FARFLOW: LEARNING FINITE-SPAN FLOW MAPS FOR FEW-STEP PROTEIN BACKBONE GENERATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Diffusion- and flow-based models generate high-quality protein backbones, but reducing sampling cost requires reliable predictions over finite time intervals. We introduce Finite-Span Anchoring and Replay Flow (FARFlow), a solution-function-based method for few-step C_α backbone generation. FARFlow separates the requested prediction span from the input noise level through an unscaled, zero-anchored additive condition. This representation leaves the local-velocity condition unchanged at zero span while providing a smoothly varying correction for finite-span requests. Direct endpoint anchoring supervises predictions using paired clean backbones, and fresh-noise replay extends this supervision to inputs reconstructed from predicted endpoints. Under constrained budgets, FARFlow achieves higher designability than compared baselines while retaining reasonable structural diversity. On the standard multi-length benchmark, it maintains high generation quality across all evaluated lengths, achieving aggregate designability comparable to higher-budget multi-step models with substantially fewer network evaluations.

1 INTRODUCTION

De novo protein backbone generation provides candidate three-dimensional structures for subsequent sequence design and experimental evaluation (Watson et al., 2023). Recent diffusion- and flow-based models have made substantial progress in this task by learning distributions over protein structures, enabling the generation of diverse and designable backbones (Bose et al., 2024; Geffner et al., 2025). However, their conventional samplers rely on iterative denoising or numerical integration of learned velocity fields, requiring repeated network evaluations that limit the throughput of candidate generation. Simply reducing the number of sampling steps can compromise designability (Yue et al., 2025), motivating methods that improve sampling efficiency while preserving backbone generation quality.

Recent advances in generative modeling learn endpoint mappings or finite-time transitions to reduce sampling costs (Song et al., 2023; Frans et al., 2025; Luo et al., 2026), with related formulations explored in protein generation (Woo et al., 2026). Such predictors must distinguish the noise level of the input from the interval requested by the prediction. A single network may be trained for both local velocity and finite-span mapping, even at the same input state. We focus on how to represent these requests and supervise the clean-endpoint predictions used in sampling. When endpoint prediction alternates with re-noising, a further consideration arises: later inputs contain structural estimates supplied by the model itself, whereas analytic training inputs are constructed from clean data.

We introduce **Finite-Span Anchoring and Replay Flow (FARFlow)**, using Proteína's coordinate Transformer within a solution-function framework. Its central representation change is an additive span condition evaluated on the original time scale and anchored at zero. Zero initialization preserves the base conditions at the start of training; zero anchoring keeps the added branch inactive for local-velocity requests after training begins. A norm bound characterizes how this correction varies near zero span. We then train the endpoint request using paired clean-backbone targets. Direct anchoring acts on analytic inputs, and fresh-noise replay applies the same supervision after re-noising a detached predicted endpoint. These additions retain the coordinate Transformer and endpoint re-noising sampler.

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.

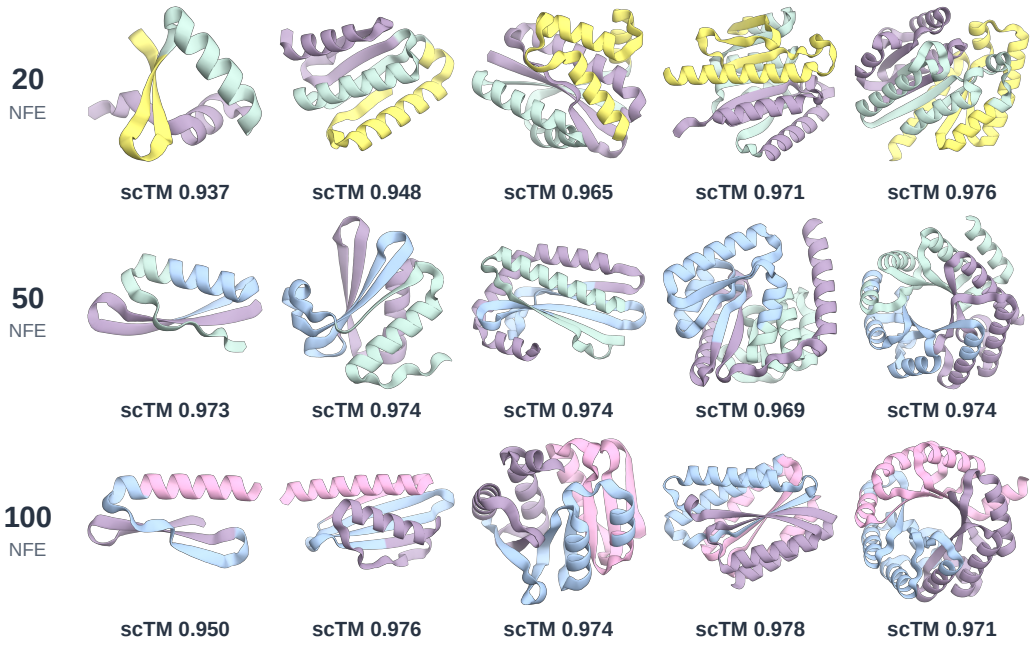


Figure 1: **Selected FARFlow backbones.** Rows show 20, 50, and 100 NFE; each structure is labelled with its scTM score. These examples illustrate generated structures, while aggregate quality is evaluated in Section 4.

We evaluate FARFlow at 20, 50, and 100 network function evaluations (NFE) on a dense 60–128-residue benchmark, and at 50 and 100 NFE on a five-length benchmark spanning 50–250 residues. The model achieves high designability while showing measurable trade-offs in diversity and reference-database novelty. Two component comparisons address different aspects of the method: one evaluates the zero-anchored span representation against the legacy encoding, while the other compares anchoring, an additional analytic-input pass, and model-induced replay.

Our main contributions are as follows:

- We develop an unscaled, zero-anchored span representation for protein solution maps. We establish its boundary and local-variation properties and evaluate the combined encoding change without adding backbone layers.
- We combine paired endpoint anchoring with fresh-noise replay. An analytic-input control retains the additional endpoint prediction to test the contribution of its model-induced input.
- We evaluate quality and sampling cost across NFE budgets and backbone lengths, identifying both competitive designability and the remaining reference-novelty trade-off.

2 Related Work

2.1 Protein Backbone Generation

Many protein backbone generators model either residue-level rigid frames or C_α coordinates. FrameDiff and RFdiffusion iteratively denoise residue positions and orientations (Yim et al., 2023b; Watson et al., 2023), whereas FoldFlow and FrameFlow apply flow matching to residue frames on $\mathrm{SE}(3)$ (Bose et al., 2024; Yim et al., 2023a). Genie and Genie2 instead diffuse C_α coordinates, using coordinate-derived frames for geometric processing (Lin & AlQuraishi, 2023; Lin et al., 2024). Proteína models a flow directly in C_α coordinate space using a scalable Transformer, without separately generating residue rotations (Geffner et al., 2025). La-Proteina jointly generates amino acid sequences and full-atom structures through partially latent flow matching (Geffner et al., 2026). FARFlow

focuses on conditioning and training finite-span endpoint maps for few-step backbone generation, using Proteína's C_α-coordinate Transformer.

2.2 Few-Step Generative Modeling

Several complementary strategies reduce sampling costs. Progressive distillation compresses a pretrained sampler into fewer student steps (Salimans & Ho, 2022), while rectified flow can progressively straighten transport trajectories through reflow, supporting coarser numerical integration (Liu et al., 2023). Related approaches learn clean-endpoint mappings or finite-interval transformations: Consistency Models map states along the same generative trajectory to a common clean endpoint (Song et al., 2023); Shortcut Models condition on noise level and requested step size (Frans et al., 2025); MeanFlow models interval-averaged velocities (Geng et al., 2025); and SoFlow learns the solution function of a velocity ODE using flow matching and solution consistency (Luo et al., 2026). These approaches are not mutually exclusive: consistency-based models can be trained by distillation or directly from data.

Protein-specific acceleration includes quaternion-flow rectification in ReQFlow (Yue et al., 2025), score distillation in SPID (Xie et al., 2025), and manifold flow-map learning in Riemannian MeanFlow (Woo et al., 2026). FARFlow builds on solution-function learning, focusing on span conditioning and paired endpoint supervision without a separate distillation teacher. Its replay mechanism differs from self-conditioning, as used in Proteína (Geffner et al., 2025): self-conditioning supplies a predicted clean structure as additional conditioning for the current noisy input; replay instead combines a predicted endpoint with independent fresh noise to construct another noisy coordinate input, then supervises a second prediction against the same paired clean backbone. In our experiments, QFlow denotes the unrectified quaternion-flow baseline defined in ReQFlow (Yue et al., 2025), whereas RMF denotes Riemannian MeanFlow (Woo et al., 2026). They provide complementary few-step reference points: QFlow tests reduced-step integration of the quaternion-flow model, while RMF directly learns manifold flow maps.

3 Finite-Span Anchoring and Replay Flow

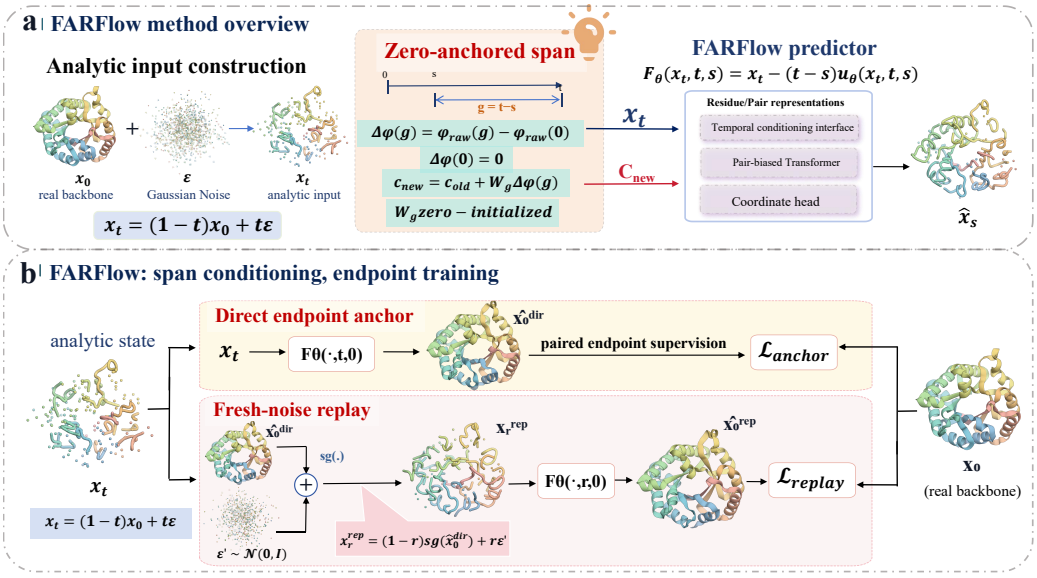


Figure 2: **Overview of FARFlow.** (a) Analytic inputs and zero-anchored span conditioning feed the solution-map predictor. (b) Direct anchoring supervises the clean-endpoint request. Replay reuses that prediction, detaches it, and mixes it with independent noise before a second prediction against the same clean target. Masking and centering are omitted here; all coordinates and noises use the projected subspace in Section 3.1. Ribbons illustrate states schematically; outputs contain only C_α coordinates.

3.1 Problem Setup and Finite-Span Prediction

We represent a clean backbone by its C_α coordinates $x_0 \in \mathbb{R}^{N\times 3}$. Let P_m mask padded residues and subtract the centroid of valid residues, with $P_m x_0 = x_0$. We use $t = 0$ for data and $t = 1$ for noise. We draw projected Gaussian noise $\xi = P_m z$, $z \sim \mathcal{N}(0, I)$, and use source and target times $0 \le s \le t \le 1$. With prediction span $g = t - s$, the conditional path and SoFlow's Euler-parameterized solution map (Luo et al., 2026) are

$$\begin{aligned} x_t &= (1-t)x_0 + t\xi, \\ \widehat{x}_s &= F_\theta(x_t, t, s) = P_m[x_t - g\, u_\theta(x_t, t, s)]\,. \end{aligned} \tag{1}$$

Here u_θ is the network's vector output and g specifies the requested prediction span. The parameterization gives $F_\theta(x_t, t, t) = x_t$; its zero-span vector output $P_m u_\theta(x_t, t, t)$ is trained against the conditional velocity $\xi - x_0$. We retain Proteína's coordinate Transformer (Geffner et al., 2025) and SoFlow's base supervision, introducing span conditioning and endpoint objectives as described below. Appendix B.1 gives the coordinate conventions.

3.2 Zero-Anchored Span Conditioning

Source time t specifies the input noise level, whereas the span $g = t - s$ specifies the prediction request. The same network must support both local-velocity supervision at $g = 0$ and finite-span predictions at $g > 0$. We retain the source-time condition and represent the span through separate additive branches in the existing sequence and pair pathways.

We evaluate sinusoidal features on the fractional span without the legacy time multiplier, subtract their value at zero, and project the difference:

$$\begin{aligned} \Delta\phi(g) &= \phi(g) - \phi(0), \\ c^{(a)} &= c^{(a)}_{\text{base}} + W^{(a)}_g \Delta\phi(g). \end{aligned} \tag{2}$$

Here $a \in \{\text{seq}, \text{pair}\}$, and each $W^{(a)}_g$ is bias-free and zero-initialized. Zero initialization makes every added span condition zero initially. Zero anchoring keeps it zero specifically at $g = 0$ throughout training. These properties preserve the base condition at the stated boundary; they do not freeze the trainable backbone.

The scale and zero reference play distinct roles. Removing the time multiplier reduces rapid phase variation with the requested span. Subtracting $\phi(0)$ fixes the boundary contribution without changing pairwise distances between span features. For frequencies ω, the resulting correction obeys $\|W^{(a)}_g \Delta\phi(g)\|_2 \le |g| \|W^{(a)}_g\|_2 \|\omega\|_2$. It therefore vanishes continuously near zero span. Under a local Lipschitz assumption on the vector predictor with respect to its conditions, its effect on the solution map is bounded by a term of order g^2 relative to suppressing the added branches at the same parameters. Appendix B.3 proves these statements and specifies their scope.

This construction connects finite-span conditioning to the network's local-velocity request without adding backbone layers. The identity $F_\theta(x_t, t, t) = x_t$ still follows from the map parameterization, independently of the encoding. Finite-span accuracy is evaluated empirically in Section 4.5, which tests the combined scale, frequency, and zero-reference encoding change.

3.3 Endpoint Anchoring and Fresh-Noise Replay

Span conditioning specifies the prediction request. We next supervise the clean-endpoint request used by the sampler, first on analytically noised data and then on inputs reconstructed from the model's own predictions.

Direct endpoint anchoring. Let $d_m(a, b)$ be the mean squared error of $P_m(a - b)$ over valid coordinate components. Following the adaptive weighting used in solution-function training (Luo et al., 2026), define

$$\ell_{\text{ep}}(\widehat{x}, x_0) = \frac{d_m(\widehat{x}, x_0)}{[\text{sg}(d_m(\widehat{x}, x_0)) + \varepsilon_{\text{ep}}]^{p_{\text{ep}}}}, \tag{3}$$

where sg denotes stop-gradient, $\varepsilon_{\text{ep}} > 0$ stabilizes the weight, and $p_{\text{ep}} > 0$ controls its error dependence. Direct anchoring supervises the endpoint prediction against the paired clean backbone:

$$\begin{aligned} \widehat{x}_0^{\text{dir}} &= F_\theta(x_t, t, 0), \\ \mathcal{L}_{\text{anchor}} &= \mathbb{E}\big[\ell_{\text{ep}}(\widehat{x}_0^{\text{dir}}, x_0)\big]\,. \end{aligned} \tag{4}$$

Whereas base consistency couples two model predictions, anchoring gives the clean-endpoint request a direct paired-data target and complements local velocity supervision.

From endpoint estimates to replay inputs. During sampling, a predicted endpoint supplies the structural component of the next noisy input. To train on this construction, choose a nonterminal time $0 < r < t \le 1$ and draw fresh projected noise $\xi' = P_m z'$ independently of the noise used for x_t. Define

$$\begin{aligned} x_r^{\text{rep}} &= (1-r)\,\text{sg}(\widehat{x}_0^{\text{dir}}) + r\xi', \\ x_r^{\text{ana}} &= (1-r)x_0 + r\xi'. \end{aligned} \tag{5}$$

The analytic reference uses the same r and ξ' solely to identify the change in the input; it requires no extra network evaluation. Independent noise varies the noisy context while retaining the predicted structural component and the same paired clean target. Their difference is

$$x_r^{\text{rep}} - x_r^{\text{ana}} = (1-r)\big[\text{sg}(\widehat{x}_0^{\text{dir}}) - x_0\big]. \tag{6}$$

Thus, the first prediction's residual enters the next input as a model-dependent structural perturbation. Fresh-noise replay trains endpoint prediction on this input using the original clean backbone as its target:

$$\mathcal{L}_{\text{replay}} = \mathbb{E}[\ell_{\text{ep}}(F_\theta(x_r^{\text{rep}}, r, 0), x_0)]\,. \tag{7}$$

Both predictions share parameters θ. Stop-gradient blocks the replay loss through the first prediction when constructing the input; the second evaluation updates the network. Anchoring constrains the paired endpoint residual, while replay supervises recovery from inputs containing that residual. Replay therefore extends training to one-step model-induced states. Appendix B.5 gives the corresponding output-residual decomposition and time-pair construction.

3.4 Training Objective and Few-Step Sampling

Training objective. We retain SoFlow's flow-matching (FM) supervision of local velocity and solution-consistency (SCM) supervision comparing predictions from nearby analytic states to a shared target time (Luo et al., 2026). The implementation uses a linear SCM surrogate, whose relation to the weighted quadratic form is derived in Appendix C. The total differentiated objective is

$$\mathcal{L} = \rho\mathcal{L}_{\text{FM}} + (1-\rho)\Big[\mathcal{L}_{\text{SCM}}^{\text{surr}} + \tilde{\lambda}_a\mathcal{L}_{\text{anchor}} + \tilde{\lambda}_r\mathcal{L}_{\text{replay}}\Big]. \tag{8}$$

Here ρ is the minibatch fraction assigned to FM, and $\tilde{\lambda}_a, \tilde{\lambda}_r$ weight the endpoint terms within the consistency branch. Both endpoint losses average over the same selected subset of that branch; expectations include structures, noises, and sampled times. These auxiliary predictions add training computation but do not alter the sampler.

Few-step sampling. For K endpoint evaluations, set $t_k = 1 - k/K$ and initialize $x_{t_0} = P_m z_{\text{init}}$ from standard Gaussian noise. Following solution-function sampling (Luo et al., 2026), alternate endpoint prediction and re-noising:

$$\begin{aligned} \widehat{x}_0^{(k)} &= F_\theta(x_{t_k}, t_k, 0), && k = 0, \dots, K-1, \\ x_{t_{k+1}} &= (1-t_{k+1})\widehat{x}_0^{(k)} + t_{k+1}\eta_k, && k = 0, \dots, K-2. \end{aligned} \tag{9}$$

Here $\eta_k = P_m z_k$ uses independent Gaussian draws. Both terms in the re-noising update already lie in the masked, centered subspace, so the outer projection is omitted; Appendix E gives the full procedure. Every request has $s = 0$ and span $g_k = t_k$, rather than the grid interval $t_k - t_{k+1}$. Return $\widehat{x}_0^{(K-1)}$ directly, without terminal re-noising or a call at $t = 0$. With one network evaluation per endpoint prediction and no additional inference branches, this procedure uses exactly K NFEs.

4 EXPERIMENTS

4.1 EVALUATION OVERVIEW

We evaluate unconditional backbone generation in terms of designability, structural diversity, and sampling cost. We assess designability with ProteinMPNN and ESMFold (Dauparas et al., 2022; Lin et al., 2023): a backbone passes if a designed sequence folds back to within 2 Å scRMSD of it. We generate and evaluate all baseline samples ourselves using this shared pipeline. Each comparison specifies its length range, sample count, and relevant metrics; detailed evaluation settings are in Appendix D.

4.2 COMPARISON WITH FEW-STEP MODELS

We compare FARFlow with QFlow (Yue et al., 2025) and RMF (Woo et al., 2026) at 20, 50, and 100 network function evaluations (NFE). For each setting, we generate ten backbones at every integer length from 60 to 128, giving 690 samples. Table 1 evaluates designability and structural self-consistency (scRMSD/scTM), together with within-set similarity (Pairwise TM) to assess diversity. We also evaluate the length dependence at 150, 200, and 250 residues, with 50 samples per model–NFE setting and length; the evaluated NFEs are shown in Figure 3a.

Table 1: **Comparison with few-step models on lengths 60–128.** Each model–budget setting contains 690 backbones generated and evaluated by us. scRMSD and scTM are means ± standard deviations. Lower Pairwise TM indicates greater diversity. FARFlow uses one fixed checkpoint across budgets. Bold denotes the best value within each NFE block.

NFE	Method	Designability ↑	scRMSD (Å) ↓	scTM ↑	Pairwise TM ↓
20	**FARFlow**	**0.816**	1.659 ± 1.557	**0.890 ± 0.097**	0.352
	QFlow	0.778	1.703 ± 1.658	0.853 ± 0.098	0.377
	RMF	0.806	**1.554 ± 1.438**	0.877 ± 0.091	**0.345**
50	**FARFlow**	**0.909**	**1.115 ± 1.023**	**0.933 ± 0.067**	0.363
	QFlow	0.870	1.432 ± 1.218	0.882 ± 0.080	0.379
	RMF	0.825	1.446 ± 1.189	0.885 ± 0.081	**0.344**
100	**FARFlow**	**0.958**	**0.933 ± 0.608**	**0.949 ± 0.054**	0.385
	QFlow	0.885	1.319 ± 1.018	0.889 ± 0.077	0.393
	RMF	0.832	1.425 ± 1.145	0.885 ± 0.081	**0.343**

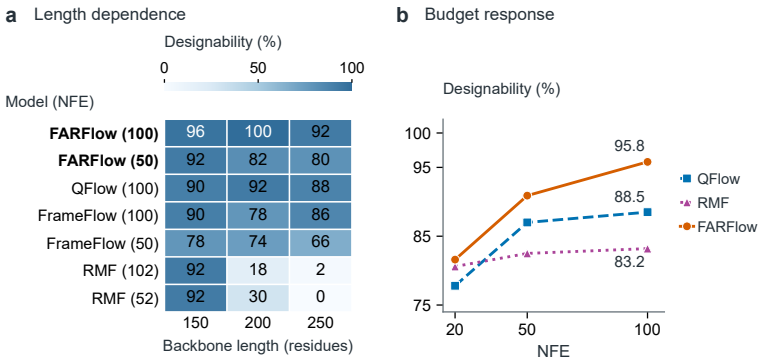


Figure 3: **Few-step quality across lengths and budgets.** (a) Designability at each length, with 50 backbones per cell. Row labels give the model and NFE. (b) The 20/50/100-NFE comparison in Table 1, with 690 backbones per setting. The panels use separate sample sets.

FARFlow maintains good designability as the sampling budget increases (Figure 3b), accompanied by improved scRMSD and scTM (Table 1). RMF produces less redundant samples by Pairwise TM, highlighting a quality–diversity trade-off. Figure 3a extends the comparison to longer backbones: FARFlow at 100 NFE maintains high designability, whereas the 50-NFE setting is more sensitive to length.

4.3 Standard Models and Generated Structures

We next compare FARFlow with standard flow and diffusion generators at lengths 50, 100, 150, 200, and 250, with 50 generated backbones per length. Table 2 reports designability, structural self-consistency, diversity, and AFDB novelty at each model's evaluation budget.

Table 2: **Comparison with standard backbone generators.** We generate and evaluate 50 backbones at each of five lengths (50–250 residues). Designability is a percentage; lower AFDB maxTM indicates greater novelty. Bold marks the best quality value in each column; NFE varies by model.

Method	NFE	Design. (%) ↑	Med. scRMSD (Å) ↓	scTM ↑	Pairwise TM ↓	AFDB maxTM ↓
FrameDiff	500	58.0	1.719	0.840	0.389	0.700
FoldFlow-Base	500	30.4	3.387	0.728	0.384	0.685
FoldFlow-OT	500	34.0	3.047	0.749	0.372	0.695
FoldFlow-SFM	500	44.0	2.297	0.790	0.395	0.681
FrameFlow	100	88.8	0.975	0.933	0.362	0.642
RFdiffusion	50	94.4	**0.709**	0.954	0.423	0.664
Proteína-60M	400	92.0	0.795	0.946	0.368	0.697
Genie2	1000	94.4	0.712	**0.958**	0.379	**0.600**
FARFlow-50	50	84.4	1.076	0.929	**0.360**	0.712
FARFlow-100	100	**94.8**	0.872	0.947	0.375	0.735

FARFlow-100 attains designability comparable to high-quality standard generators while using fewer evaluations than several high-budget baselines. The other metrics identify its current trade-offs: RFdiffusion and Genie2 retain advantages in some self-consistency measures, and FARFlow's larger AFDB similarity indicates less novel samples by this metric. FARFlow therefore offers high designability at moderate evaluation budgets, with reference-database novelty remaining a limitation. Within-set diversity and novelty relative to a reference database describe different properties: less redundant samples are not necessarily less similar to known structures.

Figure 4 complements this evaluation with selected FARFlow and Proteína generations at matched budgets of 20, 50, and 100 NFE. Figure 1 provides additional FARFlow examples.

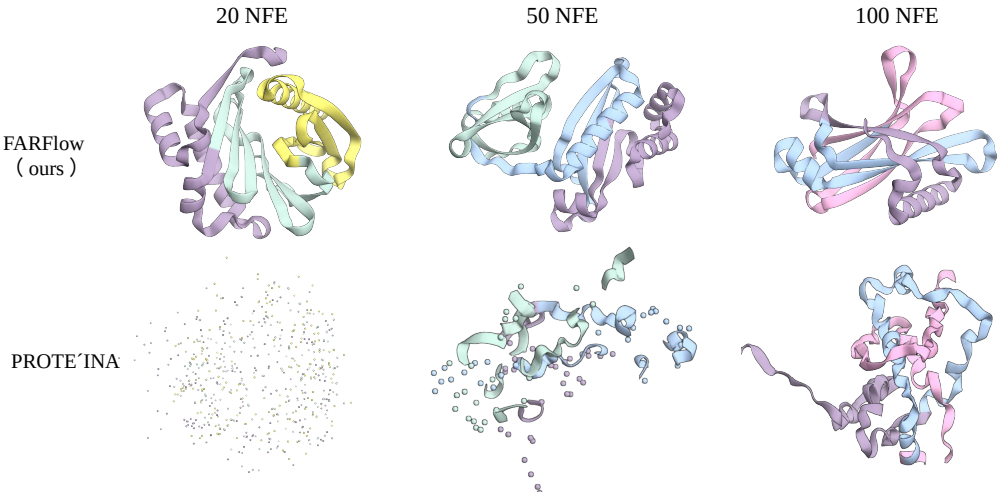


Figure 4: **Selected generations at matched sampling budgets.** FARFlow (top) and Proteína (bottom) at 20, 50, and 100 NFE. These examples provide qualitative context and are not paired comparisons; the aggregate standard-model evaluation uses the budgets in Table 2.

4.4 Sampling Efficiency

We next assess sampling cost on the 60–128-residue cohort used for the few-step comparison (Section 4.2). We compare mean backbone-generation time with designability. Figure 5 also reports D/T, where D is the designable fraction and T is generation time per backbone. This measures the rate of generating designable backbones, counting sampling time only; sequence design and folding are excluded.

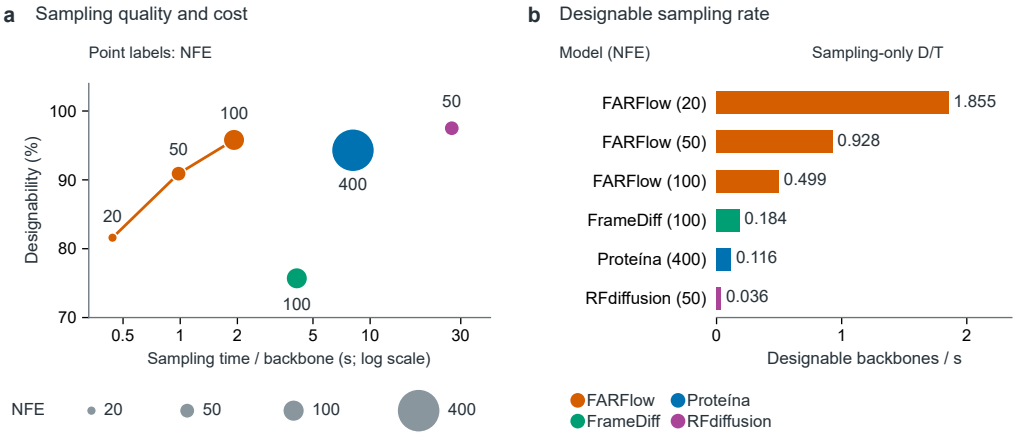


Figure 5: **Sampling cost and useful output.** (a) Designability versus generation time; labels and marker areas indicate NFE, and colors identify methods. (b) Designable backbones per sampling second, D/T. Each working point is labelled with its NFE. Timing excludes sequence design and folding; details are in Appendix D.

The 20-NFE setting favors throughput, while 50 and 100 NFE improve designability at higher sampling cost. These choices support different screening budgets. Across architectures, NFE alone does not determine elapsed time.

4.5 Ablation Study

Table 3 shows the contribution of the span representation, endpoint anchoring, and replay at 20 NFE.

Table 3: **Ablation study at 20 NFE.** Lower scRMSD indicates better sequence–structure self-consistency. Bold marks the best values within A–B and within C–E.

Variant	Span encoding	Anchor	Second input	NFE	Design. (%) ↑	scRMSD (Å) ↓
A	Legacy	–	–	20	34.0	2.72
B	Zero-anchored	–	–	20	**55.3**	**1.69**
C	Zero-anchored	✓	–	20	75.5	1.20
D	Zero-anchored	✓	Replay	20	**79.2**	**1.11**
E	Zero-anchored	✓	Analytic	20	76.4	1.17

A→B gives the largest gain (+21.3 designability points, -1.03 Å scRMSD), making the revised zero-anchored span representation the largest contributor in this staged ablation. B→C adds +20.2 points and -0.49 Å, confirming direct endpoint anchoring as another major contributor. D gives the best final result (79.2%, 1.11 Å) and outperforms both C (75.5%, 1.20 Å) and the compute-matched analytic control E (76.4%, 1.17 Å), supporting model-induced replay inputs as an additional useful mechanism. Small variations may arise from stochastic training and sampling.

5 Conclusion and Limitations

FARFlow studies the representation and supervision of finite-span requests for protein backbone generation. Its unscaled, zero-anchored conditioning separates prediction span from source-time conditioning while preserving the added branch's zero-span boundary. Direct endpoint anchoring supplies paired clean targets, and fresh-noise replay extends this supervision to inputs containing the model's structural estimates. The method retains the underlying coordinate Transformer and endpoint re-noising sampler.

Experiments show high designability across the evaluated budgets and lengths. In the staged ablation, the zero-anchored span representation gives the largest single gain, direct endpoint anchoring provides another major improvement, and replay yields the best final configuration at 20 NFE. Sampling-time measurements characterize the practical quality–cost trade-off.

FARFlow currently generates unconditional monomeric C_α backbones. Reference-database novelty remains a limitation, and computational designability does not establish experimental function or stability. Replay adds training computation and currently covers one class of single-transition model-induced inputs; extending training to multi-step rollout states is a natural next direction. A priority is also to reduce the budget below the evaluated 20–100-NFE range, ultimately toward one-step generation, where no later endpoint prediction can refine the initial output. Other directions include conditional design and joint sequence–structure generation.

AI Use Statement

Generative AI tools assisted with literature retrieval, discussions of research framing and method design, mathematical derivations and formula checking, code inspection, interpretation of results, English drafting, Chinese translation, figure editing, and LaTeX preparation. The authors are responsible for checking AI-assisted content against the implementation, experimental records, and cited sources, and retain responsibility for the final methodology, evidence, and claims.

Ethics Statement

This work studies computational generation of unconditional protein backbone coordinates. Sequence–structure self-consistency is a computational screening measure and does not establish biological function, experimental stability, or safety. Protein-design methods can support beneficial research and have potential dual-use applications. Downstream synthesis or functional deployment requires appropriate biosafety review and validation beyond the evaluations reported here.

Reproducibility Statement

Section 3 specifies the model parameterization and objectives. Appendices B and C detail coordinate conventions, span properties, endpoint gradient paths, and the SCM surrogate. Appendix E gives training and sampling procedures, and Appendix D describes evaluation cohorts and metrics. The source package includes the numerical inputs and plotting code for Figures 3 and 5. Appendix D.4 details the component-comparison protocol.

References

Joey Bose, Tara Akhound-Sadegh, Guillaume Huguet, Kilian Fatras, Jarrid Rector-Brooks, Cheng-hao Liu, Andrei Nica, Maksym Korablyov, Michael Bronstein, and Alexander Tong. SE(3)-Stochastic Flow Matching for Protein Backbone Generation. In *International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/618c95f4557c15b253fb0e6f548ea0c0-Abstract-Conference.html.

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

Kevin Frans, Danijar Hafner, Sergey Levine, and Pieter Abbeel. One Step Diffusion via Shortcut Models. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/559a0998fab1d19b80e7e43a5852401c-Abstract-Conference.html.

Tomas Geffner, Kieran Didi, Zuobai Zhang, Danny Reidenbach, Zhonglin Cao, Jason Yim, Mario Geiger, Christian Dallago, Emine Kucukbenli, Arash Vahdat, and Karsten Kreis. Proteina: Scaling Flow-based Protein Structure Generative Models. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/f4e9121ad30cd4e5528042fbfd835b3f-Abstract-Conference.html.

Tomas Geffner, Kieran Didi, Zhonglin Cao, Danny Reidenbach, Zuobai Zhang, Christian Dallago, Emine Kucukbenli, Karsten Kreis, and Arash Vahdat. La-Proteina: Atomistic Protein Generation via Partially Latent Flow Matching. In *International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/6e1734c47c0cc899021060d88f69dc65-Abstract-Conference.html.

Zhengyang Geng, Mingyang Deng, Xingjian Bai, J. Zico Kolter, and Kaiming He. Mean Flows for One-step Generative Modeling. In *Advances in Neural Information Processing Systems*, volume 38, 2025. doi: 10.52202/085713-2534. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/6d13e085b79d454da5910e4ca82a3d9d-Abstract-Conference.html.

Yeqing Lin and Mohammed AlQuraishi. Generating Novel, Designable, and Diverse Protein Structures by Equivariantly Diffusing Oriented Residue Clouds. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 20978–21002. PMLR, 2023. URL https://proceedings.mlr.press/v202/lin23a.html.

Yeqing Lin, Minji Lee, Zhao Zhang, and Mohammed AlQuraishi. Out of Many, One: Designing and Scaffolding Proteins at the Scale of the Structural Universe with Genie 2. *arXiv preprint arXiv:2405.15489*, 2024. URL https://arxiv.org/abs/2405.15489.

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

Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow Straight and Fast: Learning to Generate and Transfer Data with Rectified Flow. In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=XVjTT1nw5z.

Tianze Luo, Haotian Yuan, and Zhuang Liu. SoFlow: Solution Flow Models for One-Step Generative Modeling. In *International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/a654298de44ec95dec28aa37e8d0af51-Abstract-Conference.html.

Tim Salimans and Jonathan Ho. Progressive Distillation for Fast Sampling of Diffusion Models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=TIdIXIpzhoI.

Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency Models. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 32211–32252. PMLR, 2023. URL https://proceedings.mlr.press/v202/song23a.html.

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

Dongyeop Woo, Marta Skreta, Seonghyun Park, Sungsoo Ahn, and Kirill Neklyudov. Riemannian MeanFlow. *arXiv preprint arXiv:2602.07744*, 2026. URL https://arxiv.org/abs/2602.07744.

Liyang Xie, Haoran Zhang, Zhendong Wang, Wesley Tansey, and Mingyuan Zhou. SPID: Distilled Protein Backbone Generation. *arXiv preprint arXiv:2510.03095*, 2025. URL https://arxiv.org/abs/2510.03095.

Jason Yim, Andrew Campbell, Andrew Y. K. Foong, Michael Gastegger, José Jiménez-Luna, Sarah Lewis, Victor Garcia Satorras, Bastiaan S. Veeling, Regina Barzilay, Tommi Jaakkola, and Frank Noé. Fast protein backbone generation with SE(3) flow matching. *arXiv preprint arXiv:2310.05297*, 2023a. URL https://arxiv.org/abs/2310.05297.

Jason Yim, Brian L. Trippe, Valentin De Bortoli, Emile Mathieu, Arnaud Doucet, Regina Barzilay, and Tommi Jaakkola. SE(3) diffusion model with application to protein backbone generation. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 40001–40039. PMLR, 2023b. URL https://proceedings.mlr.press/v202/yim23a.html.

Angxiao Yue, Zichong Wang, and Hongteng Xu. ReQFlow: Rectified Quaternion Flow for Efficient and High-Quality Protein Backbone Generation. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 73828–73850. PMLR, 2025. URL https://proceedings.mlr.press/v267/yue25c.html.

A Structural Diagnostics

Secondary-structure composition and simple local-geometry statistics are reported here as diagnostics rather than ranking metrics. Figure 6 shows that FARFlow retains nontrivial helix, sheet, and coil content at both 50 and 100 NFEs. In the five-length evaluation, the mean adjacent-C_α distance is approximately 3.83 Å and the measured nonlocal clash rate is near zero. Because the benchmark does not include a natural-reference secondary-structure distribution evaluated with the identical pipeline, we use these composition statistics as diagnostics rather than ranking criteria.

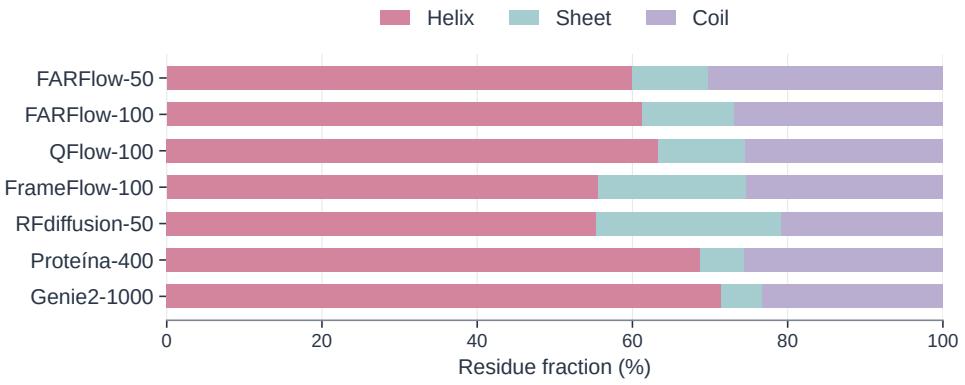


Figure 6: **Secondary-structure composition on the five-length panel.** Fractions of helix, sheet, and coil residues for representative generators. Model-label suffixes specify NFE. The figure is used as a structural diagnostic rather than a ranking criterion.

B Coordinate Conventions and Endpoint Training

B.1 Masked and Centered Coordinates

For a backbone stored in N residue positions, a fixed mask $m \in \{0,1\}^N$ selects $n = \sum_i m_i > 0$ valid residues. Define

$$\begin{aligned}(P_m y)_i &= m_i \left(y_i - \frac{1}{n} \sum_{j=1}^{N} m_j y_j \right), \\ d_m(a,b) &= \frac{1}{3n} \sum_{i=1}^{N} m_i \left\| [P_m(a-b)]_i \right\|_2^2 .\end{aligned} \tag{B.1}$$

Preprocessing centers the valid coordinates and sets padded positions to zero. Training coordinates are expressed in nanometers, with unit standard deviation for the unprojected Gaussian prior in those units; reported structural distances are converted to angstroms. The operator is an idempotent projection, so $P_m x_0 = x_0$ and $P_m^2 = P_m$. All Gaussian arrays are projected in the same way. The resulting noise is Gaussian in the masked, centered subspace; its residue coordinates are not independent after projection. The mask is fixed per structure and suppressed in F_θ's arguments.

The linear conditional path in Eq. (1) has velocity $\dot{x}_t = \xi - x_0$. Consequently, an analytic state at $l < t$ from the same data–noise pair satisfies $x_l = x_t + (l-t)(\xi - x_0) = (1-l)x_0 + l\xi$. The local predictor is $v_\theta(x_t, t) = P_m u_\theta(x_t, t, t)$. Both v_θ and the finite-span predictor share the same network parameters.

Projection and prior covariance. The matrix acting on each Cartesian coordinate is

$$Q_m = \operatorname{diag}(m) - \frac{m m^\top}{n}, \qquad P_m y = Q_m y, \qquad Q_m^\top = Q_m, \quad Q_m^2 = Q_m. \tag{B.2}$$

To verify idempotence, use $\operatorname{diag}(m)m = m$ and $m^\top m = n$ when expanding Q_m^2. Thus, P_m is the orthogonal projection onto the subspace with zero padded entries and zero valid-residue centroid. If z has independent standard Gaussian entries and $\xi = Q_m z$, then

$$\mathbb{E}[\xi_i] = 0, \qquad \operatorname{Cov}(\xi_i, \xi_j) = \left(m_i \mathbf{1}_{i=j} - \frac{m_i m_j}{n} \right) I_3. \tag{B.3}$$

This follows from $Q_m Q_m^\top = Q_m$. The centered coordinate subspace has dimension $3(n-1)$. Projecting again leaves an already centered prediction unchanged. This projection removes translation and padded coordinates; it does not enforce bond geometry or exclude steric clashes. Likewise, an analytic noisy state is a training input, not necessarily a physically realizable protein conformation.

B.2 Finite-Span Map and Local Limit

For a fixed centered input $P_m x = x$ and $t > 0$, the map satisfies

$$\begin{aligned} F_\theta(x,t,t) &= x, \\ \lim_{g \downarrow 0} \frac{x - F_\theta(x,t,t-g)}{g} &= P_m u_\theta(x,t,t) = v_\theta(x,t). \end{aligned} \tag{B.4}$$

The first identity follows by setting $g = 0$. For the second, divide $x - F_\theta(x,t,t-g) = g P_m u_\theta(x,t,t-g)$ by g and assume continuity in the target time at $s = t$. The local limit explains why FM can supervise the zero-span vector output even though the map itself is the identity there. At nonzero span, u_θ is conditioned on the requested target time and represents the corresponding finite-span request; its accuracy is evaluated empirically.

B.3 CONDITIONING PATHWAYS

We retain Proteína's coordinate Transformer (Geffner et al., 2025). The sequence condition passes through the existing conditioning layers before modulating the Transformer blocks through adaptive normalization and output scaling. The pair condition modulates the adaptive normalization used to construct the pair representation. Each pathway receives its own additive projection of $\Delta\phi(g)$ as in Eq. (2). Both projections are bias-free and zero-initialized. The coordinate-embedding pathway is unchanged, and no additional pair-update module is introduced.

For an embedding with $2h$ channels, $h > 1$, the unscaled feature uses

$$\begin{aligned} \omega_i &= B^{-i/(h-1)}, \qquad i = 0, \dots, h-1, \\ \phi(g) &= \big[\sin(g\omega_0), \dots, \sin(g\omega_{h-1}), \cos(g\omega_0), \dots, \cos(g\omega_{h-1})\big]. \end{aligned} \tag{B.5}$$

The zero-anchored encoding uses $B = 10000$ and subtracts $\phi(0)$ before projection. The legacy encoding instead multiplies the span by 2000, uses frequency base 2000, and retains its nonzero value at the origin. Thus, the A–B comparison changes the span scale, frequency base, and zero reference together; it evaluates the combined representation rather than isolating the individual effects. Source-time conditioning retains its original embedding. An odd embedding dimension is padded with one zero channel.

Behavior near zero span. Each frequency contributes $[\sin(g\omega_i), \cos(g\omega_i) - 1]$. The squared norm of this pair is $4\sin^2(g\omega_i/2)$, hence

$$\begin{aligned} \|\Delta\phi(g)\|_2^2 &= 4\sum_{i=0}^{h-1} \sin^2\Big(\frac{g\omega_i}{2}\Big) \le g^2 \sum_{i=0}^{h-1} \omega_i^2, \\ \|W_g^{(a)}\Delta\phi(g)\|_2 &\le |g|\, \|W_g^{(a)}\|_2 \left(\sum_{i=0}^{h-1} \omega_i^2\right)^{1/2}. \end{aligned} \tag{B.6}$$

Here $\|W_g^{(a)}\|_2$ is the spectral norm. The added condition therefore vanishes continuously as $g \to 0$ for fixed finite projection weights. Its sine channels vary to first order in g, while its shifted cosine channels vary to second order. Equation (B.6) characterizes the conditioning input near zero span, while coordinate-level accuracy is evaluated empirically. Zero initialization makes the added condition zero at every span initially while leaving the base network trainable.

Interval sensitivity and the role of zero anchoring. Let $\omega = (\omega_0, \dots, \omega_{h-1})$. For any two spans, trigonometric identities give

$$\begin{aligned} \|\Delta\phi(g_1) - \Delta\phi(g_2)\|_2^2 &= \|\phi(g_1) - \phi(g_2)\|_2^2 \\ &= 4\sum_{i=0}^{h-1} \sin^2\left(\frac{(g_1 - g_2)\omega_i}{2}\right) \\ &\le (g_1 - g_2)^2 \|\omega\|_2^2. \end{aligned} \tag{B.7}$$

The equality shows that subtracting the origin feature does not itself smooth pairwise feature distances. It sets the added condition to zero at the boundary. The derivative has constant norm $\|\phi'(g)\|_2 = \|\omega\|_2$. Multiplying the input by a scalar κ instead gives derivative norm $|\kappa|\|\omega\|_2$ for the same frequency set. Hence, scale controls feature sensitivity, whereas anchoring controls the boundary offset. Because the A–B encoding change also modifies the frequency base, it evaluates the complete revised span representation rather than a scale-only change. These are deterministic properties of the encoding.

Local effect on the solution map. Stack the sequence and pair conditions into c, and their projection matrices into W, so $c(g) = c_{\text{base}} + W\Delta\phi(g)$. Fix the input, source time, mask, and network parameters. Let F_θ^{base} denote the same evaluation with only the added span corrections suppressed; it is not a separately trained baseline. Assume that, for sufficiently small g, the vector predictor is

Lipschitz in these conditions with a common finite constant L_c over the compared conditions. Using the Euclidean norm for stacked conditions and the Frobenius norm for coordinate outputs gives

$$\begin{aligned}\|F_\theta(x,t,t-g) - F_\theta^{\text{base}}(x,t,t-g)\|_F &\le |g| L_c \|W\Delta\phi(g)\|_2 \\ &\le g^2 L_c \|W\|_2 \|\omega\|_2.\end{aligned} \tag{B.8}$$

For the first inequality, subtract the two maps: their shared x cancels, and the remaining vector-output difference is multiplied by g. The orthogonal coordinate projection has operator norm at most one. Apply the assumed condition-space Lipschitz bound, then Eq. (B.6), to obtain the second inequality. Thus, under the stated assumption, the added span branch changes the map only to second order near $g = 0$. This is a local conditioning bound at fixed learned parameters; finite-span accuracy, including the largest inference requests such as $g = 1$, is evaluated empirically.

B.4 Adaptive Endpoint Loss and Its Gradient

For one structure, write the endpoint loss as $\ell_{\text{ep}} = we$. Because P_m is symmetric and idempotent,

$$\begin{aligned} e &= \frac{\|P_m(\widehat{x} - x_0)\|_F^2}{3n}, \qquad & w &= [\text{sg}(e) + \varepsilon_{\text{ep}}]^{-p_{\text{ep}}}, \\ \nabla_{\widehat{x}}\ell_{\text{ep}} &= w\nabla_{\widehat{x}} e = \frac{2w}{3n} P_m(\widehat{x} - x_0). \end{aligned} \tag{B.9}$$

The denominator is evaluated from the current residual, but is held fixed during this backward pass. Differentiating the ordinary scalar function $e/(e+\varepsilon_{\text{ep}})^{p_{\text{ep}}}$ would produce a different gradient. Thus, stop-gradient implements adaptive rescaling of the squared-error gradient. The factor $3n$ averages coordinate components within each structure; the outer expectation averages structures, rather than pooling all residues across different lengths.

B.5 Replay Construction and Residual Analysis

Write $\delta_t = \text{sg}(\widehat{x}_0^{\text{dir}}) - x_0$. The analytic reference and replay state in Eq. (5) use the same time r and fresh noise ξ', so their difference is exactly $(1-r)\delta_t$. For a fixed predictor, the resulting endpoint residual decomposes as

$$\begin{aligned}\widehat{x}_0^{\text{rep}} - x_0 = F_\theta(x_r^{\text{ana}}, r, 0) - x_0 \\ + \left[F_\theta(x_r^{\text{ana}} + (1-r)\delta_t, r, 0) - F_\theta(x_r^{\text{ana}}, r, 0)\right],\end{aligned} \tag{B.10}$$

where $\widehat{x}_0^{\text{rep}} = F_\theta(x_r^{\text{rep}}, r, 0)$. The first term is the residual on the analytic reference; the second is the prediction change caused by substituting the estimated structural component. This algebraic identity isolates the change associated with the replay input construction. The analytic reference is used for analysis only, while training applies the single replay loss in Eq. (7).

For endpoint training, a uniform reference grid with budget $M \ge 2$ and index $j \in \{0, \dots, M-2\}$ supplies $t = 1 - j/M$ and $r = 1 - (j+1)/M$. These adjacent nonterminal points reproduce one endpoint prediction–re-noising transition; the prediction itself still requests the clean endpoint. The reference-budget distribution and endpoint subset size are training settings, distinct from the inference budget K.

Direct anchoring and replay use the same selected subset $\mathcal{B}_{\text{ep}}$ of the consistency minibatch. Each selected structure receives both endpoint evaluations, and each loss is averaged over $|\mathcal{B}_{\text{ep}}|$. These evaluations are additional to the base consistency computation. The first endpoint remains differentiable in the anchor loss but is detached when used to construct x_r^{rep}. The second endpoint is supervised against the original x_0, not against the first prediction. Accordingly, the endpoint terms are auxiliary paired-data objectives that directly train the clean-endpoint request on analytic and model-induced inputs.

Shared noise in the analysis. The fresh noise is independent of the noise used for the first analytic input, but is shared between x_r^{rep} and its analytic reference. If the reference instead used another independent draw ξ'', their difference would also contain $r(\xi' - \xi'')$. Sharing the reference noise isolates the predicted structural residual without attributing a noise difference to replay. This analytical coupling does not require evaluating the network on the reference input during replay training.

Gradient paths. Let coordinate arrays be vectorized when taking Jacobians. With the sampled times, noises, and target fixed, the stopped replay input gives

$$\begin{aligned} \frac{\partial x_r^{\mathrm{rep}}}{\partial \theta} &= 0 \quad \text{(autodiff)}, \\ \nabla_\theta \ell_{\mathrm{ep}}(\widehat{x}_0^{\mathrm{rep}}, x_0) &= \left(\partial_\theta F_\theta(x, r, 0)|_{x=x_r^{\mathrm{rep}}}\right)^\top \nabla_{\widehat{x}} \ell_{\mathrm{ep}}(\widehat{x}_0^{\mathrm{rep}}, x_0). \end{aligned} \tag{B.11}$$

The zero in the first line describes the implemented backward graph. The numerical replay state still changes when a later update changes the predictor. The second call receives a parameter gradient with its input held fixed; the first call separately receives the direct-anchor gradient. Shared parameters do not create a gradient path through a detached input.

C SCM Implementation and Gradient Equivalence

C.1 Base Objectives

We use the projected local predictor $v_\theta(x_t, t) = P_m u_\theta(x_t, t, t)$ and coordinate error d_m defined in Appendix B.1. Local flow-matching regression has error $e_{\mathrm{FM}} = d_m(v_\theta(x_t, t), \xi - x_0)$. For consistency, choose $0 \le s < l < t \le 1$ and construct $x_l = x_t + (l - t)(\xi - x_0)$ from the same data–noise pair. The normalized consistency residual is

$$q_{t,l,s} = \frac{F_\theta(x_t, t, s) - \mathrm{sg}(F_\theta(x_l, l, s))}{t - l}. \tag{C.1}$$

Unlike replay, this comparison uses no independent re-noising. The flow-matching loss and a quadratic expression corresponding to the implemented SCM update are

$$\begin{aligned} \mathcal{L}_{\mathrm{FM}} &= \mathbb{E}\left[\frac{e_{\mathrm{FM}}}{[\mathrm{sg}(e_{\mathrm{FM}}) + \varepsilon_{\mathrm{FM}}]^{p_{\mathrm{FM}}}}\right], \\ \mathcal{L}_{\mathrm{SCM}}^{\mathrm{quad}} &= \mathbb{E}\left[\frac{t - l}{2(t - s)} \frac{d_m(q_{t,l,s}, 0)}{[\mathrm{sg}(d_m(q_{t,l,s}, 0)) + \varepsilon_{\mathrm{SCM}}]^{p_{\mathrm{SCM}}}}\right]. \end{aligned} \tag{C.2}$$

The adaptive denominators are held fixed during differentiation, with positive stabilizers ε and weighting powers p. The target prediction in Eq. (C.1) is stopped, while the current prediction remains differentiable.

C.2 Implemented Surrogate

For a sampled triple $s < l < t$, write $g = t - s$, $h = t - l$, and $U_\theta = P_m u_\theta(x_t, t, s)$. Define the normalized masked inner product $\langle a, b\rangle_m = (3n)^{-1} \sum_i m_i a_i^\top b_i$ and the adaptive denominator $A_{t,l,s} = [\mathrm{sg}(d_m(q_{t,l,s}, 0)) + \varepsilon_{\mathrm{SCM}}]^{p_{\mathrm{SCM}}}$. The implementation uses the linear surrogate

$$\mathcal{L}_{\mathrm{SCM}}^{\mathrm{surr}} = -\mathbb{E}\left[\frac{\langle U_\theta, \mathrm{sg}(q_{t,l,s})\rangle_m}{A_{t,l,s}}\right]. \tag{C.3}$$

For each update, the input states, sampled times, reference prediction, and adaptive weight are held fixed. Inputs and predictions lie in the masked, centered subspace, so $q_{t,l,s} = [x_t - gU_\theta - \mathrm{sg}(F_\theta(x_l, l, s))]/h$. Therefore $\mathrm{d}q_{t,l,s} = -(g/h)\mathrm{d}U_\theta$. Since $d_m(q, 0) = \langle q, q\rangle_m$, its differential is $2\langle q, \mathrm{d}q\rangle_m$. For a variation $\mathrm{d}U_\theta$, the per-sample quadratic expression in Eq. (C.2) satisfies

$$\begin{aligned} \mathrm{d}\ell_{\mathrm{SCM}}^{\mathrm{quad}} &= \frac{h}{g A_{t,l,s}} \left\langle q_{t,l,s}, -\frac{g}{h}\,\mathrm{d}U_\theta \right\rangle_m \\ &= -\frac{\langle q_{t,l,s}, \mathrm{d}U_\theta\rangle_m}{A_{t,l,s}} = \mathrm{d}\ell_{\mathrm{SCM}}^{\mathrm{surr}}. \end{aligned} \tag{C.4}$$

Thus, the two expressions yield the same gradient through the active prediction under the stated stop-gradient convention. The implementation uses this per-update gradient identity, with the target and adaptive weight recomputed on subsequent updates.

The temporal prefactor follows the weighted consistency construction in Eqs. (20)–(21) of Luo et al. (2026). Under the Euler parameterization, $|b(t,s)| = t-s$. The factor $1/2$ in our quadratic expression matches the gradient normalization of the implemented linear surrogate, rather than changing its update. The current prediction in the numerator of Eq. (C.2) must remain differentiable; only the target prediction and adaptive denominator are stopped. Numerical implementations also protect small time gaps before division. Positive squared-residual diagnostics and the signed surrogate have different numerical meanings and should not be interpreted interchangeably.

C.3 Branch Weights and Subset Reduction

Equation (8) averages the FM and consistency losses within their respective minibatch subsets, then weights them by ρ and $1-\rho$. The endpoint terms are means over the selected endpoint subset of the consistency branch. If those terms are instead written outside the consistency brackets, their coefficients must be

$$\lambda_a = (1-\rho)\tilde{\lambda}_a, \qquad \lambda_r = (1-\rho)\tilde{\lambda}_r. \tag{C.5}$$

Changing the position of a term without this rescaling changes the optimization objective. No extra endpoint-subset fraction is implicit in $\mathcal{L}_{\text{anchor}}$ or $\mathcal{L}_{\text{replay}}$: they are subset means, not zero-padded means over the full batch. This reduction gives each selected structure equal weight after its own coordinate normalization. Numerical coefficients, time distributions, and subset sizes are run settings rather than consequences of the gradient identities above.

D Evaluation Details

D.1 Sequence–Structure Self-Consistency

For each generated backbone, ProteinMPNN samples eight sequences at temperature 0.1, and ESMFold predicts their structures (Dauparas et al., 2022; Lin et al., 2023). A backbone is designable if its minimum C_α self-consistency RMSD across these sequences is below 2 Å. The eight-sequence search is part of evaluation; it does not rerank or replace generated backbones. scRMSD and scTM provide continuous measures of sequence–structure agreement. Table 1 reports means and standard deviations; Table 2 reports median scRMSD and mean scTM.

For J generated backbones, let scRMSD_{ij} be the aligned C_α RMSD for designed sequence j of backbone i. The backbone-level statistic is

$$a_i = \min_{1\le j\le 8} \mathrm{scRMSD}_{ij}, \qquad d_i = \mathbf{1}\{a_i < 2\,\text{Å}\}, \qquad D = \frac{1}{J}\sum_{i=1}^{J} d_i. \tag{D.1}$$

Thus, the denominator is the number of backbones, not eight times that number. Table 1 expresses D as a fraction, whereas Tables 2 and 3 report $100D$. For the main benchmark protocols, means and medians of minimum scRMSD summarize the J values a_i, including failed designs.

D.2 Sample Sets and Baseline Generation

The few-step comparison uses every integer length from 60 to 128, with ten generated backbones per length and 690 per model–budget setting. The standard-model comparison uses lengths 50, 100, 150, 200, and 250, with 50 samples per length and 250 per setting. The length-resolved comparison in Figure 3a uses 50 samples each at lengths 150, 200, and 250. These are separate evaluation cohorts. Within each comparison, models use the same length distribution and downstream evaluation pipeline.

We generate the baseline samples ourselves and compute their evaluation metrics, including those for QFlow (Yue et al., 2025) and Riemannian MeanFlow (RMF) (Woo et al., 2026). We use the baseline implementations and model weights released by their authors. One fixed FARFlow checkpoint is used across the main NFE comparisons. For each setting, all reported metrics refer to the same generated set. In the length-resolved panel, RMF's labels 52/102 count actual network calls for configurations with 50/100 nominal steps; Table 1 retains the separately evaluated 20/50/100 budget settings.

D.3 Diversity, Novelty, and Timing

Pairwise TM measures structural similarity within a generated set; lower values indicate less redundancy. AFDB maxTM is similarity to the nearest reference structure in the AlphaFold Database; lower values indicate greater novelty. These metrics complement designability and should be interpreted together with it.

Sampling time covers backbone generation and excludes ProteinMPNN and ESMFold. For designable fraction D and mean generation time T per backbone, D/T estimates the number of designable backbones generated per second under the measured configuration. It does not include the cost of discovering which backbones are designable. The timing comparison uses the dense 60–128 length range. FrameDiff uses 100 NFEs in the timing figure and 500 in the standard-model table, so these are distinct settings. NFE and elapsed time describe different aspects of cost because network architectures differ.

D.4 Component Comparisons

A and B in Table 3 compare the legacy and zero-anchored span encodings at 20 NFE, with 150 evaluated backbones per variant. The encoding change jointly modifies span scaling, the frequency base, and zero-reference subtraction, so A–B evaluates the combined representation change.

C, D, and E compare endpoint-training inputs at 20 NFE while using the zero-anchored encoding and direct anchoring. C uses no second endpoint input. D constructs the second input by re-noising a detached model prediction with fresh Gaussian noise. E constructs the second input by analytically noising the clean backbone and serves as the compute-matched control for the additional endpoint prediction. Because D and E keep the same second endpoint pass and differ in the structural source of that input, their comparison tests whether training on a model-induced replay state is more effective than an equally sized analytic-input pass. Table 3 reports designability and scRMSD for these configurations.

D.5 Interpreting Sampling Throughput

Let T_{gen} denote the measured total backbone-generation time for the evaluated set, so that

$$T = \frac{T_{\mathrm{gen}}}{J}, \qquad \frac{D}{T} = \frac{\sum_{i=1}^{J} d_i}{T_{\mathrm{gen}}}. \tag{D.2}$$

This identity uses the designable fraction, not its percentage representation, and measures useful output per backbone-generation time. The reported timing excludes downstream sequence design and folding. Cross-method wall-clock comparisons use compatible hardware, batching, precision, and timing boundaries, while NFE is reported separately as an architecture-dependent evaluation count.

E Training and Sampling Procedures

These procedures use Section 3's notation, with masking and centering applied per structure.

E.1 Endpoint-Supervised Training Update

1. Center and mask the clean batch. Split it into the FM and consistency subsets, and construct their analytic states and base losses as in Appendix C.
2. Select the endpoint subset $\mathcal{B}_{\mathrm{ep}}$ from the consistency branch. For each selected structure, draw a reference-grid pair (t, r) with $0 < r < t \leq 1$ as in Appendix B.5, and independently draw projected noise for its analytic input x_t.
3. Evaluate $\widehat{x}_0^{\mathrm{dir}} = F_\theta(x_t, t, 0)$ and average its adaptive loss against x_0 over $\mathcal{B}_{\mathrm{ep}}$ to obtain $\mathcal{L}_{\mathrm{anchor}}$.
4. Draw fresh projected noise ξ', form $x_r^{\mathrm{rep}} = (1-r)\,\mathrm{sg}(\widehat{x}_0^{\mathrm{dir}}) + r\xi'$, and evaluate $F_\theta(x_r^{\mathrm{rep}}, r, 0)$. Average the adaptive loss against the same x_0 to obtain $\mathcal{L}_{\mathrm{replay}}$.

5. Combine the branch means using Eq. (8), then backpropagate and update the shared parameters. Disabled auxiliary terms have coefficient zero and need not be evaluated.

The analytic control replaces the second input by $x_r^{\mathrm{ana}} = (1-r)x_0 + r\xi'$, retaining the second endpoint pass and paired target. Both full replay and this control add two endpoint evaluations per selected structure beyond base FM/SCM training. These training evaluations do not count toward inference NFE.

E.2 Endpoint Prediction and Re-Noising

1. Given an integer budget $K \geq 1$ and mask m, draw $z_{\mathrm{init}} \sim \mathcal{N}(0, I)$ and set $x_{t_0} = P_m z_{\mathrm{init}}$ with $t_0 = 1$.
2. For $k = 0, \ldots, K-1$, set $t_k = 1 - k/K$ and evaluate $\widehat{x}_0^{(k)} = F_\theta(x_{t_k}, t_k, 0)$.
3. If $k < K-1$, independently draw $z_k \sim \mathcal{N}(0, I)$, set $\eta_k = P_m z_k$, and construct $x_{t_{k+1}}$ by Eq. (9). A final projection may be applied to remove numerical centering drift; it is redundant in exact arithmetic.
4. After the last endpoint call, return $\widehat{x}_0^{(K-1)}$ directly.

The sampler makes K endpoint calls and $K-1$ re-noising updates, with no call at $t = 0$. Each prediction requests span t_k, not the grid interval $1/K$. For $K = 1$, it returns $F_\theta(P_m z_{\mathrm{init}}, 1, 0)$ directly. This defines the one-call limit; its generation quality remains unevaluated.