ENERGY-GUIDED TEST-TIME REFINEMENT FOR FULL-ATOM PEPTIDE BINDER DESIGN

Anonymous authors
Paper under double-blind review

ABSTRACT

Peptide binder design generates binder sequences and structures conditioned on a target receptor. Across model architectures, scaling test-time sampling has become the dominant strategy to improve binding quality, yet it wastes computation on unpromising samples and overlooks residue-level signal. To address both, we introduce Energy-guided Test-Time Refinement (E-TTR), which reallocates the inference budget to high-potential candidates and refines them at the residue level. Concretely, given a receptor, we sample an initial pool of peptides and score each with Rosetta to obtain a binding energy together with its per-residue decomposition. For the best candidate, we re-noise only the high-energy residues and denoise them conditioned on the rest, producing a new candidate pool. This loop replaces parallel sampling with several sequential rounds, progressively improving binding quality within the same inference budget. Because standard denoisers share one timestep across the entire peptide, we post-train them via Diffusion Forcing to assign per-residue timesteps, making residue-level edits far more effective. Experiments across three benchmarks and two generative backbones demonstrate that E-TTR largely outperforms competing test-time scaling strategies in binding affinity and success rate, with up to 28% lower binding energy at matched sampling budgets.

1 INTRODUCTION

Peptides are short chains of amino acids with broad applications in therapeutics and molecular engineering (Wang et al., 2022; Bhardwaj et al., 2016). Peptide binder design aims to generate the 1D sequence and 3D structure of binders against a given receptor, which has become the *de facto* formulation for computational peptide design (Kong et al., 2024; Li et al., 2024). Progress on this task has so far come mainly from architectural innovation, spanning latent diffusion (Kong et al., 2024), flow matching (Li et al., 2024; Lin et al., 2024), and autoregressive models (Li et al., 2025a).

Recently, test-time scaling (TTS) has emerged as an effective, model-agnostic way to improve protein design quality (Didi et al., 2026; Hartman et al., 2025). It invests additional computation at inference, generating a large pool of candidates and selecting the most promising ones. Representative strategies range from basic best-of-N (BoN) sampling to search-based methods such as beam search (Didi et al., 2026), Iterative Partial Refinement (IPR) (Kang et al., 2026), Feynman–Kac Steering (FK) (Singhal et al., 2025), and Monte Carlo Tree Search (MCTS) (Kocsis & Szepesvári, 2006). However, current TTS methods share two fundamental limitations. First, candidates are sampled independently from pure noise. Once ranked, all but the best are discarded. As a result, most of the budget goes to samples that contribute nothing to the final design. Second, selection uses only the total binding energy. Yet its per-residue decomposition (Alford et al., 2017) reveals where a candidate falls short, and no existing TTS method exploits it. This raises a natural question: *How can we concentrate the inference budget on promising candidates and refine their weak positions during the search process?*

To answer this question, we propose Energy-guided Test-time Refinement (E-TTR). For a given receptor, E-TTR converts independent parallel sampling into multi-round sequential sampling, where each round builds on the best candidate of the last round, as shown in Figure 1a. This concentrates the sampling distribution around high-potential regions rather than spreading it over the full prior. Within each round, E-TTR exploits the per-residue decomposition of the binding energy to localize where a candidate underperforms. Once identified, E-TTR re-noises and denoises only these residues, leaving the others unchanged to preserve the substructure that already binds well. To support this residue-level

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

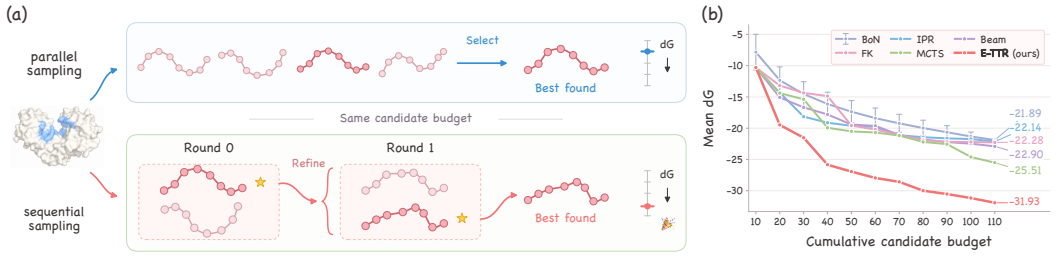


Figure 1: (a) E-TTR shifts a fixed candidate budget from independent parallel sampling to sequential, energy-guided refinement of promising candidates. (b) We compare sampling strategies on PepGLAD across 26 shared PepBench targets with increasing candidate budgets. E-TTR achieves lower mean dG than the baselines and continues to improve as the budget grows. dG is in kcal/mol.

editing, we post-train the denoiser with Diffusion Forcing (DF) (Chen et al., 2024), which assigns each residue an independent timestep, enabling it to denoise selected residues conditioned on clean ones. We compare E-TTR against representative TTS methods under the same sampling budgets in Figure 1b. As the budget grows, TTS strategies converge to around -23 dG, whereas E-TTR keeps improving and reaches -31.93 mean dG at the largest budget, demonstrating its effectiveness and scalability in improving binder design quality. The contributions of this paper are summarized below:

- We observe that existing TTS methods hit a quality ceiling in the peptide binder design task. To break through this barrier, we propose E-TTR, which replaces parallel generation with iterative sequential refinement, concentrating computation on the most promising candidate and yielding consistent improvements in peptide design quality.
- We exploit fine-grained per-residue energy decomposition to refine candidate peptides by regenerating their high-energy residues. To support this residue-level editing, we post-train the denoiser with Diffusion Forcing, enabling mixed-timestep denoising to preserve well-bound substructures while repairing weak regions.
- Experimental results show that E-TTR achieves the lowest median binding energy across all three benchmarks, with up to 28.3% improvement on PepBench. Moreover, E-TTR exhibits strong generalizability, improving all tested backbones, while other TTS methods work only on a subset.

2 Preliminaries

Before presenting our method, we introduce some key concepts relevant to this work. More related work and discussions can be found in Section 5.

Full-Atom Peptide Design. We represent a peptide as $G = (S, P)$, where S is the amino acid sequence and P denotes the 3D coordinates of backbone and side-chain heavy atoms. Existing peptide design methods (Kong et al., 2024; Li et al., 2024) typically employ an encoder-decoder architecture to map the full-atom representation into a residue-level latent space, and back, which can be formulated as:

$$R, X = \text{Encoder}(S, P), \quad \hat{S}, \hat{P} = \text{Decoder}(R, X),$$

where $R \in \mathbb{R}^{L\times d}$ and $X \in \mathbb{R}^{L\times d}$ denote the residue-level categorical and geometric features, respectively, L is the number of residues, and d is the dimension.

The generative process is then defined in the latent space over R and X via Diffusion (Ho et al., 2020) or Flow Matching (Lipman et al., 2022). The forward process jointly corrupts both features with a shared timestep $t \in [0, 1]$:

$$R_t = \alpha_t R_0 + \sigma_t \epsilon_R, \quad X_t = \alpha_t X_0 + \sigma_t \epsilon_X,$$

where $\epsilon_R, \epsilon_X \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. The noise schedule (α_t, σ_t) instantiates different frameworks, *e.g.*, $(\alpha_t = \sqrt{\bar{\alpha}_t}, \sigma_t = \sqrt{1-\bar{\alpha}_t})$ for diffusion models, and $(\alpha_t = 1 - t, \sigma_t = t)$ for flow models. A denoising network $f_\theta(R_t, X_t, t)$ is trained to reverse this process by predicting the noise:

$$(\hat{\epsilon}_R, \hat{\epsilon}_X) = f_\theta(R_t, X_t, t), \quad \mathcal{L}_{\text{DE}} = \mathbb{E}_{t,\epsilon_R,\epsilon_X}\left[\|\hat{\epsilon}_R - \epsilon_R\|^2 + \|\hat{\epsilon}_X - \epsilon_X\|^2\right]. \tag{1}$$

At inference, clean latent features (R_0, X_0) are denoised from pure noise $(R_1, X_1) \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ by integrating the learned reverse SDE or ODE, and then decoded into the full-atom peptide.

Rosetta Energy Evaluation. Rosetta provides an energy-based framework for macromolecular modeling and design, which we employ to evaluate peptide–receptor binding and guide refinement through residue-level energy decomposition. Given a peptide G and a target receptor C, Rosetta evaluates their complex with an energy function that combines weighted terms for atomic interactions and conformational preferences. The binding energy is defined as the energy difference between the bound and separated states of G and C:

$$E_{\text{rosetta_dG}}(G, C) = E_{\text{rosetta_bound}}(G, C) - E_{\text{rosetta_separated}}(G, C), \tag{2}$$

where a lower $E_{\text{rosetta_dG}}$ indicates a more favorable binding under this energy function. The binding energy can be decomposed into per-residue contributions, characterizing how each residue g_i in G contributes to $E_{\text{rosetta_dG}}$:

$$E_{\text{rosetta_dG}}(G, C) = \sum_{i=1}^{L} e_i^G + \sum_{j=1}^{L_C} e_j^C, \tag{3}$$

where L_C is the number of receptor residues, and e_i^G and e_j^C denote the bound-minus-separated energy contributions assigned to individual peptide and receptor residues, respectively. Under this decomposition, a lower e_i^G indicates indicates that residue g_i contributes more favorably to the overall binding energy.

3 METHOD

In this section, we describe E-TTR in detail. We first present how the Diffusion Forcing objective endows peptide generative models with residue-level editing capability. We then introduce the iterative refinement process guided by peptide-level and residue-level energy feedback. The overall pipeline is illustrated in Figure 2.

3.1 DIFFUSION FORCING POST-TRAINING

Existing peptide design methods (Kong et al., 2024; Li et al., 2024) follow a standard diffusion training procedure, applying a shared timestep to all tokens within a sample. Although effective for de novo generation, this shared-timestep training is misaligned with our residue-level refinement, where only a subset of tokens need regeneration, leading to marginal improvements. To address this mismatch, we employ the Diffusion Forcing (Chen et al., 2024) framework to post-train the base peptide design models, endowing them with residue-level editing capability. As shown in Figure 2a, for a peptide with L residues, we replace the scalar timestep t with a vector $\vec{t} = (t_1, \ldots, t_L)$, where t_i are independently sampled from a uniform distribution. The categorical and geometric features of each residue are then perturbed according to its assigned timestep:

$$R_{\vec{t},i} = \alpha_{t_i} R_{0,i} + \sigma_{t_i} \epsilon_{R,i}, \qquad X_{\vec{t},i} = \alpha_{t_i} X_{0,i} + \sigma_{t_i} \epsilon_{X,i}, \qquad i = \{1, \ldots, L\}. \tag{4}$$

In the following, we use $R_{\vec{t}}$ and $X_{\vec{t}}$ to denote the noised latent features of the whole peptide under the timestep vector. We then post-train the peptide generative models by predicting the input noise of each residue:

$$(\hat{\epsilon}_R, \hat{\epsilon}_X) = f_\theta(R_{\vec{t}}, X_{\vec{t}}, \vec{t}), \quad \mathcal{L}_{\text{DF}} = \mathbb{E}_{\vec{t},\, \epsilon_R,\, \epsilon_X} \left[\sum_{i:\, t_i > 0} \left(\|\hat{\epsilon}_{R,i} - \epsilon_{R,i}\|^2 + \|\hat{\epsilon}_{X,i} - \epsilon_{X,i}\|^2 \right) \right], \tag{5}$$

where the loss is evaluated only at noisy positions, while clean positions ($t_i = 0$) provide context. Additionally, the denoising process is conditioned on the receptor C, which we omit from the notation for simplicity. With this post-training, the denoiser can handle mixed-timestep inputs and denoise selected residues conditioned on clean ones. This capability forms the foundation of our test-time refinement framework, which we describe next.

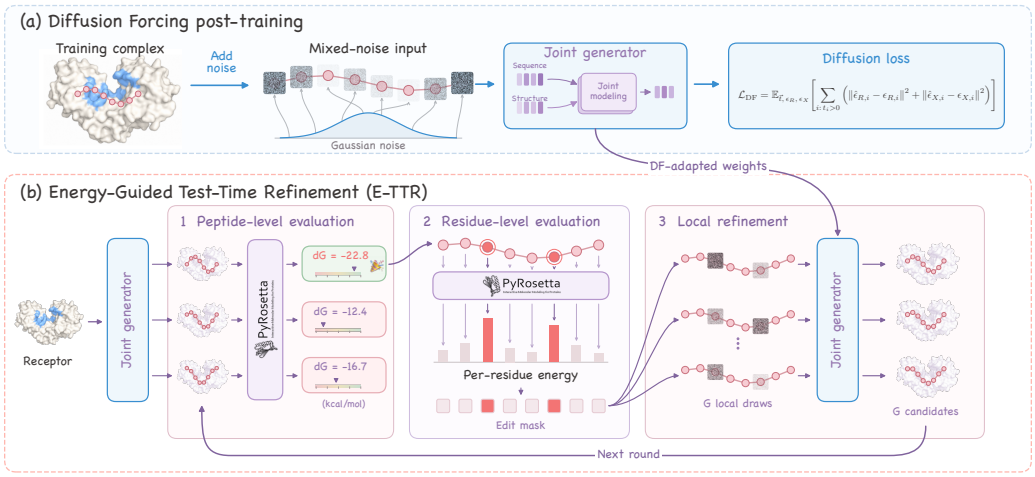


Figure 2: Overview of E-TTR. (a) Diffusion Forcing post-training independently samples per-residue timesteps, perturbs sequence and geometric features accordingly, and trains the generator to predict the added noise at noisy positions while using clean positions as context. (b) At each round, E-TTR selects the lowest-energy peptide using Rosetta and uses per-residue energy decomposition to identify high-energy residues for re-noising and joint sequence–structure regeneration.

3.2 Energy-Guided Test-Time Refinement

E-TTR operates at inference time, identifying and improving peptide candidates through energy-based evaluation and selective residue regeneration, as illustrated in Figure 2b. Specifically, we first sample B peptides independently for a target receptor C, which form a candidate pool $\mathcal{P}^{(0)} = \{G_b^{(0)}\}_{b=1}^{B}$ for the initialization of E-TTR. At each subsequent round, E-TTR selects the best candidate from the current pool based on binding energy, re-noises and regenerates its high-energy residues, and forms the next pool. We detail each step below.

Energy-guided Evaluation. Without loss of generality, we take the r-th round as an example to introduce the evaluation process based on Rosetta energy. Specifically, we evaluate each candidate in the current pool $\mathcal{P}^{(r)} = \{G_b^{(r)}\}_{b=1}^{B}$ using the Rosetta binding energy defined in Equation 2 and select the lowest-energy candidate as the starting point for refinement:

$$G_*^{(r)} = \arg\min_{G \in \mathcal{P}^{(r)}} E_{\text{rosetta_dG}}(G, C). \tag{6}$$

This ensures that each round builds upon the most promising candidate found so far. We then exploit the per-residue energy decomposition, defined in Equation 3, to localize where $G_*^{(r)}$ underperforms and select the top-k highest-energy residues to form the edit set:

$$\mathcal{M}^{(r)} = \text{TopK}_{i \in \{1,\dots,L\}}\left(e_i^{G_*^{(r)}}\right), \tag{7}$$

where TopK returns the indices of selected residues and $\mathcal{M}^{(r)}$ denotes the set of positions to be regenerated. This residue-level selection directs refinement toward residues with less favorable energy contributions, while the remaining residues still provide clean context.

Residue-level Refinement. Given the best candidate $G_*^{(r)}$ and its edit set $\mathcal{M}^{(r)}$, we now regenerate the targeted residues while keeping the rest fixed. Let $(R_*^{(r)}, X_*^{(r)})$ denote the clean sequence and geometric features of $G_*^{(r)}$. We construct a per-residue timestep vector $\vec{t}$ by assigning $t_i = m_i$ to each selected residue $i \in \mathcal{M}^{(r)}$ and $t_i = 0$ to all others, where $m_i \in (0, 1]$ controls the re-noising strength. We then re-noise the selected positions while retaining the clean features elsewhere:

$$\widetilde{R}_{\vec{t},i} = (1 - t_i)R_{*,i}^{(r)} + t_i\xi_{R,i}, \qquad \widetilde{X}_{\vec{t},i} = (1 - t_i)X_{*,i}^{(r)} + t_i\xi_{X,i}, \tag{8}$$

where $\xi_{R,i}$ and $\xi_{X,i}$ are sampled from the normal distribution. The re-noised peptide $(\widetilde{R}, \widetilde{X})$ is then denoised using the post-trained model with the per-residue timestep vector $\vec{t}$. At each denoising step,

Table 1: Peptide sequence–structure co-design results across two backbones and three datasets under a budget of 110 candidates per target. dG is reported in kcal/mol. Parentheses show relative improvements over BoN. Best and second-best results within each setting are bold and underlined, respectively. ↑ indicates higher is better and ↓ lower is better.

Method	dG mean ↓	dG median ↓	Success (%) ↑	RMSD (Å)	AAR (%)	Diversity ↑
Backbone: PepGLAD	*Dataset: PepBench (93 targets)*					
BoN-110	-19.28	-19.56	90.75	3.81	30.57	**0.709**
IPR	-18.94	-19.98	90.65	4.01	30.12	0.670
Beam	-20.37	-21.55	<u>93.01</u>	4.00	29.74	<u>0.681</u>
FK	-19.52	-19.39	91.94	4.00	29.24	0.673
MCTS	<u>-21.54</u>	<u>-21.65</u>	92.37	4.35	24.69	0.675
E-TTR	**-24.18** (+25.4%)	**-25.10** (+28.3%)	**95.48** (+5.2%)	5.08	28.66	0.680 (-4.0%)
Backbone: PepGLAD	*Dataset: PepBDB (98 targets)*					
BoN-110	-18.74	-18.35	84.39	11.34	23.70	0.701
IPR	-18.51	-18.33	82.96	13.22	24.56	0.693
Beam	**-22.26**	-19.86	85.61	12.02	22.79	0.706
FK	-19.42	-18.78	85.00	11.90	23.58	<u>0.714</u>
MCTS	-19.88	<u>-20.94</u>	<u>86.15</u>	12.86	21.18	0.696
E-TTR	<u>-21.27</u> (+13.5%)	**-21.91** (+19.4%)	**86.22** (+2.2%)	15.47	21.30	**0.719** (+2.6%)
Backbone: PepFlow	*Dataset: PepMerge (154 targets)*					
BoN-110	-28.59	<u>-27.74</u>	**98.90**	1.67	58.87	0.191
IPR	-28.42	-27.15	98.25	1.73	58.89	<u>0.193</u>
Beam	<u>-28.69</u>	-26.80	98.51	2.06	57.16	0.192
FK	-27.69	-26.40	98.51	1.96	56.99	0.188
MCTS	-26.71	-26.40	96.69	2.09	57.79	0.190
E-TTR	**-30.48** (+6.6%)	**-28.19** (+1.6%)	<u>98.64</u> (-0.3%)	1.91	57.06	**0.193** (+1.2%)

only the selected positions are updated while the clean positions remain unchanged:

$$(\widetilde{R}_{i,t_i-\Delta t}, \widetilde{X}_{i,t_i-\Delta t}) = \frac{\alpha_{t_i-\Delta t}}{\alpha_{t_i}}(\widetilde{R}_{i,t_i}, \widetilde{X}_{i,t_i}) + \left(\sigma_{t_i-\Delta t} - \frac{\alpha_{t_i-\Delta t}}{\alpha_{t_i}}\sigma_{t_i}\right)(\hat{\epsilon}_{R,i}, \hat{\epsilon}_{X,i}), \quad i \in \mathcal{M}^{(r)}, \tag{9}$$

where Δt represents a finite time interval, $(\hat{\epsilon}_R, \hat{\epsilon}_X) = f_\theta(\widetilde{R}, \widetilde{X}, \vec{t})$, and the denoising proceeds from $t_i = m_i$ to $t_i = 0$ for each selected residue. By repeating this re-noising and denoising procedure B times with independent prior samples, we obtain the next candidate pool $\mathcal{P}^{(r+1)} = \{G_b^{(r+1)}\}_{b=1}^{B}$. We iterate the evaluation and refinement steps for N rounds, yielding $B \times (N+1)$ candidates including the initial pool. All candidates are recorded in an archive, and the best one is selected as the final design. The complete procedure is summarized in Appendix A.

4 Experiments

4.1 Experimental Setup

Backbones and Datasets. E-TTR is designed as a model-agnostic TTS method for peptide binder design. To evaluate its effectiveness across different backbones, we apply it to two representative models, *i.e.*, PepGLAD and PepFlow, and evaluate on their respective test sets. For PepGLAD, we evaluate on PepBench (93 complexes) and PepBDB subset (98 complexes). For PepFlow, we evaluate on PepMerge (154 complexes). Details on data processing are provided in Appendix B.

Baselines. We compare E-TTR with BoN sampling and four refinement or search strategies: IPR (Kang et al., 2026), Beam Search (Didi et al., 2026), FK (Singhal et al., 2025), and MCTS (Kocsis & Szepesvári, 2006). All baselines are evaluated on both PepGLAD and PepFlow. Implementation details and backbone-specific adaptation are provided in Appendix D.

Evaluation Protocol. To assess binding affinity, we report the mean and median of the best Rosetta dG per target (kcal/mol), along with the success rate (dG < 0). We also report Cα RMSD, amino

Table 2: Ablation of DF post-training and E-TTR refinement across two backbones and three datasets with 110 candidates budget per target. dG is reported in kcal/mol. Parentheses show relative improvements over the setting with neither DF nor refinement.

DF	E-TTR	dG mean ↓	dG median ↓	Success (%) ↑	RMSD (Å)	AAR (%)	Diversity ↑
Backbone: PepGLAD		*Dataset: PepBench (93 targets)*					
×	×	-19.28	-19.56	90.75	3.81	30.57	0.709
✓	×	-19.77	-19.82	90.75	4.02	29.07	0.678
×	✓	-23.94	-22.95	**96.77**	5.34	26.60	**0.729**
✓	✓	**-24.18** (+25.4%)	**-25.10** (+28.3%)	95.48 (+5.2%)	5.08	28.66	0.680 (-4.0%)
Backbone: PepGLAD		*Dataset: PepBDB (98 targets)*					
×	×	-18.70	-17.71	83.27	11.05	22.56	0.754
✓	×	-18.74	-18.35	84.39	11.34	23.70	0.701
×	✓	-17.05	-16.23	76.43	14.31	21.68	**0.772**
✓	✓	**-21.27** (+13.7%)	**-21.91** (+23.7%)	**86.22** (+3.6%)	15.47	21.30	0.719 (-4.6%)
Backbone: PepFlow		*Dataset: PepMerge (154 targets)*					
×	×	-28.59	-27.74	**98.90**	1.67	58.87	0.191
✓	×	-28.96	-27.34	98.77	1.72	59.03	0.192
×	✓	-29.59	-27.92	98.18	1.73	56.95	**0.206**
✓	✓	**-30.48** (+6.6%)	**-28.19** (+1.6%)	98.64 (-0.3%)	1.91	57.06	0.193 (+1.2%)

acid recovery (AAR), and sequence–structure diversity. All methods generate 110 candidates per target, while E-TTR allocates them as 10 initial samples and 10 refinement rounds of 10 candidates each. Details on ecaluation implementation is provided in Appendix E.

4.2 Main Results

Table 1 summarizes the results across two generative backbones and their corresponding benchmarks, from which we make the following observations:

- E-TTR improves mean dG over BoN under the same candidate budget by 25.4% and 13.5% on PepBench and PepBDB with PepGLAD, respectively, and by 6.6% on PepMerge with PepFlow. These gains demonstrate the benefit of allocating candidates to energy-guided refinement.
- Unlike IPR, Beam, FK, and MCTS, which yield limited or no gains over BoN on PepFlow, E-TTR improves binding affinity across both diffusion-based and flow-based backbones. On PepFlow, it outperforms all evaluated baselines in both mean and median dG.
- E-TTR outperforms IPR and Beam Search in mean binding affinity on PepBench and the PepMerge, improving over Beam Search by 18.7% and 6.2%, respectively. Although Beam achieves a lower mean dG on filtered PepBDB, E-TTR leads in median dG and success rate.
- E-TTR achieves the highest joint sequence–structure diversity on filtered PepBDB and the PepMerge, while remaining close to Beam Search on PepBench (0.680 vs. 0.681). Although E-TTR does not lead in RMSD or AAR, these metrics measure structural and sequence similarity to the native reference rather than binding affinity. Closer recovery of a native peptide does not necessarily imply stronger binding, which is the primary objective of E-TTR.

4.3 Ablation Studies

We conduct ablation studies on three aspects of E-TTR: Component Contributions, Energy Feedback, and Re-noising Strength.

Component Contributions. We first evaluate the individual and combined effects of post-training and TTR under the same candidate budget. The results are shown in Table 2. We can observe that adding DF alone yields modest improvements in mean dG over independent sampling. Applying TTR without DF improves binding affinity on PepBench and PepMerge, but degrades performance on filtered PepBDB. Combining both components achieves the best mean and median dG in all three settings, demonstrating the necessity of both post-training and TTR.

Table 3: Ablation of energy-guided parent and residue selection on PepGLAD/PepBench with 110 candidates per target. Each selection is independently replaced by uniform random selection. dG is reported in kcal/mol. Parentheses show relative improvements over full E-TTR. Best and second-best results are bold and underlined, respectively.

Parent	Residues	dG mean ↓	dG median ↓	Success (%) ↑	RMSD (Å) ↓	AAR (%) ↑	Diversity ↑
Random	Energy	-21.47 (-11.2%)	<u>-21.00</u> (-16.3%)	<u>94.09</u> (-1.5%)	4.91	28.68	0.669 (-1.6%)
Energy	Random	<u>-21.59</u> (-10.7%)	-20.56 (-18.1%)	93.01 (-2.6%)	4.98	27.30	<u>0.680</u> (-0.01%)
Energy	Energy	**-24.18**	**-25.10**	**95.48**	5.08	28.66	**0.680**

Table 4: Effect of re-noising strength m_i on median dG across two backbones and three datasets. Each setting uses 10 initial candidates and 100 refinement candidates. Setting $m_i = 1.0$ means regenerating the selected residues from scratch.

Backbone	Dataset	Re-noising strength m_i			
		0.25	0.50	0.75	1.00
PepGLAD	PepBench	<u>-23.037</u>	-22.840	-22.691	**-25.096**
PepGLAD	PepBDB	-19.433	<u>-21.478</u>	-20.188	**-21.906**
PepFlow	PepMerge	-26.787	-26.902	<u>-27.390</u>	**-28.194**

Table 5: Budget allocation on PepGLAD/PepBench. 110 candidates in total. Median dG in kcal/mol.

Rounds	Samples/round	dG median ↓
2	50	-21.93
5	20	-22.99
10	10	**-25.10**
20	5	-24.30
50	2	<u>-24.41</u>

Energy Feedback. We evaluate the roles of Peptide-Level Evaluation and Residue-Level Evaluation by replacing their energy-guided selection with uniform random selection while keeping Local Refinement unchanged. As shown in Table 3, randomly selecting a candidate from the current pool increases mean dG from -24.18 to -21.47, while randomly selecting the same number of residues for the edit set increases it to -21.59. These results support both levels of energy feedback in guiding local refinement.

Re-noising Strength. We further examine the re-noising strength $m_i \in \{0.25, 0.5, 0.75, 1.0\}$ across all three settings under a fixed candidate budget. As shown in Table 4, full re-noising achieves the best median dG in all three settings, supporting our default choice of $m_i = 1.0$. This suggests that fully resetting selected residues may facilitate exploration beyond unfavorable local states, while unedited residues preserve the surrounding context.

4.4 Scalability

Budget Scaling. To evaluate how E-TTR benefits from additional sampling, we increase the candidate budget from 10 to 110 for both backbones. As shown in Figure 3, E-TTR with DF surpasses BoN-110 (110 candidates budget) without DF using only 40 candidates, and BoN-110 with DF using 50 candidates, on both backbones. Further refinement lowers mean dG to -25.88 and -31.54 kcal/mol on PepGLAD and PepFlow, respectively, at 110 candidates. These results demonstrate both improved candidate efficiency and continued gains with larger budgets.

Budget Allocation. We examine how to allocate a fixed budget between refinement rounds and candidates to refine per round. With 110 candidate budget and keeping the 10 initial candidates unchanged, we distribute the remaining 100 candidates across five allocations on PepGLAD with PepBench data. As shown in Table 5, our default allocation of 10 rounds with 10 candidates per round achieves the best median dG among the tested settings, suggesting that both refinement depth and candidate breadth contribute to effective budget allocation.

4.5 In-depth Analysis

Refinement Dynamics. To examine how E-TTR evolves during refinement, we track mean dG and cumulative co-diversity across rounds in Figure 4. In this analysis, co-diversity is computed over all candidates from the current and preceding rounds, using a different aggregation protocol from the main results; details are provided in Appendix E. On both backbones, cumulative co-diversity decreases and gradually levels off, while binding affinity continues to improve. This joint trend is

Table 6: Effect of the number of edited residues $|\mathcal{M}|$ on PepGLAD/PepBench with 110 candidates per target. The default setting ($|\mathcal{M}| = 3$) achieves the lowest mean and median dG. Parentheses show relative improvements over the default setting.

$\|\mathcal{M}\|$	dG mean ↓	dG median ↓	Success (%) ↑	RMSD (Å) ↓	AAR (%) ↑	Diversity ↑
1	-23.41 (-3.2%)	-21.55 (-14.1%)	**95.48** (0.0%)	5.21	24.29	**0.681** (+0.1%)
3	**-24.18**	**-25.10**	**95.48**	5.08	28.66	0.680
5	-23.19 (-4.1%)	-22.31 (-11.1%)	95.27 (-0.2%)	4.78	27.71	0.678 (-0.4%)

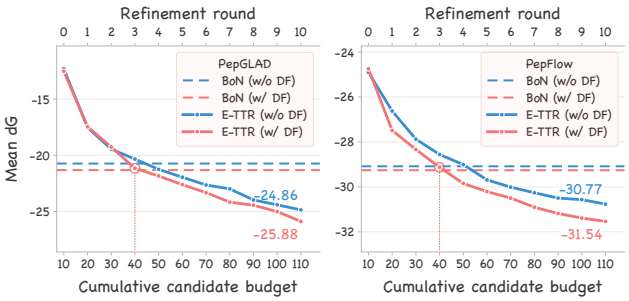


Figure 3: E-TTR improves binding affinity with increasing sampling budgets across both generative backbones.

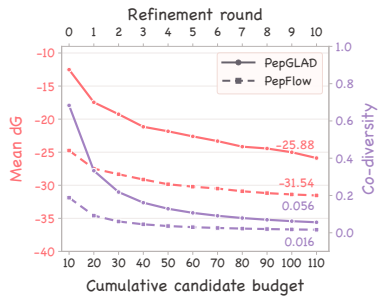


Figure 4: Binding affinity and diversity during E-TTR refinement on both backbones.

consistent with a progressively concentrated search that retains the capacity to improve candidates, supporting the effectiveness of iterative local refinement.

Peptide Length Dependence. To examine how peptide length affects refinement gains, we compare the median dG of E-TTR and BoN within each length group on PepGLAD backbone with test data from PepBench. As shown in Figure 5, with bubble areas proportional to target counts, E-TTR achieves lower median dG in most length groups. These results indicate that the benefits of E-TTR extend across peptide lengths, with a more stable improvement for longer peptides. We attribute this observation to longer peptides containing more residues may offer more opportunities for residue-level energy evaluation and local refinement for E-TTR to improve unfavorable sites while preserving the surrounding context.

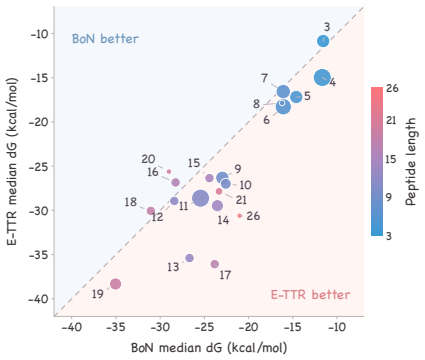


Figure 5: Median dG by peptide length on PepGLAD/PepBench.

Hyperparameters. We vary the nominal number of edited residues $|\mathcal{M}| \in \{1, 3, 5\}$ on PepGLAD with test data from PepBench while keeping the candidate budget and other settings fixed. As shown in Table 6, the default $|\mathcal{M}| = 3$ achieves the best mean and median dG, whereas $|\mathcal{M}| = 1$ and $|\mathcal{M}| = 5$ degrade median dG by 14.1% and 11.1%, respectively. Editing too few residues may limit the repair of unfavorable sites, while editing too many may disrupt favorable interactions. With similar success rates and joint diversity across settings, these results support $|\mathcal{M}| = 3$ as a balance between preserving existing context and enabling local improvement.

4.6 Visualization

Figure 6 compares binding conformations under the same candidate budget. Some baselines fail to improve upon their initial candidates and therefore return the same initial conformations despite additional sampling. In both examples, E-TTR identifies alternative conformations with lower Rosetta dG than all evaluated baselines. These comparisons illustrate how energy-guided residue-level refinement can use the available candidate budget to explore more favorable binding conformations, complementing the quantitative results above. Additional binding-conformation comparisons across PepBench, PepBDB, and PepMerge are provided in Appendix F

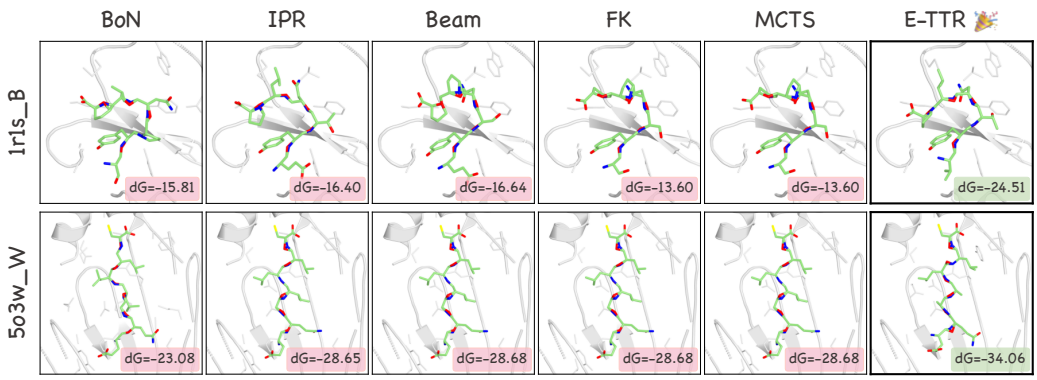


Figure 6: Binding conformations generated by different methods, with dG annotated in kcal/mol.

5 Related Work

Peptide Binder Design. Sequence-based models generate antimicrobial peptides (Das et al., 2018; Oort et al., 2021; Szymczak et al., 2023) or design peptide binders conditioned on target protein sequences (Chen et al., 2026), but do not explicitly generate their bound conformations or side-chain arrangements. Structure-aware approaches instead co-design peptide sequences and receptor-bound structures using diffusion (Wang et al., 2024; Kong et al., 2024) or flow matching (Lin et al., 2024; Li et al., 2024). Other methods construct peptides by autoregressively extending fragments from interaction hotspots (Li et al., 2025a) or jointly update sequence and structure parameter distributions through Bayesian flows (Qian et al., 2026). For cyclic peptides, AfCycDesign adapts AlphaFold2 with cyclic positional encoding for structure prediction and design (Rettie et al., 2025a), while RFpeptides combines diffusion-based backbone generation with sequence design to produce target-specific macrocyclic binders (Rettie et al., 2025b).

Test-Time Scaling. Test-time scaling improves outputs through additional inference computation, including repeated sampling and candidate selection in language and diffusion models (Snell et al., 2025; Brown et al., 2024; Ma et al., 2025). In language models, self-consistency aggregates answers across sampled reasoning paths (Wang et al., 2023), while Tree of Thoughts evaluates intermediate reasoning states to guide search and backtracking (Yao et al., 2023). In diffusion models, adaptive methods evaluate intermediate states to select promising continuations (Singhal et al., 2025; Li et al., 2025c) or expand selected branches through tree search (Guo et al., 2025; Li et al., 2025b). Iterative refinement revisits generated samples through repeated noising and reward-guided denoising (Uehara et al., 2025), while IPR regenerates selected regions conditioned on the remaining context without external verification (Kang et al., 2026). In biomolecular design, these strategies support multi-objective peptide sequence optimization (Tang et al., 2025) and reward-guided protein binder generation (Hartman et al., 2025; Didi et al., 2026). E-TTR combines peptide-level energy evaluation for candidate selection with residue-level evaluation to direct local sequence–structure refinement.

6 Conclusion

In this paper, we introduce Energy-guided Test-Time Refinement (E-TTR), which reallocates the inference budget to high-potential candidates through iterative residue-level refinement for peptide binder design. At each round, E-TTR selects the lowest-energy candidate from the current pool and exploits per-residue energy decomposition to direct local sequence–structure refinement toward high-energy residues while keeping the remaining residues unchanged. To support this residue-level editing, we post-train the denoiser with Diffusion Forcing to assign per-residue timesteps, enabling it to denoise selected residues conditioned on clean ones. Experiments across two generative backbones and three benchmarks demonstrate that E-TTR improves binding affinity at matched sampling budgets and achieves continued gains with additional sampling, supporting its effectiveness and applicability across generative models. These findings highlight residue-level energy feedback as an effective means of scaling test-time computation for peptide binder design.

AI USE STATEMENT

We used large language models to assist with manuscript writing and revision, including improving clarity, refining sentence structure, reducing redundancy, and maintaining consistent terminology. The authors directed the revisions and retain full responsibility for the scientific claims, interpretation of results, and final content of the paper.

REPRODUCIBILITY STATEMENT

To facilitate reproduction of our results, we describe E-TTR in the main text and provide implementation and experimental details in the appendix. Appendix A presents the complete refinement algorithm and its implementation on both generative backbones. Dataset sources, splits, and preprocessing procedures are documented in Appendix B. Appendix C specifies the DF post-training configuration, sampling hyperparameters, and computational environment, while Appendix D details baseline implementations and backbone-specific adaptations. Appendix E describes the Rosetta scoring procedure, evaluation metrics, and aggregation protocols for the main results and refinement analyses. Additional results and binding-conformation visualizations are provided in Appendix F.

ETHICS STATEMENT

This study investigates computational refinement of peptide binders using existing public datasets, without collecting personal information or conducting experiments on humans or animals. Our evaluation is entirely computational and does not involve peptide synthesis or biological testing. This work is intended to advance computational methods for peptide binder design. We encourage responsible use of the method, with appropriate experimental validation and biosafety considerations guiding downstream applications.

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, Jr., 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. URL https://pubs.acs.org/doi/10.1021/acs.jctc.7b00125.

Gaurav Bhardwaj, Vikram Khipple Mulligan, Christopher D. Bahl, Jason M. Gilmore, Peta J. Harvey, Olivier Cheneval, Garry W. Buchko, Surya V. S. R. K. Pulavarti, Quentin Kaas, Alexander Eletsky, Po-Ssu Huang, William A. Johnsen, Per Greisen, Gabriel J. Rocklin, Yifan Song, Thomas W. Linsky, Andrew M. Watkins, Stephen A. Rettie, Xianzhong Xu, Lauren P. Carter, Richard Bonneau, James M. Olson, Evangelos A. Coutsias, Colin E. Correnti, Thomas Szyperski, David J. Craik, and David Baker. Accurate *de novo* design of hyperstable constrained peptides. *Nat.*, 538 (7625):329–335, 2016. doi: 10.1038/NATURE19791. URL https://doi.org/10.1038/nature19791.

Bradley C. A. Brown, Jordan Juravsky, Ryan Ehrlich, Ronald Clark, Quoc V. Le, Christopher Ré, and Azalia Mirhoseini. Large language monkeys: Scaling inference compute with repeated sampling. *CoRR*, abs/2407.21787, 2024. doi: 10.48550/ARXIV.2407.21787. URL https://doi.org/10.48550/arXiv.2407.21787.

Boyuan Chen, Diego Marti Monso, Yilun Du, Max Simchowitz, Russ Tedrake, and Vincent Sitzmann. Diffusion forcing: Next-token prediction meets full-sequence diffusion. In Amir Globersons, Lester Mackey, Danielle Belgrave, Angela Fan, Ulrich Paquet, Jakub M. Tomczak, and Cheng Zhang (eds.), *Advances in Neural Information Processing Systems 37: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024. URL http://papers.nips.cc/paper_files/paper/2024/hash/2aee1c4159e48407d68fe16ae8e6e49e-Abstract-Conference.html.

Leo Tianlai Chen, Zachary Quinn, Madeleine Dumas, Christina Peng, Lauren Hong, Moises Lopez-Gonzalez, Alexander Mestre, Rio Watson, Sophia Vincoff, Lin Zhao, Jianli Wu, Audrey Stavrand, Mayumi Schaepers-Cheu, Tian Zi Wang, Divya Srijay, Connor Monticello, Pranay Vure, Rishab Pulugurta, Sarah Pertsemlidis, Kseniia Kholina, Shrey Goel, Matthew P. DeLisa, Jen-Tsan Ashley Chi, Ray Truant, Hector C. Aguilar, and Pranam Chatterjee. Target sequence-conditioned design of peptide binders using masked language modeling. *Nature Biotechnology*, 44(6):1002–1010, 2026. doi: 10.1038/s41587-025-02761-2. URL https://www.nature.com/articles/s41587-025-02761-2.

Payel Das, Kahini Wadhawan, Oscar Chang, Tom Sercu, Cícero Nogueira dos Santos, Matthew Riemer, Inkit Padhi, Vijil Chenthamarakshan, and Aleksandra Mojsilovic. Pepcvae: Semi-supervised targeted design of antimicrobial peptide sequences. *CoRR*, abs/1810.07743, 2018. URL http://arxiv.org/abs/1810.07743.

Kieran Didi, Zuobai Zhang, Guoqing Zhou, Danny Reidenbach, Zhonglin Cao, Sooyoung Cha, Tomas Geffner, Christian Dallago, Jian Tang, Michael M. Bronstein, Martin Steinegger, Emine Küçükbenli, Arash Vahdat, and Karsten Kreis. Scaling atomistic protein binder design with generative pretraining and test-time compute. *CoRR*, abs/2603.27950, 2026. doi: 10.48550/ARXIV.2603.27950. URL https://doi.org/10.48550/arXiv.2603.27950.

Yingqing Guo, Yukang Yang, Hui Yuan, and Mengdi Wang. Training-free guidance beyond differentiability: Scalable path steering with tree search in diffusion and flow models. In Danielle Belgrave, Cheng Zhang, Laura N. Montoya, Hsuan-Tien Lin, Razvan Pascanu, Piotr Koniusz, Marzyeh Ghassemi, Nancy Chen, Iván Vladimir Meza Ruíz, and Arturo Loaiza-Bonilla (eds.), *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2025, NeurIPS 2025, San Diego, CA, USA, December 2-7, 2025 / Mexico City, Mexico, November 30 - December 5, 2025*, 2025. URL http://papers.nips.cc/paper_files/paper/2025/hash/6a14c7f9fb3f42645cfa6bd5aa446819-Abstract-Conference.html.

Erik Hartman, Jonas Wallin, Johan Malmström, and Jimmy Olsson. Controllable protein design through feynman-kac steering. *CoRR*, abs/2511.09216, 2025. doi: 10.48550/ARXIV.2511.09216. URL https://doi.org/10.48550/arXiv.2511.09216.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin (eds.), *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/4c5bcfec8584af0d967f1ab10179ca4b-Abstract.html.

Taegu Kang, Jaesik Yoon, and Sungjin Ahn. Inference-time scaling in diffusion models through iterative partial refinement. *CoRR*, abs/2605.19317, 2026. doi: 10.48550/ARXIV.2605.19317. URL https://doi.org/10.48550/arXiv.2605.19317.

Levente Kocsis and Csaba Szepesvári. Bandit based monte-carlo planning. In Johannes Fürnkranz, Tobias Scheffer, and Myra Spiliopoulou (eds.), *Machine Learning: ECML 2006, 17th European Conference on Machine Learning, Berlin, Germany, September 18-22, 2006, Proceedings*, volume 4212 of *Lecture Notes in Computer Science*, pp. 282–293. Springer, 2006. doi: 10.1007/11871842_29. URL https://doi.org/10.1007/11871842_29.

Xiangzhe Kong, Yinjun Jia, Wenbing Huang, and Yang Liu. Full-atom peptide design with geometric latent diffusion. In Amir Globersons, Lester Mackey, Danielle Belgrave, Angela Fan, Ulrich Paquet, Jakub M. Tomczak, and Cheng Zhang (eds.), *Advances in Neural Information Processing Systems 37: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024. URL http://papers.nips.cc/paper_files/paper/2024/hash/88ad9774ffcb7a272457e9396f793a07-Abstract-Conference.html.

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 Ruslan Salakhutdinov, Zico Kolter, Katherine A. Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix

Berkenkamp (eds.), *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*, volume 235 of *Proceedings of Machine Learning Research*, pp. 27615–27640. PMLR / OpenReview.net, 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 *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net, 2025a. URL https://openreview.net/forum?id=jqmptcSNVG.

Xiner Li, Masatoshi Uehara, Xingyu Su, Gabriele Scalia, Tommaso Biancalani, Aviv Regev, Sergey Levine, and Shuiwang Ji. Dynamic search for inference-time alignment in diffusion models. *CoRR*, abs/2503.02039, 2025b. doi: 10.48550/ARXIV.2503.02039. URL https://doi.org/10.48550/arXiv.2503.02039.

Xiner Li, Yulai Zhao, Chenyu Wang, Gabriele Scalia, Gökcen Eraslan, Surag Nair, Tommaso Biancalani, Shuiwang Ji, Aviv Regev, Sergey Levine, and Masatoshi Uehara. Derivative-free guidance in continuous and discrete diffusion models with soft value-based decoding. In Danielle Belgrave, Cheng Zhang, Laura N. Montoya, Hsuan-Tien Lin, Razvan Pascanu, Piotr Koniusz, Marzyeh Ghassemi, Nancy Chen, Iván Vladimir Meza Ruíz, and Arturo Loaiza-Bonilla (eds.), *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2025, NeurIPS 2025, San Diego, CA, USA, December 2-7, 2025 / Mexico City, Mexico, November 30 - December 5, 2025*, 2025c. URL http://papers.nips.cc/paper_files/paper/2025/hash/899af0d66d8850318a20781484416152-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 Ruslan Salakhutdinov, Zico Kolter, Katherine A. Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*, volume 235 of *Proceedings of Machine Learning Research*, pp. 30510–30528. PMLR / OpenReview.net, 2024. URL https://proceedings.mlr.press/v235/lin24z.html.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *CoRR*, abs/2210.02747, 2022. doi: 10.48550/ARXIV.2210.02747. URL https://doi.org/10.48550/arXiv.2210.02747.

Nanye Ma, Shangyuan Tong, Haolin Jia, Hexiang Hu, Yu-Chuan Su, Mingda Zhang, Xuan Yang, Yandong Li, Tommi S. Jaakkola, Xuhui Jia, and Saining Xie. Scaling inference time compute for diffusion models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2025, Nashville, TN, USA, June 11-15, 2025*, pp. 2523–2534. Computer Vision Foundation / IEEE, 2025. doi: 10.1109/CVPR52734.2025.00241. URL https://openaccess.thecvf.com/content/CVPR2025/html/Ma_Scaling_Inference_Time_Compute_for_Diffusion_Models_CVPR_2025_paper.html.

Colin Michael Van Oort, Jonathon B. Ferrell, Jacob M. Remington, Safwan Wshah, and Jianing Li. AMPGAN v2: Machine learning-guided design of antimicrobial peptides. *J. Chem. Inf. Model.*, 61(5):2198–2207, 2021. doi: 10.1021/ACS.JCIM.0C01441. URL https://doi.org/10.1021/acs.jcim.0c01441.

Hao Qian, Shikui Tu, and Lei Xu. Full-atom peptide design via riemannian-euclidean bayesian flow networks. In Sven Koenig, Chad Jenkins, and Matthew E. Taylor (eds.), *Fortieth AAAI Conference on Artificial Intelligence, Thirty-Eighth Conference on Innovative Applications of Artificial Intelligence, Sixteenth Symposium on Educational Advances in Artificial Intelligence, AAAI 2026, Singapore, January 20-27, 2026*, pp. 24918–24926. AAAI Press, 2026. doi: 10.1609/AAAI.V40I29.39679. URL https://doi.org/10.1609/aaai.v40i29.39679.

Stephen A. Rettie, Katelyn V. Campbell, Asim K. Bera, Alex Kang, Simon Kozlov, Yensi Flores Bueso, Joshmyn De La Cruz, Maggie Ahlrichs, Suna Cheng, Stacey R. Gerben, Mila Lamb, Analisa Murray, Victor Adebomi, Guangfeng Zhou, Frank DiMaio, Sergey Ovchinnikov, and

Gaurav Bhardwaj. Cyclic peptide structure prediction and design using AlphaFold2. *Nature Communications*, 16(1), 2025a. doi: 10.1038/s41467-025-59940-7. URL https://www.nature.com/articles/s41467-025-59940-7.

Stephen A. Rettie, David Juergens, Victor Adebomi, Yensi Flores Bueso, Qinqin Zhao, Alexandria N. Leveille, Andi Liu, Asim K. Bera, Joana A. Wilms, Alina Üffing, Alex Kang, Evans Brackenbrough, Mila Lamb, Stacey R. Gerben, Analisa Murray, Paul M. Levine, Maika Schneider, Vibha Vasireddy, Sergey Ovchinnikov, Oliver H. Weiergräber, Dieter Willbold, Joshua A. Kritzer, Joseph D. Mougous, David Baker, Frank DiMaio, and Gaurav Bhardwaj. Accurate de novo design of high-affinity protein-binding macrocycles using deep learning. *Nature Chemical Biology*, 21:1948–1956, 2025b. doi: 10.1038/s41589-025-01929-w. URL https://www.nature.com/articles/s41589-025-01929-w.

Raghav Singhal, Zachary Horvitz, Ryan Teehan, Mengye Ren, Zhou Yu, Kathleen McKeown, and Rajesh Ranganath. A general framework for inference-time scaling and steering of diffusion models. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Forty-second International Conference on Machine Learning, ICML 2025, Vancouver, BC, Canada, July 13-19, 2025*, volume 267 of *Proceedings of Machine Learning Research*. PMLR / OpenReview.net, 2025. URL https://proceedings.mlr.press/v267/singhal25b.html.

Charlie Victor Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling LLM test-time compute optimally can be more effective than scaling parameters for reasoning. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net, 2025. URL https://openreview.net/forum?id=4FWAwZtd2n.

Paulina Szymczak, Marcin Możejko, Tomasz Grzegorzek, Radosław Jurczak, Marta Bauer, Damian Neubauer, Karol Sikora, Michał Michalski, Jacek Sroka, Piotr Setny, Wojciech Kamysz, and Ewa Szczurek. Discovering highly potent antimicrobial peptides with deep generative model HydrAMP. *Nature Communications*, 14(1):1453, 2023. doi: 10.1038/s41467-023-36994-z. URL https://www.nature.com/articles/s41467-023-36994-z.

Sophia Tang, Yinuo Zhang, and Pranam Chatterjee. Peptune: De novo generation of therapeutic peptides with multi-objective-guided discrete diffusion. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Forty-second International Conference on Machine Learning, ICML 2025, Vancouver, BC, Canada, July 13-19, 2025*, volume 267 of *Proceedings of Machine Learning Research*. PMLR / OpenReview.net, 2025. URL https://proceedings.mlr.press/v267/tang25n.html.

Masatoshi Uehara, Xingyu Su, Yulai Zhao, Xiner Li, Aviv Regev, Shuiwang Ji, Sergey Levine, and Tommaso Biancalani. Reward-guided iterative refinement in diffusion models at test-time with applications to protein and DNA design. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Forty-second International Conference on Machine Learning, ICML 2025, Vancouver, BC, Canada, July 13-19, 2025*, volume 267 of *Proceedings of Machine Learning Research*. PMLR / OpenReview.net, 2025. URL https://proceedings.mlr.press/v267/uehara25a.html.

Fanhao Wang, Yuzhe Wang, Laiyi Feng, Changsheng Zhang, and Luhua Lai. Target-specific *De Novo* peptide binder design with diffpepbuilder. *J. Chem. Inf. Model.*, 64(24):9135–9149, 2024. doi: 10.1021/ACS.JCIM.4C00975. URL https://doi.org/10.1021/acs.jcim.4c00975.

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), 2022. doi: 10.1038/s41392-022-00904-4.

Lijuan Wang, Chen Qiu, Liwei Xiao, and Junjie Chen. Peppfdpo: Multi-objective antimicrobial peptide generation using pareto-frontier enhanced direct preference optimization. In *IEEE International Conference on Bioinformatics and Biomedicine, BIBM 2025, Wuhan, China, December 15-18, 2025*, pp. 1–9. IEEE, 2025. doi: 10.1109/BIBM66473.2025.11356223. URL https://doi.org/10.1109/BIBM66473.2025.11356223.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V. Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023. URL https://openreview.net/forum?id=1PL1NIMMrw.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine (eds.), *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/271db9922b8d1f4dd7aaef84ed5ac703-Abstract-Conference.html.

Xiangxin Zhou, Dongyu Xue, Ruizhe Chen, Zaixiang Zheng, Liang Wang, and Quanquan Gu. Antigen-specific antibody design via direct energy-based preference optimization. In Amir Globersons, Lester Mackey, Danielle Belgrave, Angela Fan, Ulrich Paquet, Jakub M. Tomczak, and Cheng Zhang (eds.), *Advances in Neural Information Processing Systems 37: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024. URL http://papers.nips.cc/paper_files/paper/2024/hash/daef77101ba5711084a57442c8cf2709-Abstract-Conference.html.

A Complete E-TTR Algorithm

Algorithm 1 Energy-guided Test-Time Refinement (E-TTR)

Require: Receptor C; frozen generator f_θ; pool size B; rounds N; edit count $|\mathcal{M}|$; strengths $\{m_i\}$
Ensure: Archive $\mathcal{A}$ and best candidate $G^\dagger$

```
Sample initial pool P^(0) of size B conditioned on C
Score P^(0) using Eqs. 2 and 3
A ← P^(0)
for r = 0, ..., N − 1 do
    Select parent G_*^(r) using Eq. 6
    Select edit set M^(r) using Eq. 7
    Retrieve parent state (R_*^(r), X_*^(r))
    P^(r+1) ← ∅
    for b = 1, ..., B do
        Re-noise selected states using Eq. 8 with fresh noise
        Denoise using Eq. 9, fixing unselected states
        Decode G_b^(r+1); save its clean state
        Append G_b^(r+1) to P^(r+1)
    end for
    Score P^(r+1) using Eqs. 2 and 3
    Append P^(r+1) and scores to A
end for
Select lowest-energy G† from A
return A, G†
```

B Datasets and Preprocessing

Data sources and splits. For PepGLAD (Kong et al., 2024), DF post-training uses the released PepBench training and validation sets of 4,157 and 114 complexes, respectively. Evaluation uses 93 PepBench complexes and the 98-complex PepBDB subset described below. For PepFlow (Li et al., 2024), we partition the released training cache by PDB identifier into 9,356 training and 493 validation complexes using seed 12. This split is used for DF post-training; evaluation uses the released 154-complex PepMerge test set.

Table 7: Dataset statistics after processing; paired values in DF training rows denote train/val splits.

Dataset	Backbone	Usage	# Complexes	Length range (aa)	Median (aa)
PepBench	PepGLAD	DF training	4,157 / 114	4–25 / 4–25	10 / 13
PepMerge	PepFlow	DF training	9,356 / 493	3–25 / 3–25	10 / 10
PepBench	PepGLAD	Evaluation	93	3–26	10
PepBDB	PepGLAD	Evaluation	98	3–49	21
PepMerge	PepFlow	Evaluation	154	3–22	10

PepBDB filtering. Our PepGLAD models are trained and post-trained on PepBench, whereas the original PepBDB test set includes complexes that overlap with, or have similar receptors to, PepBench training examples. Evaluating the full set would therefore mix new targets with targets already represented in the training data. We screen the 190 cached complexes from the original PepBDB test set against PepBench training and validation data. A complex is removed if it shares a PDB identifier or an identical receptor sequence with a reference example. We also remove complexes with similar receptor sequences, identified using MMseqs2: the alignment must span at least 80% of each receptor's full sequence, and at least 40% of the aligned positions must contain identical amino acids. This leaves 117 complexes. Applying the same rules against the available ProtFrag pretraining data reduces the set to 105; Finally, we remove seven additional complexes that overlap with the PepBench test set under the same criteria, yielding 98 PepBDB test complexes. Filtering

was completed before generating candidates on PepBDB. These exclusions are based on the available data records, rather than a complete reconstruction of the released checkpoint's historical pretraining data.

Input preparation. We retain each backbone's cached receptor-pocket definitions and reference peptide lengths. PepGLAD uses the cached pocket residue indices and atom masks; PepFlow uses the released pocket tensors, including coordinate centering and torsion masks. For the 39 PepBDB peptides exceeding the PepGLAD checkpoint's 28-residue positional range, unseen positions reuse the final learned position embedding. Peptides are not truncated, and checkpoint files remain unchanged.

C Diffusion Forcing Post-Training Details

PepGLAD. We adapt the released PepGLAD sequence–structure co-design model (Kong et al., 2024) as follows.

- **Residue states and trainable parameters.** The pretrained autoencoder maps each peptide into the clean sequence-associated and geometric latent features R_0 and X_0 used in Equation 4. During post-training, only the noise prediction network is trainable.
- **Residue-wise time sampling.** Each peptide residue independently samples an integer step s_i uniformly from $\{0, \dots, 100\}$, shared by its sequence and geometric features and corresponding to normalized time $t_i = s_i/100$. Step 0 denotes clean context. If all peptide timesteps in a batch are zero, we randomly choose one residue and resample its step uniformly from $\{1, \dots, 100\}$.
- **Constructing mixed-noise inputs.** For residue i, we independently sample standard Gaussian noise for its sequence latent $R_{0,i}$ and geometric latent $X_{0,i}$. We perturb both features according to Equation 4, with the signal and noise coefficients taken from PepGLAD's original diffusion schedules at the assigned step s_i. We retain its geometric normalization, positional features, atom masks, and receptor–peptide graph construction. Receptor inputs remain uncorrupted.
- **Time conditioning.** Each peptide residue receives the native time encoding $[\beta_{s_i}, \sin \beta_{s_i}, \cos \beta_{s_i}]$, where β_{s_i} is the diffusion schedule value at step s_i. For receptor nodes, we first average these schedule values over the peptide in the same complex, then apply the same encoding.
- **Masked training objectives.** Following Equation 5, supervision is restricted to peptide residues with $s_i > 0$. Geometric squared errors are summed over coordinates and averaged over valid latent channels at these residues; sequence-feature errors are summed over feature dimensions and averaged over these residues. We retain the pretrained model's relative weighting of the two terms.

PepFlow. We adapt the released PepFlow model (Li et al., 2024) while retaining its representations and training objectives for structure and sequence.

- **Residue states and trainable parameters.** PepFlow represents each residue by its Cα position, backbone-frame rotation, torsion angles, and amino-acid identity. Starting from the released `model1.pt` checkpoint, we train the geometric attention network and its prediction heads while freezing the node and edge conditioning encoders.
- **Residue-wise time sampling.** We independently sample a noise level s_i uniformly from $\{0, \dots, 100\}$ for each peptide residue. PepFlow's native time τ_i runs from noise (0) to data (1), reversing the corruption-time convention in Equation 4. We set $\tau_i = 1$ for $s_i = 0$ and $\tau_i = 0.01 + 0.98(100 - s_i)/99$ otherwise. The resulting time is shared by the residue's position, rotation, angles, and sequence. If a complex has no noisy peptide residue, one residue is chosen uniformly and assigned a level sampled uniformly from $\{1, \dots, 100\}$.
- **Perturbing geometric states.** We sample Gaussian positions, centered over the peptide, and linearly interpolate them with the observed Cα positions. Backbone rotations are interpolated along geodesics from uniformly sampled rotations on $\mathrm{SO}(3)$ to the observed rotations. Torsion angles are interpolated from uniform angular samples to the observed angles using PepFlow's periodic interpolation, which accounts for angle wraparound. Each interpolation uses the assigned time τ_i and retains the original prior settings.

- **Perturbing amino-acid identities.** We convert each observed amino-acid identity into PepFlow's continuous sequence encoding and interpolate it with a Gaussian vector using the original scaling. Applying softmax to the interpolated vector gives a categorical distribution from which the noisy amino-acid identity is sampled. Clean peptide residues and receptor nodes retain their observed geometry and amino-acid identities.
- **Time conditioning.** We apply PepFlow's sinusoidal time embedding separately to each residue's τ_i, replacing the shared time embedding used for the entire complex. Receptor nodes receive the mean peptide time in their complex before embedding. The geometric attention network uses these embeddings together with the perturbed residue states and the fixed node and edge conditioning features.
- **Masked training objectives.** The network predicts clean positions, rotations, torsion angles, and amino-acid logits. We retain six losses: position reconstruction (weight 0.5), rotation vector-field prediction (0.5), backbone-atom reconstruction (0.25), amino-acid cross-entropy (1.0), angular vector-field prediction (1.0), and torsion reconstruction (0.5). Every loss is restricted to peptide residues with $s_i > 0$. Torsion losses additionally use the valid-angle masks associated with the sampled amino-acid predictions. Each loss is normalized within a complex before averaging across the batch, retaining the original time-dependent scaling for rotation and angular vector-field losses.

Training configurations. Table 8 summarizes the DF post-training settings. PepGLAD uses dynamic batches satisfying $\sum_j n_j^2 \leq 30{,}000$, where n_j is the number of peptide and receptor-pocket residues in complex j. PepFlow uses BF16 autocast while retaining FP32 parameters, optimizer states, geometric computations, and loss calculations. The checkpoint epochs indicate the models used in our experiments.

Table 8: DF post-training configurations and checkpoints used for PepGLAD and PepFlow.

Setting	PepGLAD	PepFlow
Optimizer	AdamW	AdamW
Peak learning rate	5×10^{-5}	5×10^{-5}
Weight decay	0.01	0.01
Warmup epochs	5	5
Gradient clipping norm	1.0	1.0
Batch size	Dynamic	64
Training precision	FP32	BF16 mixed precision
Checkpoint epoch used	880	400

D BASELINE IMPLEMENTATIONS

Common settings. Each method has a budget of 110 candidate slots per target. BoN samples all 110 independently. IPR, Beam, FK, and MCTS share 10 independently generated initial DF candidates and allocate the remaining 100 slots to refinement or search. IPR refines these initial candidates; the search methods start from noise and include the initial candidates only in final selection. All candidates enter the evaluation pool, including trial completions used for search decisions. Repeated candidates count toward the budget, and failed slots are not refilled. These four methods use fixed DF checkpoints. Final evaluation follows Appendix E.

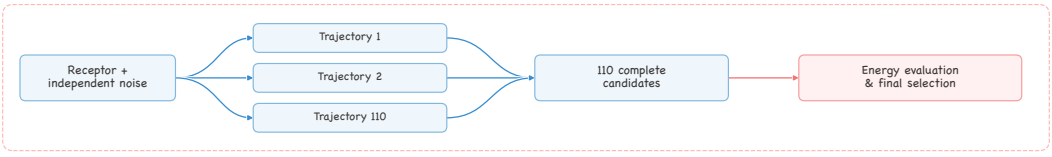


Figure 7: Best-of-N (BoN) pipeline.

Best-of-N Sampling (BoN). BoN independently samples receptor-conditioned peptides using 100 native sampling steps. Table 1 uses released checkpoints for PepBench and PepMerge, and PepGLAD

DF880 for PepBDB. Table 2 compares original and DF checkpoints. Energies guide only final selection.

Iterative Partial Refinement (IPR). IPR (Kang et al., 2026) runs 10 independent trajectories for 10 rounds, producing $10 \times 10 = 100$ additional candidates. Each round uniformly selects $\max(1, \lfloor 0.25L \rfloor)$ residues without replacement, where L is peptide length, resets their states to native priors, and applies 100 sampling steps while fixing the remaining states. PepGLAD fixes sequence and geometric latents before decoding and uses geometric guidance coefficient 0.8. PepFlow edits positions, rotations, torsion angles, and sequence states with its native sampler. Each trajectory continues from its preceding candidate without energy-based acceptance.

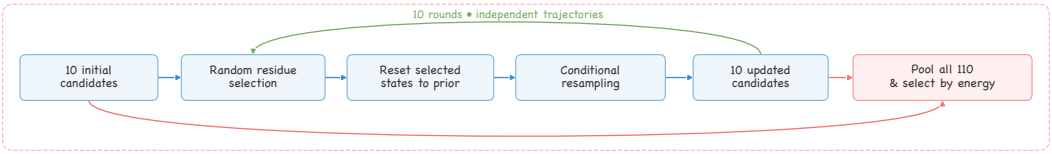


Figure 8: Iterative Partial Refinement (IPR) pipeline.

Beam Search (Beam). Following Complexa (Didi et al., 2026), Beam starts from five noise states. At each of 10 stages, each state produces two branches advancing 10 sampling steps. Each branch is then completed to step 100 for Rosetta scoring, giving $5 \times 2 \times 10 = 100$ trial candidates. The five intermediate states whose completions have the lowest dG continue to the next stage; completed peptides are retained for final evaluation. PepFlow retains the original prior states required for subsequent integration.

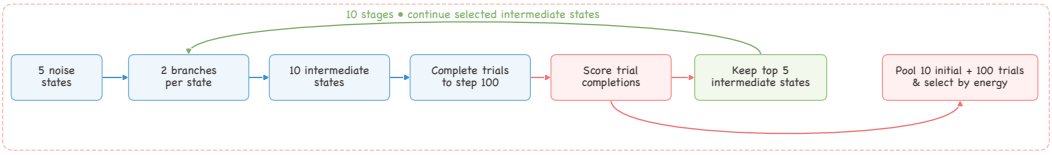


Figure 9: Beam Search (Beam) pipeline.

Feynman–Kac Steering (FK). Following Complexa, FK (Singhal et al., 2025; Didi et al., 2026) maintains five peptide-generation trajectories initialized from independent draws of the backbone's noise prior. The 100 sampling steps are divided into 10 stages. At each stage, each trajectory branches twice, advancing 10 steps per branch. We save the resulting sequence and structure representations, then complete each branch for Rosetta scoring. Five saved representations are sampled with replacement to start the next stage, favoring those whose completions have lower dG. All 10 completions per stage enter the evaluation pool, yielding 100 trial candidates.

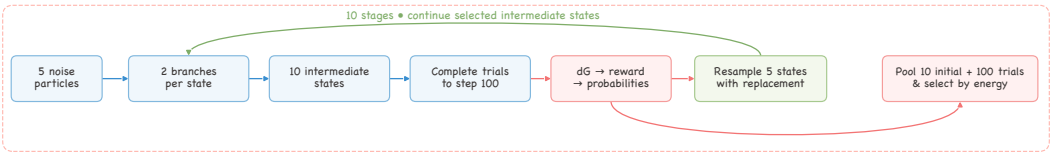


Figure 10: Feynman–Kac Steering (FK) pipeline.

Monte Carlo Tree Search (MCTS). Following Complexa, MCTS (Kocsis & Szepesvári, 2006; Didi et al., 2026) searches a tree whose nodes store intermediate sequence and structure representations and whose branches advance sampling by 10 steps. The initial root is sampled from the backbone's noise prior. Search makes 10 decisions, each preceded by 10 simulations to complete peptides at step 100. Each simulation expands a new branch with probability 0.5 or selects an existing branch by balancing higher average rewards against fewer visits; expansion is forced at childless nodes and step 90. Lower Rosetta dG gives higher reward, which updates the average rewards and visit counts along the simulation path. After 10 simulations, the child with the highest average reward and a valid completion becomes the new root. All completions enter the evaluation pool, using 100 trial slots.

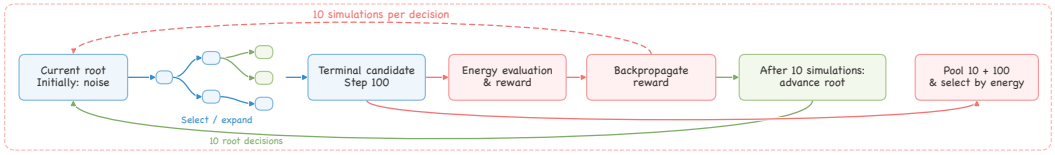


Figure 11: Monte Carlo Tree Search (MCTS) pipeline.

E Evaluation Protocols

- **Mean and Median dG.** We obtain the Rosetta binding score in Equation 2 by applying FastRelax with the ref2015 energy function, followed by InterfaceAnalyzer with separated-chain repacking. For each target, we select the 10 lowest-scoring candidates from the full generation history and rescore their original structures with independent random seeds. We take the lowest valid rescored dG per target, then report its mean and median across targets. These rescoring results do not guide generation or candidate selection. The dG curve in Figure 4 averages each target's lowest search score observed up to the current round.

  We use Rosetta both to guide refinement and to evaluate the resulting designs. Given the trade-off between physical fidelity and computational efficiency, Rosetta remains one of the few practical and widely adopted tools for large-scale structure-based evaluation of biomolecular interactions. Using optimization-aligned energy or property evaluators for both guidance and computational evaluation is also common practice in generative biomolecular design, as exemplified by prior preference-optimization methods such as AbDPO and PepPFDPO (Zhou et al., 2024; Wang et al., 2025).

- **Success Rate.** A selected candidate is successful if its rescored dG is negative. We divide the number of successful candidates by the planned evaluation count, 10 times the number of targets, and report the percentage. Missing or failed evaluations remain in the denominator.

- **Root-Mean-Square Deviation (RMSD).** We compute Cα RMSD between each selected candidate's original, unrelaxed structure and the native peptide in the receptor coordinate frame, without additional peptide superposition. Among the 10 energy-selected candidates, we take the lowest RMSD for each target and report the median across targets. RMSD measures structural recovery rather than our design objective: a conformation closer to the native peptide does not necessarily provide stronger binding or greater complex stability.

- **Amino Acid Recovery (AAR).** Following PepGLAD (Kong et al., 2024), we slide the generated sequence along the native sequence and retain the largest number of matching amino acids, divided by the full native sequence length. Positions outside the overlap count as mismatches. We take the highest AAR among the 10 energy-selected candidates for each target, then average across targets and report a percentage. AAR measures native sequence recovery, whereas our objective is favorable binding; greater sequence similarity alone does not imply stronger or more stable binding.

- **Diversity.** For each target, we follow PepGLAD and cluster sequences and structures separately using single-linkage clustering. Sequence distance is one minus the BLOSUM62 alignment score normalized by the geometric mean of the two self-alignment scores; its clustering threshold is 0.4. Structure distance is Cα RMSD in the receptor frame, with threshold 4 Å. Each diversity is the number of clusters divided by the candidate count. Their geometric mean gives joint sequence–structure diversity, which we average across targets.

  In Table 1, each target contributes 10 candidates: we remove the lowest-energy member of the initial 10 and retain the other nine, then add the lowest-energy candidate from the final refinement round. For BoN, the added candidate is selected from all 110 samples.

  In Figure 4, we instead recluster all candidates generated through round r, including initialization, so the candidate count is $10(r+1)$. We recompute joint diversity at each round and average over all 93 PepBench or 154 PepMerge targets, without energy filtering.

F ADDITIONAL VISUALIZATIONS

Figures 12–14 present additional visualizations on PepBench, PepBDB, and PepMerge.

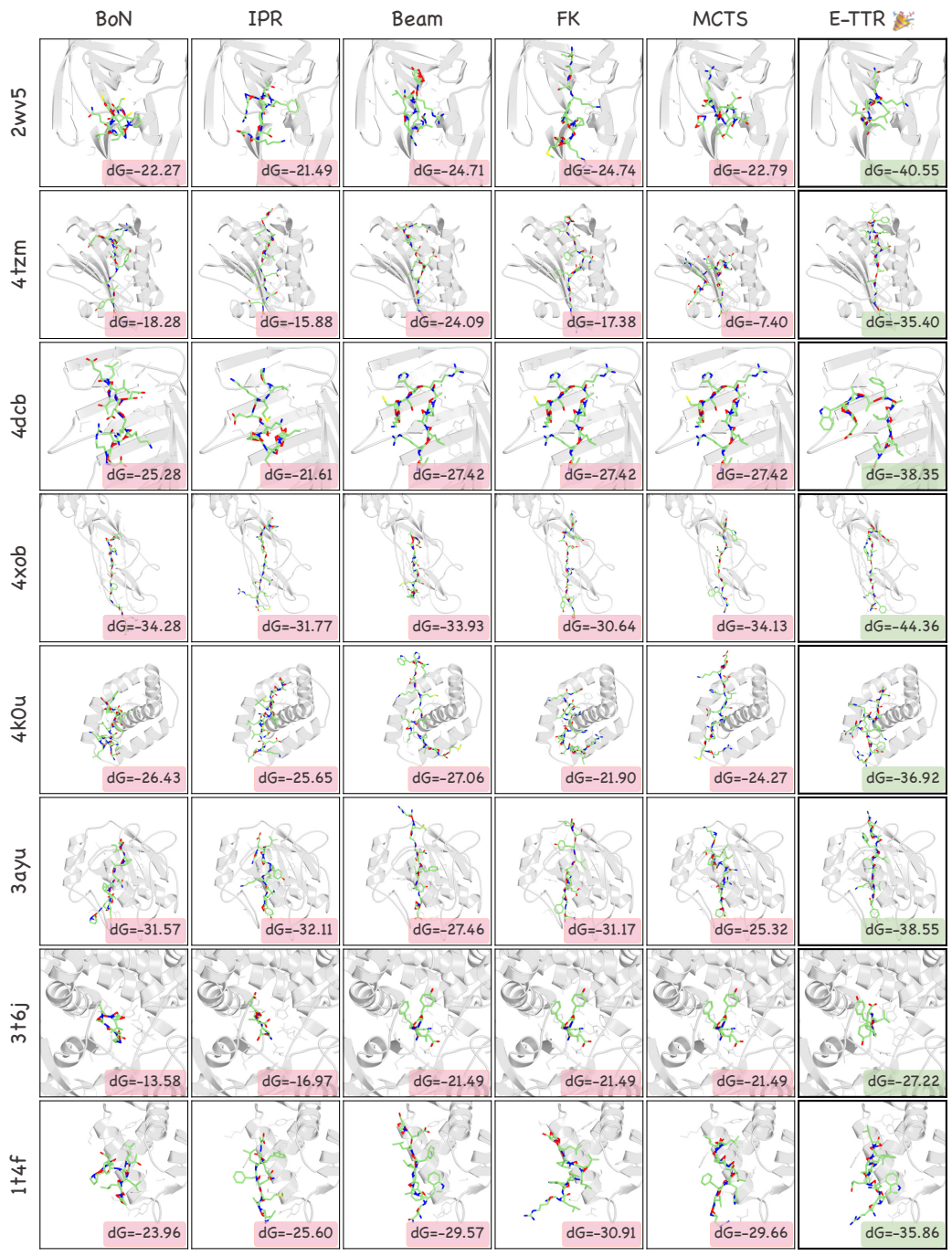


Figure 12: Additional binding-conformation comparisons on PepBench using PepGLAD, with dG annotated in kcal/mol.

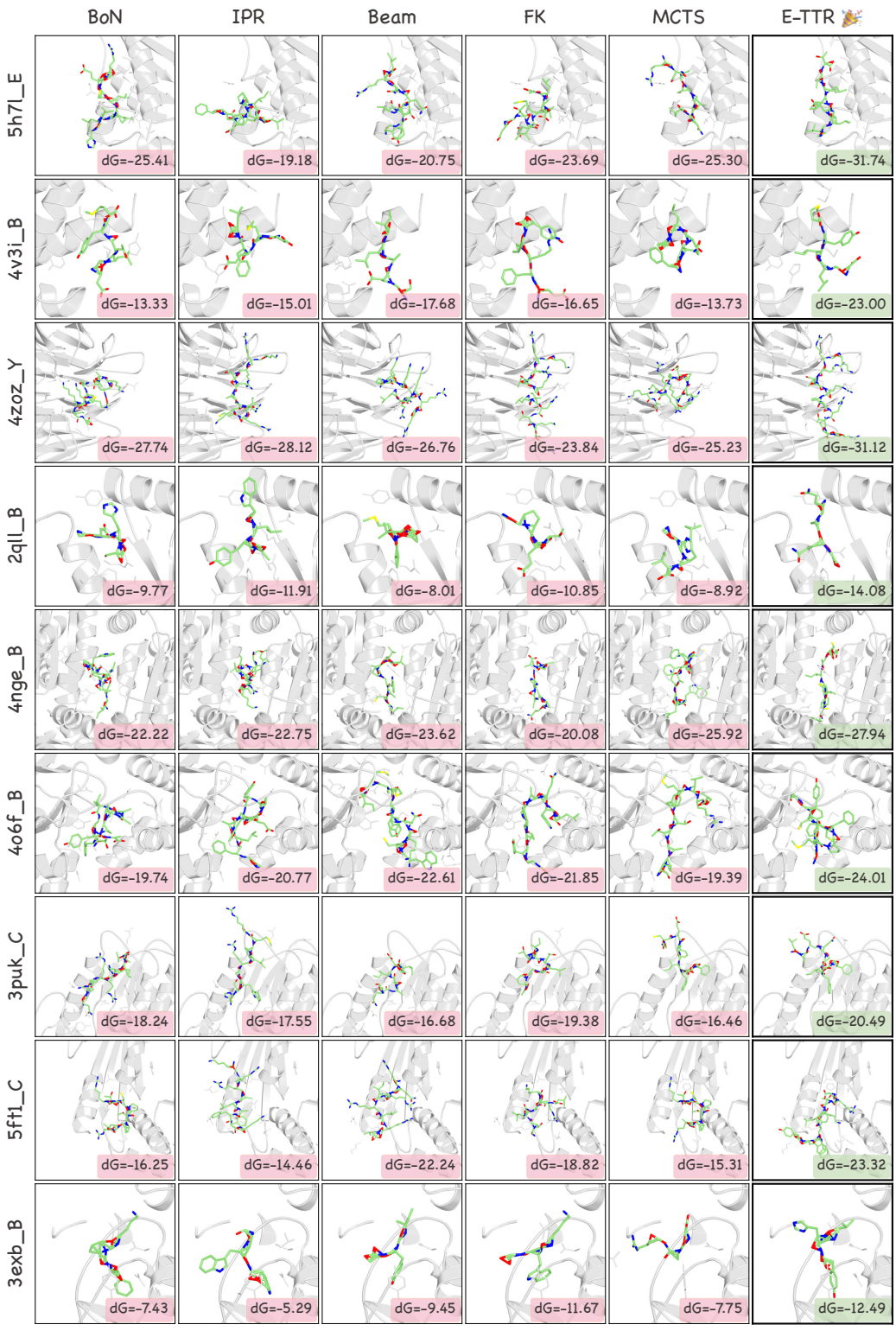


Figure 13: Additional binding-conformation comparisons on PepBDB using PepGLAD, with dG annotated in kcal/mol.

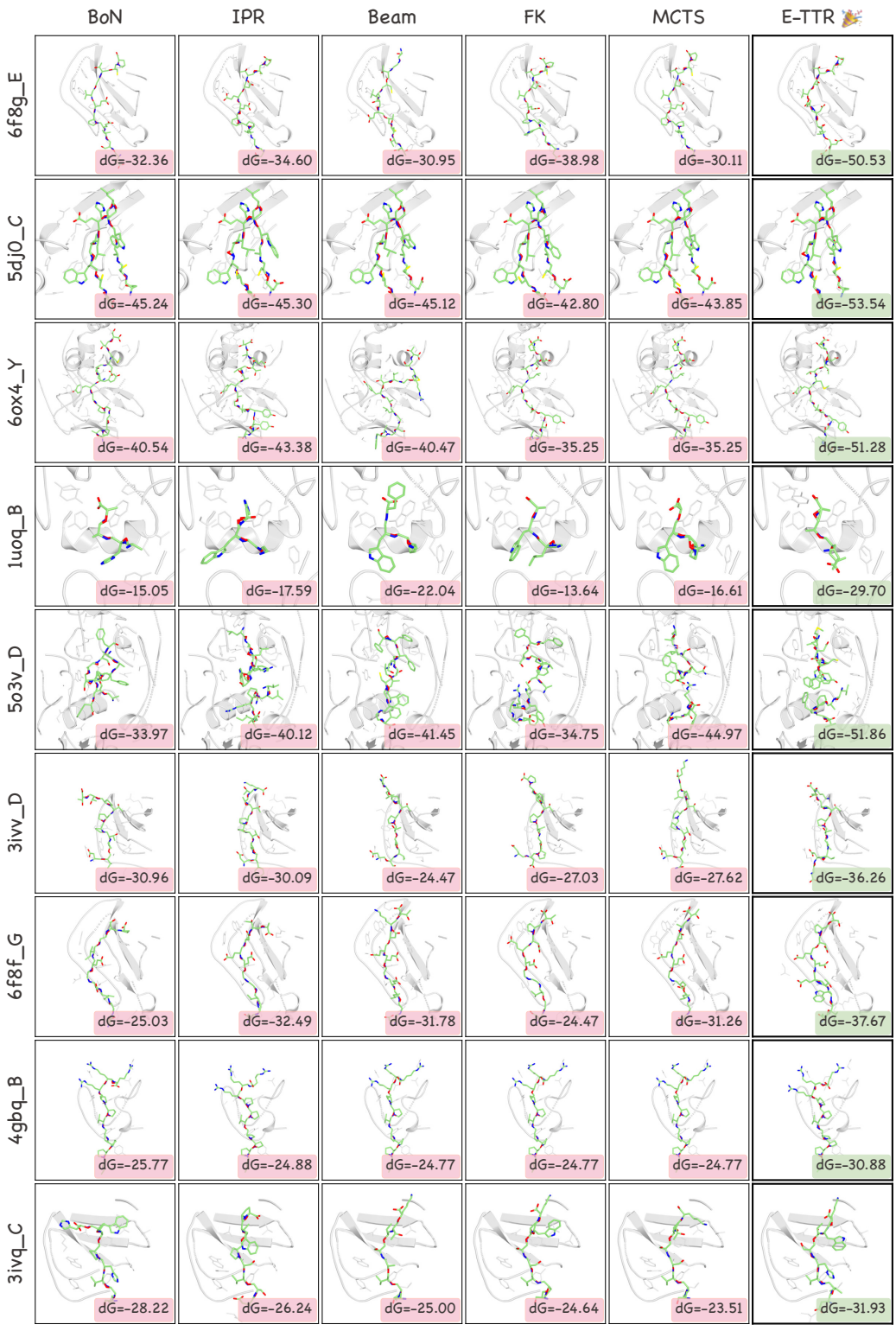


Figure 14: Additional binding-conformation comparisons on PepMerge using PepFlow, with dG annotated in kcal/mol.