# PEPTIDE DESIGN AND OPTIMIZATION VIA STRUCTURED FULL-ATOM EDITING

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Peptide lead optimization seeks to improve molecular properties through selective modifications while retaining target engagement. The development of semaglutide from glucagon-like peptide-1 (GLP-1) illustrates this principle. Applying pocket-conditioned sequence–structure generation to lead optimization requires control over both which residues are modified and which molecular variables may change. Here we introduce EditFlow, a flow-matching model for structured full-atom editing of bound peptides. Residue–variable masks specify the scope of each edit, enabling a shared conditional model to regenerate selected amino acid identities, backbone geometry, and side-chain conformations while preserving the receptor and all unselected peptide variables. Coupled with external property evaluation and search, the same frozen editor supports iterative optimization through successive local modifications. Extensive evaluations on protein–peptide complexes demonstrate state-of-the-art performance in joint sequence–structure redesign alongside competitive full-generation performance. Across single- and multi-objective tasks, EditFlow outperforms strong baselines in improving molecular properties and achieving favorable trade-offs among competing objectives. These results establish structured full-atom editing as a unified approach to peptide design and lead optimization. The source code will be available upon acceptance.

## 1 INTRODUCTION

The clinical success of glucagon-like peptide-1 (GLP-1) receptor agonists has brought peptide therapeutics into sharp focus (Muttenthaler et al., 2021; Wang et al., 2022). Nearly 100 peptide drugs have been approved worldwide, with reported sales of 13.89 billion USD for injectable semaglutide (Ozempic) and 2.72 billion USD for oral semaglutide (Rybelsus) (Xiao et al., 2025). Behind this success lies the optimization of naturally occurring peptides into effective therapeutics. Semaglutide descends from GLP-1 through amino-acid substitutions and side-chain acylation that preserved receptor engagement while extending the plasma half-life from minutes to about a week (Lau et al., 2015). The same principle of preserving target recognition while improving other molecular properties makes peptide lead optimization central to therapeutic development (Fosgerau & Hoffmann, 2015; Muttenthaler et al., 2021).

Discovering and refining such binders has long rested on display-based selection over mutagenesis libraries and on energy-based modeling of candidate complexes (Boder & Wittrup, 1997; Raveh et al., 2010; Leaver-Fay et al., 2011), both of which search a combinatorial space that grows faster than either can enumerate. Deep generative models have changed the terms of that search by learning distributions over protein sequence and structure directly (Luo et al., 2022; Watson et al., 2023; Lisanza et al., 2025; Chu et al., 2024), and for receptor-bound peptides, PepFlow, PepGLAD, PPFlow, and PepHAR now generate sequence and structure together with the target pocket as context (Li et al., 2024; Kong et al., 2024; Lin et al., 2024; Li et al., 2025). At the residue level, PepFlow and PepBFN place amino-acid identity, backbone frame, and side-chain torsions in a single generative state (Li et al., 2024; Qian et al., 2026a), while partial sampling, retained hot spots, and atom-level prompts each condition generation on molecular information already in hand (Li et al., 2024; 2025; Peng et al., 2026). Appendix A surveys these lines in full.



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.

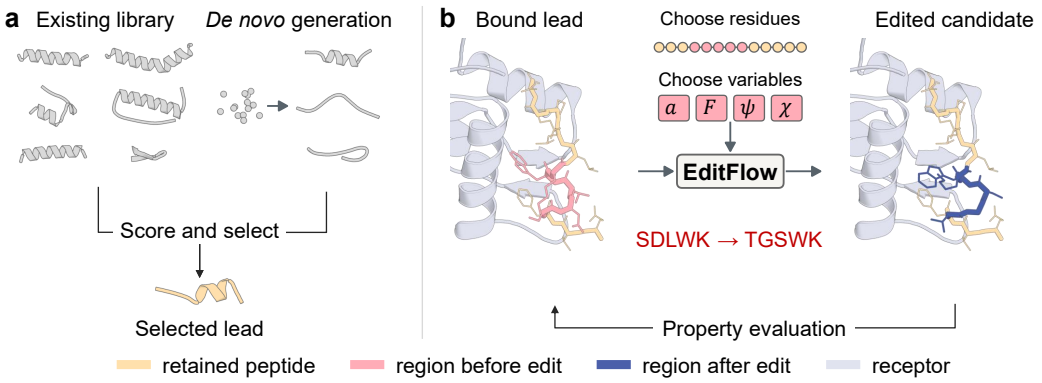



Figure 1: **Selective editing for peptide lead optimization.** (a) Schematic candidate selection from existing libraries or pocket-conditioned generation. (b) An edit specifies residue identities $a$, backbone frames $F$, and torsions $\psi, \chi$ to regenerate within a selected region. Peptide residues 20–24 undergo local sequence–structure remodeling, while the receptor and all peptide atoms outside this region are retained (PDB 1YCR). Pale blue denotes the receptor, cream the retained peptide, and pink and dark blue the editable region before and after editing. Both states share the same viewing direction and scale.

Starting from a bound lead rather than an empty pocket changes what the model is asked to do, and three difficulties follow. First, the residues of a bound peptide do not carry equal weight. A few interface residues dominate recognition and are precisely the ones a designer least wants to disturb (Bogan & Thorn, 1998; Keskin et al., 2005; Moreira et al., 2007), whereas the positions that govern solubility, permeability, or proteolytic stability usually lie elsewhere along the chain. Structure-guided optimization of the p53-derived inhibitor pDI combined interface substitutions with a C-terminal conformational adjustment (Phan et al., 2010), and Compstatin and EPI-X4 analogues were improved through changes aimed at solubility and plasma stability while their binding cores were left intact (Gorham et al., 2015; Harms et al., 2023). A design request must therefore name what is retained, which residues may change, and which molecular variables may change at those residues (Figure 1), a division that methods fixing it internally cannot express.

Second, sequence and structure cannot be chosen one after the other within such an edit. Changing a residue identity changes which side-chain torsions exist to be placed, and moving a backbone frame moves every atom rebuilt from it (Li et al., 2024; Qian et al., 2026a; Lee & Kim, 2025). A pipeline that designs a sequence onto a backbone it is not allowed to revise inherits whatever error that backbone carries, so the released variables have to be learned jointly, conditioned on the receptor and on the peptide state that stays. Methods that fix in advance which part of the molecule is given leave remodeling a backbone, redesigning a segment, and repacking an interface as separate procedures with separate models, even when they act on one complex and a designer wants to interleave them (Dauparas et al., 2022; Lee & Kim, 2025; Watson et al., 2023).

Third, an editor meant for lead optimization has to hand something scorable to whatever chooses the next candidate. Property-guided generative methods already explore peptide leads or steer sequence–structure design with external scores, mostly by folding the objective into a guidance term, a reward, or a control problem (Lin et al., 2024; Qian et al., 2026b; Geylan et al., 2025; Wang et al., 2025; Yao et al., 2025), which binds the optimizer to the generator tightly enough that a new property or a different search policy generally calls for a new guidance signal or another round of training. The properties of a lead also pull against one another, since the charged residues that favor cell penetration also lower solubility, and substitutions that improve proteolytic stability can weaken the interface they sit in (Fosgerau & Hoffmann, 2015; Muttenthaler et al., 2021). A real campaign therefore needs both single-objective gains and a set of candidates spread across the trade-offs (Deb et al., 2002; Zhang et al., 2026).

To address these difficulties we introduce EditFlow, a flow-matching model that makes the controlled change of a bound peptide the single operation behind both design and optimization. An edit names a residue region together with the physical variables released there, and a dependency-closure rule

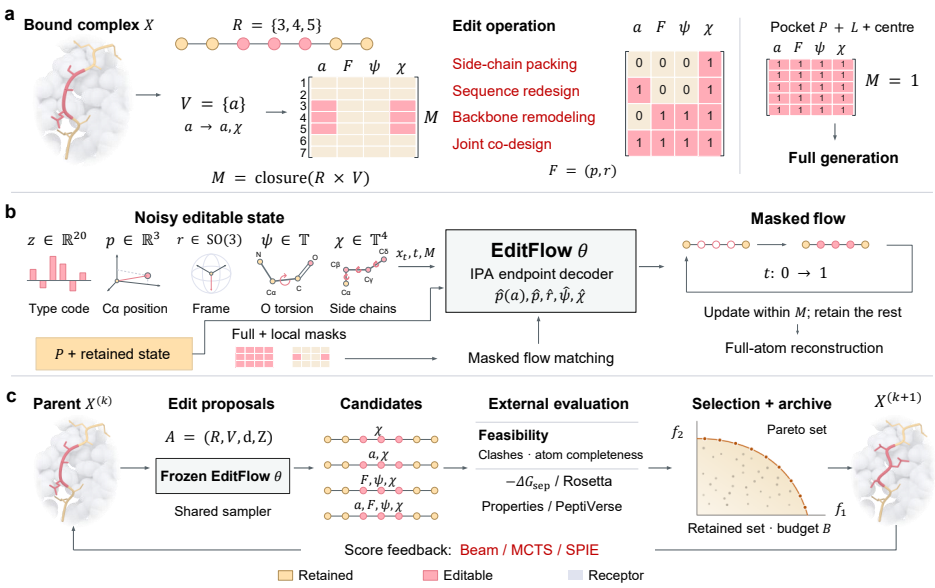



Figure 2: **Overview of EditFlow.** (a) Residue and variable selections define the edit mask through dependency closure. (b) A shared decoder learns joint sequence–structure reconstruction under different masks. (c) Property evaluation and search guide peptide optimization through the same editing interface. $d$ and $Z$ denote the sampling schedule and noise. Structures and the Pareto front are schematic.

adds the torsions implied by a change of identity or of backbone placement, so the released set stays physically coherent. One shared flow model learns the four local edit conditions from the retained complex, pairing their variable sets on the same sampled region, and learns full pocket-conditioned generation alongside them. Because every sampled edit returns a complete bound complex, anything that can score a complex can drive the editor, and a single set of weights supplies proposals to different search strategies without a search-specific objective or another round of training.

We summarize our contributions below.

- We formulate the redesign of a bound peptide as conditional full-atom generation under an explicit residue–variable instruction that preserves the unselected molecular state, which turns the scope of an edit into something the user states rather than something the method fixes.
- We train one flow model to perform full pocket-conditioned generation together with four local edit operations, and because each edit returns a complete bound complex, the same weights act as the proposal mechanism for four search policies and for every property objective we test.
- We show that this formulation pays off on both halves of the design problem. Full generation reaches 1.80 Å C$\alpha$ RMSD and 88.10% binding-site recovery, joint redesign lifts amino-acid recovery from 4.68% for a sequential Rosetta pipeline to 40.78%, and under a matched proposal budget EditFlow raises final five-objective hypervolume by 29.1–45.2% over PepFlow across all four search policies.

## 2 METHOD

**Problem setup and overview.** Given a protein pocket $P$ and a bound peptide $X^{\text{par}}$, an edit instruction names the residues $\mathcal{R}$ and the variables $\mathcal{V}$ to regenerate (Figure 2a). EditFlow generates those editable variables jointly, conditioned on the pocket and on the peptide state left in place (Figure 2b). External evaluation and search then guide repeated applications of the editor toward a property objective (Figure 2c).

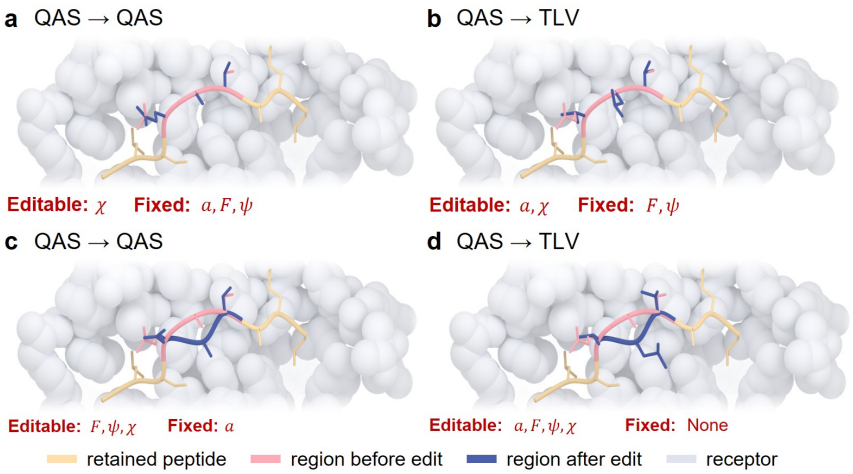



Figure 3: **Four operations for selective peptide editing.** (a) Side-chain packing. (b) Sequence redesign. (c) Backbone remodeling. (d) Joint redesign. All panels illustrate edits constructed from the 3MRE complex (receptor chains A/B, peptide chain P). Cream denotes retained peptide, pink the editable region before editing, and dark blue the edited region. Editable and Fixed refer to variables within $\mathcal{R}$, and the receptor and peptide outside $\mathcal{R}$ are retained. Example details are given in Appendix C.1.

### 2.1 Editing as conditional completion

**Structured full-atom state.** Each residue is represented by separately editable variables for residue type, backbone placement, and torsion angles. For a peptide of length $L$, let $X = (x_1, \ldots, x_L)$, where

$$x_i = (a_i, p_i, r_i, \psi_i, \chi_i), \qquad a_i \in \{1, \ldots, 20\}, \quad p_i \in \mathbb{R}^3, \quad r_i \in SO(3), \quad (\psi_i, \chi_i) \in \mathbb{T}^5. \tag{1}$$

Here $a_i$ is the residue type, $p_i$ the C$\alpha$ position, $r_i$ the backbone-frame rotation, $\psi_i$ the backbone torsion governing the O-atom position, and $\chi_i$ the vector of up to four side-chain torsions. The space $\mathbb{T}$ denotes periodic angles. Residue templates reconstruct heavy-atom coordinates, with invalid torsion slots masked according to $a_i$.

**Residue–variable instructions.** A residue set $\mathcal{R} \subseteq \{1, \ldots, L\}$ selects positions, and a variable set $\mathcal{V}$ selects editable channels. Grouping position and rotation as $F = (p, r)$ gives the channel set $\mathcal{C} = \{a, F, \psi, \chi_1, \ldots, \chi_4\}$, and dependency closure turns a request into the effective edit mask (Figure 2a),

$$M = \text{closure}(\mathcal{R} \times \mathcal{V}) \subseteq \{1, \ldots, L\} \times \mathcal{C}. \tag{2}$$

Here closure adds the variables that a requested change implies, so that changing a residue type releases its side-chain torsions and changing a backbone frame releases both $\psi$ and the side-chain torsions rebuilt from it. The resulting mask determines which variables are perturbed, supervised, and updated during sampling (Figure 3). Appendix B gives the formal closure rules and expressivity analysis, and Appendix C.1 tabulates the variable sets.

**Conditional edit distribution.** EditFlow learns a joint conditional distribution over the editable variables,

$$X'_M \sim p_\theta(X_M \mid P, X^{\text{par}}_{\bar{M}}, M), \qquad X'_{\bar{M}} = X^{\text{par}}_{\bar{M}}. \tag{3}$$

Here $\bar{M}$ contains all fixed peptide variables, including those within $\mathcal{R}$. Varying $M$ changes both the generated variables and the peptide information available for conditioning. When no peptide variables are fixed, the model performs pocket-conditioned *de novo* generation. When only a few complete residues are fixed, it performs sparse-anchor completion.

### 2.2 LEARNING A CONDITIONAL FULL-ATOM EDITOR

EditFlow learns these conditional problems with one network (Figure 2b), rather than training a separate model per edit type. Training pairs each sampled residue region $\mathcal{R}$ with the four variable sets in Figure 3 to form masks $M(\mathcal{R}, \mathcal{V}_j)$. Regions comprise one or more segments, drawn from a distribution that favors short edits and near-complete reconstruction (Appendix D). Editable variables are perturbed along flow paths, and the network predicts their endpoints from the perturbed state together with the fixed context.

**Masked generation paths.** The training path interpolates editable variables from prior noise to the training target while keeping the remaining variables fixed. Let $X^\star$ denote the ground-truth peptide state, with $\star$ marking target values. For prior noise $Z$ and time $t \in [0, 1]$, the masked path is

$$(X_t)_{iv} = \begin{cases} \mathcal{I}_v(Z_{iv}, X^\star_{iv}; t), & (i, v) \in M, \\ X^\star_{iv}, & (i, v) \notin M. \end{cases} \tag{4}$$

Residue types use continuous logits $z = c(a) \in \mathbb{R}^{20}$, and torsions are grouped as $\tau = (\psi, \chi)$. The modality-specific interpolations $\mathcal{I}_v$ are then

$$\begin{aligned} z_t &= (1-t)z_0 + t\, c(a^\star), \qquad p_t = (1-t)p_0 + t\, p^\star, \\ r_t &= r_0 \operatorname{Exp}\left[t \operatorname{Log}(r_0^{\mathsf{T}} r^\star)\right], \quad \tau_t = \left[\tau_0 + t \operatorname{wrap}(\tau^\star - \tau_0)\right]_{2\pi}. \end{aligned} \tag{5}$$

Each modality travels the path natural to its own geometry, with Euclidean interpolation for logits and positions (Lipman et al., 2023), rotation geodesics on $SO(3)$ for frames (Chen & Lipman, 2024; Bose et al., 2024), and shortest periodic paths for torsions (Chen & Lipman, 2024). Here $\operatorname{wrap}$ returns the shortest signed angular difference, and $[\cdot]_{2\pi}$ denotes reduction modulo $2\pi$. Appendix D specifies the encoding, priors, and time sampling.

**Joint endpoint prediction.** Features derived from editable residue types or geometry are masked before encoding, so the network never sees the answer it is asked to predict. The encoder maps the remaining context to residue features $\mathbf{h}$ and pair features $\mathbf{e}$, which condition endpoint prediction together with $X_t$ and $t$,

$$\begin{aligned} (\mathbf{h}, \mathbf{e}) &= \operatorname{Enc}_\theta(P, X^\star; M), \\ (\widehat{\pi}_1, \widehat{p}_1, \widehat{r}_1, \widehat{\tau}_1) &= \operatorname{Dec}_\theta(X_t, t, \mathbf{h}, \mathbf{e}). \end{aligned} \tag{6}$$

Here $\widehat{\pi}_1$ is the predicted residue-type distribution, and the remaining outputs are structural endpoints. Invariant point attention (Jumper et al., 2021) integrates the current residue types, frames, and torsions into shared features that feed every prediction head, which is what lets identity and geometry be predicted under one representation instead of two. Appendix D specifies the architecture and feature masks.

**Training objective.** For each mask, $\mathcal{L}_M$ supervises residue types and geometry at the valid editable entries, with every loss term normalized by its supervised count within a complex so that long and short edits contribute comparably. Let $M_{\text{full}}$ release all peptide variables. The five tasks share the sampled time and base noise, and their losses combine as

$$\mathcal{L}_{\text{train}} = \mathbb{E}\left[\frac{1}{2}\mathcal{L}_{M_{\text{full}}} + \frac{1}{8}\sum_{j=1}^{4} \mathcal{L}_{M(\mathcal{R}, \mathcal{V}_j)}\right]. \tag{7}$$

Appendix D gives the component losses, weights, and training schedule.

### 2.3 SAMPLING EDITED COMPLEXES

**Masked sampling.** Each step moves the editable variables toward the predicted endpoint $\widehat{X}_1$ and restores the fixed parent state,

$$X_{t+\Delta t} = \operatorname{Merge}_M\left(X^{\text{par}}, \mathcal{U}_{t,\Delta t}(X_t, \widehat{X}_1)\right), \qquad (X_{t+\Delta t})_{\bar{M}} = X^{\text{par}}_{\bar{M}}. \tag{8}$$

Here $\mathcal{U}_{t,\Delta t}$ is the modality-specific update, and $\operatorname{Merge}_M(A, B)$ takes variables in $M$ from $B$ and the remainder from $A$.

**Sampling with window refinement.** For sparse-anchor completion and property search, overlapping windows refine an initial candidate, allowing earlier choices to be revised as neighboring residues are updated. Each window restricts the original residue–variable mask to a contiguous region, so the same predictor performs the update:

$$\begin{aligned} X^{(0)} &= \mathcal{C}_\theta(P, X^{\mathrm{par}}; M), \\ X^{(k+1)} &= \mathcal{R}_\theta(P, X^{(k)}; W_k), \qquad W_k = M \cap (I_k \times \mathcal{C}). \end{aligned} \tag{9}$$

Here $\mathcal{C}_\theta$ denotes initial completion, $\mathcal{R}_\theta$ a window update, and $I_k$ a contiguous residue window over variable channels $\mathcal{C}$. Appendix E details the initialization, staged updates, sampling schedules, and computational costs.

**Reconstructing the edited complex.** The final sampled state $X^{\mathrm{out}}$ is rebuilt with residue templates, and only the editable atom groups are written back,

$$Y' = \mathrm{Merge}_{A(M)}\left(Y^{\mathrm{par}}, \mathrm{Build}(X^{\mathrm{out}})\right), \qquad Y'_{\overline{A(M)}} = Y^{\mathrm{par}}_{\overline{A(M)}}. \tag{10}$$

Here $Y^{\mathrm{par}}$ contains the parent atom coordinates, and $A(M)$ identifies the editable atom groups according to the mask and generated residue types. Appendix E specifies the atom mapping, and Appendix C proves preservation of the retained atoms.

### 2.4 Property-guided iterative editing

External property evaluation guides repeated editing of a bound peptide while the editor parameters stay as trained (Figure 2c). At iteration $k$, four operations share a sampled region $\mathcal{R}_k$ of the parent $X_k^{\mathrm{par}}$,

$$X'_{k,o} = \mathcal{E}_\theta(P, X_k^{\mathrm{par}}; \mathcal{R}_k, \mathcal{V}_o, d_k, Z_{k,o}), \qquad o \in \mathcal{O}. \tag{11}$$

Here $\mathcal{O}$ contains the operations in Figure 3, $d_k$ specifies the sampling schedule, and $Z_{k,o}$ provides independent noise. The operator $\mathcal{E}_\theta$ includes sampling and atom reconstruction.

Each candidate receives property scores $s(X)$ and a constraint violation $\nu(X)$, with larger scores preferred and $\nu(X) = 0$ marking feasibility. A search policy $\Pi$ then updates its state and selects the next edit,

$$\begin{aligned} \mathcal{B}_k &= \left\{(X'_{k,o}, s(X'_{k,o}), \nu(X'_{k,o}))\right\}_{o \in \mathcal{O}}, \\ \mathcal{S}_{k+1} &= \mathrm{Update}_\Pi(\mathcal{S}_k, \mathcal{B}_k), \\ (X_{k+1}^{\mathrm{par}}, d_{k+1}) &= \mathrm{Select}_\Pi(\mathcal{S}_{k+1}). \end{aligned} \tag{12}$$

The state $\mathcal{S}_k$ stores the retained candidates and the policy statistics, and iteration continues until the candidate-evaluation budget is exhausted. Nothing in Eq. (12) refers to the editor beyond its ability to return a complete complex, which is why the same weights drive best-of-$N$, structured Pareto iterative editing, beam search, and Monte Carlo tree search without modification. Appendix F specifies the search policies and sampling schedules.

## 3 Experiments

**Benchmark.** We evaluate on protein–peptide complexes drawn from PepBDB (Wen et al., 2019) and Q-BioLiP (Wei et al., 2024), held out by peptide sequence clustering from the training pool and expressed in a frame centered on the native peptide C$\alpha$ centroid (Li et al., 2024). Full generation and sparse-anchor completion follow the PepHAR protocol (Li et al., 2025). Appendix D provides training details, and Appendix G gives the sampling configurations, aggregation rules, and full metric definitions.

The experiments vary a single quantity, how much of the bound peptide is retained. Releasing the whole peptide gives pocket-conditioned generation (Section 3.1), retaining part of the peptide state and regenerating the editable variables gives joint redesign (Section 3.1), and retaining a few complete residues while releasing everything else gives peptide completion (Section 3.2). Section 3.3 then holds the instruction fixed and varies the objective instead, with the editor supplying every proposal.

**Metrics.** Reconstruction is measured against the reference peptide at editable positions, with receptor atoms excluded throughout. Amino-acid recovery (AAR) is the fraction of those positions carrying the reference identity, C$\alpha$ RMSD and segment BB-RMSD measure position error after peptide alignment and in the fixed complex frame, secondary-structure recovery (SSR) compares DSSP assignments (Kabsch & Sander, 1983), and binding-site recovery (BSR) compares the receptor residues the peptide contacts. Stability and affinity are computational energy pass rates relative to the reference complex under the benchmark scoring protocol. Editing adds two measures. *Interface contact F1* scores the edited residues' receptor contacts against the reference, and *segment closure* is the fraction of samples whose boundary C–N bonds fall within 0.20 Å of 1.33 Å, testing whether a released segment rejoins the retained chain. For property search, hypervolume (HV) is the region of normalized property space a candidate set dominates relative to a fixed reference point (Deb et al., 2002).

### 3.1 From generation to selective editing

**Pocket-conditioned generation.** We first assess the *de novo* design capability of EditFlow after training for local editing. As shown in Table 1, EditFlow achieves the lowest C$\alpha$ RMSD and the highest SSR and BSR among the reported methods, indicating its ability to reconstruct native-like peptide structures and recover their binding environments. Its leading Stab. and Aff. pass rates further support the energetic quality of the generated complexes under the benchmark's computational criteria. Together, these results demonstrate that EditFlow also serves as a competitive *de novo* generator, providing complete peptide candidates in addition to modifying existing bound peptides.

Table 1: Pocket-conditioned peptide generation. Dashes indicate unavailable values.

| Method | AAR (%)↑ | RMSD (Å)↓ | SSR (%)↑ | BSR (%)↑ | Stab. (%)↑ | Aff. (%)↑ | Div. (%)↑ |
|---|---|---|---|---|---|---|---|
| RFdiffusion | 40.14 | 4.17 | 63.86 | 26.71 | 26.82 | 16.53 | 38.00 |
| ProteinGenerator | 45.82 | 4.35 | 29.15 | 24.62 | 23.48 | 13.47 | 54.00 |
| Diffusion | 47.04 | 3.28 | 74.89 | 49.83 | 15.34 | 17.13 | **57.00** |
| PepFlow | **51.25** | 2.07 | 83.46 | 86.89 | 18.15 | 21.37 | 42.00 |
| PepGLAD | 20.59 | 3.83 | 80.24 | 19.34 | 20.39 | 10.47 | – |
| PepHAR ($K = 3$) | 34.36 | 2.68 | 84.91 | 86.74 | 16.62 | 20.53 | – |
| **EditFlow** | 42.77 | **1.80** | **96.68** | **88.10** | **42.18** | **30.72** | 38.40 |

**Joint redesign.** We next evaluate EditFlow's ability to jointly redesign all sequence and structural variables in a selected peptide region. The receptor and unedited peptide state remain fixed, and all methods use the same editable region and local Cartesian minimization. Appendix G details the evaluation and baseline implementations. As shown in Table 2, EditFlow achieves the highest AAR and contact F1 and the lowest BB-RMSD. Compared with sequential Rosetta redesign, AAR increases from 4.68% to 40.78%, while BB-RMSD is reduced by more than half and contact F1 nearly doubles. Both baselines recover identities at close to the 5% that a random assignment over the twenty amino acids would give, whether the region is drawn from independent priors or redesigned onto a backbone the pipeline cannot revise. Releasing identity and geometry together under the retained context lifts recovery an order of magnitude above that level, so that improved sequence recovery is accompanied by more accurate placement of the editable region and by recovery of its receptor contacts. Closure reaches 99.27% after minimization, supporting reconnection to the retained peptide under the evaluation criterion.

Table 2: Joint redesign within retained context. Values are means [95% bootstrap CI]; bold marks the best mean.

| Method | AAR (%)↑ | BB-RMSD (Å)↓ | Contact F1 (%)↑ | Closure (%)↑ |
|---|---|---|---|---|
| Independent priors | 6.14 [5.88, 6.40] | 7.52 [6.92, 8.18] | 31.56 [29.42, 33.66] | 75.97 [72.07, 79.63] |
| Sequential Rosetta redesign | 4.68 [4.24, 5.16] | 7.22 [6.62, 7.87] | 32.55 [30.32, 34.84] | 83.13 [80.00, 86.07] |
| EditFlow | **40.78** [36.18, 45.51] | **2.55** [2.36, 2.75] | **65.10** [61.74, 68.35] | **99.27** [99.05, 99.47] |

All methods use local Cartesian minimization.

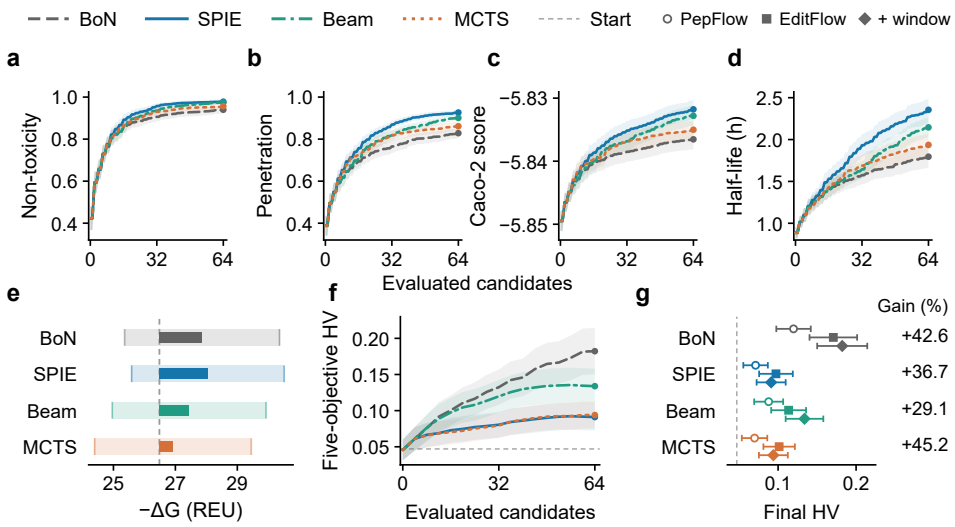



Figure 4: **Peptide property optimization.** (a–d) Mean best feasible scores during separate searches for predicted non-toxicity, penetration, Caco-2 permeability, and half-life. (e) Mean final interface score, $-\Delta G$ (REU). (f) Mean five-objective archive HV during search. Panels (a–f) use EditFlow. (g) Final HV for PepFlow and EditFlow with 200-step joint flow integration, and EditFlow with window refinement. Gain is the percentage increase from PepFlow to EditFlow under matched integration. Bands and error bars show 95% confidence intervals; dashed references in (e–g) mark mean starting values. Half-life is in hours, Caco-2 retains its native predictor scale, and non-toxicity and penetration are unitless.

### 3.2 Peptide completion from sparse anchors

Hotspot residues make key contributions to peptide–receptor interactions, motivating peptide design that preserves selected residues while regenerating the remainder (Li et al., 2025). We evaluate EditFlow's ability to complete peptide sequence and structure from sparse residue anchors while keeping the receptor and selected residue states fixed. Table 3 reports the $K = 2$ setting, testing whether EditFlow can jointly reconstruct the remaining peptide sequence and structure while preserving just two residue anchors. Appendix H.5 reports all three anchor settings and the full metrics.

Table 3: Peptide completion with two retained residue anchors. Best and second-best means are bold and underlined, respectively. RMSD includes the retained anchors.

| Method | RMSD (Å)↓ | BSR (%)↑ | Novelty (%)↑ | Div. (%)↑ |
|---|---|---|---|---|
| RFdiffusion | 2.85 | 31.14 | 45.53 | 24.14 |
| ProteinGenerator | 3.92 | 30.61 | 55.24 | 23.29 |
| PepFlow | <u>2.82</u> | 85.05 | 54.74 | 19.52 |
| PepGLAD | 3.93 | 20.02 | 54.74 | <u>30.36</u> |
| PepHAR | 2.85 | <u>85.80</u> | **74.76** | 25.32 |
| EditFlow | **2.18** | **86.80** | <u>69.32</u> | **33.10** |

EditFlow achieves the lowest RMSD and highest BSR among the compared methods, reaching 2.18 Å and 86.80%, respectively. These results support its ability to recover peptide structure and binding-site contacts from limited retained information. EditFlow also achieves the highest Div. at 33.10%, together with 69.32% Novelty, indicating variation across candidates and departure from the reference in both sequence and structure. Because each sample is conditioned on the anchors rather than extended from them, the residues in between can be placed differently on every draw while the contacts the anchors carry are preserved. This combination of reconstruction accuracy and candidate variation provides a basis for the subsequent optimization experiments, where the editor proposes the alternatives and property evaluation selects among them.

### 3.3 PEPTIDE LEAD OPTIMIZATION BY ITERATIVE EDITING

A peptide that binds is not yet a drug. Lead optimization improves measurable properties while target engagement is retained, which is precisely the operation the editing interface was built to perform. Each round scores the current candidates, selects among them, and issues the next edit, so that a single frozen editor supplies every proposal and no objective calls for retraining.

**Optimization setup.** The objectives comprise the negative Rosetta interface energy and four PeptiVerse predictions covering non-toxicity, cell penetration, Caco-2 permeability, and half-life (Zhang et al., 2026). Each objective is optimized separately, followed by joint optimization of all five. With its parameters fixed throughout, EditFlow supplies candidates to structured Pareto iterative editing (SPIE), adapted from Pareto local search (Paquete et al., 2004), best-of-$N$ (BoN) (Stiennon et al., 2020), beam search (Freitag & Al-Onaizan, 2017), and Monte Carlo tree search (MCTS) (Coulom, 2007). Each search evaluates 64 new candidates per complex, and comparisons with PepFlow use the same flow integration, mask policy, scorers, and evaluation budget.

**Single-objective results.** As shown in Figure 4a–e and Table 14, every policy improves every objective over its starting peptides, and SPIE attains the highest mean endpoint on all five. Mean predicted penetration reaches 0.927 against 0.828 for best-of-$N$, half-life 2.355 h against 1.795 h, non-toxicity 0.978 against 0.940, and the interface score 28.051 REU against 27.861 REU. The trajectories separate the policies more sharply than the endpoints do. All four behave alike over roughly the first eight evaluations, since a starting peptide never selected for these properties is improved by any feasible proposal, and they part only once those cheap gains have been spent. Non-toxicity then saturates near its ceiling with a third of the budget unspent, whereas half-life and penetration are still climbing at 64 evaluations. The interface score moves least, as it is governed by the binding mode that every proposal retains.

**Multi-objective results.** Scoring every candidate on all five objectives at once, mean HV rises from 0.047 for the starting peptides to 0.182 with BoN and 0.134 with beam search (Table 17), and no policy has flattened by the end of the budget (Figure 4f). BoN reaches the highest final HV while SPIE gives the largest single-objective gains, indicating that the policy which drives one property fastest is not the one that covers the property space best. Against PepFlow under matched search settings, EditFlow raises mean final HV by 29.1–45.2% on all four policies (Figure 4g). With every other factor held fixed, this margin must come from where the proposals land rather than from how they are selected. Window refinement raises HV further for BoN and beam search, while joint integration gives higher values for SPIE and MCTS.

In 46.5% of complexes BoN finds a candidate that matches or improves every starting score while improving at least one, the strict form of improvement a medicinal chemist would accept. Among the GRB2 SH3 candidates from the five-objective SPIE search, four substitutions raise predicted half-life from 0.518 to 0.996 h and the interface-energy score from 11.53 to 12.35 REU with the backbone retained (Appendix J.1). Appendix J follows the nine-property beam-search trajectory of Figure 5, where the interface-energy score climbs as predicted penetration falls, illustrating the conflict that makes a set of candidates worth more than a single greedy edit.

## 4 DISCUSSION AND CONCLUSION

We presented EditFlow, a flow-matching model that treats the controlled change of a bound peptide as one operation, specified by naming the residues to revise and the variables released at each of them, so that the scope of an edit becomes something a designer states rather than something a method fixes. Learning identity and geometry together matters exactly where a sequential ordering breaks down, since fixed-backbone design loses most of its accuracy once it inherits a backbone it cannot revise, whereas a joint edit lifts sequence and interface reconstruction at once. And because every sampled edit returns a complete bound complex that anything able to score a complex can evaluate, one set of weights served four search policies and five property objectives under a matched budget. Releasing receptor variables, so that an edit may account for induced fit, and admitting chemistry beyond the twenty canonical residues are both changes to what the mask may name, the natural place for a model built around that mask to grow.

## AI USE STATEMENT

AI tools assisted with manuscript organization, language editing, translation, and layout checks. All technical claims, equations, experimental design and implementation, and analyses were developed and verified by the authors. The authors also manually checked all citations and reported numerical results.

## REFERENCES

Rebecca F. Alford, Andrew Leaver-Fay, Jeliazko R. Jeliazkov, Matthew J. O'Meara, Frank P. DiMaio, Hahnbeom Park, Maxim V. Shapovalov, P. Douglas Renfrew, Vikram K. Mulligan, Kalli Kappel, Jason W. Labonte, Michael S. Pacella, Richard Bonneau, Philip Bradley, Roland L. Dunbrack, Rhiju Das, David Baker, Brian Kuhlman, Tanja Kortemme, and Jeffrey J. Gray. The Rosetta All-Atom Energy Function for Macromolecular Modeling and Design. *Journal of Chemical Theory and Computation*, 13(6):3031–3048, 2017. doi: 10.1021/acs.jctc.7b00125.

Eric T. Boder and K. Dane Wittrup. Yeast surface display for screening combinatorial polypeptide libraries. *Nature Biotechnology*, 15(6):553–557, 1997. doi: 10.1038/nbt0697-553.

Andrew A. Bogan and Kurt S. Thorn. Anatomy of hot spots in protein interfaces. *Journal of Molecular Biology*, 280(1):1–9, 1998. doi: 10.1006/jmbi.1998.1843.

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*, 2024. URL https://openreview.net/forum?id=kJFIH23hXb.

Adrian A. Canutescu and Roland L. Dunbrack, Jr. Cyclic coordinate descent: A robotics algorithm for protein loop closure. *Protein Science*, 12(5):963–972, 2003. doi: 10.1110/ps.0242703.

Ricky T. Q. Chen and Yaron Lipman. Flow matching on general geometries. In *International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/file/d1f9936d3be6997ffffab692977eebe6-Paper-Conference.pdf.

Alexander E. Chu, Jinho Kim, Lucy Cheng, Gina El Nesr, Minkai Xu, Richard W. Shuai, and Po-Ssu Huang. An all-atom protein generative model. *Proceedings of the National Academy of Sciences*, 121(27):e2311500121, 2024. doi: 10.1073/pnas.2311500121. URL https://doi.org/10.1073/pnas.2311500121.

Rémi Coulom. Efficient selectivity and backup operators in Monte-Carlo tree search. In H. Jaap van den Herik, Paolo Ciancarini, and H. H. L. M. (Jeroen) Donkers (eds.), *Computers and Games*, volume 4630 of *Lecture Notes in Computer Science*, pp. 72–83, Berlin, Heidelberg, 2007. Springer Berlin Heidelberg. ISBN 978-3-540-75538-8. doi: 10.1007/978-3-540-75538-8_7.

Justas Dauparas, Ivan Anishchenko, Nathaniel Bennett, Hua Bai, Robert J Ragotte, Lukas F Milles, Basile IM Wicky, Alexis Courbet, Rob J de Haas, Neville Bethel, et al. Robust deep learning–based protein sequence design using proteinmpnn. *Science*, 378(6615):49–56, 2022.

Justas Dauparas, Gyu Rie Lee, Robert Pecoraro, Linna An, Ivan Anishchenko, Cameron Glasscock, and David Baker. Atomic context-conditioned protein sequence design using LigandMPNN. *Nature Methods*, 22(4):717–723, 2025. doi: 10.1038/s41592-025-02626-1.

Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, 2002. doi: 10.1109/4235.996017.

Keld Fosgerau and Torsten Hoffmann. Peptide therapeutics: current status and future directions. *Drug Discovery Today*, 20(1):122–128, 2015. doi: 10.1016/j.drudis.2014.10.003.

Markus Freitag and Yaser Al-Onaizan. Beam search strategies for neural machine translation. In Thang Luong, Alexandra Birch, Graham Neubig, and Andrew Finch (eds.), *Proceedings of the First Workshop on Neural Machine Translation*, pp. 56–60, Vancouver, August 2017. Association for Computational Linguistics. doi: 10.18653/v1/W17-3207. URL https://aclanthology.org/W17-3207/.

Gökçe Geylan, Jon Paul Janet, Alessandro Tibo, Jiazhen He, Atanas Patronov, Mikhail Kabeshov, Werngard Czechtizky, Florian David, Ola Engkvist, and Leonardo De Maria. PepINVENT: generative peptide design beyond natural amino acids. *Chemical Science*, 16(20):8682–8696, 2025. doi: 10.1039/D4SC07642G.

Ronald D. Gorham, Jr., David L. Forest, George A. Khoury, James Smadbeck, Consuelo N. Beecher, Evangeline D. Healy, Phanourios Tamamis, Georgios Archontis, Cynthia K. Larive, Christodoulos A. Floudas, Monte J. Radeke, Lincoln V. Johnson, and Dimitrios Morikis. New compstatin peptides containing N-terminal extensions and non-natural amino acids exhibit potent complement inhibition and improved solubility characteristics. *Journal of Medicinal Chemistry*, 58(2): 814–826, 2015. doi: 10.1021/jm501345y.

Mirja Harms, Rikke Fabech Hansson, Andrea Gilg, Yasser Almeida-Hernández, Jessica Löffler, Armando Rodríguez-Alfonso, Monica M. W. Habib, Dan Albers, Nermin S. Ahmed, Ashraf H. Abadi, Gordon Winter, Volker Rasche, Ambros J. Beer, Gilbert Weidinger, Nico Preising, Ludger Ständker, Sebastian Wiese, Elsa Sanchez-Garcia, Alexander N. Zelikin, and Jan Münch. Development of N-terminally modified variants of the CXCR4-antagonistic peptide EPI-X4 for enhanced plasma stability. *Journal of Medicinal Chemistry*, 66(22):15189–15204, 2023. doi: 10.1021/acs.jmedchem.3c01128.

Steven Henikoff and Jorja G. Henikoff. Amino acid substitution matrices from protein blocks. *Proceedings of the National Academy of Sciences*, 89(22):10915–10919, 1992. doi: 10.1073/pnas.89.22.10915.

John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, Alex Bridgland, Clemens Meyer, Simon A. A. Kohl, Andrew J. Ballard, Andrew Cowie, Bernardino Romera-Paredes, Stanislav Nikolov, Rishub Jain, Jonas Adler, Trevor Back, Stig Petersen, David Reiman, Ellen Clancy, Michal Zielinski, Martin Steinegger, Michalina Pacholska, Tamas Berghammer, Sebastian Bodenstein, David Silver, Oriol Vinyals, Andrew W. Senior, Koray Kavukcuoglu, Pushmeet Kohli, and Demis Hassabis. Highly accurate protein structure prediction with AlphaFold. *Nature*, 596(7873):583–589, 2021. doi: 10.1038/s41586-021-03819-2.

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.

Ozlem Keskin, Buyong Ma, and Ruth Nussinov. Hot regions in protein–protein interactions: the organization and contribution of structurally conserved hot spot residues. *Journal of Molecular Biology*, 345(5):1281–1294, 2005. doi: 10.1016/j.jmb.2004.10.077.

Levente Kocsis and Csaba Szepesvári. Bandit based Monte-Carlo planning. In *Machine Learning: ECML 2006*, volume 4212 of *Lecture Notes in Computer Science*, pp. 282–293, 2006. doi: 10.1007/11871842_29.

Xiangzhe Kong, Yinjun Jia, Wenbing Huang, and Yang Liu. Full-atom peptide design with geometric latent diffusion. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 74808–74839. Curran Associates, Inc., 2024. doi: 10.52202/079017-2379. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/88ad9774ffcb7a272457e9396f793a07-Abstract-Conference.html.

Brian Kuhlman and David Baker. Native protein sequences are close to optimal for their structures. *Proceedings of the National Academy of Sciences*, 97(19):10383–10388, 2000. doi: 10.1073/pnas.97.19.10383.

Jesper Lau, Paw Bloch, Lauge Schäffer, Ingrid Pettersson, Jane Spetzler, Jacob Kofoed, Kjeld Madsen, Lotte Bjerre Knudsen, James McGuire, Dorte Bjerre Steensgaard, Holger Martin Strauss, Dorte Xenia Gram, Sanne Møller Knudsen, Flemming Seier Nielsen, Peter Thygesen, Steffen Reedtz-Runge, and Thomas Kruse. Discovery of the once-weekly glucagon-like peptide-1 (GLP-1) analogue semaglutide. *Journal of Medicinal Chemistry*, 58(18):7370–7380, 2015. doi: 10.1021/acs.jmedchem.5b00726.

Andrew Leaver-Fay, Michael Tyka, Steven M. Lewis, Oliver F. Lange, James Thompson, Ron Jacak, Kristian W. Kaufman, P. Douglas Renfrew, Colin A. Smith, Will Sheffler, Ian W. Davis, Seth Cooper, Adrien Treuille, Daniel J. Mandell, Florian Richter, Yih-En Andrew Ban, Sarel J. Fleishman, Jacob E. Corn, David E. Kim, Sergey Lyskov, Monica Berrondo, Stuart Mentzer, Zoran Popović, James J. Havranek, John Karanicolas, Rhiju Das, Jens Meiler, Tanja Kortemme, Jeffrey J. Gray, Brian Kuhlman, David Baker, and Philip Bradley. ROSETTA3: An object-oriented software suite for the simulation and design of macromolecules. In *Methods in Enzymology*, volume 487, pp. 545–574. 2011. doi: 10.1016/B978-0-12-381270-4.00019-6.

Jin Sub Lee and Philip M Kim. FlowPacker: protein side-chain packing with torsional flow matching. *Bioinformatics*, 41(3):btaf010, 2025. doi: 10.1093/bioinformatics/btaf010.

Jiahan Li, Chaoran Cheng, Zuofan Wu, Ruihan Guo, Shitong Luo, Zhizhou Ren, Jian Peng, and Jianzhu Ma. Full-atom peptide design based on multi-modal flow matching. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 27615–27640, 2024. URL https://proceedings.mlr.press/v235/li24o.html.

Jiahan Li, Tong Chen, Shitong Luo, Chaoran Cheng, Jiaqi Guan, Ruihan Guo, Sheng Wang, Ge Liu, Jian Peng, and Jianzhu Ma. Hotspot-driven peptide design via multi-fragment autoregressive extension. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/1a0672689a693e0764f93f900488b3d9-Abstract-Conference.html.

Haitao Lin, Odin Zhang, Huifeng Zhao, Dejun Jiang, Lirong Wu, Zicheng Liu, Yufei Huang, and Stan Z. Li. PPFlow: Target-aware peptide design with torsional flow matching. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 30510–30528, 2024. URL https://proceedings.mlr.press/v235/lin24z.html.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=PqvMRDCJT9t.

Sidney Lyayuga Lisanza, Jacob Merle Gershon, Samuel W. K. Tipps, Jeremiah Nelson Sims, Lucas Arnoldt, Samuel J. Hendel, Miriam K. Simma, Ge Liu, Muna Yase, Hongwei Wu, Claire D. Tharp, Xinting Li, Alex Kang, Evans Brackenbrough, Asim K. Bera, Stacey Gerben, Bruce J. Wittmann, Andrew C. McShan, and David Baker. Multistate and functional protein design using RoseTTAFold sequence space diffusion. *Nature Biotechnology*, 43(8):1288–1298, 2025. doi: 10.1038/s41587-024-02395-w.

Shitong Luo, Yufeng Su, Xingang Peng, Sheng Wang, Jian Peng, and Jianzhu Ma. Antigen-specific antibody design and optimization with diffusion-based generative models for protein structures. In *Advances in Neural Information Processing Systems*, volume 35, pp. 9754–9767, 2022. doi: 10.52202/068431-0709. URL https://papers.neurips.cc/paper_files/paper/2022/hash/3fa7d76a0dc1179f1e98d1bc62403756-Abstract-Conference.html.

Robert T. McGibbon, Kyle A. Beauchamp, Matthew P. Harrigan, Christoph Klein, Jason M. Swails, Carlos X. Hernández, Christian R. Schwantes, Lee-Ping Wang, Thomas J. Lane, and Vijay S. Pande. MDTraj: A modern open library for the analysis of molecular dynamics trajectories. *Biophysical Journal*, 109(8):1528–1532, 2015. doi: 10.1016/j.bpj.2015.08.015.

Irina S. Moreira, Pedro A. Fernandes, and Maria J. Ramos. Hot spots—a review of the protein–protein interface determinant amino-acid residues. *Proteins: Structure, Function, and Bioinformatics*, 68(4):803–812, 2007. doi: 10.1002/prot.21396.

Markus Muttenthaler, Glenn F. King, David J. Adams, and Paul F. Alewood. Trends in peptide drug discovery. *Nature Reviews Drug Discovery*, 20(4):309–325, 2021. doi: 10.1038/s41573-020-00135-8.

Luís Paquete, Marco Chiarandini, and Thomas Stützle. Pareto local optimum sets in the biobjective traveling salesman problem: An experimental study. In *Metaheuristics for Multiobjective Optimisation*, volume 535 of *Lecture Notes in Economics and Mathematical Systems*, pp. 177–199. Springer, 2004. doi: 10.1007/978-3-642-17144-4_7.

Xingang Peng, Ruihan Guo, Fenglin Guo, Ziyi Wang, Jiayu Sun, Jiaqi Guan, Yinjun Jia, Yan Xu, Yanwen Huang, Muhan Zhang, Jian Peng, Xinquan Wang, Chuanhui Han, Zihua Wang, and Jianzhu Ma. Unified modeling of 3D molecular generation via atomic interactions with PocketX-Mol. *Cell*, 189(7):1904–1922.e28, 2026. doi: 10.1016/j.cell.2026.01.003.

Jason Phan, Zhenyu Li, Agnieszka Kasprzak, Baozong Li, Said Sebti, Wayne Guida, Ernst Schönbrunn, and Jiandong Chen. Structure-based design of high affinity peptides inhibiting the interaction of p53 with MDM2 and MDMX. *Journal of Biological Chemistry*, 285(3):2174–2183, 2010. doi: 10.1074/jbc.M109.073056.

Hao Qian, Shikui Tu, and Lei Xu. Full-atom peptide design via Riemannian–Euclidean Bayesian flow networks. *Proceedings of the AAAI Conference on Artificial Intelligence*, 40(29):24918–24926, 2026a. doi: 10.1609/aaai.v40i29.39679.

Hao Qian, Pu You, Lin Zeng, Jingyuan Zhou, Dengdeng Huang, Kaicheng Li, Shikui Tu, and Lei Xu. Generative design and validation of therapeutic peptides for glioblastoma based on a potential target ATP5A. *Briefings in Bioinformatics*, 27(2):bbag188, 2026b. doi: 10.1093/bib/bbag188.

Barak Raveh, Nir London, and Ora Schueler-Furman. Sub-angstrom modeling of complexes between flexible peptides and globular proteins. *Proteins: Structure, Function, and Bioinformatics*, 78(9):2029–2040, 2010. doi: 10.1002/prot.22716.

Amelie Stein and Tanja Kortemme. Improvements to robotics-inspired conformational sampling in Rosetta. *PLoS ONE*, 8(5):e63090, 2013. doi: 10.1371/journal.pone.0063090.

Nisan Stiennon, Long Ouyang, Jeffrey Wu, Daniel Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul F Christiano. Learning to summarize with human feedback. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 3008–3021. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/1f89885d556929e98d3ef9b86448f951-Paper.pdf.

Lei Wang, Nanxi Wang, Wenping Zhang, Xurui Cheng, Zhibin Yan, Gang Shao, Xi Wang, Rui Wang, and Caiyun Fu. Therapeutic peptides: current applications and future directions. *Signal Transduction and Targeted Therapy*, 7(1):48, 2022. doi: 10.1038/s41392-022-00904-4.

Luran Wang, Chaoran Cheng, Yizhen Liao, Yanru Qu, and Ge Liu. Training free guided flow-matching with optimal control. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/19a94fdf9e1c5b387830e4c4fef6972a-Abstract-Conference.html.

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.

Hong Wei, Wenkai Wang, Zhenling Peng, and Jianyi Yang. Q-BioLiP: a comprehensive resource for quaternary structure-based protein–ligand interactions. *Genomics, Proteomics & Bioinformatics*, 22(1):qzae001, 2024. doi: 10.1093/gpbjnl/qzae001.

Zeyu Wen, Jiahua He, Huanyu Tao, and Sheng-You Huang. PepBDB: a comprehensive structural database of biological peptide–protein interactions. *Bioinformatics*, 35(1):175–177, 2019. doi: 10.1093/bioinformatics/bty579.

Wenjing Xiao, Wenjie Jiang, Zheng Chen, Yu Huang, Junyi Mao, Wei Zheng, Yonghe Hu, and Jianyou Shi. Advance in peptide-based drug development: delivery platforms, therapeutics and vaccines. *Signal Transduction and Targeted Therapy*, 10:74, 2025. doi: 10.1038/s41392-024-02107-5.

Yinghua Yao, Yuangang Pan, and Xixian Chen. Generative co-design of antibody sequences and structures via black-box guidance in a shared latent space. In *Proceedings of the Thirty-Fourth International Joint Conference on Artificial Intelligence*, pp. 9447–9455. International Joint Conferences on Artificial Intelligence Organization, 2025. doi: 10.24963/ijcai.2025/1050. URL https://www.ijcai.org/proceedings/2025/1050.

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.

Yinuo Zhang, Sophia Tang, Tong Chen, Elizabeth Mahood, Sophia Vincoff, and Pranam Chatterjee. PeptiVerse: A unified platform for therapeutic peptide property prediction. *Nature Communications*, 17(1):6819, 2026. doi: 10.1038/s41467-026-74167-w.

## APPENDIX CONTENTS

**A Related work** . . . **17**
A.1 Generative models for protein design . . . 17
A.2 Computational peptide design . . . 17
A.3 Property-oriented peptide optimization . . . 17
**B Completeness of the residue–variable edit language** . . . **17**
**C Full mathematical specification** . . . **18**
C.1 State space, restrictions, and conditional law . . . 18
C.2 Masked multimodal path and endpoint field . . . 19
**D Training and implementation details** . . . **20**
**E Conditional sampling details** . . . **22**
E.1 Selecting the window schedule . . . 23
**F Property-guided optimization algorithms** . . . **23**
F.1 Constrained multi-objective selection . . . 23
F.2 Four search policies . . . 25
**G Baselines and experimental protocols** . . . **26**
G.1 Baselines . . . 26
G.2 Evaluation set and shared settings . . . 27
G.3 Pocket-conditioned generation . . . 27
G.4 Joint redesign . . . 27
G.5 Peptide completion from sparse anchors . . . 28
G.6 Single-objective optimization . . . 28
G.7 Multi-objective optimization . . . 28
G.8 Evaluation metrics . . . 29
**H Additional reconstruction analyses** . . . **31**
H.1 Fixed-backbone sequence design . . . 31
H.2 Fixed-sequence backbone reconstruction . . . 32
H.3 Joint sequence–structure reconstruction examples . . . 32
H.4 Interface side-chain repacking . . . 32
H.5 Sparse-anchor completion metrics . . . 34
**I Additional property optimization results** . . . **35**
I.1 Single-objective results . . . 35
I.2 Multi-objective results . . . 38
I.3 Individual-property endpoints in multi-objective search . . . 39
I.4 Representative multi-objective complexes . . . 39
**J Molecular details of the GRB2 SH3 editing example** . . . **39**
J.1 An additional case using window refinement . . . 41

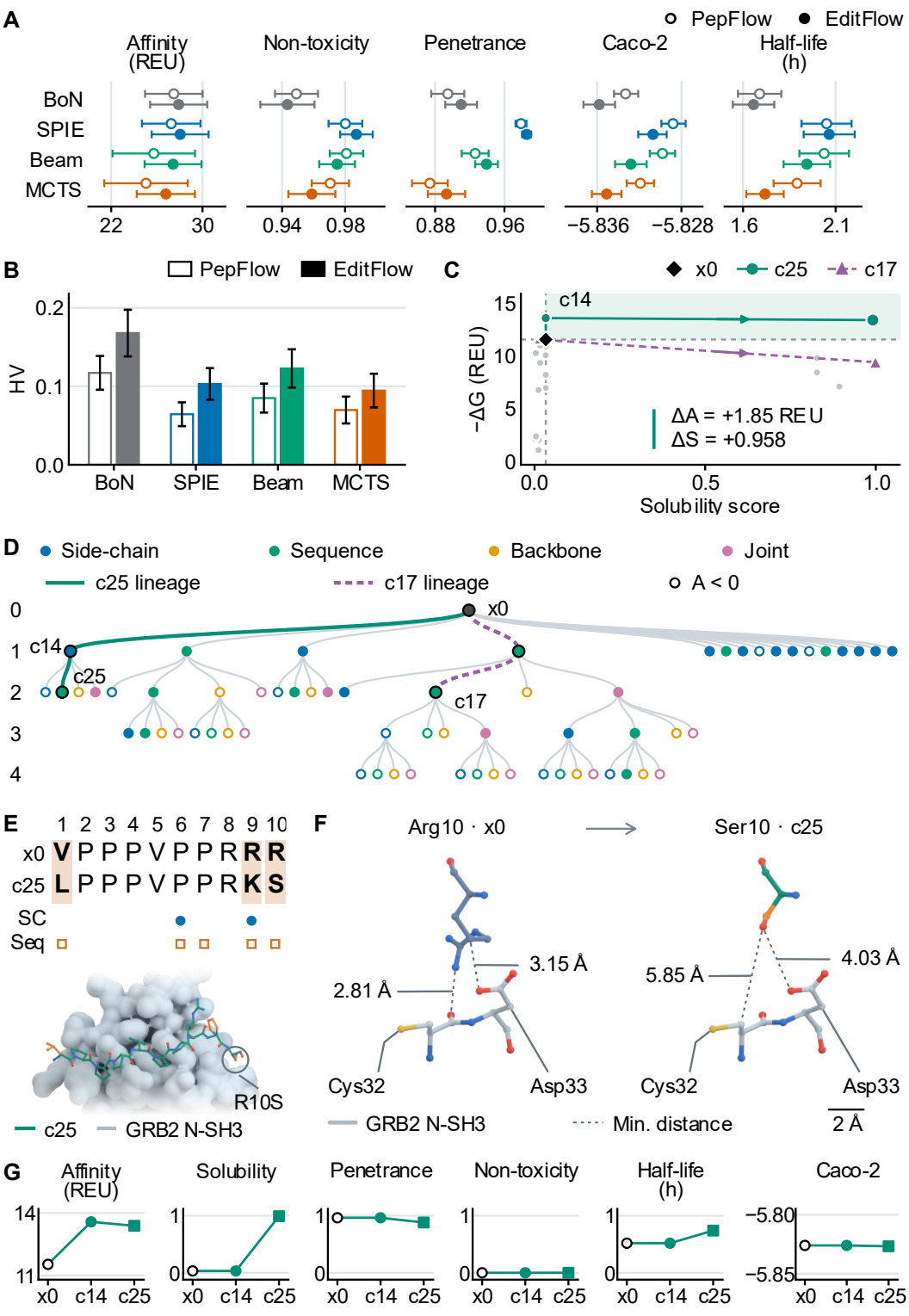



Figure 5: Additional optimization comparisons and the GRB2 SH3 editing case. (A,B) Single-objective scores and five-objective HV obtained with residue-wise extension, the simpler completion rule defined in Appendix G.7, over the benchmark complexes with 64 proposals and 95% intervals. (C) Trade-offs for 28 nonnegative-score proposals and the initial peptide. (D) All 64 proposals, where open nodes have negative interface scores. (E–G) Sequence and structure changes, local contacts, and properties along the accepted trajectory. Candidate numbers indicate proposal order. The interface score is the negative Rosetta interface energy, and sequence-derived properties are predictions.

# A Related work

## A.1 Generative models for protein design

Deep generative models have become the dominant tool for designing functional proteins. One line generates sequence alone, using protein language models or directed-evolution surrogates to propose variants that a downstream predictor then filters. A second designs sequence onto a given backbone, which is the setting of the inverse-folding models ProteinMPNN and LigandMPNN (Dauparas et al., 2022; 2025), and of side-chain placement methods such as FlowPacker that complete torsions on a fixed sequence (Lee & Kim, 2025). A third generates structure directly, where diffusion and flow models now produce backbones and full-atom states both unconditionally and conditioned on a binding partner (Luo et al., 2022; Watson et al., 2023; Lisanza et al., 2025; Chu et al., 2024). What unites the second and third lines is that each fixes in advance which part of the molecule is given and which is generated, so remodeling a backbone, redesigning a segment, and repacking an interface remain separate procedures with separate models even when they act on one complex and a designer wants to interleave them.

## A.2 Computational peptide design

Peptide design inherited its earliest methods from template matching and energy minimization, searching the PDB for sequences or structures to seed a design and then scoring candidates with hand-crafted or statistical energies (Raveh et al., 2010; Leaver-Fay et al., 2011; Alford et al., 2017). These procedures are expensive and readily trapped in local minima, which motivated the move to learned conditional distributions. PepFlow models full-atom peptides with multi-modal flow matching on a Riemannian manifold, PepGLAD generates full-atom structures through equivariant latent diffusion, PepBFN places identity and geometry in one Bayesian flow state, and PepHAR samples hot-spot residues from an energy-based density and extends fragments autoregressively through dihedral angles (Li et al., 2024; Kong et al., 2024; Qian et al., 2026a; Li et al., 2025). Retained molecular information enters these models in method-specific ways, as partial sampling over a subset of variables, as fixed anchor residues that seed an extension, or as atom-level constraints on type, position, and bonds (Li et al., 2024; 2025; Peng et al., 2026). The retained information is thus determined by the formulation rather than named by the user, and no single request can say both which residues to revise and which of their identity, backbone, and torsion channels to release.

## A.3 Property-oriented peptide optimization

Improving a lead differs from sampling a conditional distribution, because the target is a property score rather than agreement with a reference structure. PPFlow perturbs an existing peptide before regenerating it, and POTFlow builds its prior from the lead itself and couples that prior to the data distribution by optimal transport (Lin et al., 2024; Qian et al., 2026b). PepINVENT combines residue completion with reinforcement learning over a chemical reward (Geylan et al., 2025), OC-Flow casts guided generation as an optimal control problem (Wang et al., 2025), and LEAD applies black-box property guidance inside a shared antibody sequence–structure latent space without retraining (Yao et al., 2025). Each of these binds the optimizer to the generator, since the objective enters as a guidance term, a reward, or a control cost, so a new property or a different search policy generally calls for a new guidance signal or another round of training.

# B Completeness of the residue–variable edit language

This section shows that any dependency-closed edit mask in the fixed-length state space of Eq. (1) can be represented by a finite union of residue–variable instructions $\mathcal{R} \times \mathcal{V}$. Let $\mathcal{C} = \{a, F, \psi, \chi_1, \ldots, \chi_4\}$ and $\mathcal{C}_\chi = \{\chi_1, \ldots, \chi_4\}$. For a requested set $S \subseteq \{1, \ldots, L\} \times \mathcal{C}$, define

$$\mathrm{closure}(S) = S \cup \bigcup_{i:(i,a)\in S} (\{i\} \times \mathcal{C}_\chi) \cup \bigcup_{i:(i,F)\in S} \big(\{i\} \times (\{\psi\} \cup \mathcal{C}_\chi)\big). \tag{13}$$

These rules allow side chains to adapt to sequence changes and torsions to adapt to backbone changes. The operator is extensive, monotone, and idempotent: it adds the required channels, preserves set inclusion, and adds nothing further when applied twice. A mask $M$ is *dependency closed* when $\mathrm{closure}(M) = M$.

Different positions can require different operations. For example, sequence redesign at one position and backbone remodeling at another require different variable sets. For an instruction collection $\mathcal{Q}$, extend Eq. (2) by combining its requests before applying closure:

$$M^{\mathrm{req}} = \bigcup_{(\mathcal{R},\mathcal{V})\in\mathcal{Q}} (\mathcal{R}\times\mathcal{V}), \qquad M = \mathrm{closure}(M^{\mathrm{req}}). \tag{14}$$

Both a single instruction and a collection therefore produce the same form of effective mask used by the model.

**Proposition 1** (Coverage of dependency-closed masks)**.** *Every dependency-closed mask over $\mathcal{C}$ can be represented as the closure of a finite union of residue–variable instructions, with at most one instruction per channel.*

*Proof.* For any dependency-closed mask $M^\dagger$, collect the positions selected in each channel:

$$\mathcal{R}_v = \{i : (i,v) \in M^\dagger\}, \qquad M^\dagger = \bigcup_{v\in\mathcal{C}} (\mathcal{R}_v \times \{v\}). \tag{15}$$

Discarding empty sets gives at most $|\mathcal{C}| = 7$ instructions. Their union equals $M^\dagger$, and applying closure leaves it unchanged because $M^\dagger$ is dependency closed.

The result characterizes the masks expressible in the defined state space; training samples the four canonical operations over multiscale residue regions.

## C Full mathematical specification

This section makes explicit the state space, conditional paths, and preservation properties used by the method. The scope is the fixed-length peptide state in Eq. (1); Appendix B separately establishes the coverage of the residue–variable instruction language.

### C.1 State space, restrictions, and conditional law

**Full-atom state space.** Let $\mathcal{T} = \{1,\dots,20\}$ be the residue alphabet and $\mathcal{C} = \{a, F, \psi, \chi_1, \dots, \chi_4\}$, where $F = (p, r) \in \mathbb{R}^3 \times SO(3)$. The ambient peptide state space is

$$\mathcal{X}_L = \prod_{i=1}^{L} \left(\mathcal{T} \times \mathbb{R}^3 \times SO(3) \times \mathbb{T}^5\right). \tag{16}$$

Type-specific validity masks, read from a fixed topology table, remove torsion slots that do not exist for a given residue type. A dependency-closed edit set $M \subseteq \{1,\dots,L\}\times\mathcal{C}$ induces the restrictions $X_M = (X_{iv})_{(i,v)\in M}$ and $X_{\bar{M}}$.

**Canonical editing operations.** Table 4 lists the variable sets illustrated in Figure 3. Editable and fixed variables are specified within the selected region $\mathcal{R}$; the receptor and all peptide variables outside $\mathcal{R}$ remain fixed.

Table 4: Canonical operations within the selected region $\mathcal{R}$, after dependency closure. All variables outside $\mathcal{R}$ remain fixed.

| Operation | Editable variables | Fixed variables |
|---|---|---|
| Side-chain packing | $\chi$ | $a, F, \psi$ |
| Sequence redesign | $a, \chi$ | $F, \psi$ |
| Backbone remodeling | $F, \psi, \chi$ | $a$ |
| Joint redesign | $a, F, \psi, \chi$ | None |

**Structural examples.** Figure 3 uses receptor chains A/B and peptide chain P of PDB 3MRE. All panels edit residues P4–P6. Panels (a) and (c) retain QAS, and panels (b) and (d) replace QAS with TLV. The edited states are constructed from the experimental structure to illustrate the variable constraints. All four edits preserve the receptor and peptide atoms outside the selected region.

**Projection and merge.** For two compatible peptide states $A, B \in \mathcal{X}_L$, define

$$\big[\mathrm{Merge}_M(A,B)\big]_{iv} = \begin{cases} B_{iv}, & (i,v) \in M, \\ A_{iv}, & (i,v) \notin M. \end{cases} \tag{17}$$

Thus $\pi_M \,\mathrm{Merge}_M(A,B) = \pi_M B$ and $\pi_{\bar{M}} \,\mathrm{Merge}_M(A,B) = \pi_{\bar{M}} A$. The target edited-data law is the regular conditional distribution

$$q_M(dx_M \mid P, x_{\bar{M}}, M), \tag{18}$$

and EditFlow approximates it by the distribution $p_\theta(dx_M \mid P, x_{\bar{M}}, M)$ induced by prior sampling and the learned multimodal flow. This notation states the learning target; it does not assume that the sequence and geometric variables factorize. The staged sampler composes several model calls with updated context and window perturbations. Its proposal law can depend on intermediate states and is not asserted to equal $p_\theta$ or preserve $q_M$ as a stationary distribution.

### C.2 Masked multimodal path and endpoint field

Let $z = c(a) \in \mathbb{R}^{20}$ be the continuous type code and $\tau = (\psi, \chi) \in \mathbb{T}^5$. For an editable entry, the interpolation $\mathcal{I}_v(x_0, x_1; t)$ is the Euclidean line for $z$ and $p$, the shortest geodesic for $r \in SO(3)$, and the shortest periodic arc for $\tau$. The masked path is

$$\Phi_t^M(Z, X^\star)_{iv} = \begin{cases} \mathcal{I}_v(Z_{iv}, X^\star_{iv}; t), & (i,v) \in M, \\ X^\star_{iv}, & (i,v) \notin M. \end{cases} \tag{19}$$

The priors are Gaussian for $z$ and $p$, uniform on $SO(3)$ for $r$, and uniform on $\mathbb{T}$ for each torsion. Translation noise is centered over valid peptide residues before the selected frame channels are read out.

For clarity, the endpoint-induced velocity for the continuous representations can be written in the appropriate tangent coordinates as

$$\begin{aligned} u_t^z &= \frac{z_1 - z_t}{1-t}, \qquad u_t^p = \frac{p_1 - p_t}{1-t}, \\ u_t^r &= \frac{\mathrm{Log}(r_t^{\mathsf{T}} r_1)}{1-t}, \quad u_t^\tau = \frac{\mathrm{wrap}(\tau_1 - \tau_t)}{1-t}. \end{aligned} \tag{20}$$

In the endpoint-interpolation sampler, the network replaces the unknown endpoint by $\widehat{X}_1$. With $\alpha = \Delta t/(1-t)$, one solver step applies $\mathcal{I}_v((X_t)_{iv}, (\widehat{X}_1)_{iv}; \alpha)$ on editable continuous variables, samples current residue types from the type probabilities, and then uses Eq. (17). The final evaluation directly decodes the predicted endpoint.

**Proposition 2** (Masked-path and update consistency)**.** *Let $X_t = \Phi_t^M(Z, X^\star)$ be the masked interpolation path. Then*

$$(X_t)_{\bar{M}} = X^\star_{\bar{M}} \quad (t \in [0,1]), \qquad (X_0)_M = Z_M, \qquad X_1 = X^\star. \tag{21}$$

*For a continuous editable variable $v$, suppose the predicted endpoint equals $X^\star_{iv}$. With $0 \le t < 1$, $0 \le \Delta t \le 1 - t$, and $\alpha = \Delta t/(1-t)$,*

$$\mathcal{I}_v((X_t)_{iv}, X^\star_{iv}; \alpha) = (X_{t+\Delta t})_{iv}, \qquad (i,v) \in M, \tag{22}$$

*where rotational and angular interpolations follow a consistent shortest path away from the cut locus.*

*Proof.* The fixed-state and endpoint identities follow directly from Eq. (19). Along each selected interpolation path,

$$\mathcal{I}_v(\mathcal{I}_v(Z_{iv}, X^\star_{iv}; t), X^\star_{iv}; \alpha) = \mathcal{I}_v(Z_{iv}, X^\star_{iv}; t + \alpha(1-t)). \tag{23}$$

Substituting $\alpha = \Delta t/(1-t)$ gives $t + \alpha(1-t) = t + \Delta t$, establishing the update identity.

**Proposition 3** (Preservation of fixed variables)**.** *Let $M$ be the edit mask and $U_k$ an arbitrary provisional update. If*

$$(X_0)_{\bar{M}} = X^{\mathrm{par}}_{\bar{M}}, \qquad X_{k+1} = \mathrm{Merge}_M(X^{\mathrm{par}}, U_k(X_k)), \tag{24}$$

*then every sampling state preserves the fixed parent variables:*

$$(X_k)_{\bar{M}} = X^{\mathrm{par}}_{\bar{M}}, \qquad k \geq 0. \tag{25}$$

*Proof.* The initialization gives the result for $k = 0$. For every $k \geq 0$, the merge definition yields

$$(X_{k+1})_{\bar{M}} = [\mathrm{Merge}_M(X^{\mathrm{par}}, U_k(X_k))]_{\bar{M}} = X^{\mathrm{par}}_{\bar{M}}. \tag{26}$$

Thus the fixed variables are preserved at every step, regardless of the provisional update.

## D Training and implementation details

**Network architecture.** EditFlow uses a shared geometric network, following the full-atom peptide architecture of Li et al. (2024), to predict editable residue types and structures from the protein pocket, observed peptide variables, and current noisy state. Static residue features encode residue types, local atomic coordinates, and backbone dihedral angles. Pair features encode residue-type pairs, relative sequence positions, interatomic distances, and pairwise backbone dihedral angles. Features associated with editable variables are masked before encoding. Residue and pair representations have dimensions 128 and 64, respectively.

The decoder combines these representations with the current residue types, torsion encodings, and time embedding. Its geometric trunk contains six blocks, each comprising invariant point attention (Jumper et al., 2021), two sequence-transformer layers, a node transition, and a backbone-frame update. Invariant point attention uses eight heads, with eight query/key points and twelve value points per head; sequence attention uses four heads. The final backbone frames provide translation and rotation predictions, while separate multilayer perceptrons predict the distribution over 20 residue types and five torsion angles. All editing operations share this network.

**Training protocol.** EditFlow is trained for 300,000 optimizer steps on a single NVIDIA A100 GPU with 80 GB memory. The per-device batch size is 4 protein–peptide complexes, with 2 gradient-accumulation steps per optimizer update, giving an effective batch size of 8 complexes. Each complex supplies five conditional tasks: one *de novo* branch and four local editing branches. Training uses Adam with an initial learning rate of $10^{-5}$. A piecewise cosine schedule reduces the learning rate to $5 \times 10^{-6}$ at step 15,000 and $2 \times 10^{-6}$ at step 30,000, after which it remains constant. The global gradient norm is clipped at 100. The random seed is 114514, and evaluation uses the checkpoint at step 300,000.

**Type codes and noise.** For $a \in \mathcal{T} = \{1, \ldots, 20\}$, the continuous type encoding $c(a) \in \mathbb{R}^{20}$ and its prior are

$$[c(a)]_j = \begin{cases} 5, & j = a, \\ -5, & j \neq a, \end{cases} \qquad j = 1, \ldots, 20, \qquad z_0 \sim \mathcal{N}(0, 25 I_{20}). \tag{27}$$

Translation noise is drawn with standard deviation 1 in the stored coordinate units, then centered over all valid peptide residues before selecting local frame channels. This centering introduces correlations across residues; the translation prior is not a product of independent residue priors after centering. Rotations are sampled uniformly on $SO(3)$ and each torsion uniformly on the circle. For each complex, $t \sim \mathcal{U}(0.01, 0.99)$. The five branches share $t$, base noise, and the uniform random draws used to sample current residue types from the interpolated type probabilities.

**Region construction.** For a peptide of length $L$, the editable residue count is sampled as

$$u \sim \mathcal{U}(0, 1), \qquad \rho = \sin^2\Big(\frac{\pi u}{2}\Big), \qquad \ell = \min\{L, 1 + \lfloor L\rho \rfloor\}. \tag{28}$$

The induced U-shaped distribution emphasizes short edits and near-complete reconstruction. If $\ell = L$, all peptide positions are selected. Otherwise, the number of sampling blocks $b$ is drawn

from $\{1, \ldots, \min(3, \ell, L-\ell)\}$ with weights $(0.60, 0.25, 0.15)$ restricted to the feasible choices and renormalized.

The editable and retained counts are independently partitioned into $b$ positive integer parts, $m_j$ and $n_j$, by uniformly selecting cut points:

$$\sum_{j=1}^{b} m_j = \ell, \qquad \sum_{j=1}^{b} n_j = L - \ell, \qquad L_j = m_j + n_j. \tag{29}$$

After a uniformly sampled cyclic offset, the ordered peptide positions are divided into disjoint blocks of lengths $L_j$. Within block $j$, a uniformly sampled start selects a cyclic window of $m_j$ positions. The union contains exactly $\ell$ editable positions. Here $b$ counts sampling blocks; cyclic windows can produce a different number of contiguous segments in the original linear sequence.

**Losses.** Each loss averages prediction errors over the editable entries of one complex. For an error $e$ and a supervised set $\Omega$, write

$$\langle e \rangle_\Omega = \frac{\sum_{(i,v)\in\Omega} e_{iv}}{|\Omega| + 10^{-8}}. \tag{30}$$

Let $\Omega_a$, $\Omega_F$, and $\Omega_\tau$ denote the editable type, frame, and valid torsion entries. Type prediction uses cross-entropy, translation uses squared endpoint error, and rotation compares tangent displacements from the current frame. With $s(t) = [1 - \min(t, 0.9)]^{-1}$,

$$\begin{aligned} \mathcal{L}^M_{\text{type}} &= \langle -\log \widehat{\pi}_1(a^\star) \rangle_{\Omega_a}, & \mathcal{L}^M_{\text{trans}} &= \left\langle \|\widehat{p}_1 - p^\star\|_2^2 \right\rangle_{\Omega_F}, \\ \mathcal{L}^M_{\text{rot}} &= \left\langle s(t)^2 \| \operatorname{Log}(r_t^\mathsf{T} \widehat{r}_1) - \operatorname{Log}(r_t^\mathsf{T} r^\star) \|_2^2 \right\rangle_{\Omega_F}, \end{aligned} \tag{31}$$

For torsions, define the periodic error $e_\tau = 1 - \cos(\widehat{\tau}_1 - \tau^\star)$, applied to each angle. This equals the half squared sine–cosine error used in the implementation. The torsional flow and endpoint losses differ by the time weight $s(t)^2$; the backbone loss compares reconstructed N/C$\alpha$/C coordinates:

$$\begin{aligned} \mathcal{L}^M_{\text{tor-flow}} &= \left\langle s(t)^2 e_\tau \right\rangle_{\Omega_\tau}, & \mathcal{L}^M_{\text{tor-end}} &= \langle e_\tau \rangle_{\Omega_\tau}, \\ \mathcal{L}^M_{\text{bb}} &= \left\langle \sum_{q \in \{\text{N},\text{C}\alpha,\text{C}\}} \|\widehat{y}_{1,q} - y^\star_q\|_2^2 \right\rangle_{\Omega_F}. \end{aligned} \tag{32}$$

Here $\widehat{y}_{1,q}$ is reconstructed from $(\widehat{p}_1, \widehat{r}_1)$. Valid torsion slots follow a sampled predicted endpoint type at type-editable positions and the observed type elsewhere; their targets are the corresponding native torsions.

For branch mask $M$, we use

$$\mathcal{L}_M = \mathcal{L}^M_{\text{type}} + \tfrac{1}{2}\mathcal{L}^M_{\text{trans}} + \tfrac{1}{2}\mathcal{L}^M_{\text{rot}} + \mathcal{L}^M_{\text{tor-flow}} + \tfrac{1}{4}\mathcal{L}^M_{\text{bb}} + \tfrac{1}{2}\mathcal{L}^M_{\text{tor-end}}. \tag{33}$$

The five branch losses are combined using Eq. (7) and averaged over the minibatch.

**Algorithm 1: Training the conditional peptide editor**

**Input:** Native minibatch $\{(P_b, X^\star_b)\}_{b=1}^B$; model $f_\theta$
**for** each complex $b$ **do**
  Sample $t_b$, a multiscale region $\mathcal{R}_b$, and base noise $Z_b$.
  Set $M_{b,0}$ to the full-peptide mask.
  Build $M_{b,1:4}$ from the four operations in Table 4 on $\mathcal{R}_b$.
  **for** $j = 0, \ldots, 4$ **do**
    Form $X^{(j)}_{t_b} = \Phi^{M_{b,j}}_{t_b}(Z_b, X^\star_b)$.
    Hide static features selected by $M_{b,j}$.
Predict all branch endpoints in one shared network batch.
Compute $\mathcal{L}_{M_{b,j}}$ using Eq. (33).
$\mathcal{L} \leftarrow B^{-1} \sum_b [\frac{1}{2}\mathcal{L}_{M_{b,0}} + \frac{1}{8}\sum_{j=1}^4 \mathcal{L}_{M_{b,j}}]$.
Update $\theta$ with $\nabla_\theta \mathcal{L}$.

# E CONDITIONAL SAMPLING DETAILS

**Endpoint interpolation.** For a step from $t$ to $t + \Delta t$, let $\alpha = \Delta t/(1 - t)$. Type codes and translations move by $\alpha$ times their difference from the predicted endpoint. Rotations and torsions move by the corresponding fraction along the rotation geodesic and shortest periodic path. The final network evaluation returns the merged endpoint directly. Current residue types are sampled from the type-code probabilities between updates; the last evaluation uses an endpoint type or categorical draw according to the stage. Every update preserves variables outside its stage mask. The original joint flow-integration protocol is specified separately below.

**Algorithm 2: Preservation-constrained endpoint interpolation**

**Input:** Pocket $P$; parent $X^{\mathrm{par}}$; requested instruction $(\mathcal{R}, \mathcal{V})$; time grid $t_{0:N-1}$ with $N \geq 1$
**Output:** One edited complex $X'$

$M \leftarrow \mathrm{closure}(\mathcal{R} \times \mathcal{V})$.
Sample priors on $M$ and set $X_{t_0} \leftarrow \mathrm{Merge}_M(X^{\mathrm{par}}, Z)$.
**for** $n = 0, \ldots, N - 2$ **do**
  Rebuild leakage-free conditions from $P$, $X^{\mathrm{par}}_M$, and $X_{t_n,M}$.
  $\widehat{X}_1 \leftarrow f_\theta(X_{t_n}, t_n, P, X^{\mathrm{par}}_M, M)$.
  $\widetilde{X} \leftarrow \mathcal{U}_{t_n, t_{n+1}-t_n}(X_{t_n}, \widehat{X}_1)$.
  $X_{t_{n+1}} \leftarrow \mathrm{Merge}_M(X^{\mathrm{par}}, \widetilde{X})$.
Predict the final endpoint at $t_{N-1}$ and merge it with the fixed parent state.
$X' \leftarrow$ reconstruct authorized atoms and copy all remaining atoms from the parent.
**return** $X'$.

**Direct joint flow integration.** Full generation and the matched-budget optimization comparison integrate the joint flow directly with 200 evaluations. For translations and type codes, the increment is $\Delta t(\widehat{x}_1 - x_0)$, where $x_0$ is the initial prior value. Rotations move toward the predicted endpoint with geodesic fraction $10\Delta t$, and torsions with fraction $\Delta t$. A final evaluation at $t = 1$ returns the masked endpoint. These updates differ from the $\Delta t/(1 - t)$ endpoint-interpolation rule. Both procedures apply the same parent merge, so Proposition 3 holds independently of the chosen discretization.

**Sampling with window refinement.** Both initial completion and window refinement first update the permitted variables jointly, then fix the backbone and resample the permitted residue types and side chains. Let $\mathcal{J}_\theta(X; D, s, \eta)$ initialize variables in $D$ by $\mathcal{I}(Z, X; \eta)$ and integrate their joint flow from time $s$. Let $\mathcal{S}_\theta(X; D, s)$ resample permitted types and side chains from fresh priors, starting at $s$. With the pocket condition implicit, the operators in Eq. (9) are

$$\begin{aligned} \mathcal{C}_\theta(X; M) &= \mathcal{S}_\theta(\mathcal{J}_\theta(X; M, s_{\mathrm{init}}, 0); M_{\mathrm{seq}}, s_{\mathrm{init}}) , \\ \mathcal{R}_\theta(X; W) &= \mathcal{S}_\theta(\mathcal{J}_\theta(X; W, s_{\mathrm{win}}, s_{\mathrm{win}}); W \cap M_{\mathrm{seq}}, s_{\mathrm{min}}) . \end{aligned} \tag{34}$$

Here $M_{\mathrm{seq}} = M \cap (\{1, \ldots, L\} \times \{a, \chi_1, \ldots, \chi_4\})$, $s_{\mathrm{min}} = 0.01$, and $s_{\mathrm{win}} = 0.5$. Completion uses $s_{\mathrm{init}} = s_{\mathrm{min}}$; property search chooses it using Eq. (39). This composition uses generated intermediate states and is not asserted to sample Eq. (3) exactly.

When the instruction retains at least one peptide frame, translation priors are centered on the retained frame positions; when every frame is released, they fall back to the origin of the benchmark coordinate frame. The default stages are: (i) 16 joint evaluations from 0.01 to 0.99; (ii) 48 sequence/side-chain evaluations after fixing the generated frames and $\psi$; and (iii) width-five local windows in forward and reverse order. Each window starts with partial noise at $t = 0.5$, performs four joint evaluations, then uses fresh type and side-chain noise for 12 sequence evaluations. Conditional features are rebuilt between stages. At each stage, Eq. (8) preserves the variables outside the active mask. No geometric projection or property-based selection is performed inside this trajectory.

**Algorithm 3: Sparse-anchor completion with window refinement**

**Input:** Pocket $P$; parent $X^{\mathrm{par}}$; closed mask $M$ with at least one retained frame
**Output:** One completed peptide $X'$

$X \leftarrow \mathrm{JointFlow}(P, X^{\mathrm{par}}, M)$.
$M_{\mathrm{seq}} \leftarrow M \cap (\{1, \ldots, L\} \times \{a, \chi_1, \ldots, \chi_4\})$.
Reinitialize $M_{\mathrm{seq}}$ with fresh prior noise and set $X \leftarrow \mathrm{SequenceFlow}(P, X, M_{\mathrm{seq}})$.
Construct all nonempty width-five, stride-one window masks $W_q = M \cap (W_q^{\mathrm{res}} \times \mathcal{C})$.
**for** $W$ in the forward order followed by the reverse order **do**
  Partially re-noise $X_W$.
  $X \leftarrow \mathrm{JointFlow}(P, X, W)$.
  $W_{\mathrm{seq}} \leftarrow W \cap (\{1, \ldots, L\} \times \{a, \chi_1, \ldots, \chi_4\})$.
  Reinitialize $W_{\mathrm{seq}}$ with fresh prior noise.
  $X \leftarrow \mathrm{SequenceFlow}(P, X, W_{\mathrm{seq}})$.
**return** the reconstructed and parent-merged state $X'$.

**Cost and scope.** If $N_{\mathrm{win}}$ nonempty windows are visited per direction, one editable proposal uses

$$\mathrm{NFE} = 64 + 32N_{\mathrm{win}}. \tag{35}$$

A completely fixed sample returns its input without a network evaluation. Full generation opens all peptide variables and uses the joint flow without window refinement. Search policies may change the starting time of the global stages; the window restart remains 0.5. Proposal count, network evaluations per proposal, and accepted edit depth are therefore distinct costs.

### E.1 Selecting the window schedule

We selected the schedule on a development set of 12 complexes with eight samples per complex and condition. Table 5 reports means over the $K = 1, 2, 3$ sparse-anchor conditions. Each stage varies one factor from the configuration retained by the preceding stage, and we froze the retained configuration before the benchmark evaluations.

Two sweeps over width-five windows give lower C$\alpha$ RMSD and higher validity than a single sweep, than width seven, and than a third sweep, which raises the cost by 39% and reduces validity. Re-noising each window to $t = 0.5$ rather than to the prior then lowers RMSD from 3.376 to 2.877 Å and raises binding-site recovery from 77.00 to 80.72%, because a partially re-noised window starts from the geometry the parent already has. The same step narrows sequence and structural diversity from 36.71 to 30.03%. Moving the per-window budget from joint updates to sequence and side-chain updates restores that diversity to 37.78% and improves all four reported quantities relative to joint-only windows, giving the four joint and twelve sequence updates used throughout.

## F Property-guided optimization algorithms

### F.1 Constrained multi-objective selection

Let the external evaluator return objectives $\mathbf{s}(X) = (s_1(X), \ldots, s_m(X))$, all oriented so that larger is better, together with constraint violations $g(X)$. Define feasibility and total violation by

$$\mathrm{feas}(X) = \mathbf{1}[g_j(X) \le 0 \,\forall j], \qquad \nu(X) = \sum_j \max\{g_j(X), 0\}. \tag{36}$$

A candidate dominates another if it is feasible while the other is not; if both are infeasible, smaller $\nu$ is preferred; and if both are feasible, ordinary Pareto dominance over $s$ is used:

$$X \succ Y \quad \Longleftrightarrow \quad s_j(X) \ge s_j(Y)\,\forall j, \qquad s_j(X) > s_j(Y) \text{ for some } j. \tag{37}$$

The bounded archive $\mathrm{Archive}_C$ removes duplicate endpoints, retains the feasible non-dominated front, and, when necessary, truncates it to capacity $C$ by descending crowding distance, following NSGA-II (Deb et al., 2002). If no feasible candidate exists, a separate recovery pool keeps the candidates with smallest violation.

Table 5: Selection of the window-refinement schedule. Values are means over $K = 1, 2, 3$ on 12 development complexes, with eight samples per condition. RMSD is in Å; Valid, binding-site recovery (BSR), and diversity are percentages. NFE denotes network evaluations per sample. Bold marks the configuration selected at each stage.

| Configuration | Valid↑ | RMSD↓ | BSR↑ | Diversity↑ | NFE |
|---|---|---|---|---|---|
| *(a) Sweeps and window width, re-noising to the prior* | | | | | |
| No window refinement | 43.00 | 3.503 | 78.42 | 38.88 | 207 |
| Width 5, one sweep | 46.45 | 3.602 | 78.07 | 36.62 | 176 |
| Width 7, one sweep | 41.31 | 3.773 | 78.03 | 35.13 | 147 |
| **Width 5, two sweeps** | 48.75 | 3.376 | 77.00 | 36.71 | 288 |
| Width 5, three sweeps | 42.15 | 3.422 | 75.89 | 36.71 | 400 |
| *(b) Window re-noising level, width 5 and two sweeps* | | | | | |
| To the prior | 48.75 | 3.376 | 77.00 | 36.71 | 288 |
| **To** $t = 0.5$ | 45.69 | 2.877 | 80.72 | 30.03 | 288 |
| To $t = 0.8$ | 33.44 | 3.426 | 80.18 | 32.44 | 288 |
| *(c) Joint and sequence updates per window, re-noising to* $t = 0.5$ | | | | | |
| 16 joint, 0 sequence | 45.69 | 2.877 | 80.72 | 30.03 | 288 |
| 12 joint, 4 sequence | 45.03 | 2.794 | 82.23 | 38.42 | 288 |
| 8 joint, 8 sequence | 46.11 | 2.780 | 82.32 | 37.93 | 288 |
| **4 joint, 12 sequence** | 50.52 | 2.765 | 82.40 | 37.78 | 288 |

All four searches use the operator set $\mathcal{O} = \{\mathrm{Sidechain}, \mathrm{Sequence}, \mathrm{Backbone}, \mathrm{Joint}\}$. For parent $X$ and sampling level $d$, one legal cyclic multi-region set $\mathcal{R} \leftarrow \rho(X)$ is shared by four siblings:

$$\mathrm{Prop}_\theta(X, d) = \{\mathcal{E}_\theta(P, X; \mathcal{R}, \mathcal{V}_o, d, Z_o) : o \in \mathcal{O}\}, \qquad \mathrm{Eval}(\mathcal{Y}) = \{(Y, S(Y)) : Y \in \mathcal{Y}\}. \quad (38)$$

Here $\rho$ samples one to three regions while retaining at least one anchor; $d \in \{0, 1, 2\}$ selects a schedule coefficient $c_d \in \{0.15, 0.45, 0.75\}$. The adapter sets

$$s_{\mathrm{init}}(d) = \max\{s_{\min}, \min\{1, 1 - c_d\}\}, \qquad s_{\min} = 0.01. \quad (39)$$

In the final window sampler, both initial stages draw fresh prior noise and integrate from $s_{\mathrm{init}}(d)$; this coefficient changes their time grids, not the amount of parent state mixed into their initialization. Window joint updates instead interpolate the current state with noise at $s_{\mathrm{win}} = 0.5$, independently of $d$, and window sequence updates restart from fresh priors at $s_{\min}$. Sparse-anchor completion uses $s_{\mathrm{init}} = s_{\min}$. These choices distinguish the sampling procedure from an exact prior-to-data flow over the full time interval.

SPIE adapts Pareto local search (Paquete et al., 2004) to neighborhoods generated by residue–variable edits. It maintains a feasible archive $\mathcal{A}_k$ and an infeasible recovery pool $\mathcal{Q}_k$. Let $\mathcal{Y}_k = \{X \mid (X, s, \nu) \in \mathcal{B}_k\}$ denote the proposed states in the evaluated batch of Eq.(12). For multiple objectives, the updates are

$$\begin{aligned} \mathcal{A}_{k+1} &= \mathrm{ND}_C(\{X \in \mathcal{A}_k \cup \mathcal{Y}_k : \nu(X) = 0\}), \\ \mathcal{Q}_{k+1} &= \mathrm{Best}^{\nu}_{C_Q}(\{X \in \mathcal{Q}_k \cup \mathcal{Y}_k : \nu(X) > 0\}). \end{aligned} \quad (40)$$

Here $\mathrm{ND}_C$ retains non-dominated candidates and truncates to capacity $C$ by crowding distance, while $\mathrm{Best}^{\nu}_{C_Q}$ retains the $C_Q$ candidates with the smallest violations. The pseudocode uses $\mathrm{Archive}_C$ for feasibility filtering followed by $\mathrm{ND}_C$, and $\mathrm{Recovery}_{C_Q}$ for infeasibility filtering followed by $\mathrm{Best}^{\nu}_{C_Q}$. In single-objective runs, archive selection retains the highest-scoring feasible candidates instead of the non-dominated front. BoN, beam search, and MCTS instead extract an archive from their evaluated history. Capacity truncation makes the per-expansion and history-based rules potentially different.

**Algorithm 4: Shared multi-property editing interface**

**Input:** Lead $X_0$; budget $B$; search policy $\Pi$
$\mathcal{S}_0 \leftarrow \Pi.\mathrm{Init}(\mathrm{Eval}(\{X_0\}))$; $n \leftarrow 0$; $k \leftarrow 0$.
**while** $n < B$ **do**
  $(X_k, d_k) \leftarrow \Pi.\mathrm{Parent}(\mathcal{S}_k)$.
  $\mathcal{Y}_k \leftarrow \mathrm{Take}_{B-n}[\mathrm{Eval}(\mathrm{Prop}_\theta(X_k, d_k))]$.
  $\mathcal{S}_{k+1} \leftarrow \Pi.\mathrm{Update}(\mathcal{S}_k, \mathcal{Y}_k)$; $n \leftarrow n + |\mathcal{Y}_k|$; $k \leftarrow k + 1$.
**return** the archive stored in $\mathcal{S}_k$.

### F.2 Four search policies

**Algorithm 5: Best-of-$N$ (BoN)**

**Input:** Initial lead $X_0$; budget $B$; archive capacity $C$
$\mathcal{H} \leftarrow \mathrm{Eval}(\{X_0\})$; $G \leftarrow \lceil B/4 \rceil$; $n \leftarrow 0$.
**for** $g = 0, \ldots, G-1$ **do**
  $\mathcal{Y}_g \leftarrow \mathrm{Take}_{B-n}[\mathrm{Eval}(\mathrm{Prop}_\theta(X_0, g \bmod 3))]$.
  $\mathcal{H} \leftarrow \mathcal{H} \cup \mathcal{Y}_g$; $n \leftarrow n + |\mathcal{Y}_g|$.
**return** $\mathrm{Archive}_C(\mathcal{H})$.

**Algorithm 6: Structured Pareto iterative editing (SPIE)**

**Input:** Initial lead $X_0$; budget $B$; archive capacity $C$
$C_Q \leftarrow \max(1, \lfloor C/2 \rfloor)$; $\mathcal{Y}_0 \leftarrow \mathrm{Eval}(\{X_0\})$; $\mathcal{A}_0 \leftarrow \mathrm{Archive}_C(\mathcal{Y}_0)$; $\mathcal{Q}_0 \leftarrow \mathrm{Recovery}_{C_Q}(\mathcal{Y}_0)$.
Initialize parent state $\eta_0$, focus $f_0 = X_0$, $n \leftarrow 0$, and $k \leftarrow 0$.
**while** $n < B$ **do**
  $\mathcal{D}_k \leftarrow \mathcal{A}_k$ if $\mathcal{A}_k \neq \varnothing$, otherwise $\mathcal{Q}_k$.
  $X_k \leftarrow f_k$ if $f_k \in \mathcal{D}_k$, otherwise $\mathrm{Select}(\mathcal{D}_k, \eta_k)$.
  $\mathcal{Y}_k \leftarrow \mathrm{Take}_{B-n}[\mathrm{Eval}(\mathrm{Prop}_\theta(X_k, d(\eta_k)))]$.
  $\mathcal{A}_{k+1} \leftarrow \mathrm{Archive}_C(\mathcal{A}_k \cup \mathcal{Y}_k)$; $\mathcal{Q}_{k+1} \leftarrow \mathrm{Recovery}_{C_Q}(\mathcal{Q}_k \cup \mathcal{Y}_k)$.
  $(\eta_{k+1}, f_{k+1}) \leftarrow \mathrm{Adapt}(\eta_k, X_k, \mathcal{Y}_k, \mathcal{A}_{k+1}, \mathcal{Q}_{k+1})$; $n \leftarrow n + |\mathcal{Y}_k|$; $k \leftarrow k + 1$.
**return** $\mathcal{A}_k$.

Parent selection uses

$$\mathrm{Select}(\mathcal{D}) = \begin{cases} \mathrm{Uniform}(\mathcal{D}), & \text{with probability } 0.1, \\ \arg\min_{X \in \mathcal{D}} \mathrm{rank}(X), & \text{otherwise}, \end{cases} \tag{41}$$

where

$$\mathrm{rank}(X) = \begin{cases} (-\mathrm{CD}(X), e(X), \mathrm{id}(X)), & X \in \mathcal{A}, \\ (\nu(X), e(X), \mathrm{id}(X)), & X \in \mathcal{Q}, \end{cases} \tag{42}$$

where $e(X)$ is its expansion count. A retained child is successful; for an infeasible child, retention must also reduce $\nu$. The depth update is

$$d^+ = \begin{cases} \max(0, d-1), & \text{success}, \\ \min(2, d+1), & \text{two consecutive failures}, \\ d, & \text{otherwise}. \end{cases} \tag{43}$$

**Algorithm 7: Pareto beam search**

**Input:** Initial lead $X_0$; budget $B$; beam width $W$; archive capacity $C$
$\mathcal{B}_0 \leftarrow \{X_0\}$ and $\mathcal{H} \leftarrow \{\mathrm{Evaluate}(X_0)\}$.
**for** layers $d = 0, 1, \ldots$ while budget remains **do**
  Generate and evaluate four operator siblings from each parent in $\mathcal{B}_d$.
  Append all evaluated siblings in this layer to $\mathcal{H}$.
  Fill $\mathcal{B}_{d+1}$ from successive Pareto fronts of this layer.
  Break an overfull front by descending crowding distance; use least violation if needed.
**return** $\mathrm{Archive}_C(\mathcal{H})$.

**Algorithm 8: Monte Carlo tree search (MCTS) with UCT selection (Kocsis & Szepesvári, 2006)**

**Input:** Initial lead $X_0$; budget $B$; exploration constant $c$; archive capacity $C$

Create the root from $\mathrm{Evaluate}(X_0)$ and set history $\mathcal{H}$ to the root.
**while** budget remains **do**
  Descend to a leaf by maximizing $\bar{q}_j + c\sqrt{\log N_{\mathrm{parent}}/N_j}$; unvisited children have infinite priority.
  Expand the leaf with four operator siblings and evaluate them.
  **if** one objective is active **then**
    Reward positive feasible improvement; before feasibility is reached, reward feasibility or reduced violation.
  **else**
    Reward each child by whether it survives in $\mathrm{Archive}_C(\mathcal{H} \cup \text{siblings})$.
  Backpropagate the maximum single-objective reward or mean multi-objective reward.
  Add the children to the tree and to $\mathcal{H}$.
**return** $\mathrm{Archive}_C(\mathcal{H})$.

The archive is an output set, not an additional supervision signal. Model parameters $\theta$ remain fixed. Property values guide parent selection and subsequent corruption-level choices between completed proposals.

# G Baselines and experimental protocols

## G.1 Baselines

**RFdiffusion.** RFdiffusion (Watson et al., 2023) generates protein backbone structures by diffusion. In peptide design, ProteinMPNN (Dauparas et al., 2022) assigns sequences to its sampled backbones.

**ProteinGenerator.** ProteinGenerator (Lisanza et al., 2025) jointly generates amino-acid sequences and backbone structures.

**Diffusion.** Diffusion is the peptide diffusion baseline described by Li et al. (2024), trained from scratch with an architecture similar to DiffAb (Luo et al., 2022).

**PepFlow.** PepFlow (Li et al., 2024) uses multimodal flow matching to jointly model residue types, backbone frames, and torsion angles. Its full-atom model provides a comparison for generation, selective editing, and property-guided optimization.

**PepGLAD.** PepGLAD (Kong et al., 2024) performs full-atom peptide design through geometric latent diffusion.

**PepHAR.** PepHAR (Li et al., 2025) builds peptides from hotspot residues by autoregressively extending multiple fragments. It provides the sparse-anchor completion benchmark used here.

**Joint redesign baselines.** Independent priors sample residue types from training-set amino-acid frequencies and backbone geometry from a Ramachandran prior. Sequential Rosetta redesign first samples a backbone from the same geometric prior and then assigns a sequence with Rosetta fixed-backbone design (Kuhlman & Baker, 2000). These comparisons test joint sequence–structure reconstruction against independent sampling and sequential design.

**Specialized editing baselines.** ProteinMPNN (Dauparas et al., 2022), LigandMPNN (Dauparas et al., 2025), and Rosetta FixBB (Kuhlman & Baker, 2000) provide fixed-backbone sequence design comparisons. Rosetta NGK (Stein & Kortemme, 2013) and CCD (Canutescu & Dunbrack, 2003) reconstruct backbone segments through kinematic closure and cyclic coordinate descent, respectively. Side-chain comparisons include PepBFN (Qian et al., 2026a), FlowPacker (Lee & Kim, 2025), and RosettaPacker (Leaver-Fay et al., 2011).

**Search strategies.** Best-of-N (BoN) (Stiennon et al., 2020), Beam Search (Freitag & Al-Onaizan, 2017), and Monte Carlo Tree Search (MCTS) (Coulom, 2007) compare independent candidate selection, bounded iterative search, and tree-based exploration through the same editing interface. Structured Pareto Iterative Editing (SPIE) is the strategy developed here from Pareto local search (Paquete et al., 2004), using residue–variable edits to explore candidate neighborhoods. Appendix F gives the update rules and pseudocode for all four implementations.

### G.2 Evaluation set and shared settings

Following the PepHAR protocol (Li et al., 2025), the benchmark combines protein–peptide complexes from PepBDB (Wen et al., 2019) and Q-BioLiP (Wei et al., 2024), with duplicate and low-quality entries removed. The original PepHAR benchmark comprises 8,207 non-homologous examples for training and validation and 158 test complexes. The binding pocket comprises receptor residues with at least one heavy atom within 10 Å of a peptide heavy atom.

Each test record specifies a protein–peptide structure and serves as one evaluation unit. Results are averaged over generated samples within each unit, then over units. Multiple units can share a receptor. Receptor atoms are excluded from peptide reconstruction metrics. Each receptor and peptide is translated by the native peptide C$\alpha$ centroid, and translation priors are defined in this frame. Table 6 lists the sampling configuration of each evaluation.

Table 6: Evaluation configurations. All EditFlow evaluations use the same model trained for 300,000 updates.

| Evaluation | Samples or budget |
|---|---|
| Full generation | 64 per complex |
| Interface side chains | 64 per complex |
| Sequence-only segments | 64 per complex |
| Backbone / joint segments | 64 per complex |
| Sparse anchors | 8 per complex at each of $K = 1, 2, 3$ |
| Property optimization | $B = 64$ proposals per complex |

### G.3 Pocket-conditioned generation

Pocket-conditioned generation assesses the complete-peptide design capability of a model trained for local editing. All peptide variables are released, and EditFlow generates 64 candidates per complex using 200-step joint flow integration in the pocket frame defined above. Table 1 summarizes reconstruction, computational energy assessment, and structural diversity.

For Table 1, RFdiffusion, ProteinGenerator, Diffusion, and PepFlow results come from Table 1 of Li et al. (2024); the PepFlow row uses its joint backbone, sequence, and angle model. PepGLAD and PepHAR results come from the peptide-design table and AAR appendix of Li et al. (2025), with three inferred hotspots for PepHAR. The diversity column reports structural dissimilarity under the PepFlow protocol, displayed as percentages. PepGLAD and PepHAR diversity values are omitted because their evaluation protocols differ. AAR, SSR, and BSR denote amino-acid, secondary-structure, and binding-site recovery. Dashes indicate unreported values.

### G.4 Joint redesign

Joint redesign tests whether sequence and structure can adapt together within a retained peptide context. All methods regenerate the same editable region while holding the receptor and unedited peptide state fixed. EditFlow releases residue types, backbone frames, and torsions in that region and generates 64 candidates per complex.

Each method receives local Cartesian minimization with the Rosetta REF15 Cartesian score function (Alford et al., 2017; Leaver-Fay et al., 2011), using at most 2,000 steps. Minimization moves editable atoms while holding the receptor and outside peptide atoms fixed. Reconstruction and interface metrics are computed on these refined candidates.

The editable region was a contiguous window centered on the peptide sequence. Its length was 70% of the peptide length rounded to the nearest residue, bounded to 3–12 residues and capped at

the peptide length. The receptor and native residue states outside the window formed the retained context. This rule yielded whole-peptide edits for nine three-residue peptides.

### G.5 Peptide completion from sparse anchors

Sparse-anchor completion tests how much sequence and structure can be reconstructed from a few retained residues. Completion follows the evaluation protocol of Li et al. (2025), using the same evaluation targets and predefined hotspot lists for $K = 1, 2, 3$. The complete states of the selected hotspot residues and the receptor are retained while the remaining peptide types and structures are generated. When the retained residues already cover a short peptide, nothing is generated, so that complex contributes an unchanged copy to the averages. EditFlow generates eight candidates per complex for each anchor count using window refinement (Appendix E). Published baselines use 64 samples per target. The main table reports $K = 2$; Appendix H.5 gives all three settings.

### G.6 Single-objective optimization

Property searches test whether repeated local edits improve externally evaluated properties while preserving each proposal's specified context. All searches start from a bound reference peptide, evaluate 64 new candidates per complex, and retain at most 20 feasible candidates. We maximize negative Rosetta interface energy, predicted non-toxicity, cell penetration, Caco-2 permeability, and half-life. PeptiVerse (Zhang et al., 2026) predicts the sequence-dependent properties. Half-life is reported in hours, and Caco-2 values retain the predictor's native scale. Appendix F specifies the search policies.

Each expansion proposes one candidate for each of the four editing operations on a shared random region. Regions contain one to three cyclic segments and leave at least one peptide residue fixed. The region policy is the same across search methods. The retained residues are determined anew for each proposal by the complement of its effective mask relative to its parent; the region policy uses no biological hotspot annotations.

For a single objective, the reported endpoint is the highest score in the final feasible archive. Improvements are measured relative to the starting score of each run. An empty archive falls back to feasible evaluated candidates and then to the starting complex. Feasibility tests atom completeness and the configured interface-clash threshold. These checks filter candidates for archive selection. Negative changes are retained when an infeasible starting structure is replaced by a lower-scoring feasible candidate.

**Sampling.** The main single-objective experiments and the window-refinement row of Table 17 use the sequence–structure completion and overlapping-window procedure in Appendix E. All five single objectives are evaluated for the four search policies over the full test set. Each run evaluates its active objective and the shared feasibility checks.

### G.7 Multi-objective optimization

Multi-objective search tests whether repeated editing can produce candidate sets that improve several properties together. Each candidate is evaluated on all five objectives, using the starting complexes, proposal budget, editing regions, and feasibility checks above. The final archive retains up to 20 feasible Pareto-nondominated candidates, with crowding-distance truncation when capacity is exceeded. Hypervolume measures the property-space coverage of this archive.

The matched model comparison in Table 17 uses the original 200-step joint flow integration for both PepFlow and EditFlow. Both models supply proposals through the same interface, and the two arms share a mask policy, a region policy, a scorer, and a proposal budget to compare their proposal distributions under a common search procedure. Reference scores agree across the two models and four searches. Proposal budgets are equal across methods, while network evaluations and wall-clock cost differ. Table 16 adds complete single-objective results for this integration protocol.

**Additional optimization cases.** The comparisons in Figure 5A,B and its molecular case use residue-wise extension, in which predicted residue identities and torsions rebuild the editable region under standard peptide geometry while unselected variables are retained. This simpler completion rule preceded window refinement, and results obtained under the two are reported separately throughout. The case in that figure additionally scores solubility and other properties outside the five-objective benchmark.

### G.8 Evaluation metrics

**Amino-acid recovery (AAR).** Sequence recovery is evaluated at editable positions. Interface and non-interface recovery are computed over the corresponding edited positions, averaging over complexes containing such positions.

**BLOSUM62 similarity.** BLOSUM62 similarity is the percentage of editable positions with a positive substitution score relative to the reference (Henikoff & Henikoff, 1992).

**C$\alpha$ RMSD.** Full-generation and sparse-anchor C$\alpha$ RMSD use peptide alignment to the reference.

**Backbone RMSD (BB-RMSD).** Segment BB-RMSD uses observable N/C$\alpha$/C/O atoms in the fixed complex frame.

**Segment closure.** Segment closure requires affected C–N bond lengths to be within 0.20 Å of 1.33 Å.

**Interface contact F1.** Interface contact F1 measures agreement between the generated and reference contacts involving edited peptide residues. A peptide–receptor residue pair counts as a contact when its closest heavy atoms are within 5 Å. Precision is the fraction of generated contacts also present in the reference; recall is the fraction of reference contacts recovered by the candidate. Their harmonic mean gives F1, reported as a percentage. Results are averaged over samples and complexes with reference contacts at edited positions.

**Backbone success.** Backbone success is the fraction of applicable complexes for which at least one of the 64 samples passes the backbone evaluation checks and falls below the stated BB-RMSD threshold.

**Valid.** For sparse-anchor completion, Valid is the fraction of adjacent C$\alpha$ distances at or below 3.8 Å, following Li et al. (2025). The criterion bounds this distance from above only.

**Binding-site recovery (BSR).** BSR quantifies recovery of the receptor surface contacted by the native peptide. The binding site is identified separately for the generated and reference complexes, and the number of shared receptor residues is divided by the size of the reference binding site. For the EditFlow full-generation and sparse-anchor evaluations, a receptor residue is included when its C$\alpha$ atom lies within 10 Å of any peptide C$\alpha$ atom, following the released benchmark implementations. The resulting percentage is averaged within each complex and then over complexes.

**Secondary-structure recovery (SSR).** SSR compares DSSP assignments (Kabsch & Sander, 1983) of the generated and reference peptides, computed with MDTraj (McGibbon et al., 2015).

**Stability.** Stability measures how often a generated complex has a lower Rosetta total energy than its native reference. The reported percentage is the fraction of scored candidates passing this criterion, averaged over test complexes. Generated and reference complexes undergo the same Rosetta FastRelax and full-atom scoring procedure (Leaver-Fay et al., 2011). Full-generation evaluation uses one relaxation round; sparse-anchor evaluation follows PepHAR's two successive rounds and averages the two energy values (Li et al., 2025).

**Affinity.** Affinity measures how often a generated peptide has a more favorable computed binding energy than the native peptide. Rosetta InterfaceAnalyzer evaluates the peptide–receptor interface

using `dG_separated`, with repacking of the separated partners. A candidate passes when this binding-energy score is lower than that of its native reference. The reported percentage averages the per-complex fractions of scored candidates satisfying this condition. Energy evaluation uses the same relaxed structures and round-averaging procedure as the stability metric.

**Novelty.** Following PepHAR (Li et al., 2025), a generated peptide is considered novel when its TM-score and sequence identity to the native peptide are both at most 0.5. Novelty is the percentage of candidates satisfying both conditions, averaged over test complexes. Both comparisons use the whole peptide, including retained anchors; sequence identity follows PepHAR's `SequenceMatcher` implementation.

**Structural diversity.** Structural diversity measures variation among the structures generated for one complex. TM-align (Zhang & Skolnick, 2005) computes a TM-score for every distinct pair of generated peptides. The mean pairwise TM-score is subtracted from one and multiplied by 100. Table 1 reports the mean of these per-complex percentages.

**Sparse-anchor diversity.** Following PepHAR (Li et al., 2025), diversity combines sequence and structural differences among generated peptides. For each candidate pair, one minus the TM-score is multiplied by one minus the sequence similarity, using the whole peptide including retained anchors. The products are averaged over pairs, converted to percentages, and then averaged over test complexes for each anchor count.

**Property normalization.** The five objectives have different units and numerical ranges. For each objective, the minimum and maximum reference scores across ten calibration complexes define a fixed range. A candidate's score is normalized by subtracting the minimum and dividing by the difference between the maximum and minimum. All compared methods and search policies use the same calibration values. Normalized scores are not clipped, so values below zero or above one are possible. The calibration complexes are PDB entries 1R1S-B, 8DGO-C, 1JYR-L, 3MXY-L, 2H5K-C, 1GBQ-B, 1ZFP-I, 1BM2-L, 5CDW-F, and 1TZE-I.

**Hypervolume (HV).** HV measures the region of objective space covered by the feasible candidate archive relative to a fixed reference point. All five objectives are oriented so that higher scores are better and normalized as described above. Each candidate defines a five-dimensional box extending from a reference value of $-0.1$ in every objective to its normalized score vector. Candidates below the reference value in any objective contribute no volume. HV is the volume of the union of these boxes, with overlaps counted once, computed by exact recursive slicing. An empty archive has HV zero. Under the same normalization and reference point, larger HV indicates greater coverage of favorable objective combinations.

**Optimization trajectories.** Figure 4 plots performance against the number of evaluated proposals. Single-objective curves track the best feasible score so far, using the starting score until a feasible candidate is available. Multi-objective curves track the current archive HV under the update rule in Appendix G.7. Values are averaged equally over test complexes.

**Statistical uncertainty.** Optimization results report the mean across test complexes with 95% confidence intervals of 1.96 standard errors on either side. The standard error is the sample standard deviation across complexes divided by the square root of their number. Segment and side-chain results use 10,000 bootstrap resamples over complexes; Table 15 uses 10,000 paired resamples of within-complex differences. Intervals reflect one generation run, with no receptor-family grouping or multiple-comparison adjustment.

Four complexes in the test set begin from an infeasible structure. Their first feasible replacement can have a large negative interface score, shifting the mean by several REU during the first few evaluations. Figure 4e therefore presents final interface scores.

# H ADDITIONAL RECONSTRUCTION ANALYSES

## H.1 FIXED-BACKBONE SEQUENCE DESIGN

**Editable segments.** For a peptide of length $L$, the segment length is

$$\ell = \min(L, 12, \max(3, \mathrm{round}_{\text{half-up}}(0.70L))) .$$

The segment is centered, and an odd number of remaining residues leaves one additional residue on the C-terminal side. The sequence-only and backbone-only evaluations use these length-defined segments. Nine short peptides yield whole-peptide segments. Rosetta CCD and NGK require a flanking anchor on each side and are evaluated on the evaluation subset with anchors on both sides.

Sequence-only design retains the backbone and changes identities and associated side chains. Backbone-only design retains identity and changes the local frame and torsions. Local Cartesian minimization follows Appendix G.4.

**Sampling protocol.** Inference settings were selected on a development set of 128 complexes disjoint from the benchmark, sweeping joint flow integration, direct endpoint updates, and fixed-backbone sampling with window refinement, together with step counts, endpoint sequence temperatures, and static conditioning choices. The same sweep was run for EditFlow and for PepFlow, and the selected settings were frozen before any benchmark evaluation.

Both models use 20 integration steps and terminal sequence temperature 0.25. The static feature encoders hide peptide identity and geometry, while fixed-backbone geometry and retained residue identities remain available through the current state. This evaluation-time conditioning choice leaves the learned parameters unchanged. All fixed variables and backbone atoms are preserved throughout sampling. Sequence diversity is the mean pairwise Hamming difference over edited positions, averaged within each complex and then across complexes.

**Results.** Table 7 compares sequence redesign with specialized inverse-folding and energy-based methods. EditFlow reaches 46.13% amino-acid recovery, compared with 45.33% for Rosetta FixBB, 52.69% for ProteinMPNN (Dauparas et al., 2022), and 54.01% for LigandMPNN (Dauparas et al., 2025).

Table 7: Fixed-backbone sequence design. Entries are means with 95% bootstrap confidence intervals over complexes. Recovery and BLOSUM62 similarity are percentages. Interface and non-interface recovery are computed over the corresponding edited positions, averaging over complexes containing such positions. Bold marks the best mean.

| Method | AAR↑ | Interface AAR↑ | Non-interface AAR↑ | BLOSUM62↑ |
|---|---|---|---|---|
| Training-set AA prior | 5.89 [5.66, 6.13] | 5.96 [5.67, 6.25] | 6.19 [5.56, 6.83] | 16.79 [16.24, 17.35] |
| Rosetta FixBB | 45.33 [41.81, 48.88] | 46.20 [42.43, 49.89] | 36.99 [27.93, 46.64] | 57.66 [53.90, 61.30] |
| ProteinMPNN | 52.69 [49.31, 55.98] | 52.58 [48.81, 56.22] | 45.13 [35.98, 54.05] | 65.14 [61.70, 68.42] |
| LigandMPNN | **54.01** [50.55, 57.49] | **54.71** [50.87, 58.45] | **45.17** [36.31, 54.16] | **67.83** [64.15, 71.35] |
| EditFlow | 46.13 [40.99, 51.23] | 46.36 [41.02, 51.81] | 42.58 [34.31, 50.98] | 53.31 [48.67, 57.89] |

Rosetta FixBB follows Kuhlman & Baker (2000); BLOSUM62 follows Henikoff & Henikoff (1992). The fixed-backbone inference configuration is given in Appendix H.1.

We also compare PepFlow with EditFlow under the same fixed-backbone sampling conditions (Appendix H.1). Mean amino-acid recovery is 46.17% for PepFlow and 46.13% for EditFlow. Table 8 reports the paired changes. Sequence diversity is the mean pairwise Hamming difference over edited positions. Interface and non-interface recovery are computed over the corresponding edited positions, averaging over complexes containing such positions.

Table 8: Paired comparison of PepFlow and EditFlow in matched sequence design. Values are mean differences in percentage points [95% bootstrap interval over complexes].

| Metric | EditFlow minus PepFlow |
|---|---|
| Amino-acid recovery | $-0.04\,[-1.27,\ 1.18]$ |
| Sequence diversity | $+2.74\,[1.14,\ 4.36]$ |

### H.2 Fixed-sequence backbone reconstruction

Backbone reconstruction uses the length-defined segments above, retains the sequence, and releases local backbone frames and torsions. Each complex has 64 samples. Local Cartesian minimization follows Appendix G.4; Rosetta NGK uses its own refinement procedure.

Table 9 evaluates backbone remodeling with the peptide sequence retained. EditFlow improves over the Ramachandran prior on all four reported measures. Rosetta NGK attains the lowest RMSD on the subset of segments it can treat, while EditFlow covers every complex and reaches 98.64% closure after local minimization.

Table 9: Fixed-sequence backbone reconstruction. Entries are means with 95% bootstrap confidence intervals over complexes. Success and closure are percentages. Rosetta NGK and CCD use the evaluation subset with a flanking anchor on each side; the other methods use the full evaluation set. Bold marks the best mean within each cohort.

| Method | BB-RMSD↓ (Å) | Success↑ $< 1$ Å | Success↑ $< 2$ Å | Closure↑ |
|---|---|---|---|---|
| Rosetta NGK | **0.97** [0.87, 1.07] | **94.87** [91.03, 98.08] | **96.15** [92.95, 98.72] | **96.02** [92.82, 98.79] |
| Rosetta CCD | 4.33 [4.03, 4.65] | 39.74 [32.05, 47.44] | 57.05 [49.36, 64.74] | 93.56 [91.93, 95.02] |
| Ramachandran prior | 7.49 [6.87, 8.17] | 0.53 [0.00, 1.60] | 20.86 [14.97, 26.74] | 77.17 [73.39, 80.89] |
| EditFlow | **3.51** [3.33, 3.68] | **28.34** [21.93, 34.76] | **62.57** [55.61, 69.52] | **98.64** [98.29, 98.96] |

Rosetta NGK is next-generation kinematic closure (Stein & Kortemme, 2013) and CCD is cyclic coordinate descent (Canutescu & Dunbrack, 2003). All methods except NGK receive local Cartesian minimization. Success is the fraction of complexes with at least one passing sample below the stated RMSD threshold.

Rosetta NGK and CCD use the evaluation subset with a flanking anchor on each side; the other backbone methods use the full evaluation set. NGK allows a small displacement of outside peptide atoms, whereas the other procedures hold them fixed.

### H.3 Joint sequence–structure reconstruction examples

Figure 6 illustrates how the joint reconstruction mask varies with peptide length while retaining the unedited context.

### H.4 Interface side-chain repacking

**Protocol.** Interface residues have at least one valid heavy atom within 5 Å of a receptor heavy atom. We select the longest contiguous peptide segment among these residues, resolving ties toward the N-terminus. Only its valid $\chi_1$–$\chi_4$ channels are regenerated, first by the 50-step flow and then by one width-seven window sweep over the segment. Sequence, backbone, outside side chains, and receptor coordinates remain fixed. Angular errors use symmetry-corrected periodic differences, and side-chain RMSD accounts for equivalent-atom permutations before averaging over edited residues.

**Results.** Table 10 compares the generative models with dedicated packing methods. All methods reconstruct the longest contiguous interface segment with the supplied sequence and backbone fixed. FlowPacker (Lee & Kim, 2025) and RosettaPacker reach lower angular and coordinate errors in this

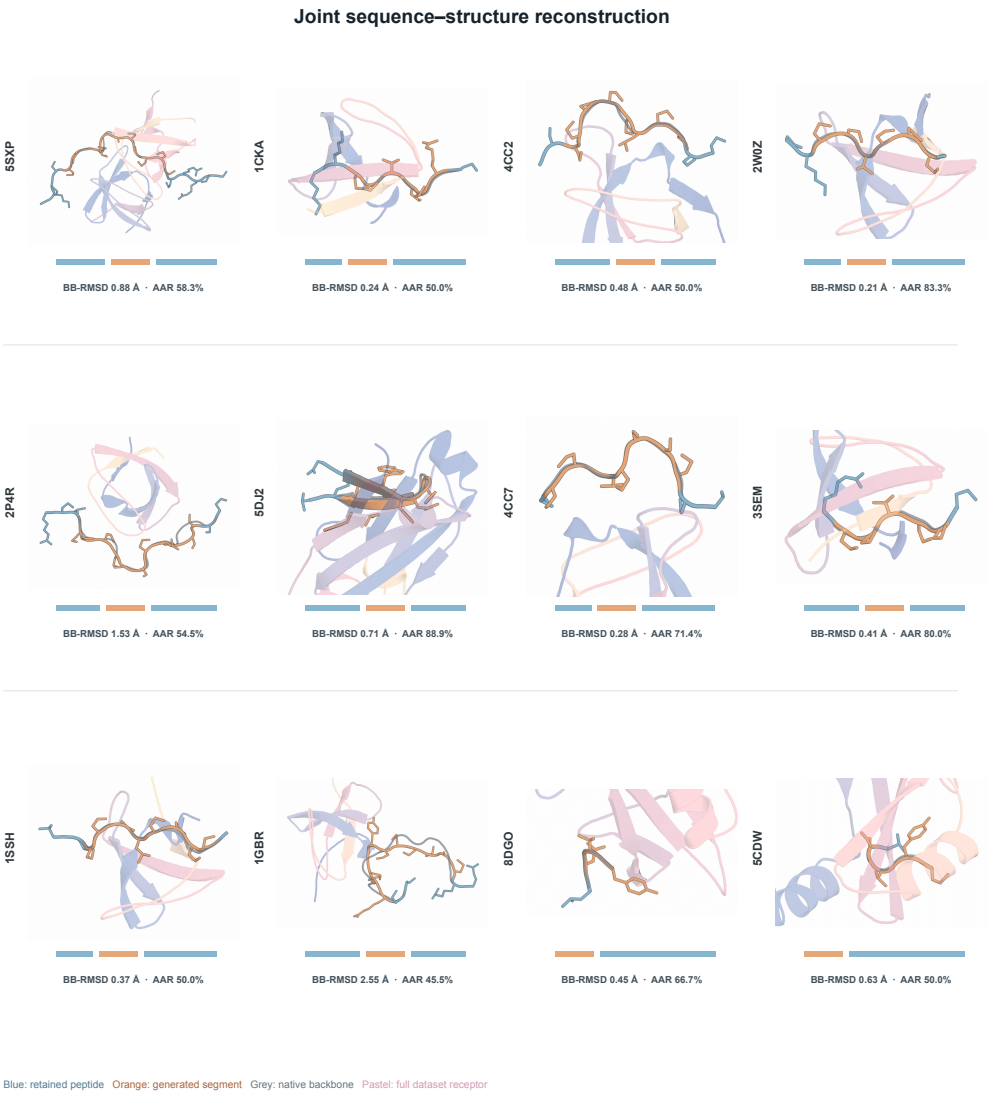



Figure 6: **Joint sequence–structure reconstruction in selected complexes.** Twelve distinct PDB complexes are ordered left to right and top to bottom by the actual generated fraction, from 57.1% to 80.0%; three examples share 66.7%. Orange shows the generated segment, blue the retained peptide, gray the native backbone, and pastel colors the complete dataset receptor. Each label reports edited-segment backbone-atom RMSD and amino-acid recovery computed from the displayed, locally minimized PDB. Examples were selected from up to 64 archived refined candidates per target, prioritizing low backbone RMSD among valid structures while choosing AAR near 50% for three cases. These quality-selected structures illustrate edit scopes and are not estimates of single-draw performance or the aggregate values in Table 2.

specialized setting, and FlowPacker has the lowest error on every individual torsion. The interface clash score counts cross-interface heavy-atom overlaps of at least 0.4 Å per 1,000 complex heavy atoms. Table 11 resolves the angular error by individual torsion.

**Local torsion refinement.** EditFlow's side-chain results in Table 10 apply a $\chi$-specific window sweep after the initial 50-step sampler. Width-seven overlapping windows sweep once from the N-terminus, with eight $\chi$ updates from $t = 0.5$ per window, and both arms preserve sequence, backbone, receptor, and outside-region side chains. Table 12 isolates this component by pairing each

Table 10: Interface side-chain repacking with generative and dedicated packing methods. Entries are means with 95% bootstrap confidence intervals over the test complexes and 64 samples per complex. Accuracy and contact F1 are percentages. Bold marks the best mean.

| Method | $\chi$-MAE (°)↓ | Acc@20° ↑ | SC-RMSD (Å)↓ | Contact F1↑ | Clash↓ |
|---|---|---|---|---|---|
| PepBFN | 86.32 [83.73, 88.94] | 13.94 [12.16, 15.83] | 2.409 [2.292, 2.532] | 74.87 [72.82, 76.80] | 37.13 [32.69, 41.73] |
| PepFlow | 68.37 [65.79, 70.88] | 28.81 [27.17, 30.46] | 2.002 [1.849, 2.161] | 82.17 [80.21, 84.00] | 18.71 [16.72, 20.70] |
| EditFlow | 56.72 [54.63, 59.02] | 27.30 [25.07, 29.42] | 1.806 [1.671, 1.951] | 81.77 [79.92, 83.47] | 23.42 [20.47, 26.60] |
| RosettaPacker | 33.25 [30.32, 36.37] | 66.86 [63.83, 69.91] | 0.965 [0.837, 1.112] | 91.42 [89.49, 93.18] | 2.92 [1.88, 4.26] |
| FlowPacker | **24.46** [22.01, 27.03] | **74.45** [71.49, 77.31] | **0.745** [0.630, 0.872] | **93.83** [92.10, 95.37] | **2.42** [1.97, 2.90] |

PepBFN and PepFlow use their published weights with updates restricted to side-chain torsions. EditFlow adds the $\chi$ window sweep described below. All methods retain the supplied sequence and backbone.

Table 11: Symmetry-corrected mean errors for individual side-chain torsions (degrees). Bold marks the lowest mean in each column.

| Method | $\chi_1$ ↓ | $\chi_2$ ↓ | $\chi_3$ ↓ | $\chi_4$ ↓ |
|---|---|---|---|---|
| PepBFN | 77.92 | 88.77 | 94.09 | 69.54 |
| PepFlow | 67.52 | 66.46 | 69.09 | 81.30 |
| EditFlow | 63.13 | 46.52 | 37.30 | 61.72 |
| RosettaPacker | 29.05 | 32.83 | 41.30 | 63.82 |
| FlowPacker | **19.57** | **27.10** | **32.71** | **45.01** |

sample with a baseline sample drawn under identical conditions, so the two samplers are compared on the same complexes.

Table 12: Paired side-chain sampling analysis. Each arm uses the test set with 64 samples per complex. Values use the primary angular and symmetry-aware coordinate metrics, and NFE is the mean number of network evaluations per sample. Bold marks the best mean for each directional metric.

| Sampler | $\chi$-MAE↓ | Acc@20° ↑ | SC-RMSD↓ | Contact F1↑ | Clash↓ | NFE |
|---|---|---|---|---|---|---|
| 50-step flow | 57.545 | 26.992 | 1.8429 | 81.221 | 23.572 | 50.00 |
| + $\chi$ window sweep | **56.725** | **27.299** | **1.8059** | **81.769** | **23.416** | 80.03 |

The paired mean difference in $\chi$-MAE is $-0.820°$ (95% interval $[-1.271, -0.383]$), and side-chain RMSD decreases by 0.037 Å (interval $[-0.052, -0.024]$ for window minus baseline). Accuracy and clash-score differences have intervals crossing zero. All generated samples in both arms leave the retained context unchanged and preserve the supplied sequence. Primary $\chi$-MAE pools valid angles within each complex before averaging over complexes, and coordinate RMSD is symmetry-aware and averaged over edited residues.

### H.5 Sparse-anchor completion metrics

Table 13 compares EditFlow with the published sparse-anchor completion baselines of Li et al. (2025), using the same evaluation targets and predefined hotspot lists under the PepHAR evaluation protocol. EditFlow's whole-peptide RMSD and binding-site recovery improve as more residues are retained, while its validity, novelty, and diversity decrease.

Published values are transcribed from the full sparse-anchor completion results in the PepHAR appendix.

Table 13: Peptide completion from sparse anchors for published baselines (Li et al., 2025) and EditFlow with window refinement. $K$ is the number of retained residues. Valid is the fraction of adjacent C$\alpha$ distances at or below 3.8 Å; RMSD is aligned whole-peptide C$\alpha$ RMSD in Å. Other numeric entries are percentages.

| Method | $K$ | Valid↑ | RMSD↓ | SSR↑ | BSR↑ | Affinity↑ | Novelty↑ | Diversity↑ |
|---|---|---|---|---|---|---|---|---|
| RFdiffusion | 1 | 66.80 | 3.51 | 63.19 | 23.56 | 20.58 | 66.17 | 22.46 |
| | 2 | 68.68 | 2.85 | 65.68 | 31.14 | 22.34 | 45.53 | 24.14 |
| | 3 | 69.88 | 4.09 | 63.66 | 26.83 | 21.26 | 55.03 | 26.67 |
| ProteinGenerator | 1 | 68.00 | 3.79 | 64.53 | 25.52 | 20.55 | 60.39 | 21.28 |
| | 2 | 69.20 | 3.92 | 66.79 | 30.61 | 21.54 | 55.24 | 23.29 |
| | 3 | 68.52 | 3.95 | 65.86 | 24.17 | 22.80 | 50.73 | 20.82 |
| PepFlow | 1 | 40.35 | 2.51 | 79.58 | 86.40 | 18.13 | 50.46 | 16.29 |
| | 2 | 49.29 | 2.82 | 79.23 | 85.05 | 18.20 | 54.74 | 19.52 |
| | 3 | 42.68 | 2.45 | 81.00 | 82.76 | 13.64 | 50.93 | 16.97 |
| PepGLAD | 1 | 53.45 | 3.87 | 76.59 | 20.15 | 12.03 | 50.46 | 30.84 |
| | 2 | 52.96 | 3.93 | 75.06 | 20.02 | 15.72 | 54.74 | 30.36 |
| | 3 | 53.51 | 3.84 | 76.26 | 19.61 | 18.27 | 50.93 | 30.99 |
| PepHAR | 1 | 56.01 | 3.72 | 80.61 | 78.18 | 19.94 | 80.61 | 29.79 |
| | 2 | 55.36 | 2.85 | 82.79 | 85.80 | 19.17 | 74.76 | 25.32 |
| | 3 | 55.41 | 2.15 | 83.02 | 88.02 | 20.65 | 72.56 | 19.68 |
| EditFlow | 1 | 56.26 | 2.80 | 79.97 | 77.59 | 9.69 | 83.82 | 39.04 |
| | 2 | 49.71 | 2.18 | 79.75 | 86.80 | 13.84 | 69.32 | 33.10 |
| | 3 | 46.17 | 2.03 | 79.90 | 89.82 | 16.71 | 56.22 | 27.57 |

Figure 7 pairs four illustrative completion examples with the unselected cohort trends for the same anchor conditions.

# I Additional property optimization results

## I.1 Single-objective results

Table 14 gives absolute endpoint scores under window refinement, corresponding to the trajectory endpoints in Figure 4a–d and the final interface scores in panel (e). Each objective is optimized independently. Table 15 reports SPIE's additional improvement relative to the other policies. The clearest feedback gains concern predicted penetration, Caco-2, and half-life under this protocol.

Table 15 compares improvements measured relative to each run's starting score. In the window-refinement non-toxicity experiments, BoN and SPIE share reference scores, as do beam search and MCTS.

Table 14: Final single-objective scores with window refinement. Entries are means with 95% confidence intervals. Each objective uses a separate search with 64 proposals per complex.

| Objective | BoN | SPIE | Beam | MCTS |
|---|---|---|---|---|
| $-\Delta G$ (REU) | 27.861 [25.361, 30.362] | 28.051 [25.588, 30.513] | 27.445 [24.962, 29.928] | 26.922 [24.394, 29.451] |
| Non-toxicity | 0.940 [0.922, 0.957] | 0.978 [0.967, 0.989] | 0.974 [0.963, 0.985] | 0.956 [0.941, 0.971] |
| Penetration | 0.828 [0.806, 0.851] | 0.927 [0.914, 0.939] | 0.901 [0.887, 0.915] | 0.861 [0.840, 0.882] |
| Caco-2 | -5.83653 [-5.83808, -5.83499] | -5.83178 [-5.83331, -5.83025] | -5.83281 [-5.83440, -5.83122] | -5.83501 [-5.83656, -5.83347] |
| Half-life (h) | 1.795 [1.670, 1.920] | 2.355 [2.211, 2.499] | 2.145 [2.028, 2.262] | 1.936 [1.796, 2.076] |

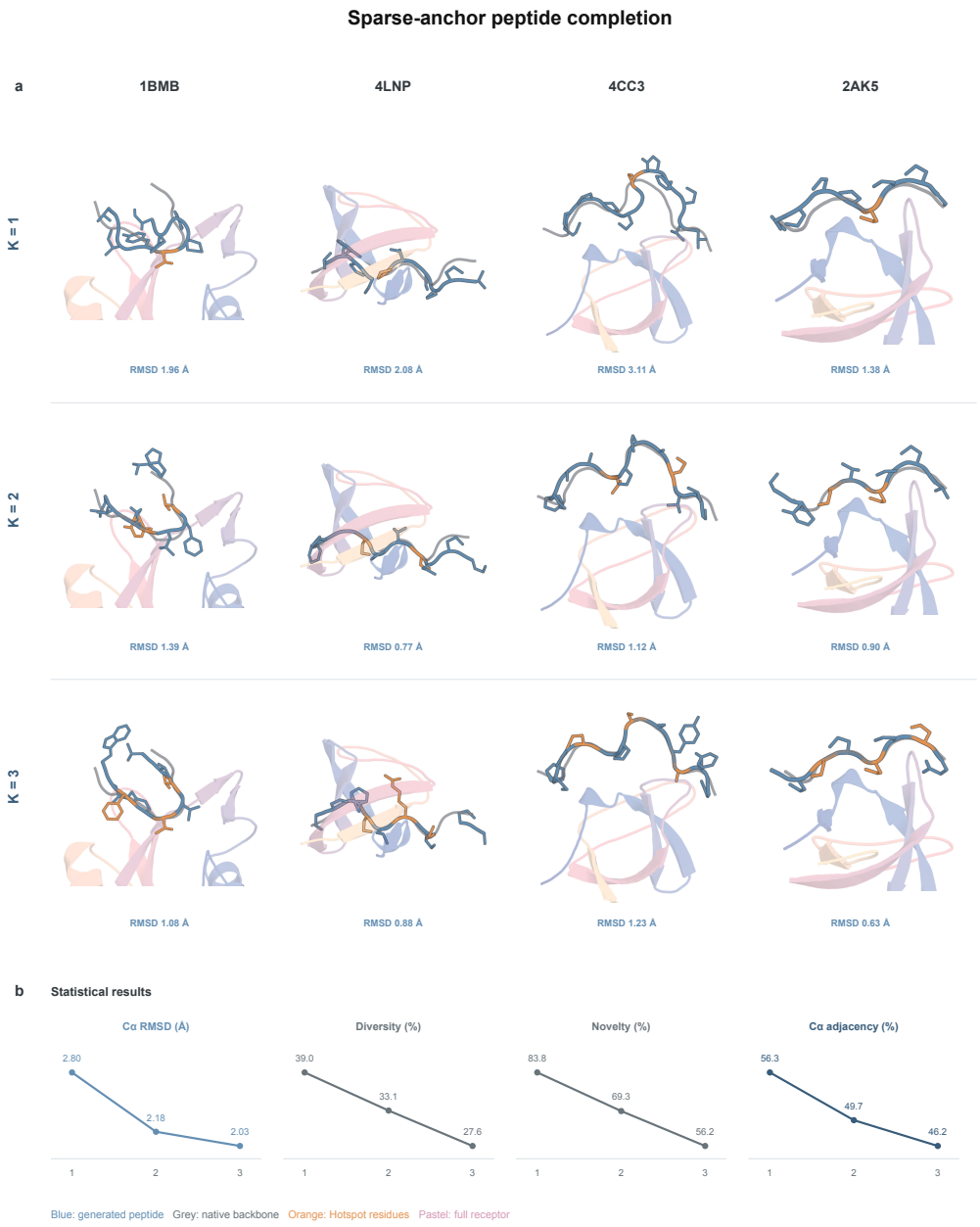



Figure 7: **Sparse-anchor completion in selected complexes and the full cohort.** (a) PDB complexes 1BMB-I, 4LNP-B, 4CC3-B, and 2AK5-D (left to right), each shown under the independent $K = 1, 2, 3$ conditions using the same sample index within a column. Orange marks retained anchor residues; blue and gray show the generated and native peptides, and pastel colors show the complete dataset receptor. Structure labels give whole-peptide aligned C$\alpha$ RMSD, which includes the retained anchors. These selected examples come from eight draws per target and do not estimate single-draw performance. (b) Means over the unselected test cohort and eight samples per $K$ for RMSD, diversity, novelty, and the fraction of adjacent C$\alpha$ distances at or below 3.8 Å.

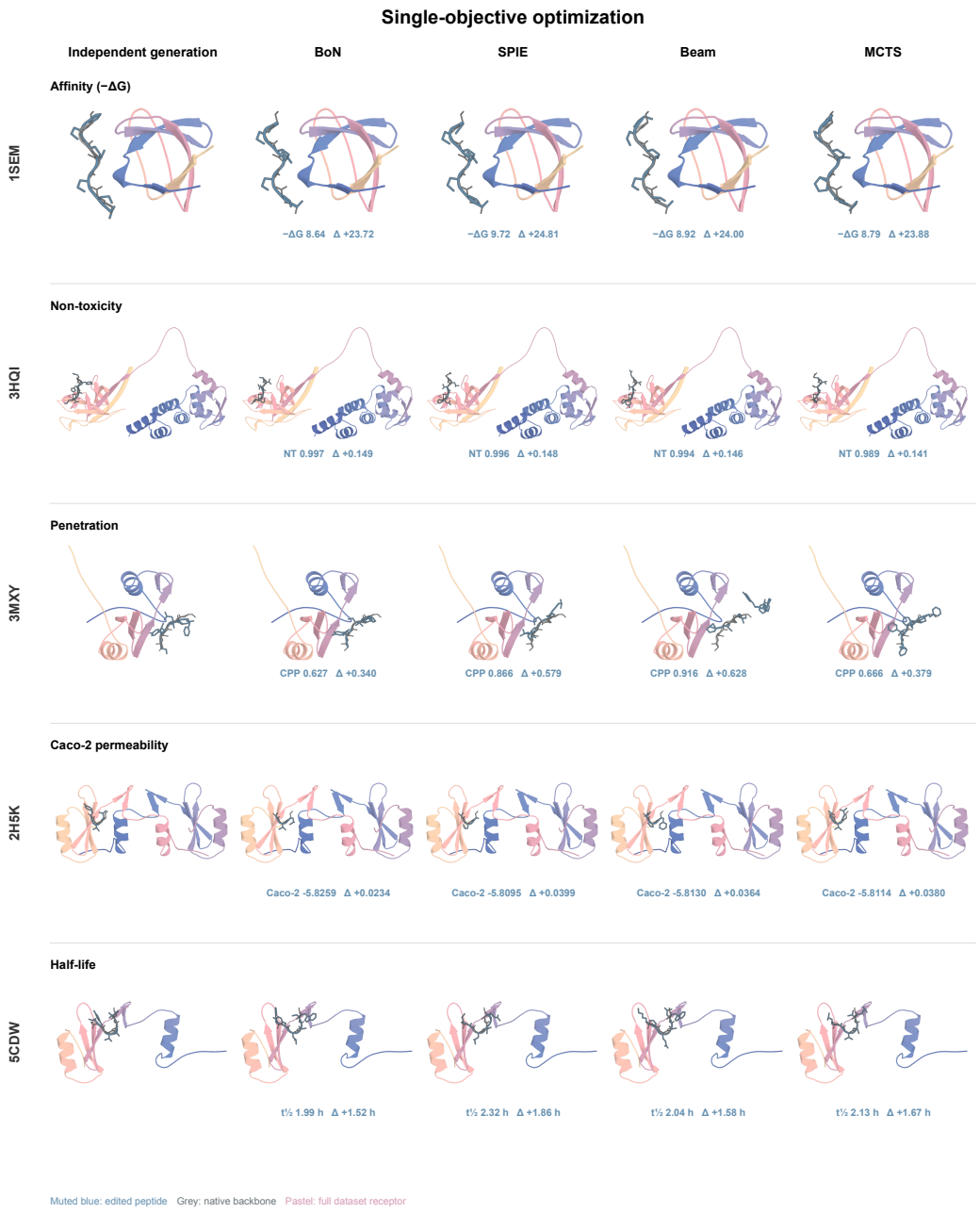



Figure 8: **Single-objective structural examples.** Rows show the interface-energy score (PDB 1SEM-C), predicted non-toxicity (3HQI-C), penetration (3MXY-L), Caco-2 permeability (2H5K-C), and half-life (5CDW-I). Columns compare independent generation with BoN, SPIE, beam search, and MCTS. Muted-blue edited peptides overlay gray native backbones; complete receptors use pastel colors. Labels give the displayed candidate's active-objective score and change from its run's starting complex. Rows were selected for four distinct positive improvements across search policies, without an additional chain-continuity filter. Each label and structure refer to the same candidate.

Table 15: Paired differences in single-objective improvement with window refinement. Entries are SPIE minus the named policy, with 95% confidence intervals from 10,000 paired bootstrap resamples over complexes. Positive values favor SPIE. Non-toxicity comparisons involving beam search or MCTS use separate reference evaluations (Appendix I.1).

| Objective | versus BoN | versus Beam | versus MCTS |
|---|---|---|---|
| $-\Delta G$ (REU) | 0.189<br>[-0.056, 0.416] | 0.606<br>[0.341, 0.912] | 1.128<br>[0.833, 1.476] |
| Non-toxicity | 0.038<br>[0.025, 0.053] | 0.005<br>[-0.001, 0.010] | 0.022<br>[0.014, 0.033] |
| Penetration | 0.098<br>[0.078, 0.119] | 0.026<br>[0.016, 0.036] | 0.065<br>[0.047, 0.084] |
| Caco-2 | 0.00476<br>[0.00399, 0.00550] | 0.00103<br>[0.00031, 0.00175] | 0.00324<br>[0.00254, 0.00394] |
| Half-life (h) | 0.560<br>[0.445, 0.681] | 0.210<br>[0.097, 0.319] | 0.419<br>[0.302, 0.537] |

Table 16: Final single-objective scores with 200-step joint flow integration. Entries are means with 95% confidence intervals over the test set, using EditFlow and 64 proposals per complex.

| Objective | BoN | SPIE | Beam | MCTS |
|---|---|---|---|---|
| $-\Delta G$ (REU) | 27.798<br>[25.301, 30.295] | 27.931<br>[25.441, 30.421] | 27.323<br>[24.833, 29.814] | 26.912<br>[24.379, 29.445] |
| Non-toxicity | 0.931<br>[0.912, 0.950] | 0.986<br>[0.975, 0.996] | 0.973<br>[0.962, 0.985] | 0.948<br>[0.929, 0.967] |
| Penetration | 0.772<br>[0.745, 0.798] | 0.866<br>[0.847, 0.884] | 0.842<br>[0.823, 0.860] | 0.792<br>[0.767, 0.817] |
| Caco-2 | -5.83822<br>[-5.83981, -5.83664] | -5.83284<br>[-5.83449, -5.83118] | -5.83411<br>[-5.83579, -5.83242] | -5.83695<br>[-5.83859, -5.83530] |
| Half-life (h) | 1.848<br>[1.735, 1.962] | 2.480<br>[2.331, 2.629] | 2.129<br>[1.998, 2.259] | 1.879<br>[1.771, 1.987] |

### I.2 Multi-objective results

Table 17: Multi-objective peptide optimization. Entries are mean final five-objective HV with 95% confidence intervals, using 64 proposals per complex. The first two rows share 200-step joint flow integration and search settings. The third uses window refinement (Section 2.3). Bold marks the highest mean in each column.

| Model | BoN | SPIE | Beam | MCTS |
|---|---|---|---|---|
| PepFlow | 0.1198<br>[0.0977, 0.1419] | 0.0711<br>[0.0554, 0.0869] | 0.0879<br>[0.0693, 0.1065] | 0.0698<br>[0.0535, 0.0861] |
| EditFlow | 0.1708<br>[0.1403, 0.2013] | **0.0973**<br>[0.0757, 0.1188] | 0.1135<br>[0.0911, 0.1358] | **0.1013**<br>[0.0812, 0.1215] |
| EditFlow,<br>window refinement | **0.1822**<br>[0.1501, 0.2143] | 0.0911<br>[0.0725, 0.1097] | **0.1338**<br>[0.1097, 0.1580] | 0.0939<br>[0.0754, 0.1124] |

Normalization and hypervolume follow Appendix G.8. With window refinement, the common mean starting HV is 0.04704. Final means are 0.18222 for BoN, 0.09109 for SPIE, 0.13384 for beam search, and 0.09393 for MCTS. Each method has one empty archive in the test set. Final candidate counts average 18.68, 19.16, 19.16, and 19.19, respectively, so BoN's higher HV is not explained by a larger archive.

The final BoN archives contain a candidate with no lower score in any objective and a higher score in at least one objective for 46.5% of test complexes. For 4.3% of test complexes, a candidate improves all five scores strictly. These conditions compare complete five-property vectors with zero

tolerance, and they describe computed scores rather than experimental binding or safety. Best-score curves that admit infeasible proposals are not interchangeable with final-archive performance.

### I.3 Individual-property endpoints in multi-objective search

Table 18: Property-wise endpoints from five-objective search with window refinement. Each column independently selects the best feasible score for that property from the final candidate set, then averages across complexes. Empty-set fallback follows Appendix G.6. The column optima can correspond to different peptides.

| Search | $-\Delta G$ (REU) | Non-tox. | Penetration | Caco-2 | Half-life (h) |
|---|---|---|---|---|---|
| BoN | 27.861 | 0.940 | 0.827 | -5.83649 | 1.789 |
| SPIE | 26.643 | 0.960 | 0.875 | -5.83430 | 1.879 |
| Beam | 27.272 | 0.963 | 0.869 | -5.83438 | 1.923 |
| MCTS | 26.904 | 0.958 | 0.871 | -5.83430 | 1.868 |

Table 18 reports the per-property endpoints obtained with the final-archive selection and fallback rule used for run summaries. Different columns can correspond to different candidates. They complement the joint candidate-set measure in Table 17.

### I.4 Representative multi-objective complexes

Figure 9 provides structural examples complementary to the aggregate hypervolume analysis. It places an independently generated peptide beside one representative from each search archive in the complete receptor context.

## J Molecular details of the GRB2 SH3 editing example

Figure 5C–G shows peptide chain B of PDB 1GBQ bound to the N-terminal SH3 domain of GRB2. The example uses EditFlow with beam search, width four, four children per expansion, and 64 proposals, completing each edit by residue-wise extension (Appendix G.7). Two accepted edits are selected from a four-layer search. The case evaluates nine computational properties, including solubility, and is separate from the five-objective comparison.

The first edit repacks side chains at peptide positions 6 and 9. The second permits identity changes at positions 1, 6, 7, 9, and 10, with proline retained at positions 6 and 7. The substitutions V1L, R9K, and R10S change the full sequence from VPPPVPPRRR to LPPPVPPRKS. Both operations keep the receptor and peptide backbone fixed. The molecular panels highlight substituted side-chain carbon atoms in orange. The local views compare Arg10 with Ser10 using the same receptor coordinates, viewing direction, and scale. Dashed guides show the minimum heavy-atom separation from the peptide side chain to each labeled receptor residue.

Eight peptide residues contact the receptor in both the initial and redesigned states. The fraction of peptide heavy atoms within 4.5 Å of the receptor changes from 0.512 to 0.513, and the number of contacting residue pairs changes from 23 to 22, so a retained backbone still admits reorganized side-chain interactions. The trajectory improves predicted solubility while predicted non-toxicity remains low in both states.

The two accepted edits move the reported properties in different directions. Predicted solubility rises from 0.033 to 0.991 and predicted half-life from 0.518 to 0.739 h, while predicted cell penetration falls from 0.964 to 0.881; the interface-energy score rises from 11.53 to 13.39 REU, having first reached 13.58 REU after the side-chain repacking alone. The C-terminal substitutions reduce the peptide's net positive charge and accompany higher predicted solubility and lower predicted cell penetration. No single candidate on this trajectory is best on every objective, which is the situation the multi-objective search of Section 3.3 is intended to expose.

The green path in Figure 5C,D connects the initial peptide to proposals 14 and 25. The alternative path ends at proposal 17 and trades a higher solubility score for a lower interface-energy score. Panel

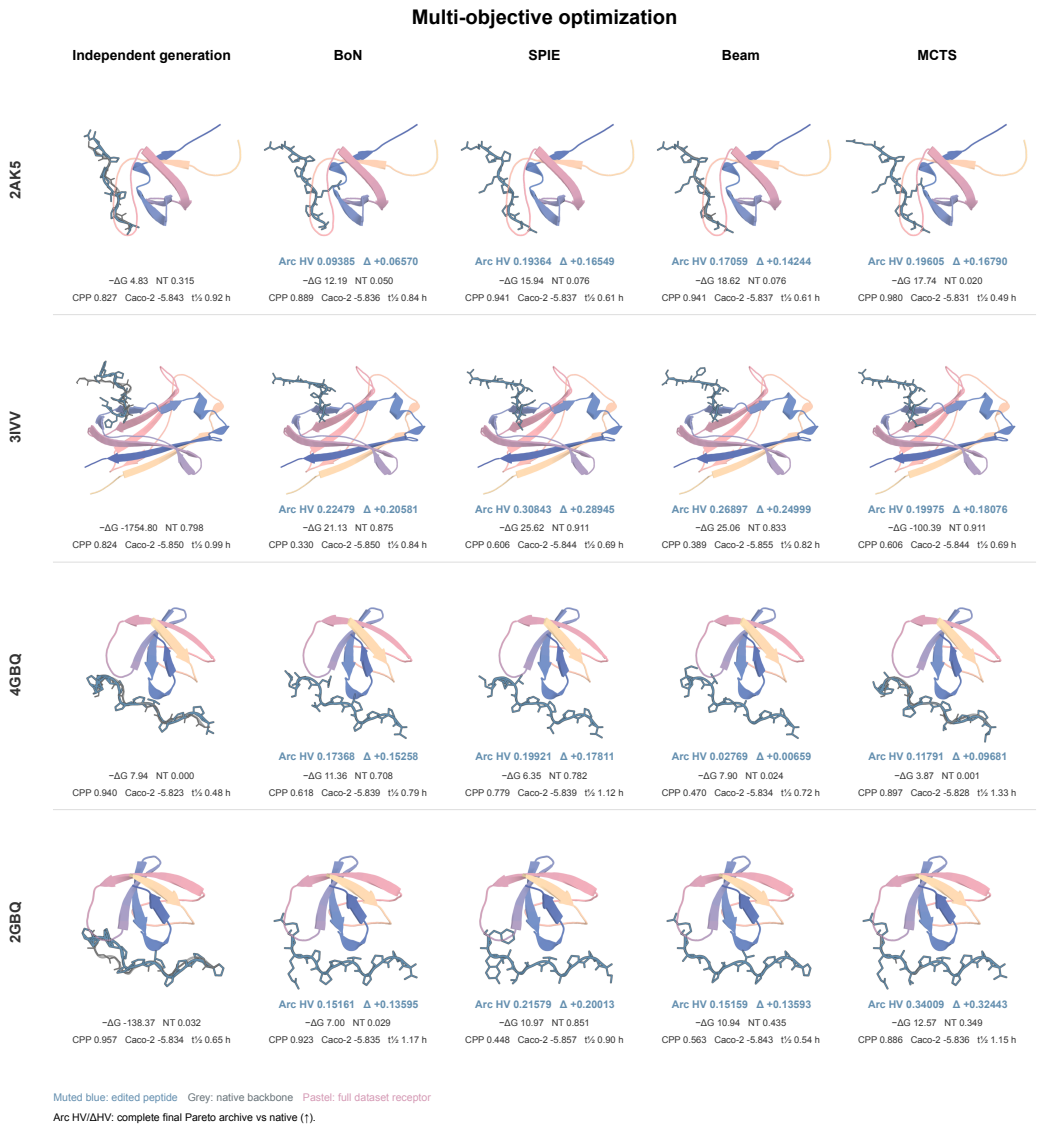



Figure 9: **Representative structures from independent generation and multi-objective editing.** Rows are PDB complexes 2AK5-D, 3IVV-D, 4GBQ-B, and 2GBQ-B. Columns compare independent generation with BoN, SPIE, beam search, and MCTS. Muted-blue edited peptides overlay gray native backbones; complete receptors use pastel colors. Arc HV and $\Delta$ quantify each method's complete final feasible archive. Search representatives are feasible, chain-continuous, edited, non-native candidates selected by maximum exclusive HV contribution. Small labels give computed $-\Delta G$, predicted non-toxicity, penetration, Caco-2 permeability, and half-life.

Table 19: Interface geometry of the GRB2 SH3 peptide before and after beam-search editing with residue-wise extension. Property changes are shown in Figure 5G.

| Property | Initial | Redesigned |
|---|---|---|
| Interface $\Delta$SASA (Å$^2$) | 982 | 884 |
| Contacting residue pairs | 23 | 22 |

C displays 28 proposals with nonnegative interface scores and the initial peptide. Panel D retains all 64 proposals. Candidate numbers indicate proposal order.

### J.1 An additional case using window refinement

The same GRB2 complex also yields a candidate from the five-objective window-refinement experiment with SPIE. Its sequence changes from VPPPVPPRRR to APPPTPRRRP, with substitutions V1A, V5T, P7R, and R10P. The peptide backbone (40 atoms) and receptor fragment (33 residues, 286 heavy atoms) have identical coordinates at PDB output precision. At a 4.5 Å heavy-atom contact threshold, 20 of 23 native residue contacts remain. Thresholds of 4.0 and 5.0 Å retain 16/21 and 25/29 contacts, respectively. Table 20 lists the property scores of this candidate.

Table 20: Scores before and after editing a GRB2 SH3 peptide with SPIE and window refinement. All edited scores refer to the same candidate. Solubility is an additional prediction outside the five search objectives.

| Property | Initial | Edited |
|---|---|---|
| Interface score $-\Delta G$ (REU) | 11.5348 | 12.3471 |
| Predicted non-toxicity | 0.000532 | 0.004498 |
| Predicted penetration | 0.96416 | 0.97711 |
| Predicted Caco-2 | -5.825979 | -5.825281 |
| Predicted half-life (h) | 0.51839 | 0.99593 |
| Predicted solubility | 0.03292 | 0.90585 |

The ancestry consists of a sequence edit, side-chain repacking, and a further sequence edit. The interface-energy score dips across the intermediate edits and recovers by the final candidate, so property gains are realized across a trajectory rather than within every individual edit. The final non-toxicity prediction remains low. Solubility is an additional reported property, not an objective in this five-property search. The geometry comparison uses the initial and final structures, since the final archive retains candidates rather than their intermediate parents.