, noise σ, threshold r, window w
Ensure: Redesigned protein $\hat{S}_1$

$(x, a) \leftarrow S_0$, $c \leftarrow 0$; $\Delta t \leftarrow 1/K$
for $k = 0, \dots, K-1$ **do**
  $t \leftarrow k\Delta t$
  $(\hat{v}, \hat{p}_a, \hat{p}_e, \hat{m}) \leftarrow f_\theta((x, a), t)$
  $\bar{x} \leftarrow x + \Delta t\, \hat{v} + \sigma\sqrt{\Delta t}\,\epsilon, \quad \epsilon \sim \mathcal{N}(0, I_{3D_t})$
  $q_a \leftarrow \left(1 - \frac{\Delta t}{1-t}\right)\delta_a + \frac{\Delta t}{1-t}\hat{p}_a$
  $\bar{a} \sim \mathrm{Cat}(q_a)$ ▷ per residue
  $(\bar{e}, c) \leftarrow \mathrm{sync_fire}_{r,w}(\hat{p}_e, c)$
  $(x, a) \leftarrow \mathrm{apply_edit}((\bar{x}, \bar{a}), \bar{e})$
end for
$\hat{S}_1 \leftarrow \mathrm{reorder}(\hat{m}, (x, a))$
return $\hat{S}_1$

Algorithm 6 Synchronized firing (sync_fire)

Require: Edit probabilities $\hat{p}_e$, counter c, threshold r, window w
Ensure: Edit decisions $\bar{e}$, updated counter c

$(p_{\mathrm{del}}, p_{\mathrm{rem}}, p_{\mathrm{spl}}) \leftarrow \hat{p}_e$
Assign candidate edits for each residue:

$$\bar{e} = \begin{cases} \mathrm{spl}, & p_{\mathrm{spl}} > r \cdot p_{\mathrm{rem}} \ \wedge \ p_{\mathrm{spl}} > p_{\mathrm{del}}, \\ \mathrm{del}, & p_{\mathrm{del}} > r \cdot p_{\mathrm{rem}} \ \wedge \ p_{\mathrm{del}} > p_{\mathrm{spl}}, \\ \mathrm{rem}, & \text{otherwise.} \end{cases}$$

if $c > 0$ or any residue has $\bar{e} \neq \mathrm{rem}$ **then**
  $c \leftarrow c + 1$
end if
if $c \geq w$ **then** ▷ fire!
  $c \leftarrow 0$
else ▷ hold
  $\bar{e} \leftarrow \mathrm{rem}$ for all residues
end if
return $(\bar{e}, c)$

C Data curation, training, and evaluation sets

We describe how Foldseek clusters are filtered into directed redesign pairs, then used to construct the training and evaluation sets.

C.1 FoldSeek clusters filtering and selection

We filter individual proteins, retain structurally compatible directed candidates, and rank the resulting pairs within each cluster. Across the $10{,}000$ selected clusters, $7{,}067$ yield at least one qualifying pair and form the NE+ group; the remaining $2{,}933$ form the NE0 group. Figure 10 summarizes the filtering waterfall.

- **Per-protein filtering.**
  1. **pLDDT:** retain pLDDT-supported termini and a core with mean pLDDT ≥ 60.
  2. **DSSP (Kabsch & Sander, 1983):** anchor each terminus to a helix or strand of at least four residues, with at most 20% trimming per end.
  3. **PAE:** require terminal PAE below 15 Å.
- **Per-candidate-pair filtering.** We form directed candidates with $\mathrm{NC}(S_0) > \mathrm{NC}(S_1)$, then apply:
  1. **Trim symmetry:** discard the 5% with the largest terminal-trim mismatch. This prevents from over-trimming the floppy regions of the N–C terminal regions.
  2. **Exposure:** require an exposed target, $\mathrm{Exp}(S_1) \geq 0.73$. N–C remodelling is moot if the target protein N–C terminus is buried.
  3. **Structural similarity:** require TM-score (Zhang & Skolnick, 2004) ≥ 0.7 to retain pairs with similar folds.
  4. **Overhang:** require at most 20 aligned terminal-overhang residues. This is to screen out target proteins that are a structural "subset" of the source proteins, normally by an additional domain.

  At most $5{,}000$ candidates per cluster undergo TM-align (Zhang & Skolnick, 2005).
- **Candidate-pair ranking.** Within each cluster, we apply the following ordered criteria:
  1. Each ranking pass uses a protein at most once as a source and once as a target.
  2. Rank by
  $$q(S_0, S_1) = \tfrac{1}{2} z(\mathrm{dExp}) + \tfrac{1}{2} z(-\mathrm{dNC}), \tag{13}$$
  where z is the within-cluster z-score. This term balances the contribution of 2 terms, ranking candidate pairs that can maximize both.

3. Prefer terminal-extension classes TIER3, TIER2, then TIER1.
4. Prefer $\mathrm{NC}(S_0) > 10\,\text{Å}$ and $\mathrm{NC}(S_1) < 10\,\text{Å}$.

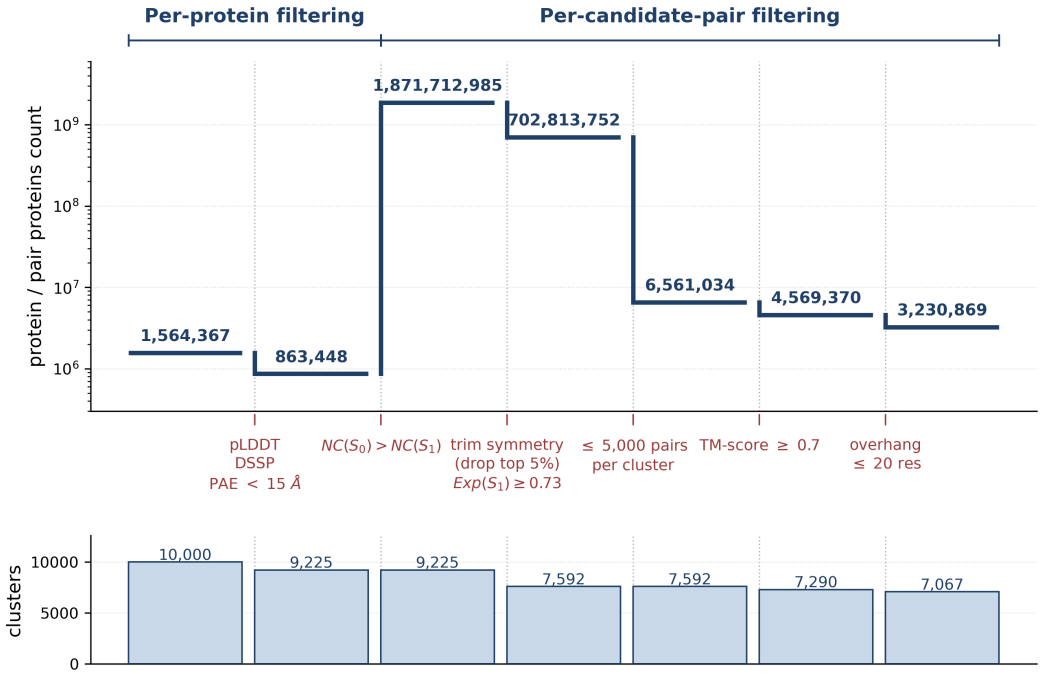


Figure 10: Curation waterfall across the 10,000 selected Foldseek clusters. The upper panel gives the number of retained proteins or candidate pairs after each filter; the lower panel gives the corresponding number of retained clusters.

C.2 Training set

We first reserve 50 clusters for unseen-cluster evaluation, sampled from clusters containing at least 20 pairs with $\mathrm{NC}(S_0) > 10\,\text{Å}$ and $\mathrm{NC}(S_1) \leq 10\,\text{Å}$. The training set retains the top 20 ranked directed pairs from each remaining NE+ cluster, or all pairs when fewer than 20 remain, producing 112,868 pairs.

C.3 Evaluation sets

We construct seen-cluster and unseen-cluster evaluations from the NE+ group, alongside three difficulty levels from the NE0 group. The additional $\mathrm{MinCut} \geq 3$ filter aims to exclude weakly connected two-lobe structures whose N–C separation could change through hinge motion. We prioritize large N–C distances and low N–C connection exposure using $\frac{1}{2}z(-\mathrm{Exp}) + \frac{1}{2}z(\mathrm{NC})$ within each cluster.

- **Seen clusters in the NE+ group.** We evaluate unseen proteins from clusters seen during training. From each eligible cluster, we select one protein **absent** from its top 30 ranked source–target training pairs, with $100 \leq D \leq 400$ and $\mathrm{MinCut} \geq 3$. We prioritize large N–C distances and low N–C connection exposure using $\frac{1}{2}z(-\mathrm{Exp}) + \frac{1}{2}z(\mathrm{NC})$ within each cluster.
- **Unseen clusters in the NE+ group.** We evaluate proteins from randomly sampled clusters from the NE+ group held out entirely from training. Eligible clusters contain at least 20 qualifying pairs with $\mathrm{NC}(S_0) > 10\,\text{Å}$ and $\mathrm{NC}(S_1) \leq 10\,\text{Å}$; evaluation proteins are selected for large N–C distances and low N–C connection exposure.
- **Unseen clusters in the NE0 group.** From unseen clusters with no qualifying training pairs, we retain proteins with $100 \leq D \leq 400$ and $\mathrm{MinCut} \geq 3$, then select one protein per eligible cluster for each difficulty level:
  - **Easy:** higher N–C connection exposure (↑ Exp) and smaller terminal distances (↓ NC), with $\mathrm{NC} > 15\,\text{Å}$.
  - **Medium:** higher N–C connection exposure (↑ Exp) and larger terminal distances (↑ NC).

• **Hard:** lower N–C connection exposure (↓ Exp) and larger terminal distances (↑ NC).

These levels reflect increasing geometric demands for terminal closure and connection exposure.

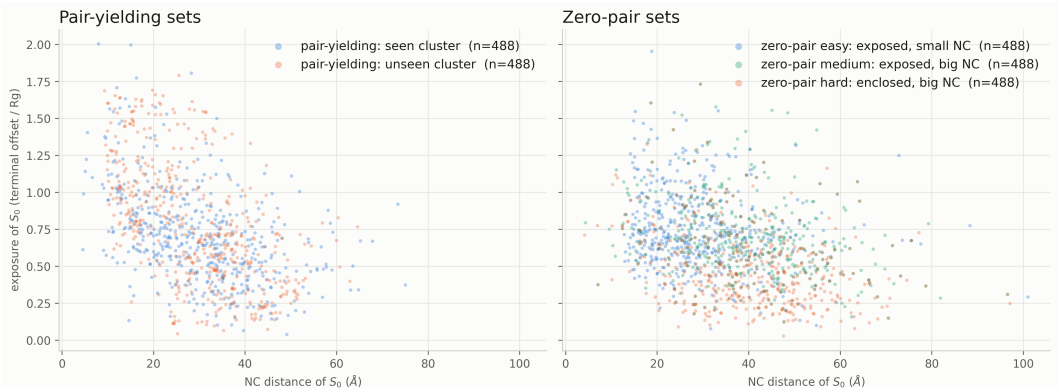


Figure 11: Starting N–C distance and connection exposure of the evaluation proteins. Seen and unseen NE+ clusters have broadly overlapping distributions. The NE0 difficulty sets shift from exposed proteins with smaller N–C distances (top left) toward enclosed proteins with larger distances (bottom right).

D Additional analyses and ablations

We report further analyses of inference outputs on the validation sets, together with the remaining training and sampling ablations.

D.1 Toy and protein branching tasks

We first use controlled two-dimensional shapes to simplify protein morphing while retaining its essential features: ordered coordinates and identities, expansion and contraction, and one-to-many outcomes from a shared source. These tasks isolate whether the model can recover the intended edit pattern and sample distinct output modes before the structural complexity of real proteins is introduced.

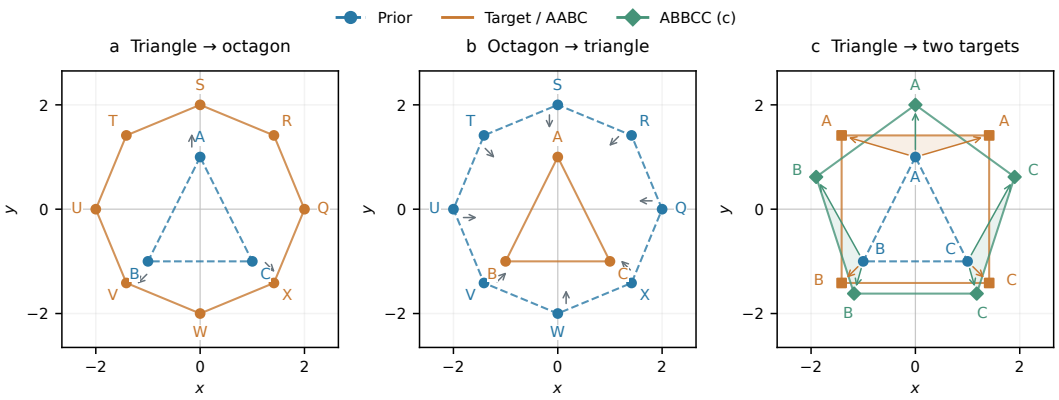


Figure 12: Controlled toy morphing tasks. Blue denotes the source and target colors denote residue identities; arrows indicate edit direction. Shaded wedges in (c) show parent–child assignments for the one-to-many task.

Expansion and contraction. The model expands a triangle with identities ABC to an octagon with identities QRSTUVWX (Figure 12a), matching the target in all 30 samples with mean TM score 0.996. It also contracts an octagon to a triangle while changing identities (Figure 12b); all 30 samples match, with mean TM score 1.000.

One-to-many branching. A shared ABC triangle maps to AABC (square) or ABBCC (pentagon) through distinct split patterns (Figure 12c). Among 500 samples, 220 match AABC (44.0%), 278 match ABBCC (55.6%), and two are unmatched (0.4%). Both recovered modes have mean TM score 0.999.

Protein branching. As a basic check of one-to-many protein morphing, we next tested whether RayMorph could retrieve multiple targets from a single source through expansion, contraction, and unchanged-length transformations (Figure 3). A shared 60-residue source maps to four target proteins of lengths 65, 53, 56, and 60, each with equal training probability. Across 30 samples, the four targets were recovered 7, 7, 6, and 6 times, respectively, with four unmatched outputs.

D.2 Complete validation results

Tables 3 and 4 report the full protein- and cluster-level evaluation results, including predicted, structure-refolded, and sequence-refolded quality. Structure-refolded results use the lowest-scRMSD refold among 8 ProteinMPNN sequences per backbone; sequence-refolded results use one ESMFold prediction of RayMorph's generated sequence per rollout. Both use 3 RayMorph rollouts per source.

	Seen clusters		**Unseen clusters**	
Criterion	Unseen Proteins (%)	Unique clusters (count)	Proteins (%)	Unique clusters (count)
Predicted structure quality				
(a) Bonds and clash-free $\geq 99\%$	84.2%	411/488	77.9%	49/50
(b) (a) + TM > 0.7	83.4%	407/488	77.5%	49/50
(c) (b) + dNC < -5 Å	64.5%	315/488	60.9%	48/50
(d) (b) + exposure criterion	73.0%	356/488	67.8%	49/50
(e) (c) and (d)	55.9%	273/488	54.1%	48/50
Structure-refolded quality				
(a) scRMSD < 2 Å	41.2%	201/488	42.0%	39/50
(b) (a) + TM > 0.7	41.0%	200/488	41.8%	39/50
(c) (b) + dNC < -5 Å	24.0%	117/488	26.8%	33/50
(d) (b) + exposure criterion	37.5%	183/488	37.3%	36/50
(e) (c) and (d)	22.3%	109/488	25.4%	30/50
Sequence-refolded quality				
(a) scRMSD < 2 Å	15.0%	73/488	7.8%	14/50
(b) (a) + TM > 0.7	15.0%	73/488	7.8%	14/50
(c) (b) + dNC < -5 Å	3.5%	17/488	4.1%	9/50
(d) (b) + exposure criterion	11.3%	55/488	7.6%	13/50
(e) (c) and (d)	2.9%	14/488	4.1%	9/50

Table 3: Detailed results (passing protein percentages and unique-cluster counts) for the NE+ group. Each set contains 488 proteins, drawn from 488 seen or 50 unseen clusters. A protein passes if at least one of three rollouts satisfies the row criteria jointly; a cluster passes if at least one protein passes, counting each cluster once. The exposure criterion is dExp > 0.3 or final Exp > 0.8. TM, dNC, and dExp are measured relative to the source on the predicted or respective refolded structures.

	Easy		**Medium**		**Hard**	
Criterion	Proteins (%)	Unique clusters (count)	Proteins (%)	Unique clusters (count)	Proteins (%)	Unique clusters (count)
Predicted structure quality						
(a) Bonds and clash-free $\geq 99\%$	85.0%	415/488	84.8%	414/488	75.2%	367/488
(b) (a) + TM > 0.7	84.4%	412/488	84.0%	410/488	74.0%	361/488
(c) (b) + dNC < -5 Å	61.9%	302/488	67.6%	330/488	59.8%	292/488
(d) (b) + exposure criterion	63.7%	311/488	62.7%	306/488	54.5%	266/488
(e) (c) and (d)	47.5%	232/488	54.1%	264/488	47.7%	233/488

Continued on next page

Table 4 (continued)

Criterion	Easy Proteins (%)	Easy Unique clusters (count)	Medium Proteins (%)	Medium Unique clusters (count)	Hard Proteins (%)	Hard Unique clusters (count)
Structure-refolded quality						
(a) scRMSD < 2 Å	32.0%	156/488	25.0%	122/488	23.2%	113/488
(b) (a) + TM > 0.7	31.8%	155/488	25.0%	122/488	22.7%	111/488
(c) (b) + $\mathrm{dNC} < -5$ Å	16.6%	81/488	14.5%	71/488	13.5%	66/488
(d) (b) + exposure criterion	24.4%	119/488	18.4%	90/488	15.0%	73/488
(e) (c) and (d)	15.2%	74/488	12.3%	60/488	10.0%	49/488
Sequence-refolded quality						
(a) scRMSD < 2 Å	2.7%	13/488	1.6%	8/488	1.0%	5/488
(b) (a) + TM > 0.7	2.7%	13/488	1.6%	8/488	1.0%	5/488
(c) (b) + $\mathrm{dNC} < -5$ Å	1.0%	5/488	0.6%	3/488	0.4%	2/488
(d) (b) + exposure criterion	2.0%	10/488	1.4%	7/488	0.8%	4/488
(e) (c) and (d)	0.8%	4/488	0.6%	3/488	0.4%	2/488

Table 4: Detailed results (passing protein percentages and unique-cluster counts) for the NE0 group. Each difficulty set contains 488 proteins from 488 unseen clusters. A protein passes if at least one of three rollouts satisfies the row criteria jointly; a cluster passes if at least one protein passes, counting each cluster once. The exposure criterion is $\mathrm{dExp} > 0.3$ or final $\mathrm{Exp} > 0.8$. TM, dNC, and dExp are measured relative to the source on the predicted or respective refolded structures.

For structure-refolded quality, all criteria are met in $109/488$ seen and $30/50$ unseen NE+ clusters, and in $74/488$, $60/488$, and $49/488$ NE0 clusters in the easy, medium, and hard sets, respectively. Sequence-refolded quality is substantially lower, indicating that the amino-acid identity head remains a limitation.

Unique-cluster counts measure coverage within each set and should not be compared directly across groups: the unseen NE+ set contains 488 proteins from 50 clusters, whereas each NE0 set contains one protein from each of 488 clusters.

D.3 Validation-output analyses

The main-text Pareto analysis examines the trade-off between N–C closure and source-fold retention for individual refolded outputs. Here, we complement that view by contrasting the distributions of terminal remodeling and residue-count change in training pairs and validation rollouts.

D.3.1 Training-pair and generated-output distributions

We compare the distribution between designable validation rollouts from NE+ unseen cluster with training dataset pairs. The distributions of N–C closure and final connection exposure broadly match those of the training pairs (Figure 13). A small 11% of rollouts have non-positive N–C closure. Since the unseen NE+ sources are already exposed, retaining their connection exposure is a favorable outcome.

Figure 14 shows that designable residue-count changes reproduce the central shape of the training distribution. Their dLen is -2.0 ± 9.2 residues, compared with -0.2 ± 19.7 in the training pairs; designable outputs become sparse toward the larger absolute changes represented in the training-pair tails.

D.3.2 Inference-time sampling controls

We vary the bridge-noise scale σ and synchronized-edit firing threshold ratio r without retraining. Each setting uses the same checkpoint and 6 proteins from unseen NE+ clusters, with 100 rollouts per protein. The default setting is $(\sigma, r) = (0.1, 5.0)$; decreasing r fires edits more often and therefore permits larger length changes. We measure diversity within each source protein: among its designable ESMFold-refolded rollouts, we compute all pairwise TM-align scores, normalize each

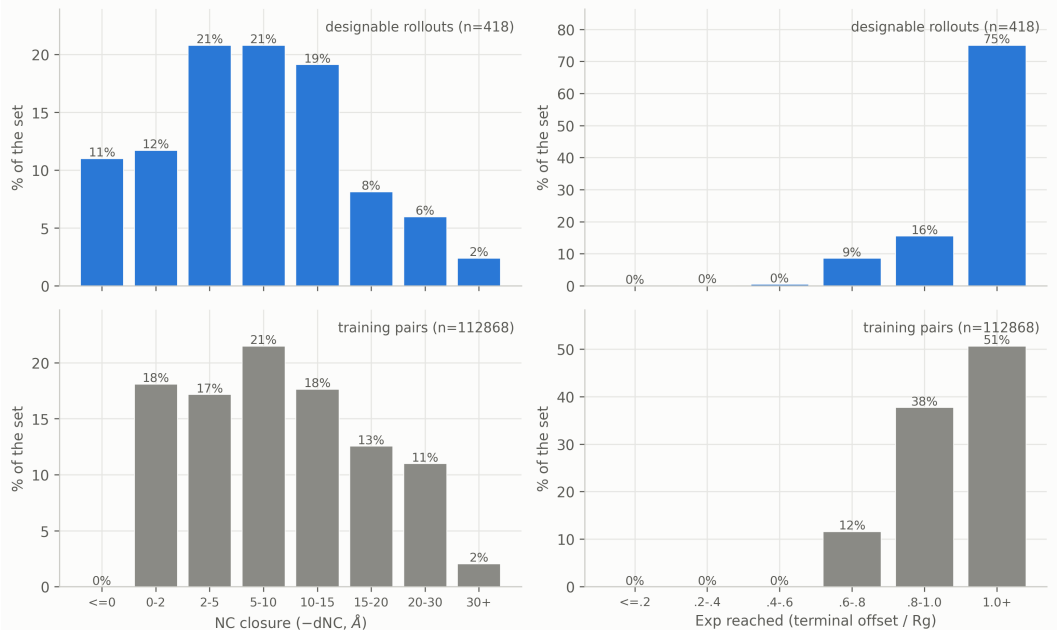


Figure 13: Terminal-remodeling distributions for designable validation rollouts (top) and directed training pairs (bottom). Left: N–C closure, $-\mathrm{dNC}$; right: final N–C connection exposure, Exp.

pair by its shorter chain, and average $1-\mathrm{TM}$ across the six sources. This measures alternative structures generated from the same input, rather than a diversity score pooled across different proteins.

Table 5: Bridge-noise ablation, with firing ratio fixed at $r = 5.0$. Designability pools all 600 rollouts; $\pm$ terms are standard deviations across proteins. Arrows indicate the preferred direction.

σ	Designable (%↑)	Diversity ($1-\mathrm{TM}$ ↑)	TM_{x_0} ↑	Edit extent $\lvert\mathrm{dLen}\rvert$
0.10 (default)	31.3	0.130 ± 0.009	0.847	19.3 ± 3.0
0.05	70.8	0.099 ± 0.009	0.863	19.4 ± 6.1
0.01	83.3	0.079 ± 0.012	0.873	19.0 ± 8.3

Table 6: Edit-firing ratio ablation, with bridge noise fixed at $\sigma = 0.1$. Lower r fires edits more often. Designability pools all 600 rollouts; $\pm$ terms are standard deviations across proteins. Arrows indicate the preferred direction.

r	Designable (%↑)	Diversity ($1-\mathrm{TM}$ ↑)	TM_{x_0} ↑	Edit extent $\lvert\mathrm{dLen}\rvert$
5.0 (default)	31.3	0.130 ± 0.009	0.847	19.3 ± 3.0
4.0	17.8	0.136 ± 0.010	0.837	24.4 ± 3.6
3.0	12.2	0.146 ± 0.014	0.816	31.0 ± 5.7

Lowering σ improves refolding designability while leaving the mean edit extent nearly unchanged, at the cost of lower within-source diversity. In contrast, lowering r increases the edit extent but sharply reduces designability; its small diversity gain does not produce distinct modes at TM-align 0.5. Thus, the bridge-noise scale is the useful control for obtaining more designable proteins in this sweep, whereas the edit-firing ratio provides limited benefit.

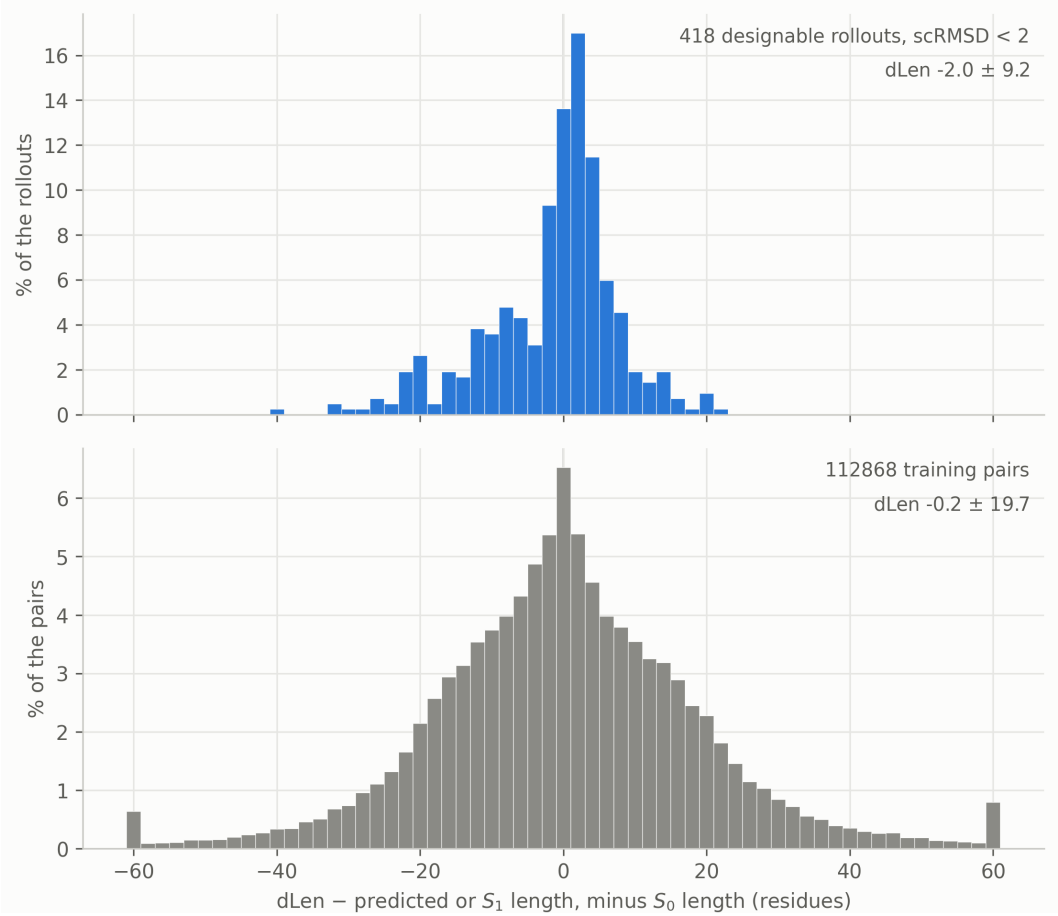


Figure 14: Residue-count change distributions for designable validation rollouts (top) and directed training pairs (bottom). The generated distribution reproduces the central training-pair shape but falls short at larger absolute changes.