ROLLBACK AND REPAIR: REUSING TRAJECTORY ASSETS FOR TEST-TIME SCALING IN PROTEIN DESIGN

Anonymous authors
Paper under double-blind review

ABSTRACT

Reward-guided test-time scaling for protein design aims to improve task-specific quality. However, existing derivative-free approaches typically treat reward feedback as a one-shot selection criterion: they retain only high-scoring candidates and discard low-scoring denoising trajectories. This strategy concentrates computational resources around a few high-reward modes and prematurely eliminates potentially recoverable intermediate states—thereby inducing diversity collapse during test-time optimization. To overcome these issues, we propose **T**rajectory **R**epair with **A**sset **C**onstruction and **E**valuation **R**euse (**TRACER**), a training-free test-time scaling framework that repurposes evaluated denoising trajectories as reusable assets, enabling rollback and targeted repair of suboptimal paths. Specifically, evaluated intermediate states—along with their associated denoising timesteps and value estimates—are organized into time-indexed trajectory assets. Leveraging these assets, TRACER selects underperforming candidates via group-relative advantages, localizes their optimal rollback times with temporal residuals, retrieves high-value references from the time-indexed assets, and repairs the rolled-back intermediate states through reference-guided re-denoising. Experiments on five protein generative models across de novo protein design, protein-binding protein design, and motif scaffolding show that TRACER consistently improves unique success rate while maintaining generative quality.

1 INTRODUCTION

Diffusion models have emerged as powerful generative models, with applications extending beyond image synthesis (Ho et al., 2020; Dhariwal & Nichol, 2021) to natural language generation (Sahoo et al., 2024; Ramesh et al., 2021), molecular generation (Jo et al., 2022; Hoogeboom et al., 2022), and biological sequence generation (Gruver et al., 2023; Zhang et al., 2025). Across these domains, diffusion models are effective at capturing complex natural distributions, making them attractive generative priors for scientific design. In such scientific contexts, generated candidates must not only be physically plausible, but also satisfy task-specific objectives such as stability, binding affinity, designability or structural constraints (Dauparas et al., 2022). These objectives are often encoded as reward or scoring functions. Thus, steering pretrained generative priors toward high-reward regions while preserving naturalness and structural diversity remains a challenging problem in scientific design.

This motivates test-time scaling for reward-guided diffusion generation: additional inference-time computation is used to bias generation toward high-reward regions while preserving the pretrained model's generative prior (Singhal et al., 2025). A classical line builds on classifier guidance, steering each denoising step by adding gradients of a reward function to the pretrained score (Song et al., 2021; Dhariwal & Nichol, 2021; Bansal et al., 2023; Song et al., 2023). In molecular design, however, gradients of the reward are often unavailable since reward feedback may come from structure prediction pipelines, docking or simulation scores (Lee et al., 2023; Castillo-Hair & Seelig, 2021; Salomon-Ferrer et al., 2013; Yu et al., 2023). This has motivated derivative-free methods, including SMC-based guidance (Dou & Song, 2024; Wu et al., 2023) and value-based sampling (Li et al., 2025; Ramesh & Mardani, 2025). The former maintains a population of particles and globally resamples them according to reward-induced importance weights, reallocating computation toward promising trajectories; the latter expands multiple candidate denoising continuations at each step and selects the one with the highest estimated value, turning reward guidance into a value-guided search process.

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.

While effective, derivative-free test-time scaling methods can induce diversity collapse through two mechanisms, as shown in Fig. 1(a). First, reward-weighted selection causes replication-driven collapse: a small number of high-weight samples are repeatedly replicated, progressively shrinking coverage of designable structural modes. Second, low-reward trajectories contain intermediate states in recoverable structural modes. By discarding the entire trajectory, reward-weighted selection removes these states from further exploration, pruning recoverable modes before they can be refined and thereby reducing mode coverage.

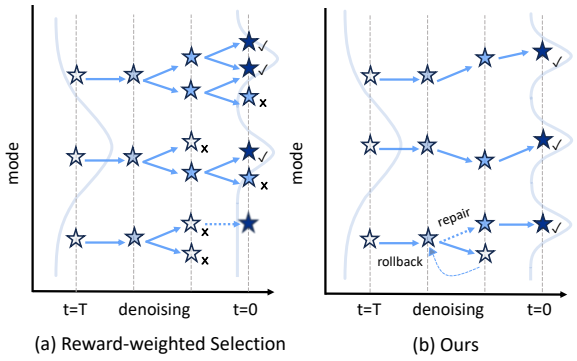

Figure 1: Illustration of different test-time scaling paradigms. Darker nodes indicate higher-reward states.

To address these challenges, we propose a novel test-time scaling framework, **T**rajectory **R**epair with **A**sset **C**onstruction and **E**valuation **R**euse (**TRACER**), which converts evaluated denoising trajectories into reusable assets for rollback and repair, as shown in Fig. 1(b). It adopts a trajectory-as-asset view: instead of treating costly reward evaluations as transient signals for ranking, replication, or pruning, TRACER organizes evaluated denoising states and their value histories into time-indexed assets. This structured repository preserves both the temporal evolution of each candidate and high-value states discovered across candidates, directly addressing the information loss caused by selection discarding. Leveraging these assets, TRACER performs a training-free repair for underperforming candidates: it rolls a deteriorating trajectory back to an intermediate state where its deterioration begins, retrieves high-value references at the same noise level, and re-denoises the candidate back to the original timestep. In this way, test-time compute is used not to replicate winners or discard losers, but to recycle expensive evaluations into reusable guidance for subsequent repair.

Specifically, TRACER first introduces time-indexed trajectory assets, organizing each evaluated intermediate state together with its denoising time and value estimate. The time index makes these assets reuse-ready by enabling states evaluated at the same noise level to be compared and retrieved as repair references. To determine where repair should occur, we propose residual rollback correction, which combines group-relative advantages with centered temporal value residuals to identify underperforming candidates and localize the rollback point where their value trajectory begins to deteriorate. Building upon constructed assets, TRACER then performs reference-guided repair and re-denoising by retrieving locally high-value assets at the same timestep, applying repair toward these references. Through this design, TRACER enables evaluation feedback to be reused as structured trajectory assets rather than consumed as a one-shot selection criterion.

Our contributions are summarized as follows: 1) We propose TRACER, a training-free test-time scaling framework that reuses costly reward evaluations as time-indexed trajectory assets, allowing reward feedback to be reused for rollback and repair rather than consumed as a one-shot selection criterion. 2) We introduce a rollback-and-repair strategy, which identifies repair candidates with group-relative advantages, localizes deterioration points via temporal residuals, and repairs rolled-back states with same-timestep high-value references from trajectory assets, improving task-specific design quality while mitigating diversity collapse. 3) We evaluate TRACER on five protein generative models and three design tasks, demonstrating consistent improvements across task-specific quality and diversity metrics and favorable scaling with increased inference compute.

2 Related Work

2.1 Generative Models for Protein Design

Protein generation is an important problem in molecular design, where the goal is to produce diverse, physically plausible structures that also satisfy task-specific functional constraints. Recent protein generators can be broadly grouped into diffusion-based and flow-based paradigms. Diffusion-based models such as RFdiffusion (Watson et al., 2023), Chroma (Ingraham et al., 2023) and RFdiffusion-All-Atom (Krishna et al., 2024) learn iterative denoising priors for backbone structures. Protpardelle (Chu et al., 2024; Lu et al., 2025) extends this to all-atom protein generation. Flow-

based models such as Proteina (Geffner et al., 2025) and APM (Chen et al., 2025) learn continuous generative dynamics for scalable backbone generation and all-atom complex design. More recent interaction-aware systems, including ODesign (Zhang et al., 2025) and Proteina-Complexa (Didi et al., 2026), further push these priors toward target-conditioned interaction design and atomistic binder generation. These advances establish strong generative priors (Wu et al., 2024; Yim et al., 2023; Jendrusch & Korbel, 2025), but they primarily improve the generator itself; our focus is orthogonal, treating such models as fixed backbones and asking how evaluated intermediate states can be reused during inference. Recent systems, most notably Proteina-Complexa, further combine trained protein generators with test-time compute. However, they do not address how to effectively exploit costly task-specific reward at inference time without sacrificing diversity.

2.2 Test-time Scaling

A complementary line of work studies test-time scaling of pretrained diffusion models (Song et al., 2020; Karras et al., 2022). Most classical approaches involve classifier guidance (Song et al., 2021; Dhariwal & Nichol, 2021; Bansal et al., 2023). Reward gradients are used to tilt the diffusion model's score, $s_\theta(x_t, t) + \nabla_{x_t} v(x_t)$, where s_θ is the marginal score and $v(x_t)$ is a value function that estimates future terminal reward. However, gradient-based guidance is limited to differentiable rewards, which is challenging due to the non-differentiable nature of reward feedback in molecular design (Uehara et al., 2024). The simplest derivative-free strategy is Best-of-N sampling, which allocates additional test-time compute to generating multiple independent candidates and selecting those with the highest terminal rewards. Recent derivative-free methods exploit additional test-time compute through sampling, search, and resampling (Dou & Song, 2024; Wu et al., 2023; Phillips et al., 2024; Cardoso et al., 2024; Uehara et al., 2025). SMC-based guidance methods usually propagate multiple trajectories and reweight or resample them using reward-derived weights or potentials, as in Feynman–Kac steering (FKS) (Singhal et al., 2025; Del Moral et al., 2006). Value-based sampling approaches further treat denoising as beam search (Fernandes et al., 2025; Li et al., 2025) or a tree-structured decision process and allocate extra computation to promising branches or trajectories, for example through Monte Carlo Tree Search (MCTS) and related test-time search procedures (Yoon et al., 2025; Ramesh & Mardani, 2025). Zhang et al. (2026) further combine tree-based global search with verifier-guided local search via annealed Langevin dynamics. TRACER instead treats evaluated trajectories as reusable assets, turning reward feedback from a signal for search allocation into guidance for cross-trajectory repair.

3 Method

We propose a training-free test-time scaling framework for protein design with pretrained diffusion models. Existing derivative-free methods such as SMC-based guidance and beam search typically rank candidate trajectories by reward, allocate computation to high-reward trajectories, and discard or downweight low-reward ones. In this sense, the reward acts as a one-shot selection criterion, consumed once to decide which trajectories survive. In contrast, we convert reward evaluations into reusable trajectory assets by combining intermediate states together with their estimated values. The resulting procedure replaces selection and discard test-time scaling with a repair process: underperforming candidates are rolled back to the point where deterioration begins, repaired using high-value references, and then re-denoised through the reverse process. An overview is provided in Fig. 2.

3.1 Turning Evaluation into Assets

We consider reward-guided test-time scaling for a pretrained diffusion model p_θ over a design space $\mathcal{X}$, conditioned on optional context c. Let x_t denote the diffusion state at time $t \in \{0, \dots, T\}$, where x_T is the initial noisy state and $x_0 \in \mathcal{X}$ is the terminal clean design. The pretrained model induces the reverse denoising path:

$$p_\theta(x_{0:T} \mid c) = p_T(x_T) \prod_{t=T}^{1} p_\theta(x_{t-1} \mid x_t, c), \tag{1}$$

where p_θ denotes the pretrained one-step denoising transition.

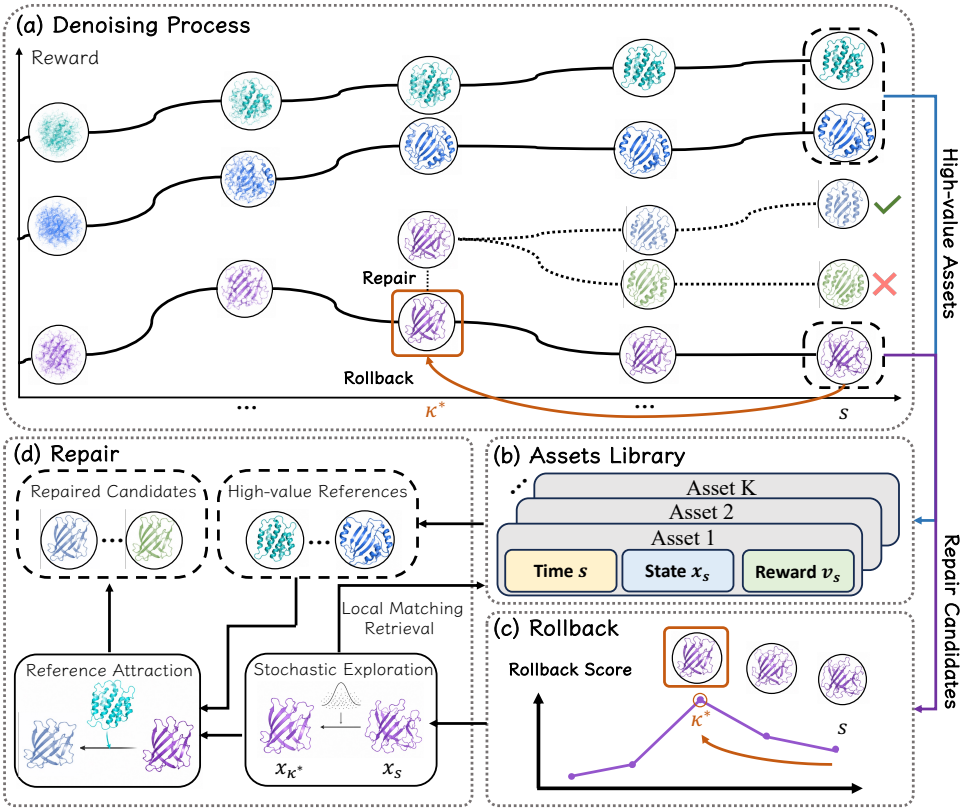

Figure 2: Overview of TRACER. (a) The denoising process of the generative model. (b) The assets library organizes evaluated states, timesteps, and rewards as time-indexed trajectory assets. (c) Residual rollback identifies underperforming candidates via group-relative advantages and localizes rollback points using cumulative degradation from temporal residuals. (d) Reference-guided repair retrieves high-value assets at the same timestep, repairs the rolled-back state through stochastic exploration and reference attraction, and retains the best improved proposal after re-denoising.

Given a terminal reward $r(x_0, c)$, reward-guided diffusion aims to bias the clean-design marginal toward the exponentially tilted distribution:

$$\pi_\lambda(x_0 \mid c) \propto p_\theta(x_0 \mid c) \exp\{\lambda r(x_0, c)\}. \tag{2}$$

Ideally, the corresponding intermediate reward would summarize the terminal rewards of clean designs obtained by continuing the reverse process from x_t:

$$V_t^\lambda(x_t, c) = \frac{1}{\lambda} \log \mathbb{E}_{x_{0:t-1} \sim \prod_{\tau=t}^{1} p_\theta(x_{\tau-1} \mid x_\tau, c)} \left[\exp\{\lambda r(x_0, c)\} \right]. \tag{3}$$

Because this conditional expectation is generally intractable, we use a terminal-rollout score for intermediate-state evaluation. Specifically, starting from x_t, we run the pretrained reverse process to completion, obtaining a terminal design $\tilde{x}_0 \sim p_\theta(x_0 \mid x_t, c)$. We then define $\widehat{V}_t(x_t, c) = r(\tilde{x}_0, c)$, which serves as a practical surrogate for V_t^λ.

Since adjacent states x_t and x_{t+1} are highly correlated and evaluating the reward proxy at every denoising step is often computationally redundant, we perform evaluation only at selected denoising steps, specified by an evaluation schedule $\mathcal{S} = \{s_0, s_1, \ldots, s_B\}$, where $T \geq s_0 > s_1 > \cdots > s_B \geq 0$. At each scheduled evaluation time $s \in S$, we evaluate all candidates at the same noise level by $v_s^i = \widehat{V}_s(x_s^i, c)$, and convert the resulting states into reusable trajectory assets. Formally, each asset is defined as a triplet:

$$a_s^i = \left(s,\, x_s^i,\, v_s^i\right), \qquad \mathcal{A}_s(c) = \{a_s^i\}_{i=1}^{|\mathcal{A}_s|}. \tag{4}$$

The triplet assets support two complementary views. First, along the trajectory of a single candidate, the value history allows us to identify where candidate i's value begins to deteriorate along the

denoising path. Second, across candidates, high-value states evaluated at the same denoising step serve as repair references for underperforming candidates. Consequently, rather than being used once to rank candidates and then discarded, the reward signal is preserved alongside its associated intermediate states, forming structured trajectory assets that enable future repair operations.

3.2 Residual Rollback

At a current scheduled evaluation time $s \in S$, all candidates are evaluated under the same context c, so their values are directly comparable. We define the group-relative advantage of candidate i: $A_s^i = v_s^i - \bar{v}_s$, where $\bar{v}_s = \frac{1}{N}\sum_{j=1}^{N} v_s^j$. Candidates whose values do not exceed the current group mean are selected for repair: $\mathcal{I}_s = \{i : A_s^i \leq 0\}$. The complementary high-value candidates are defined as $\mathcal{H}_s = \{i : A_s^i > 0\}$ for later repair. For each candidate $i \in \mathcal{I}_s$, we use its history assets to measure how its value evolves across scheduled evaluation times. For two consecutive evaluation times s_ℓ and $s_{\ell+1}$, we define the temporal residual:

$$\delta^i_{\ell\to\ell+1} = v^i_{s_{\ell+1}} - \gamma^{s_\ell - s_{\ell+1}} v^i_{s_\ell}, \qquad \ell = 0, \ldots, b-1, \tag{5}$$

where $\gamma \in (0, 1]$ discounts value across denoising progress.

To remove transition-level shifts shared by all candidates, we center the residuals within each transition: $z^i_\ell = \delta^i_{\ell\to\ell+1} - \bar{\delta}_{\ell\to\ell+1}$, where $\bar{\delta}_{\ell\to\ell+1} = \frac{1}{N}\sum_{j=1}^{N} \delta^j_{\ell\to\ell+1}$. A negative centered residual indicates that candidate i's value evolution is worse than the group-average transition. We therefore define its deterioration mass over this transition as $d^i_{s_\ell\to s_{\ell+1}} = \max\{-z^i_\ell, 0\}$. Then, for a given rollback time κ, we compute the cumulative downstream degradation from κ to the current evaluation time s as: $C_i(\kappa; s) = \sum_{u\in S:\, \kappa\geq u>s} d^i_u$. Large $C_i(\kappa; s)$ indicates that substantial negative residual mass accumulates after κ, suggesting that rolling candidate i back to κ can explain and potentially repair the observed degradation. To avoid unnecessarily long rollback, we define a rollback score as the cumulative downstream degradation minus a rollback-gap penalty, and select the maximizing rollback point:

$$\kappa^\star_i = \arg\max_{\kappa\in\mathcal{S}} \left[C_i(\kappa; s) - \omega \frac{\kappa - s}{H_{\max}} \right], \tag{6}$$

where $\omega \in (0, 1]$ controls the rollback-gap penalty, and $H_{\max}$ denotes the maximum rollback horizon. This criterion selects an intervention time that accounts for downstream degradation while preventing excessive re-denoising.

3.3 Reference-guided Repair and Re-denoising

Residual rollback determines where to intervene, while trajectory assets determine how to intervene. For each candidate $i \in \mathcal{I}_s$, we roll the current state x^i_s back to $\kappa^\star_i$ by the diffusion forward noising transition and the p-th stochastic rollback state is defined as:

$$x^{i,p}_{\kappa^\star_i} = x^i_s + (1 - M_i) \odot \eta \sqrt{\sigma^2_{\kappa^\star_i} - \sigma^2_s}\, \epsilon^{i,p}, \qquad \epsilon^{i,p} \sim \mathcal{N}(0, I), \tag{7}$$

where σ_t denotes the noise level associated with denoising step t, η controls the amount of re-noising, and M_i masks conditional atoms.

For each rollback state $x^{i,p}_{\kappa^\star_i}$, we retrieve a high-value reference $y^{i,p}_{\kappa^\star_i}$ from the high-value assets $\mathcal{H}_{\kappa^\star_i}(c)$ at the same timestep. The reference is selected according to a local compatibility distance over the design region Ω_i:

$$d_{\Omega_i}(x, y) = \frac{1}{|\Omega_i|} \sum_{a\in\Omega_i} \left\| (x_a - \bar{x}_{\Omega_i}) - (y_a - \bar{y}_{\Omega_i}) \right\|_2^2, \tag{8}$$

where $\bar{x}_{\Omega_i}$ and $\bar{y}_{\Omega_i}$ are the centroids of the design region.

Given the retrieved reference $y^{i,p}_{\kappa^\star_i}$, we construct the repaired rollback proposal by applying a masked residual attraction toward the reference:

$$\widetilde{x}^{i,p}_{\kappa^\star_i} = x^{i,p}_{\kappa^\star_i} + (1 - M_i) \odot \alpha \left(y^{i,p}_{\kappa^\star_i} - x^{i,p}_{\kappa^\star_i} \right). \tag{9}$$

Here, α controls the strength of attraction toward the retrieved high-value reference. The repaired intermediate state is then passed through the pretrained denoising kernel: $\widetilde{x}_s^{i,p} \sim p_\theta^{\kappa_i^\star \to s}\left(\cdot \mid \widetilde{x}_{\kappa_i^\star}^{i,p}, c\right)$. Thus, each repair candidate i generates P re-denoised proposals $\{\widetilde{x}_s^{i,p}\}_{p=1}^{P}$, where the p-th proposal is repaired by the p-th retrieved high-value reference. We select the proposal with the highest value and accept it only if it improves the current candidate value. If accepted, the re-denoised proposal becomes the updated state of candidate i at evaluation time s, and its state-value pair is recorded as a time-s trajectory asset. Consequently, reference-guided repair exploits high-value local structure while preserving the pretrained diffusion prior through re-denoising. We further show this induces a bounded value drift after re-denoising; a detailed analysis is provided in Appendix A.

4 Experiments

4.1 Experimental Setup

Generation Tasks and Benchmark We evaluate our framework on three important protein design tasks: *de novo* protein design, protein-binding protein design and motif scaffolding. 1) *De novo* protein design requires models to generate protein backbones without any prior conditioning, reflecting the model's explorable protein design space. 2) Protein-binding protein design involves generating protein binders against specified protein targets. We perform this task on 10 well-known targets from CaoData (Cao et al., 2022), a dataset derived from classical *de novo* protein design campaigns based on target structures. 3) Motif scaffolding requires designing diverse protein structures to accommodate given motifs—geometric coordinates of residues associated with biochemical function—while preserving their spatial arrangements. We evaluate this task on MotifBench (Zheng et al., 2025), a classical benchmark for quantifying model performance on motif scaffolding tasks.

Generative Models We validate our method on representative recent protein generative models spanning two major trajectory-based paradigms of continuous-time structure generation. The first group consists of diffusion-based models, which generate structures through iterative denoising of noisy states, including ODesign (Zhang et al., 2025) and Protpardelle-1c (Lu et al., 2025). The second group consists of flow-based models, including APM (Chen et al., 2025), Proteina (Geffner et al., 2025), and Proteina-Complexa (Didi et al., 2026). Evaluating across both diffusion and flow-based generators provides a systematic assessment of the generality of our framework across heterogeneous sampling dynamics and design settings.

Compared Methods We compare TRACER with the following state-of-the-art (SOTA) methods: Monte Carlo Tree Search (MCTS) (Yoon et al., 2025; Ramesh & Mardani, 2025) and Feynman–Kac Steering (FKS) (Singhal et al., 2025). MCTS is a value-based sampling method and treats the iterative generation process of flow and diffusion models as a tree, where different paths in the tree correspond to different stochastic denoising trajectories, and then searches within that tree. FKS is an SMC-based guidance method that generates multiple parallel sample trajectories from flow and diffusion models and iteratively resamples these particles based on their computed scores.

Implementation Details Generally, we set temporal discount $\gamma = 0.96$ and rollback penalty $\omega = 0.4$, attraction strength $\alpha = 0.02$ and stochastic exploration $\eta = 0.7$. The largest rollback gap H_{max} is set to 25% of total denoising steps according to the model. We retrieve the top $P = 4$ assets for repair-eligible candidates. For *de novo* protein design, we generate 200 proteins for each model. In protein-binding protein design and motif scaffolding, we generate 100 proteins for each target. All experiments are conducted on a single NVIDIA A800 GPU.

Evaluation Metrics For *de novo* protein design, we evaluate structural quality using pLDDT and Cα RMSD. For protein–protein binder design, we use pLDDT as a structural confidence metric and minimum iPAE as an interface-quality metric. For motif scaffolding, we follow MotifBench and report Novelty and Motif Score. Across all tasks, we further report the unique success rate (USR), following Didi et al. (2026). USR is defined as the number of structurally distinct successful clusters divided by the total number of generated samples, and is reported as a percentage. Success criteria follow those used in the corresponding original benchmark papers. Detailed definitions are provided in Appendix E.

Table 1: Results of *de novo* protein design.

Methods	pLDDT↑	Cα RMSD↓	USR↑
APM	81.15	3.58	17.5
+ MCTS	81.69	2.74	15.0
+ FKS	81.16	2.84	7.5
+ **TRACER**	**81.92**	**2.60**	**19.5**
Proteina	90.09	1.89	9.5
+ MCTS	90.80	1.81	3.0
+ FKS	90.24	1.83	2.5
+ **TRACER**	**90.94**	**1.78**	**11.0**
ODesign	86.68	1.74	26.5
+ MCTS	87.47	1.62	17.0
+ FKS	87.25	1.64	16.5
+ **TRACER**	**87.78**	**1.59**	**32.0**

Table 2: Results of protein-binding protein design.

Methods	IL7RA pLDDT↑	IL7RA min iPAE↓	IL7RA USR↑	PDL1 pLDDT↑	PDL1 min iPAE↓	PDL1 USR↑
APM	83.94	10.46	6	78.00	9.28	8
+ MCTS	84.63	10.06	4	81.94	4.33	8
+ FKS	81.74	10.36	2	80.72	7.71	3
+ **TRACER**	**85.70**	**9.60**	**7**	**81.98**	**4.27**	**10**
Proteina-C	86.77	9.73	19	86.77	3.96	1
+ MCTS	90.98	1.34	40	88.75	2.77	1
+ FKS	87.11	4.83	20	86.65	3.84	0
+ **TRACER**	**91.00**	**1.32**	**50**	**88.85**	**2.73**	**2**
ODesign	86.64	5.65	26	83.41	9.32	1
+ MCTS	84.61	7.24	5	83.44	8.81	1
+ FKS	86.47	5.53	5	82.95	10.26	0
+ **TRACER**	**86.74**	**5.45**	**48**	**84.30**	**7.55**	**2**

Table 3: Results of motif scaffolding.

Methods	09_7CG5 Novelty ↑	09_7CG5 Motif Score ↑	09_7CG5 USR ↑	07_6E6R Novelty ↑	07_6E6R Motif Score ↑	07_6E6R USR ↑
Protpardelle-1c	0.251	30.00	2	0.199	52.50	5
+ MCTS	0.748	17.50	1	0.688	30.00	2
+ FKS	0.000	0.00	0	0.719	46.67	4
+ **TRACER**	**0.756**	**39.38**	**3**	**0.726**	**64.62**	**8**
ODesign	0.675	39.38	3	0.678	46.67	4
+ MCTS	0.687	52.50	5	0.655	57.27	6
+ FKS	0.552	39.38	3	0.652	17.50	1
+ **TRACER**	**0.698**	**57.27**	**6**	**0.698**	**61.25**	**7**

4.2 Main Results

We compare TRACER with previous test-time scaling methods on three important tasks in the field of protein design, as shown in Table 1–Table 3. We denote Proteina-Complexa as Proteina-C for brevity and additional targets are reported in Appendix B. The results lead to three key observations: 1) Existing test-time scaling baselines struggle to preserve diversity. Although MCTS and FKS can improve structural quality metrics, they consistently reduce USR on most tasks, indicating that reward-weighted selection concentrates samples around few modes and causes diversity collapse. 2) TRACER consistently outperforms baselines across tasks and backbones. It improves performance across metrics and evaluated settings, with particularly consistent gains in successful design recovery. 3) TRACER is especially effective in low-success settings. In cases where the source generator yields few unique successful structures, selection-based baselines show unstable USR gains: MCTS improves USR only in some settings, whereas FKS often reduces USR and may even collapse it to zero. In contrast, TRACER repairs underperforming trajectories rather than discarding them, allowing recoverable intermediate states to be rolled back and re-denoised into additional unique successful structures.

4.3 Effectiveness of Components

We investigate the impact of individual design choices by comparing TRACER with partial variants, as shown in Table 4. Here, *TD* denotes temporal discounting in residual rollback, *RGP* denotes the rollback-gap penalty, *HVR* denotes high-value reference retrieval, *RA* denotes reference attraction in reference-guided repair and *ARF* denotes the hard accept-reject filter. The corresponding variants are implemented by setting $\gamma = 1$, setting $\omega = 0$, replacing high-value retrieval with random retrieval, setting $\alpha = 0$ and removing the hard accept-reject filter, respectively. Our analysis yields several key observations: 1) Removing TD yields only marginal gains over the ODesign baseline and worsens Cα RMSD. This indicates that temporal discounting is important for comparing value changes across denoising intervals; without it, residual-based rollback localization becomes less reliable, causing repair to be applied at suboptimal steps and inducing structural drift.

2) Removing RGP improves pLDDT and USR only moderately but degrades Cα RMSD, suggesting that rollback becomes overly aggressive without an explicit penalty on rollback depth. The absence of RGP is consistent with longer re-denoising horizons and stronger noise-matched perturbations, thereby weakening geometric consistency. 3) Replacing HVR with random reference retrieval consistently weakens TRACER. This suggests high-value references provide more reliable structural cues and are more likely to correspond to repair directions, whereas random references may introduce less coherent guidance. 4) Removing RA reduces all metrics compared with TRACER, demonstrating that high-value assets provide useful references for constructing effective repair directions. 5) The variant without ARF still improves pLDDT and USR over the source model, suggesting that reference-guided repair can generate useful repair signals even without explicit filtering.

Table 4: Impact of component variations.

Method	pLDDT ↑	Cα RMSD ↓	USR ↑
ODesign	86.68	1.74	26.5
TRACER	**87.78**	**1.59**	**32.0**
w/o TD	86.73	1.84	28.5
w/o RGP	86.92	1.80	29.0
w/o HVR	86.81	1.72	27.5
w/o RA	87.39	1.63	30.5
w/o ARF	87.19	1.75	28.0

Table 5: Empirical rollback distribution and repair outcomes.

RRD	SR	AR ↑	RI ↑
1.00	**47.01**%	94.12%	+0.39
0.67	34.10%	**97.30**%	**+0.43**
0.33	18.89%	87.80%	+0.34

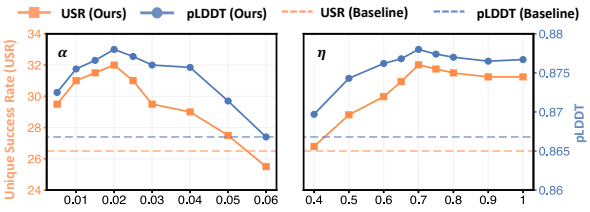

Figure 3: Sensitivity analysis of reference-guided repair on the ODesign *de novo* protein design task.

4.4 Rollback and Repair Mechanism Analysis

To better understand how our rollback-and-repair mechanism operates, we analyze repair behavior across relative rollback depths (RRDs), normalized by the maximum evaluation window, where RRD=1.00 denotes the deepest rollback. Table 5 reports the selection ratio (SR), acceptance rate (AR), and reward improvement (RI). The results yield two main observations: 1) Repairs remain effective across all depths, with high AR and positive RI, indicating that TRACER benefits from multiple recovery horizons. 2) Repair effectiveness is non-monotonic with rollback depth: despite deeper rollbacks being selected more frequently, RRD=0.67 achieves the highest AR and RI. This supports our gap-regularized rollback design in Eq. (6), which balances recovery horizon against excessive rewinding.

4.5 Sensitivity Analysis of Reference-Guided Repair

We further evaluate the robustness of TRACER to the two key coefficients in reference-guided repair: the reference attraction strength α and the stochastic exploration coefficient η, as shown in Fig. 3. This yields three key observations: 1) TRACER outperforms the ODesign baseline under most settings, suggesting that the rollback and repair mechanism is not sensitive to a narrow hyperparameter range. 2) Varying α shows that reference guidance should be kept moderate. A very small α provides weak repair, while $\alpha = 0.02$ gives the best performance. Further increasing α degrades performance, because pulling the rolled-back state too far along the reference direction can make it less compatible with the intermediate-state distribution expected by the pretrained denoising dynamics. 3) Varying η shows a trade-off between stochastic exploration and re-denoising stability. The scale $\sigma_{\kappa_i^\star,s}$ reflects the noise gap between the rollback checkpoint and the current evaluation step, while η rescales the injected perturbation. A small η provides limited stochastic exploration around the reference-guided repair direction, whereas an overly large η injects excessive noise and makes the subsequent re-denoising less stable. Together, these results show that both the reference attraction strength and the injected noise scale should be kept moderate. We therefore set $\alpha = 0.02$ and $\eta = 0.7$ in all experiments.

4.6 Compute Scaling and Efficiency Analysis

In Section 4.2, experiments were conducted under a fixed final generation budget. Here we further evaluate the computational efficiency of TRACER under the same inference setting on the ODesign *de novo* protein design task, as shown in Fig. 4. In addition to structural quality metrics, we report the cumulative number of unique successful structures obtained under different GPU-time budgets, which measures how effectively test-time computation is converted into diverse successful designs. The trajectory reuse incurs negligible memory overhead: TRACER stores at most 50 intermediate structures and their value estimates, requiring only 4.48 MB (0.11% of the pre-trained model size) under the default ODesign setting. Notably, unguided ODesign performs better under small compute budgets but quickly saturates as inference compute increases. In contrast, TRACER exhibits substantially stronger scaling: its unique successful designs increase as the budget grows, eventually surpassing the baseline, while consistently outperforming MCTS and FKS. This scaling behavior supports our hypothesis that reusing evaluated trajectory assets for targeted repair becomes increasingly effective as additional inference compute becomes available. Additional compute-scaling results for protein-binding protein design and motif scaffolding are provided in Appendix B.2.

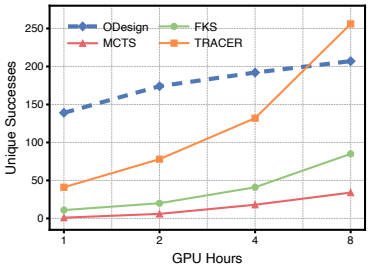

Figure 4: Inference-time scaling on ODesign *de novo* protein design.

4.7 Qualitative Results

Figure 5 qualitatively demonstrates TRACER's improvements across three protein design tasks. For *de novo* design, TRACER produces a more regular structure, increasing pLDDT from 73.07 to 95.26. For binder design, it improves target–binder compatibility, reducing minimum iPAE from 10.84 to 4.79. For motif scaffolding, it better recovers the prescribed motif geometry, reducing motif Cα RMSD from 2.26 to 0.95. These examples complement our quantitative results, showing that TRACER improves distinct task-specific design objectives through trajectory repair.

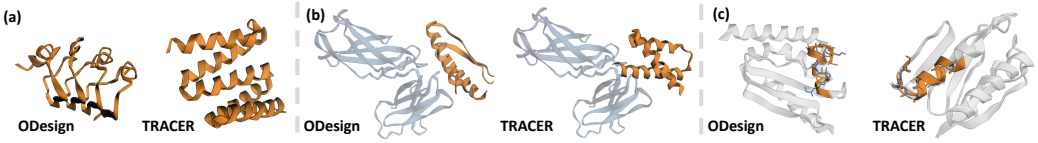

Figure 5: Qualitative results on ODesign. (a) *De novo* design. (b) Binder design, with the target in orange and generated backbone in gray. (c) Motif scaffolding, with the reference motif in blue, recovered motif in orange, and generated scaffold in gray.

5 Conclusion

In this paper, we propose TRACER, a novel training-free test-time scaling framework for protein design that transforms evaluated denoising trajectories into reusable trajectory assets. Instead of treating reward feedback as a one-shot selection criterion, TRACER organizes intermediate states, denoising timesteps, and value estimates into time-indexed assets, enabling underperforming trajectories to be rolled back, matched with high-value references at the same timestep, and repaired through reference-guided re-denoising. This rollback-and-repair mechanism provides a new way to reuse expensive test-time evaluations, improving task-specific optimization while mitigating the diversity collapse commonly caused by selection-based scaling methods. We demonstrate consistent improvements across five protein generative models and three representative design tasks, including de novo generation, protein-binding protein design, and motif scaffolding. Multi-seed and compute-scaling experiments further show that these gains are robust to stochastic variation and persist as additional inference compute becomes available.

REFERENCES

Josh Abramson, Jonas Adler, Jack Dunger, Richard Evans, Tim Green, Alexander Pritzel, Olaf Ronneberger, Lindsay Willmore, Andrew J Ballard, Joshua Bambrick, et al. Accurate structure prediction of biomolecular interactions with alphafold 3. *Nature*, 630(8016):493–500, 2024.

Arpit Bansal, Hong-Min Chu, Avi Schwarzschild, Soumyadip Sengupta, Micah Goldblum, Jonas Geiping, and Tom Goldstein. Universal guidance for diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 843–852, 2023.

Longxing Cao, Brian Coventry, Inna Goreshnik, Buwei Huang, William Sheffler, Joon Sung Park, Kevin M Jude, Iva Marković, Rameshwar U Kadam, Koen HG Verschueren, et al. Design of protein-binding proteins from the target structure alone. *Nature*, 605(7910):551–560, 2022.

Gabriel Victorino Cardoso, Yazid Janati El Idrissi, Sylvain Le Corff, and Eric Moulines. Monte carlo guided diffusion for bayesian linear inverse problems. In *ICLR International Conference on Learning Representations*, 2024.

Sebastian M Castillo-Hair and Georg Seelig. Machine learning for designing next-generation mrna therapeutics. *Accounts of chemical research*, 55(1):24–34, 2021.

Ruizhe Chen, Dongyu Xue, Xiangxin Zhou, Zaixiang Zheng, xiangxiang Zeng, and Quanquan Gu. An all-atom generative model for designing protein complexes. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=Afmi28vgIf.

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.

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.

Pierre Del Moral, Arnaud Doucet, and Ajay Jasra. Sequential monte carlo samplers. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 68(3):411–436, 2006.

Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021.

Kieran Didi, Zuobai Zhang, Guoqing Zhou, Danny Reidenbach, Zhonglin Cao, Sooyoung Cha, Tomas Geffner, Christian Dallago, Jian Tang, Michael M. Bronstein, Martin Steinegger, Emine Kucukbenli, Arash Vahdat, and Karsten Kreis. Scaling atomistic protein binder design with generative pretraining and test-time compute. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=qmCpJtFZra.

Zehao Dou and Yang Song. Diffusion posterior sampling for linear inverse problem solving: A filtering perspective. In *The Twelfth International Conference on Learning Representations*, 2024.

Guilherme Fernandes, Vasco Ramos, Regev Cohen, Idan Szpektor, et al. Latent beam diffusion models for decoding image sequences. *arXiv e-prints*, pp. arXiv–2503, 2025.

Tomas Geffner, Kieran Didi, Zuobai Zhang, Danny Reidenbach, Zhonglin Cao, Jason Yim, Mario Geiger, Christian Dallago, Emine Kucukbenli, Arash Vahdat, and Karsten Kreis. Proteina: Scaling flow-based protein structure generative models. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=TVQLu34bdw.

Nate Gruver, Samuel Stanton, Nathan Frey, Tim GJ Rudner, Isidro Hotzel, Julien Lafrance-Vanasse, Arvind Rajpal, Kyunghyun Cho, and Andrew G Wilson. Protein design with guided discrete diffusion. *Advances in neural information processing systems*, 36:12489–12517, 2023.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

Emiel Hoogeboom, Vıctor Garcia Satorras, Clément Vignac, and Max Welling. Equivariant diffusion for molecule generation in 3d. In *International conference on machine learning*, pp. 8867–8887. PMLR, 2022.

John B Ingraham, Max Baranov, Zak Costello, Karl W Barber, Wujie Wang, Ahmed Ismail, Vincent Frappier, Dana M Lord, Christopher Ng-Thow-Hing, Erik R Van Vlack, et al. Illuminating protein space with a programmable generative model. *Nature*, 623(7989):1070–1078, 2023.

Michael Jendrusch and Jan O Korbel. Efficient protein structure generation with sparse denoising models. *Nature machine intelligence*, 7(9):1429–1445, 2025.

Jaehyeong Jo, Seul Lee, and Sung Ju Hwang. Score-based generative modeling of graphs via the system of stochastic differential equations. In *International conference on machine learning*, pp. 10362–10383. PMLR, 2022.

Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. *Advances in neural information processing systems*, 35:26565–26577, 2022.

Rohith Krishna, Jue Wang, Woody Ahern, Pascal Sturmfels, Preetham Venkatesh, Indrek Kalvet, Gyu Rie Lee, Felix S Morey-Burrows, Ivan Anishchenko, Ian R Humphreys, et al. Generalized biomolecular modeling and design with rosettafold all-atom. *Science*, 384(6693):eadl2528, 2024.

Seul Lee, Jaehyeong Jo, and Sung Ju Hwang. Exploring chemical space with score-based out-of-distribution generation. In *International Conference on Machine Learning*, pp. 18872–18892. PMLR, 2023.

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 *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=6QbbaEGkO7.

Zeming Lin, Halil Akin, Roshan Rao, Brian Hie, Zhongkai Zhu, Wenting Lu, Nikita Smetanin, Robert Verkuil, Ori Kabeli, Yaniv Shmueli, et al. Evolutionary-scale prediction of atomic-level protein structure with a language model. *Science*, 379(6637):1123–1130, 2023.

Tianyu Lu, Richard Shuai, Petr Kouba, Zhaoyang Li, Yilin Chen, Akio Shirali, Jinho Kim, and Po-Ssu Huang. Conditional protein structure generation with protpardelle-1c. *bioRxiv*, 2025.

Angus Phillips, Hai-Dang Dau, Michael John Hutchinson, Valentin De Bortoli, George Deligiannidis, and Arnaud Doucet. Particle denoising diffusion sampler. In *Proceedings of the 41st International Conference on Machine Learning*, pp. 40688–40724, 2024.

Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation. In *International conference on machine learning*, pp. 8821–8831. Pmlr, 2021.

Vignav Ramesh and Morteza Mardani. Test-time scaling of diffusion models via noise trajectory search. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=fnkkY7UGRl.

Subham S Sahoo, Marianne Arriola, Yair Schiff, Aaron Gokaslan, Edgar Marroquin, Justin T Chiu, Alexander Rush, and Volodymyr Kuleshov. Simple and effective masked diffusion language models. *Advances in Neural Information Processing Systems*, 37:130136–130184, 2024.

Romelia Salomon-Ferrer, David A Case, and Ross C Walker. An overview of the amber biomolecular simulation package. *Wiley Interdisciplinary Reviews: Computational Molecular Science*, 3(2): 198–210, 2013.

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 *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=Jp988ELppQ.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020.

Jiaming Song, Qinsheng Zhang, Hongxu Yin, Morteza Mardani, Ming-Yu Liu, Jan Kautz, Yongxin Chen, and Arash Vahdat. Loss-guided diffusion models for plug-and-play controllable generation. In *International Conference on Machine Learning*, pp. 32483–32498. PMLR, 2023.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021.

Masatoshi Uehara, Yulai Zhao, Kevin Black, Ehsan Hajiramezanali, Gabriele Scalia, Nathaniel Lee Diamant, Alex M Tseng, Tommaso Biancalani, and Sergey Levine. Fine-tuning of continuous-time diffusion models as entropy-regularized control. *arXiv preprint arXiv:2402.15194*, 2024.

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 *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=9qzpNSTUYp.

Michel Van Kempen, Stephanie S Kim, Charlotte Tumescheit, Milot Mirdita, Jeongjae Lee, Cameron LM Gilchrist, Johannes Söding, and Martin Steinegger. Fast and accurate protein structure search with foldseek. *Nature biotechnology*, 42(2):243–246, 2024.

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, et al. De novo design of protein structure and function with rfdiffusion. *Nature*, 620(7976):1089–1100, 2023.

Kevin E Wu, Kevin K Yang, Rianne Van den Berg, Sarah Alamdari, James Y Zou, Alex X Lu, and Ava P Amini. Protein structure generation via folding diffusion. *Nature communications*, 15(1): 1059, 2024.

Luhuan Wu, Brian Trippe, Christian Naesseth, David Blei, and John P Cunningham. Practical and asymptotically exact conditional sampling in diffusion models. *Advances in Neural Information Processing Systems*, 36:31372–31403, 2023.

Jason Yim, Brian L Trippe, Valentin De Bortoli, Emile Mathieu, Arnaud Doucet, Regina Barzilay, and Tommi Jaakkola. Se (3) diffusion model with application to protein backbone generation. In *International Conference on Machine Learning*, pp. 40001–40039. PMLR, 2023.

Jaesik Yoon, Hyeonseo Cho, Doojin Baek, Yoshua Bengio, and Sungjin Ahn. Monte carlo tree diffusion for system 2 planning. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=XrCbBdycDc.

Jiwen Yu, Yinhuai Wang, Chen Zhao, Bernard Ghanem, and Jian Zhang. Freedom: Training-free energy-guided conditional diffusion model. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 23174–23184, 2023.

Odin Zhang, Xujun Zhang, Haitao Lin, Cheng Tan, Qinghan Wang, Yuanle Mo, Qiantai Feng, Gang Du, Yuntao Yu, Zichang Jin, et al. Odesign: A world model for biomolecular interaction design. *arXiv preprint arXiv:2510.22304*, 2025.

XiangCheng Zhang, Haowei Lin, Haotian Ye, James Zou, Jianzhu Ma, Yitao Liang, and Yilun Du. Inference-time scaling of diffusion models through classical search. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=b7Ftp6U78i.

Zhuoqi Zheng, Bo Zhang, Kieran Didi, Kevin K Yang, Jason Yim, Joseph L Watson, Hai-Feng Chen, and Brian L Trippe. Motifbench: A standardized protein design benchmark for motif-scaffolding problems. *arXiv preprint arXiv:2502.12479*, 2025.

Rollback and Repair: Reusing Trajectory Assets for Test-Time Scaling in Protein Design

Appendix

The appendix is organized as follows:

- Appendix A provides additional theoretical analysis and proof details for TRACER.
- Appendix B presents additional experimental results on protein-binding protein design and motif scaffolding.
- Appendix C provides extended ablation studies on hyperparameter sensitivity.
- Appendix D outlines further implementation details of our method.
- Appendix E details the evaluation metrics and success criteria used in our experiments.
- Appendix F discusses the limitations of TRACER.
- Appendix G discloses AI use in this paper.

A Theoretical Analysis

Notation. Let p_θ denote the pretrained reverse denoising model, and let x_s^i denote candidate i at evaluation time s. In the algorithm, its value is estimated using a terminal rollout:

$$X_0^{i,s} \sim p_\theta(\cdot \mid x_s^i, c), \qquad v_s^i = r(X_0^{i,s}, c).$$

Thus, v_s^i is a sampled reward rather than a deterministic function of x_s^i.

For the theoretical analysis, we instead consider the expected terminal-rollout value

$$\bar{V}_s(x, c) := \mathbb{E}_{X_0 \sim p_\theta(\cdot|x,c)} \left[r(X_0, c) \right]. \tag{10}$$

This quantity differs from the risk-sensitive value V_s^λ defined in the main text. Our analysis concerns the local stability of $\bar{V}_s$, while the algorithm uses sampled rollout rewards for selection.

For a repair candidate i, let κ denote its selected rollback time, and let y_κ denote the retrieved reference. Write $B_i = 1 - M_i$ for the mask over editable atoms and $\sigma_{\kappa,s} = (\sigma_\kappa^2 - \sigma_s^2)^{1/2}$. For a realized rollback perturbation ϵ, define

$$x_\kappa = x_s^i + B_i \odot \eta \sigma_{\kappa,s} \epsilon, \qquad \widetilde{x}_\kappa = x_\kappa + \alpha B_i \odot (y_\kappa - x_\kappa).$$

Let $P_\theta^{\kappa \to s}(z, c)$ denote the distribution of the state at time s obtained by re-denoising from z at time κ. We define the expected replay value as

$$U_{\kappa \to s}(z, c) := \mathbb{E}_{X_s \sim P_\theta^{\kappa \to s}(z,c)} \left[\bar{V}_s(X_s, c) \right]. \tag{11}$$

The inner value $\bar{V}_s$ averages over terminal-rollout randomness, while the outer expectation averages over re-denoising randomness from κ to s.

Proposition: Local stability of expected rollout value. Assume $0 \le \alpha \le 1$. For a local retrieval radius $r_i > 0$, define

$$G_{\mathrm{loc}}(r_i) := \{\|B_i \odot (y_\kappa - x_\kappa)\|_F \le r_i\}. \tag{12}$$

Define the rollback consistency condition

$$G_{\mathrm{rb}}(\epsilon_{\mathrm{cyc}}, L_{\mathrm{rb}}) := \left\{W_1\big(P_\theta^{\kappa\to s}(x_\kappa, c), \delta_{x_s^i}\big) \le \epsilon_{\mathrm{cyc}} + L_{\mathrm{rb}}\eta\sigma_{\kappa,s}\|B_i \odot \epsilon\|_F\right\}. \tag{13}$$

Here, W_1 uses the Frobenius ground metric.

Assume the following regularity conditions hold on the relevant state domains, including the supports of the re-denoising laws.

Assumption 1. The expected terminal-rollout value is L_V-Lipschitz:

$$|\bar{V}_s(z, c) - \bar{V}_s(z', c)| \le L_V\|z - z'\|_F. \tag{14}$$

Assumption 2. The re-denoising law is L_{dyn}-Lipschitz in W_1:

$$W_1(P_\theta^{\kappa\to s}(z, c), P_\theta^{\kappa\to s}(z', c)) \le L_{\mathrm{dyn}}\|z - z'\|_F. \tag{15}$$

All relevant expectations and first moments are assumed finite.

Claim. On $G_{\mathrm{loc}} \cap G_{\mathrm{rb}}$, the expected value drift satisfies

$$\begin{aligned}&\big|U_{\kappa\to s}(\widetilde{x}_\kappa, c) - \bar{V}_s(x_s^i, c)\big| \\ &\quad\le L_V\epsilon_{\mathrm{cyc}} + L_V L_{\mathrm{rb}}\eta\sigma_{\kappa,s}\|B_i \odot \epsilon\|_F + L_V L_{\mathrm{dyn}}\alpha r_i.\end{aligned} \tag{16}$$

Proof. By the repair update and the local retrieval condition,

$$\|\widetilde{x}_\kappa - x_\kappa\|_F = \alpha\|B_i \odot (y_\kappa - x_\kappa)\|_F \le \alpha r_i. \tag{17}$$

Using Assumption 1 and the Kantorovich–Rubinstein inequality,

$$\begin{aligned}&\big|U_{\kappa\to s}(\widetilde{x}_\kappa, c) - \bar{V}_s(x_s^i, c)\big| \\ &\quad\le L_V W_1\big(P_\theta^{\kappa\to s}(\widetilde{x}_\kappa, c), \delta_{x_s^i}\big).\end{aligned} \tag{18}$$

The triangle inequality gives

$$\begin{aligned}&W_1\big(P_\theta^{\kappa\to s}(\widetilde{x}_\kappa, c), \delta_{x_s^i}\big) \\ &\quad\le W_1(P_\theta^{\kappa\to s}(\widetilde{x}_\kappa, c), P_\theta^{\kappa\to s}(x_\kappa, c)) \\ &\qquad + W_1\big(P_\theta^{\kappa\to s}(x_\kappa, c), \delta_{x_s^i}\big).\end{aligned} \tag{19}$$

By Assumption 2 and Eq. 17, the first term is at most $L_{\mathrm{dyn}}\alpha r_i$. On G_{rb}, the second term is at most

$$\epsilon_{\mathrm{cyc}} + L_{\mathrm{rb}}\eta\sigma_{\kappa,s}\|B_i \odot \epsilon\|_F.$$

Combining these bounds proves Eq. 16.

Relation to cached rollout scores. The algorithm compares sampled rewards rather than expected values. Define the discrepancy of the cached score as

$$e_s^i := |v_s^i - \bar{V}_s(x_s^i, c)|.$$

A further application of the triangle inequality yields

$$\begin{aligned}&\big|U_{\kappa\to s}(\widetilde{x}_\kappa, c) - v_s^i\big| \\ &\quad\le L_V\epsilon_{\mathrm{cyc}} + L_V L_{\mathrm{rb}}\eta\sigma_{\kappa,s}\|B_i \odot \epsilon\|_F + L_V L_{\mathrm{dyn}}\alpha r_i + e_s^i.\end{aligned} \tag{20}$$

This inequality holds for the realized cached score, including when it has been retained through proposal selection. We do not assume that such selected scores are unbiased, nor do we provide a concentration bound for e_s^i.

Remarks. The proposition provides a conditional local stability bound for the expected terminal-rollout value. Its three terms capture rollback consistency, stochastic perturbation magnitude, and reference-induced displacement. Comparison with a cached sampled reward additionally incurs the discrepancy e_s^i.

The bound controls the magnitude of expected value drift; it does not establish expected reward improvement or distributional preservation. The accept–reject rule ensures improvement in the compared rollout scores, but does not by itself guarantee improvement in expected terminal reward.

Interpretation of the three error terms. Eq. 16 decomposes the value drift into three factors. The first term, $L_V \epsilon_{\mathrm{cyc}}$, measures the intrinsic consistency of the rollback and re-denoising cycle. It captures whether an unrepaired rollback state can return to the current state under the pretrained dynamics, and therefore quantifies the reliability of using rollback as an intervention point. The second term, $L_V L_{\mathrm{rb}} \eta \sigma_{\kappa,s} \|B^i \odot \epsilon\|_F$, corresponds to the stochastic exploration introduced by matched re-noising. Thus, η controls a practical exploration–stability trade-off: it should be large enough to explore alternative local continuations, but not so large that the replay becomes dominated by injected noise. The third term, $L_V L_{\mathrm{dyn}} \alpha r_i$, measures the additional value drift introduced by reference-guided repair. It captures the joint effect of repair strength and retrieval distance: increasing α strengthens the guidance from the retrieved reference, whereas local retrieval keeps r_i small and limits how far the repaired rollback state deviates from the original rollback state before re-denoising.

Table 6: Protein-binding protein design results on CD3d, EGFR, PD-1, and FGFR2.

Methods	CD3d			EGFR			PD-1			FGFR2		
	pLDDT ↑	min iPAE ↓	USR ↑	pLDDT ↑	min iPAE ↓	USR ↑	pLDDT ↑	min iPAE ↓	USR ↑	pLDDT ↑	min iPAE ↓	USR ↑
APM	70.74	6.64	7	86.27	13.31	4	62.23	13.48	1	75.08	6.71	8
+ MCTS	71.72	4.29	3	85.95	13.06	2	62.56	12.21	1	79.83	4.30	3
+ FKS	78.07	4.09	3	83.72	9.61	2	63.83	10.55	1	80.69	2.82	6
+ TRACER	**78.86**	**3.50**	**9**	**86.97**	**9.51**	**5**	**63.96**	**10.35**	**2**	**80.91**	**2.79**	**10**
Proteina-C	86.14	2.95	1	84.89	8.76	11	81.96	9.37	1	80.95	3.13	0
+ MCTS	91.56	1.05	1	90.00	2.57	22	88.01	2.56	7	87.91	0.99	11
+ FKS	90.25	1.33	1	87.15	5.32	12	85.62	5.10	4	85.72	1.57	5
+ TRACER	**91.59**	**1.04**	**3**	**90.29**	**2.42**	**25**	**88.50**	**1.75**	**10**	**88.18**	**0.92**	**14**
ODesign	74.18	8.82	5	85.26	9.96	16	64.50	15.97	2	75.33	5.03	19
+ MCTS	71.80	6.99	1	85.99	9.32	6	61.38	14.14	0	78.27	2.37	8
+ FKS	75.42	9.70	1	85.31	9.92	10	65.45	16.25	1	77.86	4.33	8
+ TRACER	**75.47**	**6.78**	**7**	**86.16**	**7.85**	**20**	**65.97**	**14.01**	**3**	**79.13**	**2.16**	**27**

Table 7: Protein-binding protein design results on InsulinR, PDGFR, VirB8, and TrkA.

Methods	InsulinR			PDGFR			VirB8			TrkA		
	pLDDT ↑	min iPAE ↓	USR ↑	pLDDT ↑	min iPAE ↓	USR ↑	pLDDT ↑	min iPAE ↓	USR ↑	pLDDT ↑	min iPAE ↓	USR ↑
APM	84.30	7.91	3	77.46	8.08	13	83.50	10.63	7	76.37	9.44	3
+ MCTS	86.26	5.26	5	80.43	5.66	5	86.57	9.90	4	77.11	5.83	3
+ FKS	86.33	7.80	3	80.40	4.35	2	86.07	8.19	4	83.07	7.92	2
+ TRACER	**87.03**	**5.13**	**6**	**81.27**	**4.26**	**15**	**86.97**	**7.81**	**9**	**84.56**	**5.28**	**4**
Proteina-C	85.80	4.41	6	84.48	2.95	1	85.62	7.87	8	85.69	5.87	15
+ MCTS	90.91	1.34	12	88.81	1.10	10	89.91	1.59	26	93.75	1.07	54
+ FKS	89.75	2.26	24	86.91	1.70	3	88.98	3.24	32	89.68	3.16	32
+ TRACER	**91.81**	**1.34**	**28**	**89.87**	**1.04**	**12**	**89.97**	**1.38**	**34**	**94.43**	**1.02**	**59**
ODesign	86.39	4.08	43	80.68	4.08	48	81.54	6.94	24	80.26	7.29	29
+ MCTS	87.51	3.74	18	81.53	3.78	11	81.72	6.73	8	81.29	6.18	11
+ FKS	87.13	3.91	13	81.72	3.74	11	81.74	6.62	8	80.73	6.93	13
+ TRACER	**88.67**	**3.12**	**49**	**81.96**	**3.43**	**57**	**82.68**	**6.32**	**45**	**82.31**	**5.07**	**37**

B Further Experimental Results

B.1 Additional Per-Target Results

In Section 4.2, we report representative results on *de novo* protein design, protein-binding protein design, and motif scaffolding. Here we provide the full per-target results for the remaining protein–protein binder targets and motif scaffolding cases.

Additional protein-binding protein results. Tables 6 and 7 extend the binder-design evaluation to the remaining eight targets. The results support two observations. First, TRACER consistently improves interface quality across all additional protein-binding settings. It improves the source generators by an average of +4.08 pLDDT and reduces minimum iPAE by 3.14, achieving the best min iPAE in every reported setting. Second, these quality gains do not come from mode concentration: TRACER improves or preserves USR relative to the source generator in all settings, whereas MCTS and FKS often reduce USR despite improving some reward-related metrics.

Additional motif scaffolding results. Tables 8–10 report the remaining motif scaffolding cases. TRACER improves or matches the source generator on Novelty, Motif Score, and USR in every additional setting. The key trend is that TRACER improves novelty without sacrificing motif satisfaction: it obtains the highest USR in all settings and the highest novelty in all settings. The fully unsuccessful Protpardelle-1c cases, such as 02_1ITU and 11_3TQB/14_5IUS, further clarify that TRACER depends on an informative asset library rather than creating successful modes from an entirely non-successful base distribution. For the remaining MotifBench targets not reported in the paper, both the baselines and TRACER achieve zero success, reflecting the difficulty of these targets.

Summary. Overall, the additional targets confirm the main paper claim: TRACER improves task-specific quality while preserving diverse successful modes. The gains are consistent across heterogeneous generators and targets, and the observed failures align with the method's dependence on recoverable intermediate trajectory assets.

Table 8: Motif scaffolding results on 01_1LDB, 02_1ITU, 04_5WN9, and 05_5ZE9.

Methods	01_1LDB			02_1ITU			04_5WN9			05_5ZE9		
	Novelty ↑	Motif Score ↑	USR ↑	Novelty ↑	Motif Score ↑	USR ↑	Novelty ↑	Motif Score ↑	USR ↑	Novelty ↑	Motif Score ↑	USR ↑
Protpardelle-1c	0.226	39.38	3	0.000	0.00	0	0.309	17.50	1	0.210	17.50	1
+ MCTS	0.576	30.00	2	0.000	0.00	0	0.602	17.50	1	0.000	0.00	0
+ FKS	0.000	0.00	0	0.000	0.00	0	0.677	17.50	1	0.000	0.00	0
+ **TRACER**	**0.646**	**46.67**	**4**	0.000	0.00	0	**0.692**	**30.00**	**2**	**0.760**	**30.00**	**2**
ODesign	0.539	17.50	1	0.648	39.38	3	0.746	46.67	4	0.657	52.50	5
+ MCTS	0.000	0.00	0	0.767	39.38	3	0.701	39.38	3	0.636	46.67	4
+ FKS	0.000	0.00	0	0.727	17.50	1	0.801	17.50	1	0.652	39.38	3
+ **TRACER**	**0.751**	**30.00**	**2**	**0.775**	**52.50**	**5**	**0.820**	**57.27**	**6**	**0.685**	**61.25**	**7**

Table 9: Motif scaffolding results on 06_6E6R, 10_7WRK, 11_3TQB, and 14_5IUS.

Methods	06_6E6R			10_7WRK			11_3TQB			14_5IUS		
	Novelty ↑	Motif Score ↑	USR ↑	Novelty ↑	Motif Score ↑	USR ↑	Novelty ↑	Motif Score ↑	USR ↑	Novelty ↑	Motif Score ↑	USR ↑
Protpardelle-1c	0.238	64.62	8	0.229	52.50	5	0.000	0.00	0	0.000	0.00	0
+ MCTS	0.633	61.25	7	0.000	0.00	0	0.000	0.00	0	0.000	0.00	0
+ FKS	0.690	52.50	5	0.000	0.00	0	0.000	0.00	0	0.000	0.00	0
+ **TRACER**	**0.746**	**78.75**	**15**	**0.251**	**61.25**	**7**	0.000	0.00	0	0.000	0.00	0
ODesign	0.694	**67.50**	**9**	0.683	17.50	1	0.682	61.25	7	0.682	17.50	1
+ MCTS	0.727	61.25	7	0.000	0.00	0	0.712	30.00	2	0.719	17.50	1
+ FKS	0.709	46.67	4	0.697	17.50	1	0.708	30.00	2	0.716	17.50	1
+ **TRACER**	**0.770**	**67.50**	**9**	**0.753**	**30.00**	**2**	**0.724**	**64.62**	**8**	**0.725**	**30.00**	**2**

Multi-seed Experiment To assess robustness to stochastic variation, we repeat MCTS, FKS, and TRACER over five independent random seeds and report the mean and variance in Tab. 11. TRACER consistently retains the best performance across all three protein-design tasks, with small run-to-run variation. In particular, its USR gains persist across *de novo* design, protein-binder design, and motif scaffolding, further supporting the conclusion that TRACER's gains in task-specific quality and unique successful generation are robust to stochastic variation.

B.2 Additional Compute-Scaling Results

Table 10: Motif scaffolding results on 15_7A8S, 16_7BNY, 17_7DGW, and 28_5YUI.

Methods	15_7A8S			16_7BNY			17_7DGW			28_5YUI		
	Novelty ↑	Motif Score ↑	USR ↑	Novelty ↑	Motif Score ↑	USR ↑	Novelty ↑	Motif Score ↑	USR ↑	Novelty ↑	Motif Score ↑	USR ↑
Protpardelle-1c	0.000	0.00	0	0.000	0.00	0	0.000	0.00	0	0.000	0.00	0
+ MCTS	0.000	0.00	0	0.000	0.00	0	0.000	0.00	0	0.000	0.00	0
+ FKS	0.000	0.00	0	0.000	0.00	0	0.000	0.00	0	0.000	0.00	0
+ **TRACER**	0.000	0.00	0	0.000	0.00	0	0.000	0.00	0	0.000	0.00	0
ODesign	0.614	17.50	1	0.580	46.67	4	0.926	39.38	3	0.647	17.50	1
+ MCTS	0.631	17.50	1	0.589	30.00	2	0.927	30.00	2	0.708	17.50	1
+ FKS	0.636	17.50	1	0.585	39.38	3	0.927	30.00	2	0.741	17.50	1
+ **TRACER**	**0.677**	**39.38**	**3**	**0.641**	**57.27**	**6**	**0.931**	**52.50**	**5**	**0.762**	**30.00**	**2**

Table 11: Multi-seed mean (variance) across three protein-design tasks.

	De novo design			Protein-binder design (IL7RA)			Motif scaffolding (09_7CG5)		
Method	**pLDDT↑**	**Cα RMSD ↓**	**USR↑**	**pLDDT↑**	**min iPAE↓**	**USR ↑**	**Novelty↑**	**Motif Score↑**	**USR ↑**
ODesign	86.68 (0.06)	1.74 (0.02)	26.5 (0.10)	86.64 (0.06)	5.65 (0.04)	26.2 (0.16)	0.675 (0.01)	40.83 (8.51)	3.2 (0.16)
+ MCTS	87.47 (0.02)	1.62 (0.01)	16.9 (0.04)	84.61 (0.01)	7.24 (0.02)	5.2 (0.16)	0.687 (0.01)	53.45 (3.64)	5.2 (0.16)
+ FKS	87.25 (0.06)	1.64 (0.05)	16.5 (0.50)	86.47 (0.08)	5.53 (0.06)	4.8 (0.56)	0.552 (0.01)	42.29 (12.76)	3.4 (0.24)
+ **TRACER**	**87.78 (0.04)**	**1.59 (0.03)**	**32.1 (0.24)**	**86.74 (0.05)**	**5.45 (0.03)**	**48.0 (0.40)**	**0.698 (0.01)**	**58.07 (2.53)**	**6.2 (0.16)**

We extend the compute-scaling evaluation in Section 4.6 to protein-binding protein design on IL7RA and motif scaffolding on 09_7CG5, using ODesign as the base generator. Figure 6 reports the cumulative number of unique successful designs under different GPU-time budgets for unguided ODesign, MCTS, FKS, and TRACER.

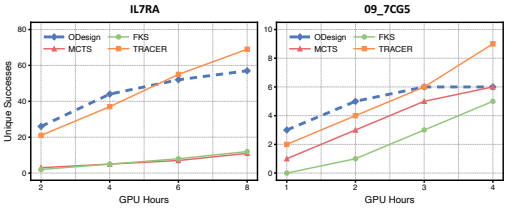

Figure 6: Additional inference-time compute-scaling results.

Across both cases, TRACER exhibits a consistent compute-dependent advantage: unguided sampling can be more effective at small budgets, while TRACER catches up and ultimately achieves more unique successful designs as the compute budget increases. TRACER also consistently outperforms MCTS and FKS across the evaluated budgets. Together with the de novo design results, these findings suggest that the benefits of rollback and repair extend to binder design and motif scaffolding.

B.3 Additional Baseline Comparison

To further contextualize TRACER within recent work on inference-time scaling of diffusion models, we compare with Zhang et al. (2026) across de novo design, protein-binding protein design (IL7RA), and motif scaffolding (09_7CG5). Table 12 reports the results. TRACER shows stronger overall performance across the three tasks, with particularly consistent gains in de novo design and motif scaffolding. These results support the use of intermediate states as reusable trajectory assets for adaptive, reference-guided trajectory repair.

C Additional Ablation Study

Section 4.5 analyzes the sensitivity of TRACER to the reference attraction strength α and the stochastic exploration coefficient η. We further study three residual-rollback hyperparameters: the temporal discount factor γ, the rollback-gap penalty ω, and per-candidate repair budget P. All experiments are conducted on the ODesign *de novo* protein design task.

Key observations. Figure 7 provides three observations. First, TRACER remains above the unguided ODesign baseline across the tested γ values, and $\gamma = 0.96$ gives the best trade-off between USR and pLDDT. This complements the component ablation in Section 4.3: temporal discounting is useful for residual-based rollback localization, while the method is not sensitive to a narrowly tuned

Table 12: Comparison with Zhang et al. (2026).

Model + Method	De Novo Design			Protein-Binding Protein Design			Motif Scaffolding		
	pLDDT ↑	Cα RMSD ↓	USR ↑	pLDDT ↑	min iPAE ↓	USR ↑	Novelty ↑	Motif Score ↑	USR ↑
ODesign	86.68	1.74	26.5	86.64	5.65	26.0	0.675	39.38	3.0
+ Zhang et al. (2026)	87.11	1.64	26.0	84.13	7.34	19.0	0.527	30.00	2.0
+ TRACER (ours)	**87.78**	**1.59**	**32.0**	**86.74**	**5.45**	**48.0**	**0.698**	**57.27**	**6.0**

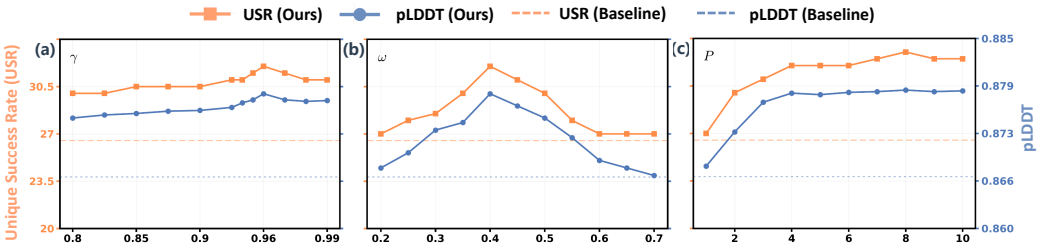

Figure 7: Sensitivity of TRACER to residual-rollback hyperparameters on the ODesign *de novo* protein design task. Dashed lines denote the unguided ODesign baseline.

discount value. Second, the best performance is obtained with an intermediate rollback-gap penalty. This follows the role of ω in Eq. (6), where it explicitly regularizes the rollback horizon. A very small ω weakens rollback-depth regularization, whereas a very large ω suppresses useful rollback interventions. Third, increasing P improves performance mainly at small budgets and saturates around $P = 4$. Thus larger P increases the chance of finding an improving continuation, but also adds extra re-denoising and reward-evaluation cost. We therefore use $P = 4$ in the main experiments, which captures most of the gain while preserving inference efficiency.

D More Implementation Details

D.1 More Test-time Scaling Method Details

Feynman–Kac Steering (FKS). FKS guides the discrete reverse diffusion process by sampling from a reward-tilted version of the model distribution. The tilted trajectory distribution is:

$$q_\beta(\tau \mid c) \propto p_\theta(\tau \mid c) \exp(\beta r(x_0, c)),$$

where c denotes the conditioning information, $r(x_0, c)$ is the reward of the final clean sample, and β is an inverse-temperature parameter controlling the strength of reward steering.

At each search update, the method maintains N particles, where each particle corresponds to a partial denoising trajectory. Each particle is expanded into L candidate next states, yielding $N \times L$ intermediate candidates. These candidates are rolled out to x_0 and evaluated by the reward. FKS then resamples N particles from the candidate pool according to reward-induced probabilities, rather than deterministically retaining the top-N candidates. Following prior work (Didi et al., 2026), we set $L = 4$ and the inverse temperature to $\beta = 10$.

Monte Carlo Tree Search (MCTS). MCTS treats reverse diffusion as a search tree over partial denoising trajectories. Each node corresponds to a state x. Expanding a node samples child states by running the reverse process from the current scheduled time to the next one. The resulting children are rolled out to x_0 for reward evaluation. For an expanded child node u with parent node v, we use the UCB score:

$$\mathrm{UCB}(u) = Q(u) + C\sqrt{\frac{\log(n(v)+1)}{n(u)+1}},$$

where $Q(u)$ is the average reward estimated for node u, $n(u)$ is the visit count of node u, $n(v)$ is the visit count of its parent node, and C is the exploration constant. The first term favors child nodes with high estimated reward, while the second term encourages exploration of less-visited child nodes.

Algorithm 1 Trajectory Repair with Asset Construction and Evaluation Reuse (TRACER)

Input: Pretrained denoising model p_θ, context c, candidate set $\{x_T^i\}_{i=1}^N$, reward proxy $\widehat{V}$, evaluation schedule $\mathcal{S} = \{s_0, \ldots, s_B\}$, asset library $\mathcal{A} = \emptyset$, asset capacity K, discount γ, rollback penalty ω, maximum rollback horizon $H_{\max}$, noise scale η, reference strength α, proposal number P.
Output: Final clean designs $\{x_0^i\}_{i=1}^N$.
for evaluation step $s \in \mathcal{S}$ **do**
  Denoise all candidates to step s using the pretrained reverse transition p_θ;
  Estimate values $v_s^i = \widehat{V}_s(x_s^i, c)$ for all candidates;
  Add time-indexed assets $\mathcal{A}_s \leftarrow \mathcal{A}_s \cup \{(s, x_s^i, v_s^i)\}_{i=1}^N$;
  If $|\mathcal{A}| > K$, evict the oldest assets until $|\mathcal{A}| \le K$;
  Compute group-relative advantages and select repair candidates;
  for candidate $i \in \mathcal{I}_s$ **do**
    Compute centered temporal residuals from the value history of candidate i;
    Select rollback time $\kappa_i^\star$ by downstream degradation with rollback penalty;
    for $p = 1, \ldots, P$ **do**
      Roll back x_s^i to $\kappa_i^\star$ by matched stochastic re-noising;
      Retrieve high-value references at the same time step from $\mathcal{A}_{\kappa_i^\star}$ by local compatibility;
      Apply masked residual attraction toward the retrieved reference;
      Re-denoise the repaired rollback state back to step s using p_θ;
      Evaluate repaired proposal value $\widetilde{v}_s^{i,p}$;
    end for
    Select the best proposal $p^\star = \arg\max_p \widetilde{v}_s^{i,p}$;
    if $\widetilde{v}_s^{i,p^\star} > v_s^i$ **then**
      Replace x_s^i with $\widetilde{x}_s^{i,p^\star}$ and update v_s^i;
      Add repaired state-value pair (s, x_s^i, v_s^i) to $\mathcal{A}_s$;
      If $|\mathcal{A}| > K$, evict the oldest assets until $|\mathcal{A}| \le K$;
    end if
  end for
end for
return $\{x_0^i\}_{i=1}^N$.

MCTS uses stochastic expansion in the diffusion process. At each node, a new child is expanded with probability ϵ; otherwise, the search selects the already expanded child with the highest UCB score. If the current node has no children, a new child is always expanded. For reward evaluation, intermediate states are completed to clean samples before scoring. The resulting reward $r(x_0, c)$ is used to update the value estimates of the visited nodes. After each round of N_{sim} simulations, the search advances to the child node with the highest estimated reward, and the procedure is repeated until the trajectory reaches x_0. In our experiments, we set $\epsilon = 0.5, C = 1.0, N_{\text{sim}} = 10$.

D.2 Pseudocode of TRACER

Algorithm 1 outlines the complete execution workflow of our proposed TRACER framework. Operating over a scheduled set of denoising evaluation steps, TRACER continuously converts reward feedback into reusable time-indexed trajectory assets. The procedure follows a conditional rollback-and-repair logic: 1) at each scheduled step, TRACER evaluates all candidates, organizes their state–time–value triplets, and uses group-relative advantages to identify underperforming candidates; 2) for each repair candidate, TRACER localizes the rollback time using centered temporal residuals, retrieves high-value assets from the same denoising time, and constructs multiple reference-guided repair proposals. The best repaired proposal is accepted only when it improves the current value, after which the repaired state is merged back into the asset library. In this way, TRACER differs from selection-based test-time scaling methods: instead of discarding low-scoring trajectories, it reuses evaluated intermediate states as structured assets for corrective search, improving task-specific protein design without retraining the pretrained generator.

D.3 More Implementation Details of TRACER

In *de novo* protein design, we choose pLDDT predicted by ESMFold (Lin et al., 2023) as reward. In protein-binding protein design, we use the negative min iPAE predicted by AlphaFold3 (Abramson et al., 2024) as the reward, so that lower interface prediction error corresponds to higher reward. In motif scaffolding, we use negative Cα-aligned Cα RMSD. For each model, we perform four evaluations, uniformly distributed over the final 25% of the denoising trajectory.

E More Experimental Details

E.1 Details of Metrics

pLDDT: pLDDT (predicted Local Distance Difference Test) is a confidence score used to evaluate the reliability of the local structure in predicted proteins. It is defined by the confidence of model predictions, assigning a confidence value to each residue. A higher pLDDT score indicates greater model confidence and suggests increased structural stability.

min iPAE: min iPAE (minimum interface predicted Aligned Error) is an interface-level confidence score used to evaluate whether two interacting protein chains are predicted to bind in a reliable relative orientation. It is computed from the predicted aligned error between residues across the interface, and the minimum value is used to reflect the best predicted interface quality. A lower min iPAE score indicates higher confidence in the predicted interaction and suggests better binding accuracy.

Novelty: Novelty quantifies the structural distance of successful scaffolds to known structures in the PDB. Following MotifBench, for each successful scaffold, Foldseek-Search is used to approximate the highest TM-score to any structure in the PDB. For each success cluster, novelty is computed as the mean of $1 - \mathrm{TMscore}$ within the cluster, and the final novelty is averaged across clusters. A higher novelty score indicates that the successful scaffolds are more structurally distinct from known PDB structures.

Motif Score: Motif score is a task-specific metric for motif scaffolding that evaluates the number of unique successful scaffolds satisfying the target motif constraints. Following MotifBench, for each motif case, it is computed from the number of unique successful solutions n using the saturating function $100 \cdot \frac{1-\left(1+\frac{n}{\alpha_{\mathrm{MB}}}\right)^{-1}}{1-\left(1+\frac{100}{\alpha_{\mathrm{MB}}}\right)^{-1}}$, where $\alpha_{\mathrm{MB}} = 5$. A higher Motif score indicates that the method generates more diverse successful scaffolds while correctly preserving the target motif.

USR: USR (Unique Success Rate) measures the fraction of generated samples that correspond to structurally distinct successful designs. Specifically, successful samples are clustered using Foldseek (Van Kempen et al., 2024) based on their structures, and USR is computed as the number of clusters normalized by the total number of generated samples. A higher USR indicates that the method generates a larger fraction of structurally diverse successful designs.

E.2 Details of Success Criteria

We follow the success criteria used in the corresponding original papers for each task and generative model. Table 13 summarizes the criteria used in our evaluation.

F Limitations

First, our evaluations focus on protein design. Although this setting covers multiple representative protein design tasks, including de novo generation, protein-binding protein design, and motif scaffolding, we have not evaluated TRACER on other biomolecular modalities such as DNA or RNA design. Extending TRACER to these settings may require modality-specific definitions of intermediate states, reward proxies, and local compatibility metrics. Second, our current implementation is specialized to trajectory-based protein structure generators. The constructed assets are used within the same generator and design task, and we do not claim that trajectory assets can be directly reused across different molecular representations, reward functions, or model architectures. Future work will

Table 13: Success criteria used for each task and generative model.

Task	Model	Success criterion
De novo protein design	APM	Backbone-aligned RMSD < 2.0 Å.
	Proteina	Backbone-aligned RMSD < 2.0 Å.
	ODesign	pLDDT > 70 and backbone-aligned RMSD < 2.0 Å.
Protein-binding protein design	APM	pLDDT > 80 and ipTM > 0.8.
	Proteina-C	iPAE < 7 Å, complex pLDDT > 90, and binder scRMSD < 1.5 Å.
	ODesign	iPAE < 10.85, pLDDT > 80, ipTM > 0.5, and complex RMSD < 2.5 Å.
Motif scaffolding	Protpardelle–1c	N, Cα, C-aligned motif RMSD $<$ 1.0 Å and Cα-aligned Cα RMSD < 2.0 Å.
	ODesign	N, Cα, C-aligned motif RMSD $<$ 1.0 Å and Cα-aligned Cα RMSD < 2.0 Å.

investigate how the rollback-and-repair principle can be adapted to broader biomolecular design problems beyond proteins.

G USAGE OF LLMS

Large language models were employed as an auxiliary tool to support manuscript preparation, including grammar checking, sentence refinement, and clarification of technical descriptions. All AI-suggested text was carefully reviewed and revised by the authors to ensure accuracy, clarity, and scientific integrity.